SPOCK: A Sensor Value Prediction and Online Control Algorithm for Building Resource Management

Neil Klingensmith
University of Wisconsin
Computer Engineering
Madison, WI, USA
naklingensmi@wisc.edu

Anantharaghavan Sridhar
University of Wisconsin
Electrical Engineering
Madison, WI, USA
ananth.sridhar@wisc.edu

Zachary LaVallee
University of Wisconsin
Mechanical Engineering
Madison, WI, USA
zlavallee@wisc.edu

Suman Banerjee
University of Wisconsin
Computer Science
Madison, WI, USA
suman@cs.wisc.edu

ABSTRACT

We present SPOCK, an algorithm for forecasting future sensor readings based on historical trends. Using sensor value forecasting, we can plan control decisions for building automation systems hours in advance, with the goal of improving resource consumption efficiency. We test SPOCK on a dataset gathered by our water softener optimization engine [13] on the UW-Madison campus. We use sensor value forecasts generated by SPOCK to plan water softener regenerations up to 24 hours in advance. We compare the performance of SPOCK to an oracle forecaster, which knows exactly what the sensor readings will be for all times in the future—this is possible because we are retroactively analyzing a dataset that has already been collected. For the dataset we analyze here, we demonstrate that SPOCK performs only about 2% worse than the oracle forecaster. Despite some errors in SPOCK's forecasts (compared to the oracle), it is still capable of making reasonable control decisions that are nearly optimal. SPOCK can reduce regeneration frequency—corresponding to reduction in wasted water— by 10% compared to existing algorithms for scheduling regenerations.

CCS Concepts

•**Computing methodologies → Modeling methodologies;**

Keywords

Forecasting; Water Treatment; Building Automation; Smart Buildings

BuildSys 2016 November 16–17, 2016, Stanford, CA, USA

ISBN 978-1-4503-4264-3/16/11…$15.00

DOI: http://dx.doi.org/10.1145/2993422.2993584

1. INTRODUCTION

A large body of work in the smart buildings and home automation research community focuses on the problem of inferring building or room occupancy using sensor data. The end goal of occupancy detection is to optimize building systems, streamline resource consumption, and improve the overall "user experience" for building occupants. System designers may also wish to predict future occupancy patterns in order to plan ahead for the arrival or departure of occupants. The Nest Thermostat's infrared motion detector is an example of a device that uses sensors to infer occupancy and control building systems [1].

In this work, we explore an alternative approach: we use raw historical sensor data as input to a forecasting algorithm that predicts future sensor readings. We use forecast sensor readings rather than forecast occupancy inferences to predict future resource-consuming activities within the building. We then make control decisions based directly on the forecast sensor readings.

This approach may have the advantage of retaining a high-bandwidth data stream, useful for predicting future activity. A lot of potentially useful information is lost in the process of generating the binary occupancy indicator—data that may be useful in making control decisions. By directly using historic sensor readings and future forecasts to make control decisions, our approach has the advantage of retaining the high-bandwidth data stream, which has more information content than a binary occupancy indicator function. This approach could result broadly in more accurate and more useful predictions than predictions of the occupancy statistic. Here, we study its application to one building automation system—the water softener—and demonstrate that it performs well on a dataset to which we have access.

Resource consumption forecasting potentially has applications in many building-related systems. Some microgrid controllers have experimented with rudimentary short-term energy consumption forecasting to plan control actions for energy sources and loads [6, 9, 10]. It may also be possible to achieve higher accuracy in sensor value forecasting by integrating sensor data from multiple modalities—for example electricity consumption, water flow, and natural gas consumption. In this study, however, we only explore sensor forecasting for a single sensor modality. We use resource

consumption forecasting in a water treatment controller application called AWESOME [13] to optimize water softener regenerations.

In this work, we develop a sensor forecasting algorithm we call SPOCK[1]. We evaluate the idea of using raw historical sensor readings to predict future sensor readings and take control actions based on the predictions. In the water softener application, the salient control action that can be taken is to regenerate the water softener. We do not directly compare to the occupancy detection approach in this work because, in our example application, an occupancy statistic would have limited utility. We leave that evaluation for future work.

