> Itron white paper

Forecasting / Load Research

A Unified Framework for Near-term and Short-term System Load Forecasting

Dr. Frank A. Monforte
Director, Forecasting Solutions

Itron

A Unified Framework for Near-term and Short-Term System Load Forecasting[1]

[1] The framework presented in this paper is the result of research efforts for the New York ISO and the Midwest ISO. Arthur Maniaci of the New York ISO is credited with the original and key insight to introduce ramp rate models into the framework. Research conducted with Sen Li, Liying Wang, and Kala Seidu of the Midwest ISO led to the application of the Kalman Filter for data filtering. Dave Fabiszak, Christine Fordham, Stuart McMenamin, John Pritchard, Mark Quan, and Richard Simons of Itron contributed valuable insights toward the final framework. The modeling framework presented in this paper is available in Itron's Automated Forecasting System, MetrixIDR, Version 4.2. This paper presents the opinions of the author and does not necessarily reflect the views of the New York ISO or the Midwest ISO.

Introduction

There is a rich body of literature on the topic of short-term (next-day to ten days ahead) hourly load forecasting that system operators rely on to set the day-ahead generation schedule for meeting tomorrow's loads. Most of the short-term literature focuses on the use of neural networks to produce day-ahead load forecasts[2]. The premise underlying this literature is that neural network techniques are well suited to the problem of modeling the nonlinear relationship between system loads and weather.

While the short-term forecasting modeling problem is well documented, there is limited literature that addresses the need for precise five-minute ahead to one hour-ahead, real time load forecasts at the five-minute level of load resolution. During this *golden hour*, system operators make the critical operating decisions as to which generation units will be dispatched to meet system demand. While advanced statistical modeling represents *industry best practices* for the short-term forecasting problem, it is not atypical for a system operator to rely upon the similar-day based forecasting algorithm that came installed with his energy management system. In many cases, the system operator manually shapes the selected similar day to craft the load forecast against which he will operate.

At least in part, the current state of near-term and short-term load forecasting reflects the business processes that are in place for most system operators. It is not uncommon for the next-day load forecast to be set independently of the near-term forecasts. In fact, these forecasts are often developed by different groups within the same organization. The published next-day forecast sets the generation schedule that feeds the real-time operations desk. As the day unfolds, system operators make adjustments to the generation schedule that was set during the previous day to account for projected near-term system imbalances. In most cases, these real-time adjustments do not feed forward to help set the next-day forecast. Further, the day-ahead forecast problem tends to be at the 15-minute (or higher) level of load resolution, while the near-term forecast problem is at the five-minute level of detail. In sum, many system operators are left with a mixed bag of forecasting techniques and no means of producing a single operational forecast that runs from five-minutes ahead to several days ahead.

This paper presents a forecasting framework to produce five-minute level load forecasts for forecast horizons from five-minutes ahead to several days ahead. The framework addresses two key challenges to system load forecasting: (1) data filtering to smooth through the most recent SCADA (Supervisory Control and Data Acquisition) reads to prevent load spikes from propagating through the forecast horizon, and (2) developing accurate near-term and short-term forecasts, given forecasts of calendar, solar, economic and weather conditions. A framework for data filtering is presented in Section 2. This is followed with an overview of a unified near-term and short-term forecasting framework in Section 3.

Data Filtering

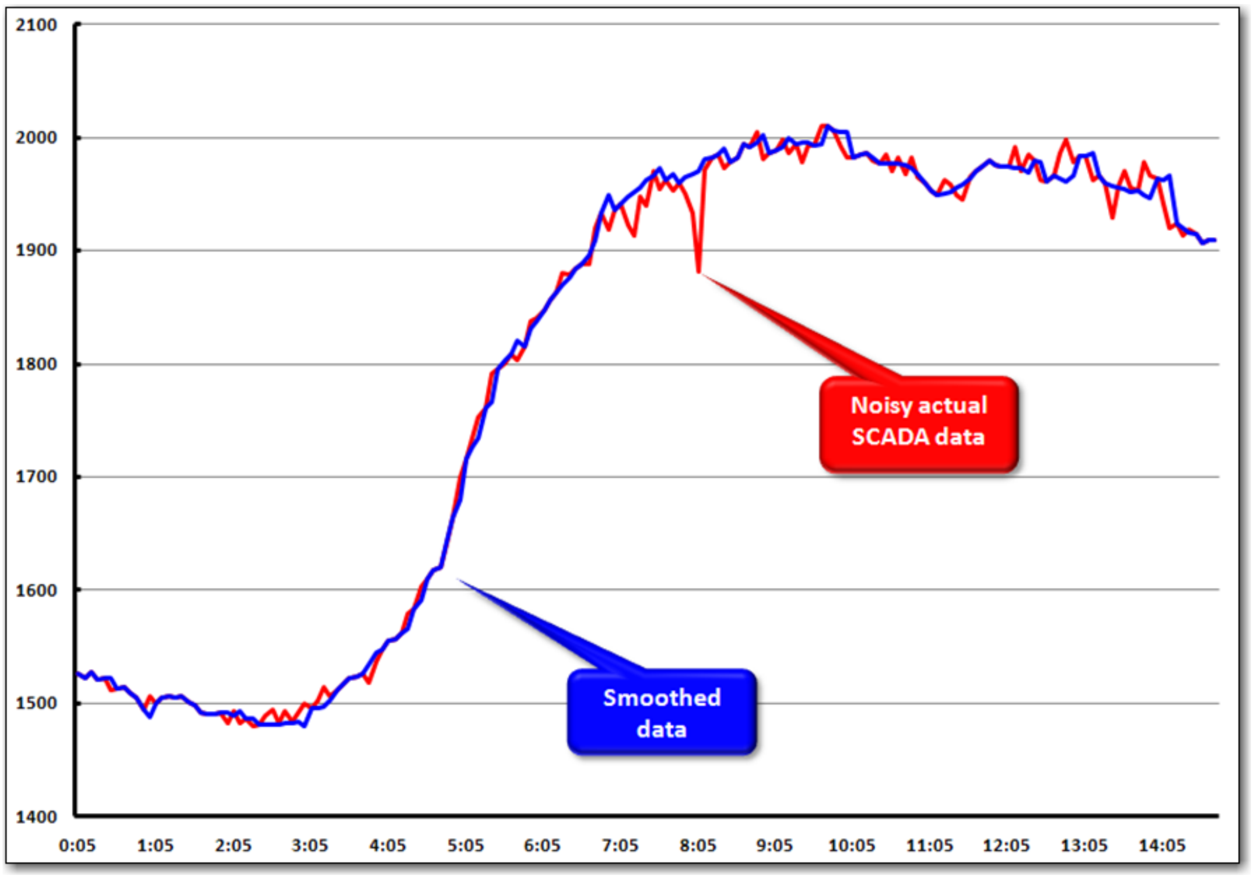
To dispatch generation efficiently to meet system loads, the system operator needs an accurate five-minute level load forecast for a rolling one to six hour window. Ideally, this forecast would be updated every five minutes to reflect the most recent state of the system, as measured by the SCADA system. The first of the near-term forecasting challenges is noisy SCADA data. An example of noisy versus smoothed SCADA data is presented in Figure 1, in which the red line represents the noisy load data while the blue line represents the corresponding smoothed data series.

From a forecasting perspective, the ideal situation is to have a *smooth* stream of five-minute data values feeding into the forecast model. This is especially true with autoregressive forecasting models that will echo into the forecast period any spikes or noise in the SCADA reads. Feeding a sequence of *smooth* five-minute SCADA reads through an autoregressive forecasting model will lead to a *smooth* sequence of load forecasts. Conversely, noisy five-minute SCADA reads will lead to a noisy sequence of load forecasts.

[2] See *Short-Term Energy Forecasting with Neural Networks* for an overview of short-term energy forecasting using regression and neural network models.

McMenamin, S., and F. Monforte. *Short-Term Energy Forecasting with Neural Networks.* The Energy Journal, International Association for Energy Economics. Volume 19, Number 4 (1998).

