Surrogate-Based Optimization Using Machine Learning

Mohammad (Jabs) Aljubran
Energy Resources Engineering Department
Stanford University
aljubrmj@stanford.edu

Christian Tae
Department of Electrical Engineering
Stanford University
ctae@stanford.edu

1. INTRODUCTION

Geothermal and hydrocarbon reserves represent major natural underground resources that fuel the energy sector. Engineering these assets involves computational flow dynamics (CFD) to understand flow behavior in the rock porous media. Modern wells are drilled horizontally and equipped with state-of-the-art instrumentation which provides continuous flowrate data streams in space and time. One major component of these completions is the downhole inflow control valve (ICV) which regulates the flow of reservoir fluids from different branches. Given multiple inputs and outputs, optimization of these designs is a major task.

Using a solver, e.g. genetic algorithm (GA), to optimize over a numerical CFD simulator is computationally expensive. Hence, surrogate-based optimization (SBO), seen in **Fig. 1**, is a common approach to efficiently find the optimal solution. Machine learning (ML) techniques hold enormous potential in building models that provide accurate, fast, and computationally inexpensive proxies that feed the SBO loop and map the objective function with minimal sample points, allowing the optimization solver to locate the global optimum.

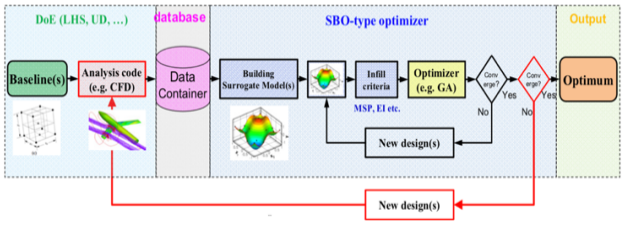

Fig. 1—SBO Approach: Initial space-filling sample is used to fit a surface response for optimization. Adaptive sampling and optimization continue repeatedly until convergence [8].

Using an oil and water reservoir model with a single trilateral well that is equipped with three downhole ICV devices, this project will focus on the regression supervised learning objective of predicting cumulative oil and water surface production rates (y_1, y_2) over time (x_1) based on the three ICV settings (x_2, x_3, x_4), as illustrated in **Table 1**, using support vector machines (SVM) and fully connected artificial neural networks (ANN). Data is generated using a physics-based, numerical reservoir CFD simulator, and split into training, validation, and testing to verify model fit and generalization. Ultimately, the ML statistical model can replace the numerical counterpart in the SBO process with comparable accuracy while substantially reducing the computational cost.

Table 1—Prediction Task Example: Predicting production at different cases (A – F)

Variables Over Time (x_1)	A	B	C	D	E	F
ICV #1 (x_2)	10	8	7	0	3	2
ICV #2 (x_3)	10	10	5	6	4	0
ICV #3 (x_4)	10	0	7	2	5	1
Oil (y_1) **and Water** (y_2)	Given Data			Predicted Output		

2. LITERATURE REVIEW

ML algorithms were applied to various aspects of underground resources. Geothermal applications included revealing cyclic patterns of seismic spectra in California Geysers field using K-Means clustering [10], predicting the geothermal heat flux in Greenland with ~85% accuracy using gradient boosted regression tree [17], and generating spatial temperature maps in Cyprus using fully connected ANN [11].

Recent Hydrocarbon reservoir research and development adopted many ML approaches to model underground fluids and porous rock aspects. These efforts included modelling crude oil phase behavior and rock permeability in heterogeneous reservoirs with over 0.8 correlation coefficient (R^2) using Gaussian-kernel SVM regression [3,5], forecasting multiphase underground fluid flow rates based on production line pressure and temperature using fully connected ANN

trained with imperialist competitive algorithm optimizer [2], and predicting North Sea wire-line log rock porosity using fully ANN with over 0.92 R^2 [9].

