Machine Learning in Modern Well Testing

Yang Liu and Yinfeng Qin

December 11, 2009

1 Introduction

Well testing is a crucial stage in the decision of setting up new wells on oil field. Decision makers rely on the metrics to evaluate the candidate wells' potential. One important metric is permeability, measuring the ability of porous material to transmit fluids. High permeability often leads to high yielding.

In a conventional well test, the well is controlled to produce at a constant flow rate, and the pressure is measured for a couple of hours (Figure 1). This pressure curve will be used to interpret the reservoir parameters, including the permeability k and initial pressure P_i. To interpret the pressure curve, a radial flow with infinite boundary model is utilized, whose mathematical solution may be simply written in the Equation 1. Key parameters in Equation 1 are: p_{wf}, the measured bottom hole pressure, P_i, the initial pressure; q, the constant flow rate; k, the reservoir permeability. Traditionally the permeability may be interpreted by comparing the observed pressure curve with the calculated overlay template (Figure 2).

$$p_{wf} = P_i - \frac{qB\mu}{k}(\log t + C) \tag{1}$$

Nowadays, newly introduced Permanent Downhole Gauge (PDG), is widely used. PDG can measure both the pressure and varied flow rate for a long duration (Figure 3). However, current well test remains the conventional approach, interpreting only on a piece of pressure curve corresponding to a constant flow rate. Obviously, this method wastes most data and the resulting interpretation is not convincing.

This study tries to use machine learning approach to develop a method that is able to make an interpretation on a modern well test by taking all measurements into account. We would like to proceed in two steps. First, all the measured noisy data are used to train a machine learning model, which gives a good prediction given any flow rate history. Upon the completion of this step, the reservoir parameters, which are the goal of the well test, are actually stored in the machine learning model. Secondly, we try to interpret the well test result by extracting the reservoir parameters from the learning model. Two difficulties lie in the process: first, the current physical model is designed for constant flow rate, which is not the case in a modern well test; second, while in traditional well test the flow rate is accurate, in our problem both the flow rate and pressure are noisy.

Section 2 first discusses learning the data set by Locally Weighted Projection Regression (LWPR) algorithm. Section 3 discuss applying the maximize likelihood method in a Hilbert space by defining a transformation $\phi(x)$. Finally, Section 4 summarizes the whole project.

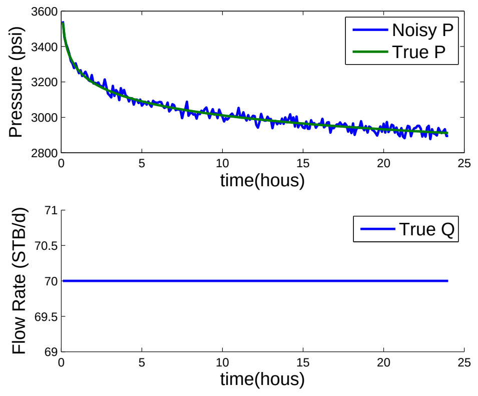


Figure 1: Pressure and flow rate signals from subsurface in a conventional well test.

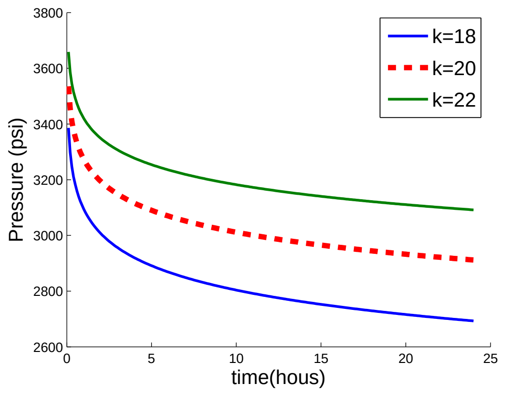


Figure 2: Pressure curves with different reservoir permeabilities.