AWESOME, our previous work related to water softeners, is essentially a feedback control system for water softeners. Its key innovation is that it closed the control loop by adding a water hardness sensor to identify and respond to filtration medium depletion. Existing water softener controllers do not include such a sensor, and as such, they cannot reliably determine when the filtration medium needs to be regenerated. By measuring the hardness of the water after it has been treated, AWESOME was able to identify when the filtration medium was depleted and alert the softener that it was time to regenerate. The softener would respond by regenerating immediately, using a backup unit to treat water during the regeneration. Many small water softeners, such as those used in private residences and small commercial buildings, do not have backup units that can be called on to treat water during a regeneration cycle. Instead, smaller softener systems simply bypass the filtration medium during regeneration cycles, sending hard water to the building. Because there is no backup, those systems must be regenerated during periods of time when the building is likely to use little or no soft water. For such systems, it is not appropriate to use AWESOME by itself, since it may initiate a regeneration during a period of high water demand. We have developed SPOCK as an additional feature that can be added on to AWESOME. SPOCK can use the sensor data gathered by AWESOME to predict when periods of low water demand will occur and schedule regeneration cycles appropriately.

What is a water softener? Water softeners stop lime buildup in pipes and equipment by removing dissolved minerals – Calcium and Magnesium ions – from tap water. A water softener's job is to remove minerals from water it treats. This is typically accomplished by exchanging Calcium and Magnesium ions with Sodium ions. As Calcium and Magnesium-rich water passes through a filtration medium, Sodium ions, weakly bound to the medium, are released into solution. The Calcium and Magnesium ions replace the Sodium on the filtration medium. Eventually, the surface of the filtration medium becomes saturated with Calcium and Magnesium ions, and it can no longer treat water. It must be regenerated by flushing with concentrated salt brine, which replaces Calcium and Magnesium ions with Sodium, preparing the softener to treat more water.

The objective of this work is to schedule regenerations just before the filtration medium becomes saturated using information gathered from flow and hardness sensors. We impose the additional constraint that the softener must be regenerated during a time of minimal water consumption by the building. This is because, for many water softening systems, it is not possible to treat water during a regeneration[2]. During a regeneration, the filtration medium is bypassed, and all water used by the building will be hard. We want to minimize the consumption of bypassed hard water by scheduling a regeneration during a period of low water use. We will do this by forecasting future water flow volumes, and scheduling a regeneration during a time of predicted low flow.

The sensor readings we deal with in this work are water flow rate and quality. Flow rate data is generally cyclical, following 24-hour periods of human activity. As such, it is fairly easy to predict future flow rates with high accuracy. The water quality sensors measure mineral concentration or *hardness* of water after being treated by a water softener. The water hardness traces we collected are sparse—most sensor readings are zero, with a few blips of high hardness.

Our goal in this work is to predict with some reasonable accuracy future values of water flow and hardness and use those predictions to decide when we should regenerate a water softener. We evaluate our methods by comparing to the existing *reserve capacity* method implemented on most residential water softeners today. We also compare our forecasting algorithms to the *oracle forecaster*, one that knows exactly what the future values of all sensor readings will be. We find that, despite some errors in sensor reading forecasts, SPOCK performs nearly optimally as compared to the oracle forecaster.

In this work, we make the following key contributions:

- **We develop a sensor value forecasting algorithm that predicts future sensor readings based on historical data.**
- **We design a control algorithm that uses sensor value forecasts to schedule water softener regenerations based on a cost function that trades efficiency for utility.**
- **We compare our control algorithm to existing techniques for scheduling regenerations, demonstrating that we can increase efficiency of water usage by 10%.**

2. BACKGROUND

2.1 Water Softener Application

In previous work, we presented an online feedback control system for water softeners called AWESOME that reduces the amount of salt and water wasted by these systems [13]. Water softeners, which remove Calcium from hard water, must be regenerated periodically because the filtration medium inside the water softener becomes saturated with Calcium after it has treated a lot of water. The regeneration process flushes Calcium from the filtration medium, making it ready to treat more water. To flush the filtration medium, water softeners use large amounts of salt and water, which is wasteful and can pollute the environment.

Legacy water softener controllers do not actually measure the mineral concentration of the water they treat. Instead,

[1]SPOCK stands for **S**ensor **P**rediction and **O**nline **C**ontrol.

[2]Some industrial water softeners, like those studied in AWESOME, have backup units that allow them to treat water during a regeneration. In this work, we assume that feature is not available

they guess at how much water they can treat before regenerating. However, many parameters of the system can change over time, rendering the guess inaccurate. This can result in reduced performance of the water softener or wasteful water and salt usage. The AWESOME system improves the efficiency and reliability of water softeners by monitoring mineral concentration in real time and immediately initiating a regeneration of the water softener when it detects that the filtration medium has been depleted.

