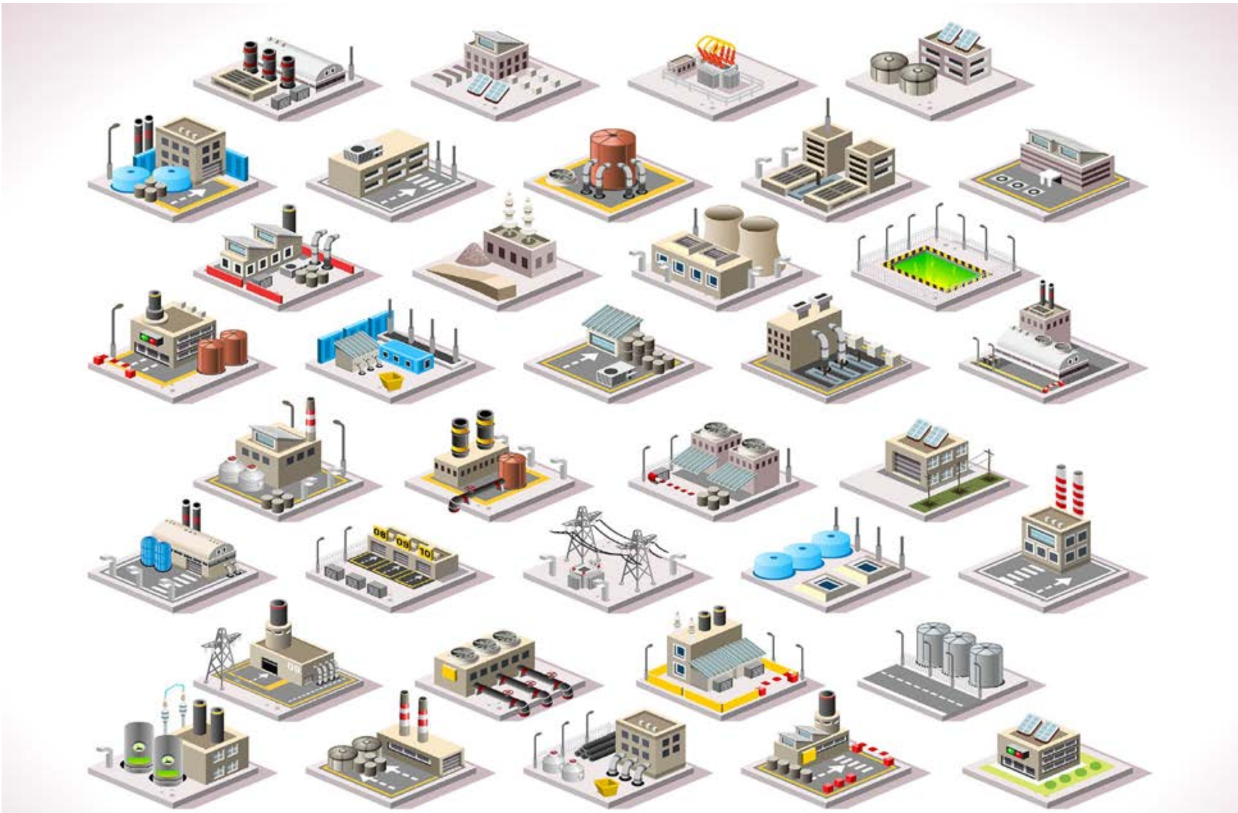

White Paper

OptiRamp® Modeling

Predictive Control Methodology

Vadim Shapiro
Dmitriy Khots, Ph.D.

Table of Contents

Introduction

The *OptiRamp* Modeling Submodule is a comprehensive suite of cross-industry standard and custom data analysis and modeling techniques that serve as a key component of the Advanced Dispatch Control System (ADCS) developed by Statistics & Control, Inc. (S&C). The Modeling Submodule consists of three parts: data processing, model construction, and self learning. Data processing incorporates data import, storage, and noise reduction methods, while the model construction portion uses a wide array of statistical and mathematical modeling techniques to predict target outcomes. Self-learning combines model parameter tuning and monitoring techniques to ensure continuous quality performance of each process model. Figure 1 illustrates interactions between the Modeling Submodule parts.

