Automated analysis of temperature data in fractured wells

Dante Isaac Orta-Aleman (dorta@stanford.edu). Department of Energy Resources Engineering, Stanford University

1 Abstract

Two Machine Learning regression models: the Lasso and a Random Forest were implemented to estimate the fluid flow-rate coming out of individual fractures in a hydraulically fractured well. Both models proved to capture the underlying physics of flow in simple injection processes.

2 Introduction

Hydraulic fracturing technology has made a huge impact in the Oil and Gas industry by enabling the production of deposits that otherwise would be unfeasible to produce. However, the physics of the stimulation process (fracking) are still not fully understood and the industry still faces the challenge of evaluating the effectiveness of a stimulation process.

By estimating the fluid flow rate coming out of each fracture, it is possible to know the size of the fractures and if there is the need to restimulate the well. Previous work has proven that temperature data from the wellbore can be used identify the location of fractures and estimate the flow rate coming out of them (Priscila Ribeiro, 2014).

The goal of this project is to use a Machine Learning model to estimate the flow rate coming out of individual fractures using temperature data. The input data for the model is welbore temperature readings during an injection process. The data is obtained using the AD-GPRS flow simulator.

3 Related Work

Most of the work relating temperature data with flow rate has been done using a full physics approach. This involves using numerical simulations along with real data to study the effect of flow rate in temperature readings (Kevin Raterman, 2017), (Priscila Ribeiro, 2014), (Jose Sierra, 2008). However, most of the work involving Machine Learning for temperature data analysis has been done by research gropus inside Oil and Gas companies and even if it has been presented in conferences, it has mostly gone unpublished, protected by the industry's privacy practices.

Nevertheless, there have been multiple applications of Machine Learning for flow rate interpretation and reservoir characterization. (Chuan Tian, 2015) successfully applied Kernel Ridge Regression to model pressure and temperature data from Permanent Downhole Gauges (PDGs). Other teams in major oil companies have applied Machine learning with different success rates (Niranjan Subrahmanya, 2014).

4 Dataset and Features

The dataset is simulated data from an injection process done using the AD-GPRS flow simulator developed by the SUPRI-B: Reservoir Simulation research group at Stanford. The geometry used for the simulation isa single 200 m horizontal well with three fractures, spaced each by 50 m. Each fracture is a vertical plane, perpendicular to the wellbore. The length of the fractures are 5, 15 and 25 m respectively.

The output of the simulation consists of spatially tagged temperature and flow-rate time series. Table 1 contains the description of the data variables[1] as they come out of the simulator.

Feature Name	Description	Data Type
Spatial location	Location of the point along the wellbore (m)	numeric
Temperature	Absolute temperature of the point (K)	numeric
Time	Time in the simulation (days)	numeric

Table 1. Data Variables coming out of the simulation

Previous work suggest that the time and spatial derivatives of temperature are relevant variables when relating flow-rate with temperature in the wellbore (Priscila Ribeiro, 2014). The derivatives are numerically computed using a two point approximation.
Even though the fluid flow is a time dependent process, when estimating flow-rate the relevant variable is the temperature rate of change at a point in time, not the time itself. Therefore, time is not explicitly considered in the variables but only accounted for by using the time derivative.
Spatial location is only relevant as it indicates where the fracture is, pointing to the relevant part of the dataset, as we only care about the flow-rate coming out of the fracture. Thus, we only consider the data in the fracture location and nerby it for the spatial derivative computation. Therefore, the model variables[2] are as specified in Table 2.

Feature Name	Description	Data Type
Temperature	Absolute temperature of the point (K)	numeric
dT/dx	Temperature spatial derivative (K/m)	numeric
dT/dt	Temperature time derivative (K/day)	numeric

Table 2. Model variables before any polynomial expansion

As the dimensionality of the data is low, a polynomial expansion up to the third power is introduced. This is done using the Scikit Learn PolynomialFeatures function, which computes both the polynomials and the interactions between them. By introducing these terms, the features set's dimensionality increases from 3 to 19. Before feeding the variables to both algorithms, a standardization process was made to set every variable's mean to zero and unitary standard deviation.