More rigorous efforts involved using ML to build a CFD reservoir model by learning historical patterns based on real and/or numerically-generated synthetic data without explicitly programming the physics of the flow. Various implementations involved pressure, temperature, and hydrocarbon flow rate prediction over time in heterogenous, single- and multi- phase reservoirs using feature-based kernel ridge regression, SVM, stochastic gradient descent, decision tree, random forest, AdaBoost, gradient boosting, k-nearest neighbors, fully connected ANN, and nonlinear autoregressive exogenous (NARX) models. Gaussian kernel ridge regression, SVM, and NARX were found superior in predicting single-phase flow rate based on temperature, multi-phase (oil and water) flow rate in multi-well fields, feature-free scenarios when knowledge about the physics of the problem is limited, respectively [16, 18, 19, 20].

Limited efforts involved applying ML in ICV-equipped, multilateral wells to build a proxy for predicting multi-phase flow. Recent efforts compared the performance of a 10-neuron, 1-layer fully connected ANN to other proxy construction techniques for different training dataset sizes with results reaching up to 0.95 R^2 [1]. Hence, the current study will expand on this area and explore several ML techniques and architectures to build an efficient prediction proxy.

3. DATASET

3.1 Data Generation

Eclipse, a numerical reservoir CFD simulator, is used to construct a homogenous oil reservoir model with two-phase flow (oil and water), mainly driven by an underlying water aquifer, seen in **Fig. 2**. A segmented, trilateral producer well is located at the grid center and completed across the reservoir top layer with three ICV devices (each holds 11 discrete settings from 0 to 10 indicating constriction area range of 0 to 0.001 ft^2) installed at each lateral tie-in segment.

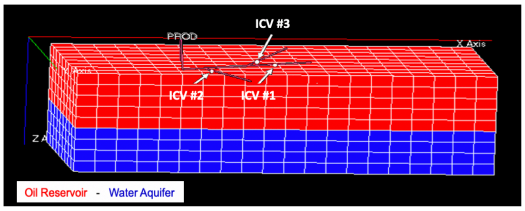

Fig. 2—Eclipse Model: Cartesian reservoir model is initially at 100% oil saturation on top (red) and 100% water saturation on bottom (blue). A branched producer is located at the center with three ICV devices.

A total of 1,330 simulation runs (Equivalent to 11^3 which represents all possible ICV setting combinations, excluding the case with no production where all ICV settings are set at 0—fully shut off) were performed to output oil and water surface flow rates at each ICV over a period of 4,000 days. Using an eight-day timestep, a total of 667,660 temporal sample points is generated. The arithmetic means of oil and water production (y_1, y_2) are 2.16E6 and 2.03E6, respectively. Only 10% (133 runs, equivalently 66,766 temporal points) of the total 1,330 runs is used for training and validation while the remaining 90% is left out for testing. This is because the SBO objective is to construct an accurate and computationally inexpensive model that replicates the numerical simulator with the smallest training and validation dataset. Of that 10%, training and validation account for 80% and 20%, respectively.

3.2 Latin Hypercube Sampling (LHS)

With only 133 simulation runs for training and validation, these should be selected carefully to represent the global domain of ICV setting combinations. A suitable sampling technique is LHS, seen in **Fig. 3**, which is commonly used in Monte Carlo simulation sampling to reduce the variance given a fixed number of sample points from a multidimensional domain [21]. Compared to random sampling, LHS provides a more representative space sample, allowing for training a more robust ML model fit with a smaller set of simulation runs, which is aligned with SBO.

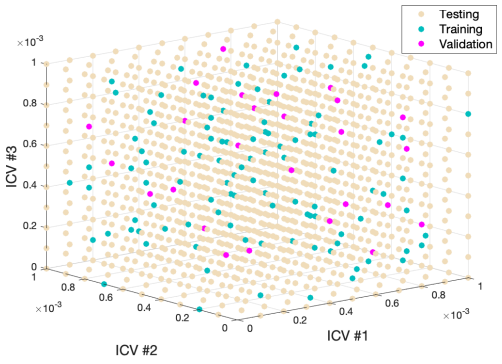

Fig. 3—LHS Design: LHS is used to select the training and validation samples, such that it optimally fills the three-dimensional space of ICV setting combinations.