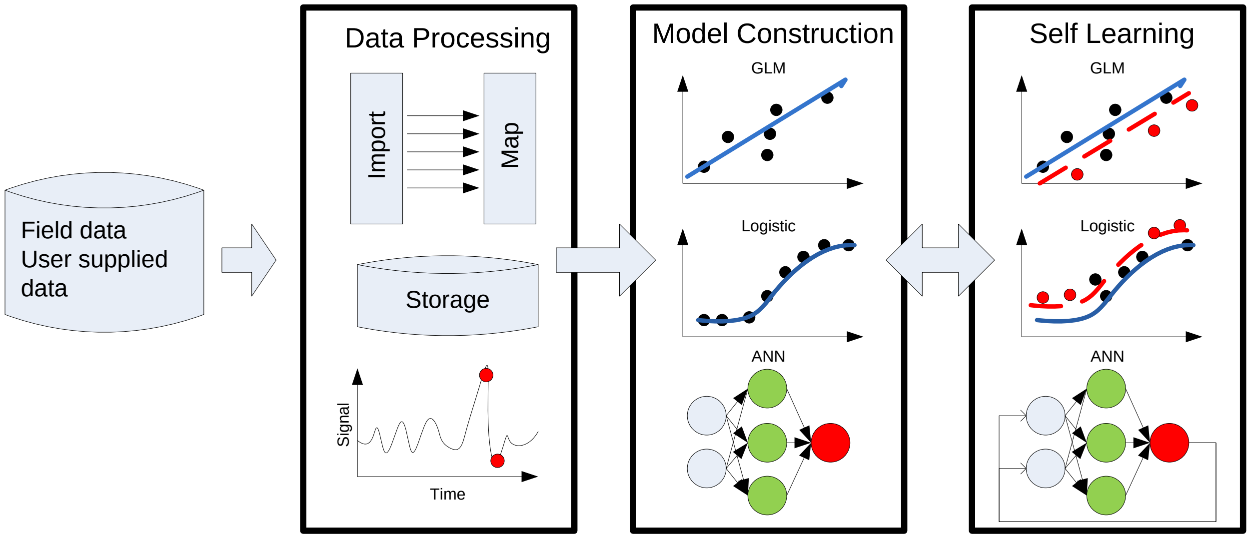

Figure 1. *OptiRamp* modeling submodule schema

The Modeling Submodule is designed to function at both the conceptual design and production stages of control system implementation. During the conceptual design stage, a process baseline model is created, allowing for dynamic studies and simulations. The production stage fully utilizes static and dynamic models that, depending on process perturbations, can be updated in real time.

Modeling Submodule output consists of parameter estimates for each process model and is used by various *OptiRamp* modules and submodules, including the *OptiRamp* Real-Time Optimization and Scheduling Submodules.

The *OptiRamp* Modeling Submodule provides a user-friendly interface to tune each process model using a limited predefined set of configuration parameters that can be adjusted in the field in real-time mode.

Data Processing

The *OptiRamp* Modeling Submodule is equipped with S&C proprietary tools for data capture, compression, storage, and signal processing. The submodule can capture real-time transmitter data as well as user-supplied data in MS Excel spreadsheets and text files (delimited and fixed length), compress the data using a proprietary archiving tool, and store it on an internal database. The submodule is also equipped with an interface to capture metadata, such as user-supplied baseline model parameters.

Data Import

The Modeling Submodule uses a data import interface with customizable field extensions that can handle thousands of inputs. Data can be imported using standard database connections, such as ODBC, as well as from user-supplied MS Excel spreadsheets or text files. A metadata import interface allows users to input baseline model parameters and to adjust current model parameters. The user-friendly point-and-click mapping interface allows for easy implementation; however, import coding is also accessible to power users.

Data Storage

Data are stored with a state-of-the-art archiving tool that compresses large quantities of data produced by industrial continuous processes. The data can be stored in raw and filtered forms to maximize space capacity.

Noise Reduction

The *OptiRamp* Modeling Submodule employs a wide array of signal processing tools 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. The signal can be transformed (filtered) using the following techniques that are implemented in the submodule.

Linear Transformation

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.

Nonlinear Transformation

Most transformations utilized in the *OptiRamp* Modeling Submodule are nonlinear in nature because they violate one of the linear transformation assumptions. The main nonlinear transformations are described below.

Z-Transform

The Z-transform converts the discrete time signal into a frequency domain representation (a histogram). Given a discrete time signal $x(t)$, the bilateral Z-transform is given by equation (1).

$$X(z) = Z\{x(t)\} = \sum_{t=-\infty}^{\infty} x(t) z^{-t} \tag{1}$$

The unilateral Z-transform is shown in equation (2)

$$X(z) = Z\{x(t)\} = \sum_{t=0}^{\infty} x(t) z^{-t}, \tag{2}$$

where z is a complex number.

