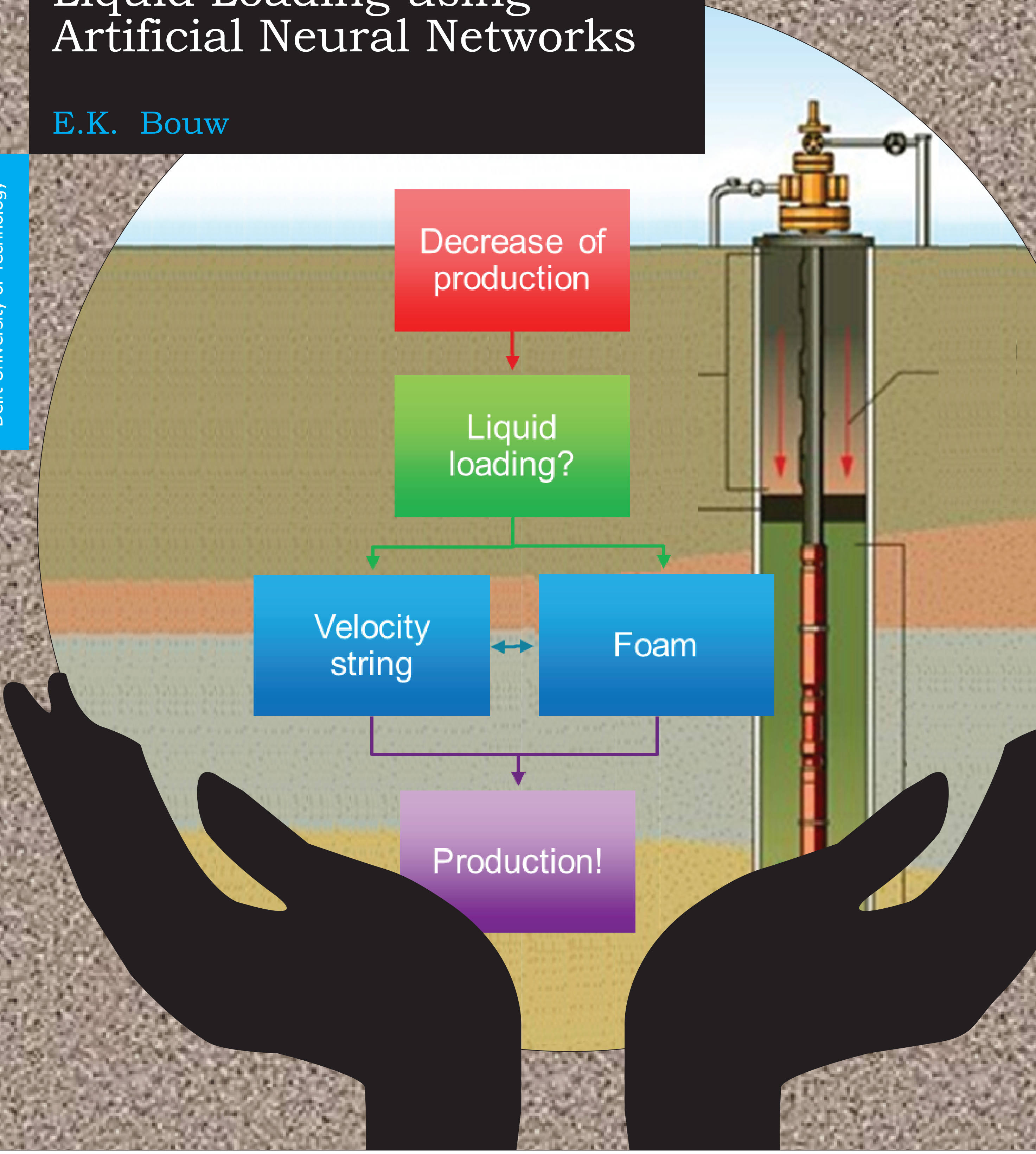
Analysis of End of Field Life Techniques and predicting Liquid Loading using Artificial Neural Networks

E.K. Bouw

Delft University of Technology



Analysis of End of Field Life Techniques and predicting Liquid Loading using Artificial Neural Networks

by

E.K. Bouw

in partial fulfillment of the requirements for the degree of

Master of Science
in Petroleum Engineering

at the Delft University of Technology,

Supervisor:	Dr. A. Barnhoorn,	TU Delft
Thesis committee:	Prof. Dr. P. L. J. Zitha,	TU Delft
	Dr. D. V. Voskov,	TU Delft
	F. Yavuz,	EBN
	R. Godderij,	EBN

TU Delft Delft University of Technology

Abstract

Liquid loading of a gas well is the inability of the produced gas to remove the produced liquids from the wellbore. It is one of the major issues that decreases flow production substantially or even stops flow completely for wells that are in the mature or tail-end production phase. To sustain gas production, the problem can be overcome with End of Field Life (EoFL) techniques. Of these techniques, velocity strings and foam injection are the two most popular ones in the Netherlands. To date, the availability of data on the potential of EoFL techniques is limited and this study aims to quantify the potential volume gain from these two EoFL techniques.

Moreover, it is difficult to determine when to install these techniques due to the prediction uncertainty of the liquid loading moment. A solution can be found in artificial intelligence. Using big data may predict future instability of production rates and may therefore be very useful in predicting the liquid loading moment in advance. In this thesis a first step is undertaken to use artificial intelligence, in particular artificial neural networks (ANN), to predict the onset of liquid loading applying actual field data.

In total, sixty-four liquid loading wells were examined in terms of production quantity. It can be concluded that the volume gain using a velocity string or foam injection has a very wide spectrum. About half of the wells produced 10 to 15 million Nm^3 more due to one of these techniques, while other wells even produced 100 million Nm^3 or more. The wide range is due to the dependency on well and environmental conditions. Next to this it may be noted that operators had expected a larger volume gain from these EoFL techniques.

Of the sixty-four wells, fifteen wells were examined in more detail, particularly in terms of economic gain. The techniques show a high success rate for over 70% of the wells with a NPV of some 20 million euros per well. The remaining wells showed a negative NPV, but only due to external factors, for example a leaking tubing. Therefore, EoFL techniques have shown to be valuable but more research should be undertaken to enhance knowledge on improving volume gains for wells in the mature or tail-end production phase in the Netherlands.

When predicting the onset of liquid loading, ANN forecasted future gas rates from historical monthly production data using the tubing size as a variable input parameter. The Nonlinear Autoregressive with External Input (NARX) is trained using the Levenberg-Marquardt algorithm. In order to improve training performance pre-processing was undertaken, the Lowess filter was applied and normalization was conducted.

The network showed a satisfactory prediction of the gas flow rate. Having the lowest mean squared error, the network was constructed with two layers, each containing 75 neurons. Coleman criterion was introduced to indicate the onset of liquid loading. When the predicted flow rate falls below the Coleman rate, the liquid loading alerter is triggered. The alerter forecasts the month in which liquid loading may occur up to a maximum period of twelve months. The prediction becomes more precise as the alerter approaches the the liquid loading moment and it may be concluded that the alerter is accurate up to two months in advance. Moreover, the alerter is able to predict nine months prior to liquid loading with a deviation of up to two months either side. Nevertheless the Coleman Criterion has the tendency to calculate lower critical rates than other methods. Therefore it may predict liquid loading to occur at a future month than the exact month of liquid loading.

Acknowledgments

This research would not have been possible without the help and support from others, therefor I would like to thank those who contributed to my research process.

First of all I would like to thank EBN for providing the data, software applications, office facilities and a great working environment. From EBN a special thanks goes to my supervisor Ferhat Yavuz. He helped me, amongst others, to get acquainted with the subject, gave suggestions and feedback and arranged meetings with operators. I would also like to thank Raymond Godderij for his feedback and ideas and all other colleagues at EBN that helped me during my research.

Furthermore a special thanks goes to my supervisor, Auke Barnhoorn, from Delft University of Technology for his time, suggestions and the feedback on my report. Next I would like to thank Pacelli Zitha for his feedback and suggestions.

Finally, I would like to give a special thanks to my colleague-interns for the feedback and the good laughs we had. And last, but not least, I would like to thank my family, Wilco Vlenterie and friends for their support throughout my study period.

E.K. Bouw
Delft, January 2017

Contents

List of Figures

1 Introduction

Many gas fields in the Netherlands are in the mature or tail-end production phase, presenting operational and financial challenges when producing these assets [Yavuz and Jansen, 2013]. There are approximately 260 active fields in EBN's portfolio, of which the remaining recoverable gas volumes are categorized in four different segments; build up, plateau, mature and tail end. The majority (85%) of the remaining fields are considered to be mature or in the tail-end production (Figure 1.1).

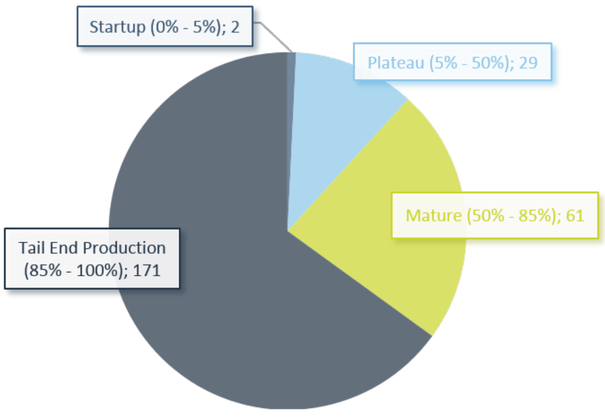


Figure 1.1: Category of active fields in the Netherlands [Yavuz and Jansen, 2013].

To be able to produce from these mature or tail-end production wells the concept of liquid loading needs to be introduced. Liquid loading is one of the major issues that decrease flow production substantially or even stops flow completely. To overcome this problem several End of Field Life (EoFL) techniques have been introduced by operators in the Netherlands in the last ten years. Most commonly applied technology in the Netherlands are velocity strings and foam injection; both batch and continuous, (Figure 1.2). The analysis on the historical production performance of the Dutch gas fields has shown that the techniques are effective, not only on the recoverable reserves, but they also have a positive effect on platform life extension [Yavuz et al., 2013].

A solution for liquid loading can thus be provided by EoFL techniques. However until today it is difficult to know when to install these techniques due to the prediction uncertainty of the liquid loading moment. This means that the well might already be liquid loading or even stopped producing completely when only then the operator decides to implement an EoFL technique. An inefficient approach, leading to financial losses. A solution can be found in artificial intelligence. Learning from historical data with the use of artificial intelligence is going to have a major effect on the efficiency of operations in the near future in all fields. Considering the subject of liquid loading, big data may predict future instability of production rates and is therefore very useful in predicting liquid loading in advance. A first attempt using artificial intelligence, in particular artificial neural networks (ANN), to predict the liquid loading moment is performed in this thesis.

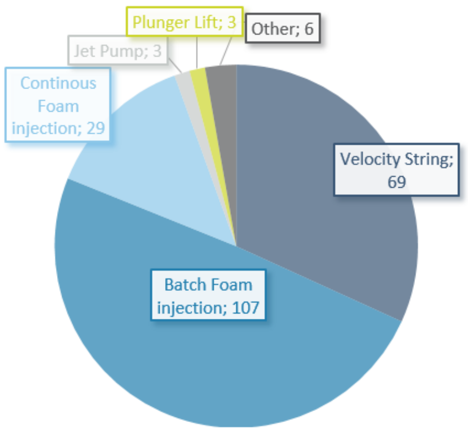


Figure 1.2: Distribution of deliquification technologies of gas wells in the Netherlands [Yavuz and Jansen, 2013].

1.1. Research Objectives

This report focuses on two topics considering EoFL wells. As it is important for an operator to know when liquid loading will occur and how to remove a liquid column, this thesis looks at predicting the onset of liquid loading and at deliquifying gas wells considering multiple EoFL techniques to sustain gas production of gas fields in the Netherlands.

At first, the analysis is performed to understand the problem of liquid loading and its solutions. At the start, the production data of a small number of wells will be gathered in order to perform a quick analysis study to determine the potential of EoFL techniques. When this proved to be successful a larger amount of wells, different techniques and various operators, will be considered and an analysis of the added value of the deliquified wells will be made. To predict the volume gain by EoFL techniques the TNO deliquification tool* is used. The tool simulates the effects of EoFL techniques and a comparison can be made to the production performance without a mitigation technique [Schiferli et al., 2013]. At the end the economic gain will be calculated for a well using an EoFL technique.

Conducting this study provides operators with lessons for the future and to quantify the potential of different EoFL techniques in terms of both added recoverable resources and economic value added.

Secondly an ANN is built to predict the onset of liquid loading. To date, it is very difficult to predict the moment when gas flow is unable to lift liquid to surface. Therefore a forecast is made of the historical monthly production rate and from this a prediction will be made of the month in which liquid loading occurs. The neural network (NN) will be built using the MATLAB® toolbox [MATLAB®, 2016f].

1.2. Research Outline

The next two chapters will explain the fundamentals of the two research topics. Fundamentals of Gas Well Deliquification describes a production profile, the concept of liquid loading and the two techniques studied to remove water from the well. Fundamentals of Artificial Neural Network shows how the network is built and trained. The next chapter, Methodology, will describe the step by step procedure taken during this research. The approach has changed several times during the development of the neural network to obtain the optimum network in predicting the liquid loading moment. Then the results are shown and discussed extensively in its succeeding chapter. Conclusions and recommendations are presented in the final chapters.

* The TNO deliquification tool is build by TNO in cooperation with EBN and several operators. Contact for further requirements.

2 Fundamentals of Gas Well Deliquification

The share of gas is increasing in the world's energy supply. Special attention needs to be given to those wells that experienced declining gas production or even stopped production completely [Lea et al., 2008]. The reduction of gas production is due to declining reservoir pressure and gas velocities, next to increased water production. Increased water production can cause a column of water to accumulate at the bottom of the well, preventing reservoir fluids from entering the wellbore. This phenomena is called liquid loading of the well [Rao, 1999]. In this chapter the principle behind the reasons for liquid loading, together with techniques to stimulate the well, are explained.

2.1. Production Profile

The production of an oil or gas field tends to pass through a number of stages. This can be described by the production curve shown for oil in Figure 2.1. After an oil field is discovered, the new field is appraised to determine the development potential of the reservoir. If it meets volume and production rates required for commercial viability, further development follows and the first oil production marks the beginning of the build-up phase. The production gradually builds up to the plateau phase, where the fully installed extraction capacity is used, before finally arriving at the onset of decline as subsurface conditions will no longer be able to support this rate of extraction. The decline phase ends in abandonment once the economic limit is reached [Höök, 2009]. The moment of abandonment is preferably extended up to the last drop of oil. The gas profile looks similar, but often shows a shorter plateau phase. In order to extend abandonment, end of field life techniques are implemented. The aim of this study is predict the liquid loading moment in order to avoid abandonment in an early stage.

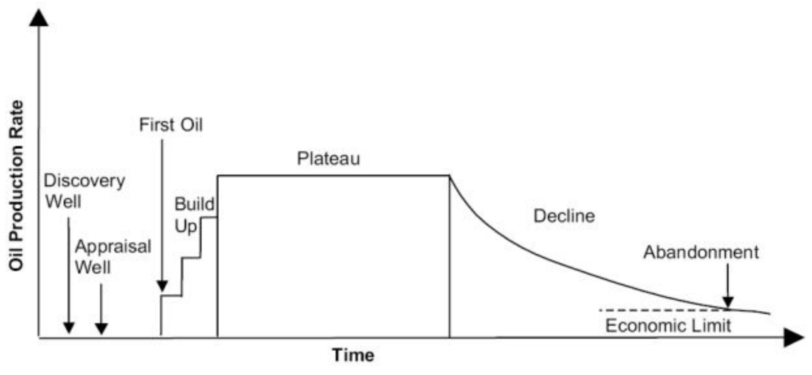


Figure 2.1: A theoretical production curve, describing the various stages of maturity [Höök, 2009].

2.2. Liquid Loading

Liquid loading of a gas well is the inability of the produced gas to remove the produced liquids from the wellbore. When a gas well is producing, the pressure in the gas reservoir is high and the gas velocity in the tubing is sufficient to drag the liquid upwards to the surface. However after several years, towards the end of field life, the pressure in the reservoir has become so low that the gas does not meet critical velocity and an accumulation of liquid down-hole occurs. The accumulation of liquid will impose an additional back pressure on the formation that can significantly affect the production capacity of the well [Turner et al., 1969]. The liquid can come from interstitial water in the reservoir matrix or it can be formed due to condensation of water vapor and hydrocarbon gas as the pressure and temperature decrease along the trajectory of the well tubing [van Nimwegen, 2015]. The production will cease entirely to the extent that the well has to be shut in, even though there is still natural gas remaining in the reservoir.

2.2.1. Multiphase Flow

To understand the effects of liquids in a gas well, it is important to understand the liquid and gas phases interaction under flowing conditions. Four flow regimes are presented for vertical multiphase flow [Lea et al., 2008]. These four patterns are determined by the velocity and the relative amount of the gas and liquid phase (Figure 2.2). A gas well may go through any or all of these flow regimes during its producing life.

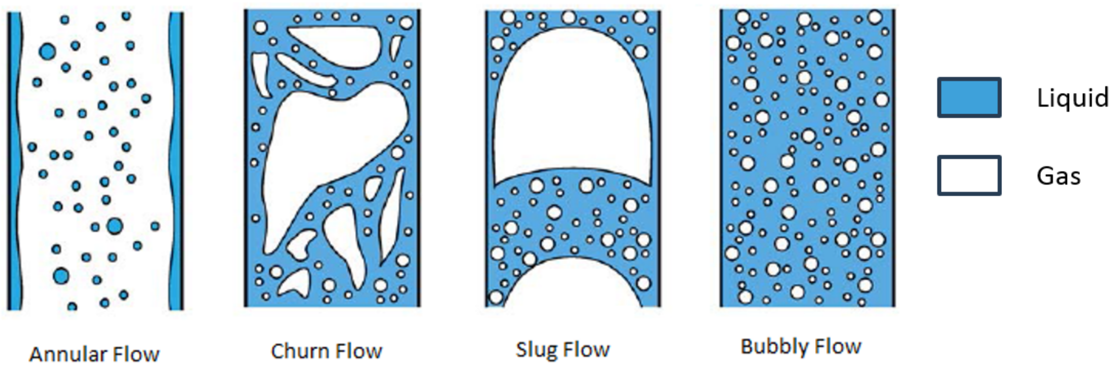


Figure 2.2: The four flow regimes for vertical multiphase flow [Lea et al., 2008].

- **Annular Flow.** Annular flow occurs at high gas velocities, in which gas is the continuous phase and the liquid is present in dispersed droplets in the gas and in a thin film at the wall of the pipe.
- **Churn Flow.** When gas velocity is decreased, the flow changes from a continuous gas phase to a continuous liquid. This marks the transition to churn flow. So instead of moving upwards the liquid film reaches a certain point where it starts to move downwards, liquid loading is related to this transition.
- **Slug Flow.** As the gas rate decreases even further, gas bubbles alternate with liquid slugs.
- **Bubbly Flow.** At last at even lower gas flow rates, bubbly flow occurs, where gas is present as small bubbles, rising in the liquid.

2.2.2. Indicators of Liquid Loading

As the flow rate declines or the well stops production completely due to liquid loading, it is important that the effects are detected at an early stage to prevent costly losses in gas production and reservoir damage. There are several symptoms that indicate when a gas well is having problems with liquid loading and are presented shortly below.

- **Presence of orifice pressure spikes**. It is possible to recognize the start of liquid loading by looking at the slugs of liquid produced at the well head. This slug flow is illustrated by two-pen recorder charts as pressure spikes. Accompanied by a rapid drop in production and a drop in surface tubing pressure the two-pen chart is a sure indication of liquid loading problems. However it is not the most efficient technique as many wells have a liquid knock-out before these orifice measurements [Lea et al., 2008].

- **Shooting fluid levels on flowing gas wells**. Analysis of the acoustic fluid levels acquired on gas wells can be used to determine the amount of liquid loading into the formation, the approximate gas rate into tubing, the equivalent gradient of the gaseous liquid column in the tubing at the flowing bottom hole pressure [Rowlan et al., 2006].
- **Increasing difference between the tubing and casing pressure**. When liquids accumulate in the bottom of the wellbore, the added pressure head on the formation lowers the tubing pressure. Moreover when gas is produced from the reservoir gas percolates into the tubing casing annulus, causing an increase in the surface casing pressure. Therefore in a flowing well, a decrease in tubing pressure and a corresponding increase in casing pressure are indicators of liquid loading [Lea et al., 2008].
- **Pressure survey shows a sharp change in pressure gradient**. Pressure surveys measure the pressure with depth of the well either while shut in or while flowing. The measured pressure gradient is a direct function of the density of the medium and the depth, and for a single static fluid, the pressure with depth should be nearly linear. Since the density of the gas is significantly lower than that of water or condensate, the measured gradient curve will exhibit a sharp change of slope when the standing liquid in the tubing is encountered. The difficulty lies however in the presence of a two-phase flow regime as this changes the slope measured by the survey and wells having a tapered tubing string as the change in cross-sectional flow area will cause a change in the flow regime [Lea et al., 2008].
- **Erratic production and increase in decline rate**. An important indication of downhole liquid loading problems is the shape of a decline curve [Park, 2008]. The decline curve should be analyzed for long periods and changes in the general trend need to be diagnosed. This is explained by Figure 2.3. The smooth exponential type decline curve represents a single gas production, while the sharply fluctuating curve is a an indicative of liquid loading as it shows a sudden departure from the existing curve to a new steeper slope. Well abandonment will occur far earlier than with the original curve [Lea et al., 2008].

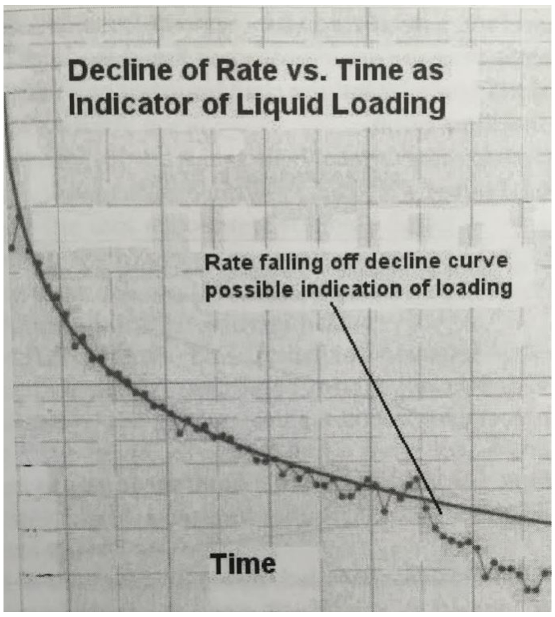


Figure 2.3: Decline rate showing the onset of liquid loading [Lea et al., 2008].

- **Calculating minimum critical Velocity**. High rate wells produce liquid together with gas production. After some time the rate drops as reservoir pressure decreases. The well will produce below a critical rate that can transport liquid to the surface. This results in an accumulation of liquids in the wellbore and the question arises until when the gas can still bubble through the accumulated liquids. The minimum gas velocity required to carry liquids to the surface can be calculated by Turner rate [Turner et al., 1969] and Coleman Criterion [Coleman et al., 1991] and by doing so provides an indication of the onset of liquid loading [Lea et al., 2008]. The method is explained in the succeeding section.

All these methods are indicators of liquid loading. However, they do not predict the liquid loading moment but only record the moment it happens. Therefore, in this research, the onset of liquid loading is predicted by forecasting the flow rate. When the rate is declining, as was seen in Figure 2.3, and the flow rate falls below the minimum critical velocity a prediction of the onset of liquid loading can be made.

2.2.3. Critical Velocity

In this section the method to predict the onset of liquid loading is presented. The Turner criterion [Turner et al., 1969] is most commonly used to predict liquid loading. This technique was developed for a substantial accumulation of well data and has been shown to be reasonably accurate for vertical wells. The method is applicable at any point in the well and should be used in conjunction with methods of Nodal Analysis if possible [Lea et al., 2008]. There are two possibilities when gas is flowing upwards through the tubing. Either gas velocity is sufficient to drag liquid upwards, such that the average velocity is upwards, or the gas velocity is insufficient and liquid moves downwards on average, causing liquid to accumulate at the bottom of the well. The velocity a liquid droplet can attain against gravity, thus the minimum velocity the gas flow needs in order to produce liquids along with it, is called the critical gas velocity [Binli, 2009]. As the Turner relation was developed from data from surface tubing pressures mostly greater than 1000 psi, Turner's critical rate can be used for high pressures but it is not as accurate for a low tubing head pressures. Therefore the Coleman criterion [Coleman et al., 1991] is introduced. Similar relationships describe the minimum critical flow rate for lower surface tubing pressures, only without the Turner 1.2 adjustment that Turner used to fit his data. In this thesis wells at the end of their producing lives are examined, therefore Coleman is considered most suitable. The Turner velocity is shown in Equation 2.1 and the Coleman criterion in Equation 2.2.

$$U = 1.92\frac{\sigma^{1/4}(\rho_L - \rho_g)^{1/4}}{\rho_g^{1/4}} \tag{2.1}$$

$$U = 1.59\frac{\sigma^{1/4}(\rho_L - \rho_g)^{1/4}}{\rho_g^{1/4}} \tag{2.2}$$

where U is the critical velocity, σ is the surface tension, and ρ_L and ρ_g the density of liquid and gas respectively. For water the surface tension is 60 dyne/cm and the density is 67 lbm/ft^2. The gas density in lbm/ft^3 is given by Equation 2.3.

$$\rho_g = \frac{M_{air}\gamma_g P}{R(T+460)Z} = 2.715\gamma_g\frac{P}{(460+T)Z} \tag{2.3}$$

where M_{air} is the molar mass of gas, P is wellhead pressure in bar, γ_g is the gas gravity, R is the gas constant, T is the wellhead temperature and Z is the gas compressibility. Although critical velocity is the controlling factor, one usually thinks of gas wells in terms of production rate rather than velocity in the wellbore. From the ideal gas law the rate can be calculated and is shown in Equation 2.4.

$$Q = \frac{PT_{sc}AU}{P_{sc}ZT} = \frac{3.067PUA}{(T+460)Z} \tag{2.4}$$

where A is the tubing cross-sectional area. The second formula gives the critical gas flow rate in field units, namely MMscf/d.

2.2.4. Systems Nodal Analysis

A useful tool for the analysis of well performance is system Nodal Analysis. It divides the total well system into subsystems at a specific location called the nodal point. One subsystem considers the inflow from the reservoir into the well and the other one considers the outflow system from the bottom of the well to surface. For each of these two subsystems the pressure at the nodal point is calculated and plotted as two separate, independent pressure-rate curves.

The reservoir production performance is shown in the inflow performance relation (IPR). The IPR curve plots the flow rate against the bottomhole pressure. When the bottomhole pressure is equal to the reservoir pressure the flow rate is zero and the maximum flow rate is given where the bottom hole pressure is zero [Gromotka, 2015]. The reservoir pressure slowly decreases over the years and the reservoir curve moves towards lower gas flow rates and lower bottomhole pressures.

The tubing performance curve (TPC) depicts the relation between the pressure drop of the well and the flow rate at the well head. The tubing pressure drop is essentially the sum of the surface pressure, the hydrostatic pressure of the fluid column, and the frictional pressure loss resulting from the flow of the fluid out of the well. Thus it describes the performance of a specific tubing size, depth and wellhead conditions. The gas in a well often contains liquid, i.e. there is two-phase flow. This results in a specific shape for the TPC curve. The pressure drop over the well needs to be large enough to compensate the forces working on the fluid. At low flow rates the gravitational force is large due to the large ratio of liquid [Gromotka, 2015]. The flow is typically bubbly flow, a flow regime that allows liquids to accumulate in the wellbore. Slightly to the left of the minimum in the TPC, the flow is often in the slug flow regime, also inefficient as the liquids are only transported to surface periodically in the form of large slugs. At high flow rates the frictional force between the fluid and the well is large. This happens at the right side of the minimum and flow is usually in the mist flow regime that effectively transports small droplets of liquids to surface. This results in a so called J-curve, shown in Figure 2.4 [Lea et al., 2008]. The TPC curve is a combination of all possible steady state production conditions for the well, given certain conditions.

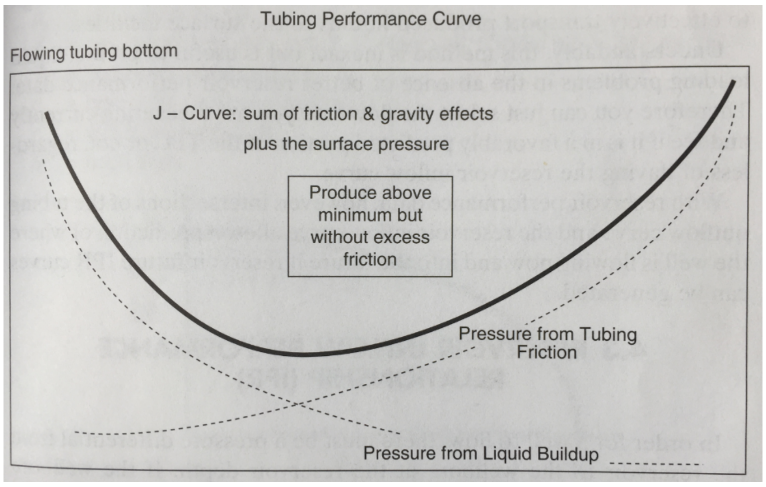


Figure 2.4: Standard shape of the TPC curve [Lea et al., 2008].

The intersection of the inflow and outflow curves is the predicted operating point where the flow rate and pressure from the two independent curves are equal. The flow rate is called the natural production rate. The amount of operating points can be zero, one and even two. If there are two operating points, however, one is often unstable. An operating point is stable when the flow adjusts back to the operation point after slight perturbation from natural production conditions. Otherwise it is considered unstable. The principle is shown in Figure 2.5 and Figure 2.6 [Lea et al., 2008]. The rule of thumb is that the flow rate to the left of the minimum of the TPR curve is unstable.

At low reservoir pressures there is no stable operation point (the two curves do not intersect) and production of gas is no longer possible. The goal of deliquifying gas wells is to change the TPC in such extent that at lower reservoir pressure a stable operation point can still be found. This can be read in the next section.

2.3. Gas Well Deliquification Methods

Several methods have been developed to postpone the onset of liquid loading. These deliquification techniques can be subdivided into two categories, namely the methods that use the energy of the well

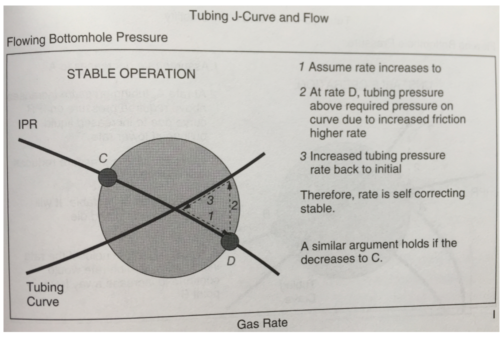

Tubing J-Curve and Flow
Flowing Bottomhole Pressure
STABLE OPERATION
IPR
C
D
1
2
3
Tubing Curve
Gas Rate
1 Assume rate increases to
2 At rate D, tubing pressure above required pressure on curve due to increased friction higher rate
3 Increased tubing pressure rate back to initial
Therefore, rate is self correcting stable.
A similar argument holds if the decreases to C.

Figure 2.5: The stable operation of the TPC and IPR curve.

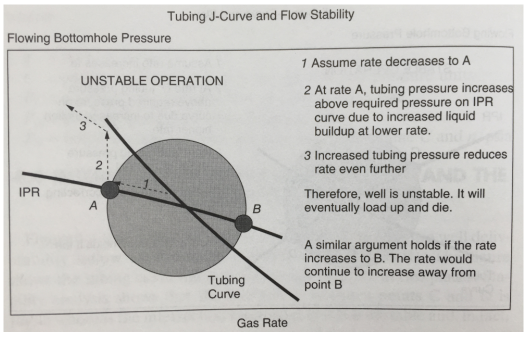

Tubing J-Curve and Flow Stability
Flowing Bottomhole Pressure
UNSTABLE OPERATION
IPR
A
B
1
2
3
Tubing Curve
Gas Rate
1 Assume rate decreases to A
2 At rate A, tubing pressure increases above required pressure on IPR curve due to increased liquid buildup at lower rate.
3 Increased tubing pressure reduces rate even further
Therefore, well is unstable. It will eventually load up and die.
A similar argument holds if the rate increases to B. The rate would continue to increase away from point B

Figure 2.6: The unstable operation of the TPC and IPR curve.

fluids to lift liquid to surface and methods that use an external energy source to lift liquids. The latter includes downhole pumps or the injection of compressed lift gas. In this research focus lies on the first category including two techniques; velocity strings and the injection of surfactants.