[1] We refer to data variables as the "raw" variables coming from the simulator.
[2] The model variables are those that we introduce to the model (before the polynomial expansion).

5 Regression Methods

Two different regression methods are used in the project: The Lasso Regression and a Random Forest. These two methods were as they are capable of handling non-linear behavior[3] and are robust against irrelevant features.

5.1 Lasso Regression

The Lasso is a regularized linear regression method. It can be classified as a "shrinkage method" since it retains a subset of the predictors while discarding the rest. This has the effect of making the model more interpretable and potentially lowering the prediction error compared to the Ordinary Least Squares Regresssion (Trevor Hastie, 2008).

The Lasso estimate is defined as:

$$\min_{\alpha}\left\{\sum_{i=1}^{N}\left(y_i-\sum_{k=1}^{K}\alpha_k T_k(x_i)\right)^2+\lambda\cdot J(\alpha)\right\},\qquad J(\alpha) = \sum_{k=1}^{K}|\alpha_k|$$

Equation 1. Lasso estimate

As it can be seen from Equation 1. Lasso estimateEquation 1, the Lasso uses Least Squares as the loss function while regularizing the parameters by penalizing the regression's coefficients by their L1 norm. The effect of the L1 regularization is to effectively set to zero some of the coefficients, in contrast to the Ridge Regression which only reduces the size of them (Trevor Hastie, 2008).

5.2 Random Forest Regression

A Random Forest is a tree-based regression algorithm that uses *bagging* to reduce the variance of individual decision trees. It does so by buiding a large collection of de-correlated trees and averaging the result of them (Trevor Hastie, 2008).

The Random Forest algorithm keeps most of the individual trees' advantages, implicit feature selection, non-linearity and fast computation, while reducing the inaccuracy and instability of individual trees. The estimate for the Forest is the following:

$$\hat{f}^B_{\text{rf}}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \qquad T_b(x) = \sum_{m=1}^{M} c_m I(x\in R_m)$$

Equation 2. Random Forest estimate

Equation 2 shows that the prediction of a Random forest is the average of the individual trees that compose it. The trees predictions are respectively the average value of the target variable over individual regions R_m, represented by the indicator function I(x). The regions R_m are chosen by recursively splitting the dataset while maximizing an information criterion like the Ginny coefficient. This is a greedy approach to getting the best splitting that would minimize the least squared error.

[3] The Lasso can model non-linear behavior when coupled with a polynomial expansion.

6 Training & Results

Both algorithms were implemented using the Scikit-Learn Python library. The two algorithms were trained using the simulated data corresponding to 3 days of injection. The dataset is small sized, having 110 data point for each fracture, which means an average time step of approximately half an hour.

To obtain the regularization parameter λ for the Lasso, a search was done on the training set using 100 possible values with a path length of 1e-3. The chosen λ was For the Random Forest, the number of trees was chosen using a search from the set [10,20,50,100,250,500], choosing 20 as the best value. The train/test split was done using 90% of the dataset for training and 10% for test. As the dataset is small, the 10% test set was recursively chosen using 10 folds to be able to produce an out of sample prediction for the whole 3 days curve.

For the Lasso Regression, the target variable was also changed to be the log(Flow-Rate). This improved the performance of the algorithm, as the flow-rate values were spread a wide range of values. For the Random Forest this modification was not necessary, as the algorithm is invariant to monotinic variable transformations (Friedman, 2017).

The results give the Lasso as the best model from the two with a smaller Mean Squared Error (MSE) and Mean Absolute Error (MAE)

Scoring Metric	Random Forest	Lasso
Mean Squared Error	0.32	0.23
Mean Absolute Error	0.54	0.17

Table 3. Performance scores for the test set

From Figure 1, it can be noticed that the Lasso Regression is also less noisy than the Random Forest. This could be explained by the relative simplicity of the Lasso, compared with the piecewise constant function aggregation that the forest is made of. The overall results are promising, showing that Machine Learning algorithms can capture the underlying physics for simple cases like this.

