Application of Gradient Boosting for Virtual Flow Metering ⋆

Timur Bikmukhametov * **Johannes Jäschke** *

* *Dept. of Chemical Engineering, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, (e-mail: timur.bikmukhametov@ntnu.no, johannes.jaschke@ntnu.no)*

Abstract: Data-driven solutions for multiphase flowrate estimation in oil and gas production systems are among the alternatives to first principles virtual flow metering systems and hardware flow metering installations. Some of the most popular data-driven methods in this area are based on artificial neural networks which have been proven to be good virtual flow metering tools. However, neural networks are known to be sensitive to the scaling of input data, difficult to tune and provide a black-box solution with occasionally unexplainable behavior under certain conditions. As an alternative, in this paper, we explore capabilities of the Gradient Boosting algorithm in predicting oil flowrates using available field measurements. To do this, we use an efficient implementation of the algorithm named XGBoost. In contrast to the neural networks, this algorithm is insensible to data scaling, more intuitive in tuning as well as it provides an opportunity to analyze feature influence. We show that the algorithm provides accurate flowrate predictions under various conditions and can be used as a back-up as well as a standalone multiphase flow metering solution.

Keywords: Virtual flow metering, machine learning, flow estimation, gradient boosting, soft sensing.

1. INTRODUCTION

Accurate measurements of oil, gas and water flowrates are a critical part in optimization, reservoir management and flow assurance of petroleum production systems (Falcone et al. (2001)). A traditional method of measuring these flowrates is well testing which can be conducted by re-routing a well stream into a test separator or by changing wellhead choke opening and tracking the change of the rates at an inlet separator. Another alternative are multiphase flow meters (MPFMs) which allow to avoid separating the multiphase flow streams and measure the flowrates from each single well or a cluster of wells in real time. Despite this advantage, MPFMs are expensive and can be a subject to degradation and costly repair (Patel et al. (2014)).

An alternative way of estimating the multiphase flowrates is to combine field measurements such as pressure and temperature with first principles mathematical models which accurately represent specific system parts or the system as a whole. Some measurements are used as input parameters (as model boundary conditions) together with tuning variables such as flowrate or choke discharge coefficient. The remaining parameters are estimated by the models. The differences between the estimated and actual measurement values are minimized by an optimization solver. This approach is called Virtual Flow Metering (VFM) and can be used as a back-up to MPFMs as well as a standalone metering solution.

As an alternative to the first principles models, one can use a data-driven approach in order to estimate the flowrates. In this case, the specifics of the production system such as geometry of the well tubing or choke are not considered and only field measurements are used to identify the system model. The advantage of using these models is a low computational cost and relative simplicity in comparison to the first principles VFM systems. These facts are especially of advantage if one has no access to the first principles models and uses it as a black-box system, for instance, in a commercial multiphase flow solver. In this case, computing gradients for optimization is computationally expensive while the data-driven models provide the gradients at a much lower cost. Moreover, a well-trained data-driven model allows to predict the flowrates in real time with a sampling time of seconds while VFM based on first principles models may have a time delay due to solving the embedded non-linear optimization problem.

The most popular data-driven approach in VFM is feed-forward neural networks (NNs) with various modifications of structures and weights optimization, see Berneti and Shahbazian (2011) and AL-Qutami et al. (2018) with the associated references. Despite the high accuracy of NNs, there are some disadvantages associated with them. First, it is difficult to establish a good rule for NN architecture construction, such that creating a successful structure of the NN requires strong user's experience and can be time consuming. Also, the accuracy of NNs is dependent on the scale of the input features and target variables, such that

⋆ The authors gratefully acknowledge the financial support from the center for research-based innovation SUBPRO, which is financed by the Research Council of Norway, major industry partners, and NTNU.

NNs require data normalization (Sola and Sevilla, 1997). This is especially the case in VFM since the scale of the features varies dramatically. Also, the resulted NN is used as a black-box and sometimes it is difficult to understand the reason behind its behavior. Hyperparameters tuning to avoid model overfitting is also often a challenge in NNs training.