3.3 Standardization

Standardization reduces the time required to find the optimal parameters while training as it limits oscillation before reaching the loss minimum [7]. It reshapes the cost function into a circle in two dimensions and sphere in three dimensions which allows the optimizer to converge in a smaller number

of iterations [18]. As seen in **Eq. 1**, input features are normalized and scaled using their corresponding mean μ_j and standard deviation σ_j. Training data means and standard deviations are also used to transform the validation and testing data.

$$x'_j = \frac{x_j - \mu_j}{\sigma_j} \qquad \textbf{Eq. 1}$$

4. METHODS

This project investigates the potential of kernel SVM regression and fully connected ANN in statistically capturing the underlying physics of the numerical model. Different kernels and architectures are explored and compared based on mean-square error (MSE) and bias-variance tradeoff diagnostics. Sensitivity analysis and learning curve evaluation are used to tune hyperparameters and avoid underfitting or overfitting the training and validation dataset.

All input features ($x_1 - x_4$) will be utilized as they are physically critical to the behavior of the response variables. Meanwhile, the first few models will be trained to predict y_1 only. Afterwards, y_2 will be included along with y_1 as output variables in a single, combined model as they are physically dependent.

4.1 SVM Regression[1]

SVM regression is a large-margin algorithm that finds parameters W and b to map $f(x) = WX + b$ using the input (X) and output (y) training examples. This is achieved by solving the optimization problem given in **Eq. 2** which uses regularization term C over m training examples with penalty terms (ξ_i^+, ξ_i^-) for the soft margin ϵ.

$$\min_{w,b,\xi^+ + \xi^-} \frac{1}{2} W^T W + C \sum_{i=1}^{m} (\xi_i^+ + \xi_i^-)$$

$$s.t. -\epsilon - \xi_i^- \le y_i - W^T x_i - b \le \epsilon + \xi_i^+ \qquad \textbf{Eq. 2}$$

In order to fit non-linear curves, lower dimensional features can be mapped to high-dimensional space using kernel functions. Tuning of the respective hyperparameters is required for optimal regression. Given feature vectors x_i, x_j, **Eqs. 3-5** show the kernel functions to be used in this study.

$$\text{Linear: } K(x_i, x_j) = x_i^T x_j \qquad \textbf{Eq. 3}$$

$$\text{Polynomial: } K(x_i, x_j) = (1 + x_i^T x_j)^d \qquad \textbf{Eq. 4}$$

$$\text{Gaussian: } K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \qquad \textbf{Eq. 5}$$

1 The MATLAB implementation was followed to program these algorithms: https://github.com/aljubrmj/CS229.git

4.2 Fully Connected ANN[1]

ANN is a fully interconnected network of layers and neurons which feeds input examples $x^{(i)}$ forward to layers, and activates neurons with activation functions to map the corresponding output examples $y^{(i)}$. Backpropagation is then applied to compute the gradient of the cost function, and thus update the ANN weights (Θ).

These weights are initiated using the Nguyen-Widrow algorithm which ensures that the initially active region of the network neurons is evenly distributed to cover the input space [15]. Meanwhile, logarithmic sigmoid is used as the hidden layer neuron transfer function while a purely linear transfer function is used at the output layer neurons. As seen in **Eq. 6**, regularized mean-square error (MSE) is used as the cost function ($J(\Theta)$) in this application where m is the number of training examples, K is the number of output variables, λ is the regularization term, L is the number of layers, s_l is the number of neurons in the l^{th} layer.