Discrete-Time Fourier Transform

The discrete-time Fourier transform (DTFT) is a special case of the Z-transform when $z = e^{i\omega}$. Equation (3) shows the DTFT.

$$X(\omega) = \sum_{t=-\infty}^{\infty} x(t) e^{-i\omega t} \tag{3}$$

In most cases, $x(t)$ values are the sampled values of the process variable over a parametrically defined time frame, *T*. In this case, the sampling rate is $f = \frac{1}{T}$. Thus, DTFT is an approximation to the continuous time Fourier transform.

Kalman Filter

The Kalman filter is a recursive estimator of the current process variable value or of an entire process state consisting of multiple variable values. This filter uses the previous estimated value and the current measurement to produce the filtered value. The Kalman filter signal model is provided by the linear stochastic difference equation shown in equation (4)

$$x_k = F_k x_{k-1} + B_k u_k + w_k, \tag{4}$$

where x_k is the current process variable value or state, x_{k-1} is the previous value or state, F_k is the value (state) transition model, B_k is the control input model applied to a vector consisting of control variables u_k, and w_k is the multivariate process noise with zero mean and covariance Q_k. The measurement at each time increment is assumed to have white noise calculated by equation (5)

$$z_k = H_k x_k + v_k, \tag{5}$$

where H_k is the observation model and v_k is the white noise with zero mean and covariance R_k.

The Kalman filter uses a two-stage process to create the state estimate: predict and update. The predict stage is calculated using equations (6) and (7).

$$\hat{x}_k = F_k x_{k-1} + B_k u_k \text{ (predicted estimate)} \tag{6}$$

$$P_k = F_k P_{k-1} F_k^T + Q_k \text{ (predicted estimate covariance)} \tag{7}$$

The update stage is calculated by equations (8) through (12).

$$\hat{y}_k = z_k - H_k \hat{x}_k \text{ (residual measurement)} \tag{8}$$

$$S_k = H_k P_k H_k^T + R_k \text{ (residual covariance)} \tag{9}$$

$$K_k = P_k H_k^T S_k^{-1} \text{ (optimal Kalman gain)} \tag{10}$$

$$\tilde{x}_k = \hat{x}_k + K_k \hat{y}_k \text{ (updated estimate)} \tag{11}$$

$$P_k = (I - K_k H_k) P_k \text{ (updated estimate covariance)} \tag{12}$$

Regression

The *OptiRamp* Modeling Submodule primarily uses regression to filter time-dependent process variables using the floating window mechanism. The filter uses an ordinary least squares (OLS) regression, which is described in more detail in the "Modeling Methods" section.

Model Construction

The *OptiRamp* Modeling Submodule software allows S&C to model continuous processes of various complexity. Once the target variable is identified, users can select the necessary modeling technique and the option for automated algorithm selection. Algorithms implemented in the submodule are described.

During the conceptual design stage, the submodule creates a baseline process model that is used to study the dynamic behavior of the process by running simulations according to the identified relationship between process variables in real time. Users can also supply the initial baseline model via the metadata import interface.

Proprietary computational algorithms allow the submodule to possess a large operational range to reliably replicate the real process in all operational situations of interest. The submodule can simultaneously create both static (steady state) and dynamic (transitional state) models from real-time data for all processes because of its capability to run on a stand-alone basis on multiple separate servers.

Modeling Methods

The *OptiRamp* Modeling Submodule is equipped with the following modeling methods.

Least Squares Regression

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 (13).

$$S = \sum_{i=1}^{n} \left(y_i - P(\beta, x_{ij}) \right)^2 , \tag{13}$$

where $P(\beta, x_{ij})$ is an mth-degree polynomial with k independent variables x_j and coefficients β.

Logistic Regression

Logistic regression is used to predict categorical targets and, in particular, binary events (for example, compressor surge). The model estimates coefficients β for the logit function shown in equation (14):

$$\ln\left(\frac{p_i}{1-p_i}\right) = P(\beta, x_{ij}), \tag{14}$$

where $P(\beta, x_{ij})$ is an mth-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 Network

Artificial neural networks (ANNs) are nonlinear statistical modeling techniques designed to mimic the human brain. The ANN algorithm in the *OptiRamp* Modeling Submodule 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 2 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.