2.3.1. Tubing Size

The reason to run a smaller tubing instead of the production string is to increase the velocity for a given rate and sweep the liquid out of the well and the tubing. The concept of a smaller diameter is too reduce the cross-sectional flow area. The smaller cross-sectional flow area increases the gas velocity in the tubing. The higher gas velocity at the bottom of the tubing provides more transport energy to lift liquid up out of the well [Lea et al., 2008].

The basic concept of tubing design is to have a large enough tubing diameter such that frictional pressure losses in the tubing are minimal, and a small enough tubing such that the well is restored back to flowing production [Rao, 1999]. In general, faster velocity reduces the liquid holdup, the percentage of liquid by volume in the tubing and lowers the flowing bottomhole pressure attributed to gravity effects of the fluids in the tubing. However, a tubing size too small for the production rate can cause excess friction and requires a larger flowing bottomhole pressure. When using a velocity string for liquid loading care needs to be taken to the fact that in a small tubing a volume of fluid that may have been negligible in larger tubing can be significant in small tubing.

When resizing tubing, the reservoir inflow is needed from a reservoir model or an IPR curve obtained from well test data. From the nodal concepts, seen in the previous section, a tubing curve for various sizes can be generated. The shift of the TPC for smaller diameters can be seen in Figure 2.7. The curve is shifted upwards to the left from the original TPC and thus the critical velocity is lower than in the larger tubing.

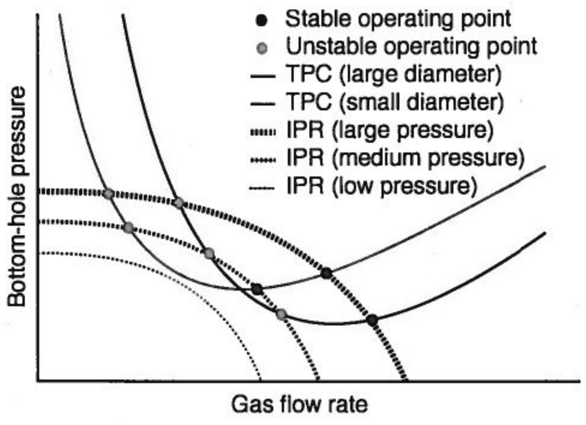


Figure 2.7: The TPC of the original tubing and the shifted TPC when installing a velocity string [van Nimwegen, 2015].

2.3.2. Foam Assisted Lift

The principal benefit of using foam as a gas well dewatering method is that liquid is held in the bubble film and exposed to more surface area resulting in less gas slippage and a low density mixture. The foam is effective in transporting the liquid to the surface in wells with very low gas rates when liquid holdup would otherwise result in sizable liquid accumulation and high multiphase flow pressure losses [Lea et al., 2008].

To explain in more detail, the gas bubbles are separated from each other in foam by a liquid film. Surfactants are employed to reduce the surface tension of the liquid to enable more gas-liquid dispersion. The liquid film between bubbles has two surfactant layers back to back with liquid contained between them. This method of tying the liquid and gas together can be effective in removing liquid from low volume gas wells.

The principle can also be explained by looking at Equation 2.2. When foam reduces the surface tension, it reduces the required critical velocity. Foam also reduces the density of the liquid droplets to a complex structure containing water and/or condensate and gas. A rule of thumb presented by Weatherford is that foaming water will reduce critical velocity by about two-thirds in a gas well by reducing the surface tension and the liquid density simultaneously [Lea et al., 2008]. Next to this the TPC curve is also an indicator that production is possible again due to surfactants, as surfactants change the fluid properties as said before, the TPC changes. The TPC will shift to the left and when it crosses the IPR curve, the well is back into production. This can be seen in Figure 2.8 [van Nimwegen, 2015].

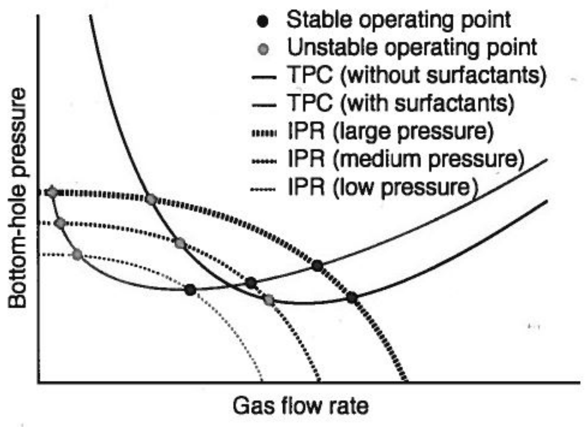


Figure 2.8: The TPC curve considering only water in the tubing and the shifted TPC after foam injection [van Nimwegen, 2015].

Foam is a simple and inexpensive EoFL method for low rate wells. The foaming tendency for various systems depends on the amount and type of well fluids and on surfactant effectiveness. Wells producing substantial condensate (greater than 50%) may not foam. The method is mainly applicable to wells with low gas rates. In gas well liquid removal applications, the liquid gas surfactant mixing must be accomplished downhole and often in the presence of both water and liquid hydrocarbons. There are several ways to inject surface active agents (surfactants), namely dropping soap sticks down the tubing, batch treating down the annulus (with no packer present) or lubricating a capillary string down the tubing for injection of surfactants.

3
Fundamentals of Artificial Neural Networks

3.1. Introduction

Artificial neural networks can realize complex learning and adaptation tasks by imitating the function of biological neural systems. In contrast to knowledge-based techniques, no explicit knowledge is needed for the application of neural nets. Their main strength is the ability to learn complex functional relations by generalizing from a limited amount of training data. Neural networks can thus be used as a black-box model for nonlinear systems and can be trained by using input and output data observed in the system.

In Figure 3.1 a model of ANN is considered. The mathematical model mimics the functionality of biological neurons (called artificial neurons) in various levels of detail. It basically is a static function with several inputs (representing dendrites) and one output (the axon). Each input is associated with a weight factor (synaptic strength)[Babuska, 2010]. The weighted inputs are added up and passed through a nonlinear function, which is called the activation function. The value of this function is the output of the neuron.

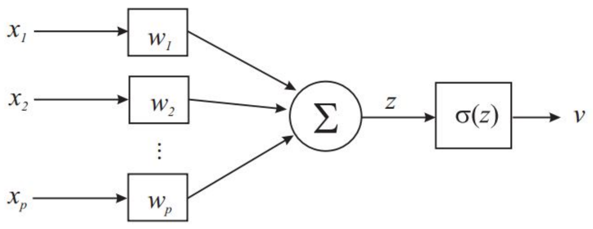


Figure 3.1: Artificial Neuron [Babuska, 2010].

To put this scheme into a mathematical formula, Equation 3.1 is presented below.

$$z = \sum_{i=1}^{p} w_i x_i = w^T x \tag{3.1}$$

Sometimes, a bias is added when computing the neuron's activation. The bias can be regarded as an extra weight from a constant input as is shown in Equation 3.2.

$$z = \sum_{i=1}^{p} w_i x_i + b = [w^T b] \begin{bmatrix} x \\ 1 \end{bmatrix} \tag{3.2}$$

The activation function maps the neuron's activation z into a certain interval. Often used activation functions are a threshold, sigmoidal and tangent hyperbolic functions.

3.2. Architecture

The most common net is the artificial feedforward neural network, shown in Figure 3.2. The network with several layers are called multi-layer neural networks, as opposed to single-layer networks that only have one layer. They consist of one input layer, one output layer and a number of hidden layers in between them. The layers consist of simple nonlinear processing elements, the neurons. The neurons are interconnected through adjustable weights. The information relevant to the input and output mapping of the net is stored in these weights. In the feedforward neural network the information flows only in one direction, from the input layer to the output layer. The sum of the products of the weights and the inputs is calculated in each node, and if the value is above some threshold (typically 0) the neuron fires and takes the activated value (typically 1); otherwise it takes the deactivated value (typically -1) [Babuska, 2010].

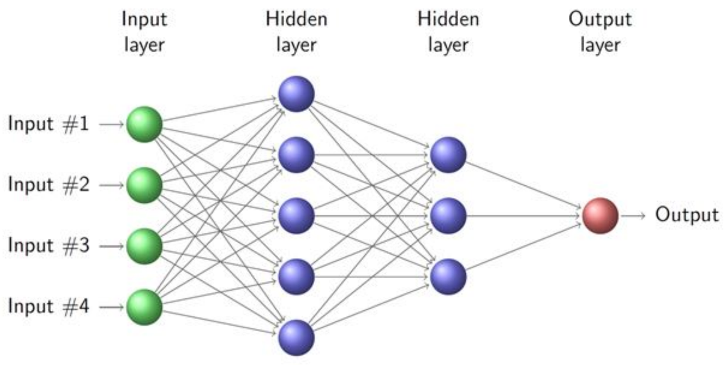


Figure 3.2: A multi-layer feedforward neural network (www.omniresources.com).

3.3. Training

The question for training a neural network is how to determine the appropriate structure (number of hidden layers, number of neurons) and parameters of the network. The training itself is the adaptation of weights in a multi-layer network such that the error between the desired output and the network is minimized.

3.3.1. Feedforward Computation

From the network inputs x_i, $i = 1, ..., N$, the outputs of the first hidden layer are first computed. Then using these values as inputs to the second hidden layer, the outputs of this layer are computed, and so on. Finally, the output of the network is obtained [Babuska, 2010].

The computation proceeds in three steps:

1. Compute the activations z_j of the hidden layer neurons:

$$z_j = \sum_{i=1}^{p} w_{ij}^h x_i + b_j^h, \quad j = 1, 2, ..., h \tag{3.3}$$

where w_{ij}^h and b_j^h are the weight and the bias of the hidden layers, respectively.

2. Compute the outputs v_j of the hidden layer.

$$v_j = \sigma(z_j), \quad j = 1, 2, ..., h \tag{3.4}$$

3. Compute the outputs y_l of output layer neurons.

$$y_l = \sum_{j=1}^{h} w_{jl}^o v_j + b_l^o, \quad l = 1, 2, ..., n \tag{3.5}$$

where w_{jl}^{o} and b_{l}^{o} are the weight and the bias of the output layers, respectively.

3.3.2. Weight Adaptation

When the output of the network is computed, it is compared to the desired output. The difference between the output of the network and the predicted output is called the error. By backward computation the weights are adjusted layer by layer, starting with the output layer, in order to decrease this error. The error is used to adjust the weights in the net via the minimization of a certain cost function, in this research this will be the mean squared error.

3.4. Performance

To evaluate the performance of the neural network the mean square error (MSE) is used in this study, shown in Equation 3.6.

$$RMSE = \frac{1}{n}\sum_{i-1}^{n} = (y_i^{act} - y_i^{pred})^2 \qquad (3.6)$$

where y^{act} is the true output and the y^{pred} is the predicted data, and n is the number of data points [Chakra et al., 2013].

Regression analysis is carried out for demonstrating the model performance during validation phase. The model performance assessed by regression analysis is illustrated using regression plots. With these plots one can assess how consistent the forecasted data is with the actual data. It is considered that if the regression of a model follows the output = target line more closely the model can perform better prediction, since the output = target lines represent the best fit [Kooijman, 2011].

4 Methodology

4.1. Production Analysis

To understand the liquid loading phenomena and the influence of a velocity string or foam injection 64 wells are considered in this thesis from the 260 wells that are in EBN's portfolio (Figure 4.1). A data base is built with these wells including the reservoir parameters, total produced volume, producing years, volume gain, economical costs of mitigation technique etc. A short study is conducted on the history of these wells and the important parameters; reservoir pressure, well head pressure, gas density, reservoir temperature, liquid gas ratio and the non-Darcy coefficient (A) and the friction factor (F). These parameters are introduced in the TNO deliquification tool. The tool predicts the ultimate recovery, abandonment pressure, cumulative production and the producing years for several techniques and compares this to the case without mitigation. In the Appendix the interface of the tool is shown. A short description of the parameters is given below.

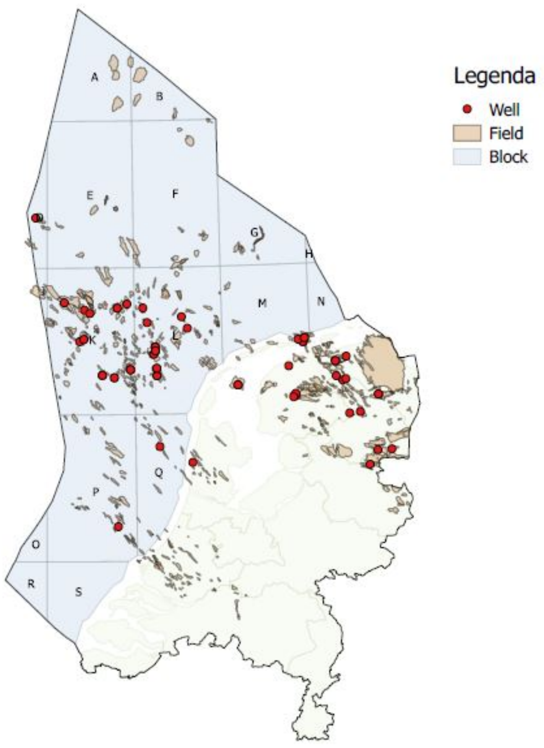


Figure 4.1: A map of the Netherlands showing the location of the wells, both onshore and offshore, that are used in this research.

- **Wellbore Properties** These include the wellhead pressure and the wellhead temperature at the time of installing a velocity string or injecting foam.

- **Reservoir Properties** These include the reservoir pressure, the initial reservoir temperature and the dynamic gas initially in place (GIIP). The reservoir pressure of fluids within the pores of a reservoir, usually comprise the hydrostatic pressure or the pressure exerted by a column of water from the formation's depth to sea level. Both the initial pressure, measured in a discovery well, and the reservoir pressure before installing an EoFL technique are taken into account in calculating the volume gain.

- **Gas Density** The gas density can be calculated by the sum of the density of air and the specific gas gravity, meaning the density of the gas relative to air [Jansen, 2015]. At standard conditions (15 °C and 1 atm) the gas density is 1.2260 kg/m^3. The gas gravity at standard pressure and temperature is searched for every well specifically, usually around 0.65.

- **Gas Liquid Ratio** The ratio of produced gas volume to total produced liquid (oil and water) volume at surface [Jansen, 2015].

- **Darcy and non-Darcy Coefficient** A is the Darcy coefficient, also known as the laminar flow coefficient. When flow is laminar the Darcy equation is valid and the flow describes a linear relationship between volumetric flow rate and pressure gradient, but for non-Darcy flow a second term is introduced by Forcheimer, represented by the B factor. The B factor is the non-Darcy coefficient, also called the frictional flow (F) in the deliquification tool and it incorporates the pressure losses related to turbulence in the fluid [Veeken et al., 2010]. The quadratic equation for saturated oil and gas wells is seen in Equation 4.1 [Schiferli et al., 2013]. From this equation the A & F factors are determined. The pressure and flow rate are taken from the well test done in a newly drilled well. Note that these values change slightly during production and as this research looks at wells at the end of their lives the A & F factors may not be exact. Though it is the best representation available here.

$$P_{res}^2 - P_{wf}^2 = Aq + Bq^2 \tag{4.1}$$

To understand the influence of these parameters on the gas production several wells are tested with the deliquification tool. These wells were chosen due to the data available from operators. The parameters described above are needed when using the tool, next to the well completion. The quantification of the analysis focuses on velocity strings and foam. For a velocity string the tubing size is needed to calculate the operational window and flow rates. For application of foam, a reduction of 50% of critical velocity is assumed.

Figure 4.2 shows the production profile of well 1. From the production history it strikes the eye that the velocity string is implemented only in December 2010, although the well has not been producing since may 2002 because of back-out production from other wells in the field. The well is closed in, pressure builds up and the well produces from 2008 to the beginning of 2010 with an insignificant amount. With a $3\frac{1}{2}$ inch velocity string another 122 million Nm^3 is produced in about $2\frac{1}{2}$ year, which is a good 6.9% of the total production of the well.

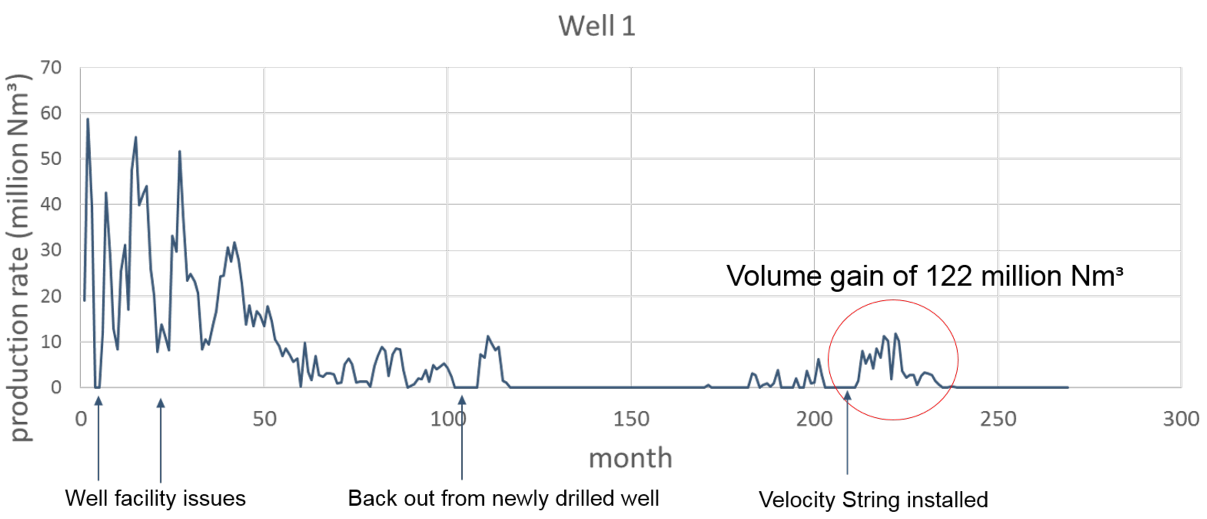


Figure 4.2: Production profile of well 1 including historic events.

4.1.1. Coleman Rate

In figure 4.3 the Coleman rate, calculated from Equation 2.2, is plotted versus the flowing tubing head pressure (FTHP). The critical rate is calculated by equation 2.4. From the following parameters, wellhead temperature of 50 °C and a specific gas gravity of 0.6, a gas compressibility factor is calculated by Beggs and Brill [Guo and Ghalambor, 2005]. The higher the pressure the higher critical rate is needed to keep production going. Note that the rate is given in Nm^3/month and the FTHP in bar. This is more practical in this research as the flow rates from the data base are given in cubic meters per month as well. The critical rate is calculated for each well to determine the liquid loading moment. The table showing the critical rate per well is shown in the Appendix.

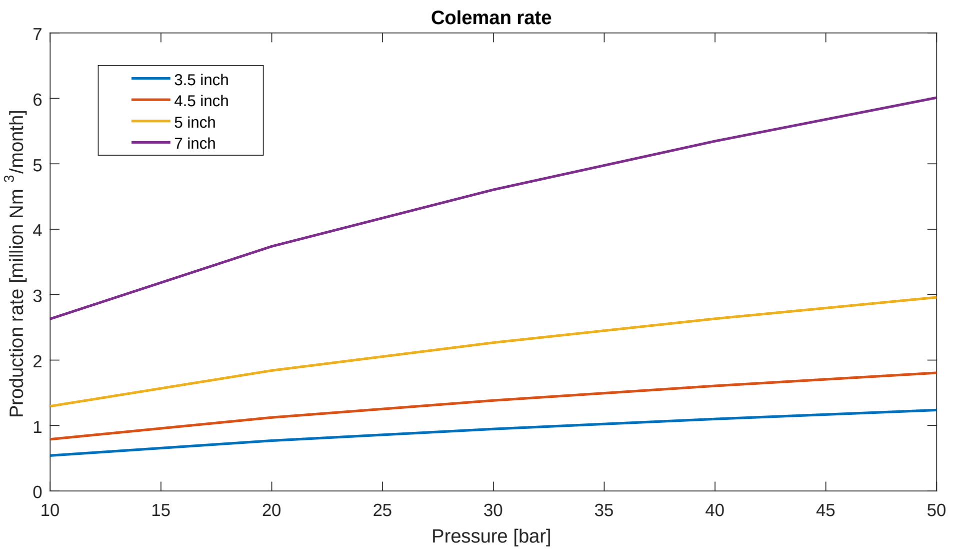


Figure 4.3: The Coleman critical rate versus the flowing tubing head pressure for a tubing size of 3.5, 4.5, 5 and 7 inch.

4.1.2. Analysis Performed

After investigating the production profiles of the wells and completing the data base, the following analyzes were carried out for all gas wells including the influence of EOFL techniques:

- A comparison of the volume gain and the additional producing years by a velocity string between the prediction from the deliquification tool, the operators expectation before implementation of a velocity string and the actual volume gain of the EoFL technique
- A sensitivity study on the parameters needed to run the deliquification tool
- Difference of actual gained volume and the expected volume gain by the operator for velocity strings, continuous foam and batch foam
- Influence of field size to production gain and the month in which a EoFL technique is installed, assuming this is the time a well can produce before liquid loading
- Difference of volume gain between velocity string and batch & continuous foam
- Economic analysis

4.2. Forecasting using Artificial Neural Networks

Literature has shown that ANN can be used to forecast gas production. In this research NN is trained to forecast the production rate per month with which the onset of liquid loading can be predicted. When the predicted flow rate falls below the Coleman rate, the liquid loading alerter is triggered. The aim is to have the system supply a time-based prediction, thus presenting the month in which liquid loading happens. In this chapter pre-processing, designing the neural net and the training procedure are described.

4.2.1. Input & Target Variables

The input variables need to be chosen carefully as it will influence the target values directly. The aim is to generate a production forecast per month. The production forecast will be called the target. A time series of historical production rates per month is therefore needed as one of the inputs. The production rate is influenced by several indicators, but in order to keep the training scheme simple the wells are subdivided according to the tubing size. Production will be higher in a larger tubing diameter and vice versa. The operator's choice of tubing size will depend on the flowing tubing head pressure, the size of the reservoir, etc. Thus the production curve will vary between these differently sized wells. By implementing them into the ANN, the network learns a specific pattern for each tubing size as input value. The diameters considered are 7, 5, 4.5 and 3.5 inch.

The MATLAB® neural network toolbox [MATLAB®, 2016f] has several wizards to solve different kind of problems. Prediction is a kind of dynamic filtering, in which past values of one or more time series are used to predict future values. Dynamic neural networks are used for nonlinear filtering and prediction. Therefore the Dynamic Time series is used, and in particular the Nonlinear Autoregressive with External (Exogenous) Input (NARX). This network predicts series y(t) given d past values of y(t) and another series x(t). Thus y(t) represents the production data and x(t) corresponds to the tubing diameter, in this case a constant. The network is shown in Figure 4.4.

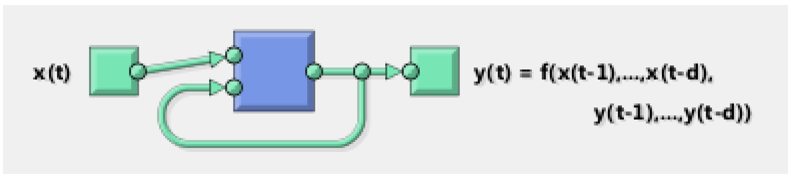


Figure 4.4: Dynamic Time series network [MATLAB®, 2016f].

Since only monthly data is available at EBN for this research, a compensation was needed for their inherent inaccuracy. The monthly data is an average of all producing days, thus the data is less precise than daily data. The days that a well has not produced are not included in this average, let alone the hours. The data acquisition in this thesis has revealed the many imperfections in producing numbers and the time a well can produce. It seems that producing only several days of a full month was the rule rather than the exception. By taking into account the up-time, meaning the days that a well was producing, a new monthly data set is obtained. This was simply done by dividing the producing days by the days in the specific month and then dividing the production rate per month by this ratio. The data now gives the total production of a particular month if it would have produced all 30 or 31 days. The large fluctuations are hereby smoothed.

4.2.2. Data Pre-processing

In the development of an ANN system the input data needs to be pre-processed before performing a prediction. In this research the two procedures in order to do so are noise reduction and normalization. The raw field data, and in particular the monthly data, include noise. Thus the production profile does not look as ideal as in Figure 2.1. Noise includes well problems that hinder production or even moments that the well is closed-in, often several times during a wells producing life. This could be due to several reasons; installation problems, back out production by another well in the same field, depletion and salt plugging problems, a sidetrack drilled to increase flow or intermittent production used when the bottomhole pressure is low.

The NN model will not be learning correctly when the well is not producing and it will look like liquid loading has already occurred when this is not the case, so the 0 values have been omitted by taking the average of the value before and after the 0.

Moreover when a well is not producing at its capacity for a certain pressure another EoFL technique might have been used to solve the issue, for example a compressor. When a tight reservoir is considered, fraccing may be a solution, also after some years of production. These stimulation methods are seen in the production profile, but will only cause confusing when training NN. Therefore these events are not wanted as input data in the training program. Besides, looking at the production profile described in the previous chapter, the phase most important factor in predicting the liquid loading moment will be the decline phase. The plateau phase is taken into account too since the longer the plateau lasts, the later liquid loading will occur. The other production data previous to this phase have been neglected manually before training NN. Additionally the intermittent production after liquid loading and the production data after using an EoFL technique have been erased as well. It was seen that the performance of the forecast was better than when including this data.

From the original 64 wells, 40 wells are used for training. This decrease of input wells occurred during pre-processing, including missing data of the up-time or a different tubing size than selected etc. The decrease is also caused by wells that produced only for 1 or 2 years. Their production decline was almost a straight line down and therefore not representable for the learning process. The performance of the training is increased by using good representable production profiles.

Smoothing Filter

To remove large peaks a smoothing filter was used. Several filters were considered, namely moving average, Lowess, Loess and Savitzky-Golay filtering.

Moving average is a type of low pass filter that transforms the time series monthly production data into smooth trends [Chakra et al., 2013]. From [Smith, 1997] is learned that the moving average is the most common filter, mainly because of it's simplicity and because it is optimal in reducing random noise while retaining a sharp step response. This filter does weighted averaging of past data points within a specified time span to generate a smoothed estimate of a time series. By doing so, it loses data when averaging. The span ranged from 6 to 10 months and an accurate prediction of the month in which liquid loading occurs is lost.

Therefore a different filter was needed. Lowess and Loess showed a smooth curve as well. The names Lowess and Loess are derived from the term "locally weighted scatter plot smooth", as both methods use locally weighted linear regression to smooth data. The smoothing process is considered local because, like the moving average method, each smoothed value is determined by neighboring data points defined within the span. The process is weighted because the toolbox defines a regression weight function for the data points contained within the span. Finally, the methods are differentiated by the model used in the regression: Lowess uses a linear polynomial, while Loess uses a quadratic polynomial [MATLAB®, 2016d].

Savitzky-Golay filtering can be thought of as a generalized moving average. The filter coefficients are derived by performing an unweighted linear least-squares fit using a polynomial of a given degree. For this reason, a Savitzky-Golay filter is also called a digital smoothing polynomial filter or a least-squares smoothing filter. A higher degree polynomial makes it possible to achieve a high level of smoothing without attenuation of data features [MATLAB®, 2016d].

The Lowess filter was proven to be best. The polynomial filters may give negative flow rates when filtering on the lower flow rate values. The Lowess filter did not show any negative value, while the Loess and Savitzky-Golay filters showed some negative values and were therefore not chosen as the filter for the production data.

The Lowess filter with a span of 7 is shown for one of the wells in Figure 4.5. The Savitzky-Golay is shown in Figure 4.6 showing a large negative value.

Logarithm

As the logarithmic data gives an almost straight curve, as can be seen in Figure 4.7, the option was presented to train the NN on the logarithmic scale to get a higher performance. It is believed that with a smooth curve the training is more accurate.

Logarithmic Return

The logarithmic return will give the difference between a data point and the next. The difference may give a better indication of the liquid loading moment, as the flow rate seems to change rapidly for many

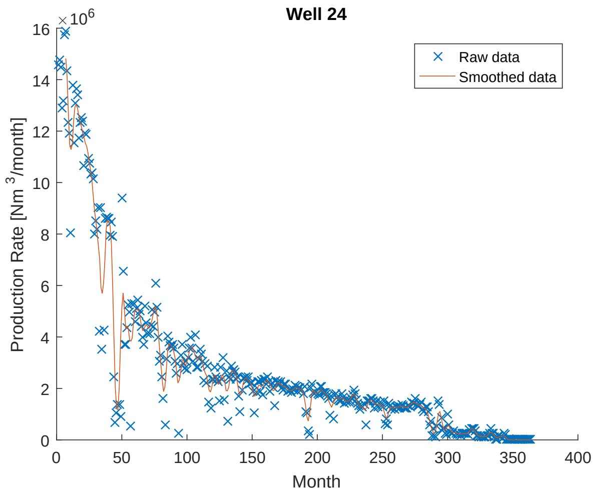


Figure 4.5: The production profile for one of the wells is shown for both the filtered and raw data by the Lowess filter.

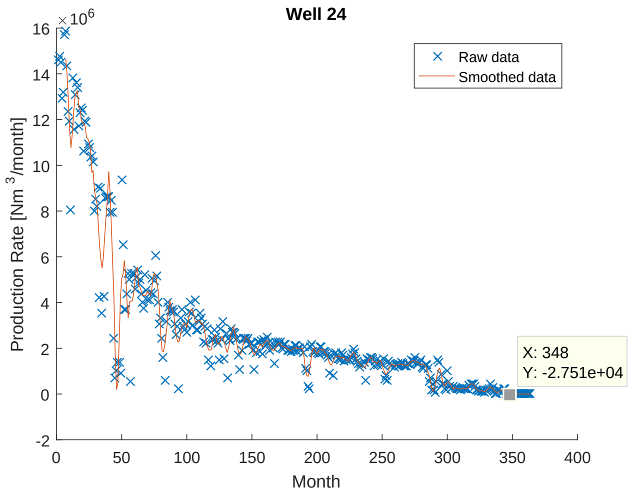


Figure 4.6: The production profile for one of the wells is shown for both the filtered and raw data by the Savitzky-Golay filter.

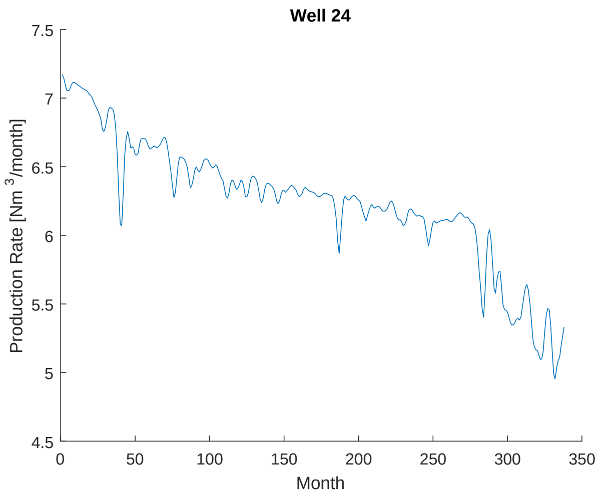


Figure 4.7: The production profile of one of the wells is shown with the logarithm on the production rate.

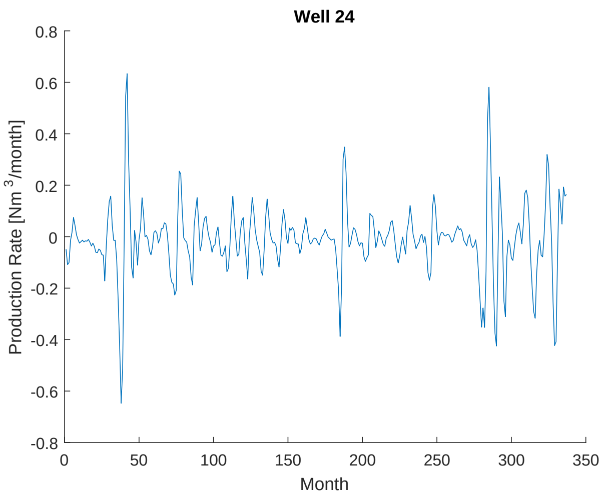


Figure 4.8: The production profile of one of the wells is shown with the log return on the production rate.

wells when liquid loading occurs. The log return is computed as shown in Equation 4.2 and for one of the wells the log return graph is shown in Figure 4.8.

$$r_t = log(\frac{Q_t}{Q_{t-1}}) \tag{4.2}$$

where r is the log return, Q the flow rate and t is the time-step.