$$J(\Theta) = \frac{1}{2m} \sum_{i=1}^{m} \sum_{k=1}^{K} \left(y_k^{(i)} - \hat{y}_k^{(i)}\right)^2 + \frac{\lambda}{2m} \sum_{l=1}^{L-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} \left(\Theta_{j,i}^{(l)}\right)^2 \qquad \textbf{Eq. 6}$$

Note that in the presence of activation functions, the ANN cost function is usually nonconvex and requires a robust optimization solver. To locate the optimal ANN weights, the Levenberg-Marquardt algorithm (LMA) is used as it is generally stable and fast [12, 14]. It alternates between the Gauss-Newton and gradient descent methods to approximate the Hessian matrix by using a performance/learning gain measure (mu) that is adjusted adaptively throughout the learning process to control the learning rate and ensure robust convergence. Meanwhile, the regularization parameter λ is optimized using the Bayesian regularization backpropagation (BRB) algorithm which locates and trains on effective weights while effectively turning off the irrelevant counterparts [4, 6, 13].

5. RESULTS

5.1 SVM Kernels

Using only y_1 as output, linear (denoted by 'degree 1' for convenience) and polynomial (degrees of 2-10) kernels are examined with 20% holdout validation for varying regularization C values which were tuned repeatedly before selecting the shown range in **Fig. 4**. Similarly, Gaussian kernel hyperparametric sweep over C and γ with 20% holdout is seen in **Fig. 5**.

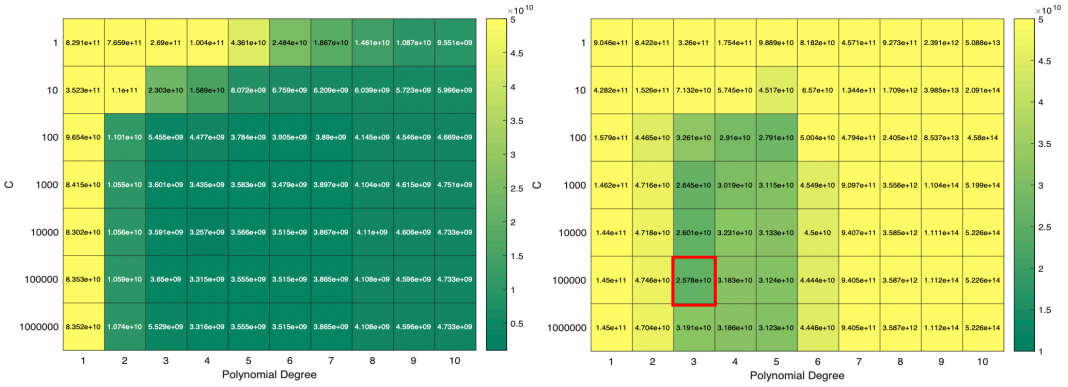

Fig. 4—Linear/Polynomial Kernel SVM: Training (left) and validation (right) MSE with best model boxed in red.

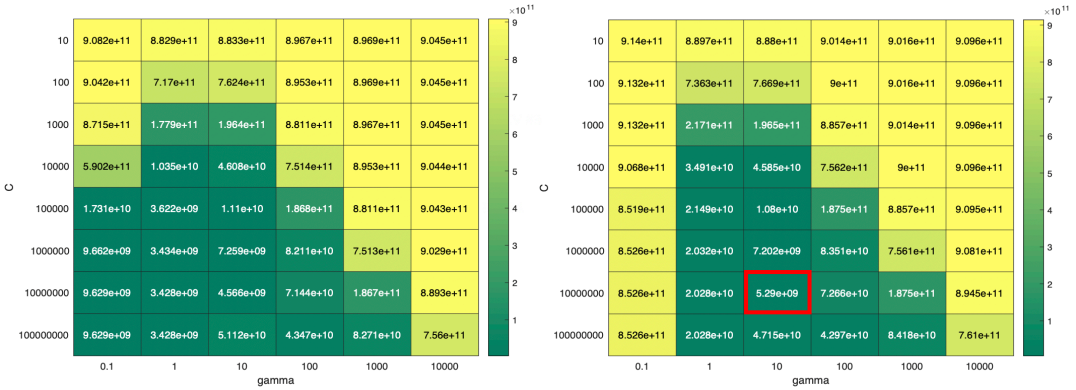

Fig. 5—Gaussian Kernel SVM: Training (left) and validation (right) MSE with best model boxed in red.