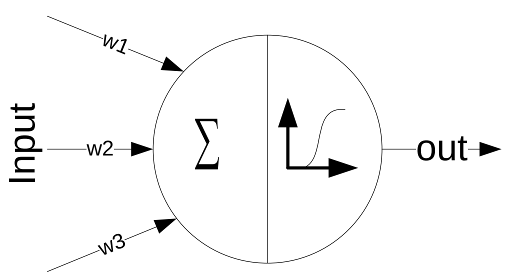

Figure 2. Artificial neuron with three inputs

The MLP weights are initially chosen at random and are then adjusted such that the error function (15) is minimized on the training set:

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - F(x_i)\right), \tag{15}$$

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

Static Model

The *OptiRamp* Modeling Submodule is well suited to model steady state operating conditions by describing dependencies between input and output process variables in a stable environment. The initial static model is built during the conceptual design stage and is then continuously updated using a floating window algorithm. The window size and step are dependent on the model process and can range from seconds to months.

Given the filtered dependent variable y and a vector of filtered independent variables x_i—where $i = 1, ..., N$—the Modeling Submodule builds the initial static model $y = f_{ini}(x_i)$ based on a training set collected in time interval T with a sufficient population in each subset of N-dimensional Euclidean space. The model goodness-of-fit is evaluated using R^2_{adj} for OLS, the Kolmogorov-Smirnov test for logistic regression and for neural networks predicting categorical targets, and average error for neural networks predicting continuous targets.

The model is then iteratively rebuilt such that the goodness-of-fit statistic remains within acceptable parameters whenever the independent variable domain refreshes with new data. The challenger model $y = f_{challenger}(x_i)$ automatically replaces the production model $y = f_{prod}(x_i)$ if its goodness-of-fit measure is better.

The key advantage of the static model is that it allows S&C to describe interactions of steady state main process characteristics, such that the impact of control actions can be predicted with accuracy.

Dynamic Model

The Modeling Submodule 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 $y(t)$ at a time instance t and a vector of filtered independent variables $x_i(t - \tau_{x_i})$—where $i = 1, ..., N$ and τ_{x_i} is a time shift specific to each independent variable x_i—the Modeling Submodule builds the initial dynamic model $y(t) = f_{ini}(x_i(t - \tau_{x_i}))$ 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). Similar to the static model construction, challenger models are iteratively rebuilt and tested against the current model until the new model predictive power is greater.

The submodule uses the same modeling techniques used with the static model. Additionally, the submodule is equipped with autoregressive models that are used by the *OptiRamp* Scheduling Submodule for forecasting purposes. For example, the linear version of the *ARMAX(p, q, b)* model can be defined via equation (16):

$$X_t = \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \sum_{i=1}^{q} \eta_i d_{t-i}\,, \tag{16}$$

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.

Self Learning

The *OptiRamp* Modeling Submodule is equipped with a robust system for automated tuning of model parameters based on a predefined model fit. 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 $y = P(\beta, x_{ij})$ or $\ln\left(\frac{p_i}{1-p_i}\right) = P(\beta, x_{ij})$, where P is the m^{th}-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 (described in the "Static Model" section above) 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, the submodule selects between one and ten hidden layers, randomly varying the number of neurons between one and ten in each layer until the error function is minimized. Thus, the Modeling Submodule allows for automated model parameter tuning, e.g., the degree in polynomial regression is chosen to minimize the selected error function, such as sum of squared errors in OLS regression or $-2\ln(\Lambda)$ 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. The Modeling Submodule utilizes Statistical

Process Control 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 $i = 1, \ldots, n$. Whenever the filtered signal values consistently fall outside $\left[\overline{X} - 3s, \overline{X} + 3s\right]$, which is indicative of a process change, both dynamic and static models are retrained.

Self learning and parameter tuning is crucial for ensuring optimal performance under severe process changes and allows the ADCS to make reliable automated control decisions.

About Statistics & Control, Inc.

S&C—an engineering consulting and technology company headquartered in West Des Moines, IA—solves complex challenges for customers through its unique technology and its highly seasoned team of professionals. The company has a global portfolio spanning the energy, oil and gas, utility, and digital oil field industry sectors. S&C provides clients with turbomachinery control solutions that easily integrate with the existing system as well as *OptiRamp*® solutions, which focus on process and power analytics to optimize processes and, in turn, reduce costs and increase reliability. S&C also provides consulting, dynamic system studies, modeling, automation, training and OTS, and support services.

Statistics & Control, Inc.
4401 Westown Pkwy, Suite 124
West Des Moines, IA 50266 USA
Phone: 1.515.267.8700
Fax: 1.515.267.8701