Normalization

Normalization is a process of standardizing the possible numerical range of the input data. It enhances the fairness of training by preventing an input with large values from swamping out another input that is equally important but with smaller values. Normalization is also recommended because the network training parameters can be tuned for a given range of input data; thus, the training process can be carried over to similar tasks, for example oil production instead of gas production [Patro and Sahu, 2015]. Linear normalization was not possible as the values between the highest producing well and the well producing the least amount is too large. Namely the smallest data point was 20 times smaller than the largest data point. Therefore normalization using the logarithm of the minimum and maximum values was taken. The maximum value of the difference between the maximum and minimum production data point has a value of $9.84e^7$. Equation 4.3 shows how the data is normalized using this logarithm form. By doing so the data was normalized between 0 and 1 and the largest point is 1.2 times larger than the smallest point. This equation was used to normalize all the input and the output variables of NN. The method processes the input variable without any loss of information and it's transform is mathematically reversible.

$$x' = \frac{log(max(x) - min(x))}{log(9.84e^7)} \frac{x}{max(x) - min(x)} \tag{4.3}$$

where x' is the normalized input/output vector and x is the original input/output vector. This normalization process was applied to the whole data sets.

4.2.3. Training Function

The training of a multi-layer network is formulated as a nonlinear optimization problem with respect to the weights. There are various methods that can be applied, in this case the MATLAB® toolbox is used to train the network. Unless otherwise stated the default options are used of this toolbox.

The network training function is the Levenberg-Marquardt method, which is a second-order gradient. Levenberg-Marquardt updates weight and bias values according to Levenberg-Marquardt optimization. It is generally the fastest backpropagation algorithm in the toolbox, and therefore recommended as a first-choice algorithm, though it does require more memory than other methods [MATLAB®, 2016f].

The backpropagation learning process sends the input values forward through the network and computes the difference between the calculated output and the corresponding desired target. Based on this difference the weights between the neurons update up to the moment the computed output values best approximate the desired target values [Rojas, 1996]. Two other optimization methods were considered; the scaled conjugate gradient backpropagation and the Bayesian regularization backpropagation. The scaled conjugate gradient is more suitable in low memory situation and the Bayesian takes a longer, but could be better for challenging problems [MATLAB®, 2016f]. Both were less efficient.

The network will stop the training when 35 validation checks are reached, this means that the error in the validation increased during these 35 iterations. The performance, calculated by the mean squared error, gives an indication of how well the prediction was made. The interface of the MATLAB® toolbox that was used to create the neural network is shown in the Appendix.

4.2.4. Training, Validation & Testing

The historical data is used for training, validation and testing. In doing so the complete data set is divided into these 3 segments. The training set is the data set that is used for training, thus to adjust the weights on the neural network. The validation set is used to minimize overfitting. The weights of the network are not adjusted with this data set, it is just verifying that any increase in accuracy over the training data set actually yields an increase in accuracy over a data set that has not been seen by the network before, or at least the network hasn't trained on. If the accuracy over the training data set increases, but the accuracy over then validation data set stays the same or decreases, then overfitting occurs and training should be stopped. Thus the validation set indicates whether the learning of the network can be finished. At last the testing set is the data set that is used only for testing the final solution in order to confirm the actual predictive power of the network on a data set that was not used in neither the training and the validation process [Kooijman, 2011]. In this case 70% of the data will be used for training, 15% for validation and 15% for testing.

There are several different ways in dividing the data set in these ratios. Division could be done in blocks, randomly or interleaved. The division by interleaved data points is chosen as the data sets are now equally divided during production time, which will give the best result. Division in blocks has the training data in the first 70% of points, the validation in the next 15% and the testing set in the last 15% of data points. This results in training in the decline phase and not in the liquid loading phase and training can be misleading. Random division could end up in different performances for each training, which is not wished for. The way the interleaved is divided over the data set is shown in Figure 4.9.

4.2.5. Hidden Layers & Hidden Neurons

The multi-layer neural network can contain only one hidden layer, which can give a good result, but more hidden layers can give a better fit, though the training takes more time. The number of hidden neurons in the hidden layer has an influence on the generalization of the network. Too many hidden neurons create over-fitting, which causes the network to memorize results instead of generalizing. Working with too few neurons will result in a network that is not able to learn the correct input and output algorithm. An optimum number of hidden neurons is therefore essential [Kooijman, 2011]. From literature several rules of thumb were inspected. The first one is a rough approximation by the geometric pyramid rule. For a three layered network (meaning an input, hidden and output layer) with n input neurons and m output neurons, the hidden layer would have $\sqrt{(n \times m)}$ neurons [Kaastra and Boyd, 1995]. Another rule of thumb's states that the number of hidden-layer neurons should be about 75% of the input variables, while again another one suggest that the number of hidden-layer neurons should be approximately 50% of the total number of input and output variables [Chakra et al., 2013]. From these rules one can conclude that it is assumed that the training set is at least twice as large as the number of weights. If this is not the case, then these rules of thumb can quickly lead to overfitted models since the number of hidden neurons is directly dependent on the number of input neurons and thus the weights. Using the first rule of thumb, training the neural network was started and modified when needed.

The options considered when doing a forecast vary from how many months are used for the prediction and how many months can be predicted. As many as possible is the aim, but too many will give a forecast of already predicted data. This will give a larger error and will not help improve the performance. Thus a optimum number of months is therefore essential too.

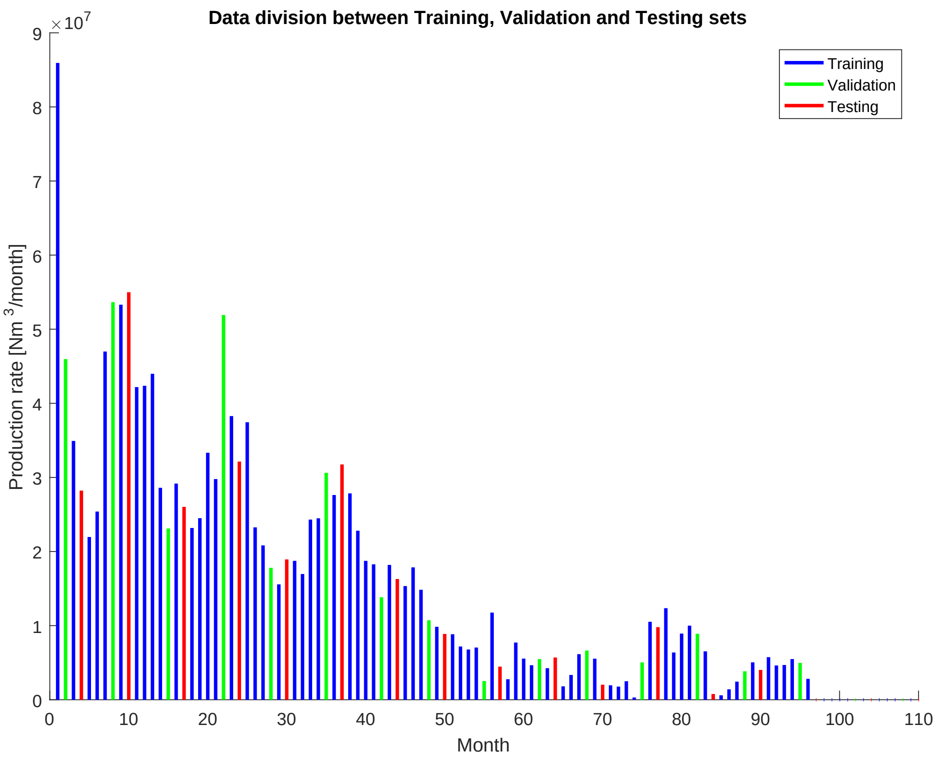


Figure 4.9: The production data is divided interleaved among training, validation and testing sets.

4.2.6. Activation Function

The activation function is applied to the weighted input of a neuron to produce the final output. Each network only has one activation function, which is used for all hidden neurons in that network. The activation function is important when creating a neural networks for two reasons. Firstly, if there were no activation functions, the whole neural network can be reduced to a group of linear function of the network input. So, without activation functions, a neural network can not learn non-linear relationships. And secondly, each neuron can be seen as recognizing a certain feature, with an activation of zero indicating the absence of that feature [Kriesel, 2005].

There are three different activation functions that can be used in the MATLAB® toolbox when training the neural network; the logistic transfer function (logsig), the hyperbolic tangent (tanh or tansig) and the linear function (purelin) [MATLAB®, 2016b], Figure 4.10.

For the hidden layers, a nonlinear transfer function must be used, thus the purelin function is not an option. For the error backpropagation that is used in this research an activation function that is differentiable, smooth, monotonic, and bounded must be used. Both the logsig and tanh fit the requirements.

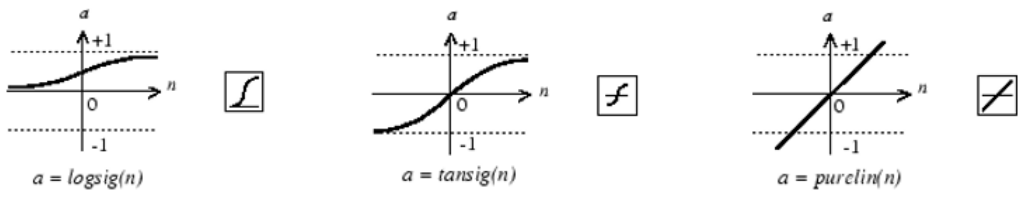


Figure 4.10: Three activation functions; logsig, tansig and purelin.

4.3. Predicting liquid loading

When an accurate forecast of the production rate is made, then the liquid loading moment can be predicted. Coleman criterion was introduced to indicate the onset of liquid loading. The table showing the Coleman rates for each well is shown in the Appendix. When the predicted flow rate falls below the Coleman rate, the liquid loading alerter is triggered. In the month the alerter is triggered the system gives the expected month in which liquid loading happens. Every month a new prediction can be made until it finds the true value for the liquid loading moment.

The results of the different neural nets, including the variable training parameters, the result of the most accurate forecast and the liquid loading prediction are described in the next chapter.

5 Results

5.1. Production Analysis

5.1.1. Influence of Field Size on the Liquid Loading Moment

A study was conducted on the size of the field and the moment when a well is liquid loading. Figure 5.1 shows a column chart. The months in which a well is liquid loading is shown on the x-axis and the y-axis shows how many wells fall in this range. The three colors show a small, medium and large field. A small field means a GIIP lower than 10 bcm, a medium has a GIIP between 10 and 20 bcm, and a GIIP larger than 20 bcm is a large field. As expected the larger the GIIP of a field, the more a well can produce and the longer it takes for a well to start liquid loading. Off course it differs per well how much a well can produce and the quality of the reservoir has a large influence, but the trend can be seen for these 64 wells.

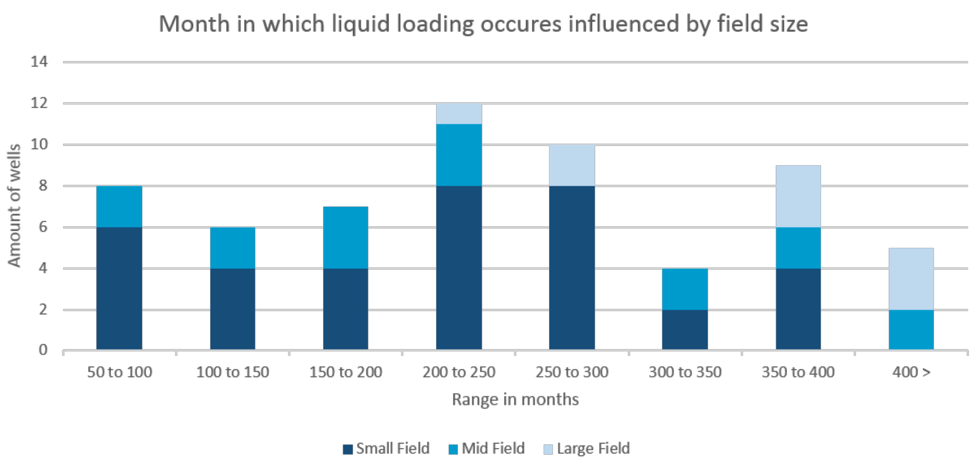


Figure 5.1: The influence of field size on the liquid loading moment.

5.1.2. Volume Gain by Velocity String and Foam

The volume gain of all wells was investigated. To understand the effectiveness of the different EoFL techniques, a velocity string or foam (both batch and continuous) are shown in the column chart 5.2. From this chart it can be concluded that for these wells the gain is expected to be 10 to 50 million m^3 as about 45% lies in this range. However other wells produce more than 100 or even 200 million m^3. The wide range is due to the dependency on well and environmental conditions. It can also be seen that, as expected, continuous foam and velocity string will give a higher cumulative production than batch foam. Two foam projects are found to be very effective with a cumulative production higher than 200 million m^3.

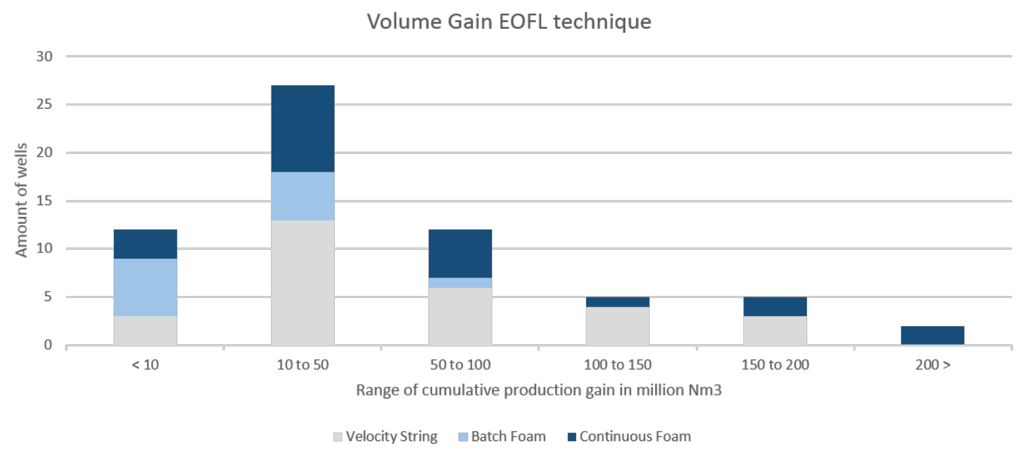


Figure 5.2: The volume gain of 64 wells by a velocity string, batch foam or continuous foam.

5.1.3. Actual Volume Gain compared to Expected Values

From the database the volume gains expected by the operator were gathered. An interesting insight was found when comparing it to the cumulative production that was actually gained by an EoFL technique. For velocity strings one can see from Figure 5.3 that the expected value determined by operators before the work-over was more than the volume actually gained. Several wells are still flowing, which are therefore still uncertain if they rise above expectation. However from the graph this is not likely. Half of the foam wells however produce better than expected. This is useful insight as it means that foam projects are very effective. As they are cheaper than velocity strings, foam could be an option first. Note that when the well is dead implementing a velocity string is properly more effective, as the foam may not work when there is too much water in the well.

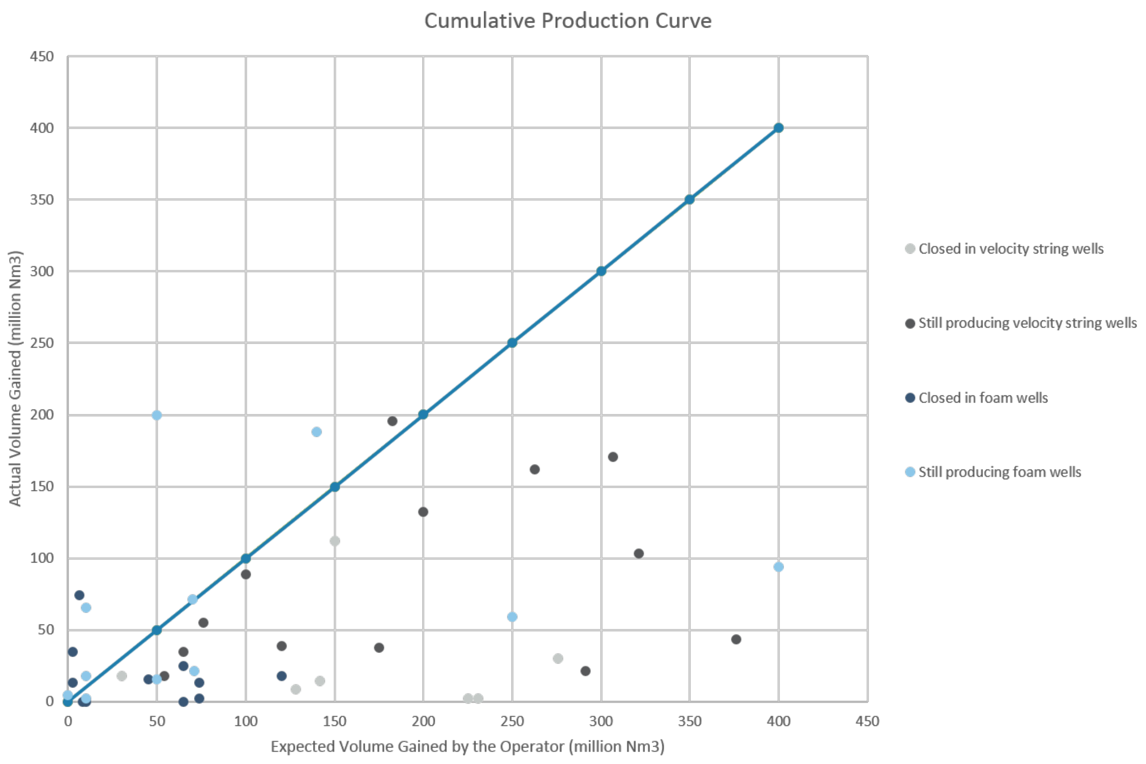


Figure 5.3: The volume gain of the actual gain compared to the expectations of the operator before stimulation of the well.

5.1.4. TNO Deliquification Tool

With the TNO tool an estimation was conducted on the gain by an EoFL technique. The tool calculates from the parameters described in the previous section the production gain by a certain EoFL technique and the amount that would have been produced without mitigation. 15 wells were studied by the tool. 11 wells had a velocity string installed and in 4 wells foam was injected. The result can be seen in Figure 5.4. In this figure the cumulative production is shown for all 15 wells. All wells have 3 bars representing the cumulative production calculated by the tool, the actual volume gain up til today and the expectation of the operator before mitigation. From these 15 wells it can be concluded that the simulations from both the operator and the TNO deliquification tool predict a higher cumulative gas production than that was actually produced by an EoFL technique. The reasons of this difference are explained in the next chapter.

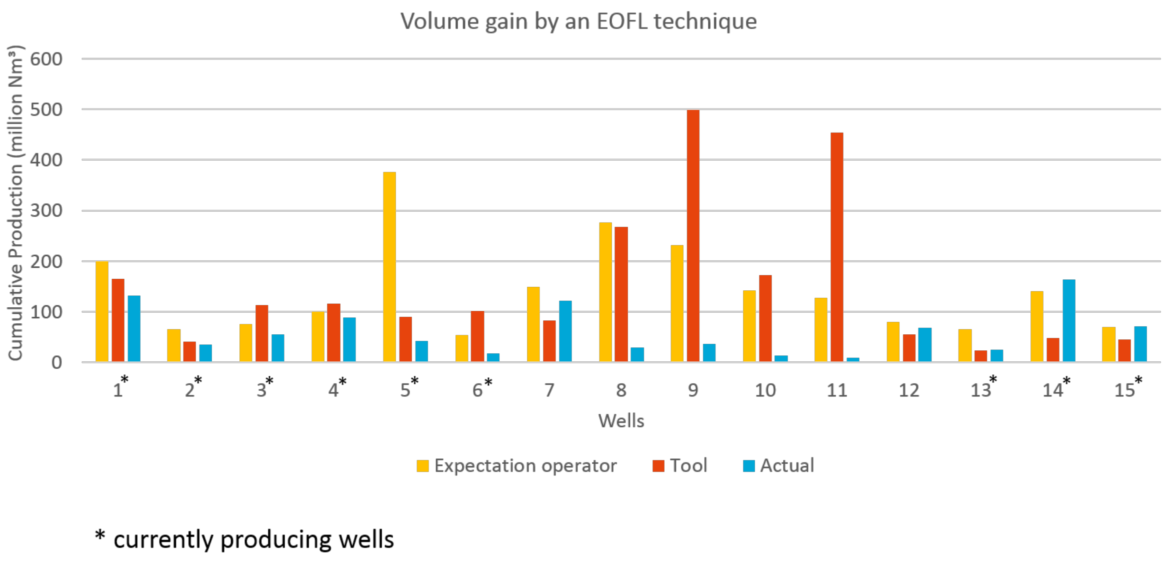


Figure 5.4: The volume gain estimated by the TNO deliquification tool, the actual gained volume and the expectations of the operator before using a velocity string (first 11 wells) or foam (last 4 wells).

5.1.5. Sizing Tubing

A difficult matter is choosing the tubing size. The deliquification tool shows the cumulative gain versus the producing years for different velocity string sizes. Two wells were considered and the tubing size was varied between $2\frac{3}{8}$, $2\frac{7}{8}$, $3\frac{1}{2}$ and $4\frac{1}{2}$. The first well is shown in Figure 5.5. The smaller the velocity string the higher the cumulative gain and vice versa. This well had a velocity string installed with a tubing size of $3\frac{1}{2}$ inch in diameter, representing the second blue graph from left. This size was indeed the best option as it gives the highest cumulative production in the least amount of years. A smaller tubing size gives you a little larger gain, but with only a maximum of 10 million Nm^3 in about 2 years. Economically wise it is best to make profit as fast as possible.

In figure 5.6 the next well is shown. This well had a velocity string installed with a tubing size of $2\frac{3}{8}$ inch in diameter, representing the fourth blue graph from left. This tubing size gives the highest return in less than 3 years. Therefore this is a very suitable size. However as the gain is not even 5 million Nm^3 less when producing 2 years with a velocity string size of $2\frac{7}{8}$, this would also have been a suitable candidate.

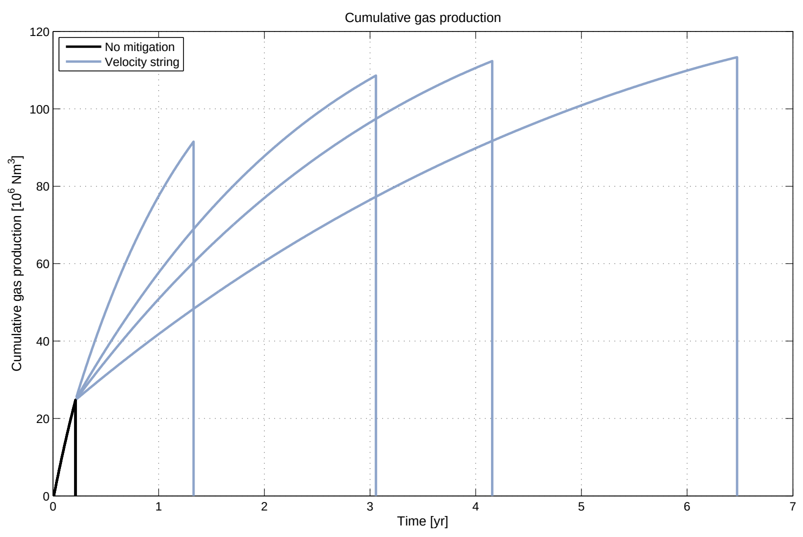


Figure 5.5: The influence of the velocity string diameter on the cumulative production and the producing years. The blue graphs show the volume gain by the different sizes of velocity strings ($2\frac{3}{8}$, $2\frac{7}{8}$, $3\frac{1}{2}$ and $4\frac{1}{2}$) installed. Left to right representing the largest to the smallest diameter. The $3\frac{1}{2}$ showing the highest profit in the least amount of time.

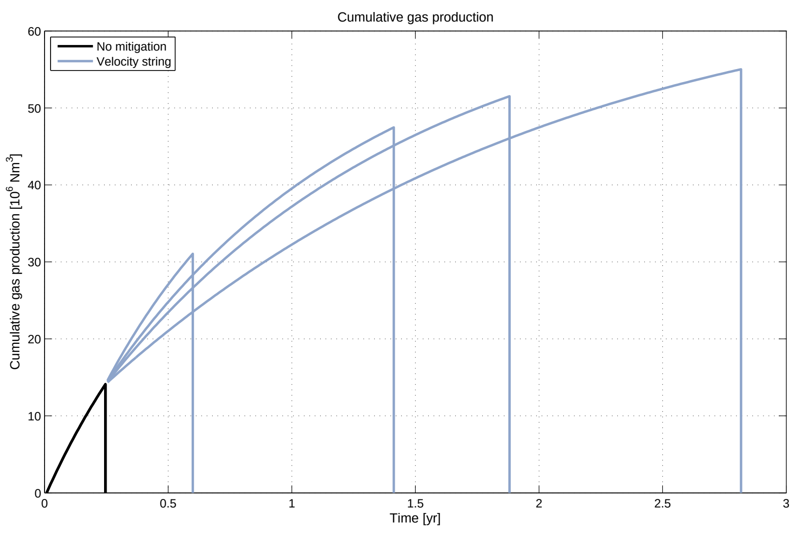


Figure 5.6: The influence of the velocity string diameter on the cumulative production and the producing years. The blue graphs show the volume gain by the different sizes of velocity strings ($2\frac{3}{8}$, $2\frac{7}{8}$, $3\frac{1}{2}$ and $4\frac{1}{2}$) installed. Left to right representing the largest to the smallest diameter. The $2\frac{3}{8}$ and $2\frac{7}{8}$ are showing the highest profit.

5.1.6. Some Examples

To illustrate the matter into further detail 2 wells are described in the succeeding sections. The first one has a velocity string implemented and the second one has had foam injection.

Well 3

A velocity string was implemented in well 3 with a diameter of 3.5 inch. The left Figure 5.7 shows the TPC curve of the well and the right Figure 5.8 shows the TPC after the installation of a velocity string. The IPR curve does not change as reservoir pressure stays the same, but the TPC curve has moved to the left due to the smaller tubing size. The well has a lower critical rate than without the velocity string and therefore the well can produce for a longer period in time.

The TPC curve already looks promising. Next the simulation is run for well 3. As expected the

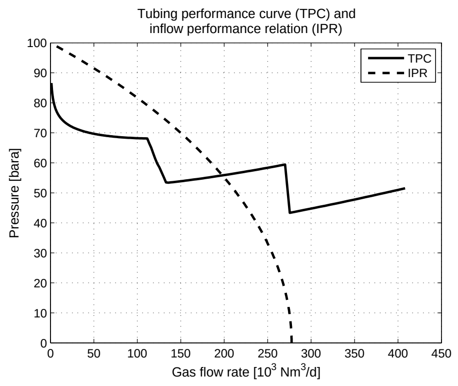


Figure 5.7: The TPC and IPR curve for well 3 without mitigation.

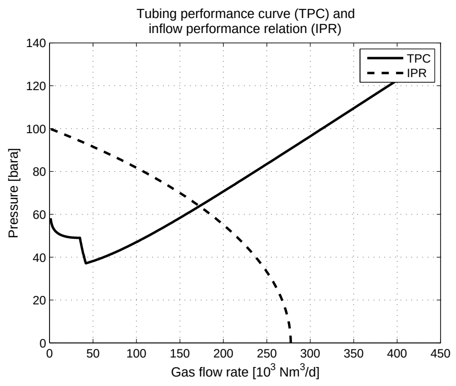


Figure 5.8: The TPC and IPR curve for well 3 with a velocity string.

abandonment pressure is lower for a smaller tubing size and the critical flow rate is of lower value. This can be seen in Figure 5.9, where the gas rate is plotted versus the reservoir pressure. Without mitigation the abandonment pressure is around 86 bar and the flow rate 130, 000 m^3 per day and after installing a velocity string the abandonment pressure is around 46 bar and the critical flow rate at 20, 000 m^3 per day. The graph on the right side, Figure 5.10 shows the years one can keep producing with the mitigation technique versus the reservoir pressure. The expected producing time is 2.5 years more with a velocity string.

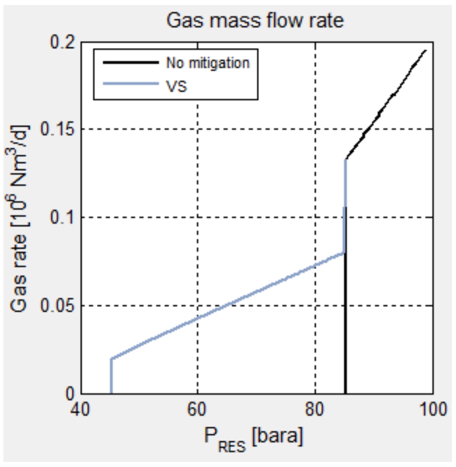


Figure 5.9: The flow rate plotted versus the reservoir pressure for well 3.

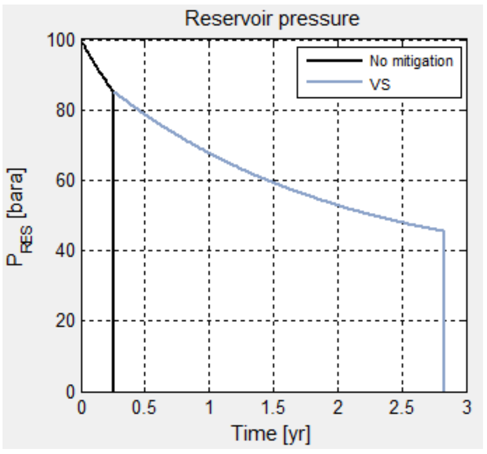


Figure 5.10: The reservoir pressure plotted versus the producing years for well 3.

The operator is most interested in the larger amount of gas that is produced due to a velocity string. The tool shows a successful operation with a volume gain of 41 million m^3. The volume gain can be seen in Figure 5.11, where the cumulative gas production in million is plotted versus the time in years.

The production data of the well is compared to the tool's simulation. Both graphs can be seen in Figure 5.12. The tool curve is the ideal case, while the actual curve fluctuates a lot. Still the curves are very alike in values and it proves an accurate simulation of the TNO tool. It seems however that, as was seen before, there is a trend of the tool predicting a bit higher than the actual gain. This issue is mainly due to well problems and will be discussed in more detail in the next chapter.

It can be concluded that the velocity string proved to be effective for this well. The well produced 34.7 million m^3 in 2.8 years, and is still producing. The expected gain by the operator was 65 million m^3, but it is not surprising that the tool predicts a higher volume gain than the operator as the well is still in production. The investment made was about 1.5 million and the NPV was 5.3 million euro's. Thus a nice profit is already made.

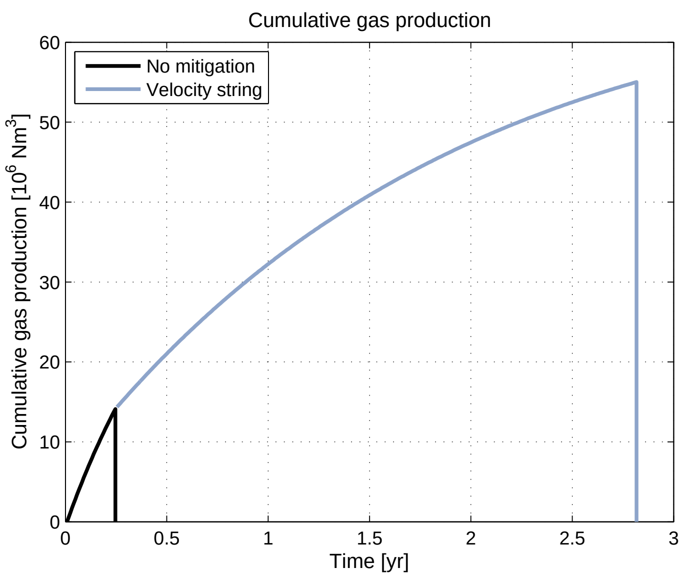


Figure 5.11: The cumulative production versus the producing years is simulated with and without mitigation for well 3.

