ESP OPTIMIZATION USING REAL-TIME SIMULATION

Lawrence Youngblood, Statistics & Control, Inc., lyoungblood@stctrl.com

Synopsis

Electric submersible pump (ESP) systems are effective artificial lift methods to pump production fluids to the surface; however, running these pumps outside of their specified ranges is both inefficient and causes equipment damage. Additionally, many failure tracking systems used to monitor ESP run life only track basic failure attributes. Predictive analytics is a data mining technique that extracts information from data and uses it to predict trends and behavior patterns that can be applied to the past, present, and future.

Predictive analytics—combined with highly accurate, dynamic models used for real-time simulation and optimization—provide an effective method to visually represent and analyze equipment performance degradation. This analysis in turn allows for planning, scheduling, and dynamic batch optimization. Operators and decisions makers alike may view the information generated through predictive analytics via a dynamic web interface.

This paper describes how real-time modeling, auto-tuning, and simulation may be used to track basic ESP attributes, optimize operation, and allow for a more holistic view of the system. Additionally, information about how this solution is currently being applied to a site with 1800 wells is provided. This solution helps reduce costs, extend the life cycle, and improve ESP availability and reliability.

Introduction

Natural reservoir production is in constant decline; generally speaking, the reservoir decline rate is 5% to 7% each year, although every reservoir decline rate is different. In order to boost production and prolong the longevity and effectiveness of reservoirs, artificial lift has been applied and is currently estimated as being used in approximately 90% of the wells worldwide. ESPs are extensively used because they are an effective, reliable artificial lift method to pump production fluids to the surface.

ESPs have many advantages, such as being a relatively cost-effective method, low maintenance, and able to withstand harsh environments. They are also versatile and can be used to pump disposal or injection fluids, liquid petroleum products, and treatment chemicals, among others.[1] Despite these advantages, running inefficient ESPs increases the operating and maintenance costs (such as the expensive operation pulling), reduces production (both in general and also lost production if operation is pulled), and also decreases the life of the ESP.

This paper presents how real-time modeling, auto-tuning, and simulation may be used to track basic ESP attributes, optimize operation, and allow for a more holistic view of the system. A key concept for modeling, auto-tuning, simulation, and optimization is predictive analytics, which can be defined as “an area of data mining that deals with extracting information from data and using it to predict trends and behavior patterns. Often the unknown event of interest is in the future, but predictive analytics can be applied to any type of unknown whether it be in the past, present or future.”[2] Figure 1 shows analytics maturity and how predictive analytics fits in the progression.

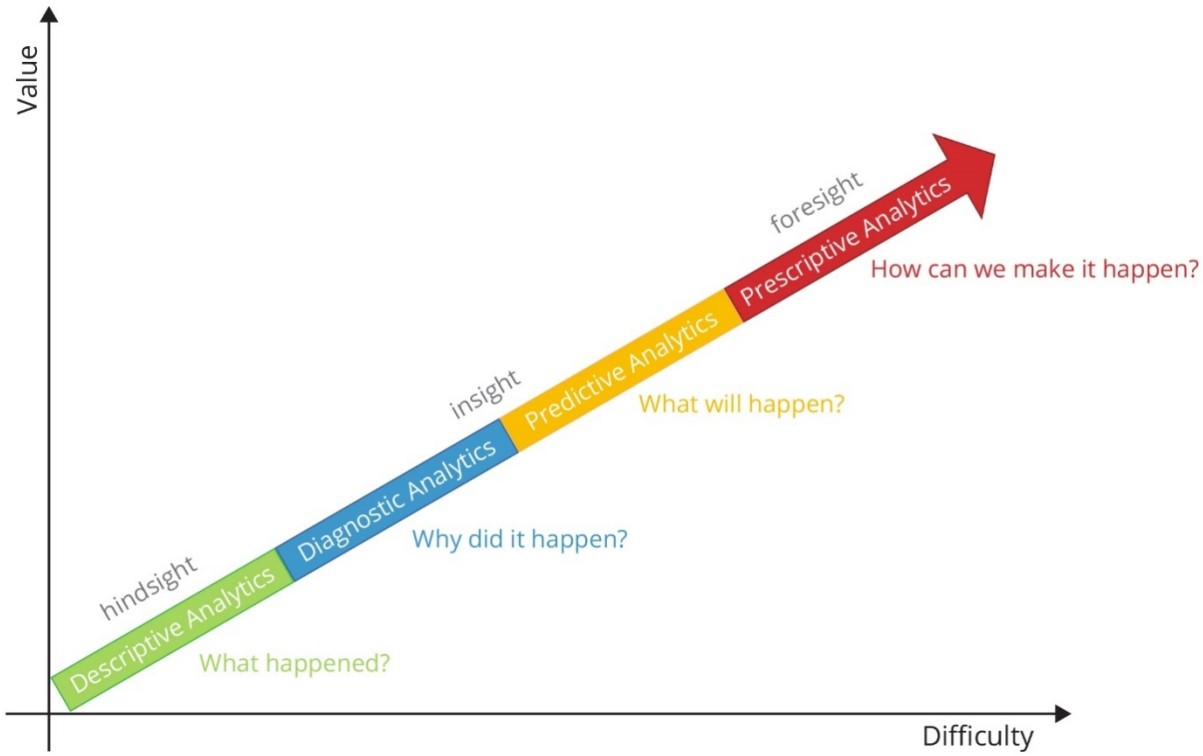

Figure 1. Analytics maturity

A site with 1800 wells is used as a business case to demonstrate the approach and techniques to optimize ESP operation using real-time simulation.

Development

Model Construction and Self-Learning

The first step in this approach is to construct a baseline model that will be used for simulation and optimization. Modeling consists of three components: data processing, model construction, and self-learning. Data processing focuses on data import, storage, and noise reduction methods. Model construction is based on manufacturer specifications and process design data and uses an array of statistical and mathematical modeling techniques to predict target outcomes. Finally, self-learning combines model parameter tuning and monitoring techniques to ensure continuous quality performance of each process model. Figure 2 illustrates the interactions between these three modeling components.