Gradient Boosting (GB) is another efficient method for solving non-linear classification and regression problems by constructing an ensemble of weak learners (simple algorithms) into a strong learner which is used to solve a particular problem (Friedman (2001)). One of the most popular modifications of GB is applying regression trees as weak learners which is called Tree Gradient Boosting. Among various implementations of Tree Gradient Boosting, eXtreme Gradient Boosting (*XGBoost*) by Chen and Guestrin (2016) is one of the widely used algorithms for solving machine learning problems. In this work, we also apply this algorithm implementation. In contrast to NNs, GB does not require scaling of the features which makes it more convenient for VFM applications. In addition, despite many hyperparameters, the tuning process of GB can be more intuitive and flexible compared to NN's tuning. For instance, increasing the number of trees by one allows a careful model adjustment while increasing the number of nodes in NNs by one may lead to a strong change of the model performance and possible overfitting, especially in small datasets. Another advantage of GB is the feature importance analysis option which gives an opportunity to better understand the algorithm behavior and get additional insights of the system parameters.

In this paper, we analyze capabilities of XGBoost in predicting oil flowrates from a subsea well under realistic conditions. We show how the algorithm can be used in different field development strategies as a back-up system for a multiphase flow meter or a standalone solution using the information from well tests. In addition, we analyze the performance of K-Fold and early stopping cross-validation strategies together with a tuning procedure for selecting an accurate set of XGBoost hyperparameters for VFM applications.

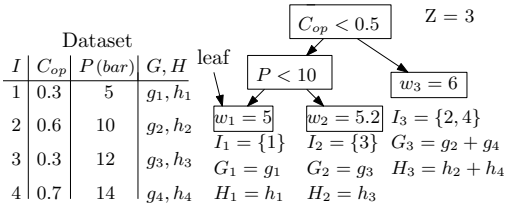
2. XGBOOST ALGORITHM

In this section, we give an overview of basic principles of Gradient Boosting and its implementation in XGBoost algorithm based on the paper by Chen and Guestrin (2016).

Consider a dataset $\mathcal{D} = \{(x_i, y_i)\}$ ($i = 1...n$, $x_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$), meaning that we have m features for each of n observation examples which correspond to the target variable y. A tree ensemble prediction for a given observation i is produced as a sum of predictions from K additive functions

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i) \tag{1}$$

where f_k is a regression tree predicting the value $f_k(x_i)$ for the *i-th* example. By training an ensemble of regression trees, we want to minimize the objective function with a loss term (l) and a regularization term (Ω)



I	C_{op}	$P\,(bar)$	G, H
1	0.3	5	g_1, h_1
2	0.6	10	g_2, h_2
3	0.3	12	g_3, h_3
4	0.7	14	g_4, h_4

Fig. 1. Example of a regression tree in XGBoost for VFM

$$L(\phi) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \tag{2}$$

where

$$\Omega(f) = \gamma Z + \frac{1}{2}\lambda \|w\|^2 \tag{3}$$

where γ and λ are hyperparameters which penalize the model complexity defined by the number of leaves Z and leaf weight values w. The loss term (l) can be expressed in a form of the user's interest, for instance, as the mean squared error for regression problems.

The objective in (2) is minimized in an iterative manner by adding a regression tree at each iteration. This leads us to the following objective function at *t-th* iteration

$$L^t = \sum_i l(y_i, \hat{y_i}^{t-1} + f_t(x_i)) + \Omega(f_t) \tag{4}$$

Applying a second order Taylor expansion and removing the terms independent of f_t, it can be shown that the following approximation of (4) can be obtained (Chen and Guestrin, 2016)

$$\tilde{L}^t = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{5}$$

where g_i and h_i are the first and second order derivatives of $l(y_i, \hat{y_i}^{(t-1)})$ w.r.t. $\hat{y_i}^{(t-1)}$. Defining I_j as a group of observations in the *j-th* leaf in a particular tree structure and taking into account that the tree produces the same weights score for the observations in one leaf, we can compute the optimal leaf weights w_j^* and the corresponding optimal value of the objective approximation $\tilde{L}^t$ (Chen and Guestrin, 2016)

$$w_j^* = \frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{6}$$

$$\tilde{L}^t(q) = -\frac{1}{2} \sum_{j=1}^{T} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma Z \tag{7}$$