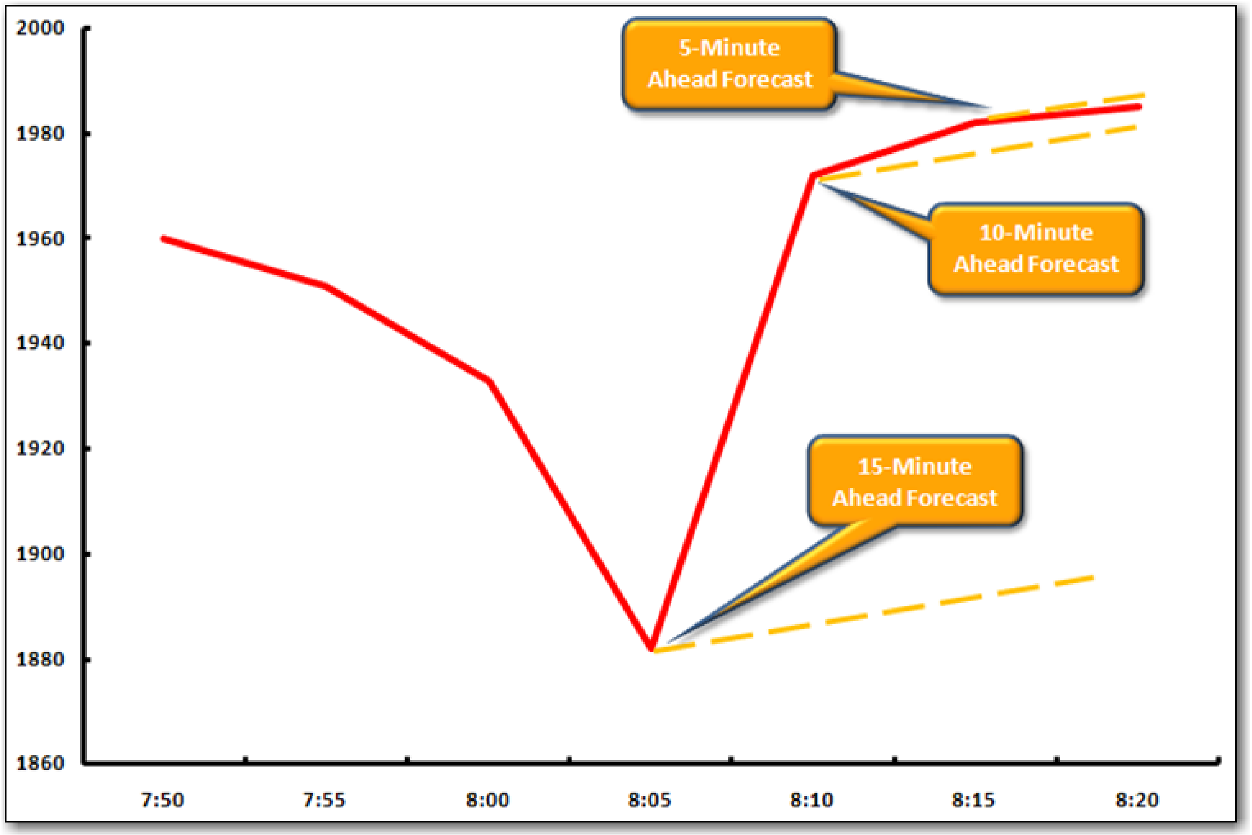
Figure 1: Data Quality Issues – Example of Noisy versus Smooth Data



To help fix ideas, compare the 15-minute ahead forecast for 8:20 a.m. that was generated using SCADA data through 8:05 a.m. to both the 10-minute ahead forecast for 8:20 a.m. that was generated using SCADA data through 8:10 a.m. and to the five-minute ahead forecast for 8:20 a.m. that was generated using SCADA data through 8:15 a.m. In this case, the sequence of load forecasts for 8:20 a.m. would be deemed *smooth* if the alternative forecasts for 8:20 a.m. are not significantly different from each other.

Consider the noisy data in Figure 2, which are derived from load data for 8:05 a.m. through 8:20 a.m. presented in Figure 1. An autoregressive model that launches off the 8:05 a.m. data leads to a 15-minute ahead load forecast for 8:20 a.m. that is significantly below the actual value. The same autoregressive model that launches off the 8:10 a.m. data point jumps up, reflecting the increase in loads between 8:05 a.m. and 8:10 a.m. The relatively smooth transition in actual loads from 8:10 a.m. to 8:15 a.m. leads to a relatively stable 10-minute ahead and five-minute ahead forecast for loads at 8:20 a.m. Because of the large variation in the forecasts for 8:20 a.m., many system operators will conclude that the sequence of 15-minute ahead load forecasts cannot be relied upon for generation dispatching.

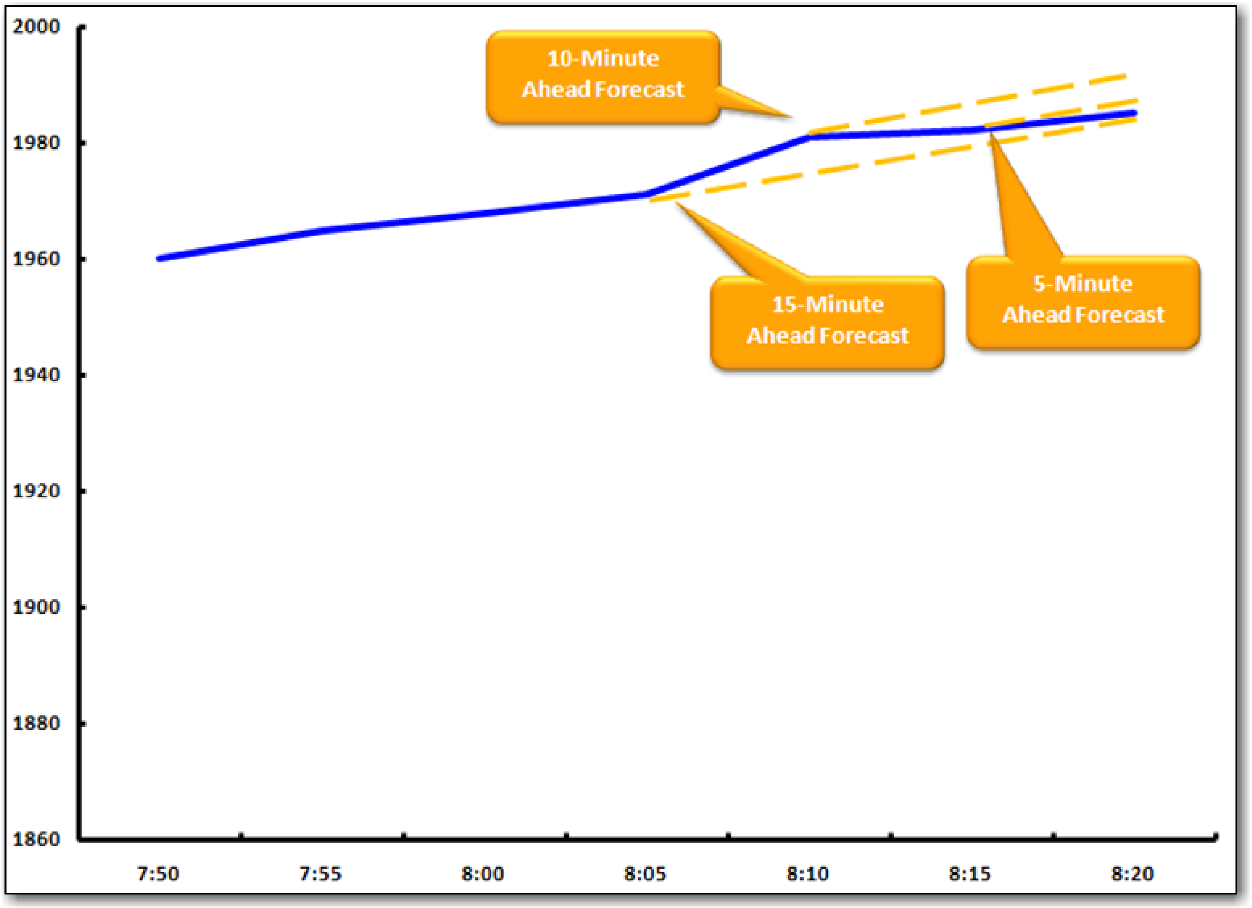
Figure 2: Example of Forecasting Using Noisy Data



In contrast, the sequence of 8:20 a.m. forecasts that results from an autoregressive model launching off the smooth data series presented in Figure 3 are approximately the same. The smooth sequence of forecasts gives credibility to the sequence of 15-minute ahead forecasts. This in turns means that they can be relied upon for generation dispatching.

This example illustrates the necessity for a means of smoothing SCADA reads. Ideally, the technique will keep those data points that represent systematic changes in the state of system demand, while filtering out those data points that represent random fluctuations. The filtered data series should lead to a relatively smooth sequence of load forecasts.