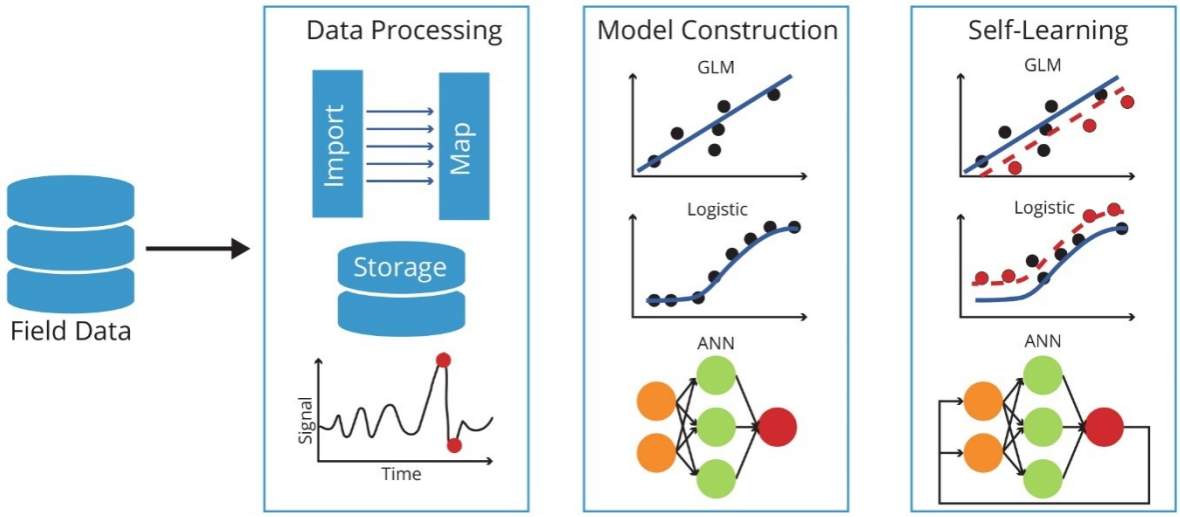

Figure 2. Model component interactions

Data Processing

Data come from a number of sources, including several hardware interfaces, and from equipment from different manufacturers. To overcome this challenge, a platform- and vendor-neutral tag aggregator is used as a central information hub to store, share, map, and process all data. This tag aggregator uses all standard interfaces, including OPC, OPC UA, and ODBC and interfaces to link to SCADA/DCS/PLC, in order to collect data from all data sources. The tag aggregator also allows key performance indicators (KPIs) to be configured during data collection. KPIs help quickly understand process conditions and make better and faster decisions based on usable data.

Collected data are processed to filter raw data, detect outliers, and reduce noise. Process variables can be defined as discrete signals that are measured at a transmitter frequency for real-time analysis and at a predefined frequency for simulation purposes. Data can be transformed (filtered) using linear or nonlinear transformations.

Linear transformations are used for scaling purposes. A linear transformation is a map $f : X \to Y$ that satisfies the following conditions: $f(\alpha x) = \alpha f(x)$ and $f(x_1 + x_2) = f(x_1) + f(x_2)$, where X and Y are the linear signal spaces, $x_1, x_2 \in X$, and α is a real coefficient.

Most transformations are nonlinear in nature because they violate one of the linear transformation assumptions. The main nonlinear transformations are Z-transform, discrete-time Fourier transform, Kalman filter, and regression (commonly ordinary least squares).

Processed and smoothed data then undergo "data justification," where data outliers are rejected whenever they fall beyond a specified distance from expected model values or whenever user-defined criteria are exceeded. In particular, let $\hat{x}(t)$ be the smoothed value of measured variable x at time t. The algorithm then creates an indicator variable R that drives rejection logic according to equation (1).

$$R(\hat{x}(t)) = \begin{cases} 1 \; if \; |\hat{x}(t)| \geq \delta_x \\ 0 \; if \; |\hat{x}(t)| < \delta_x \end{cases}, \tag{1}$$

where δ_x is either a calculated value based on the distribution of $\hat{x}(t)$ over all $t \in [T_0, \ldots, T_1]$—e.g., $\delta_x = \mu_{\hat{x}} \pm 2\sigma_{\hat{x}}$ (where $\mu_{\hat{x}}$ is the distribution mean and $\sigma_{\hat{x}}$ is its standard deviation)—or is a process engineer defined value (e.g., Upper Control Limit or Lower Control Limit). The output of data justification is the set of all accepted measured process variables values.

Model Construction

Complex, dynamic process models may be constructed using the least squares regression, logistic regression, and artificial neural networks.

Least squares regression is a method of fitting a hyperplane onto a cloud of points in a Euclidean space. Given a continuous target variable *y*, the method calculates coefficients β that minimize equation (2).

$$S = \sum_{i=1}^{n} \left(y_i - P\left(\beta, x_{ij}\right) \right)^2 , \tag{2}$$

where $P\left(\beta, x_{ij}\right)$ is an m[th]-degree polynomial with *k* independent variables x_j and coefficients β.

Logistic regression predicts categorical targets and, in particular, binary events (for example, compressor surge). The model estimates coefficients β for the logit function shown in equation (3):

$$\ln\left(\frac{p_i}{1 - p_i} \right) = P\left(\beta, x_{ij}\right), \tag{3}$$

where $P\left(\beta, x_{ij}\right)$ is an m[th]-degree polynomial with *k* independent variables x_j and coefficients β and $p_i = E\left(\frac{Y_i}{n_i} \mid X_i \right)$ is the probability of event *Y* occurring given the values of independent variable vector *X*.

Artificial neural networks (ANNs) are nonlinear statistical modeling techniques designed to mimic the human brain. The ANN algorithm is the multilayer perceptron (MLP). The MLP network consists of individual neurons, each of which receives *n* weighted inputs that are summarized and transferred to the neuron output.

Figure 3 displays the structure of the individual neuron with three inputs. The composition of the combination and transfer function constitute the activation function. The MLP model uses the hyperbolic tangent as the activation function.

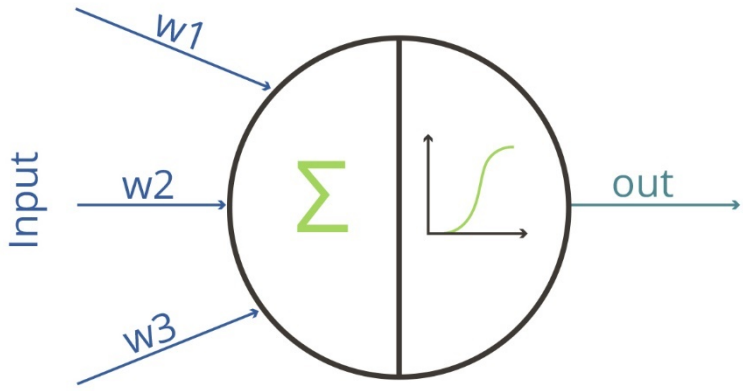

Figure 3. Artificial neuron with three inputs

MLP weights are initially chosen at random and are then adjusted such that the error function (4) is minimized on the training set:

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - F\left(x_i\right)\right), \tag{4}$$

where N is the sample size, y_i is the actual value, and $F\left(x_i\right)$ is the MLP output.