where q denotes a particular tree structure. Equation (7) is used as an evaluation criteria to find an optimal split of the tree. The tree is grown greedy to avoid enumerating all possible structures q meaning that the algorithm starts splitting from a single leaf and adds branches according to (7). Fig. 1 shows a simple example of an XGBoost regression tree with the algorithm notations and measurement data of pressure and choke opening used in VFM. To get a better understanding of the splitting procedure, consider I_L and I_R to be the left and right groups of observations after the tree node split. Having this information, we can calculate a loss reduction caused by the split

$$L_{split} = \frac{1}{2}\left[\frac{\left(\sum\limits_{i\in I_L} g_i\right)^2}{\sum\limits_{i\in I_L} h_i + \lambda} + \frac{\left(\sum\limits_{i\in I_R} g_i\right)^2}{\sum\limits_{i\in I_R} h_i + \lambda} - \frac{\left(\sum\limits_{i\in I} g_i\right)^2}{\sum\limits_{i\in I} h_i + \lambda}\right] - \gamma \quad (8)$$

The loss reduction (8) is used to evaluate each possible split by linear scanning of sorted values for each feature in each node. The best split is considered to be the one which gives the *maximum* value of the loss reduction. When the splitting is finished, the leaf values are assigned according to (6).

For a more detailed explanation of XGBoost algorithm derivation and its additional features such as random data sampling or shrinking tree outputs the interested reader is referred to the original paper by Chen and Guestrin (2016).

3. PRODUCTION SYSTEM MODELING

We consider a simple subsea production system which consists of an oil well, a flowline, a riser and an inlet separator with a constant pressure. The well is equipped with a multiphase flow meter, choke, pressure and temperature sensors which are installed at the bottomhole, upstream and downstream of the choke. In addition, information about the choke opening is available. The system performance is simulated in OLGA which is considered to be one of the leading simulation tools for multiphase flow transport in oil and gas production systems. To manipulate the choke opening and inflow sources as well as collect simulation results, we use MATLAB together with an OPC server. Such a simulation setup enables advanced options for controlling variables and allows introducing additional system dynamics to mimic a real system behavior. The production system is shown in Fig. 2.

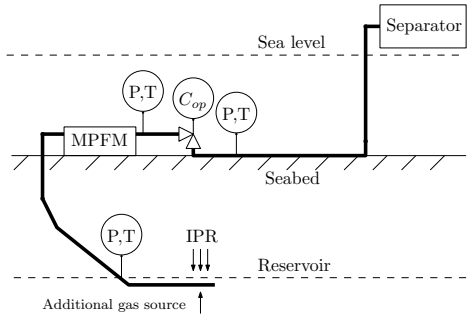


Fig. 2. Schematic representation of the production system

To model the reservoir inflow, we use Inflow Performance Relationship (IPR) formulated by Vogel's equation (Vogel et al. (1968)). To mimic dynamics of the reservoir, we introduce a *linear reservoir pressure decline* with respect to the production time. Also, to simulate a possible dynamic response of the near-wellbore region to the change of the bottomhole pressure caused by the choke position change and occasional gas breakthroughs from injection wells, we include an additional gas source which has a periodic form represented by the following relationship

$$\dot{m}_{Source} = \dot{m}_{max}\left[1 + a \cdot sin\left(\frac{\pi \cdot s}{T_{source}}\right)\right] \quad (9)$$

where $\dot{m}_{max}$ denotes the maximum mass gas source value, s - the time step, a and T_{source} - the amplitude and the period of the *sin* function respectively. This trick together with the reservoir pressure decline is done in order to mimic a possible real system behavior and challenge the VFM to predict the varying flowrates. Otherwise, a particular value of the choke position would correspond to a specific value of the flowrate without any additional disturbance which makes the case unrealistic as well as simplifies the training and predicting process for the machine learning algorithm.

Table 1. System and simulation parameters

Parameter	Value	Parameter	Value
True vertical depth	2010 m	$\dot{m}_{\max}$	0.35 kg/s
Measured depth	3110 m	T_{Meter}	72
Flowline length	1000 m	T_{Source}	144
Riser length	100 m	a	0.5

To calculate the multiphase flowrate meter predictions, we assume that 100% flowrate measurements by the MPFM are within ±5% accuracy with respect to the true value. To avoid using unrealistic random fluctuations around the mean, we use a periodic relationship which allows to escape a noisy and unstable device behavior under stable flow conditions

$$Q_{Meter} = Q_{True}\left[1 + 0.05 \cdot sin\left(\frac{\pi \cdot s}{T_{Meter}}\right)\right] \quad (10)$$

where T_{Meter} denotes the period of the *sin* function.