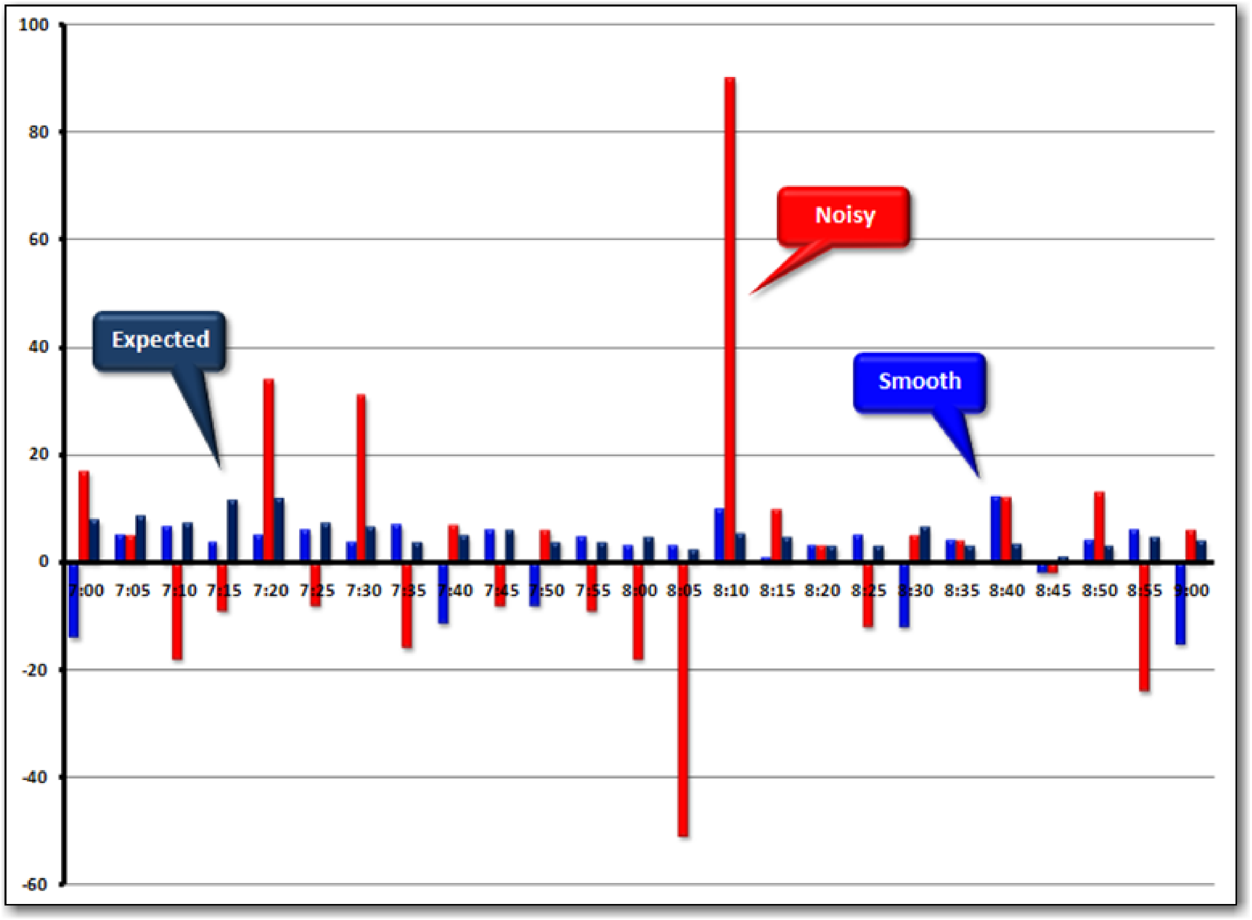
Figure 3: Example of Forecasting Using Smooth Data



When we look at the graph of five-minute loads presented in Figure 1, it is easy to identify systematic changes in loads, as opposed to random fluctuations. To quantify what the eye sees, it helps to view the load data as a sequence of five-minute deltas or ramp rates (i.e., the change in loads from one five-minute period to the next). Taken in isolation we can ask, “Is the observed ramp rate reasonable for this period?” If the observed ramp rate is close to the expected value, we can feel reasonably confident that the observed load value represents systematic variation rather than random fluctuation. Conversely, if the observed ramp rate is significantly different than what we expect, we will want to ignore the most recent observed load and instead launch off our best estimate of what the load should be given the state of the system leading up to this point.

Figure 4 presents the ramp rates corresponding to the load data presented above. Here, the ramp rates computed from the smooth data series are represented by the blue bars. The ramp rates computed from the noisy data are represented by the red bars. The black bars represent the average or expected ramp rates for these data. Based on visual inspection, the smooth ramp rates track relatively close to the expected ramp rates, while the noisy data are at times significantly different from what is expected. For purposes of forecasting, the smooth data are a better indicator than the noisy data of the systematic changes in the system loads. This implies that the smooth data series should be used for forecasting rather than the observed SCADA reads.

Figure 4: Observed Versus Expected Ramp Rates



Kalman Filter

For near-term forecasting, it is not uncommon to see a Kalman Filter being used to smooth the incoming SCADA reads[3]. The Kalman Filter is a recursive algorithm that estimates the state of a system in the presence of measurement error. The Kalman Filter is applicable to system load forecasting since the state of the system (i.e., system load) is measured imprecisely using SCADA metering. Estimates of the current state of the system are fast because only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state. This is different than other time series approaches that leverage the full length of historical data to develop an estimate of the current state. The Kalman Filter has computational appeal because it only needs the most recent observation to form the data smoothing once the filter has been primed with exogenous data.

The Kalman Filter utilizes two sets of equations: Update equations and Measurement equations. These two sets of equations are presented below.

[3] The Kalman Filter was introduced in *A New Approach to Linear Filtering and Prediction Problems* by R.E. Kalman, Journal of Basic Engineering 82 (1): 35-35 (1960). A tutorial on Kalman Filtering is provided in *An Introduction to the Kalman Filter* by G. Welch and G. Bishop, TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill (2006).

Update Equations

$$\hat{L}_t^{\sim} = A_t \hat{L}_{t-1} + e_t$$

where

$\hat{L}_t^{\sim}$ is an *a priori* estimate of the load in period (t)

$\hat{L}_{t-1}$ is an ex post estimate of the load from the prior period (t-1)

e_t is the process error that measures the variability of loads in period (t)

The Update equation provides an *a priori* estimate of the current system load. This estimate is a linear combination of the *ex post* system load estimate from the prior period and a measure of process error. Process error can be thought of as the volatility of the load at a given time period. The process error will be different for different periods of the day. For example, there will be large variations in loads during the afternoon hours when customer loads are driven by variations in weather and operating conditions. In contrast, loads during the early morning hours tend not to vary as much because the impact of weather and operating conditions are mitigated by the fact that most businesses are closed during those hours. The parameter vector (A), which defines the relationship between prior and current period loads, is allowed to vary through the day. This is needed since during the ramp up in loads in the morning hours the relationship will be positive and greater than 1.0, while the ramp down in the late evening hours will call for a value less than 1.0.

Measurement Equations

$$L_t^O = B_t L_t + u_t$$

where

L_t is the true system load at time (t) that is not observed directly

$\mathrm{L}_\mathrm{t}^\mathrm{O}$ is the observed system load at time (t)

u_t is the measurement error at time (t)

The Measurement equation states that the measured load is equal to the true load plus a random measurement error. We know this is the case with SCADA reads where dropped pulses and other reading errors give imprecise measures of the true state of the system.

The Update and Measurement equations are two independently generated estimates of the true load at time (t). The Kalman Filter suggests a means for constructing a weighted average of these two estimates to derive a final estimate of the system load at time (t). The weighting scheme can be written as follows:

$$\hat{L}_t = \hat{L}_t^{\sim} + K_t\left(L_t^O - \hat{L}_t^{\sim}\right)$$

The Kalman Filter equation states that our best predictor of the current system load is a weighted average of our *a priori* estimate from the Update equation and the *innovation* or *residual* between our *a priori* estimate and the observed load. The weight is referred to as the Kalman Gain (K). Kalman derived the optimal means of computing the Kalman Gain as follows:

$$K_t = \frac{\sigma_{e,t}^2}{\sigma_{e,t}^2 + \sigma_{u,t}^2}$$

Here, the Kalman Gain is a function of the variance of the process noise from the Update equation and the variance of the measurement error from the Measurement equation. The greater the process noise variance relative to the measurement error variance, the greater the weight that is placed on the observed system load. Conversely, the greater the measurement error variance relative to the process noise variance the greater is the weight placed on the *a priori* estimate. This weighting scheme has intuitive appeal in that the noisier the load, the less weight is placed on the most recent observations. In this case, the Kalman Filter will *smooth* through the data.

Implementation Challenges

To implement the Kalman Filter as described above, two sets of parameters need to be fed into the algorithm as exogenous factors. The first set is the parameter vector (A) of the Update equation that relates prior period loads to current loads. One option is to use the historical average of the ratio of Load at time (t) to the Load at time (t-1) to provide an estimate of the vector (A).

The second parameter set that needs estimating is the period-specific Update equation error variances ($\sigma^2_{e,t}$). These can be computed from the historical data as follows:

$$\sigma^2_{e,t} = \frac{\sum_{n=1}^{N}(\bar{L}^t - L^t_n)^2}{N}$$

Here, we use the historical variance of the load at each time period (t) as a proxy of the Update equation error variance.

To compute the measurement error variance for a given observation, we use:

$$\sigma^2_{u,t} = (\bar{L}^t - L^O_t)^2$$

Because the proxy of the Update equation error variance requires looping over all the data in the historical data set, it is best to compute these values externally and pass them in as a vector of exogenous data. The measurement error variance, however, can be computed each period given an exogenously supplied vector of expected loads by period.