A dynamic model represents dynamic characteristics of process parameters via a set of mathematical formulas that describe the transitional function of the process. Variable values at different time frames are used to predict future fluctuations of the dependent variable.

Given the filtered dependent variable $y\left(t\right)$ at a time instance t and a vector of filtered independent variables $x_i\left(t-\tau_{x_i}\right)$—where $i=1,...,N$ and τ_{x_i} is a time shift specific to each independent variable x_i—the initial dynamic model $y\left(t\right)= f_{ini}\left(x_i\left(t-\tau_{x_i}\right)\right)$ is based on a training set collected in time interval T with a sufficient population in each subset of N-dimensional Euclidean space.

The key to building the dynamic model is the optimal value of each independent variable's time shift, which is found through simulation and genetic algorithms. The appropriate time shift allows the submodule to determine not only what but when the impact will occur given a current control action (change in the independent variable).

Self-Learning (Auto-Tuning)

In the music industry, auto-tuning is used to correct a singer's bad notes and/or wavering pitch. This same concept is applied to the model by analyzing current operating conditions as a series of equations, which will allow bad model parameters to be corrected. By tracking performance and self-learning, the model can be predicted and the accuracy can be improved without human interaction.

For OLS and logistic regression, parameters are the polynomial power and the presence of interaction terms within the polynomial. For neural networks, the parameters are the number of hidden layers and the number of neurons in each layer (the activation function is fixed as a hyperbolic tangent).

Given regression models $y = P\left(\beta, x_{ij}\right)$ or $\ln\left(\frac{p_i}{1-p_i}\right) = P\left(\beta, x_{ij}\right)$, where P is the mth-degree polynomial, the initial model build chooses m randomly and selects all polynomial terms. Each iteration of the algorithm increments/decrements the degree by a single power until goodness-of-fit statistics do not change significantly. Once the optimal polynomial degree is chosen, the interaction terms are deselected one at a time to reduce complexity of the problem without losing predictive power.

For the neural network, between one and ten hidden layers is selected, randomly varying the number of neurons between one and ten in each layer until the error function is minimized, thus allowing for automated model parameter tuning. For example, the degree in polynomial regression is chosen to minimize the selected error function, such as sum of squared errors in OLS regression or $-2\ln\left(\Lambda\right)$ in logistic regression, where Λ is the ratio of likelihood of random chance to likelihood of the selected model.

Self-learning continues in real time; as the model training sample is enriched with new data, the models with selected parameters are retrained on more recent data and the parameters are re-estimated to accommodate changes in input signals. Statistical process control is used to detect these

fluctuations. Descriptive statistics, such as mean ($\overline{X} = \frac{\sum_{i=1}^{n} x_i}{n}$), median (midpoint of the ordered values), and standard deviation ($s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \overline{X})^2}{n-1}}$) are calculated for each value, where x_i are the values of a particular process variable for $i = 1, ..., n$. Whenever the filtered signal values consistently fall outside $\left[\overline{X} - 3s, \overline{X} + 3s\right]$, which is indicative of a process change the dynamic model is retrained.

Self-learning and parameter tuning is crucial for ensuring optimal performance under severe process changes.

Real-Time Simulation

Process simulation concurrently occurs with data justification. The tuned model receives real-time data to replicate ESP and process performance. The simulation algorithm uses particle filtering methods that are based on dynamic state space models described by equation (5).

$$\begin{cases} w_t = f(w_{t-1}) \\ x_t = g(w_t) \end{cases}, \tag{5}$$

where *f* and *g* are estimated using polynomial regression, w_t is a vector of state parameters at time *t*, and x_t are observed (measured) variables. Then, w_t is estimated using sequential importance sampling or Sequential Monte Carlo sampling (from a simulated distribution).

The outputs from the data justification and the simulation algorithm—the smoothed measured variable values and corresponding simulated values—are continuously evaluated. Specifically, let *y(t)* be the simulated value corresponding to $\hat{x}(t)$. Then set $\hat{x}(t) - y(t)$. Objective function (6) is optimized by applying ordinary least squares line of best fit techniques. Ultimately, simulation parameters are adjusted and the process model is tuned to reflect predefined accuracy criteria.

$$\sum_t \left(\hat{x}(t) - y(\mathrm{t})\right)^2 \rightarrow \min \tag{6}$$

Accuracy is measured according to statistical process control concepts. The idea is to build the distribution of *dP* over time such that its mean is as close to zero as possible while most observations fall within a predefined Upper Specification Limit, *USL*. This concept is illustrated in Figure 4.