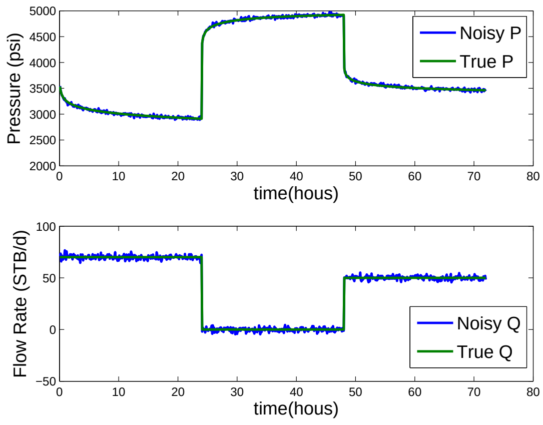


Figure 3: Pressure curves with different reservoir permeabilities in a modern well test.

2 Locally Weighted Projection Regression

2.1 Locally Weighted Projection Regression Algorithm

Locally Weighted Projection Regression (LWPR) is an algorithm that achieves nonlinear function approximation in high-dimensional spaces with locally weighted linear regression in each dimension (Atkeson, Moore, & Schaal, 1997). The LWPR algorithm is improved over the Locally Weighted Regression (LWR) algorithm by use of a projection process.

The workflow of LWPR is as following: (1) Project the training data into higher-dimensional spaces. A subset projected on each projection direction will be obtained. (2) Solve a LWR system on each subset. A linear hypothesis will be trained on each projection direction. (3) Sum up all hypothesis on all dimensions to reconstruct the hypothesis in the original one-dimensional space.

2.2 LWPR in Real Time Space

First the LWPR algorithm was applied to a synthetic pressure generated from constant flow rate without noise. In cases with constant flow rates(Figure 4 & 5), LWPR works very well.

When the flow rate is not constant, the pressure transient is no longer increasing or decreasing monotonically. The incorrect predictions will be more prevalent(Figure 6). The LWPR algorithm fails when the flow rate changes quickly (Figure 7).

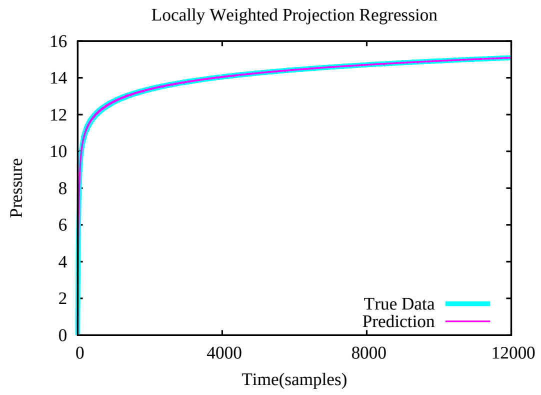


Figure 4: Synthetic pressure generated from constant flow rate without noise.

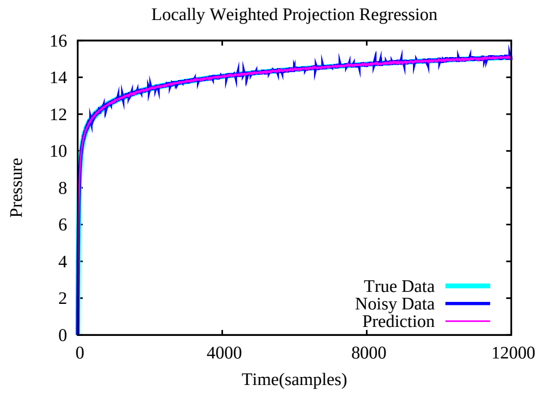


Figure 5: Synthetic pressure generated from constant flow rate with noise.

$$p_{wf} = \int_0^t q'(\tau)\left[p(t-\tau)+S\right]\mathrm{d}\tau \tag{2}$$

Currently we suspect this suffers from two reasons: first is the relatively slow learning rate of the algorithm, second is that the pressure is a result of convolution of previous flow rates (Horne, 1995), as described in Equation 2. To solve this problem, one choice is to convert the data set into a space where the pressures are independent of each other. There actually is such a space where the pressures are deconvolved, namely the Laplace space.