Two-Stage Modified Kalman Filter

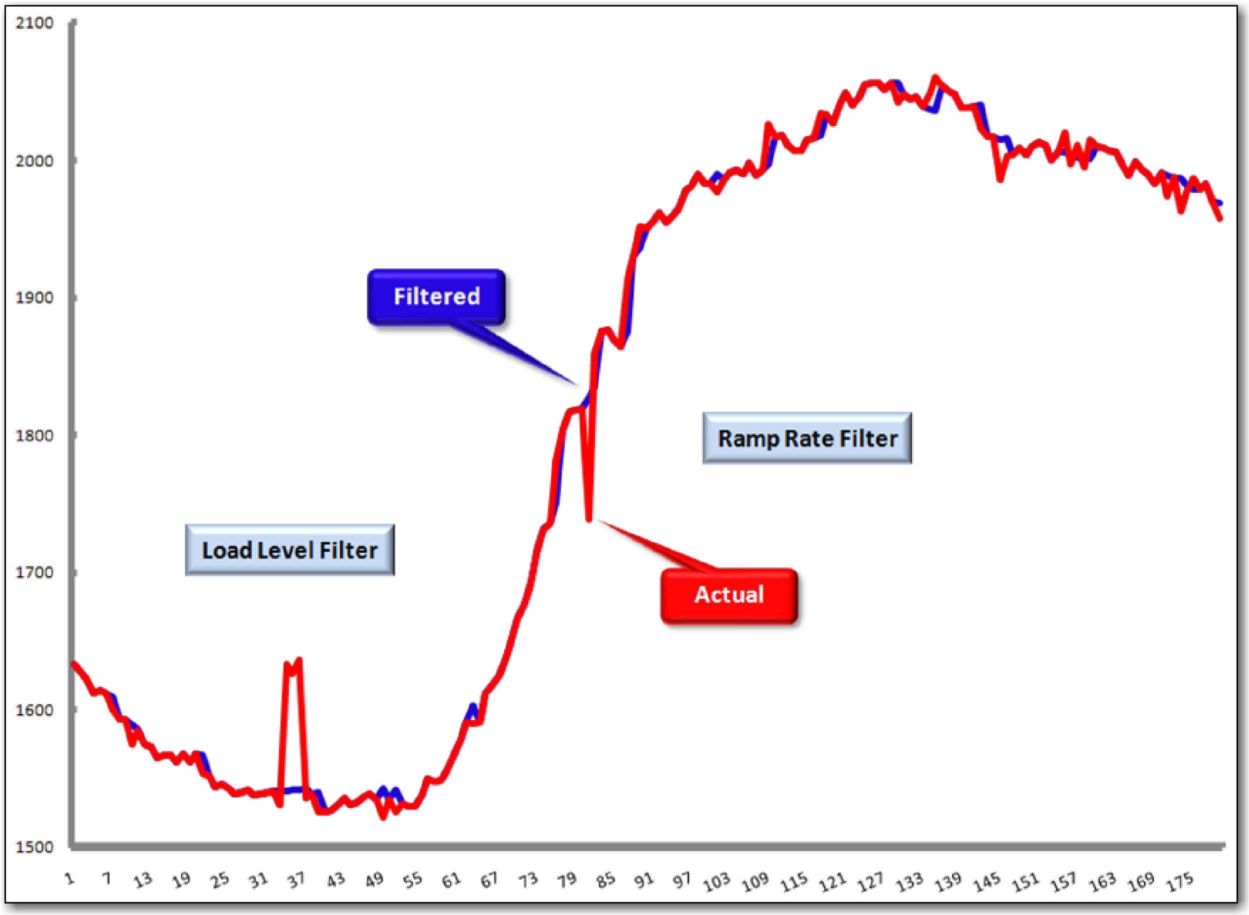
As presented above, the Kalman Filter is a data estimation framework. That is, the observed load is replaced with an estimated value in each period, regardless of whether the observed data point is an outlier or not. The estimated value is then used by the near-term and possibly short-term forecasting frameworks. For accurate system load forecasting, however, we do not want to lose the information that is contained in the actual loads, except in those cases where the observed load is an outlier. We need a data filtering algorithm that passes the following to the forecast models: observed loads that lie within expected ranges and estimated loads when observed loads fall outside expected ranges. In other words, we want to use the observed loads when the data are *good* and estimated loads when they are not.

This suggests a two-stage algorithm is needed to meet the requirements of load forecasting. The first stage is used to identify data outliers like the 8:05 a.m. down spike shown in Figure 1. The second stage is then used to replace the outliers with an estimated or filtered data value. An example of a two-stage filter is presented below.

Stage 1: Outlier Detection

Visual inspection of one day of five-minute loads makes it easy to detect outliers in the data. Two separate data filters can be used to capture the occurrence. The first filter determines whether or not the five-minute change in loads is within the bounds of expectation. The Ramp Rate Filter is based on comparing observed ramp rates to expected ramp rates. The second filter determines whether or not the load value is at an acceptable level. The Load Level Filter is based on comparing observed loads to expected loads. While the Ramp Rate Filter has power in detecting spikes, the Load Level Filter has power in detecting load shifts. Figure 5 provides an example of the type of data anomalies captured by these two filters. An observed load that satisfies both the Ramp Rate Filter and the Load Level Filter is accepted as a valid data point. The exact calculations are presented below.

Figure 5: Examples of Ramp Rate and Load Level Data Filtering



Ramp Rate Filter

The Ramp Rate Filter compares the observed ramp rate to an expected ramp rate. If the observed ramp rate is significantly different from the expected ramp rate, there is an indication that the current load value is a spike (either upward or downward). This implies the following binary test.

$$\mathrm{RR\_Filter}_t = \left(\mathrm{R}_t^{\mathrm{O}} \geq \mathrm{LowerBound}_t^{\mathrm{R}}\right) \times \left(\mathrm{R}_t^{\mathrm{O}} \leq \mathrm{UpperBound}_t^{\mathrm{R}}\right)$$

where

$$\mathrm{R}_t^{\mathrm{O}} = \mathrm{Load}_t^{\mathrm{O}} - \mathrm{Load}_{t-1}^{\mathrm{O}}$$

$$\mathrm{LowerBound}_t^{\mathrm{R}} = \widehat{\mathrm{R}}_t - \left(\mathrm{RR\_Tolerance} \times \widehat{\sigma}_t^{\mathrm{R}}\right)$$

$$\mathrm{UpperBound}_t^{\mathrm{R}} = \widehat{\mathrm{R}}_t + \left(\mathrm{RR\_Tolerance} \times \widehat{\sigma}_t^{\mathrm{R}}\right)$$

Here, the Ramp Rate Filter returns a binary variable that has a value of 1.0 if the observed ramp rate in time period (t) is between the expected bounds (inclusive of the boundaries) for the ramp rate and 0.0 otherwise. The bounds are computed as the expected ramp rate $\left(\widehat{\mathrm{R}}_t\right)$ +/- a ramp rate tolerance times the standard deviation around the expected ramp rate $(\widehat{\sigma}_t^{\mathrm{R}})$. The value for the expected ramp rate and its standard deviation are provided exogenously to the test. In practice, these values are derived from the historical data using models of both the ramp rates and the ramp rate variances. The value of the ramp rate tolerance determines the precision of the test. The tighter the bounds, the more likely any given observation will be replaced with an expected value. If the expected values represent smooth values from a model, the ramp rate tolerance can then be thought of as a *smoothing* parameter.

Load Level Filter

The Load Level Filter compares the observed load to an expected load. It is intended to capture load shifts. If the observed load is significantly different from the expected load there is an indication of a load shift. This implies the following binary test:

$$\text{Load\_Filter}_t = \left(\text{Load}_t^O \geq \text{LowerBound}_t^L\right) \times \left(\text{Load}_t^O \leq \text{UpperBound}_t^L\right)$$

where

$$\text{LowerBound}_t^L = \hat{L}_t - \left(\text{Load\_Tolerance} \times \hat{\sigma}_t^L\right)$$

$$\text{UpperBound}_t^L = \hat{L}_t + \left(\text{Load\_Tolerance} \times \hat{\sigma}_t^L\right)$$

Here, the Load Level Filter returns a binary variable that has a value of 1.0 if the observed load in time period (t) is between the expected bounds for the load and 0.0 otherwise. The bounds are computed as the expected Load $(\hat{L}_t)$ +/- the load tolerance times the standard deviation around the expected load $(\hat{\sigma}_t^L)$. Like the ramp rate tolerance, the load tolerance can be thought of as a smoothing parameter. The tighter the tolerance, the more likely the observed data are replaced with an estimated value.

Stage Two: Modified Kalman Filter

Given the results of the Stage One Ramp Rate and Load Level Filter tests, a modified Kalman Filter can be written as follows.

$$\text{KalmanFilter}_t = \text{RR\_Filter}_t \times \text{Load\_Filter}_t$$

The Kalman Update Equations are then written as:

$$\tilde{R}_t = \hat{R}_t + \left[\text{KalmanFilter}_t \times \left(R_t^O - \hat{R}_t\right)\right]$$

$$\tilde{L}_t = \hat{L}_{t-1}^F + \tilde{R}_t$$

$$\hat{L}_t^F = \tilde{L}_t + \left[\text{KalmanFilter}_t \times \left(L_t^O - \tilde{L}_t\right)\right]$$

where

$\tilde{R}_t$ is the filtered estimate of the ramp rate for time period (t)

$\tilde{L}_t$ is the ex ante estimate of the load value for period (t)

$\hat{L}_t^F$ is the Kalman Filtered load for period (t)

In the case where the current load value satisfies both the Ramp Rate and Load Level Filters, then $\hat{L}_t^F$ will be equal to the observed load. In the case where either the Ramp Rate test, the Load Level test, or both fail, the observed load value is replaced with the ex ante estimate, or $\tilde{L}_t$.