During a regeneration—a period of about 60-90 minutes—the water softener's filtration medium is unavailable for treating water. Instead, during that period, the softener will be bypassed, and any water used by the building will be untreated hard water. If AWESOME initiates a regeneration during a peak water usage period, it could cause large volumes of hard water to flow to the building, potentially causing damage to the plumbing system.

Instead, we want to find a way to schedule a regeneration of the water softener during a time when the building's water usage is at a minimum–preferably just before the filtration medium depletes.

2.1.1 Standard Regen Method: Reserve Capacity

Most water softeners currently use a method called *reserve capacity* to schedule regenerations. The system's reserve capacity is the expected volume of water that will be used by the building in a 24-hour period. This estimate is loosely based on the building size, number of occupants, occupant activities, etc., and it is hard-coded into the water softener's controller at installation time by the installer. The water softener is then put into service. It begins treating water, and, using a flowmeter, it subtracts the volume of water it treats from its theoretical capacity. Each morning at 2 AM (a time when water usage is assumed to be low), the water softener controller compares the remaining theoretical capacity to the reserve capacity to determine if it has enough remaining capacity to treat water for another full day. If the softener's remaining capacity is less than the reserve capacity, it regenerates at 2 AM—even if the softener's remaining capacity is one gallon less than the reserve. If the remaining capacity is greater than the reserve capacity, it waits until the next day at 2 AM to recompute.

Clearly, there are many potential pitfalls in this approach. The building's actual water consumption patterns may not match the reserve capacity hardcoded into the softener's controller. Also, 2 AM may not be the only opportune time of day to regenerate: there may be multiple periods during the course of a day when flow is low enough to regenerate.

To study the efficiency of the reserve capacity method, we took the flow rate measurements gathered from one of our installations and processed them with the reserve capacity algorithm. The algorithm identified times when the water softener would have regenerated using a reserve capacity. We varied the reserve capacity—the expected volume of water used by a building in a day—from 2,000 to 26,000 gallons, and we recorded the actual number of gallons at which the softener regenerated. This experiment was done on a dataset consisting of 225 days of flow rate data starting in June, 2015. The results of this test are shown in Figure 1.

The average water usage of the building that this test was run in is consistently about 14,000 gallons. However, to ensure that we never overrun the softener's theoretical capacity of 36,000 gallons, the reserve capacity needs to be set to at least 18,000 gallons. If we set the reserve capacity to 18k gallons, the softener regenerates on average at 30k gallons, wasting 17% of its theoretical capacity on average. That is the best the reserve capacity algorithm can achieve using rigorous statistical analysis of the building's water consumption with nine month's data as input—methods and data that are not available to water softener installers.

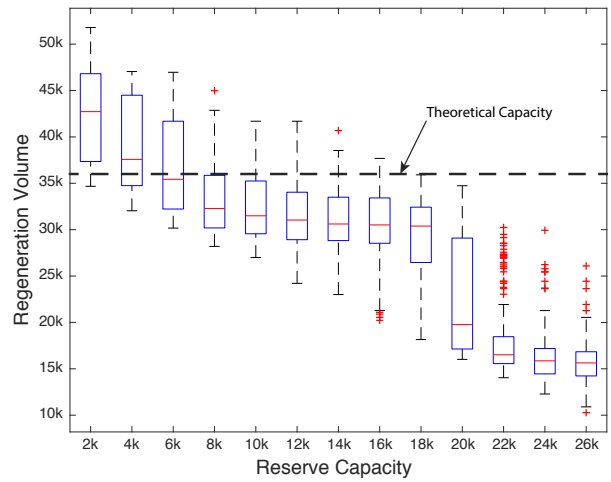

Figure 1: Reserve capacity (x-axis) vs. regeneration volume (y-axis) for flow data collected in a senior living facility over the course of 225 days in 2015-2016. For the same flow data, we simulated reserve capacity-based regenerations, using the same theoretical water softener tank capacity of 36k gallons. We varied the reserve capacity from 2k to 26k gallons to determine what the optimal setting should be. To ensure that we do not overrun the theoretical capacity of the softener, we must set the reserve capacity to 18k gallons, which wastes 6k gallons per cycle on average—more than 17% of the theoretical capacity.