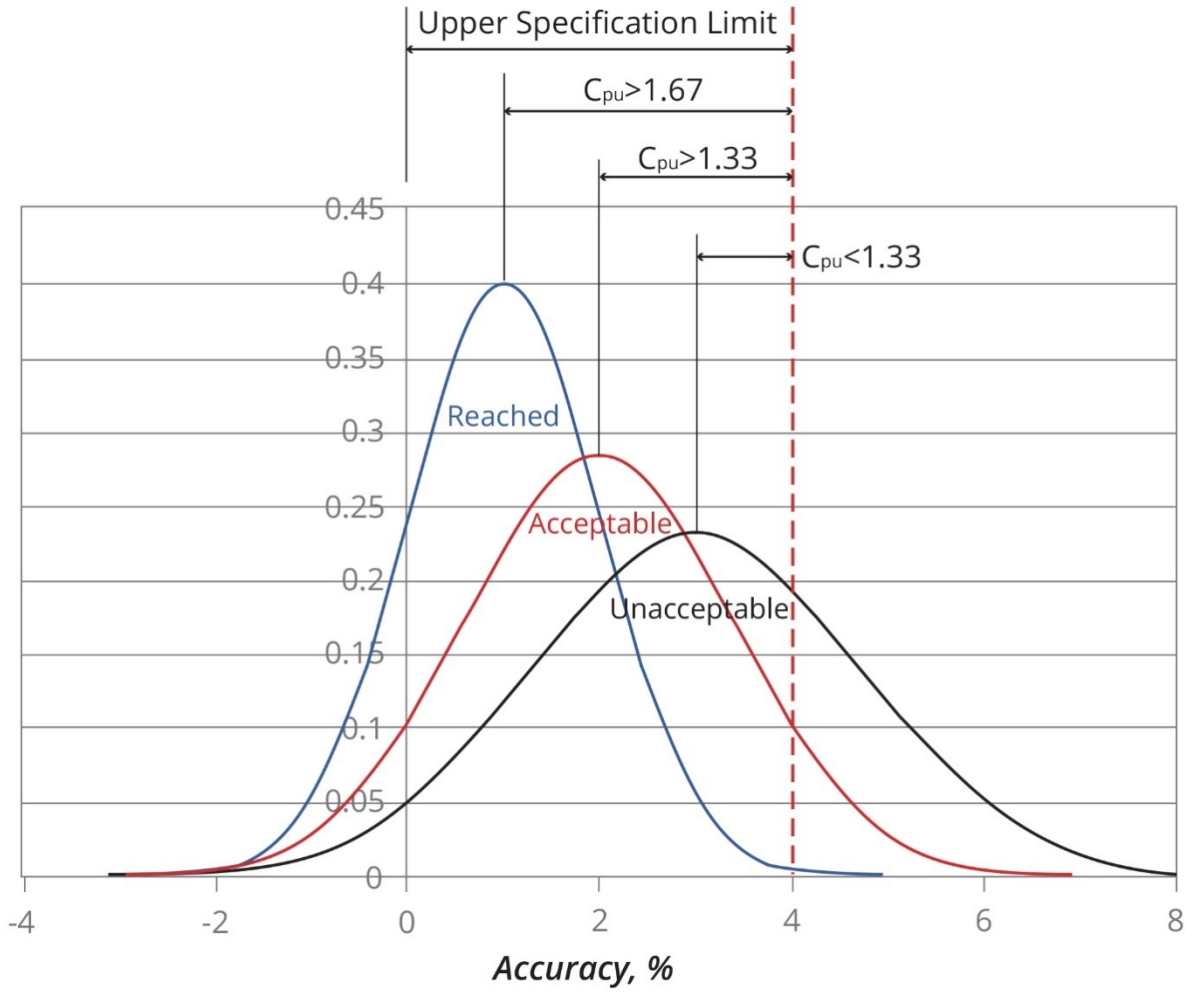

Figure 4. Accuracy concept

The algorithm starts with calculation of the so-called process capability index, which is illustrated in equation (7). Note, in the case of accuracy, calculations involve only the Upper Specification Limit.

$$C_{p,u} = \frac{USL - \mu}{3\sigma}, \tag{7}$$

where μ is the mean of *dP* distribution and σ is the standard deviation within a parametrically defined time period. Since accuracy has one-sided specifications, the demands on $C_{p,u}$ values are not as stringent as with two-sided specifications. Thus, following generally accepted statistical process control principles, the process is considered "capable" or "in control" whenever $C_{p,u} \geq 1.33$. Typically, the system learning and optimization processes will continue until $C_{p,u} \geq 1.67$. In this case, the desired accuracy level would be considered reached.

The final output of the algorithm is a set of adjusted coefficients that are provided to the process model. The overall algorithm repeats whenever process changes occur.

Batch Optimization

Batch optimization may be used to optimize a large number of ESPs at the same time as well as optimize for multidimensional objective functions.

The general schematic for optimization is as follows:

- Objective function: Criteria for optimization that could include maximizing production, minimizing energy costs, maximizing profit, or minimizing fuel gas flow
- Equations of energy and material balance: Dependence equations of energy and material balance for ESP models
- Constraints: Ambient conditions, ESP operating limits, equipment technical states, etc.
- Decision points: Set point controlled by the operator

Mass Balance Reconciliation

Mass balance reconciliation is derived from the physical law of conservation of mass, which can be summarized as "what comes in must come out." The idea is that mass or energy cannot be created or destroyed within the confines of an isolated physical system. Typically, the industry standard for an isolated physical system is either a production unit or an entire plant. The material balance of a system can be generally characterized by the relationship $Input = Output + Accumulation$.

For steady state processes with multiple flows, let $X_i(t)$, $i = 1,...,n$ denote the inflows; $Y_j(t)$, $j = 1,...,m$ denote the outflows; and $Z_p(t)$, $p = 1,...,k$ denote the accumulated quantities within each flow infrastructure. The material or energy balance is then described by equation (4):

$$\sum_{i=1}^{n} X_i(t) = \sum_{j=1}^{m} Y_j(t) + \sum_{p=1}^{k} Z_p(t) \tag{4}$$

Transition state processes with multiple flows require as additional term: τ_j, $j = 1,...,m$ represents the lag for each outflow. The material or energy balance is then described by equation (5):

$$\sum_{i=1}^{n} X_i(t) = \sum_{j=1}^{m} Y_j(t+\tau_j) + \sum_{p=1}^{k} Z_p(t), \tag{5}$$

where *t* ranges over a pre-defined timeframe $t \in [T_1, T_2]$.

One of the main concerns of process operators occurs whenever there is a discrepancy within established material or energy balances. The primary causes for such discrepancies are signal noise, equipment failure, and unknown leaks. Transforming material balance equations (4) and (5) into an optimization problem helps minimize discrepancies by finding optimal signal errors. The key assumption for data reconciliation is that there are no gross errors in the system.

Given a set of inflows $X_i(t)$, $i = 1,...,n$, outflows $Y_j(t)$, $j = 1,...,m$, and accumulations $Z_p(t)$, $p = 1,...,k$, let $\delta_{X_i}, \delta_{Y_j}, \delta_{Z_p}$ denote the tolerances for inflows, outflows, and accumulations, respectively. The tolerances are individually chosen for each flow measurement based on measurement device characteristics and confidence levels dictated by the process. Tolerances can also be defined for certain calculated variables. For example, to calculate gas accumulation within a plant's fuel network, a series of assumptions are made with respect to the fuel network volume and fuel gas composition; thus, tolerances can be defined according to the calculation's confidence level.