Implementation Challenges

To implement the Modified Kalman Filter, four sets of models are needed: Ramp Rate Model, Ramp Rate Variance Model, Load Level Model, and a Load Variance Model. Each set of models is composed of 288 equations; one equation for each of the five-minute intervals in the day. Since these models are used for data filtering and not load forecasting, it is recommended that relatively simple model specifications be used. For example, a day type model that allows for different weekday versus weekend days by month should be sufficient to filter most data outliers. Additional model complexity that accounts for prevailing weather conditions and observance of Daylight Saving Time will increase the power of the Ramp Rate and Load Level Filters.

A Unified Framework for Near-Term and Short-Term Load Forecasting

System load forecasting is about leveraging where the system has been as measured by the most recent SCADA reads and where the system is going as projected by forecasted calendar, solar, and weather conditions. Knowing where the system has been is very useful for forecasting loads five minutes from now. For example, if it is 8:00 a.m. and the focus is forecasted loads at 8:05 a.m., then knowing actual loads through 8:00 a.m. is a very powerful forecast driver. In contrast, if the focus is the load at 8:05 a.m. seven days from now, then knowing what loads are at 8:00 a.m. today is not as useful as knowing that seven days from now it will be a hot July 4. At some point in the forecast horizon, the explanatory power of the most recent loads becomes less important than knowing future calendar, solar, and weather conditions. The unified near-term and short-term forecasting framework is designed to exploit this concept. It does this by using a combination of three different forecast models. For very near-term forecast horizons, the unified forecast is derived from a set of highly autoregressive model specifications that exploit the most recent SCADA data. For longer forecast horizons, the unified forecast is derived from a set of non-autoregressive models that are driven by calendar, solar, and weather conditions. A model of five-minute ramp rates is then introduced to bridge the gap between the two alternative forecasts.

How far into the forecast horizon should the near-term forecast model be used?

To answer this question, we construct a sequence of highly autoregressive forecast models for selected periods of the day. The sequence of forecast models for loads at time (T) can be expressed as follows:

Five-Minute Ahead Forecast Model: $Load_T = a_0 + \sum_{j=1}^{24} a_j Load_{T-j}$

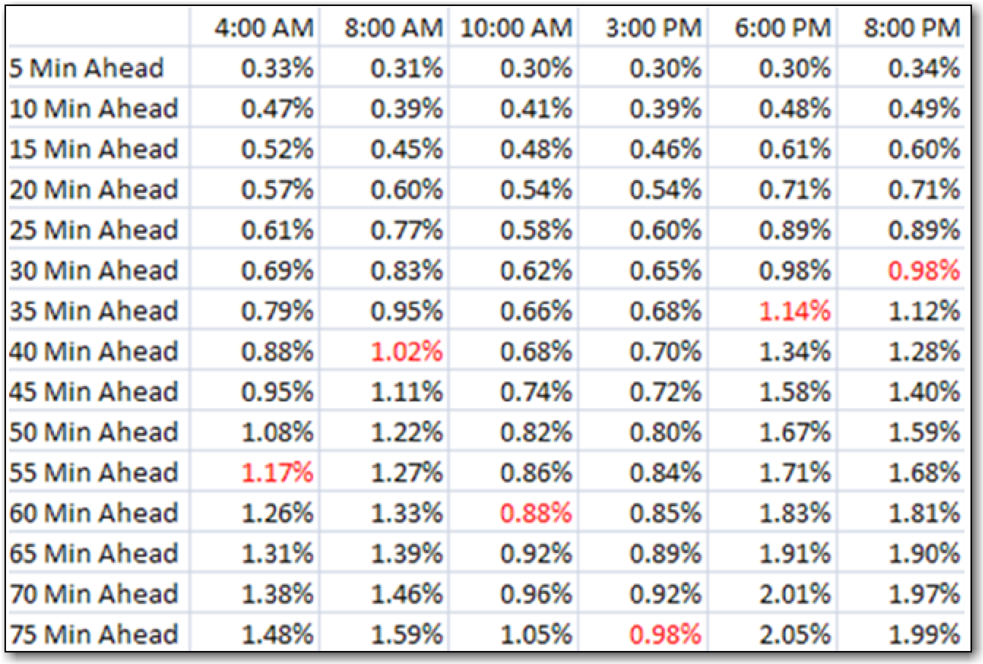
10-Minute Ahead Forecast Model: $Load_T = a_0 + \sum_{j=2}^{24} a_j Load_{T-j}$

15-Minute Ahead Forecast Model: $Load_T = a_0 + \sum_{j=3}^{24} a_j Load_{T-j}$

75-Minute Ahead Forecast Model: $Load_T = a_0 + \sum_{j=15}^{24} a_j Load_{T-j}$

Table 1 is the sequence of forecast model MAPEs (Mean Absolute Percent Errors) by length of the forecast horizon. The columns represent the target forecast period. The rows represent the length of the forecast horizon. The row labeled *5 Min Ahead* contains the five-minute ahead forecast MAPEs. Highlighted in red is the forecast horizon at which the forecast MAPE from the autoregressive model exceeds the forecast MAPE for the corresponding short-term forecasting model. In this example, the forecast MAPE for loads at 8:00 a.m. from the short-term forecast model is 1.0%. For forecast horizons of longer than 35 minutes ahead, this implies that it would be better to use the forecast from the short-term model.

Table 1: Forecast MAPEs by Length of Lag Structure



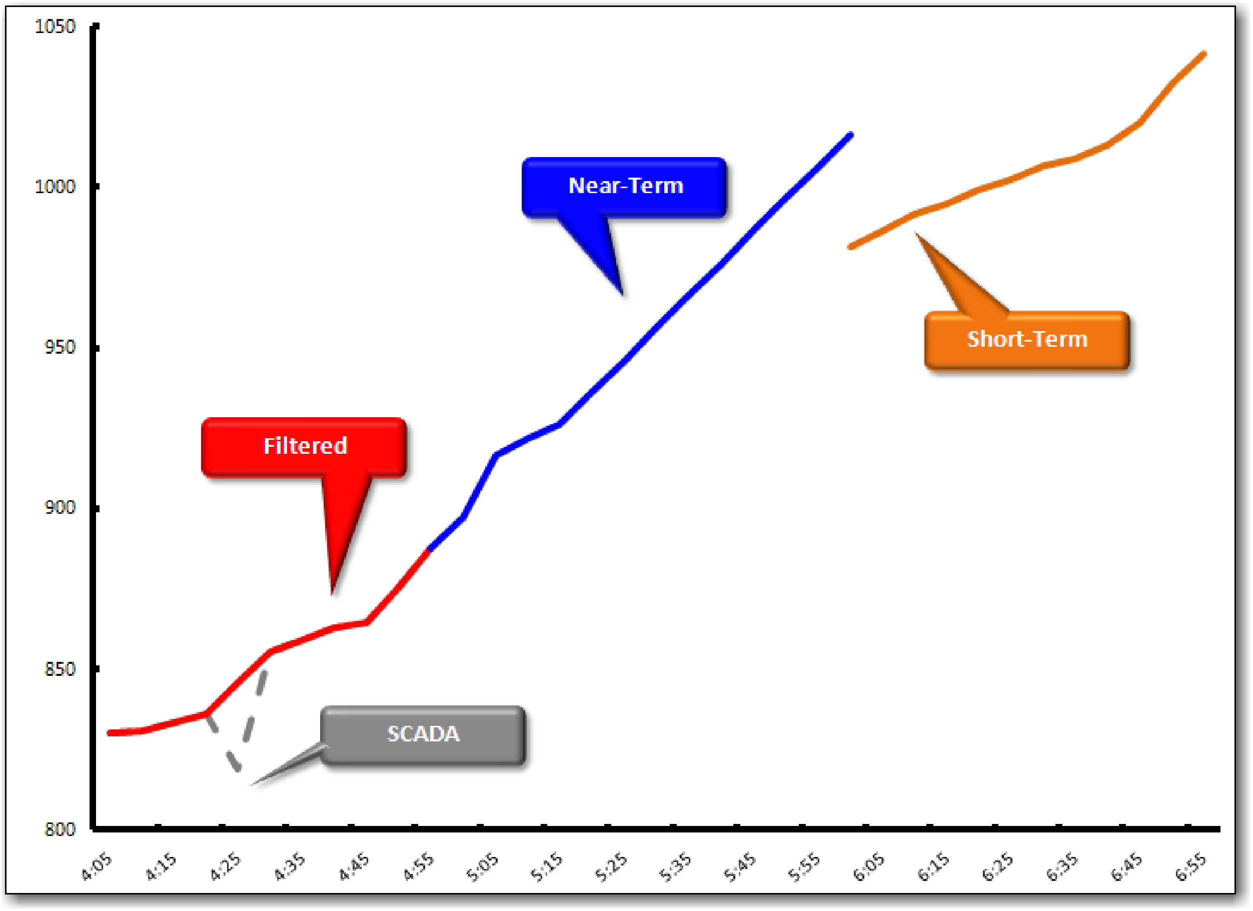
	4:00 AM	8:00 AM	10:00 AM	3:00 PM	6:00 PM	8:00 PM
5 Min Ahead	0.33%	0.31%	0.30%	0.30%	0.30%	0.34%
10 Min Ahead	0.47%	0.39%	0.41%	0.39%	0.48%	0.49%
15 Min Ahead	0.52%	0.45%	0.48%	0.46%	0.61%	0.60%
20 Min Ahead	0.57%	0.60%	0.54%	0.54%	0.71%	0.71%
25 Min Ahead	0.61%	0.77%	0.58%	0.60%	0.89%	0.89%
30 Min Ahead	0.69%	0.83%	0.62%	0.65%	0.98%	0.98%
35 Min Ahead	0.79%	0.95%	0.66%	0.68%	1.14%	1.12%
40 Min Ahead	0.88%	1.02%	0.68%	0.70%	1.34%	1.28%
45 Min Ahead	0.95%	1.11%	0.74%	0.72%	1.58%	1.40%
50 Min Ahead	1.08%	1.22%	0.82%	0.80%	1.67%	1.59%
55 Min Ahead	1.17%	1.27%	0.86%	0.84%	1.71%	1.68%
60 Min Ahead	1.26%	1.33%	0.88%	0.85%	1.83%	1.81%
65 Min Ahead	1.31%	1.39%	0.92%	0.89%	1.91%	1.90%
70 Min Ahead	1.38%	1.46%	0.96%	0.92%	2.01%	1.97%
75 Min Ahead	1.48%	1.59%	1.05%	0.98%	2.05%	1.99%