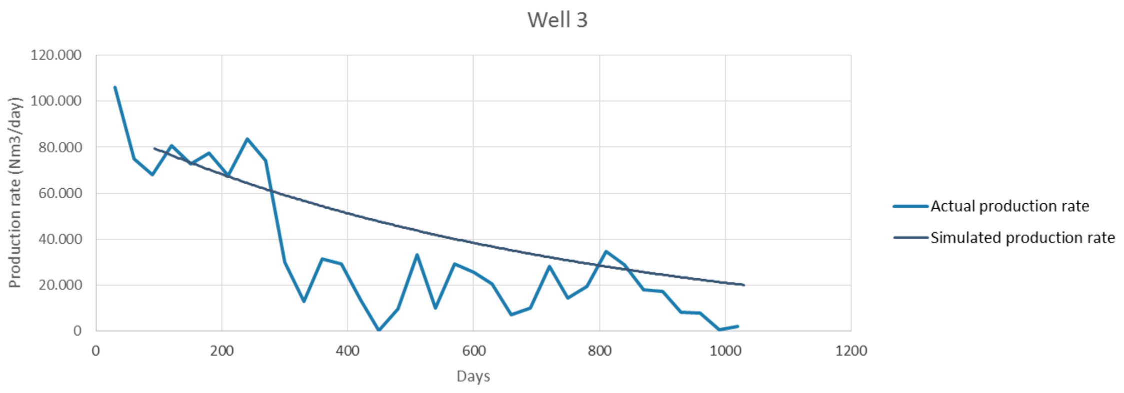


Figure 5.12: The production rate per day simulated with velocity string by the TNO deliquification tool and the actual production rate per day after installing a velocity string.

Well 14

The next well considered is a foam well. Foam was injected about two years ago and the well is still producing. The TPC and IPR curve are shown in Figure 5.13. Unfortunately it is not possible to make a new TPC curve after foam injection with the tool.

Next the simulation is run for well 14. As expected the abandonment pressure is lower after the foam is injected and the critical flow rate has decreased. This can be seen in Figure 5.14, where the gas rate is plotted versus the reservoir pressure. Without mitigation the abandonment pressure is around 28 bar and the minimum flow rate $20,000$ m^3 per day. The abandonment pressure is around 25 bar and the critical flow rate $10,000$ m^3 per day after foam injection. Figure 5.15 shows the years the well can keep producing with the mitigation technique versus the reservoir pressure. The expected producing time is 2.5 years. But in this case the well would also have produced 1.5 more years without foam. It seems that foam is injected a little too early.

When looking at the cumulative gas gained by foam injection in Figure 5.16, the gain is only 50 million m^3 higher than without foam. This is not a lot considering the long producing time of a year. Without foam it produces about 250 million m^3 in 1.5 years. Off course production will be less when the well has been producing for a long time, but the operator could have waited here. After 1 year the tool shows a successful operation of foam injection with a volume gain of exactly 47.8 million m^3.

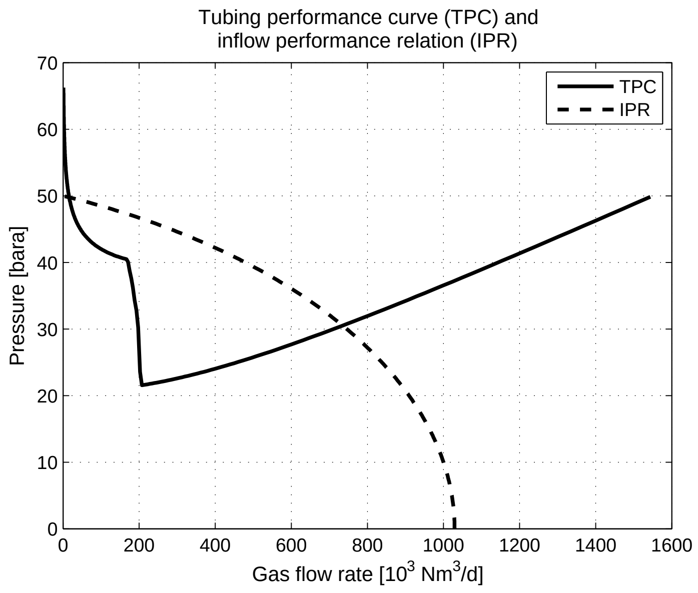


Figure 5.13: The TPC and IPR curve for well 14 without mitigation.

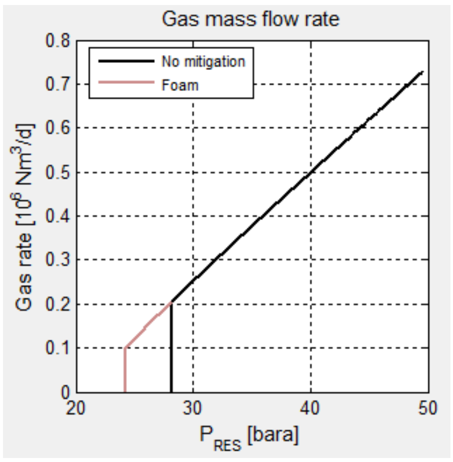


Figure 5.14: The flow rate plotted versus the reservoir pressure for well 14.

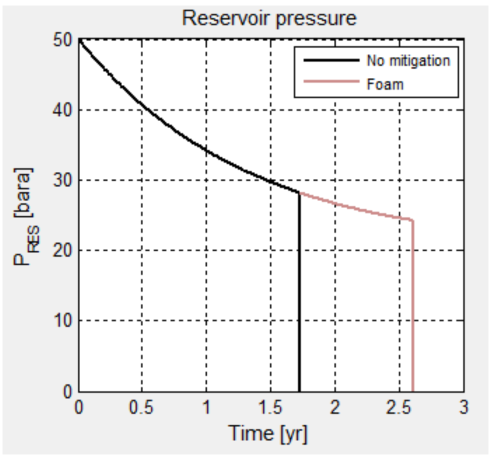


Figure 5.15: The reservoir pressure plotted versus the producing years for well 14.

The production data of the well is compared to the tool's simulation. This comparison confirms the presumption of the premature injection of foam. Figure 5.17 shows the production profile simulated by the tool and the actual production curve. The tool's prediction consists of two parts; the gray line shows the curve without mitigation and the dark blue line shows the production rate with foam injection. The actual production rate is less than half the rate the tool predicts. The conditions inserted into the tool give a high production, but in the field this is not the case. It may be that the LGR was higher or other conditions were not accurate. If this is the case, it may be the reason for the tool to predict foam injection to be useful at a later moment than the operator does. The well produces in total a 164 million after foam is injected. This gain is represented in the positive NPV. However the operator expected a cumulative production of 310 in these two and a half years. The well is still producing, so the production will still increase, but from the simulation it can be said that foam injection was injected too early or the tool predicts too optimistic with the given parameters.

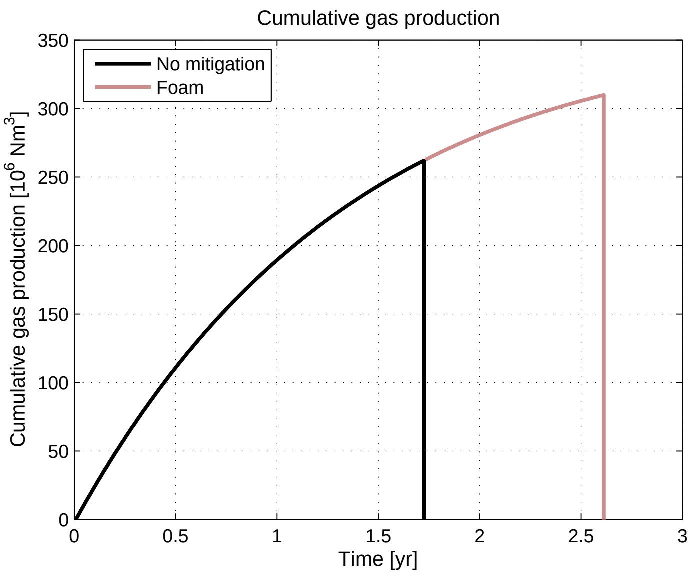


Figure 5.16: The cumulative production versus the producing years is simulated with and without mitigation for well 14.

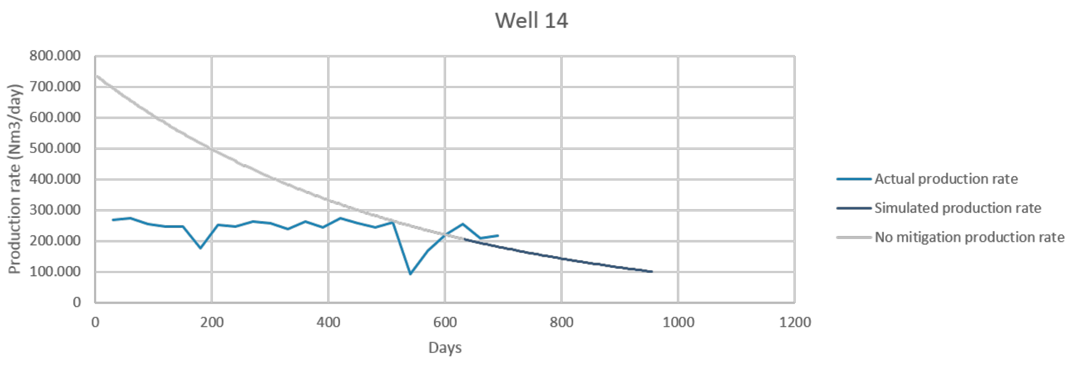


Figure 5.17: The production rate per day simulated with and without mitigation by the TNO deliquification tool and the actual production rate after foam injection.

5.1.7. Net Present Value

In this section the net present value (NPV) is calculated for the 15 wells that are also evaluated with the tool. The net present value is calculated by Equation 5.1 with an discount rate of 8%.

$$NPV = \sum_{t=0}^{N} \frac{R_t}{(1+i)^t} \tag{5.1}$$

where N is the total number of periods, t the time of the cash flow, i is the discount rate and R_t is the net cash flow at time t.

From EBN the gas price in €/MWh is received per month. The heating value is found for each well, ranging from 36.7 to 40.5 MJ/m^3. Note that 1 kWh = 3.6 MJ. The NPV can be calculated with the actual gas rates for the wells that have stopped production, but for the still producing wells the TNO's prediction of future gas rates had to be included in the calculation. In Figure 5.18 the NPV's per well is shown. 9 mitigation projects were successful and 6 of them were not.

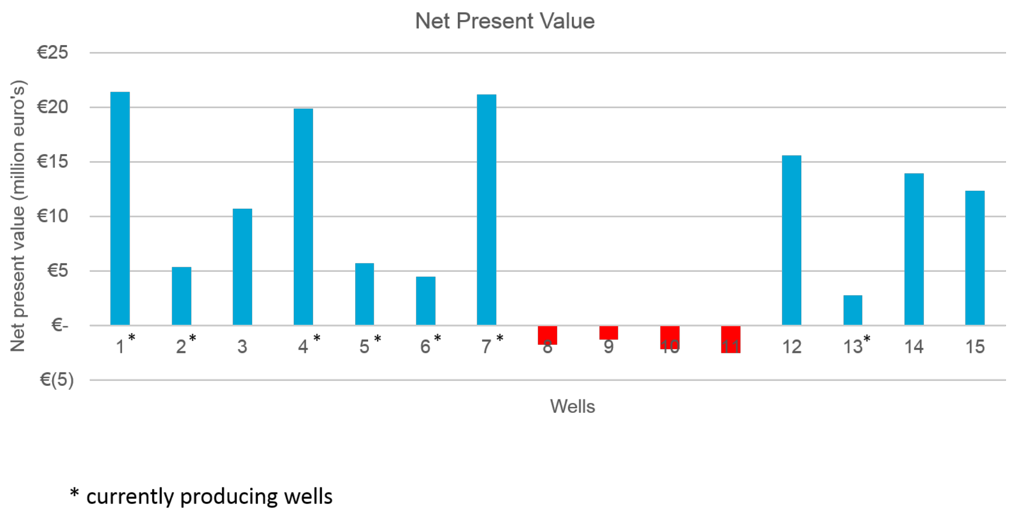


Figure 5.18: The net present value calculated for the 15 wells that were studied. The first 11 wells had a velocity string installed and for the last 4 wells the profit was made by injecting foam.

5.2. Choosing the best Neural Network

The quality of the network performance depends on input data, the number of input neurons, hidden neurons, the activation function and months being predicted. The only way to determine the right parameters and the best performance is the method of trial and error.

5.2.1. Auto-correlation

The auto-correlation shows which month is most important in the prediction of the month that is predicted. This could be the month just previous to the month predicted, which is the case for the two wells shown in Figure 5.19 and Figure 5.20. The value on the y-axis shows the influence the month has to the predicted month (the first bar). It can also happen that the months in the same period the year before are most important. For example when a lot of maintenance done in the summer months, the winter months the year before are in that case more similar and thus have greater influence to the month predicted in the winter the year after. This is shown in Figure 5.21. From all 40 wells it can be concluded that the first option is the case for 95% of the wells, therefore seasonal dependency is not taken into account when training NN.

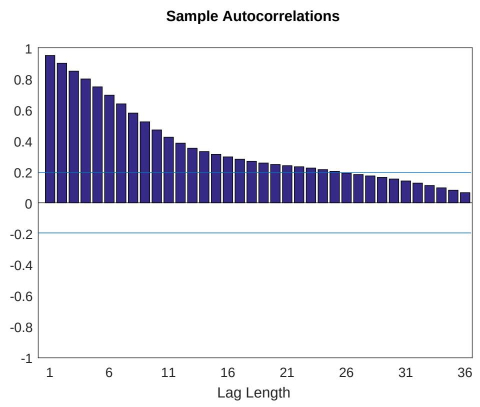


Figure 5.19: The autocorrelation plot shows a well where the month being predicted is mostly depended on its previous month, then the month before that etc. The graphs shows a well where 2 years influence the prediction.

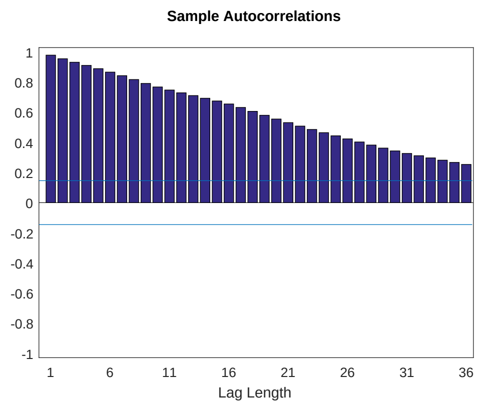


Figure 5.20: The autocorrelation plot shows a well where the month being predicted is mostly depended on its previous month, then the month before that etc. The graph shows a well where more than 3 years influence the prediction.

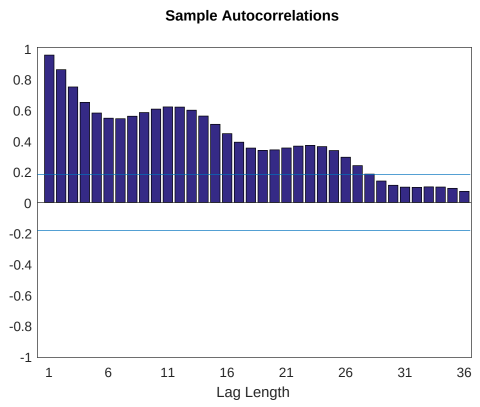


Figure 5.21: The autocorrelation plot shows a well of which the month being predicted is seasonally dependent.

The auto-correlation plots above show how many historical months should be taken into account for the training process. Figure 5.19 and Figure 5.21 show that for these wells 2 years are important when forecasting the gas rate for a certain month, while Figure 5.20 shows that 3 years are of importance.

So from the auto-correlation plots for all wells it was concluded that for about 75% of the wells the prediction was influenced by 2 to 3 years of the historical data while for the other wells only 1 or 2 years should be taken into account for. More months would have a negative effect on the prediction. The trainings were started with an input of 3 years and compared to the performance of the trainings including an input of 2 years.

5.2.2. Designing Neural Networks

The hidden layer(s) provide the network with its ability to generalize. A neural network with one hidden layer with a sufficient number of hidden neurons should be capable of approximating any continuous function. In practice, networks with one or two hidden layers are widely used an have performed very well [Kaastra and Boyd, 1995]. Because of this, and the fact that an increasing the number of hidden layers also increases the computation time and the danger of overfitting, ANN was trained with 1 and 2 layers.

The hidden neurons are varied in one or two hidden layers. As a start the number of neurons trained were 5, 10, 15, 20, 25, 30 and 40. From this the best network was chosen and more trainings were performed with 50, 60, 75, 100 and 150 neurons. The optimum number of neurons is determined on the minimum value of the best validation performance. In other words the lowest mean squared error. Note that every training is different and one training could have a lucky good or bad performance, therefore each training with a certain amount of neurons was done 5 times and the one with the lowest mean squared error and thus the best performance was taken.

The activation function for each neuron gives the weight to the input signal. As described in the previous chapter there are two activation function that can be used for this network. The tanh function and the logsig function.

All these parameters were taken into account when training and varied along the process. Figure 5.22 shows the performance plot of the trainings, including four curves. These curves include the 1 and 2 layers that were trained, the neurons trained ranged from 5 to 40 neurons and both activation functions. First of all it can be concluded from the graph that a 2 layered network has a better performance. This is not surprising as 2 hidden layers can capture more non-linearity and handle more complex data. The 1-layered-logsig curve has a surprisingly low MSE for 15 neurons, but climbs back up with a higher number of neurons. It may just have been a lucky shot. For a higher number of neurons the tansig performs better than the logsig curve. Thus tansig is preferred. Adding to this is the fact that the logistic function will generate values close to 0 if the argument of the function is substantially negative. Thus, the output of this very hidden neuron will be close to zero, and thereby lowering the learning rate for all subsequent weights. This means that it will almost stop learning. The tanh function, however, will in the same situation generate a value close to -1, and thus will maintain learning.

It can be concluded that the 2-layered-tansig-network is the best network to take. Especially the network with 20 neurons or more showed a lower MSE than the other three curves. From this graph the network with 30 neurons showed the best performance. However more trainings were performed with a higher number of neurons and the performance can be seen in Figure 5.23. The MSE even lowered for these trainings. The networks with 75 neurons and 100 neurons show similar results, but the network with 75 has a slightly lower MSE with $5.9484e^{-0.5}$ than the one of 100 with an MSE of$5.9498e^{-0.5}$. One trial was also done with a 150 neurons, MSE however did not improve ($6.1653e^{-0.5}$) and as the training takes 20 hours, more trainings with 150 neurons was not done. It seems that a network with more neurons does not give a improvement in MSE and a training that takes less time is preferred. Note that if more trainings than 5 trials are conducted per set of parameters the more accurate the result will be and it may be that the network with a 100 or 150 neurons were better after all, but a limit has to be drawn time wise and of the capacity of the consumer computer. This is discussed further in the next chapter. Next training was performed with a network with 75 neurons that has 2 years as input data. The results were compared to the one with 3 years, but did not show an improvement as it had a MSE of $7.3e^{-0.5}$. Thus to conclude: 2-layered-tansig-network with 75 neurons trained on 36 months of historic data was chosen as the best network.

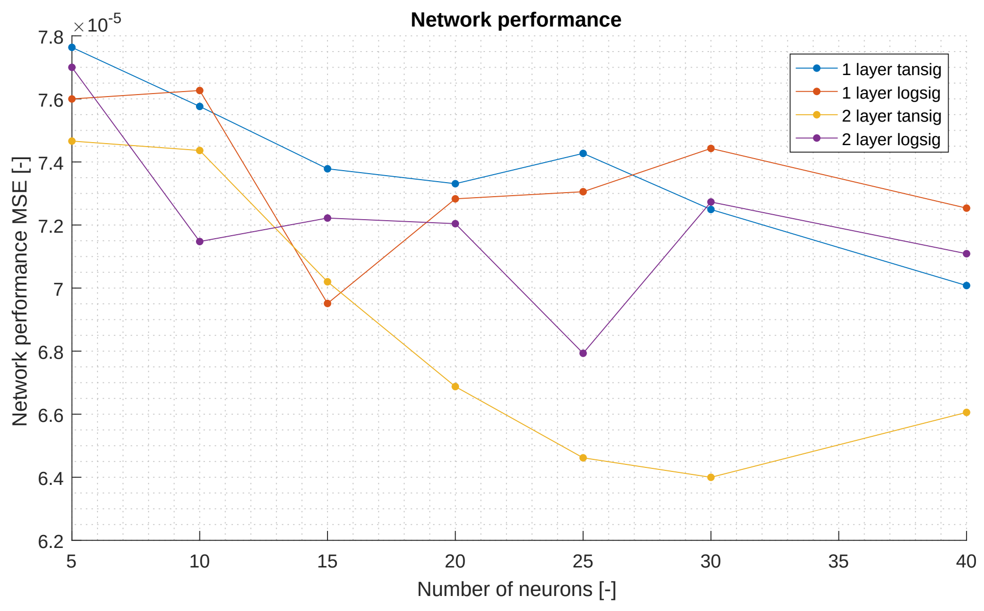


Figure 5.22: Sensitivity analysis based on the number of layers, the activation function and the number of neurons is shown. The 2-layered-tansig-network showing the lowest MSE.

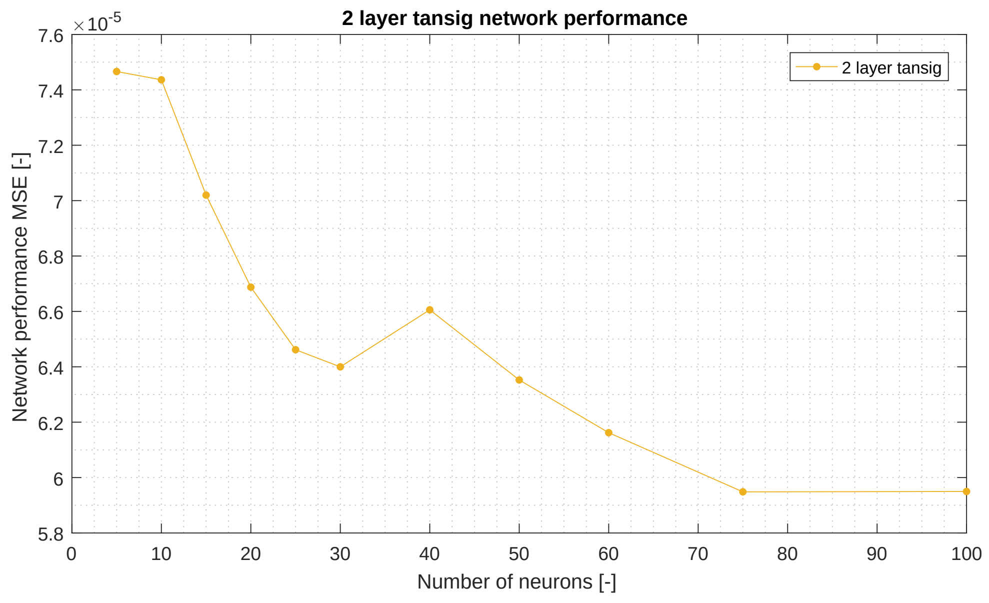


Figure 5.23: Sensitivity analysis based on a 2-layered-tansig-network. 75 neurons show the lowest MSE with a value of $5.9484e^{-0.5}$ and thus the best performance.

5.2.3. Performance and Regression Analysis

The performance plot shows three curves: a training, validation and testing curve. One to fit the models and make changes, one to compare models, and one to demonstrate the effectiveness of that model. When training progresses the curves naturally drop with each iteration. When the validation error stops dropping and starts to increase, while the training error is steadily decreasing, then a situation of overfitting may have occurred. The stopping criterion of the training is before overfitting starts and is the moment where the error of the validation curve reaches its minimum error [Kooijman, 2011].

The training of the network can be considered reasonable when the plots indicate the following:

- The mean-square error is small
- The test set error and validation set error have similar characteristics
- No significant overfitting has occurred

With the regression plot the prediction performance of the neural network is assessed. The plot contains the training, validation, testing curves and a plot including all curves. The dashed line indicates the perfect result where the output is similar with the target. The solid line represents the best possible fit of the data between the output and the target. The mean squared error gives an indication of the linear relationship between the output and target where R $= 1$ indicating an exact relationship and R $= 0$ that there is no linear relationship between output and target [MATLAB®, 2016e].

As said before, the network with the optimum number of neurons will be used here to explain the performance and the regression plot. The performance plot is shown in Figure 5.24 and its regression plot is shown below the performance plot in Figure 5.25.

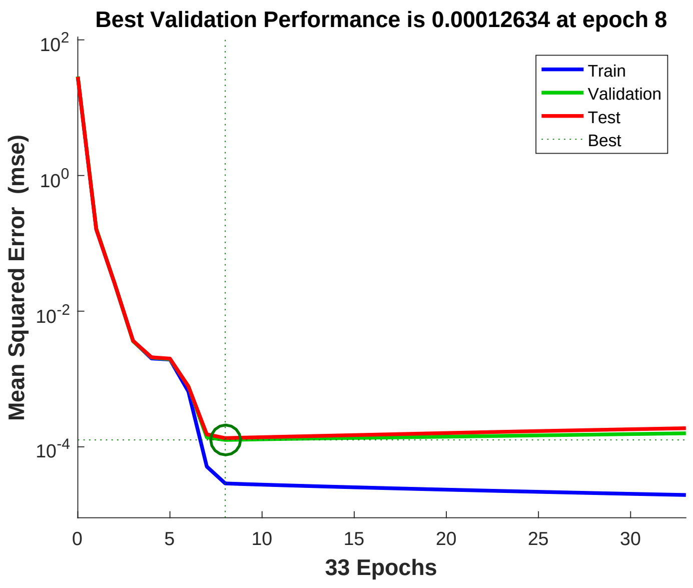


Figure 5.24: The best validation performance plot for the network with 75 neurons. The low mean squared error and the training, validation and testing curves show a good training performance.

This figure does not indicate any major problems with the training. The MSE is small, the validation and test curves are very similar and the model has stopped before overfitting occurred, namely at epoch 8, before the validation curve increases.

The regression plot shows a good fit. The training, validation and test regressions are above 0.99. The model is based on many data points, so many that the Y=T curve is not visible anymore. A lot of data means that the result is very reliable.

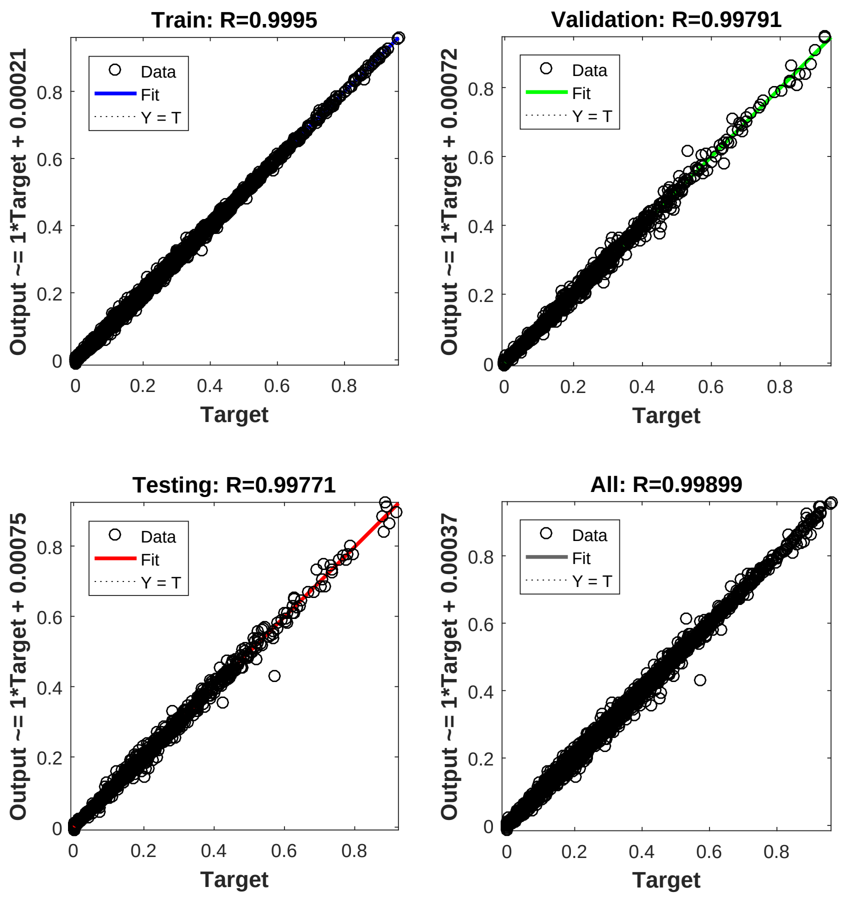


Figure 5.25: The regression plot for the neural network with 75 neurons.

5.2.4. Logarithm & Log Return

The network was trained with the logarithmic data. Note that 3 trainings were done and that the validation checks were reduced to 10 checks due to the long computation time. The performance curve in Figure 5.26 shows that the training was not as good as without logarithmic data. The MSE can not be compared because of different input data, but the validation and test set error are relatively far apart. Also both curves fluctuate a lot before training is stopped, this could mean overfitting has already occurred. Next when looking at the regression plot in Figure 5.27 one can also conclude training is less good than without logarithmic data. The regression is still above 0.99, but there a many points around 0 and the relation between the input and target is worst than the previous example. Thus the network is insufficient to be used as prediction model. As an example, two wells of which the forecast of the flow rate was performed by a NN that was trained on logarithmic data can be seen in the Appendix.

Next to the network trained on logarithmic data, the log return network did not show any improvements. The validation performance plot and the regression plot are shown in the Appendix.

5.3. Production Forecast

The optimum neural network has been chosen. It contains 2 layers, 75 neurons with the tanh functions and it has 36 months as historical production rates taken into account. The forecast can now be done. How well is the performance of the forecast? Can one use it to predict future flow rates and above all the liquid loading moment? In the next four figures several example wells are shown. In blue the raw data and in red the filtered data. The moment of prediction starts where the dotted line begins. The dotted green line is the predicted flow rates predicting 1 year ahead.

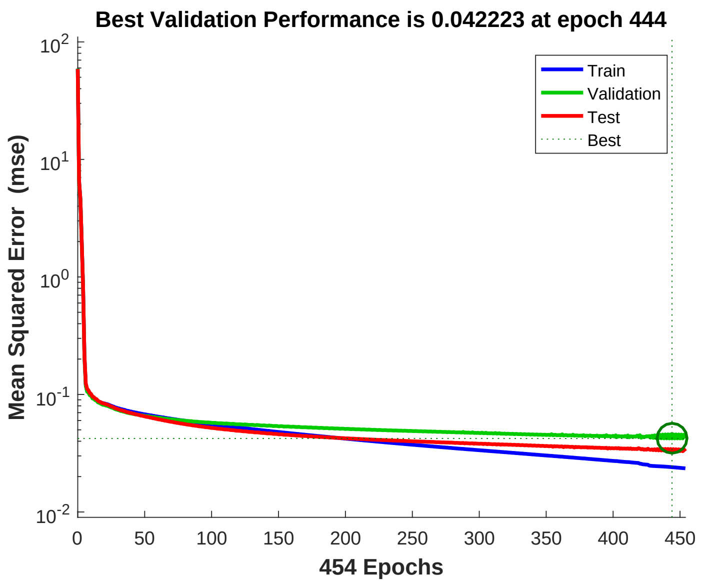


Figure 5.26: The validation plot for the logarithmic network.