4. CASE STUDIES

We consider two different cross-validation strategies: K-Fold and early stopping. In K-Fold cross-validation, the available data is divided into training and test datasets. The training set is again divided in *K-folds* meaning that the model is trained and tested K times. The obtained accuracy on K test sets are summed and averaged to make conclusions about the overall model accuracy. Finally, the algorithm is tested on the test dataset to evaluate the model generalization. In early stopping, the available data set is divided into 3 subsets: training, validation and test. The algorithm is trained on the training set while the error is also monitored on the validation set. The training is stopped when the error on the validation set stops decreasing after adding a specified number of new trees. The performance of the obtained model is checked using the test dataset.

In this work, the test datasets are selected to be 15% of the available training data for both K-Fold and early stopping. In early stopping, another 15% of the data is dedicated for the validation dataset.

The data are produced using the production system architecture shown in Fig. 2. The performance of the system is simulated for a period of 2 years. The obtained production profile without the well tests performance is shown in Fig. 3. The period is divided into 4 quarters 180 days each. At the beginning of each quarter, a well test is conducted to obtain reliable information about the well performance. We collect the measurements every 8 hours during the normal production time and every 30 mins during the well tests. The following measurements are collected for the algorithm training and predicting the flowrates in the future time period:

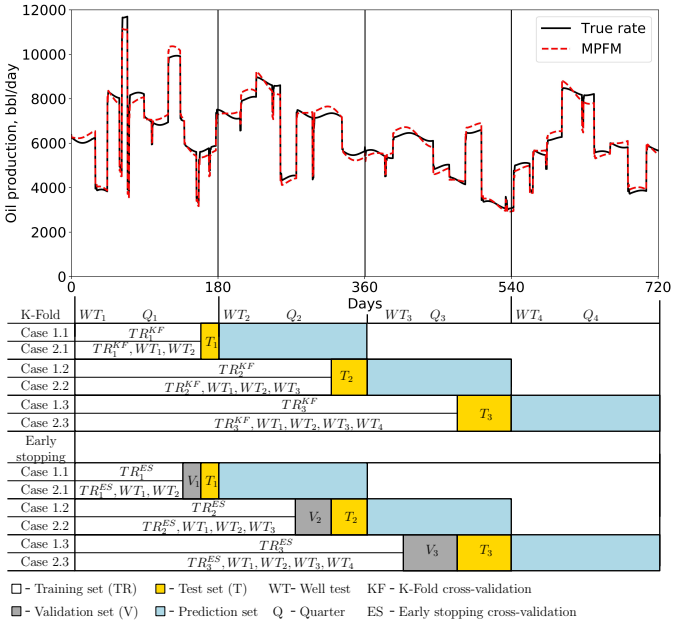


Fig. 3. Production profile with data splitting schemes

- Pressure and temperature at the bottomhole, upstream and downstream of the choke
- Choke opening and oil flowrates from MPFM or well tests

We analyze 3 case studies which have several sub-cases each. For 2 case studies we also compare the performance of K-Fold and early stopping cross-validation approaches. Each case considers a separate field development strategy, so we analyze the performance of GB VFM for various situations of production operation. The detailed description of each case study is discussed below.

4.1 Case 1 - MPFM data

In this case, we assume that we do not have information from the well tests and use the flowrate measurements from the MPFM only. This case is possible when well testing is expensive and rarely performed. For this case, we perform 3 cases studies by extending the training datasets as the production time evolves. For instance, in the first study (Case 1.1) we assume that the data from the first half a year is available for training (Q_1 in Fig. 3) and we would like to predict the flowrates for Q_2. As the time evolves and we obtain more training data, in Case 1.2 we use the data from Q_1 and Q_2 for training and testing the model and perform predictions on Q_3. This procedure is done for K-Fold and early stopping approaches. Fig. 3 visualizes the dataset splitting for training, validation, testing and predicting for each case and each cross-validation method.