Material and energy balance reconciliation occurs either globally across the entire plant or at each individual production unit. The measured and calculated variable errors are estimated within the acceptable tolerances. The statistical data reconciliation method is based on least squares optimization. Let Δ_{X_i}, Δ_{Y_j}, Δ_{Z_p} denote the target errors in $X_i(t)$, $Y_j(t)$, and $Z_p(t)$, respectively. The optimization problem for steady state processes is then set up as shown in equation (6)

$$\sum_{i,j,p}\left[\left(X_i(t)+\Delta_{X_i}\right)-\left(\left(Y_j(t)+\Delta_{Y_j}\right)+\left(Z_p(t)+\Delta_{Z_p}\right)\right)\right]^2, \tag{6}$$

minimizing the equation subject to constraints $\left|\Delta_{X_i}\right| \le \delta_{X_i}$, $\left|\Delta_{Y_j}\right| \le \delta_{Y_j}$, and $\left|\Delta_{Z_p}\right| \le \delta_{Z_p}$. For transition state processes, the optimization problem is given with similar constraints, but the objective function is given by equation (7).

$$\sum_{i,j,p}\left[\left(X_i(t)+\Delta_{X_i}\right)-\left(\left(Y_j(t+\tau_j)+\Delta_{Y_j}\right)+\left(Z_p(t)+\Delta_{Z_p}\right)\right)\right]^2 \tag{7}$$

Defining each outflow lag, τ_j, occurs during the process modeling stage. Its values are used in this optimization problem as known quantities. Optimization is used to find the optimal solution to the stated problem. The solution for the optimization problem is the set of estimated measurement/calculation errors $\overline{\Delta}_{X_i}$, $\overline{\Delta}_{Y_j}$, and $\overline{\Delta}_{Z_p}$ as well as the balanced measurement/calculated signals computed via $\overline{X}_i(t) = X_i(t) + \overline{\Delta}_{X_i}$, $\overline{Y}_j(t) = Y_j(t) + \overline{\Delta}_{Y_j}$, and $\overline{Z}_p(t) = Z_p(t) + \overline{\Delta}_{Z_p}$.

Planning & Scheduling

On a daily basis, the production scheduler must balance oil production with a wide range of constraints. Constructing a production balance model provides the production scheduler with several advantages, such as simulating and optimizing an entire month's production schedule during the month, which helps avoid pitfalls of manual or rule-based scheduling.

The foundation of planning and scheduling is the material balance equation to build process operating forecasts according to planned production. Like material balance reconciliation, planning and scheduling is an optimization problem that is solved by dividing the whole problem into two parts:

1. Production forecast
2. Selection of producers and consumers

Forecasting

Various forecasting methods may be used to forecast production depending on the forecast period. Longer periods typically involve seasonality, while shorter forecast periods (for example, during process transition) do not have a seasonal component. Possible forecasting methods are exponential smoothing, double exponential smooth, Holt-Winters smoothing, and predictive modeling.

Exponential smoothing is a common forecasting technique that is similar to moving average time series forecasting. The goal is to forecast either a discrete or a continuous variable based on its historical values so that the older values get exponentially smaller weights.

Single exponential smoothing is a single parameter forecast given by equation (8):

$$S_t = \alpha \cdot X_t + (1-\alpha) \cdot S_{t-1}, \tag{8}$$

where S_t is the forecast value at time t, X_t is the series value at time t, and $0 < \alpha < 1$ is the smoothing parameter.

Due to the recursive nature of this algorithm, it is crucial to select the appropriate initial value: either the first time series element or the average of N initial series elements.

Double exponential smoothing is a useful technique when a trend component is suspected in the time series. Thus, two time series components (level and trend) need to be smoothed. The recursive algorithm for this approach is given by equation (9):

$$S_t = \alpha \cdot y_t + (1-\alpha) \cdot (S_{t-1} + b_{t-1}), \tag{9}$$

where $b_t = \beta \cdot (S_t - S_{t-1}) + (1-\beta) \cdot b_{t-1}$ and $0 < \alpha,\beta < 1$ are the smoothing parameters. The following initial value selection is allowed: $S_1 = y_1$ and $b_1 = \frac{y_n - y_1}{n-1}$ for a positive integer, n.

Holt-Winters smoothing, also known as triple exponential smoothing, is the optimal forecasting method for a time series that has both a trend and seasonality.

For multiplicative seasonality, the recursive algorithm is defined in the set of equations represented as equation (10):

$$\begin{aligned} \bar{R}_t &= \alpha \cdot \frac{y_t}{\bar{S}_{t-L}} + (1-\alpha) \cdot (\bar{R}_{t-1} + \bar{G}_{t-1}) \text{ (overall smoothing)} \\ \bar{G}_t &= \beta \cdot (\bar{S}_t - \bar{S}_{t-1}) + (1-\beta) \cdot \bar{G}_{t-1} \text{ (trend smoothing)} \\ \bar{S}_t &= \gamma \cdot \frac{y_t}{\bar{S}_t} + (1-\gamma) \cdot \bar{S}_{t-L} \text{ (seasonal smoothing)}, \end{aligned} \tag{10}$$

where $0 < \alpha,\beta,\gamma < 1$ are the smoothing parameters and L is the season length. The forecast for the next time frame is then given by equation (11).

$$y_t = (\bar{R}_{t-1} + \bar{G}_{t-1}) \cdot \bar{S}_{t-L} \tag{11}$$

This method also allows for predictive horizon forecasting T time increments ahead. This forecast is given by equation (12):

$$y_{t+T} = (\bar{R}_{t-1} + T \cdot \bar{G}_{t-1}) \cdot \bar{S}_{t+T-L} \tag{12}$$

The initial conditions are given by $\bar{G}_0 = \frac{\bar{y}_m - \bar{y}_1}{(m-1) \cdot L}$, $\bar{R}_0 = \bar{x}_1 - \frac{L}{2}\bar{G}_0$, and $\bar{S}_0 = \frac{\bar{x}_t}{\bar{x}_i - \left[\frac{(L+1)}{2} - j\right] \cdot \bar{G}_0}$, where $\bar{x}_i$ is the average for the season corresponding to the time increment t and j is the index of time increment t within the season.