2.3 LWPR in Laplace Space

To apply the machine learning algorithm in Laplace space, the workflow is natural and straight-forward: (1) Transform the data set into Laplace space numerically. (2) Apply the machine learning method (LWPR) in Laplace space. Obtain the prediction

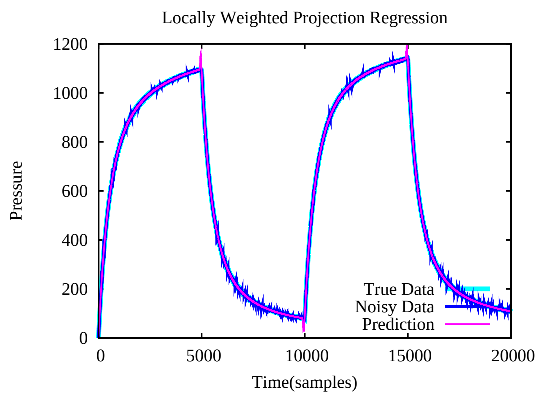


Figure 6: Synthetic pressure generated from changed flow rate with noise.

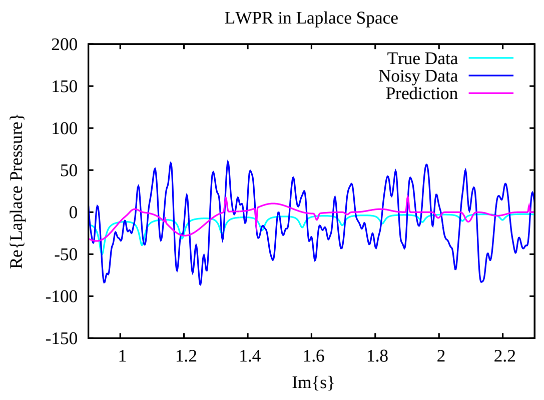


Figure 8: Synthetic pressure with noise in Laplace space.

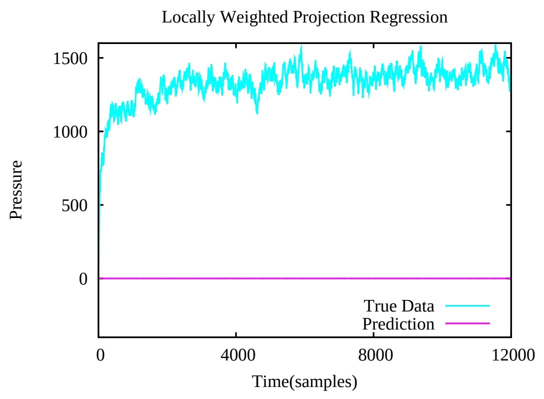


Figure 7: Synthetic pressure generated from fast changing flow rate without noise.

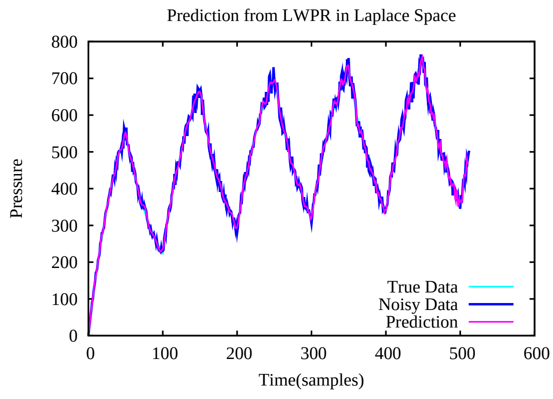


Figure 9: Synthetic pressure with noise in real time space.

in Laplace space. (3) Invert the prediction numerically from Laplace space back into time space. Figure 8 shows the result in Laplace space. From the figure, it is clear that the method works well in Laplace space.