The problem that this work tries to solve is to use the building's actual water consumption patterns to adaptively choose a regeneration time that will maximize the system's efficiency without regenerating too late.

2.2 Related Work

The electric power research community has a body of work dedicated to load forecasting, used primarily for the purposes of generation planning, fuel purchasing, etc. [10]. In general, the time horizons of interest to the electric grid community are on the order of days to weeks. Charytoniuk and Chen achieved accurate forecasts of grid demand for time horizons of several hours using artificial neural networks [4]. We are not aware, however, of work in the electric power or microgrid community that responds to load forecasts with automatic control actions, a feature that is central to our work.

There has been much work in the building automation community centered on detecting and predicting occupancy. Beltran et. al. used thermal sensors deployed in common spaces to infer occupancy [3]. Ranjan et. al. used fixture-level water consumption data to infer individual room occupancy within a building [17]. Others have used electric

power readings from smart meters to infer occupancy as well [5, 12].

There has also been a large body of work focused on responding to occupancy inferences or predictions [2, 7, 11, 16, 18]. Most of this work has focused on controlling thermostats in response to occupancy inferences or predictions. Our work differs from this category because it directly uses historical sensor readings to predict future resource utilization without first narrowing the data down to a binary occupancy indicator.

3. METHODS

In this section, we explain the data collection and processing techniques we used to process sensor readings taken from the water treatment system.

3.1 Data Collection

AWESOME uses water flow and quality sensors to track the status of a water softener in real time. We constructed custom hardware to gather and record data from water softeners. The data it collects is transmitted to a remote database for storage and post-processing. When backend algorithms running on the remote database server determine that the water softener needs to be regenerated, they transmit a signal to AWESOME, which forces the water softener to regenerate. A diagram of the dataflow used by AWESOME is shown in Figure 2.

The AWESOME computer board includes an embedded computer and several sensors to track the health of a water softener system:

The AWESOME embedded computer is a custom device featuring a Freescale ColdFire microcontroller operating at 60 MHz. The AWESOME board is a derivative of Emonix [14], and it includes many of the same software features—multithreaded tasking, POSIX-like programming environment, BSD sockets, etc. It also includes an xBee WiFi network controller which allows it to directly connect to the building's network. The board includes six programmable digital/analog sensor inputs through which sensors can be connected to collect data. An on-board 32 kbyte SRAM device is included to allow the board to cache data samples in the event of a network outage.

The AWESOME embedded device is responsible for sampling sensors, preprocessing and caching the data, and relaying it to the backend database through the building's WiFi network. When the backend determines that a regeneration is required, it sends a message to the embedded device, which initiates a regeneration of the water softener.

A Calcium Ion Selective Electrode (ISE) produces a voltage signal that is proportional to the Calcium concentration of the water sample. These devices are extremely sensitive to temperature and other factors, and their outputs are known to drift over time. By itself, the Calcium ISE cannot tell us the concentration of Calcium in our water samples, so we need to use auxiliary sensors to augment our data.

A Water Temperature Sensor is used in combination with the Calcium ISE to estimate the Calcium ion concentration in the water sample. We use the Nernst Equation [8] to combine the Calcium ISE voltage and the water temperature to get an estimate of the Calcium ion concentration. Because the Calcium ISE's output voltage is prone to drift over time, we cannot use the raw output of the Nernst equation to make decisions about whether or not the water is hard.

A Flowmeter is already installed in most commercial water softeners. Flow rate data is used by the stock water softener controller to decide when the softener should regenerate—in fact, this is the only sensor included by default on most commercial softener systems. Fortunately, all water softener flow meters use the same interface to communicate with their controllers. AWESOME can intercept and record the flow rate signal.

3.2 Algorithm

The algorithm described in this section takes flowrate, hardness, and other sensor readings (described above) and identifies an optimal time to regenerate the water softener. The data processing techniques described in this work can be broken down into three major steps:

1. Using historical data gathered from the AWESOME sensor, forecast water flow and hardness for a 24-hour time horizon.

2. Construct two cost functions from the forecast data: one that represents the cost of regenerating the water softener (increasing during times of higher flow) and one that represents the cost of not regenerating (increasing with higher hardness). Regenerating the water softener during times when water flow is high would result in large volumes of untreated hard water being passed through building systems. This cost function will have large values during times when flow is forecast to be high and small values during times when flow is forecast to be low.