For additive seasonality, the recursive algorithm is defined as the set of equations represented as equation (13):

$$\begin{aligned} \bar{R}_t &= \alpha \cdot (y_t - \bar{S}_{t-L}) + (1-\alpha) \cdot (\bar{R}_{t-1} + \bar{G}_{t-1}) \text{ (overall smoothing)} \\ \bar{G}_t &= \beta \cdot (\bar{S}_t - \bar{S}_{t-1}) + (1-\beta) \cdot \bar{G}_{t-1} \text{ (trend smoothing)} \\ \bar{S}_t &= \gamma \cdot (y_t - \bar{S}_t) + (1-\gamma) \cdot \bar{S}_{t-L} \text{ (seasonal smoothing)} \end{aligned} \tag{13}$$

The forecast for the next time frame is then given by $y_t = \bar{R}_{t-1} + \bar{G}_{t-1} + \bar{S}_{t-L}$.

Note that the smoothing parameters $0 < \alpha,\beta,\gamma < 1$ are best estimated by genetic algorithms that minimize the fitness function given by the forecast mean absolute percentage error (MAPE).

Predictive modeling is used to create regression forecasts based on predictors other than time series variables. Given the forecast target variable, ordinary least squares and maximum likelihood estimations are used to estimate the autoregressive moving average (ARMA) and the autoregressive moving average with exogenous inputs (ARMAX) coefficients. In particular, the linear version of the *ARMAX(p, q, b)* model can be defined as shown in equation (14):

$$X_t = \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \sum_{i=1}^{q} \eta_i d_{t-i}, \quad (14)$$

where the first sum is the p^{th} order autoregressive component, the second sum is the q^{th} order moving average, and the third sum is the exogenous variable time series. The autoregressive component measures the dependence of a series value on its past values, while the moving average component uses errors between series and forecast values to estimate the series' next value.

This portfolio of tried-and-true forecasting methods creates robust and accurate production forecasts.

Scheduling

ESPs efficiently and reliably lift high volumes of oil and other fluids from wellbores. Typically, each ESP is oversized for its reservoir, which means that the flow is larger than the replenishment of oil into the reservoir. In such conditions, the work schedule of a single pump may mean 10 minutes of work and 50 minutes of rest. An ESP pad is a collection of pumps that collectively produce flow; a typical pad size is 20 to 30 individual pumps, which presents a challenge in terms of flow stability. An ESP pad scheduling algorithm maximizes flow stability by minimizing variance through optimal pump start time scheduling. The idea of the algorithm is to build the well pad flow variance as a function of individual pump start times in a cycle and then minimize it using a genetic algorithm. Table 1 describes the variables used in the scheduling algorithm.

Table 1. Scheduling algorithm parameters

Parameter	Description
$i=1,\ldots,N$	Index of pumps in pad
t_i	Start time of i^{th} pump; becomes decision variables in optimization problem
τ_i	Run time duration for i^{th} pump; known as a priori and typically range between 5 and 20 minutes
K	Time index (e.g., every second or every minute in an hour)
t^K	Time between 0 and 60
$t^K, \lvert t^K \rvert = M$	Cardinality that is set by a parameter
$\delta_i(t^K)$	Indicator that i^{th} pump is on (equals 1) or off (equals 0)
$F_i(t^K)$	Flow of i^{th} pump; known as a priori

Equation (15) is used to define wehter a pump is on or off.

$$\delta_i(t^K) = \begin{cases} 1, t^K \in [t_i, t_i + \tau_i] \\ 0, t^K not \in [t_i, t_i + \tau_i] \end{cases} \quad (15)$$

The overall well pad flow at any t^K is defined using equation (16).

$$F(t^K) = \sum_{i=1}^{N} F_i\,(t^K)\delta_i(t^K) \quad (16)$$

$F(t^K)$ can be shown to be a random variable that follows a distribution specific to each well pad. In general, distribution can be given using equation (17).

$$F \sim D(\mu, \sigma^2), \quad (17)$$

where μ is the mean (average) flow (determined using equation 18), while σ^2 (determined using equation 19) is the flow variance. It is this variance that the scheduling algorithm minimizes.

$$\mu = \frac{\sum_{k=1}^{M}\left(\sum_{i=1}^{N} F_i(t^K)\delta_i(t^K)\right)}{M} \quad (18)$$

$$\sigma^2 = \frac{\sum_{k=1}^{M}\left(\sum_{i=1}^{N} F_i(t^K)\delta_i(t^K) - \frac{\sum_{k=1}^{M}\left(\sum_{i=1}^{N} F_i(t^K)\delta_i(t^K)\right)}{M}\right)^2}{M} \quad (19)$$

The objective is then to find t_i and δ_i such that $\sigma^2 \rightarrow min$ holds true and the constraints $t_i \geq 0$ and $t_i + \tau_i \leq 60$ are satisfied.

The scheduling solution is numerically derived using a genetic algorithm, which requires two components: solution space coding and fitness function identification. In the case of ESP scheduling, the solution space is the collection of N-dimensional vectors $\{t_1,\ldots,t_N\}$ that satisfy the above constraints, and the fitness function is the objective function provided in equation (19). For ease of notation, let $W = \sigma^2$ be the fitness function. The algorithm proceeds as follows:

1. Define the parameter $Q > 0$ as an integer.
2. Choose a random sample of size Q from the solution space, and call each element in the sample S_j, where $j = 1,\ldots,Q$.
3. Evaluate the fitness function for each S_j with $W_j = W(S_j)$.
4. Select the "parent" solutions for "breeding" using the fitness proportionate selection method, which means that the probability of a parent being selected is given by $\rho_s = \frac{W_s}{\sum_{j=1}^{Q} W_j}$. Then a random selection is made similar to how a roulette wheel is rotated.
    a. Calculate the cumulative probability distribution (CDF) over the list of individuals in the sample using a probability proportional to the fitness of the individual.
    b. Select a uniform random number in [0,1].
    c. The inverse of CDF for the selected random number is the selected individual.
5. Execute crossover and mutation operations on the parent solutions. Crossover is achieved by taking two (or more) parents and recombining their elements (start times), which results in two (or more) children solutions. Mutation can be defined by randomly changing a single start time (single point mutation) or rearranging start times (swapping, inversion, or scramble). Visually, these operators are described in Figure 5.