4.2 Case 2 - MPFM and well test data

In this case, we combine the well tests and MPFM data for training. Well tests can be conducted even if MPFMs are installed in order to calibrate these devices as well as update information about reservoir properties and the well performance. Similarly to Case 1, we extend the training datasets as the time evolves. The well testing procedure is explained in more details in Case 3 description. As

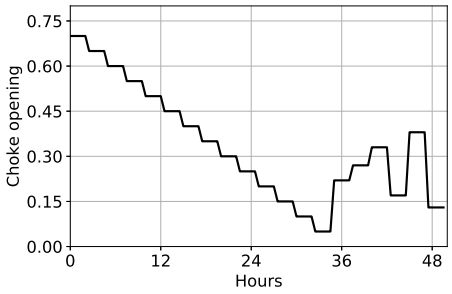


Fig. 4. Choke opening procedure during well testing

in Case 1, we conduct the studies for K-Fold and early stopping cross-validation. Fig. 3 shows the dataset splits for training, validation, testing and predicting for the subcases of Case 2.

4.3 Case 3 - Well test data

In this case, we assume that the MPFM is not installed at the wellhead and training data is available only from the well tests. This situation can happen when MPFMs are not economically or operationally feasible to install because of high cost or flow assurance challenges and instead well tests are performed to track the production rates. To generate feasible data for GB VFM, we propose well testing with step-wise changes of the choke opening over the possible operating range. In this case, we assume the choke opening to be within the range of 0.05 and 0.7 and we perform the well test with a choke opening step of 0.05. Also, we perform additional tests around the expected well operating point. In this case, we expect the operating point to be within the range of 0.10 and 0.4 and perform several additional step changes over this range. The choke opening sequence used for well testing is shown in Fig. 4. We perform the same well test procedure every 180 days and then use the combination of the well test data for training as the time evolves.

The problem in this situation is the fact that the amount of data is limited, so that obtaining a validation and test datasets for model evaluation and overfitting control is difficult and case dependent. Moreover, even K-Fold cross-validation may not help in this case, since there are only a few measurements for each point of the gradually changing choke opening. In this work, we assume that there is no available data for model testing and train the model until the training dataset is well fitted. Table 2 shows the matrix of the datasets usage in Case 3.

Table 2. Matrix of datasets for Case 3

Case	Train	Validation	Test	Prediction
Case 3.1	WT_1	-	-	Q_1
Case 3.2	WT_1, WT_2	-	-	Q_2
Case 3.3	WT_1, WT_2, WT_3	-	-	Q_3
Case 3.4	WT_1, WT_2 WT_3, WT_4	-	-	Q_4

4.4 XGBoost application and tuning

In this work, we use Python implementation of XGBoost. To select hyperparameters, we use random search approach which in many cases can be more efficient than

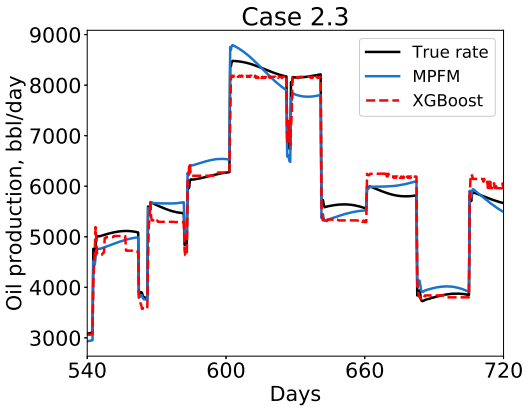


Fig. 5. Comparison of GB VFM and MPFM predictions

grid search (Bergstra and Bengio (2012)). To explore a large subspace of the hyperparameters, we perform 10 random searches 50 iterations each for both K-Fold and early stopping in each sub-case study. In Case 1 and Case 2 this search is performed for both K-Fold and early stopping. This approach allows quick exploration of many hyperparameters combinations and comparison of several algorithms which are found the best within its own search procedure. The best algorithm within one random search is selected based on the mean absolute error criterion. Among 10 best algorithms from each random search, the one is chosen which has the smallest training error and a test error with a low prediction variance. Each random search is run with a new but predefined random state for results reproducibility. To make a fair comparison of K-Fold and early stopping methods, the same random states are used for both strategies. For K-Fold cross-validation studies, the number of folds is chosen to be 5.