The prediction in the Laplace space was then converted into real time space, as shown in Figure 9. The overall trend is captured well. Two zoom-in views are also provided in Figure 10 and Figure 11.

Although the LWPR regression obtains good prediction in the Laplace space, the performance is slow. There is heavy computation in the process of transforming and inverting the data between the real time space and the Laplace time space, which cost more than 95% CPU time. Therefore, a roll-back is required: how can we train the machine learning algorithm in the real time space but void the problem of data dependency? Section 3 will proposes another learning algorithm to answer this question.

3 Maximize Likelihood in Hilbert Space

3.1 Super Position

First we need to understand the physical essence of the pressure transient when the flow rates are varied. When the flow rates are varied, the pressure transients are formed by a physical process named **Super Position**. The pressure transients caused by varied flow rates are actually a combination of multiple pressure transients each of which is corresponding to a constant flow rate. Figure 12 demonstrates this process.

The super position enables us to re-write the control equation of the pressure transient in a modern well test, as shown in Equation 3.

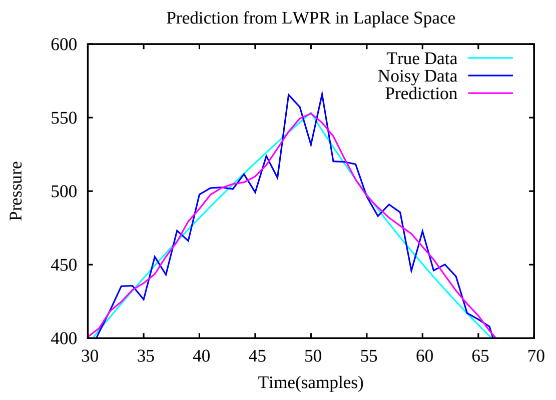


Figure 10: Synthetic pressure with noise in real time space: zoom-in view 1.

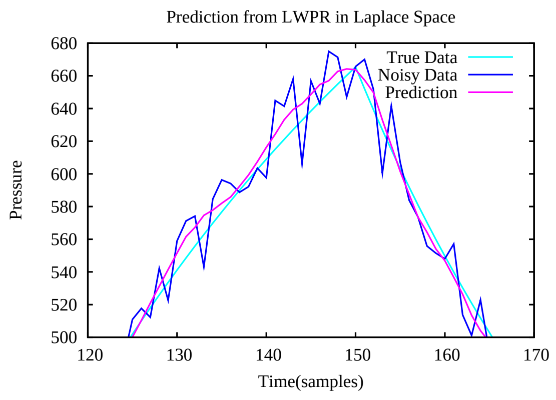


Figure 11: Synthetic pressure with noise in real time space: zoom-in view 2.

$$p_{wf}^{(i)} = P_i - \sum_{j=1}^{i-1} \frac{(q_j - q_{j-1}) B\mu}{k} \left(\log(t_i - t_j) + C\right) \tag{3}$$

With Equation 3, Section 3.2 solves the problem by learning in a selected Hilbert space.

3.2 Application in Hilbert Space

With Equation 3, we may map each input vector $x^{(i)} = [1; q^{(i)}; t^{(i)}]^T$ by a function ϕ, shown in Equation 4.

$$\phi\left(x^{(i)}\right) = \begin{pmatrix} 1 \\ \sum_{j=1}^{i-1}\left(q^{(j)} - q^{(j-1)}\right) \\ \sum_{j=1}^{i-1}\left(q^{(j)} - q^{(j-1)}\right)\log\left(t^{(i)} - t^{(j)}\right) \end{pmatrix} \tag{4}$$

With this mapping, the pressure transient P_{wf}, Equation 3 may be written as

$$P_{wf}^{(i)} = \theta^T \phi\left(x^{(i)}\right) \tag{5}$$

So instead of feeding the learning algorithm $h_\theta(x) = \theta^T x$ with data $x^{(i)}$, we will feed the with vector $\phi\left(x^{(i)}\right)$. In this selected Hilbert space, the learning hypothesis becomes

$$h_\theta\left(\phi\left(x^{(i)}\right)\right) = \theta^T \phi\left(x^{(i)}\right) \tag{6}$$