The results indicate that during periods of the day where the load shape is relatively stable (e.g., 10:00 a.m. and 3:00 p.m.), the most recent load data contain good predictive power for relatively long forecast horizons. Conversely, during periods of the day marked by turning points in the load (e.g., 8:00 a.m., 6:00 p.m., and 8:00 p.m.), the most recent load data contain good predictive power for relatively short forecast horizons. This implies that the forecast framework needs to be flexible in terms of defining the cross over point between the near-term and short-term forecasts for each period or blocks of periods of the day.

How do we bridge the gap between the near-term and short-term forecasts?

The above analysis suggests that at some point in the forecast horizon, it is better to use the forecast results from the short-term models rather than the results from the autoregressive models. While this sounds straightforward, in practice, the two alternative forecasts rarely line up at the cross over point. An example of the gap that can exist between the end of the near-term forecast horizon and the beginning of the short-term forecast is presented in Figure 6. There are three ways to bridge the gap.

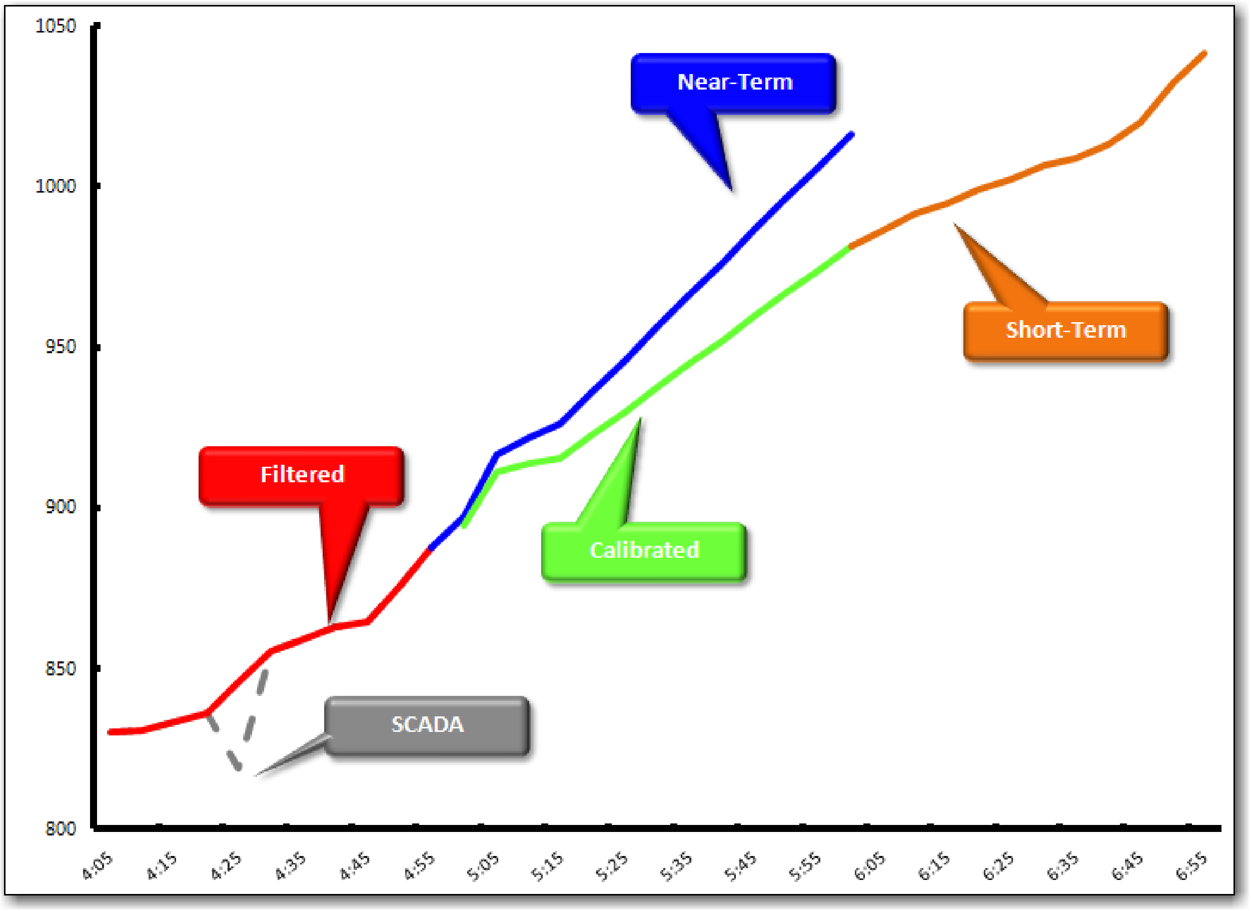
Figure 6: Bridging the Gap Between the Near-and Short-Term Forecasts



Option 1: Calibrate the Near-Term Forecast to the Short-Term Forecast

With this option, the near-term forecast is rotated up or down to ensure that both forecasts are the same at the cross over point. An example of calibration is depicted in Figure 7. Here, the green line represents the calibrated near-term forecast. While calibration solves the bridging-the-gap problem, it does so by reducing the forecast power of the near-term forecast. That is, the calibrated near-term forecast (green line) is not what comes out of the near-term forecasting models (blue line).

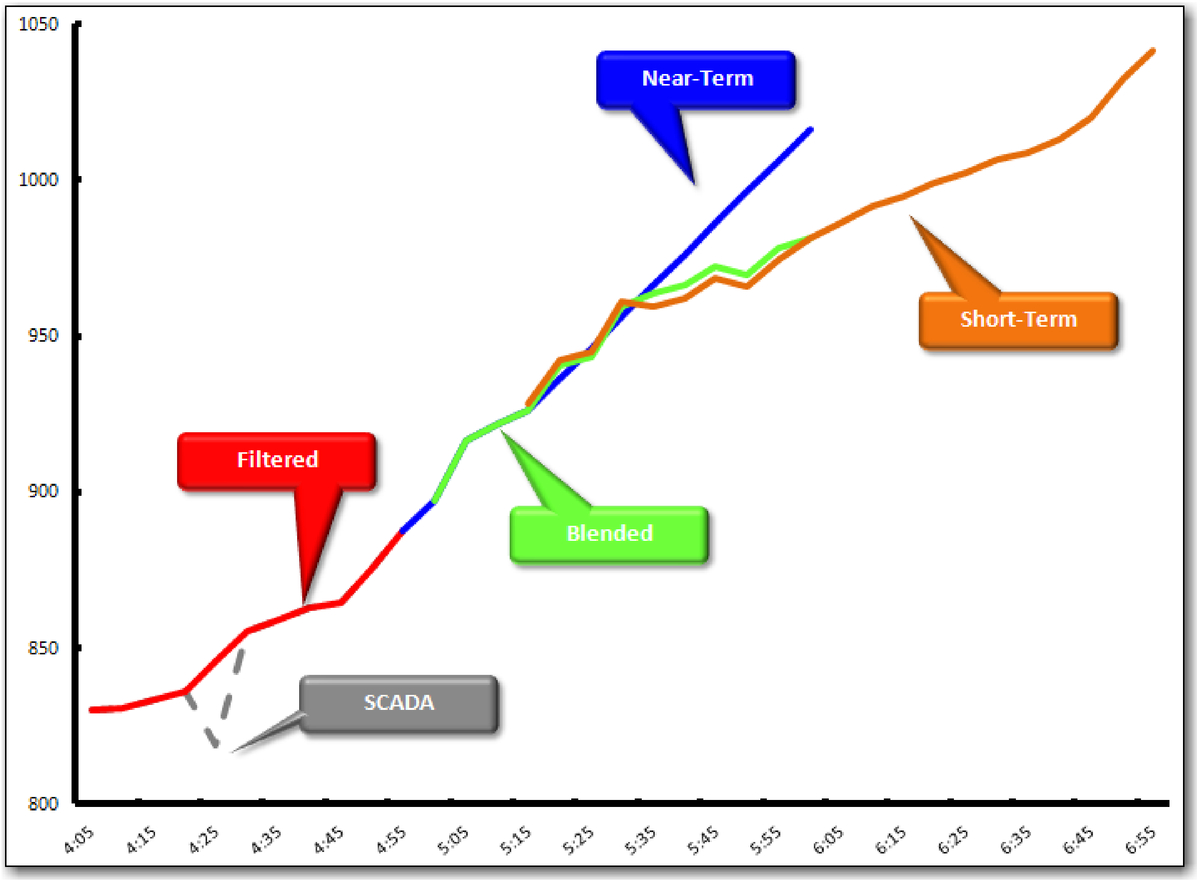
Figure 7: Bridging the Gap with Near-term Model Calibration