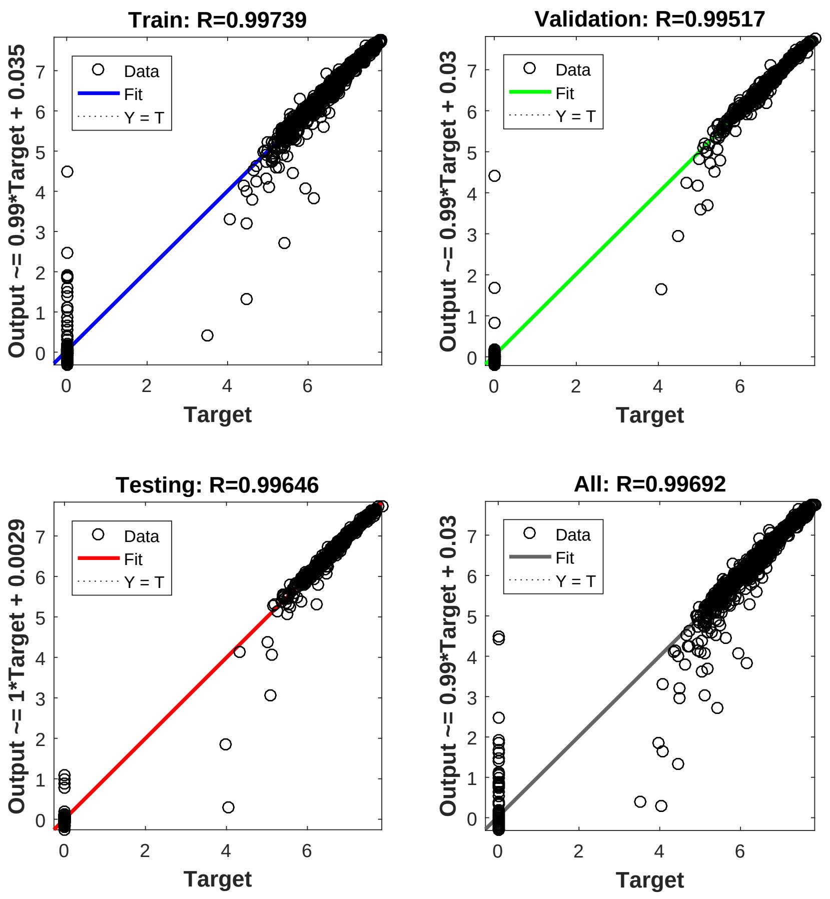


Figure 5.27: The regression plot for the logarithmic network.

Figure 5.28 shows the prediction of well 6. One can see that the production rate is predicted very well. The error between the the actual rate and the predicted rate is only 4%. Another example is shown in Figure 5.29. This well has a very accurate prediction, with only a 1% error. One might think though that this pattern is more easy for the NN to learn, as the prediction is quite a straight line. But the thought is blown away when looking at Figure 5.30. The production profile shows a very fluctuating production profile. The predicted profile shows a similar fluctuation and an error of again only 4% is received. The trained network proves to be very good and can predict the production rate very accurately.

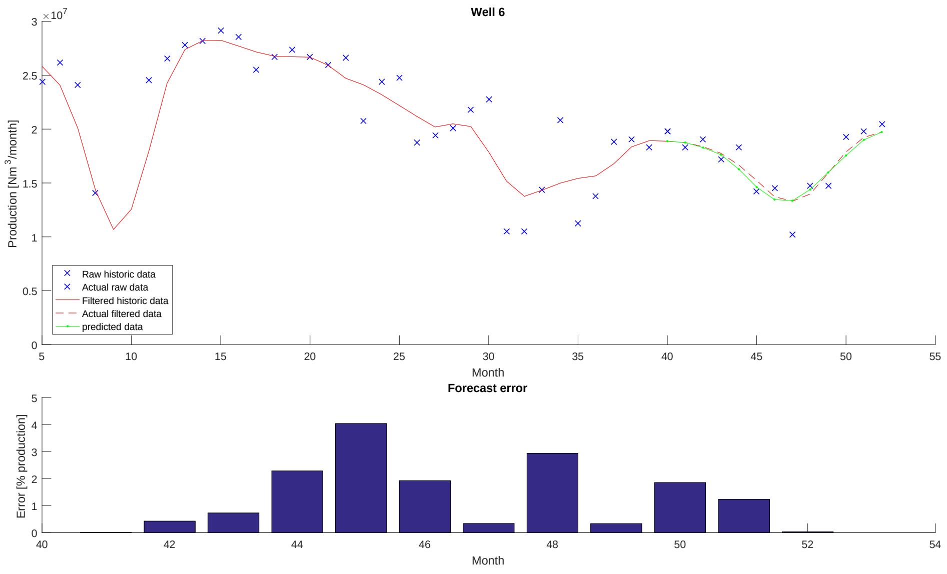


Figure 5.28: The actual and the predicted flow rate by ANN for well 6.

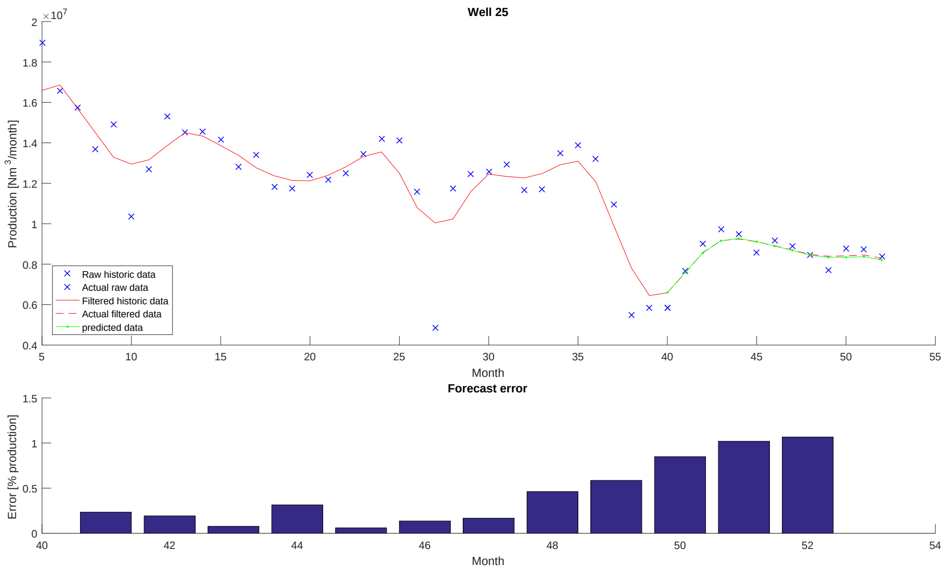


Figure 5.29: The actual and the predicted flow rate by ANN for well 25.

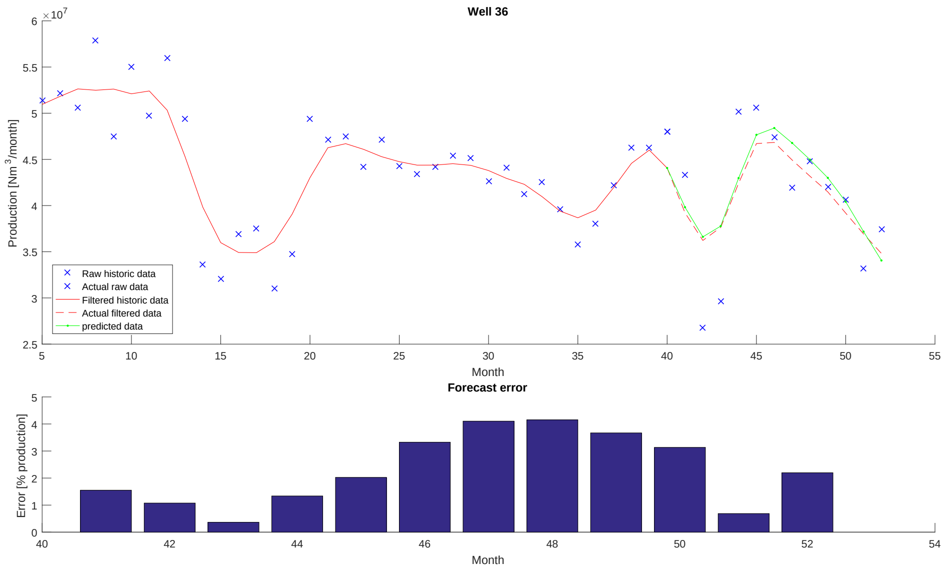


Figure 5.30: The actual and the predicted flow rate by ANN for well 36.

Unfortunately some wells were not as good predicted as the three wells above. In Figure 5.31 well 8 is shown and the prediction is not that good. The network did not predict the drop in month 42. The error is therefore almost 60%. The result is not strange though. The network trained well on the production curve, but it does not predict a sudden decrease in flow rate when trained on this production curve. Still it can be said that overall the flow rate is predicted very accurately.

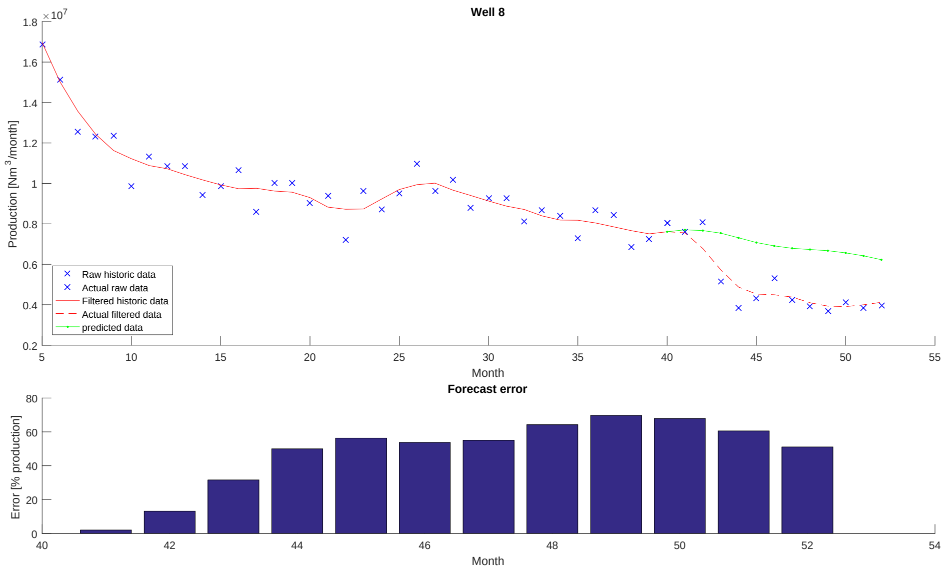


Figure 5.31: The actual and the predicted flow rate by ANN for well 8.

5.4. Predicting Liquid Loading

The network makes an accurate forecast and will be used as an alerter to predict liquid loading in a specific month. Well 1 is taken as example well, this well is liquid loading in month 57. Figure 5.32 shows this well. The prediction prior to the liquid loading month is shown on the x-axis, thus the actual month in which the alerter is making the prediction. The prediction is done 12 months in advance. The y-axis presents the prediction error that is to say the predicted month before or after the actual liquid loading moment. Month 0 represents the liquid loading month. Showing therefore the months in negative if predicted before liquid loading and positive when predicted after liquid loading.

For well 1 this means that the alerter predicted liquid loading to happen 12 months before liquid loading occurs. However it predicted liquid loading too early by 9 months. Thus in month 45 liquid loading was predicted in month 48. Then 4 months in advance the prediction is changed from under-prediction to over-prediction. The over-prediction is only by 2 months and 2 months before liquid loading the prediction finds the correct liquid loading moment. So even though there is a large fluctuation, the operator is alerted in advance. No stimulation was done yet for this well and it stopped producing after 98 months. In between month 57 and 98 the well produced only 4 million Nm^3 on average by intermittent production which can be seen in the production profile in Figure 4.2. Then another 100 months past by before a velocity string was installed. With the alerter this could have been done a lot more efficiently.

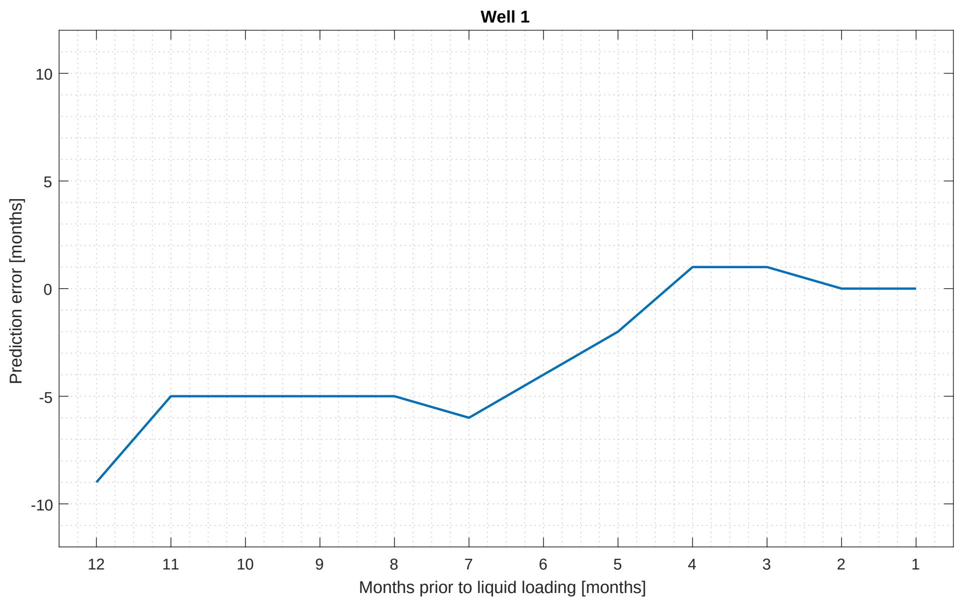


Figure 5.32: The prediction error in months versus the month in which the prediction is made prior to liquid loading is shown for well 1. Liquid loading occurs in month 0.

A second prediction is shown 5.33. Well 26 is liquid loading in month 65. For well 26 the alerter predicted liquid loading to happen 12 month prior to liquid loading by only one month under-prediction. This is a very good prediction. Thus in month 53 liquid loading was predicted in month 64. Then 5 months prior to liquid loading the alerter adjust its prediction to month 65, being the exact liquid loading month (at the 0 point). Closer to the event it shows some over-prediction with one or two months, but it finds the true value again. The small fluctuation are not of great importance. The fact stays that the well shows an accurate prediction of the liquid loading moment. The operator will know 12 months in advance that it needs to be careful for liquid loading to happen. The operator has a lot of time to start thinking about mitigation techniques. Again in the real scenario the operator waited too long, namely until month 172 before injecting foam, while the well was producing less than 1 million Nm^3 per month. The well is still flowing with the same amount of production, so foam increased production time.

The prediction plot for all wells is seen in Figure 5.34. The plot is similar to the previous prediction plot only a second y-axis is implemented showing the number of wells that are predicting in a certain

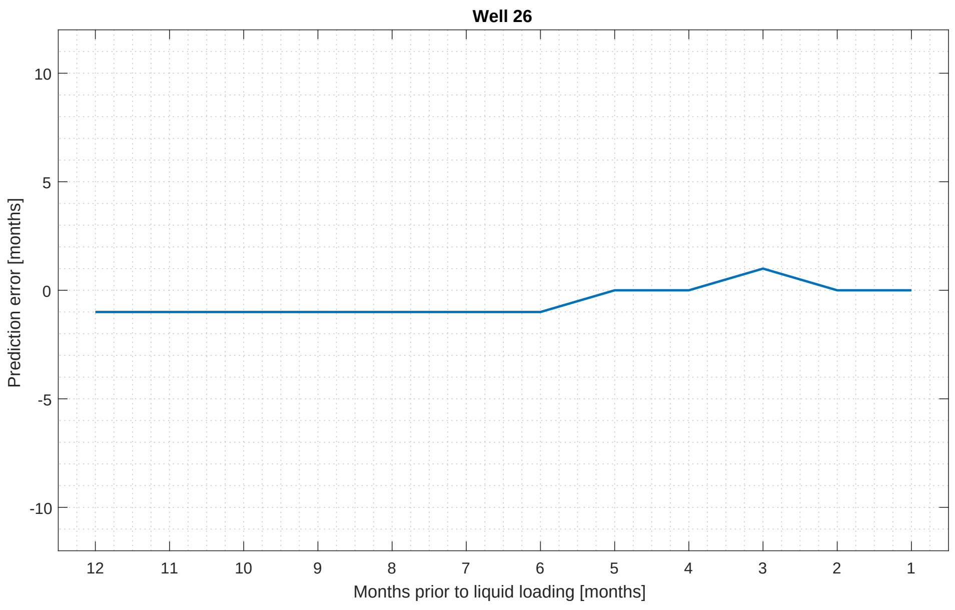


Figure 5.33: The prediction error in months versus the month in which the prediction is made prior to liquid loading is shown for well 26. Liquid loading occurs in month 0.

month prior to liquid loading. The prediction shows a similar trend to what was seen for well 1 and well 26. About 12 months prior to liquid loading the alert seems to be under-predicting. A few months prior to the event the alerter seems to be over-predicting. Note however the range in which the well can predict. 12 months prior to liquid loading the well can only under-predict, while a few months prior to liquid loading the chance of over-prediction is high. This would mean that 4 or 5 months in advance it would be the best time to predict liquid loading. Another feature that can be seen is that only 5 wells are predicting liquid loading 12 months in advance, the number steadily increases, and 1 or 2 months prior to the event 35 wells predict liquid loading. For 5 wells liquid loading is only predicted correctly one month before liquid loading occurs, they were over-predicting. Also note that predictions before the 12 months prior to liquid loading are not included in the figure. The alerter corrects itself on the false prediction, but they are misleading.

Thus the alerter is accurate, but for some wells only a short time in advance. The implications of this is discussed in the next chapter.

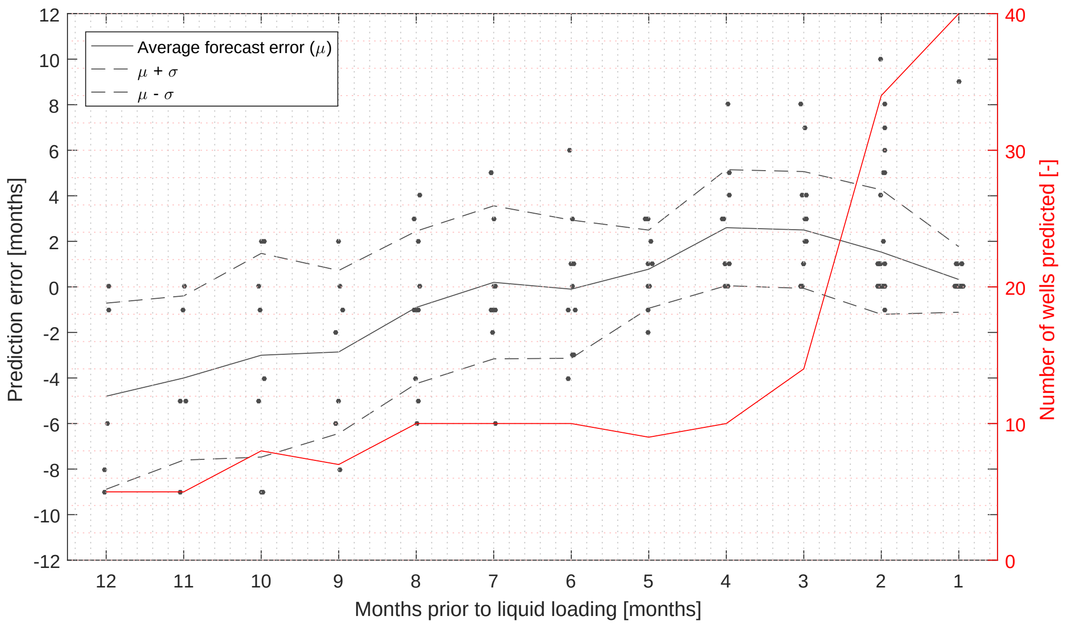


Figure 5.34: The prediction error in months versus the month in which the prediction is made prior to liquid loading is shown for all wells. Liquid loading happens at month 0. The red curve shows the number of wells that are predicting.

6 Discussion

6.1. Analysis EOFL techniques

6.1.1. Data Mining

The need for EoFL techniques is becoming more important as gas fields get into the mature or tail-end production phase. It is challenging to quantify the gain of these techniques. Furthermore data required to quantify the gain of these techniques is sometimes not available. The gathering of data and keeping it in a big data base is not always properly done, let alone in the early days. However research should always be done as the information from data could be very valuable. Recent projects are already mapped better with the help of better technology and large databases that can be build nowadays. Next to this research will improve when all operator are willing to share information about their EoFL projects. This research made an effective step in quantifying the value of EoFL techniques with the available data.

6.1.2. Sensitivity Study TNO Tool

For one of the wells a sensitivity study was done by means of a tornado plot. The parameters were varied based on the accuracy of measurement. This accuracy was extensively discussed with a field engineer. For the well head pressure, reservoir pressure, gas density, reservoir temperature, liquid gas ratio, A and F factors the following percentages were applied respectively; 2%, 2%, 5%, 1%, 20%, 10% and 10%. The tornado plot is shown in Figure 6.1. As expected the reservoir pressure has a large influence on the cumulative production. When reservoir pressure is low the drag force is insufficient to lift liquids up. As the gas liquid ratio is difficult to measure accurately, this factor was varied to the actual cumulative production by 20% and thus has a large influence on the cumulative production too. It can also be seen that the laminar flow factor is of importance too. Therefore it is important that the operator measures this factor accurately. The reservoir temperature and the friction factor do not have a large influence on the cumulative production.

The abandonment pressure will indicate at what reservoir pressure liquid loading occurs. To continue producing it is important to keep pressure high. The abandonment pressure is largely dependent on the liquid gas ratio as can be seen in Figure 6.2. A large liquid column will create a certain back-pressure and the draw-down pressure will therefore be lower and production ceases. It is apparent that reservoir pressure does not have a large influence on the abandonment pressure. This could be due to the small percentage that is given to the reservoir pressure.

The tornado plots of the relative gain by mitigation, the ultimate recovery and the years are shown in the Appendix.

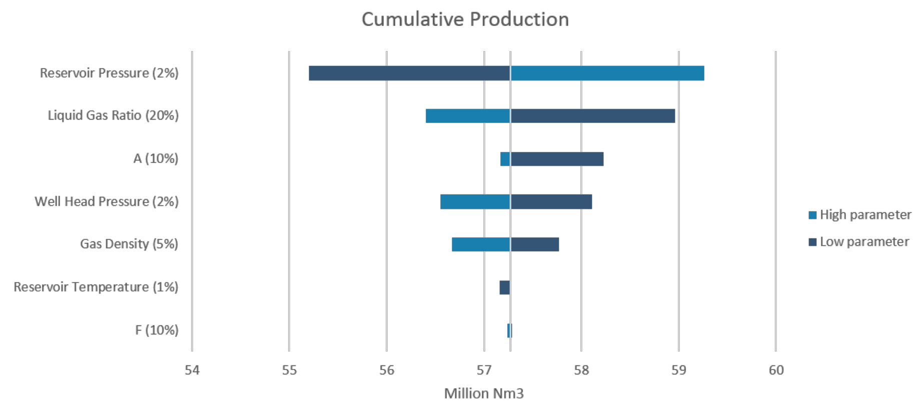


Figure 6.1: The sensitivity of the given parameters to the cumulative production.

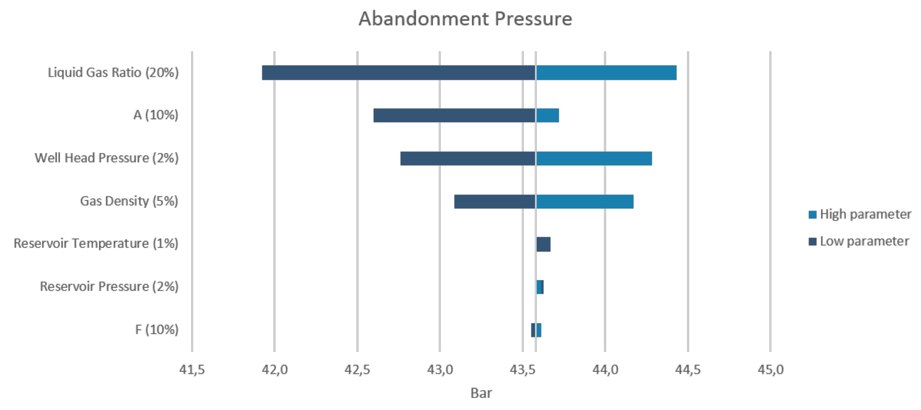


Figure 6.2: The sensitivity of the given parameters to the abandonment pressure.

6.1.3. Difference Actual and Prediction

The difference between actual and predicted cumulative production can vary a large amount. It is seen that for the 15 wells that were studied the operator predicts almost always above reality. They are very optimistic in their calculations. The tool is also optimistic for velocity string projects, however it predicts less than a foaming well actually produces.

There are many reasons why the operator and tool predict more than the well produces. The main issue could be maintenance. Any repairs, pipeline shutdowns, fish etc. are not taken into account when simulating production rates. These problems decrease production enormously. Another factor could be that the tool does not include other wells in the field. From P/Z plots the GIIP for each well is found. When more wells are in the same field the dynamic GIIP of the well is calculated by accounting for the amount of production it has had from the total reservoir. As this is still an estimation the tool is biased. The reason that the tool predicts pessimistic for foaming wells could be due to the assumption of foam giving a 50% decrease in critical rate. When the critical rate is decreased by a larger percentage the production will be predicted higher and vice versa.

Wells 5 and 7 from Figure 5.4 had a very low production after the installation of a velocity string. They are not producing anymore. The gain was so low that the investment made for the mitigation technique was not worth it. A leaking tubing was one of the reasons for the bad performance.

Well 6 and 8 were predicted way to high by the tool. Both had a velocity string installed with a tubing size of $2\frac{3}{8}$ inch. Well 6 had a former tubing with size 5 inch and well 8 with a 5.5 inch. Both had a liner with size 7 inch. Going from these large tubing sizes to a very small velocity string one expects a high gain. Next to this the LGR were low and the reservoir pressure at the start of the mitigation was relatively high. These conditions together simulate a high cumulative gain. This also explains the negative NPV values for the 4 wells. A velocity string costs a few million euro's. The production was to little to compensate for this investment.

According to the tool foam was injected into well 14 too early. The well could have produced on for another 1.7 years before it would stop producing. With foam it produced a year longer with a cumulative amount of 48 million m^3. This is very low compared to the predictions. It may be that the operator should have waited with foam injection. The uncertainty here lies in the accuracy of the data, the tool seems to predict a higher production rate than was possible for this well. As foam is very cheap and the well did produce a fair amount the NPV turned out to be positive after these three years.

6.2. Artificial Neural Networks

The alerter was build on the basis of neural networks. The network is designed with 2 layers each containing 75 neurons that use the tanh function. It takes 36 months into account when predicting the next month. The network forecasts the production rate, showing an accurate prediction. The liquid loading moment can be found when the flow rate falls below the critical rate calculated by the Coleman criterion. The alerter depicts the month in which this happens. The alerter had a high accuracy, but the network can always be improved. Several ways are discussed in this chapter. The next chapter proposes some recommendations for optimizing NN even further.

6.2.1. Prediction Accuracy

It was seen from Figure 5.34 that the alerter predicts the month prior to liquid loading for almost all wells accurately. However it predicts it only shortly in advance, one or two months. Only for $1/4$ of the wells it predicts liquid loading half a year prior to liquid loading. When the alerter predicts half a year prior to the event the prediction error varies $+/-\ 3$ months. This should be suitable for an operator. When the alerter is predicting liquid loading a few months earlier than the actual month in which liquid loading occurs, there is no problem. From field measurements one detects liquid loading on a short notice, which helps in deciding when to use an EoFL technique. Foam injection needs to be done when the well is still flowing, so it is better to inject a little too early than too late. So when the well is over-predicting it is a bigger problem. The well may already be liquid loading without a triggered alerter. Luckily the alerter only over-predicts for a small amount of wells when predicting close to the liquid loading moment. The risk of being to late is thus small.

The period of an accurate prediction would preferably be extended to half or a whole year. The difficulty at the moment lies in the sharp decrease in production rate at the onset of liquid loading. The alerter does not expect such a sudden drop and therefore only predicts liquid loading to happen when the actual month is also at lower gas rates. This sharp drop may be less when more data points are included, for example with daily data.

Another way to increase prediction accuracy is to increase the data by the number of wells. These wells can be in the Netherlands or from abroad. Liquid loading is a worldwide issue and the more wells, and thus data, can help increasing the accuracy of the prediction.

6.2.2. Daily Data

Training on daily production data can also be useful as more data is available as input. One training was done with daily data to compare the training performance with the monthly data. In order to do so four wells were chosen that had accurate daily data. The same network with two layers and 75 neurons with the tanh function was used. Instead of taking 3 years as input, only 1 year was taken as the consumer computer was not capable of training a larger network memory wise. For one of the wells the forecast of the gas flow rate per day can be seen in Figure 6.3. The same four wells were also trained with monthly data to compare the results. The monthly data was trained with 3 years of input data. The monthly predicted flow rates can be seen in Figure 6.4.

The forecast of daily data is very accurately. At a few data points the percentage error is very high, but this was due to the low production on certain days. The absolute error is very low, lower than the one of monthly data. The high error for these peaks can be solved by pre-proccessing the daily data better, for example use a filter with a large span. The next step is to predict liquid loading with daily data. On day 560 the alerter predicted the onset of liquid loading in month 899. The actual liquid loading moment calculated by Coleman was on day 894. So the alerter predicted quite accurately a year in advance. The exact moment was predicted 2.5 months prior to liquid loading. This is similar to the predictions of the monthly forecast. However unfortunately this prediction is not accurate to field data. The well produced with a stable flow after 900 days. When determining the liquid loading moment from the monthly forecast, it was seen that liquid loading was suspected in month 168, meaning around day 5040. The reason for the alerter to predict liquid loading too early is the fact that the daily data needs to be smoothed more than the monthly data. Due to the low outliers the alerter triggered too early. Thus once again pre-proccessing is important before training daily data.

However the daily data did predict liquid loading 2.5 months prior to liquid loading, while the monthly data predicted liquid loading only 1 month before the event. Considering the fact that training was only conducted with 4 wells, it is difficult to quantify if daily data predicts liquid loading more accurate or

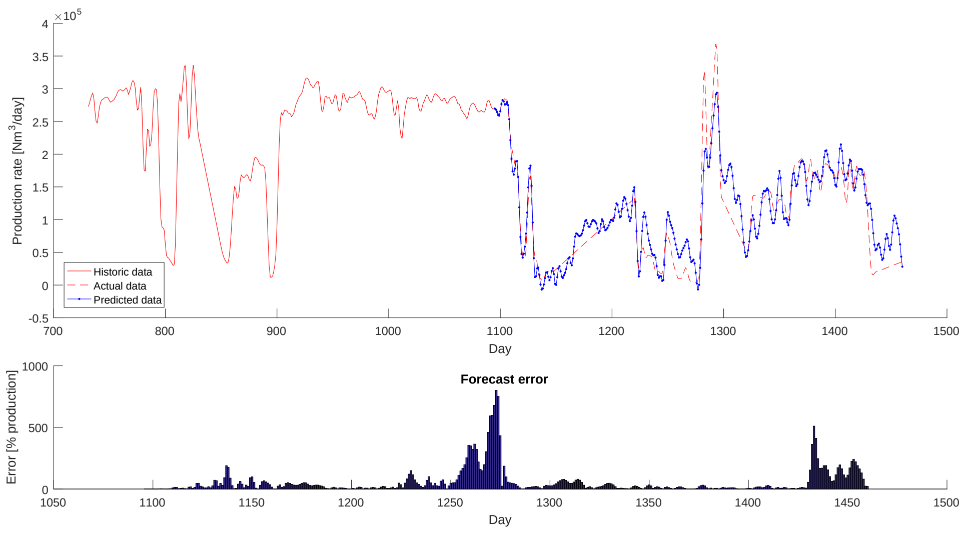


Figure 6.3: The actual daily flow rate and the daily predicted flow rate trained by NN on 4 wells with daily input data.