So to train the learning algorithm, we just need to estimate θ by stochastic gradient descent method. After the hypothesis θ is obtained, we may give a pressure transient prediction with any given flow rate history by Equation 5. Besides accurate prediction, we would also like to interpret the reservoir parameters like P_i and k. It is actually very straight forward after θ is obtained. Comparing with Equation 3, we can get

$$\begin{cases} P_i = \theta_0 \\ C = \frac{\theta_1}{\theta_2} \\ k = \frac{B\mu}{\theta_2} \end{cases} \tag{7}$$

After training the learning hypothesis with all $\phi\left(x^{(i)}\right)$, the reservoir parameters obtained are listed in Table 1. The results are very close to the true values of the reservoir parameters (Figure 14). The trend of the prediction is very good, but the curve is oscillating because the flow rate history is noisy. Section 3.3 improves the training process by imposing a pre-processing on the noisy flow rates.

Table 1: Parameter Interpretation from Machine Learning

Parameters	True Value	Learning Value
P_i	5000	4989
k	20	20.89

3.3 Smooth Flow Rate by Edge Preserving Filter

We would like to smooth the flow rate, because the prediction would be corrupted by the noise in flow rate. However, simple smoothing techniques can blur the edges at transition positions, and introduce error in all the data that follows. As a result, we come up with the idea to use edge-preserving filters widely used in computer vision community. Specifically, we choose to use bi-lateral filter, which in essence is described by

$$W(x;x_i) = \exp\left[-\left(\frac{(f(x)-f(x_i))^2}{\sigma_f^2} + \frac{\|x-x_i\|^2}{\sigma_x^2}\right)\right] \tag{8}$$

The weight of data x to x_i combines both magnitude and spatial differences, in contrast to normal filters taking into account only spatial information.

By smoothing the flow rate with Edge Preserving Filter first and then applying the machine learning algorithm discussed in Section 3.2, the results are much better, shown in Figure 15 and Table 2.

Table 2: Parameter Interpretation from Machine Learning with Pre-processing on flow rates

Parameters	True Value	Learning Value
P_i	5000	4997
k	20	20.07

4 Summary

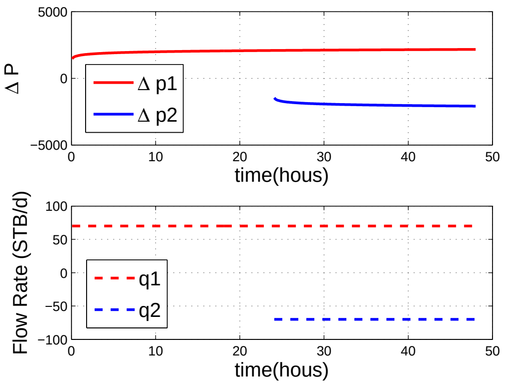
In this work, we first tried LWPR in real time and Laplace space to learn the underlying model of well-testing data. The prohibitive computation cost lead us to re-consider the problem, and come up with the idea to apply superposition to re-organize the data and put them into a unified linear model. Based on this model, gradient descent is used to learn the model parameters, which reveals the desired physical metrics of the well. Finally we utilize edge-preserving filter to smooth flow rate and achieve further improved accuracy.

References

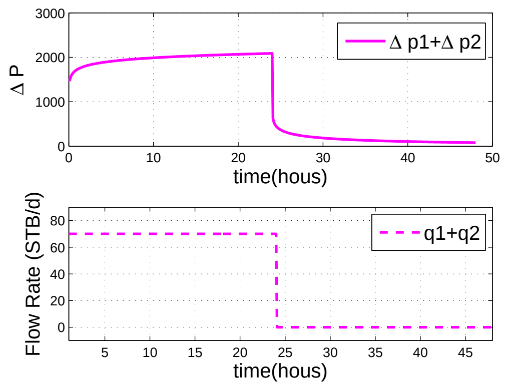
[1] R. N. Horne. Modern Well Test Analysis. *Petroway Inc.*, 1995.