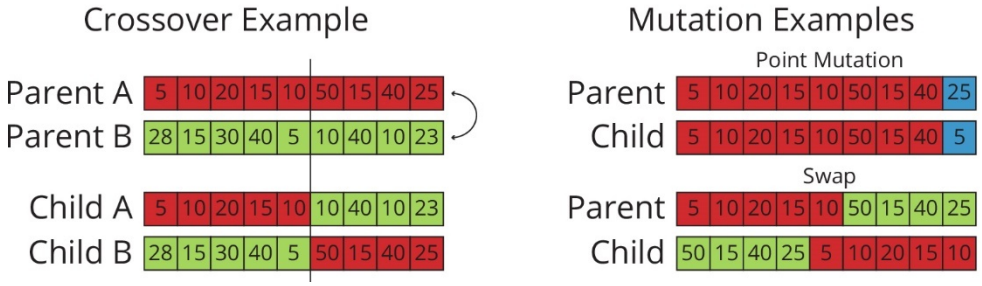

Figure 5. Crossover and mutation operations

6. Evaluate the resulting population for fitness, $minW_j$, and the process iterates until one of the following four criteria is satisfied:
    a. A solution is found that satisfies the objective function.
    b. New generation does not produce a better fit (within parameter $\epsilon > 0$).
    c. Maximum number of population generation iterations is exceeded (within parameter $\gamma > 0$).
    d. Allocated resources are exceeded.

The result is the optimal start time for all pumps; this algorithm is recalculated every cycle (or every hour) to constantly optimize ESP scheduling.

Business Case

As a business case, a site with 1800+ producing wells under ESP control will be used to demonstrate the approach and techniques to optimize ESP operation using real-time simulation. In the overall oil and gas production subsystem, the scope for ESP optimization using real-time simulation covers the production wells, ESPs, valves, and pipeline, as shown in Figure 6. These components plus regulatory control are all included in the dynamic models. This solution approach is extensible and may be used to model fields of all sizes as well as handle field expansions, as demonstrated in Figure 7.