In order to tune the algorithm within the random search, we use the following approach:

1. Choose a relatively low learning rate (0.1 as an initial guess is used).
2. Specify a uniformly distributed space for regularization parameters γ and λ from small to relatively large values ([0.1...20] range is used).
3. Specify a reasonable array of values for minimum child weights ([1...8] range with step 1 is used).
4. Perform random search and final model selection by controlling the upper bound of the ranges for the maximum tree depth and number of estimators.
5. In case of early stopping, select the best model from the random search and train until the error does not decrease for 5 newly added trees.
6. If the performance is not satisfactory, reduce the learning rate, increase the number of estimators and repeat the search.

We do not use random data sampling because for some cases the amount of data is limited, so that to perform a better comparison it is not used for other cases as well.

5. RESULTS AND DISCUSSION

In this section, we analyze the simulation results for each simulation case separately and afterwards make general remarks about the performance of XGBoost for Virtual Flow Metering. To evaluate the performance on the predicting datasets, we use Mean Absolute Squared Error (MAPE) which shows the average absolute percentage deviation of the predictions from the true value

$$MAPE = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \cdot 100 \qquad (11)$$

To compare the XGBoost performance for Case 1 and Case 2, we also calculate MAPE between the multiphase flow meter measurements and the true rate. Even though the predictions of the MPFM are computed using artificial assumptions (the *sin* function form of the measured error), these data is a basis for XGBoost training, so that we aim to get the algorithm predictions as good as the MPFM or even better when adding the well test data in training.

Table 3. MAPE of GB VFM and MPFM

Method	MAPE					
	Case 1.1		Case 1.2		Case 1.3	
GB VFM	KF*	ES**	KF	ES	KF	ES
	5.65%	8.3%	5.34%	5.25%	4.75%	3.39%
	Case 2.1		Case 2.2		Case 2.3	
GB VFM	KF	ES	KF	ES	KF	ES
	2.08%	2.38%	3.93%	4.35%	3.41%	4.07%
	Case 1.1/2.1		Case 1.2/2.2		Case 1.3/2.3	
MPFM	3.15%		3.09%		3.14%	
	Case 3.1	Case 3.2		Case 3.3	Case 3.4	
GB VFM	6.3%	3.79%		4.44%	4.09%	

* - K-Fold cross-validation
** - Early stopping cross-validation

Table 3 summarizes the simulations results from all the cases. For the sub-cases of Case 1, we see that with the increase of the dataset size, the performance with both cross-validation methods improves. Another interesting observation is that K-Fold cross-validation outperforms early stopping in Case 1.1, in Case 1.2 the methods give similar results while in Case 1.3 early stopping outperforms K-Fold method. The main reason for this is the fact that the validation set increases from case to case which provides an opportunity for early stopping to control the number of trees in a more robust manner. However, one should notice that this situation might not always be the case. For instance, if the validation dataset was very different from the prediction dataset, early stopping would potentially show less accurate performance than K-Fold method. On the other hand, if the well production is not expected to have severe changes (as on the considered case), early stopping can be a more accurate method for GB VFM training than K-Fold approach as shown.

Another important observation from Table 3 is the fact in all the sub-cases of Case 2 the error is lower both for K-Fold and early stopping than in the corresponding sub-cases of Case 1. This shows that adding the information from the well tests helps to improve flowrate predictions from GB VFM. Another observation for Case 2 is that fact K-Fold outperforms early stopping in each sub-case. This shows that for the data with higher variability added by the well tests, K-Fold cross-validation can be a more robust way of the algorithm training. Potentially, the performance of early stopping in cases with well testing data can be improved by a better selection of the data splitting strategy. For instance, a part of the well test dataset can be included into the validation set while in

our work we used well test data in the training dataset only.

Overall, we observe that the MAPE from GB VFM is comparable with the error from the MPFM, especially in Case 1.3 and Case 2.3 when the dataset becomes relatively big. An example of the flowrate predictions by GB VFM is shown in Fig. 5. The figure shows that during some production time the constant piecewise approximations by the regression trees is good enough and have values closer to the true rate than the simulated MPFM rate predictions while in some parts constant piecewise predictions can be relatively inaccurate. Potentially, the performance of GB VFM can be further improved by applying linear function approximations instead of constant piecewise ones which may have a better ability to extrapolate the flowrate predictions.