Option 2: Blending the Near-Term and Short-Term Forecasts Together

Under this option, a weighted average of the near-term and short-term forecasts is used to cover the period of the gap and the surrounding periods. An example of blending is depicted in Figure 8. Here, the green line is the blended forecast. The blended forecast equals the near-term forecast for a user-specified period of time. This is followed by a weighted average of the near-term and short-term forecasts. The blending scheme would progressively put more and more weight on the short-term forecast the further out into the forecast horizon. Finally, the blended forecast becomes the short-term forecast. The challenge with the blending approach is constructing an optimal blending scheme that retains the forecast power of the near-term forecast while at the same time introducing the forecast power of the short-term forecast. It is also plausible to expect that the blending scheme would be different for different periods of the day. A downside of blending is that the weighted average forecast is a relatively smooth forecast. This smooth forecast may miss systematic ramp rate changes that occur at critical points in the day.

Figure 8: Bridging the Gap with Model Blending



Option 3: Use a Ramp Rate model to Bridge the Gap

The third option is to use the predicted ramp rates from a Ramp Rate model to bridge the gap between the near-term and short-term forecasts. Figure 9 presents the five-minute daily load shapes for a typical utility system. The graph shows all of the data for one month plus the average load shape across the month (in blue); each line represents one day. These data show a large variation in the load shapes across the days, ranging from a low of approximately 1,000 MW to a high of 2,500 MW. This reflects variations due to day-of-the-week (e.g., weekdays versus weekends) and weather conditions. The near-term and short-term models are designed to predict the 1,500 MW swing in loads from one day to another.

Figure 9: Load Data

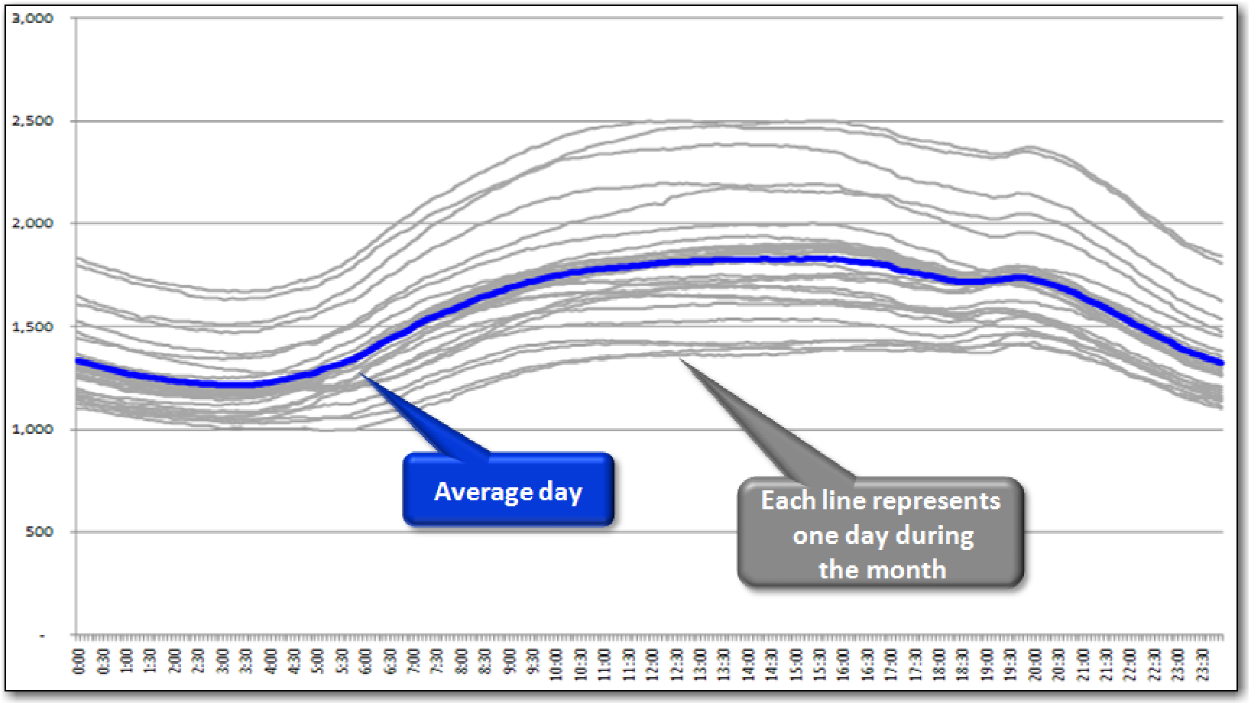


Figure 10 depicts the corresponding ramp rates along with the average ramp rate shape for these data. The average ramp rate shape captures changes in loads that consistently occur across the days. For example, a 10 MW swing in loads that consistently occurs from 8:00 a.m. to 8:05 a.m. will be captured by a ramp rate model. By using the ramp rate model to bridge the gap between the end of the near-term forecast and the beginning of the short-term forecast, we are able to capture these consistent ramp rate changes. These ramp rate changes would be lost using a blending algorithm since the weighted averaging of the blending will smooth through the near- and short-term forecasts. This would be especially true if the short-term model is a model of 15-minute, 30-minute or 60-minute loads. At that level of load resolution, systematic ramp rate changes at the five-minute level will be lost. Figure 11 depicts how the results of the ramp rate model can be used to bridge the gap between the autoregressive near-term forecast and the short-term forecast.

Figure 10: Ramp Rate Data

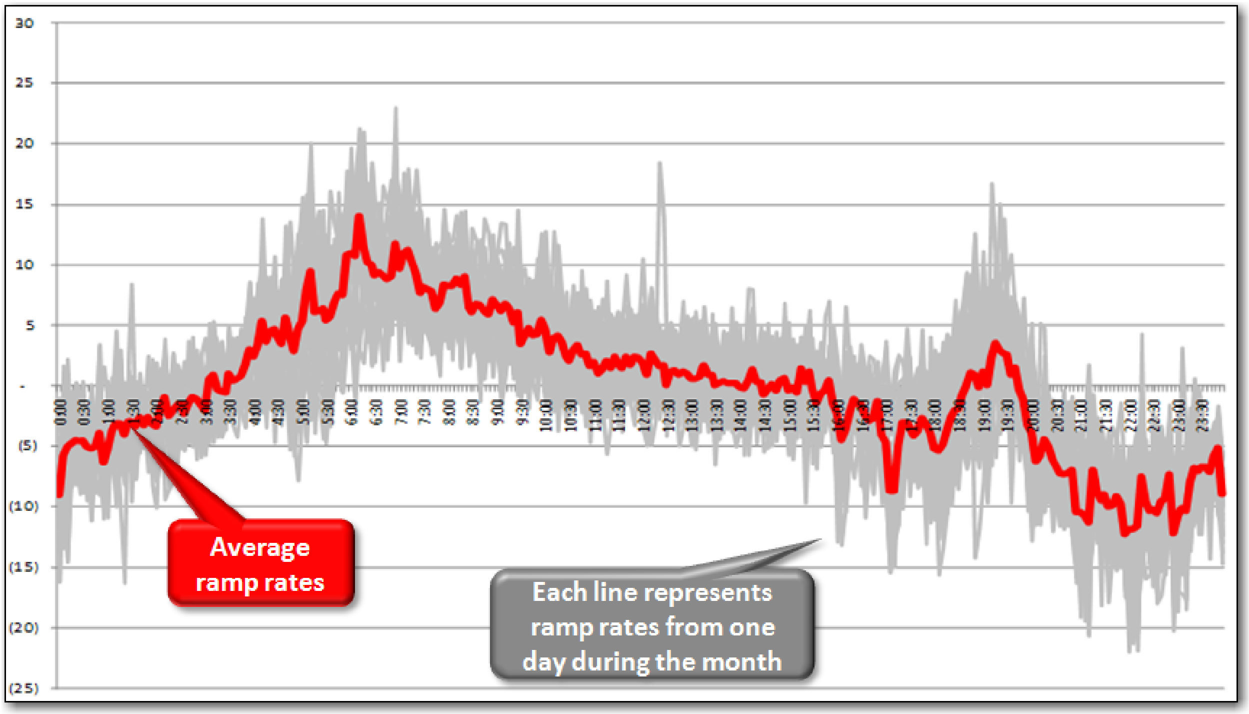
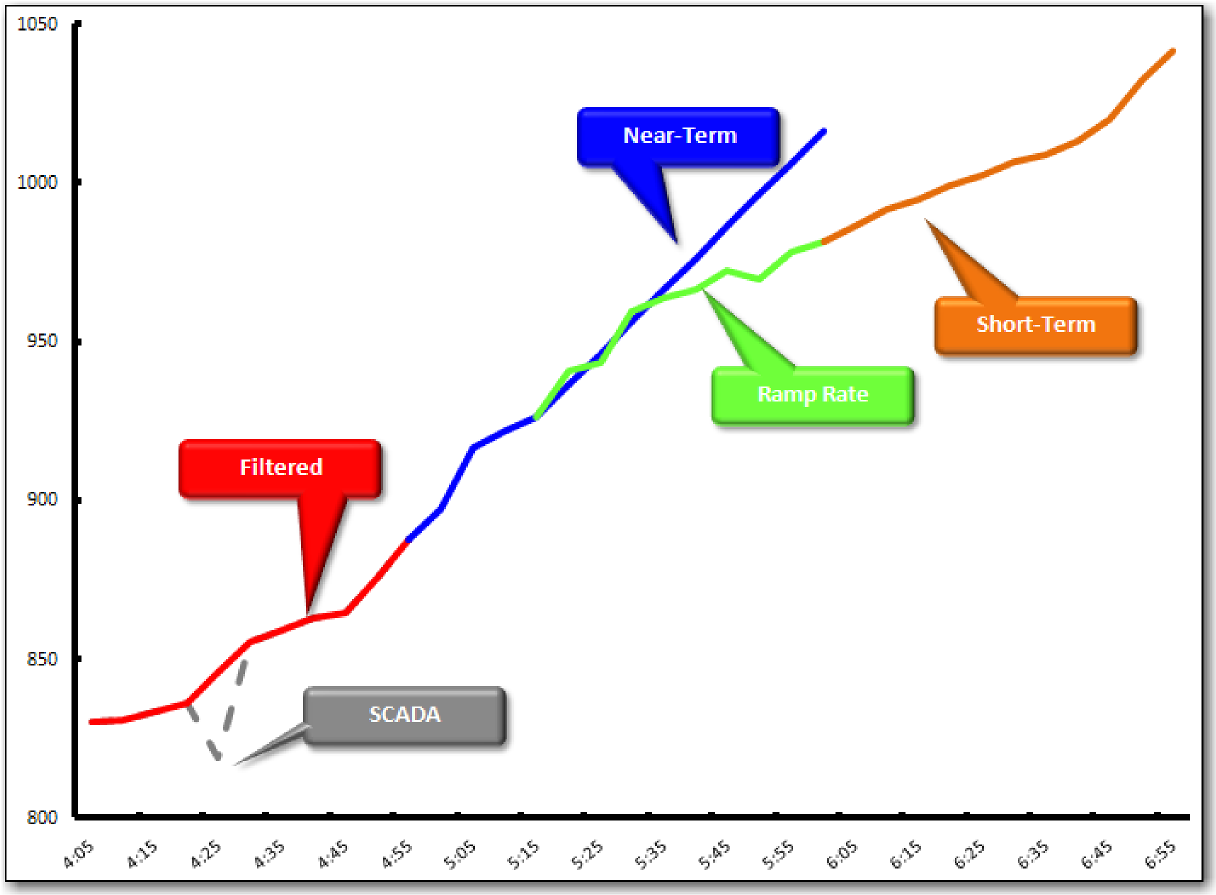