With no regularization $\lambda = 0$, ANN is trained using LMA to predict y_1 with holdout validation. **Figs. 6 and 7** show training and validation MSE sensitivity to the number of neurons and layers, respectively. Note that the MSE magnitudes in the case of deeper ANN architectures (more than one layer) are significantly lower than those with shallow ANN architectures (one layer).

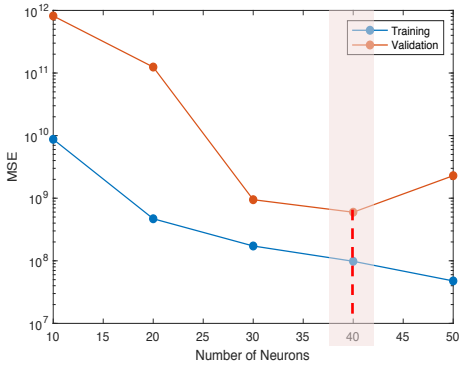

Fig. 6—ANN Neurons: Training and validation MSE plotted as a function of number of neurons for a single-layer ANN with the 40-neuron model showing best fit.

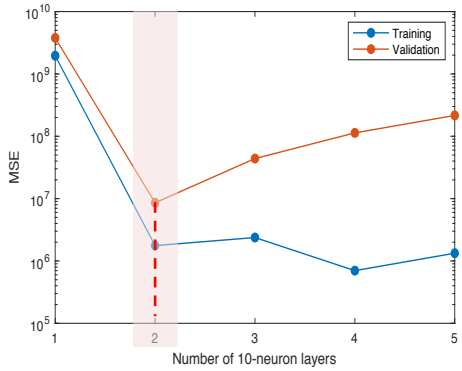

Fig. 7—ANN Layers: Training and validation MSE plotted as a function of number of layers (10 neurons each) with the 2-layer ANN showing best fit.

6. Discussion

Starting with the linear and polynomial kernel SVM, Fig. 4 indicates high training and validation MSE as C and polynomial degree decrease due to the high bias (underfit), and low training MSE but high validation MSE as C and polynomial degree increase due to the high variance (overfit). The right fit is found at the minimum validation MSE with a 3rd degree polynomial kernel and C value of 1E5, yielding testing MSE of 9.2E9. Meanwhile, Fig. 5 shows Gaussian kernel performance

where decreasing C and increasing γ result in high bias (underfit) while increasing C and decreasing γ cause high variance (overfit). The right fit is found at the minimum validation MSE with C and γ values of 1E6 and 10, respectively, yielding testing MSE of 1.2E10 which is higher than that of the 3rd degree polynomial kernel, making the latter the best SVM kernel.

Figs. 6 shows the effect of only increasing neuron count while maintaining a single layer. A small number of neurons results in a high-bias underfit while an excessively large number of neurons results in a high-variance overfit to the data. In a 1-layer ANN, a total of 40 neurons is found to give the best fit to the data with testing error of 9.0E9 which barely outperforms the aforementioned 3rd degree polynomial kernel SVM.

Fig. 6 shows that adding depth to the architecture significantly decreases MSE. A 10-neuron, 2-layer ANN was found to outperform the aforementioned architectures in predicting y_1, with testing MSE of 7.5E7. Moreover, this architecture was re-trained with BRB regularization, lowering the validation MSE to 5.3E6; however, the testing MSE increased to 2.7E8. One explanation is that incorporating BRB and validation simultaneously could have resulted in overfitting the training and validation data. Thus, introducing BRB requires further tuning of the architecture for optimal results, and/or additional training examples and/or output variables to enrich the learning process with more information.