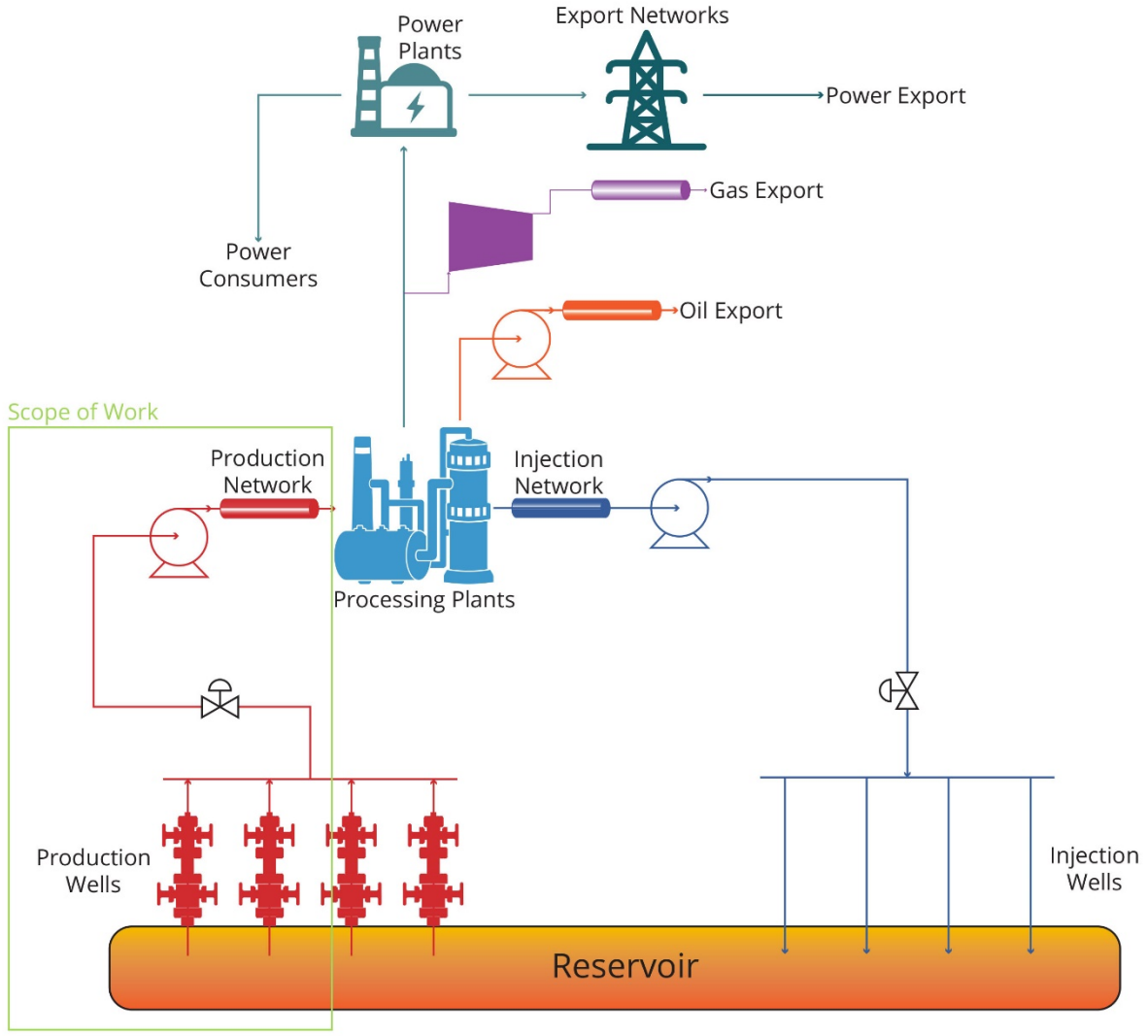

Figure 6. Scope of work

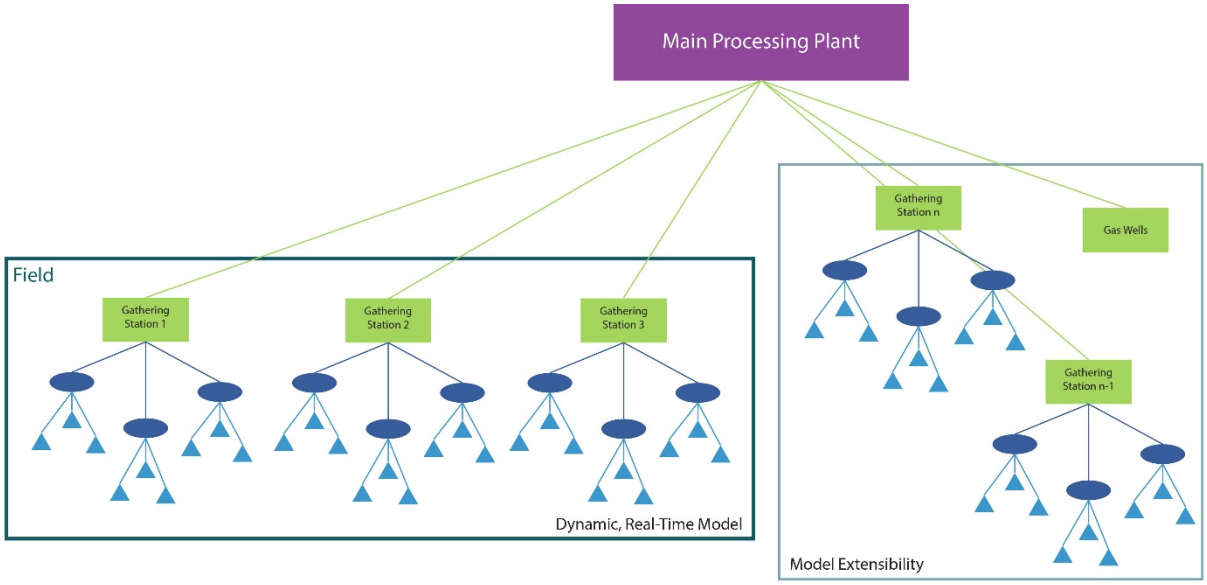

Figure 7. Model extensibility

After creating the baseline models and letting the system learn and auto-tune, the following steps are taken to optimize the ESP control:

1. Install Pump Unit Diagnostics to reduce widespread pressure oscillations for each gathering station.
2. Apply Dynamic, Real-Time Optimization to control individual wells and gathering station pressure set points based on increasing production to minimize system disturbances.
3. When a disturbance occurs, regulatory control and mass balance reconciliation reduce the oscillation time.
4. Increase production by 10% (this has been confirmed in a small-scale pilot to prove the algorithms and advanced technology).
5. Increase liquid extraction and dropout and electrical consumption by 1+%.

Reducing widespread pressure oscillations leads to a more stable network with fewer fluctuations, less electricity use, and more oil and gas throughput. Figure 8 shows an example of pressure oscillations without this solutions (blue lines) and with the solution applied (green lines).

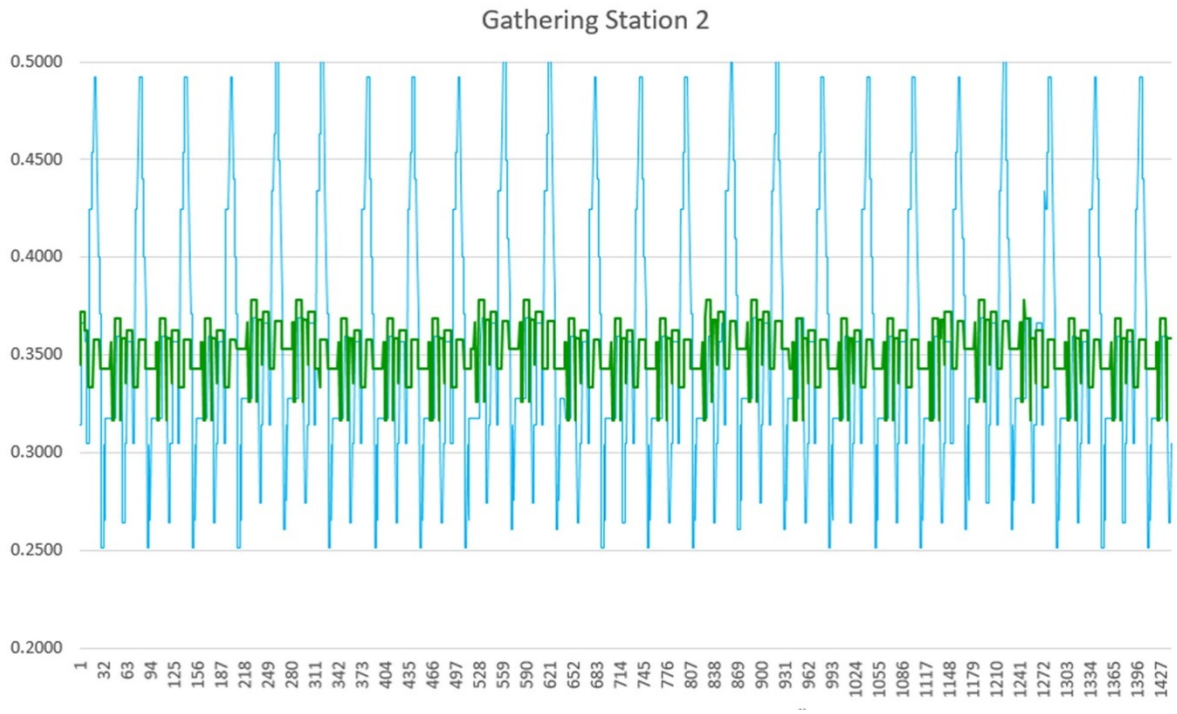

Figure 8. Pressure oscillations without (blue) and with (green) solution

When applied over time, seasonal swings, large fluctuations, and major events that cause long-terms oscillations (thereby reducing throughput) are also minimized.

Conclusions

Efficient ESPs lead to lower operation and maintenance costs, extended ESP life, and improved ESP availability and reliability. The method and techniques of real-time modeling, auto-tuning, and simulation may be used to track basic ESP attributes, optimize operation, and allow for a more holistic view of the system. One of the greatest areas for optimization is around pressure oscillations. Reducing widespread pressure oscillations leads to a more stable network with fewer fluctuations, less electricity use, and more oil and gas throughput.

Bibliography

1. Electrical submersible pumps. (n.d.). Retrieved May 18, 2016, from http://petrowiki.org/Electrical_submersible_pumps. Society of Petroleum Engineers
2. Predictive analytics. (n.d.). Retrieved May 18, 2016, from https://en.wikipedia.org/wiki/Predictive_analytics

Brief CV

Name: Lawrence Youngblood

Education: B.Sc. Electrical & Electronics Engineering, Oregon State University, 1991; Registered Professional Engineer (PE), State of California

Companies worked for during career: Chevron

Current Position: Vice President of Corporate Development, Statistics & Control, Inc.