Another interesting observation from Table 3 is that the smallest errors are achieved in Case 2.1 even though this case does not have the largest training dataset. The reason for this is the fact that the choke openings values in Q_2 (prediction dataset for Case 2.1) coincidentally matched the values considered during the well tests multiple times. Since the flowrate estimates from the well tests does not include the MPFM uncertainty, the resulted error is even lower than the error from the MPFM. This result is promising meaning that by performing a well-planned well testing around the expected operating point can lead to very accurate flowrate predictions by GB VFM.

As for the sub-cases of Case 3, we see that initially the error decreases as the dataset size increases but in Case 3.3 the error increases again with a subsequent decrease in Case 3.4. The reason to have Case 3.2 error the lowest is the same as for Case 2.1 - the choke openings values in Q_2 (prediction dataset for Case 3.2) exactly coincided multiple times with the values considered during the well tests. But as a general tendency we can see that the increase of the dataset helps to improve the flowrate predictions.

One of the reasons to include an additional sub-case (Case 3.1) in Case 3 was to see if we can use well tests from the beginning of the field operation for VFM purposes without a need of MPFM installation. As we can see from Table 3 the error in Case 3.1 is relatively large in comparison with the MPFM while with the new data obtained the error becomes comparable. Thus, potentially the combination of the well testing performed in a step-wise choke opening manner with GB VFM can be used as a standalone solution. However, at the initial production phase the accuracy can be low. One solution for this problem can be performing longer and more rigorous well tests for the initial stage with reducing well test complexity as the time evolves.

One should also notice that the training was done without validation and test datasets, so that even well fitted algorithm produced good results. In a real case, the well tests measurements may not have such a good accuracy as in the considered case, so that an overfitted model may give worse predictions than the presented ones. In this case, obtaining more data from well testing and using it as a validation/test datasets can be a solution.

6. CONCLUSION

In this work, the XGBoost implementation of Gradient Boosting machine learning algorithm was used to predict oil flowrates from a simple subsea production system under various field development strategies. The algorithm showed a performance comparable with a hardware multiphase flow meter and has a potential to be used as a back-up as well as a standalone solution for Virtual Flow Metering even provided with a small training dataset. Depending on the available dataset size and variability, K-Fold or early stopping cross-validation strategies can be used to obtain a good algorithm performance. Random search strategy of the algorithm selection combined with a careful parameter tuning produces good results of the flowrate predictions. The simulation results showed that by combining GB algorithm with the flowrate measurements from well testing over a wide operating range of the well, it is possible to make accurate flowrate predictions starting from an early well production stage. The future work can address improvements of GB application for VFM by using a linear model as a weak learner to increase extrapolation capabilities as well as challenges associated with the uncertainty of the flowrate measurements and limited data availability from the well tests.

REFERENCES

AL-Qutami, T.A., Ibrahim, R., Ismail, I., and Ishak, M.A. (2018). Virtual multiphase flow metering using diverse neural network ensemble and adaptive simulated annealing. *Expert Systems with Applications*, 93, 72–85.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb), 281–305.

Berneti, S.M. and Shahbazian, M. (2011). An imperialist competitive algorithm artificial neural network method to predict oil flow rate of the wells. *International journal of computer applications*, 26(10), 47–50.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794. ACM.

Falcone, G., Hewitt, G., Alimonti, C., Harrison, B., et al. (2001). Multiphase flow metering: current trends and future developments. In *SPE annual technical conference and exhibition*. Society of Petroleum Engineers.

Friedman, J.H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Patel, P., Odden, H., Djoric, B., Garner, R.D., Vea, H.K., et al. (2014). Model based multiphase metering and production allocation. In *Offshore Technology Conference-Asia*. Offshore Technology Conference.

Sola, J. and Sevilla, J. (1997). Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, 44(3), 1464–1468.

Vogel, J. et al. (1968). Inflow performance relationships for solution-gas drive wells. *Journal of petroleum technology*, 20(01), 83–92.