Figure 11: Bridging the Gap with the Ramp Rate Model



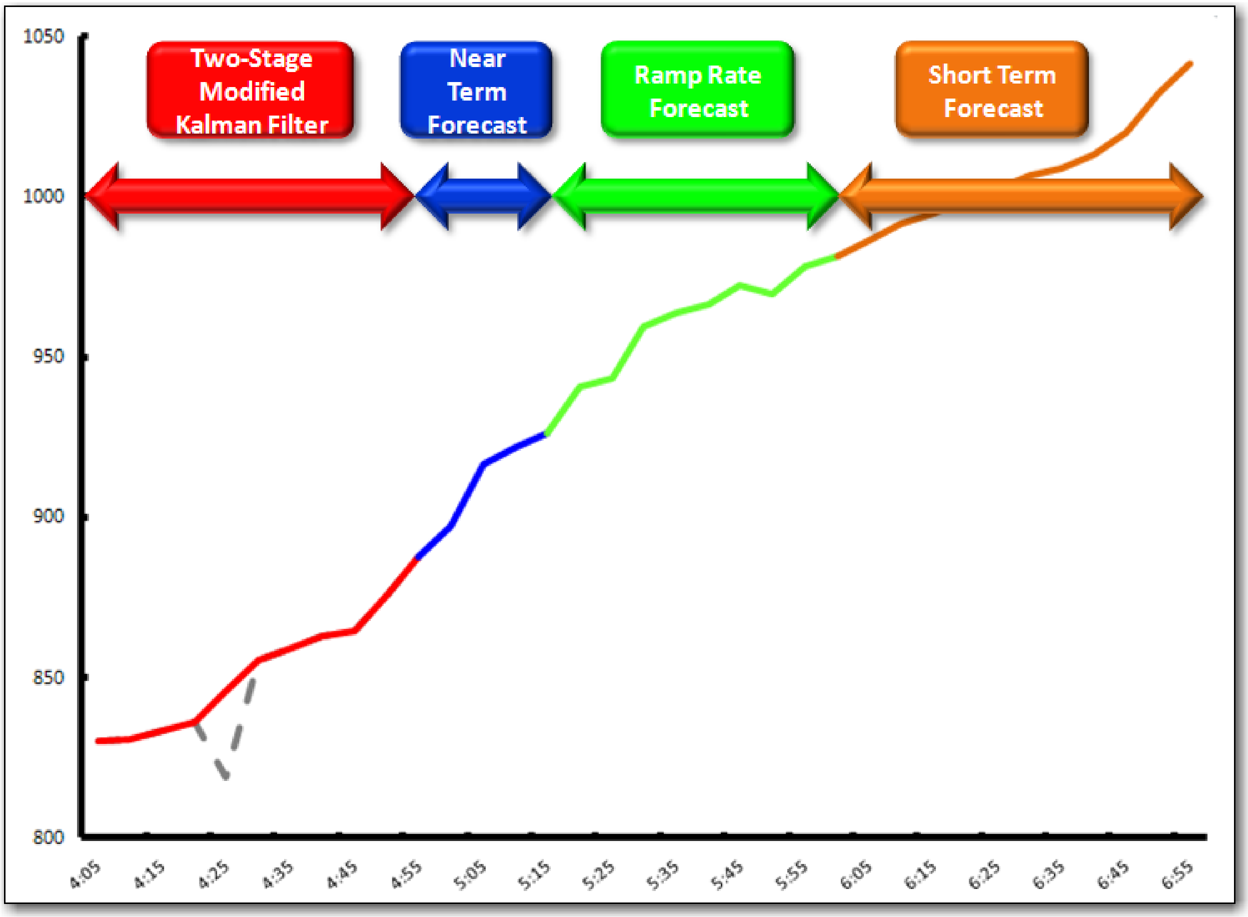
Framework Summary

As depicted in Figure 12, the key features of the unified near-term and short-term forecasting framework are as follows.

- The Two-Stage Modified Kalman Filter is used to smooth through outliers in the SCADA reads. The results of the filter are passed to the near-term load forecasting model.
- The highly autoregressive near-term load forecasting model that leverages the most recent load data is then used to forecast loads over the next 5, 10, 15, and possibly 20 minutes ahead.
- The forecasted ramp rates from a Ramp Rate model are used to blend the forecasts from near-term load forecast model to the forecast from the short-term forecast model.
- The short-term load forecast derived from a model is driven by calendar, solar, and weather conditions. This is the same short-term model that is used to produce the next-day forecast.

In the end, the forecast operator sees a single load forecast that runs from the end of the last actual observation through the next-day or several days ahead. As a result, the forecast operator has a unified forecasting framework that meets both his near-term and short-term forecasting requirements.

Figure 12: Elements of a Unified Near-Term and Short-Term Load Forecasting Framework



The forecast framework presented here addresses two key challenges to system load forecasting: (1) data filtering to smooth through the most recent SCADA reads to prevent load spikes from echoing into the forecast horizon, and (2) developing accurate near-term and short-term forecasts given forecasts of calendar, solar, economic and weather conditions. This general forecast framework is made specific to each system, through the process of estimating the models used to drive the Two-Stage Modified Kalman Filter, as well as the near-term load forecasting model, the ramp rate forecasting model, and the short-term forecasting model. The framework works well for large, relatively stable systems where the short-term models can accurately capture the influence of calendar, solar and weather conditions on the loads, as well as small, relatively noisy systems or sub-regions of a larger system where the data filtering can *smooth* the SCADA data.

Itron Inc.

Itron Inc. is a leading technology provider to the global energy and water industries. Our company is the world's leading provider of metering, data collection and utility software solutions, with nearly 8,000 utilities worldwide relying on our technology to optimize the delivery and use of energy and water. Our products include electricity, gas, water and heat meters, data collection and communication systems, including automated meter reading (AMR) and advanced metering infrastructure (AMI); meter data management and related software applications; as well as project management, installation, and consulting services. To know more, start here: www.itron.com

To know more, start here: www.itron.com

Itron Inc.
Corporate Headquarters
2111 North Molter Road
Liberty Lake, Washington 99019
U.S.A.
Tel.: 1.800.635.5461
Fax: 1.509.891.3355