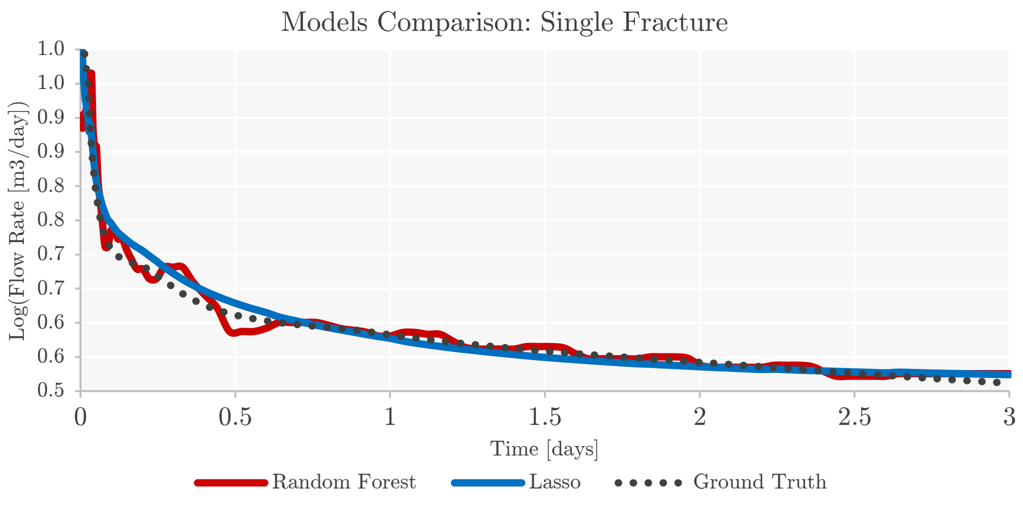

Figure 1. Out of sample predictions for both models

7 Future Work

There are two main directions where the future research needs to be headed towards. The first one is to test different algorithms to try to lower the prediction error. An interesting comparison would be to implement a Ridge Regression and compare it with the Lasso, as both of them are similar shrinkage methods. Another obvious candidate is a Neural Network, given the multiple successes the method has had in other prediction problems.

The second, and possibly most relevant direction is to try the model in more complicated physical scenarios. Different injection profiles and injection pressures need to be tested, as well as the possibility of estimating the flow-rate from different sized fractures using the same trained model. The idea being to push the Machine Learning algorithms to the limit of their capacity to infer the physics from the limited information present in the data.

8 Aknowledgements

The author wishes to thank Prof. Roland Horne for the insight and comments he kindly provided. Special aknoledgements also go to Timur Garipov for his help in using the AD-GPRS flow simulator. Without his help, running the simulations would have been impossible.

9 References

Chuan Tian, R. H. (2015). Applying Machine Learning Techniques to Interpret Flow Rate, Pressure and Temperature Data From Permanent Downhole Gauges. *SPE Western Regional Meeting.* Garden Grove, California: Society of Petroleum Engineers.

Friedman, J. (2017). Stats 315B Course Notes. Stanford, California.

Jose Sierra, J. K. (2008). DTS Monitoring Data of Hydraulic Fracturing: Experiences and Lessons Learned. *SPE Annual Technical Converence and Exhibition.* Denver, Colorado: Society of Petroleum Engineers.

Kevin Raterman, H. F. (2017). Sampling a Stimulated Rock Volume: An Eagle Ford Example. *Unconventional Resources Technology Conference.* Austin, Texas: Society of Petroleum Engineers.

Niranjan Subrahmanya, P. X.-B. (2014). Advanced Machine Learning Methods for Production Data Pattern Recognition. *SPE Intelligent Energy Conference and Exhibition.* Utrecht, The Netherlands: Society of Petroleum Engineers.

Priscila Ribeiro, R. H. (2014). Detecting Fracture Growth Out of Zone Using Temperature Analysis. *SPE Annual Technical Conference and Exhibition.* Amsterdam, The Netherlands: Society of Petroleum Engineers.

Trevor Hastie, R. T. (2008). *The Elements of Statistical Learning, Second Edition.* Stanford, California: Springer.