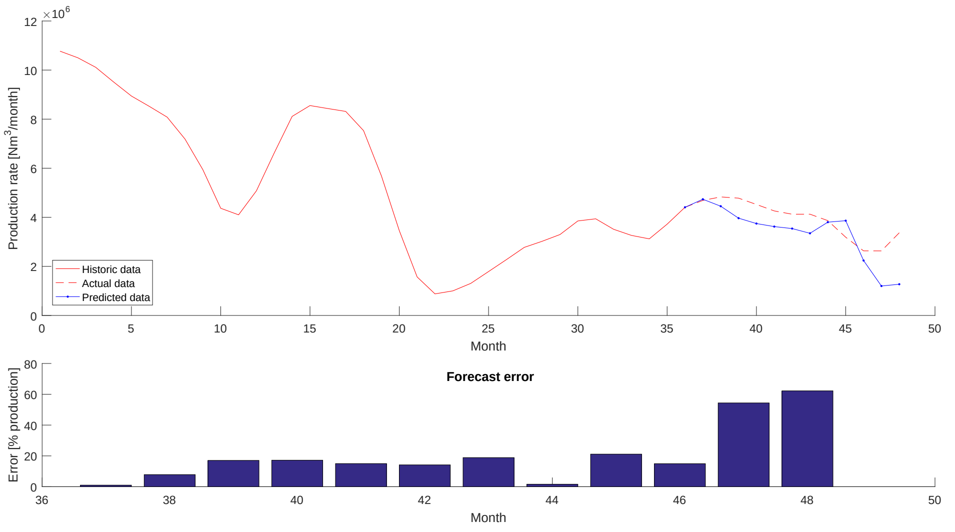


Figure 6.4: The actual monthly flow rate and the monthly predicted flow rate trained by NN on 4 wells with monthly input data.

not. Considering the results daily data looks promising as it can predict liquid loading a longer period in advance, but first more training needs to be conducted with a suitable filter and more wells.

6.2.3. Tubing Head Pressure

One important parameter that is missing is the flowing tubing head pressure (FTHP). The pressure would have given extra information to the network if it was used as an input value next to the flow rate. Two inputs may bring the forecast to a more precise solution as pressure and production rate are coupled. When pressure decreases, the flow rate will decrease as well. However when the well is closed-in, meaning 0 production rate, the FTHP will build-up. After start-up the production rate is then higher in the next couple of days than before the well was closed-in. When including pressure one needs to train the network on daily data. Unfortunately the daily pressures were incorrect for many wells. As an inaccurate input would only confuse the training process even further, the option was disregarded.

6.2.4. Presence of Multiple Wells in a Field

An uncertainty factor is the presence of other wells in a wells vicinity. Thereby the production pattern is influenced. This can to some extend been reflected in the well pressure [Chakra et al., 2013]. As the neural network does not know the production performance is influenced by external factors the forecast trains on the influenced production pattern. It may have been better not to be included.

6.2.5. Influence of Important Parameters

The input values for the neural network were tubing size and gas rates. Coleman et al. [Coleman et al., 1991] states that the wellbore cross-sectional area obviously is one of the most important variables in the critical rate calculations. Thus next to gas rates the tubing size needs to be taken into account. Coleman et al. states that liquid/gas ratios below 22.5 bbl/MMscf, which is equal to 126.4 cm^3/m^3, have no influence in determining the onset of load-up, meaning the gas flow rate is the dominant factor. In this research the choice was therefore made to not include the LGR as an input in the neural network. Taking along the fact that LGR's have a large inaccuracy when measured. Due to this the LGR will not add much to the predicting of the liquid loading moment. The paper also examined that other variables such as temperature, gas gravity and inter-facial tension show minor influence on the accuracy of critical rate calculations.

6.2.6. Coleman Criterion

As described before the critical rate is calculated by the Coleman Criterion. This Coleman critical rate is calculated by Equation 2.4. The critical rate is mainly depended on the well head pressure and the tubing size. From the TPC curve the critical rate can also be determined. All parameters inserted into the tool are accounted for in the TPC and thus in calculating the critical rate. It is difficult to know which method is more accurate, but a comparison between the two is made in Figure 6.5 for the wells that were studied with the TNO tool. This figure shows that the Coleman Criterion for most wells gives a lower critical rate than the TPC curve does. This means that Coleman predicts liquid loading to happen in a later stage. The alerter created by neural net may therefore be too optimistic. This means that the liquid loading moment can be earlier than predicted by the alerter. The alerter is a good tool and helps operators to predict the liquid loading moment in advance. Nonetheless the operator needs to keep looking at field conditions and use the methods described in chapter 2 as well for final judgment.

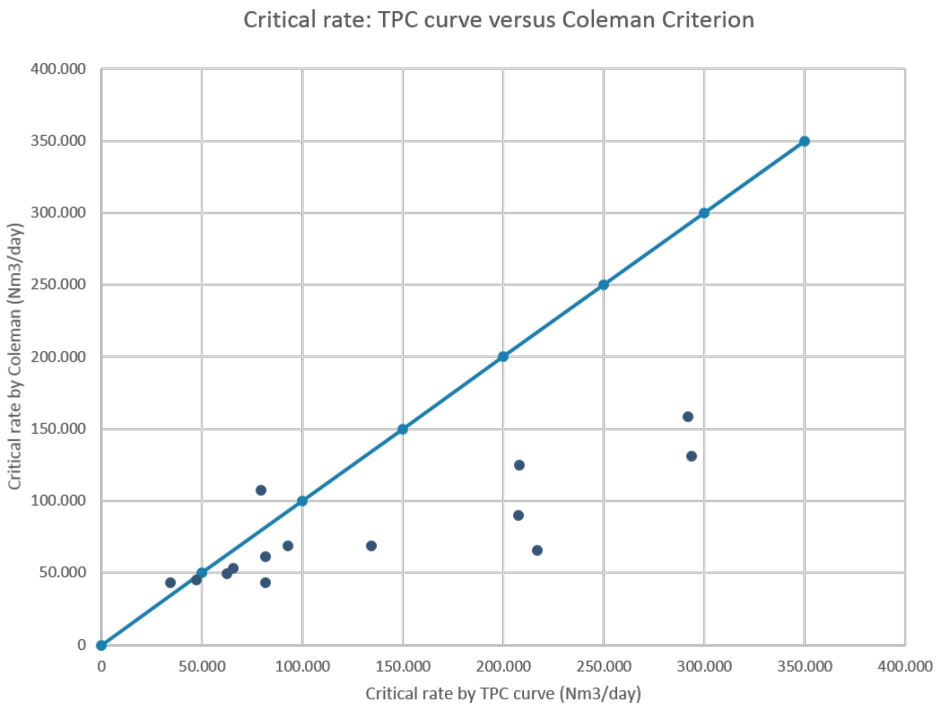


Figure 6.5: The comparison between the critical rate calculated by the Coleman Criterion or determined by the TPC curve.

Coleman Criterion was also compared to field data. The question rises if the critical rate calculated by Coleman resembles that of the minimum rate at which the well was producing. From the field data it is difficult to tell. As said before the decline curve is not a smooth curve and due to the fluctuation it is challenging to determine the exact onset of liquid loading. Though to give a representation of the accuracy of the Coleman rate the following graph is presented (Figure 6.6). From this graph it can be concluded that the critical rates resemble each other and thus, having said that, the Coleman is an accurate method to determine the critical rate for these wells.

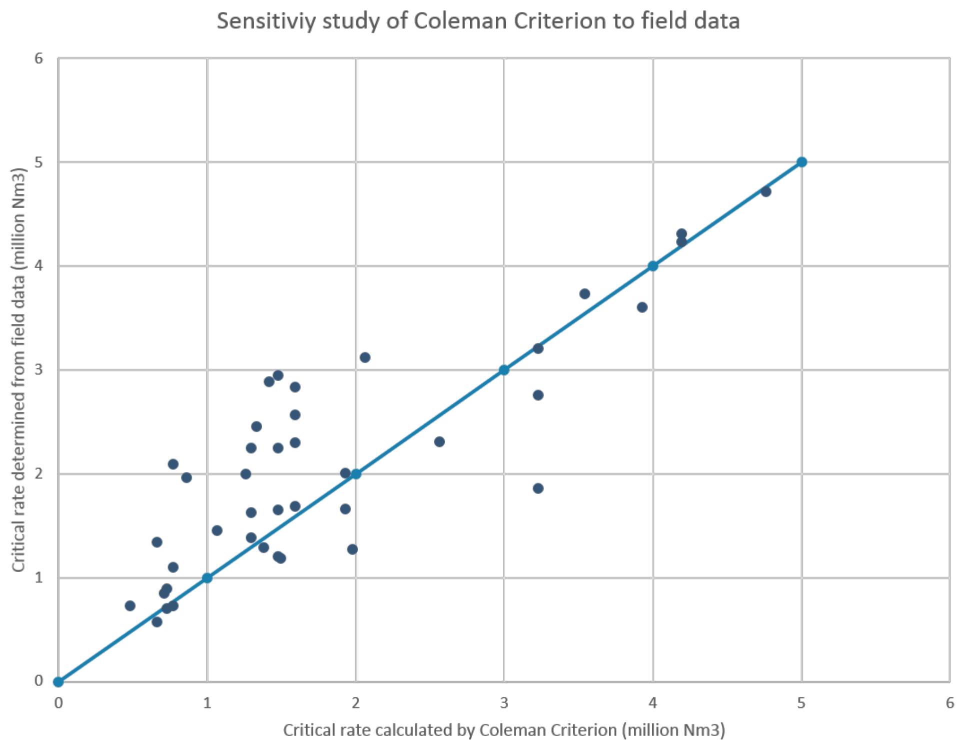


Figure 6.6: The comparison between the critical rate calculated by the Coleman Criterion and the minimum rate at which a well can produce from field data.

6.2.7. Research on Liquid Loading Prediction

The onset of liquid loading is observed when producing at a low rate by the indicators that were discussed in chapter 2. All of them can only detect liquid loading when liquid loading is already happening. Calculating the minimum critical rate, however, is an accurate indicator that may provide DCA or the forecast build by ANN to predict the liquid loading moment. A lot of research was conducted in order to find the best method in calculating the critical rate. The most famous, and discusses previously, are Turner [Turner et al., 1969] and Coleman [Coleman et al., 1991]. Following up on Turner many scientist have improved his research. Nosseir et al. [Nosseir et al., 2000] expanded the Turner Criterion for a entrained drop model to more than one flow regime in a well considering different flow conditions. Turner assumed that the flow regimes were confined between $10^4 < \text{Re} < 2 \times 10^5$ but Nosseir states that due to the wide range of pressures, temperatures, and flow rates encountered in gas wells, this is not necessarily confined. Therefore two analytical models were developed. One for the transition, and the other one for highly turbulent flow regime. These equations will describe the critical rate more accurately depending on the flow regime. Li [Li et al., 2002] adopts the view that the liquids droplets entrained in gas wells tend to be flat shape and deduced a new formula for continuous removal of liquids from gas wells. The results calculated from this formula are smaller than that of Turner's. However, Boyun Guo [Guo et al., 2005] states that Turner's method underestimates the minimum gas velocity for liquid removal and therefore determined the minimum kinetic energy of gas that is required to lift liquids. Applying the minimum kinetic energy criterion to the four-phase (gas, oil, water, and solid particles) flow model resulted in a closed-form analytical equation for predicting the minimum gas flow rate that is more accurate than Turner's model. Veeken [Veeken et al., 2009] also presents a Modified Turner expression, which includes an upward correction to best-fit a set of offshore field data. This Modified Turner minimum stable rate better matches the minimum stable gas rate derived from multiphase flow correlations.

Since critical gas rate equations only give a simple idea for the minimum rates, liquid loading can be determined by nodal analysis. This will be more detailed since it considers the complete flow path of fluids from reservoir to wellhead. The problem however lies in the fact that a lot of reservoir and well data is need to calculate the TPC and IPR curve and therefore it was not possible to make use of nodal analysis in this research.

All methods have their advantages and disadvantages. Coleman was chosen in this research to calculate the critical flow rates as it provides a simple equation without the need of flow regimes and actual rates that are difficult to determine. Moreover it is suitable for low pressure gas wells and shown to be accurate to field data. Nonetheless more research should be conducted to examine the accuracy of the other methods when predicting the onset of liquid loading.

6.2.8. Decline Curves

Decline curve analysis (DCA) is a graphical procedure used for analyzing declining production rates and forecasting future performance of oil and gas wells. Oil and gas production rates decline as a function of time. The loss of reservoir pressure or changing relative volumes of the produced fluids, are usually the cause. Fitting a line through the performance history and assuming this same trend will continue in future forms the basis of DCA concept. It is important to note here that in absence of stabilized production trends the technique cannot be expected to give reliable results. A stabilized production trend is however not often seen. Although the DCA is a good technique it is important that research is done for other methods. The advantage of the liquid loading alerter made by ANN is the fact that it is based on real field data. Thus including production instabilities. Next to this the forecast is made on a time basis. An easy measure for the operator to know the moment of liquid loading and to get ready to perform mitigation.

6.2.9. Alternative Purposes

NN is applicable in various data considered research. In the petroleum industry neural nets have also shown to be very valuable. Research was done where ANN was trained successfully in order to predicted fractures based on wireline log data [Kooijman, 2011]. Equilibrium ratios play a fundamental role in the understanding of phase behavior of hydrocarbon mixtures. They are important in predicting compositional changes under varying temperatures and pressures conditions in reservoirs, surface separators, production and transportation facilities. [Habiballah et al., 1996] presents a new approach for predicting equilibrium ratios using ANN. Next to this ANN was used to determine the optimum number and location of the infill production new wells under development stage of the field life for a oil field in Iraq [Ghazwan, 2012].

Taken this in account, and the many more research that is conducted, artificial intelligence is making solid steps towards becoming more and more accepted in the oil and gas industry and provides for new methods for identification, prediction and control.

6.2.10. Applicability

ANN has proven to be very useful for many application in the oil and gas industry. This research also shows the potential that is offered by ANN to forecast gas rates. To train more complex networks one has to start thinking of professional servers rather than consumer electronics due to memory restrictions. In this research, training of one network already took several hours to days, let alone using more neurons. Therefore the time limits the building of ANN. If hardware is available that can decrease time and complexity the possibilities of ANN will increase even further. Having said this, it can only be possible if data is available. The famous saying 'garbage in is garbage out' is not just a saying. The quality of the data will influence the result enormously. As artificial intelligence is better understood today and its valuable benefit to research, it is important that companies, not only in the Netherlands but around the world, gather data in order to use ANN or other methods of artificial intelligence within the coming years.

7 Future Outlook

7.1. Recent Projects

As EOFL techniques will become more and more needed due to well depletion in the near future, it is important that this analysis is continued for new projects. Data keeping of recent projects, this included wells where EoFL techniques are applied after 2010, is much better done than in the past. This increases the ease and accuracy to quantify the gain of these projects. Big data will provide a lot of insight in technical projects and will help further innovation. The operator plays a large role in this and it is important that they monitor the information correctly. At the moment we are not there yet, but this research as a first step in quantifying these techniques is a good start and research should continue into this field. More wells can be evaluated, not only various operators, but also techniques, other than velocity strings and foam. The more wells/data the more accurate results and then these past operation will show a path for the future.

7.2. Clustering

Instead of forecasting a neural clustering network can be made. When clustering, the data is divided up into different groups to which they fit best [MATLAB®, 2016c]. For example the tubing size is a cluster. But even going a step back. From all the wells in the Netherlands a selection was made manually of the wells that are liquid loading. The production rate of these wells were then used as input for the neural network. From several features the clustering network can make a selection from all these wells, which would save a lot of time, especially when using several hundred wells as input data. Then when predicting liquid loading, wells that are or are not seasonal dependent can be clustered together. Giving a more accurate prediction. As liquid loading is dependent on different parameters the liquid loading moment may be found by clusters or the clusters are used as a pre-processing method after which a forecast is made.

7.3. Forecasting with Pressure Data

As discussed in the previous section the pressure data is important to take into account. When correct pressure measurements are found the daily pressures can be used. A method to predict a decrease in the overall production of a well can be done by looking at the pressure build-ups. When a well is producing at the maximum rate and closed in for whatever reason, the decreased FTHP during production will increase until it reaches reservoir pressure again. If this is not the case the build-up will not be as high, therefore they will be called mini build-ups. A neural network or another potential network based on artificial intelligence can learn the pattern of the mini build-ups happening. Then an indication of the moment of these mini build-ups can be predicted and an alerter can be made based on this. The method does not only apply for liquid loading, as the reason for a smaller build-up is not known, but it can be used for any particular problem in the well. For example salt precipitation. If the prediction can be made in an early stage the operator has time to find out and fix the problem without wasting a lot of time.

7.4. Different Algorithms

Learning from big data is going to have a major effect on the efficiency of operations in the future. Artificial neural networks may not be the optimal algorithm to forecast flow rates. Another one to consider is genetic algorithm (GA). GA is a search algorithm based on the mechanics of natural selection. The basic techniques of the GA are designed to simulate processes in natural systems necessary for evolution by means of ”survival of the fittest”. Given a problem a GA generates a set of possible solutions and evaluates each in order to decide which solutions are fit for reproduction. If a particular solution is more fit then it will have more chances to generate new solutions. As such they represent an intelligent exploitation of a random search used to solve optimization problems. While randomized, GA is no simple random walk. They efficiently exploit historical information to speculate on new search points with expected improved performance [Goldberg, 1989].

So instead modeling complex relationship between inputs and outputs data or to find patterns in data like NN does, the GA finds a solution as a search technique. Both have their advantages and disadvantages, ANN is much more efficient because a gradient is already available, while GA does not have a gradient and no data is needed beforehand. The data appears during the training process. Therefore GA can perform a directed search of the solution space, find the shortest route between two points. By doing so it may find a more accurate solution.

Both have different approaches but they can also be used together. In doing so the flow rate can first be trained on the ANN, as was done in this research, then GA can be used to try to find an even better solution to predict liquid loading.

8 Conclusions

Analyzing EoFL techniques

- The majority of the gas wells in the Netherlands are considered to be mature or in the tail-end production stage, leading to a ceased production and stimulation of the well is required to keep production from these wells going.
- Liquid loading is one of the major issues that decrease production and EoFL techniques are needed to remove the liquid column from the well.
- From the wells studied in this research it can be concluded that the volume gain will be around 10 to 15 million Nm^3 with a velocity string installed or foam injection for a large amount of the wells. Other wells produce even more than a 100 million Nm^3.
- It can be concluded that the operator predicts more than the volume gain is after using a mitigation technique.
- The TNO tool is of large value when performing a quick scan in the volume gains of foam and velocity strings.
- Velocity strings and foam show a high success rate for the wells studied. 11 out of 15 wells showed a very high NPV, some even a NPV of 20 million euros or more. 4 wells had a negative NPV, but only because production was ceased immediately due to well problems.
- From this research mitigation techniques are shown to be very valuable and more research should be done to further investigate methods to keep production going in the Netherlands.
- More energy needs to be put in correctly organizing and gathering data. It would improve future studies.

Neural networks

- With ANN it is possible to forecast monthly production data from historic field data.
- Pre-processing is the most important step in building a network. Famous saying: 'Garbage in is garbage out' is not just a saying, the patterns in data will be learned wrongly when data is inaccurate and then the forecast will not be accurate.
- A filter needs to be used to for smoothing the production rate. The Lowess filter was chosen.
- The data was normalized to enhance fairness of training.
- The Levenberg-Marquardt was used as training function.
- The data was divided interleaved between a training, validation and testing set with the ratio 70%, 15% and 15% respectively.

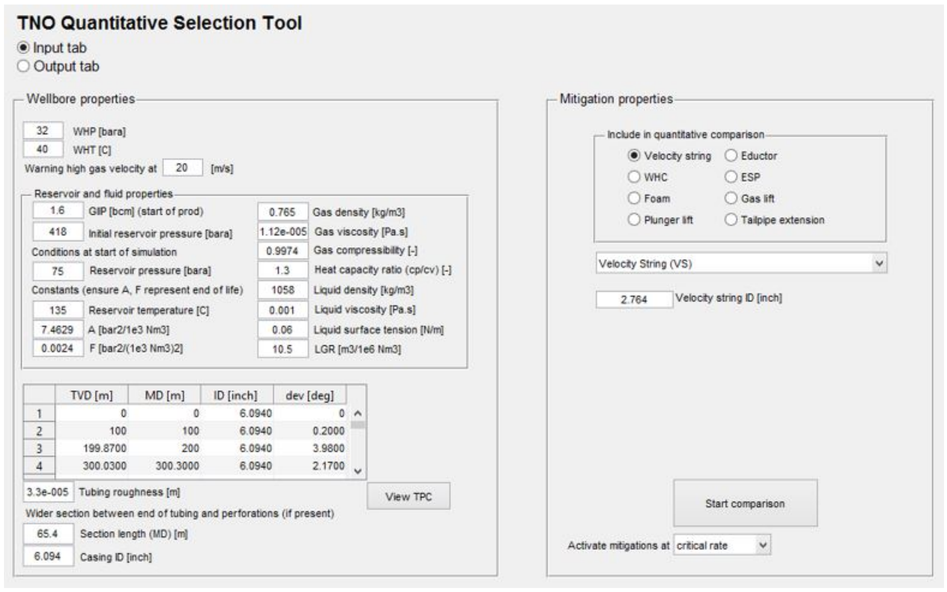
- The network was constructed with two layers, each containing 75 neurons. This network had the lowest mean squared error. Each neuron has the tanh function as activation function.
- Training the network on logarithmic data and log return data did not increase performance.
- The network predicted the flow rates very accurately for most wells. It can be said that neural networks works well when forecasting gas rates.
- Coleman Criterion is an accurate model to calculate the critical rate. The critical rate determined by field data shows similar results to the Coleman rate. However Coleman rate has the tendency to calculate lower minimum rates then the TPC curves does. The alerter could therefore be more optimistic than it should.
- The network is very accurate when predicting liquid loading. It has difficulties with predicting the precise month, but is always very close. Most of the time it predicts the exact moment of liquid loading only one or two months in advance. This unfortunately is very short.
- Using daily data and tubing head pressures could increase training performance, and thus a more accurate forecast. Next to using another method then time-series, namely clustering.

Appendix A

Velocity String wells & Foam wells			
Well Name	Coleman rate (Nm^3/month)	Well Name	Coleman Rate (Nm^3/month)
1	4,760,496	21	664,230
2	3,925,250	22	483,301
3	4,191,284	23	728,807
4	4,191,284	24	769,068
5	1,932,581	25	769,068
6	1,478,359	26	728,807
7	1,478,359	27	3,542,755
8	2,063,562	28	3,228,844
9	1,977,107	29	3,228,844
10	1,294,560	30	3,228,844
11	1,294,560	31	1,294,560
12	1,383,052	32	1,419,610
13	1,334,659	33	1,478,359
14	1,498,069	34	1,589,709
15	1,259,027	35	1,589,709
16	1,064,215	36	1,932,581
17	862,220	37	1,589,709
18	2,562,743	38	1,589,709
19	769,068	39	1,478,359
20	707,889	40	664,230

Table 1: Table showing the critical rate calculated from the Coleman criterion for each well.

Appendix B



TNO Quantitative Selection Tool

[✓ Input tab]
Output tab

Wellbore properties

32	WHP [bara]
40	WHT [C]
Warning high gas velocity at 20	[m/s]

Reservoir and fluid properties

1.6	GIP [bcm] (start of prod)	0.765	Gas density [kg/m3]
418	Initial reservoir pressure [bara]	1.12e-005	Gas viscosity [Pa.s]
Conditions at start of simulation		0.9974	Gas compressibility [-]
75	Reservoir pressure [bara]	1.3	Heat capacity ratio (cp/cv) [-]
Constants (ensure A, F represent end of life)		1058	Liquid density [kg/m3]
135	Reservoir temperature [C]	0.001	Liquid viscosity [Pa.s]
7.4629	A [bar2/1e3 Nm3]	0.06	Liquid surface tension [N/m]
0.0024	F [bar2/(1e3 Nm3)2]	10.5	LGR [m3/1e6 Nm3]

	TVD [m]	MD [m]	ID [inch]	dev [deg]
1	0	0	6.0940	0
2	100	100	6.0940	0.2000
3	199.8700	200	6.0940	3.9800
4	300.0300	300.3000	6.0940	2.1700

3.3e-005 Tubing roughness [m]

Wider section between end of tubing and perforations (if present)

65.4 Section length (MD) [m]

6.094 Casing ID [inch]

Mitigation properties

Include in quantitative comparison

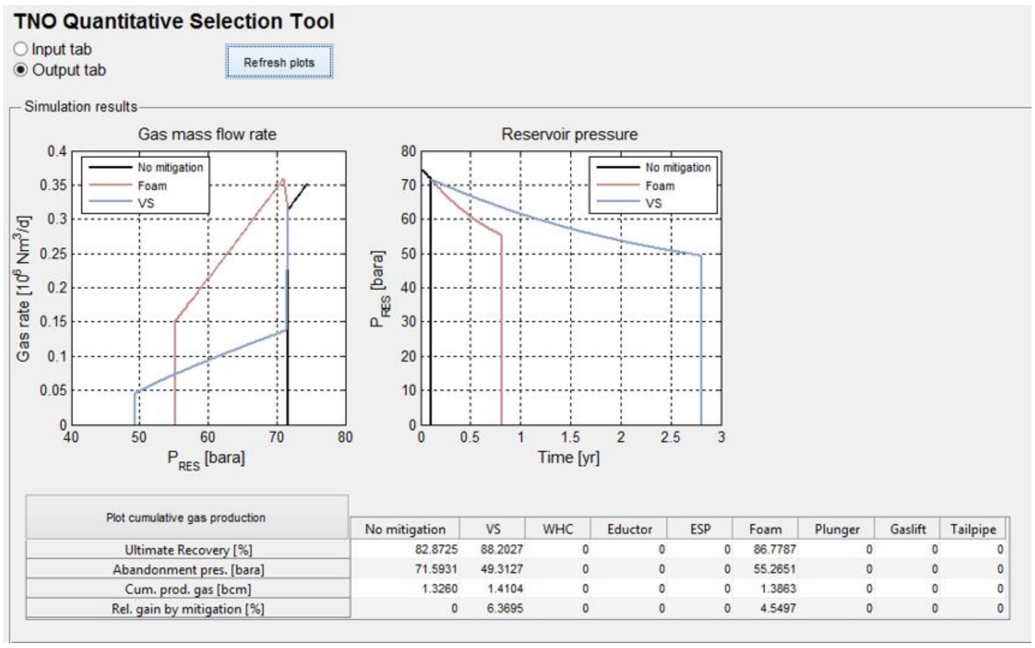
[✓ Velocity string] Eductor
WHC ESP
Foam Gas lift
Plunger lift Tailpipe extension

Velocity String (VS)

2.764 Velocity string ID [inch]

Activate mitigations at critical rate

Figure 1: The interface of the TNO deliquification toolbox, showing the input values [MATLAB®, 2016a].



TNO Quantitative Selection Tool

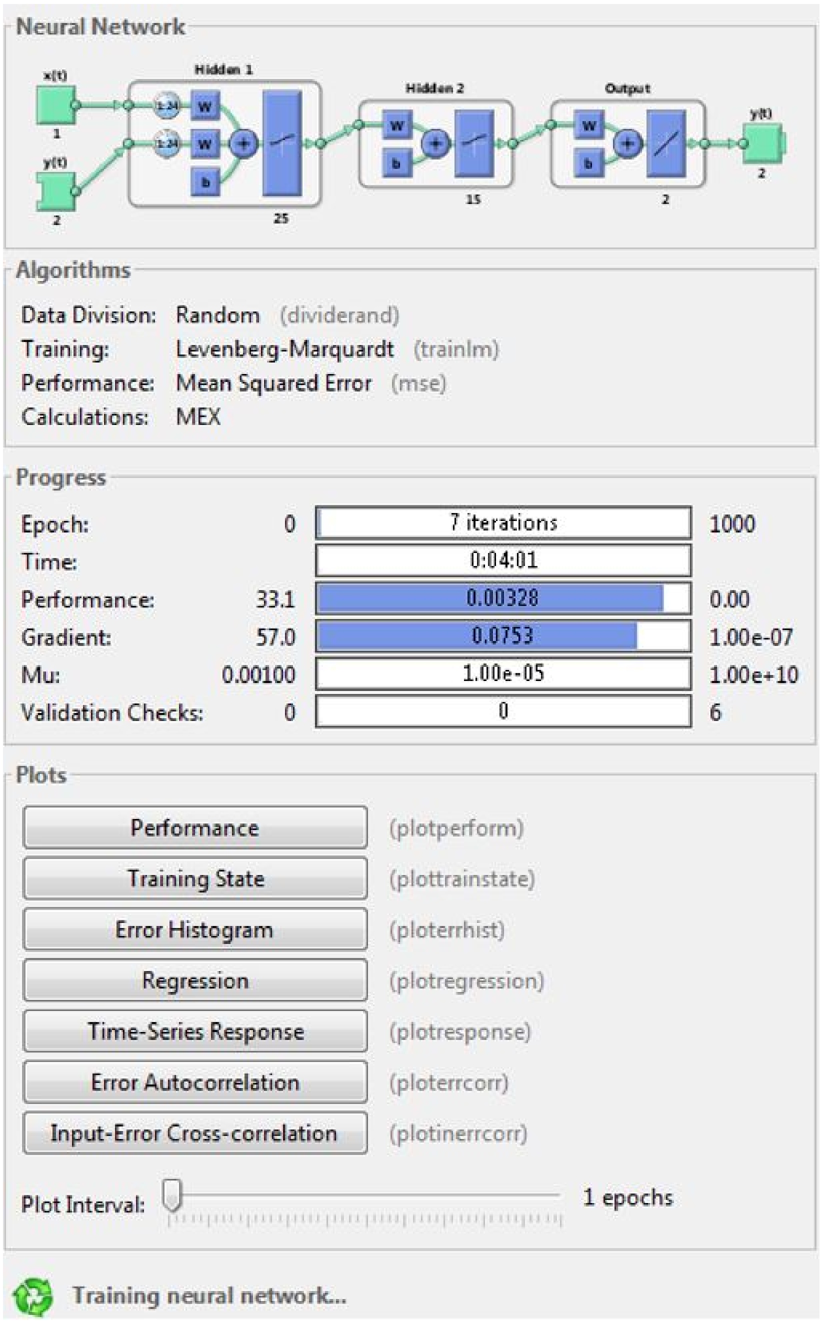
Input tab
[✓ Output tab]

Simulation results

	No mitigation	VS	WHC	Eductor	ESP	Foam	Plunger	Gaslift	Tailpipe
Ultimate Recovery [%]	82.8725	88.2027	0	0	0	86.7787	0	0	0
Abandonment pres. [bara]	71.5931	49.3127	0	0	0	55.2651	0	0	0
Cum. prod. gas [bcm]	1.3260	1.4104	0	0	0	1.3863	0	0	0
Rel. gain by mitigation [%]	0	6.3695	0	0	0	4.5497	0	0	0

Figure 2: The interface of the TNO deliquification toolbox, showing the results [MATLAB®, 2016a].

Appendix C



Algorithms

Data Division:	Random	(dividerand)
Training:	Levenberg-Marquardt	(trainlm)
Performance:	Mean Squared Error	(mse)
Calculations:	MEX	

Progress

Epoch:	0	7 iterations	1000
Time:		0:04:01	
Performance:	33.1	0.00328	0.00
Gradient:	57.0	0.0753	1.00e-07
Mu:	0.00100	1.00e-05	1.00e+10
Validation Checks:	0	0	6

Plots

Performance	(plotperform)
Training State	(plottrainstate)
Error Histogram	(ploterrhist)
Regression	(plotregression)
Time-Series Response	(plotresponse)
Error Autocorrelation	(ploterrcorr)
Input-Error Cross-correlation	(plotinerrcorr)

Plot Interval: 1 epochs

Training neural network...

Figure 3: The interface of the artificial neural networks toolbox [MATLAB®, 2016a].

Appendix D

Cumulative production as input data

An operator thinks in cumulative gas production rather than the data of abandonment. Therefore, as a second option, the cumulative production was taken into account as an input value next to monthly rates and the tubing size. This training is similar to the training of the production rate and carried out simultaneously. Thus the historical cumulative production is the input and the target is a forecast of the cumulative production. As the liquid loading moment is expected to happen at high and more constant slope of the cumulative curve it might add to a precise prediction of the onset of liquid loading.

Figure 4 shows the prediction of ANN for one of the wells. The monthly flow rate is shown at the top and the cumulative production at the bottom. The error percentage in between the two graph shows the difference between the actual data and the predicted data. Unfortunately this method does not work. The forecast of the cumulative production is influenced by the monthly rates and shows negative signs. This is not possible and the option was therefore disregarded. Research should be considered on how to train the factors independently but still considering the cumulative production as an input.