Next, both oil and water production (y_1, y_2) are used as output variables, and the learning curve analysis process is repeated. A 10-neuron, 3-layer ANN is the best-performing model when trained with validation and BRB, yielding validation MSE of 7.4E6, testing MSE of 2.2E8 (Note that these MSE values cannot be compared directly to the cases with only y_1 as output), and testing mean correlation coefficient (R2) of 0.99, seen in **Fig. 9**.

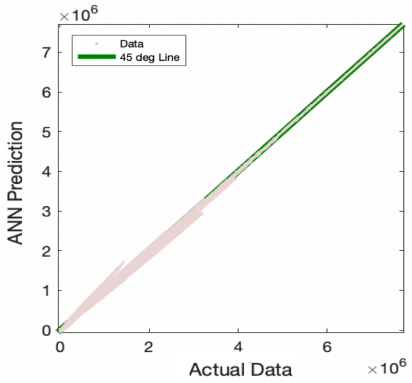

Fig. 9—Regression Plot: predicted and actual data are compared for the testing dataset, showing an excellent match with 0.99 R^2.

Note that the ANN predictions match the actual data accurately at higher production magnitudes and slightly deviate and scatter at lower magnitudes, closer to 0 where some of the ICV devices are shut off. Hence, if adaptive sampling is applied in the case of SBO to further improve the ML proxy, more training examples would be required at these lower ranges.

To further verify the accuracy of this ML fit for SBO applications, an integer nonlinear programming optimization problem is used as a metric. As seen in **Eq. 7**, the objective is to find the optimal ICV settings (x_2, x_3, x_4) that maximize profit after $x_1 = 2{,}000$ days of production. Revenue comes from selling produced oil $g(x)$ at price (p_o) of 50 \$/barrel while cost is incurred from managing produced waste water $h(x)$ with cost (c_w) of 20 \$/barrel.

$$\max_x f(x) = p_o\, g(x) - c_w\, h(x) \qquad \textbf{Eq. 7}$$

$$s.t. \quad \begin{cases} x_2, x_3, x_4 \le 10 \\ x_2, x_3, x_4 > 0 \\ integers \end{cases}$$

GA optimization solver is used to optimize once using Eclipse, the numerical reservoir simulator, to find $g(x)$ *and* $h(x)$ and another using the ANN proxy. **Table 2** shows that the using ANN as proxy in SBO saves 88% of total optimization time compared to using the complex numerical simulator directly. The approximate optimal ICV setting and profit are also very close to those predicted using the time-consuming numerical simulator with a considerable small difference in the setting of ICV #1 (7 versus 10). This exercise proves that the achieved model is sufficiently accurate based on this 'application-based metric'.

Table 2 —SBO Optimization: ANN proxy is compared to the numerical simulator in SBO

Proxy	$x^* = [x_2, x_3, x_4]$	$f(x^*)$	Time
ANN	[7 2 1]	\$ 90,989,317	2.0 hrs
Eclipse	[10 2 1]	\$ 90,898,380	16.7 hrs

7. CONCLUSIONS

The goal of the study was to replicate a numerical CFD reservoir simulator using ML for SBO applications. Tuned 3rd degree polynomial kernel showed the best performance compared to the other SVM kernel counterparts. Meanwhile, ANN was found superior in predicting oil and water production rates when compared to SVM. ANN model accuracy was verified for SBO applications using a CFD optimization problem. Future work could investigate the use of recurrent neural networks (RNN) as to highlight the sequential nature of the data at hand. Also, heterogeneous gas reservoir CFD models could be explored.

CONTRIBUTIONS

Mohammad (Jabs) generated the synthetic data using Eclipse reservoir simulator. Christian implemented and tuned SVM while Jabs implemented and tuned ANN. Both contributed equally to the development of this report.

REFERENCES

[1] Abukhamsin, A.Y. (2017). Inflow profiling and production optimization in Smart wells using distributed acoustic and Temperature measurements (Unpublished doctoral dissertation). Stanford University, CA.