[2] C. Atkeson, A. Moore, and S. Schaal. Locally Weighted Learning. *Artificial Intelligence Review*, Vol. 11(4), pp 76-113, 1997.

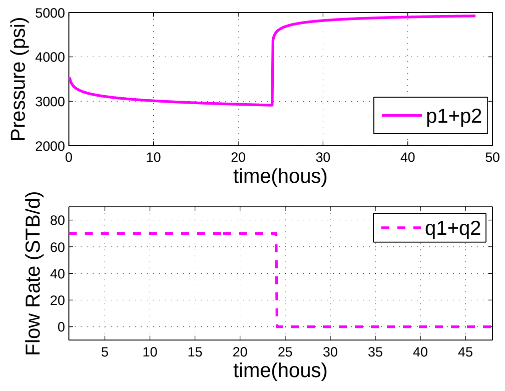
[3] A. Ng. Machine Learning Lecture Notes. *Stanford University*, 2009.



(a)



(b)



(c)

Figure 12: A demonstration of super position: (a) two separated constant flow and their pressure drop, (b) the combination of the two constant flow forms a varied flow and its corresponding pressure drop, and (c) the varied flow rate and the corresponding pressure transient when the initial pressure is considered.

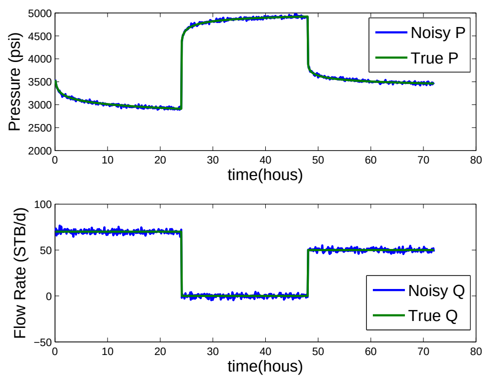


Figure 13: Noisy pressure transient and noisy varied flow rates from a modern well test.

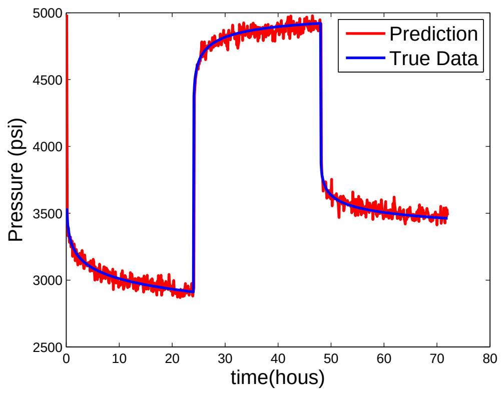


Figure 14: Pressure prediction after machine learning with noisy varied flow rates.

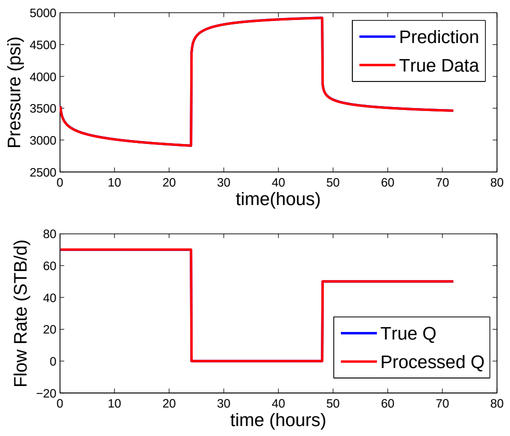


Figure 15: Pressure prediction after machine learning with smoothed varied flow rates.