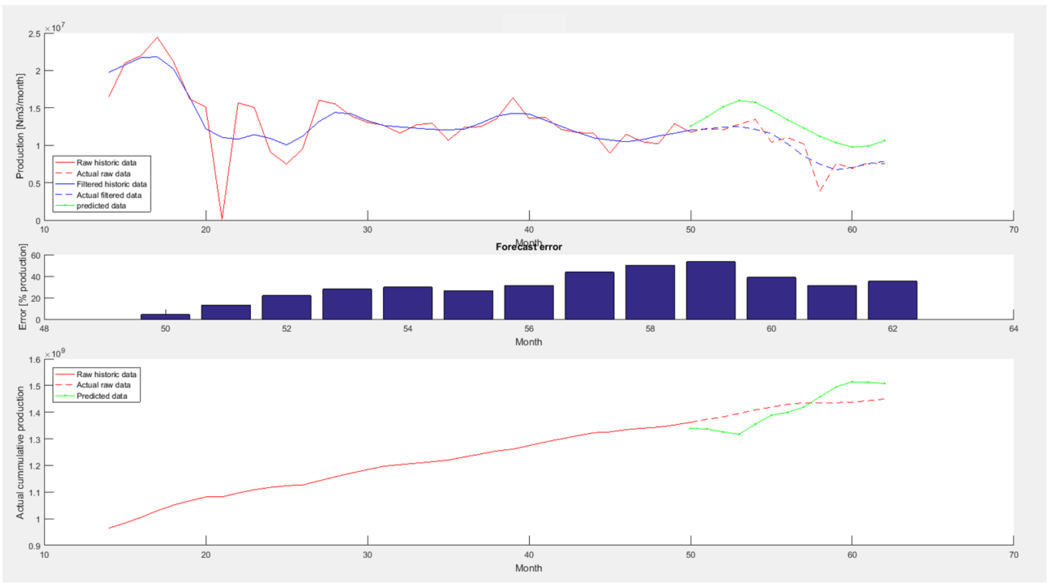


Figure 4: The monthly gas rate and the cumulative production forecasted by ANN. The error percentages shows the difference between actual filtered data and the predicted data.

Appendix E

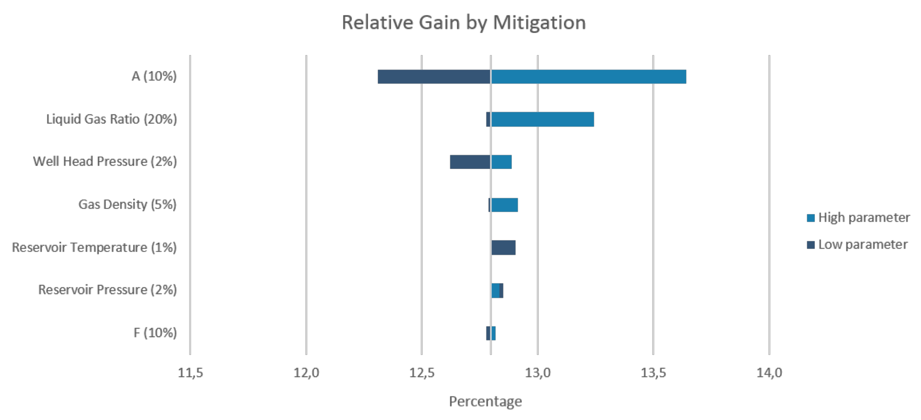


Figure 5: The sensitivity of the given parameters to the relative gain by an EOFL technique in comparison with no mitigation.

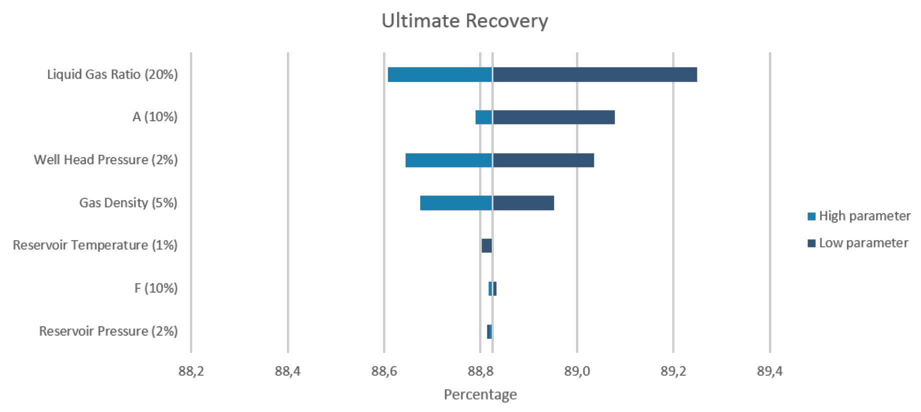


Figure 6: The sensitivity of the given parameters to the ultimate recovery.

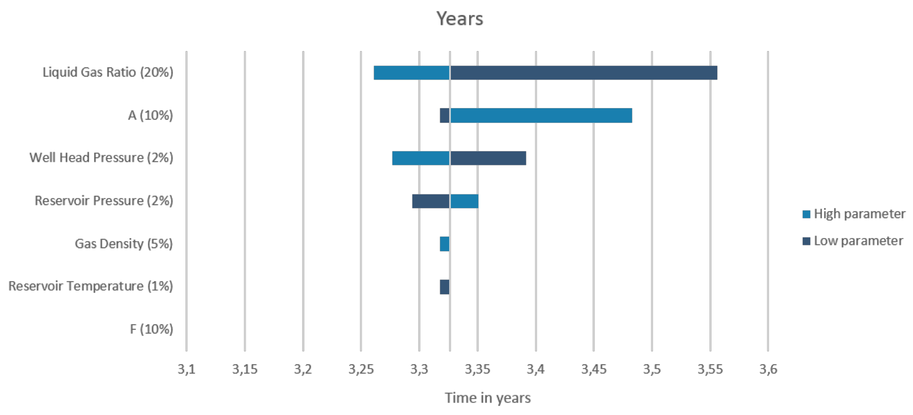


Figure 7: The sensitivity of the given parameters to the production years.

Appendix F

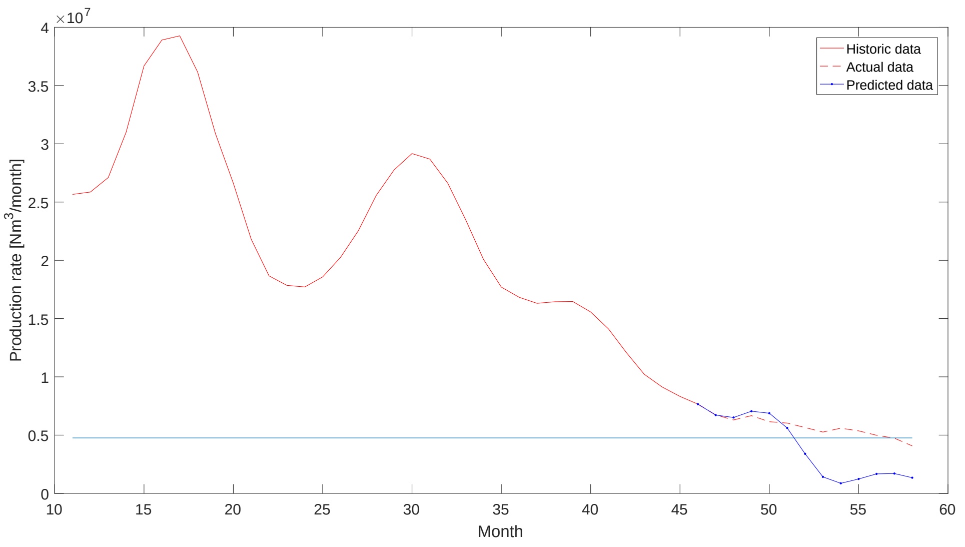


Figure 8: The actual and the predicted flow rate by ANN for well 1. When the well falls below the Coleman Critical rate liquid loading occurs. Each month a new prediction is made until the actual moment of liquid loading is found.

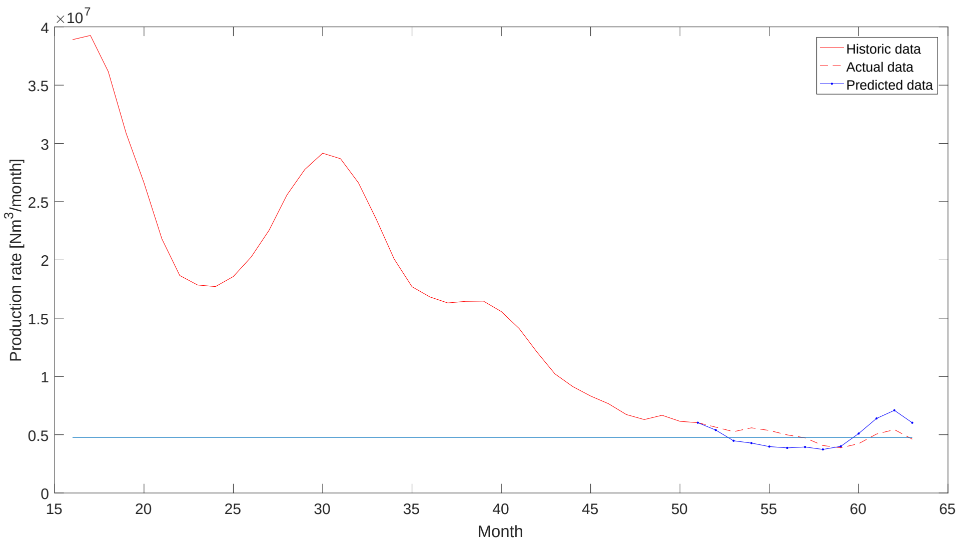


Figure 9: The actual and the predicted flow rate by ANN for well 1. When the well falls below the Coleman Critical rate liquid loading occurs. Each month a new prediction is made until the actual moment of liquid loading is found.

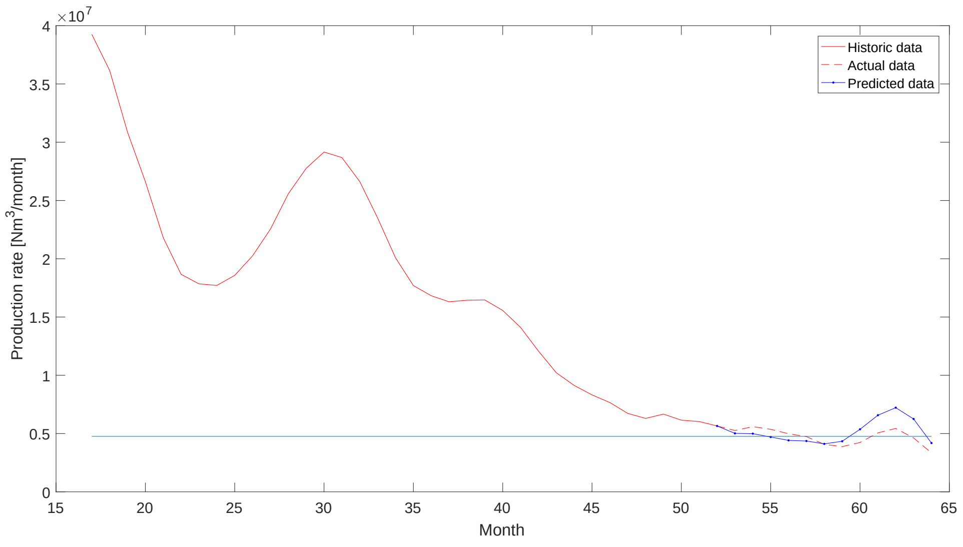


Figure 10: The actual and the predicted flow rate by ANN for well 1. When the well falls below the Coleman Critical rate liquid loading occurs. Each month a new prediction is made until the actual moment of liquid loading is found.

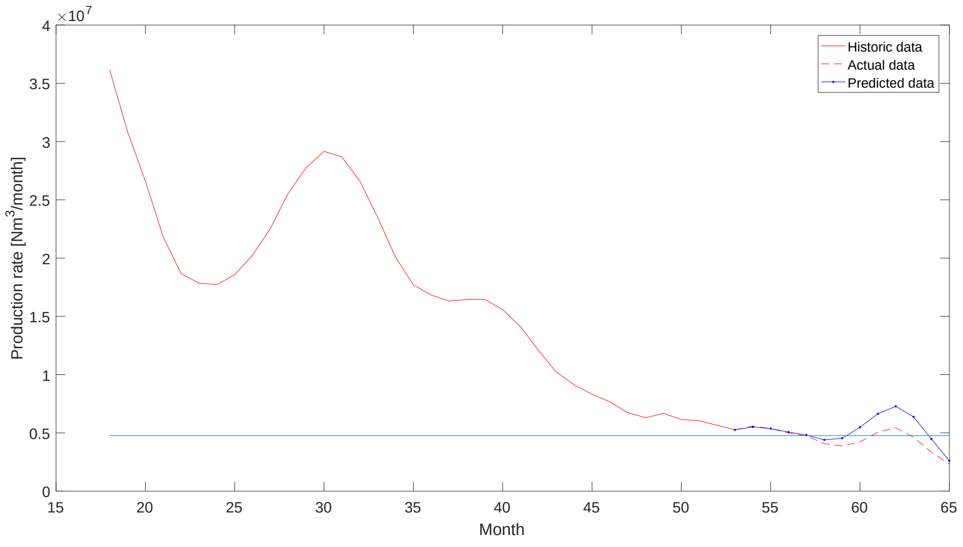


Figure 11: The actual and the predicted flow rate by ANN for well 1. When the well falls below the Coleman Critical rate liquid loading occurs. Each month a new prediction is made until the actual moment of liquid loading is found.

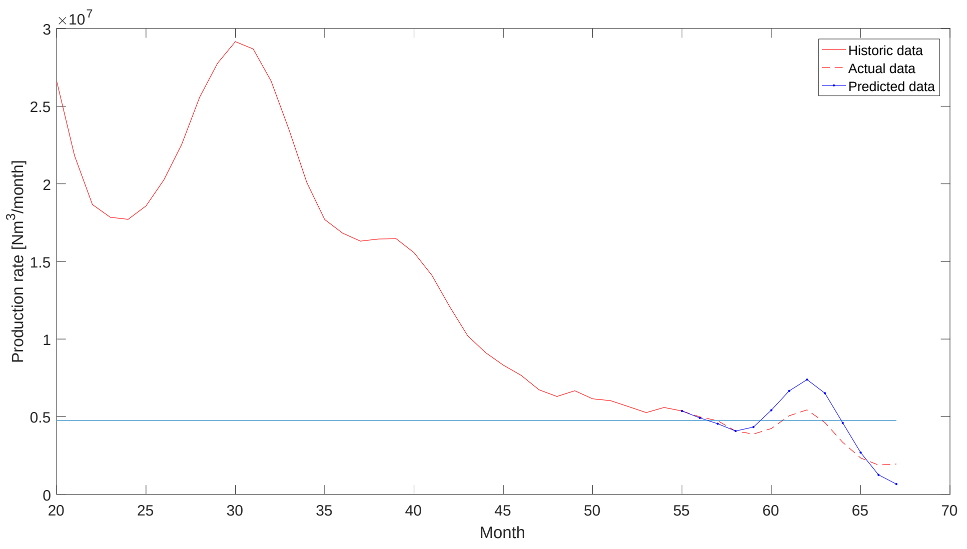


Figure 12: The actual and the predicted flow rate by ANN for well 1. When the well falls below the Coleman Critical rate liquid loading occurs. This happened in month 57, which was predicted accurately in 2 months in advance.

Appendix G

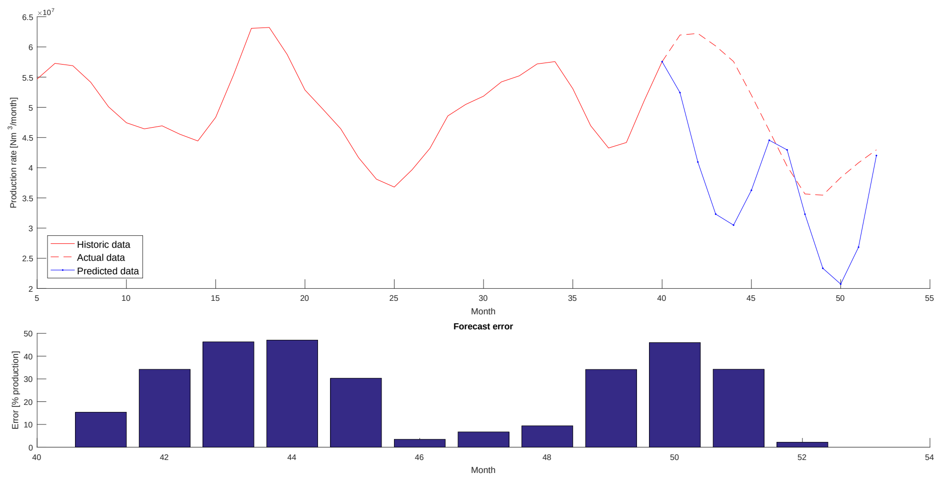


Figure 13: The actual and the predicted flow rate by ANN trained on the logarithmic data. The prediction did not improve.

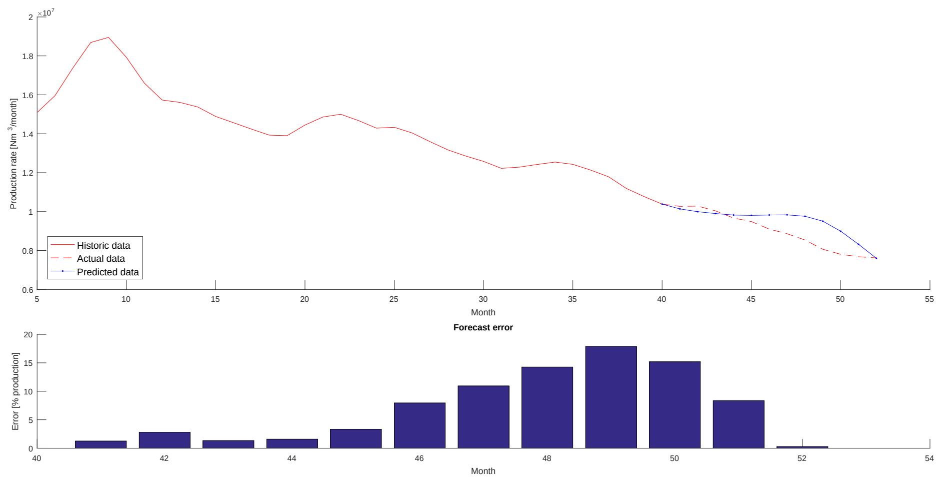


Figure 14: The actual and the predicted flow rate by ANN trained on the logarithmic data. The prediction did not improve.

Appendix H

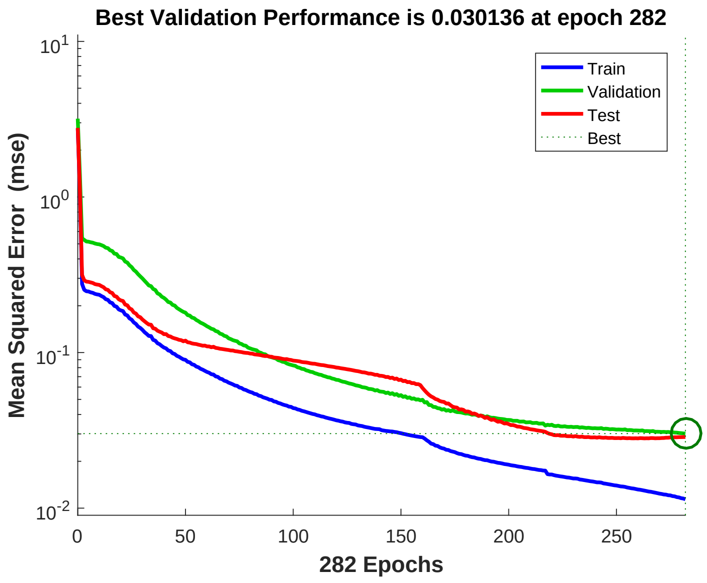


Figure 15: The regression plot for the log return network.

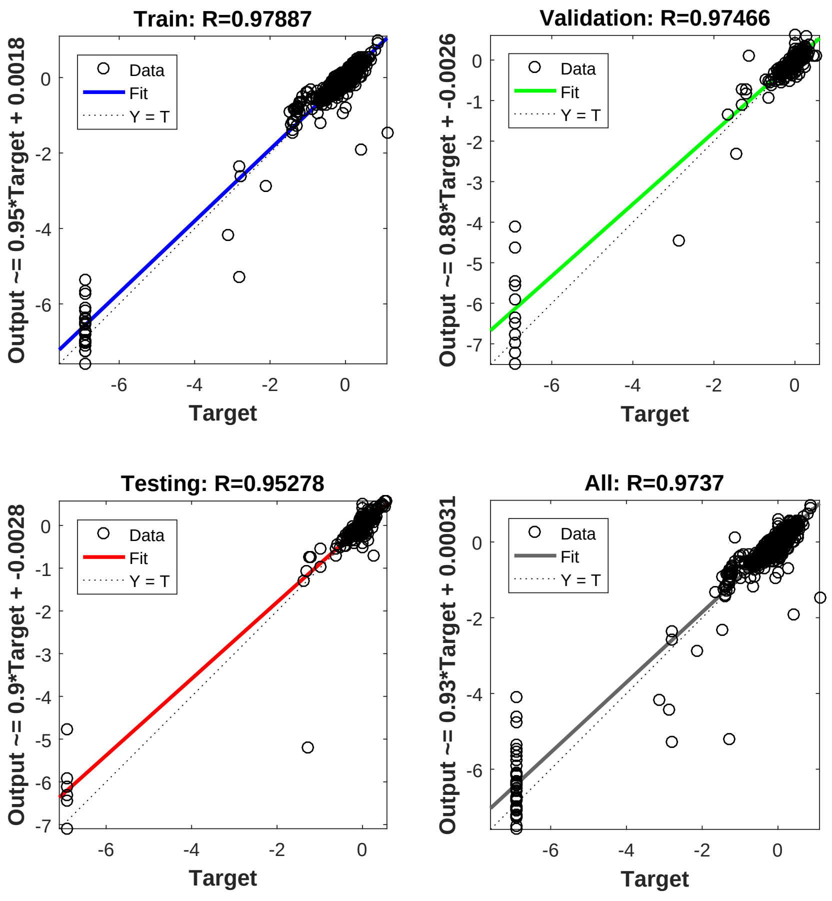


Figure 16: The regression plot for the log return network.

Appendix I

```matlab
close all;
%% Options
% Data processing
PFM_READ_DATA        = 1;
PFM_PRE_FILTER       = 1;
PFM_FILTER_DATA      = 1;    % Enable MATLAB data filter
PFM_LOGRETURN_DATA   = 0;    % Enable log-return data
PFM_LOG_DATA         = 0;    % Enable data in log scale
PFM_NORM_DATA        = 1;    % Enable normalize data
    PFM_NORM_SCALE_ONLY = 1;    % Don't assume data is according to normal dist, only scale data to domain [1 0]
PFM_TRAIN_NETWORK    = 0;    % Enable training the network
PFM_FORECAST         = 0;    % Enable forcasting the network
PFM_LIQUID_LOADING   = 1;    % Enable finding liquid-loading point for all wells
% Plotting
PFM_FILTER_SHOW      = 0;    % Show the filter results for all wells
PFM_DIFF_SHOW        = 0;    % Show the log-returns for all wells
PFM_LOG_SHOW         = 0;    % Show the lograthmic results for all wells
PFM_NORM_SHOW        = 0;    % Show the normalized data for all wells
PFM_ACF_SHOW         = 0;    % Show the autocorrelation function results
PFM_RESPONSE_SHOW    = 0;    % Show the open-loop response
PFM_FORECAST_SHOW    = 0;    % Show the closed-loop forcast
%% Read data from CSV file
if PFM_READ_DATA
    tic;
    % Start the timer
    filename        = '../Putten_productionrate_tubingsize_new_zondernaLLmoment_zondercumprod.csv';
    fid             = fopen(filename);
    % Open file 'filename'
    tline           = fgetl(fid);
    % Read the first line from file (contains well names)
    col_line         = fgetl(fid);
    fclose(fid);
    % Close the file
    wellNames       = strsplit(tline,';');
    % Split the well names seperated by a ';'
    colNames        = strsplit(col_line,';');
    data            = dlmread(filename,';',2,0);
    % Read 'filename' from line 2, column 0 onwards

    fcid            = fopen('../Critical_rate.csv');
    % Open file 'filename'
    tcline          = fgetl(fcid);
    % Read the first line from file (contains well names)
    fclose(fcid);
    critWellNames   = strsplit(tcline,';');
    critical_data    = dlmread('../Critical_rate.csv',';',1,0);

    colsPerWell      = 2;
    % Number of data-columns per well
    nrOfTargets      = 1;
    % Number of targets for the network
    % Calculate data set values
    nrTimesteps      = size(data,1);
    % The maximum number of timesteps in the data
    nrWells          = size(data,2) / colsPerWell;
    % Calculate number of wells
    fprintf('Read data from CSV file\t\t\t took %0.2f s\n', toc);
    % Print what was done and how long it took
    % Set the wellEnds
    wellEnd          = zeros(nrWells, 1);
    % This empty matrix will hold the nr of timesteps for each welll
    critical_rates   = zeros(nrWells, 1);
    for p=1:nrWells
        % For all the wells
        for pc=1:length(critWellNames)
            if strcmp(critWellNames{pc}, wellNames{p * colsPerWell})
                % If wellname is the same then well p == well pc
                critical_rates(p) = squeeze(critical_data(1,pc));
                % Store the critical rate of well pc in critical_rates for
                    well p
                break;
                % We saved the critical_rate, let's break the for loop
            end
        end
        wellEnd(p) = length(data(1:end-find(flip(data(:,p * colsPerWell))
            > 0,1, 'first')+1, p * colsPerWell));
        % Find the last element in data which is not 0
    end
end
%% Pre-filter the data (remove 0s)
if PFM_PRE_FILTER
    tic;
    for p=1:nrWells
        I = find(data(1:wellEnd(p), p * colsPerWell) == 0);
        i = 1;
        while ~isempty(I)
            index = I(i);
            if index == 1
                % Careful there is no previous data
                if length(I) == 1  || I(i+1) ~= index+1
                    data(index, p * colsPerWell) = data(index+1, p *
                        colsPerWell);
                else
                    curpos   = index;
                    curi     = i;
                    while curi < length(I) &&  curpos + 1 == I(curi+1)
                        curpos = curpos+1;
                        curi = curi+1;
                    end
                    data(index:curpos, p * colsPerWell) = repmat(data(
                        curpos+1, p * colsPerWell),length(index:curpos),1);
                    I(i:curi) = [];
                end
            elseif index == length(data(1:wellEnd(p), p * colsPerWell));
                % Careful there is no leading data
                % Leading zeros are cut off before this point, last zero
                % will never be at index length(data)
            else
                curpos   = index;
                curi     = i;
                while curi < length(I) && curpos+1 == I(curi+1)
                    curpos   = curpos + 1;
                    curi     = curi + 1;
                end
                data(index:curpos, p * colsPerWell) = repmat(data(index-1, p * colsPerWell), 1, length(index:curpos)) + (1:length(index:curpos)) .* (data(curpos+1, p * colsPerWell) - data(index-1, p * colsPerWell))/length(index:curpos);
                I(i:curi) = [];
            end
        end
    end
    fprintf('Pre-filtered data\t\t\t\t took %0.2f s\n', toc);
end
%% Filter the data (see references below)
for p=1:nrWells
data(wellEnd(p)+1:wellEnd(p)+ 20, p * colsPerWell)   = 1;
data(wellEnd(p)+1:wellEnd(p)+ 20, p * colsPerWell-1) = data(wellEnd(p), colsPerWell * p - 1);
end
wellEnd     = wellEnd + 20;
nrTimesteps = nrTimesteps + 20;

if PFM_FILTER_DATA
    tic;
    % Start the timer
    filterData  = zeros(size(data));
    % Empty matrix to hold all the filtered data
    windowSize  = 7;
    % Filter window size
    b           = (1/windowSize) * ones(windowSize,1);
    % Equal filter
    a           = 1;
    % Filter parameter a
    for p=1:nrWells
        % For all wells
        for c=1:(colsPerWell-1)
            filterData(1:wellEnd(p), p * colsPerWell -c)    = data(1:wellEnd(p), p * colsPerWell - c);
            % Also copy over the other columns
        end
        filterData(1:wellEnd(p), p * colsPerWell)       = smooth(1:wellEnd(p), data(1:wellEnd(p), p * colsPerWell), windowSize, 'lowess'); % Lowess filter
    end
    filterData(1:(windowSize-1),:) = [];
    % Remove the first 'windowsize-1' samples from the data
    wellEnd      = wellEnd - (windowSize - 1);
    % The wellEnds will also have to be reduces by 'windowsize-1'
    nrTimesteps = nrTimesteps - (windowSize - 1);
    % So will the timesteps
    fprintf('Filtered the data\t\t\t\t took %0.2f s\n', toc);
    % Print what was done and how long it took
else
    windowSize  = 1;
    % Windowsize is used further on in the script, so set it to 1
    filterData  = data;
    % Bypass the filter
end
I = find(filterData < 0);
if ~isempty(I)
    fprintf('\nWARNING: Filtered data contains negative elements.\n\n');
    filterData(filterData <0) = 0;
    % Remove all zeros from data
end
%% Differentiate the data
if PFM_LOGRETURN_DATA
    tic;
    % Start the timer
    wellEnd = wellEnd - 1;
    % The returns will have 1 sample less
    nrTimesteps = nrTimesteps - 1;
    % The returns will have 1 sample less
    diffData = zeros(size(filterData,1)-1,size(filterData,2));
    % Create a matrix to hold the differential data
    for p=1:nrWells
        % For all the wells
        % Calculate the log-return
        diffData(1:wellEnd(p), p * colsPerWell) = log(1 + max(-0.999,diff(
            filterData(1:wellEnd(p)+1, p * colsPerWell))./max(1,filterData
            (1:wellEnd(p), p * colsPerWell))));
        for c=1:(colsPerWell-1)
            diffData(1:wellEnd(p), p * colsPerWell -c) = filterData(1:
                wellEnd(p), p * colsPerWell - c);
            % Also copy over the other columns
        end
    end
    fprintf('Calculated log-returns\t\t\t took %0.2f s\n', toc);
    % Print what was done and how long it took
else
    diffData = filterData;
    % Bypassing log-returns
end
%% Logarithmic scale
if PFM_LOG_DATA
    tic;
    % Start the timer
    logData      = zeros(size(diffData));
    % Create empty matrix for the logarithmic data
    for p=1:nrWells
        % For all well
        for c=1:(colsPerWell-1)
            logData(1:wellEnd(p), p * colsPerWell -c)  = diffData(1:
                wellEnd(p), p * colsPerWell - c);
            % Also copy over the other columns
        end
        logData(1:wellEnd(p), p * colsPerWell)     = log10(diffData(1:
            wellEnd(p), p * colsPerWell));
        % Logarithmically scale the data
    end
    fprintf('Scaled data logarithmically\t\t took %0.2f s\n', toc);
    % Print what was done and how long it took
else
    logData = diffData;
    % Bypass the logarithmic scale
end
%% Normalize the data
if PFM_NORM_DATA
    tic;
    % Start the timer
    normData    = zeros(size(data));
    % Create empty matrix to store the normalized data
    wellMean    = zeros(1,nrWells);
    % Create empty matrix to store the means per well
    wellStd     = zeros(1,nrWells);
    % Create empty matrix to store the std per well
    for p=1:nrWells
        % For all wells
        tmpTarget   = logData(1:wellEnd(p), p * colsPerWell);
        % Store the production data in a temporary matrix for easy access
        if PFM_NORM_SCALE_ONLY
            wellMean(p) = min(tmpTarget);
            % Store the production mean for this well
            wellStd(p)  = max(tmpTarget) - min(tmpTarget);
            % Store the production std for this well
        else
            wellMean(p) = mean(tmpTarget);
            % Store the production mean for this well
            wellStd(p)  = std(tmpTarget);
            % Store the production std for this well
        end
        for c=1:(colsPerWell-1)
            if strcmp('Cum. Production',colNames{p * colsPerWell -c})
                normData(1:wellEnd(p), p * colsPerWell -c) = logData(1:
                    wellEnd(p), p * colsPerWell - c) ./ logData(wellEnd(p),
                     p * colsPerWell - c);
                % Also copy over the other columns
            else
                normData(1:wellEnd(p), p * colsPerWell -c) = logData(1:
                    wellEnd(p), p * colsPerWell - c);
                % Also copy over the other columns
            end
        end
        normData(1:wellEnd(p), p * colsPerWell)        = (tmpTarget -
            wellMean(p));
        % Normalize the production data based on mean and std
    end
    max_wellStd = max(wellStd);