[2] Ahmadi, M. A., Ebadi, M., Shokrollahi, A., & Majidi, S. M. J. (2013). Evolving artificial neural network and imperialist competitive algorithm for prediction oil flow rate of the reservoir. Applied Soft Computing, 13(2), 1085-1098.

[3] Al-Anazi, A., & Gates, I. D. (2010). A support vector machine algorithm to classify lithofacies and model permeability in heterogeneous reservoirs. Engineering Geology, 114(3-4), 267-277.

[4] Burden, F., & Winkler, D. (2008). Bayesian regularization of neural networks. In Artificial neural networks (pp. 23-42). Humana Press.

[5] El-Sebakhy, E. A. (2009). Forecasting PVT properties of crude oil systems based on support vector machines modeling scheme. Journal of Petroleum Science and Engineering, 64(1-4), 25-34.

[6] Foresee, F. D., & Hagan, M. T. (1997, June). Gauss-Newton approximation to Bayesian learning. In Proceedings of International Conference on Neural Networks (ICNN'97) (Vol. 3, pp. 1930-1935). IEEE.

[7] Grus, J. (2015). Data science from scratch. Sebastopol, CA: O'Reilly. pp. 99, 100. ISBN 978- 1-491-90142-7.

[8] Han, Z.-H. (2016). A generic surrogate-based optimization code for aerodynamic and multidisciplinary design. 30th Congress of the International Council of the Aeronautical Sciences. Daejeon, Korea.

[9] Helle, H. B., Bhatt, A., & Ursin, B. (2001). Porosity and permeability prediction from wireline logs using artificial neural networks: a North Sea case study. Geophysical Prospecting, 49(4), 431-444.

[10] Holtzman, B. K., Paté, A., Paisley, J., Waldhauser, F., & Repetto, D. (2018). Machine learning reveals cyclic changes in seismic source spectra in Geysers geothermal field. Science advances, 4(5), eaao2929.

[11] Kalogirou, S. A., Florides, G. A., Pouloupatis, P. D., Panayides, I., Joseph-Stylianou, J., & Zomeni, Z. (2012). Artificial neural networks for the generation of geothermal maps of ground temperature at various depths by considering land configuration. Energy, 48(1), 233-240.

[12] Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. Quarterly of applied mathematics, 2(2), 164-168.

[13] MacKay, D.J. (1992). Bayesian Interpolation. Neural Computation, Vol. 4, No. 3, 1992, pp. 415 to 447).

[14] Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. Journal of the society for Industrial and Applied Mathematics, 11(2), 431-441.

[15] Nguyen, D., & Widrow, B. (1990, June). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In 1990 IJCNN International Joint Conference on Neural Networks (pp. 21-26). IEEE.

[16] Orta Aleman, D.I. (2018). Fracture flow rate estimation using machine learning on temperature data (Unpublished master's thesis). Stanford University, CA.

[17] Rezvanbehbahani, S., Stearns, L. A., Kadivar, A., Walker, J. D., & van der Veen, C. J. (2017). Predicting the geothermal heat flux in Greenland: a Machine Learning approach. Geophysical Research Letters, 44(24).

[18] Ristanto, T. (2018). Machine learning applied to multiphase production problems (Unpublished master's thesis. Stanford University, CA.

[19] Tian, C. (2014). Applying machine learning and data mining techniques to interpret flow rate, pressure and temperature data from permanent downhole gauges (Unpublished master's thesis). Stanford University, CA.

[20] Tian, C. (2018). Machine learning approaches for Permanent downhole gauge data interpretation (Unpublished doctoral dissertation). Stanford University, CA.

[21] Zolan, A. J., Hasenbein, J. J., & Morton, D. P. (2017, December). Optimizing the design of a latin hypercube sampling estimator. In 2017 Winter Simulation Conference (WSC) (pp. 1832-1843). IEEE.