```

```
        for p=1:nrWells
            wellStd(p) = log10(max_wellStd)/log10(wellStd(p))*wellStd(p);%
                /10;
            normData(1:wellEnd(p), p * colsPerWell) = normData(1:wellEnd(p
                ), p * colsPerWell) / wellStd(p);
        end
    fprintf('Normalized data\t\t\t\t\t took %0.2f s\n', toc);
    % Print what was done and how long it took
else
    normData = logData;
    % Bypass data normalization
end
%% Plot data
if PFM_FILTER_SHOW
    for p=1:nrWells
        f = figure();
        hold all;
        % Plot the raw target data
        plot(1:(wellEnd(p)+(windowSize-1)), data(1:(wellEnd(p)+(windowSize
            -1)), p * colsPerWell),'x');
        % Plot the filtered target data
        plot(windowSize:(wellEnd(p)+(windowSize-1)), filterData(1:wellEnd(
            p), p * colsPerWell),'-');
        hold off;
        title(wellNames{p * colsPerWell});
        xlabel('Month');
        ylabel('Production Rate [Nm^3/month]');
        legend('Raw data','Smoothed data');
    end
end
if PFM_DIFF_SHOW
    for p=1:nrWells
        f = figure();
        hold all;
        plot(1:wellEnd(p), diffData(1:wellEnd(p), p * colsPerWell));
        hold off;
        title(wellNames{p * colsPerWell});
        xlabel('Month');
        ylabel('Production Rate [Nm^3/month]');
    end
end
if PFM_LOG_SHOW
    for p=1:nrWells
        % Plot the log target data
        f = figure();
        hold all;
        plot(1:wellEnd(p), logData(1:wellEnd(p), p * colsPerWell),'-');
        hold off;
        title(wellNames{p * colsPerWell});
        xlabel('Month');
        ylabel('Production Rate [Nm^3/month]');
    end
end
if PFM_NORM_SHOW
    for p=1:nrWells
        % Plot the norm target data
        f = figure();
        hold all;
        plot(1:wellEnd(p), normData(1:wellEnd(p), p * colsPerWell),'-');
        hold off;
        title(wellNames{p * colsPerWell});
        xlabel('Month');
        ylabel('Production Rate [Nm^3/month]');
    end
end
%% Plot auto-correlation of production data
if PFM_ACF_SHOW
    maxLag = 72;
    % Maximum lag to test
    for p=1:nrWells
        % For all wells
        figure();
        % New figure
        acf(normData(1:wellEnd(p), p * colsPerWell), maxLag);
        % Show the auto-correlation
    end
end
%% Format the data according to NN
if PFM_TRAIN_NETWORK
    tic;
    % Pre-define the input and target cells
    input_seq   = cell(colsPerWell - nrOfTargets, nrTimesteps);
    target_seq  = cell(1, nrTimesteps);
    % For all timestepts in the data
    for time=1:nrTimesteps
        tmpInput    = NaN * zeros(colsPerWell - nrOfTargets, nrWells);
        tmpTarget   = NaN * zeros(nrOfTargets, nrWells);
        % For all wells in the data
        for p=1:nrWells
            % For all Elements per well (except for the target which is the last column)
            if time <= wellEnd(p)
                for e=1:(colsPerWell - nrOfTargets)
                    % Store this Element for this timestep and well
                    tmpInput(e,p)  = normData(time, (p-1) * colsPerWell + e);
                end
                % Store the Target for this timestep and well
                for e=1:nrOfTargets
                    tmpTarget(e,p) = normData(time, (p-1) * colsPerWell + (colsPerWell-nrOfTargets) + e);
                end
            end
        end
        % Store in overall structure
        input{time}    = tmpInput;
        target{time}   = tmpTarget;
        clear tmpInput tmpTarget;
    end
    fprintf('Prepared input data and targets\t took %0.2f s\n', toc);
end
%% Start NN
```

```
if PFM_TRAIN_NETWORK
    tic;
    X = input;
    T = target;
    trainFcn = 'trainlm';  % Scaled conjugate gradient backpropagation.
    % Create a Nonlinear Autoregressive Network with External Input
    pastSteps       = 36;
    % Number of past months used for prediction
if ~isempty(colsPerWell*find(wellEnd < pastSteps))
    fprintf('\nWarning, several wells dont meet the required %i past steps
        :\n', pastSteps);
    wrongWells = find(wellEnd < pastSteps);
    for i = 1:length(wrongWells)
        fprintf('%s only %i months\n', wellNames{colsPerWell * wrongWells(
            i)}, wellEnd(wrongWells(i)))
    end
    fprintf('\n');
end
    inputDelays      = 1:pastSteps;
    feedbackDelays   = 1:pastSteps;
    hiddenLayerSize = [75 75];
    % Number of neurons used in Network
    net              = narxnet(inputDelays,feedbackDelays,hiddenLayerSize,'
        open',trainFcn);
    net.layers{1}.transferFcn = 'tansig';
    net.layers{2}.transferFcn = 'tansig';

    net.inputs{1}.processFcns = {'removeconstantrows'};
    net.inputs{2}.processFcns = {'removeconstantrows'};

    [x,xi,ai,t] = preparets(net,X,{},T);
    net.divideFcn                 = 'divideint';
    % Divide data interleaved
    net.divideMode               = 'time';
    % Divide up every time

    net.divideParam.trainRatio   = 70/100;
    net.divideParam.valRatio     = 15/100;
    net.divideParam.testRatio    = 15/100;

    net.trainParam.max_fail = 25;
    net.trainParam.epochs = 1000;

    % Choose a Performance Function
    net.performFcn = 'mse';
    % Mean Squared Error

    % Choose Plot Functions
    net.plotFcns = {'plotperform','plottrainstate', 'ploterrhist', ...
        'plotregression', 'plotresponse', 'ploterrcorr', 'plotinerrcorr'};

    % Train the Network
    [net,tr] = train(net,x,t,xi,ai);

    % Test the Networkplotp
    y           = net(x,xi,ai);
    e           = gsubtract(t,y);
    performance = perform(net,t,y);

    % Recalculate Training, Validation and Test Performance
    trainTargets        = gmultiply(t,tr.trainMask);
    valTargets          = gmultiply(t,tr.valMask);
    testTargets         = gmultiply(t,tr.testMask);
    trainPerformance    = perform(net,trainTargets,y);
    valPerformance      = perform(net,valTargets,y);
    testPerformance     = perform(net,testTargets,y);

    fprintf('\nPerformance           : %0.5e\n',performance);
    fprintf('Train performance     : %0.5e\n',trainPerformance);
    fprintf('Validation performance : %0.5e\n',valPerformance);
    fprintf('Test performance      : %0.5e\n\n',testPerformance);

    % Plots
    figure, plotperform(tr)
    if PFM_RESPONSE_SHOW
        for p=1:nrWells
            figure, plotresponse(t,y,'sampleIndex',p);
            title(wellNames{p * colsPerWell});
        end
    end
    fprintf('Finished training neural network took %0.2f s\n', toc);
end
%% Multi-step Prediction
if PFM_FORECAST
    tic;
    knownSamples = pastSteps;
    % Change if you want the first part to predict with open-loop
    futurePredictions = 12;
    % Number of predicted months

    if knownSamples + futurePredictions > nrTimesteps
        disp('WARNING: more predictions requested than data available');
        futurePrecictions = nrTimesteps - knownSamples;
        % Maximum number of prediction possible from data
    end

    startMonth                = 40;
    % Month in which prediction starts
    knownOutputTimesteps      = (startMonth - pastSteps + 1):(knownSamples +
         startMonth - pastSteps);
    predictOutputTimesteps    = (startMonth - pastSteps + 1):(knownSamples +
         startMonth - pastSteps) + futurePredictions;
    X1                        = X(knownOutputTimesteps);
    T1                        = T(knownOutputTimesteps);
    [x1,xio,aio, t1c]         = preparets(net,X1,{},T1);
    [y1,xfo,afo]              = net(x1,xio,aio);

    X2                        = X(predictOutputTimesteps);
    T2                        = T(predictOutputTimesteps);
    [netc,xic,aic]            = closeloop(net,xfo,afo);
    [x2c,x2ic,a2ic,t2c]       = preparets(netc,X2,{},T2);
    [y2,xfc,afc]              = netc(x2c,x2ic,a2ic);
    multiStepPerformance    = perform(net,t2c,y2);
    fprintf('\nMulti-step performance : %0.5f\n\n',multiStepPerformance);
    %% Reformat output data
    wellPrediction  = zeros(length(y2), nrWells, nrOfTargets);
    true_prediction = zeros(length(y2), nrWells, nrOfTargets);
    for p=1:nrWells
        for ti=1:length(y2)
            for tar=1:nrOfTargets
                wellPrediction(ti,p,tar) = y2{ti}(tar,p);
            end
        end
        if PFM_LOGRETURN_DATA
            if PFM_NORM_DATA
                true_prediction(:,p,1) = data(1,p * colsPerWell) * cumprod(exp(wellPrediction(:,p,1) * wellStd(p) + wellMean(p)) - 1);
            else
                true_prediction(:,p,1) = data(1,p * colsPerWell) * cumprod(exp(wellPrediction(:,p,1)) - 1);
            end
        else
            if PFM_NORM_DATA && PFM_LOG_DATA
                true_prediction(:,p,1) = 10.^(wellPrediction(:,p,1) * wellStd(p) + wellMean(p));
            else
                if PFM_NORM_DATA
                    true_prediction(:,p,1) = wellPrediction(:,p,1) * wellStd(p) + wellMean(p);
                else
                    if PFM_LOG_DATA
                       true_prediction(:,p,1) =  10.^(wellPrediction(:,p,1));
                    else
                        true_prediction(:,p,1) = wellPrediction(:,p,1);
                    end
                end
            end
        end
    end
    fprintf('Finished forcasting data\t took %0.2f s\n', toc);
end
%% Plot forecast
if PFM_FORECAST_SHOW
    error           = zeros(size(wellPrediction, 1), nrWells, nrOfTargets);
    rel_error       = zeros(size(wellPrediction, 1), nrWells, nrOfTargets);
    for p=1:nrWells
        figure();
        subplot(3,1,[1 2]);
        hold all;
        % Plot the raw target data
        plot(predictOutputTimesteps(1:end-futurePredictions), data(predictOutputTimesteps(1:end-futurePredictions) + (windowSize-1), p * colsPerWell),'bx');
        plot(predictOutputTimesteps(end-futurePredictions:end), data(
                predictOutputTimesteps(end-futurePredictions:end) + (windowSize
                -1), p * colsPerWell),'bx');
            % Plot the filtered target data
            curMonthData = data(startMonth + windowSize - 1, p * colsPerWell);
            if PFM_FILTER_DATA
                plot(predictOutputTimesteps(1:end-futurePredictions),
                    filterData(predictOutputTimesteps(1:end-futurePredictions),
                     p * colsPerWell),'r-');
                plot(predictOutputTimesteps(end-futurePredictions:end),
                    filterData(predictOutputTimesteps(end-futurePredictions:end
                    ), p * colsPerWell),'r--');
                curMonthData = filterData(startMonth, p * colsPerWell);
            end
            % Plot the predicted output data
            plot(predictOutputTimesteps((end - size(true_prediction,1)):end),
                [curMonthData; true_prediction(:,p,1)],'g.-');
            title(wellNames{p * colsPerWell});
            xlabel('Month');
            ylabel('Production [Nm^3/month]');
            if PFM_FILTER_DATA
                legend('Raw historic data','Actual raw data','Filtered
                    historic data','Actual filtered data','predicted data', '
                    Location', 'southwest');
            else
                legend('Raw historic data','Actual raw data','predicted data',
                     'Location', 'southwest');
            end
            subplot(3,1,3);
            hold all;
            error(:,p,1)      = true_prediction(:,p,1) - filterData(
                predictOutputTimesteps(end-futurePredictions+1:end), p *
                colsPerWell);
            rel_error(:,p,1)  = abs(error(:,p,1)) ./ filterData(
                predictOutputTimesteps(end-futurePredictions+1:end), p *
                colsPerWell);
            bar(predictOutputTimesteps(end-futurePredictions+1:end),rel_error
                (:,p,1).*100);
            xlabel('Month');
            ylabel('Error [% production]');
            title('Forecast error');
        end
end
%% Regression plots
plotregression(t(tr.trainInd),y(tr.trainInd),'Train',t(tr.valInd),y(tr.
    valInd),'Validation',t(tr.testInd),y(tr.testInd),'Testing',t,y,'All');
%% Plot
figure();
hold all;
H1 = plot([tr.trainInd; tr.trainInd],[zeros(size(data(tr.trainInd,2)))';
    data(tr.trainInd,2)'],'b','LineWidth',2);
H2 = plot([tr.valInd; tr.valInd],[zeros(size(data(tr.valInd,2)))'; data(tr
    .valInd,2)'],'g','LineWidth',2);
H3 = plot([tr.testInd; tr.testInd],[zeros(size(data(tr.testInd,2)))'; data
    (tr.testInd,2)'],'r','LineWidth',2);
set(gca,'XLim',[0 wellEnd(1)]);
title('Data division between Training, Validation and Testing sets');
```

```
xlabel('Month');
ylabel('Production rate [Nm^3/month]');
legend([H1(1) H2(1) H3(1)],{'Training' 'Validation' 'Testing'});
hold off;
%% Find liquid loading point
if PFM_LIQUID_LOADING
    wellLL              = zeros(1,nrWells);
    wellPredRes         = NaN*ones(nrWells,futurePredictions);
for p=1:nrWells
    wellLL(p) = find(filterData(:,p*colsPerWell) < critical_rates(p),1,'
        first');
end
    for p=1:nrWells
        % For all wells
        critical_rate        = critical_rates(p);
        % Critical rate for all wells
        fprintf(strcat(['\nFinding critical rate %0.2f for ' wellNames{p *
            colsPerWell} '\n']),critical_rate);
        % Print on which well we're calculating
        startMonth           = pastSteps+1;
        % The initial start month is the required previous months + 1
        liquid_loading       = false;
        % Set the indicator for liquid loading to false
        futurePredictions    = 12;
        % How many months should we predict?
        knownSamples         = pastSteps;
        % No open-loop predictions, only future data
        while liquid_loading==false && startMonth <= (wellEnd(p) -
            futurePredictions)
            % While there is no liquid loading, and there is still data
                for the well to evaluate prediction
            if filterData(startMonth, colsPerWell * p) < critical_rate
                % Is the filtered data below the critical rate?
                fprintf('Liquid loading occured in month %i\n',startMonth)
                    ;
                % Liquid loading occured
                liquid_loading = true;
                % Set the liquid loading indicator to true
                continue;
            end
            if knownSamples + futurePredictions > nrTimesteps
                % Test if we want more than there is data available
                disp('WARNING: more predictions requested than data
                    available');
                % Print a warning that we are changing the nr of future
                    predictions
                futurePrecictions = nrTimesteps - knownSamples;
                % Get the maximum nr of predictions possible to still
                    evaluate performance
            end
            % knownOutputTimesteps    = (startMonth - pastSteps + 1):(
                knownSamples + startMonth - pastSteps);
            % What is "known" to our network -> 'pastSteps' months before
                'startMonth'
            predictOutputTimesteps  = (startMonth - pastSteps + 1):(
                knownSamples + startMonth - pastSteps) + futurePredictions;
        % What do we want the network to predict -> from 'pastSteps'
            months before 'startMonth' to 'futurePredictions' months
            past 'startMonth'
        X1                          = X(knownOutputTimesteps);
        % Label the known input data as X1
        T1                          = T(knownOutputTimesteps);
        % Label the known output data (target) as T1
        [x1,xio,aio, t1c]           = preparets(net,X1,{},T1);
        % Prepare the network for data of this format
        [y1,xfo,afo]                = net(x1,xio,aio);
        % Create the network
        % Next the the network and its final states will be converted
            to closed-loop forms;
        X2                          = X(predictOutputTimesteps);
        % Label the 'to predict' input data as X2
        T2                          = T(predictOutputTimesteps);
        % Label the 'to predict' output data (target) as T2
        %[netc,xic,aic]       = closeloop(net,xfo,afo);
        [x2c,x2ic,a2ic,t2c]         = preparets(netc,X2,{},T2);
        % Prepare the network for data of this format
        [y2,xfc,afc]                = netc(x2c,x2ic,a2ic);
        % Create the closed loop network
        multiStepPerformance        = perform(net,t2c,y2);
        % Evaluate it's performance
        wellPrediction              = zeros(length(y2), 2);
        % Create empty matrix to hold the predictions for this well
        true_prediction             = zeros(length(y2), 2);
        % Create empty matrix to hold the predictions for this well in
             L
        for t=1:length(y2)
            % For all predictions
            for tar=1:nrOfTargets
            wellPrediction(t,tar) = y2{t}(tar,p);
            % Store the prediction in wellPrediction
            end
        end
        if PFM_LOGRETURN_DATA
            if PFM_NORM_DATA
                true_prediction(:,1) = data(1,p) * cumprod(exp(
                    wellPrediction(:,1) * wellStd(p) + wellMean(p)) -
                    1);
                % Calculate back to L
            else
                true_prediction(:,1) = data(1,p) * cumprod(exp(
                    wellPrediction(:,1)) - 1);
                % Calculate back to L
            end
        else
            if PFM_NORM_DATA && PFM_LOG_DATA
                true_prediction(:,1) = 10.^(wellPrediction(:,1) *
                    wellStd(p) + wellMean(p));
                % Calculate back to L
            else
                if PFM_NORM_DATA
                    true_prediction(:,1) = wellPrediction(:,1) *
                        wellStd(p) + wellMean(p);
                % Calculate back to L
            else
                if PFM_LOG_DATA
                    true_prediction(:,1) =  10.^(wellPrediction(:,1));
                    % Calculate back to L
                else
                    true_prediction(:,1) = wellPrediction(:,1);
                    % Calculate back to L
                end
            end
        end
    end
    I = find(true_prediction(:,1) < critical_rate, 1, 'first');
    % Find the index of the first instance of true_prediction that is below the 'critical_rate'
    if ~isempty(I)
        % If the index is not empty there were predictions below 'critical_rate'
        if startMonth >= (wellLL(p) - futurePredictions)
            wellPredRes(p,I) = startMonth + I - wellLL(p);
        end
        liquid_loading_month = startMonth + I;
        % Estimated liquid loading is 'I' months after the current month we're examining -> startMonth + I
        fprintf('In month %i Liquid loading is suspected in month %i\n',startMonth,liquid_loading_month);
        % Print month that is suspecting liquid loading
        figure(p);
        % Open a figure to show this prediction
        % Plot the raw target data of the known part
        plot(predictOutputTimesteps(1:end-futurePredictions), data(predictOutputTimesteps(1:end-futurePredictions) + (windowSize-1), p * colsPerWell),'bx');
        hold all;
        % Plot the raw target data of the predicted part
        plot(predictOutputTimesteps(end-futurePredictions:end), data(predictOutputTimesteps(end-futurePredictions:end) + (windowSize-1), p * colsPerWell),'bx');
        curMonthData = data(startMonth + windowSize - 1, p * colsPerWell);
        if PFM_FILTER_DATA
            % Plot the filtered target data of the known part
            plot(predictOutputTimesteps(1:end-futurePredictions), filterData(predictOutputTimesteps(1:end-futurePredictions), p * colsPerWell),'r-');
            % Plot the filtered target data of the predicted part
            plot(predictOutputTimesteps(end-futurePredictions:end), filterData(predictOutputTimesteps(end-futurePredictions:end), p * colsPerWell),'r--');
            curMonthData = filterData(startMonth, p * colsPerWell);
        end
        % Plot the predicted output data
        plot(predictOutputTimesteps((end - length(true_prediction)):end), [curMonthData; true_prediction(:,1)],'g.-');
```

```
                plot([predictOutputTimesteps(1) predictOutputTimesteps(end)],[critical_rate critical_rate]);
                hold off;
                title(wellNames{p * colsPerWell});
                xlabel('Month');
                ylabel('Production rate [Nm3/month]');
                drawnow;
                pause;
                % Pause the script to allow inspection of the prediction
            end
            startMonth = startMonth + 1;
            % Increase our startmonth with 1
        end
    end
end
```

```
close all;
%% Load dataset
load('./loopdata/2-layers-75-neurons-tansig-trial-1.mat');
%% Set parameters
pastSteps           = 36;                                                    % Number of past months used for prediction
futurePredictions   = 12;
PFM_LIQUID_LOADING  = 1;
%% Find liquid loading point
if PFM_LIQUID_LOADING
    startMonth                  = pastSteps+1;
    knownSamples                = pastSteps;
    knownOutputTimesteps        = (startMonth - pastSteps + 1):(knownSamples + startMonth - pastSteps);
    X1                          = X(knownOutputTimesteps);
    T1                          = T(knownOutputTimesteps);
    [x1,xio,aio, t1c]           = preparets(net,X1,{},T1);
    [y1,xfo,afo]                = net(x1,xio,aio);
    [netc,xic,aic]              = closeloop(net,xfo,afo);

    wellLL              = zeros(1,nrWells);
    wellPredRes         = NaN*ones(nrWells,futurePredictions);
    for p=1:nrWells
        wellLL(p) = find(filterData(:,p*colsPerWell) < critical_rates(p),1,'first');
    end
    for p=1:nrWells
        % For all wells
        critical_rate       = critical_rates(p);
        % Critical rate for all wells
        fprintf(strcat(['\nFinding critical rate %0.2f for ' wellNames{p * colsPerWell} '\n']),critical_rate);
        % Print on which well we're calculating
        startMonth          = pastSteps+1;
        % The initial start month is the required previous months + 1
        liquid_loading      = false;
        % Set the indicator for liquid loading to false
        futurePredictions   = 12;
        % How many months should we predict?
        knownSamples        = pastSteps;
        % No open-loop predictions, only future data
        while liquid_loading==false && startMonth <= (wellEnd(p) - futurePredictions)
            % While there is no liquid loading, and there is still data for the well to evaluate prediction
            if filterData(startMonth, colsPerWell * p) < critical_rate
                % Is the filtered data below the critical rate?
                fprintf('Liquid loading occured in month %i\n',startMonth);
                % Liquid loading occured
                liquid_loading = true;
                % Set the liquid loading indicator to true
                continue;
            end
            if knownSamples + futurePredictions > nrTimesteps
                % Test if we want more than there is data available
                disp('WARNING: more predictions requested than data available');
                % Print a warning that we are changing the nr of future predictions
                futurePrecictions = nrTimesteps - knownSamples;
                % Get the maximum nr of predictions possible to still evaluate performance
            end
            end
            startMonth = startMonth + 1;
            % Increase our startmonth with 1
        end
end
%% Make plot
figure();
[ax,H1,H2] = plotyy(repmat(1:futurePredictions,3,1)',[mean(wellPredRes,'omitnan'); mean(wellPredRes,'omitnan')-std(wellPredRes,'omitnan'); mean(wellPredRes,'omitnan')+std(wellPredRes,'omitnan')]',1:futurePredictions,sum(~isnan(wellPredRes)));
set(H1,'Color',[0.3 0.3 0.3]);
set(H2,'Color',[1 0 0]);
set(ax(2),'YColor',[1 0 0]);
axes(ax(1));
axis([0.5 futurePredictions+0.5 -futurePredictions futurePredictions]);
set(ax(1),'YTick',-futurePredictions:2:futurePredictions);
axes(ax(2));
axis([0.5 futurePredictions+0.5 0 nrWells]);
set(H1(2),'LineStyle','--');
grid minor;
xlabel(ax(1),'Months prior to liquid loading [months]');
ylabel(ax(1),'Prediction error [months]');
ylabel(ax(2),'Number of wells predicted [-]');
set(H1(3),'LineStyle','--');
set(ax(1),'XTick',1:futurePredictions);
set(ax(1),'XTickLabel',{12 11 10 9 8 7 6 5 4 3 2 1});
legend(ax(1),'Average forecast error (\mu)','\mu + \sigma','\mu - \sigma','Location','NorthWest');
axes(ax(1));
hold all
```

```
for p = 1:nrWells
    H = scatter(1:12,wellPredRes(p,:),'filled','CData',[0.3 0.3 0.3],'
        SizeData',8, 'jitter','on', 'jitterAmount',0.05);
end
putNr = 1;
figure();
plot(1:12,wellPredRes(putNr,:),'LineWidth',1.5)
axis([0.5 futurePredictions+0.5 -futurePredictions futurePredictions]);
grid minor;
title(['Well ' num2str(putNr)]);
xlabel('Months prior to liquid loading [months]');
ylabel('Prediction error [months]');
set(gca,'XTick',1:futurePredictions);
set(gca,'XTickLabel',{12 11 10 9 8 7 6 5 4 3 2 1});
```

```
fileList = dir('./loopdata');
resultMatrix = nan(2,2,5,250);
for f=3:length(fileList)
    load(['./loopdata/' fileList(f).name],'activationFunction','
        hiddenLayerSize','af','trial','performance');
    nrLayers    = length(hiddenLayerSize);
    nrNeurons   = hiddenLayerSize(1);
    fprintf('%d layers %d neurons %s trial %d: %e\n',nrLayers, nrNeurons,
        activationFunction, trial, performance);
    resultMatrix(nrLayers,af,trial,nrNeurons) = performance;
end
%% Plot
layer1Results = squeeze(resultMatrix(1,:,:,:));
layer1Results_tansig = min(squeeze(layer1Results(1,:,:)),[],'omitnan');
layer1Results_logsig = min(squeeze(layer1Results(2,:,:)),[],'omitnan');

layer1Results_tansig_x = find(~isnan(layer1Results_tansig));
layer1Results_logsig_x = find(~isnan(layer1Results_logsig));

layer1Results_tansig_y = layer1Results_tansig(layer1Results_tansig_x);
layer1Results_logsig_y = layer1Results_logsig(layer1Results_logsig_x);

layer2Results = squeeze(resultMatrix(2,:,:,:));
layer2Results_tansig = min(squeeze(layer2Results(1,:,:)),[],'omitnan');
layer2Results_logsig = min(squeeze(layer2Results(2,:,:)),[],'omitnan');

layer2Results_tansig_x = find(~isnan(layer2Results_tansig));
layer2Results_logsig_x = find(~isnan(layer2Results_logsig));

layer2Results_tansig_y = layer2Results_tansig(layer2Results_tansig_x);
layer2Results_logsig_y = layer2Results_logsig(layer2Results_logsig_x);

figure();
min_comp_plot = min([length(layer1Results_tansig_y) length(
    layer1Results_logsig_y) length(layer2Results_tansig_y) length(
    layer2Results_logsig_y)]);
hold all;
plot(layer1Results_tansig_x(1:min_comp_plot), layer1Results_tansig_y(1:
    min_comp_plot),'Color',[      0    0.4470    0.7410],'DisplayName','1
    layer tansig','Marker','.','MarkerSize',12);
plot(layer1Results_logsig_x(1:min_comp_plot), layer1Results_logsig_y(1:
    min_comp_plot),'Color',[0.8500    0.3250    0.0980],'DisplayName','1
    layer logsig','Marker','.','MarkerSize',12);
plot(layer2Results_tansig_x(1:min_comp_plot), layer2Results_tansig_y(1:
    min_comp_plot),'Color',[0.9290    0.6940    0.1250],'DisplayName','2
    layer tansig','Marker','.','MarkerSize',12);
plot(layer2Results_logsig_x(1:min_comp_plot), layer2Results_logsig_y(1:
    min_comp_plot),'Color',[0.4940    0.1840    0.5560],'DisplayName','2
    layer logsig','Marker','.','MarkerSize',12);
hold off;
grid minor;
title('Network performance');
xlabel('Number of neurons [-]');
ylabel('Network performance MSE [-]');
legend show;

figure();
plot(layer2Results_tansig_x, layer2Results_tansig_y,'DisplayName','2 layer
     tansig','Color',[0.9290    0.6940    0.1250],'Marker','.','MarkerSize'
    ,12);
grid minor;
title('2 layer tansig network performance');
xlabel('Number of neurons [-]');
ylabel('Network performance MSE [-]');
legend show;
```

Bibliography

R. Babuska. *Knowledge-Based Control Systems*. TU Delft, 2010.

O. Binli. Overview of solutions to prevent liquid loading problems in gas wells. 2009.

C. N. C. Chakra, K. Y. Song, D. N. Saraf, and M. M. Gupta. Production forecasting of petroleum reservoir applying higher-order neural networks (honn) with limited reservoir data. *International Journal of Computer Applications*, 2013.

S.B. Coleman, H.B. Clay, D.G. McCurdy, and H. L. Norris. A new look at predicting gas-well load-up. *Journal of Petroleum Technology*, 1991.

N.S. Ghazwan. Application of neural network to optimize oil field production. *Asians Transactions on Engineering*, 2012.

D.E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. 1989.

Z.J.G. Gromotka. The stability region of the tubing performance relation curve. 2015.

B. Guo and A. Ghalambor. *Natural Gas Engineering Hand Book*. GULF Publishing Company, 2005.

B. Guo, A. Ghalambor, and C. Xu. A systematic approach to predicting liquid loading in gas wells. *SPE Production & Operations*, 2005.

W.A. Habiballah, R.A. Startzman, and A. Barrufet. Use of neural networks for prediction of vapor/liquid equilibrium k-values for light-hydrocarbon mixtures. *SPE Reservoir Engineering*, 1996.

M. Höök. Depletion and decline curve analysis in crude oil production. 2009.

J. D. Jansen. *Nodal Analysis of Oil and Gas Wells*. TU Delft, 2015.

I. Kaastra and M. Boyd. Designing a neural network for forecasting financial and economic time series. *Elsevier*, 1995.

D.A. Kooijman. Analysis of natural fractures in the basal zechstein carbonates in the dutch offshore area using wireline log data. 2011.

D. Kriesel. *Neural Networks*. 2005.

J. F. Lea, H. V. Nickens, and M. R. Wells. *Gas Well Deliquification*. Elsevier, 2008.

M. Li, S. Lei, and S. Li. New view on continuous-removal liquids from gas wells. *SPE Production & Operations*, 2002.

MATLAB®. version 9.1 (r2016b). 2016a.

MATLAB®. Matlab and multilayer neural network architecture. 2016b.

MATLAB®. Matlab and cluster data with a self-organizing map. 2016c.

MATLAB®. Matlab and filtering and smoothing data. 2016d.

MATLAB®. Matlab and analyze neural network performance after training. 2016e.

MATLAB®. Matlab and neural network time-series prediction and modeling. 2016f.

M.A. Nosseir, T.A. Darwich, Sayyouh M.H., and El Sallaly M. A new approach for accurate prediction of loading in gas wells under different flowing conditions. *SPE Production & Facilities*, 2000.

H.Y. Park. Decision matrix for liquid loading in gas wells for cost/benefit analysis of lifting options. *Texas A&M University*, 2008.

S.G.K. Patro and K.K. Sahu. Normalization: A preprocessing stage. 2015.

B. Rao. Designing coiled tubing velocity strings. *CTES, L.C*, 1999.

R. Rojas. *Neural Networks*. 1996.

O.L. Rowlan, J.N. McCoy, and A.L. Podio. Acoustic liquid level determination of liquid loading in gas wells. *The Geological Society of America*, 2006.

W. Schiferli, J. de Boer, and E. Nennie. Predicting the gains of deliquification. *Gas Well Deliquification Workshop*, 2013.

S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Pub, 1997.

R.G. Turner, M.G. Hubbard, and A.E. Dukler. Analysis and prediction of minimum flow rate for the continuous removal of liquids from gas wells. *Journal of Petroleum Technology*, 1969.

D. van Nimwegen. *The Effect of Surfactants on Gas-Liquid Pipe Flows*. Uitgeverij BOXPress, 2015.

K. Veeken, B. Hu, and W. Schiferli. Gas-well liquid-loading-field-data anaysis and multiphase-flow modeling. *SPE Production & Operations*, 2009.

K. Veeken, B. Hu, and W. Schiferli. New perspective on gas well liquid loading and unloading. *Society of Petroleum Engineers*, 2010.

F. Yavuz and F. Jansen. Successfully managing mature gas fields in the netherlands. *SPE 166362*, 2013.

F. Yavuz, E. Kreft, and R. Gijbels. Managing mature gas fields in the netherlands - future outlook. *SPE 166627*, 2013.