3. Compare the two cost functions to find the optimal time window for regeneration, trading water usage efficiency with utility.

3.2.1 Sensor Forecasting

The two metrics we are interested in forecasting are water flow and water hardness. These are the two important inputs to the cost function, which we will use to decide when to regenerate the water softener.

Flow Forecasting

We want to regenerate the water softener at a time when minimal water will be used by the building. To choose such a time, we will forecast water flow for a 24 hour time horizon. We do this because we do not want to begin regenerating the water softener at a time when the building's water consumption is likely to increase significantly in the near future. Since the water softener takes 60-90 minutes to regenerate (during which time it cannot treat water), we need to find a time frame when water flow is likely to be minimal.

A statistical analysis of water flow patterns in various buildings we studied indicate that the flowrate function is not a stationary process[3]. This finding makes sense because activities in a building will likely follow daily circadian fluctuations. A histogram of water flow rates from a residence hall is shown in Figure 3. At 4 AM, when most residents are asleep, the histogram indicates that the recorded flow rates

[3]A stationary process is one whose statistical properties do not change as a function of time. In other words, X_t is stationary iff $F_X(x_{t_1}...x_{t_k}) = F_X(x_{t_{1+l}}...x_{t_{k+l}})$

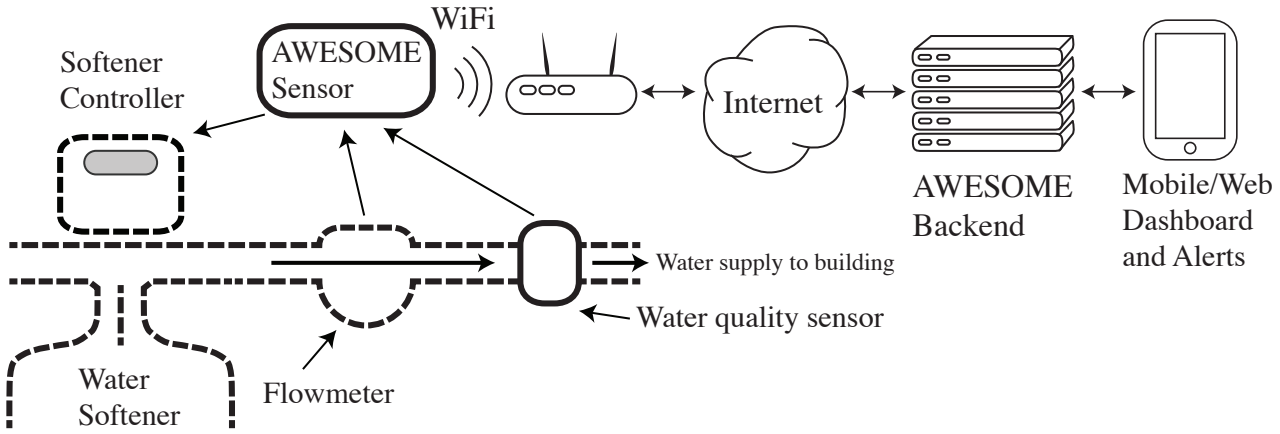

Figure 2: Dataflow diagram for the AWESOME system. Components with solid lines are part of AWESOME, and components with dashed lines are part of the existing softener system.

are mostly zero. Flow rates increase around 8 AM when residents wake up to go to class. At noon, flowrates increase further because lunch is served in the building's cafeteria. Since the distribution of water flow rates changes during the course of the day, we conclude that the process is nonstationary.

Since the flowrate signal is nonstationary, we need an analysis method that can deal with data that has a distribution that varies slowly over time. We chose to use an autoregressive model as a tool to forecast future flowrate patterns [15].

The flowrate signal that we are studying tends to be bursty—it tends to be high for a short time when a fixture is turned on and low when the fixture is off. In a large building such as those that we study here, that burstyness is contributed by hundreds of fixtures, each contributing to a noisy signal with a lot of high-frequency components. Unfortunately for us, autoregressive models are fairly sensitive to noise in the input signal. To deal with noise in the flowrate signal, we chose to first pass it through an integrator, which computes the total volume of water consumed by the building since the beginning of the timeframe of interest. This cumulative flow signal is probably a more useful metric anyway, since we are often concerned with the total amount of resource consumption.

The cumulative volume signal, having been passed through an integrator, is very smooth, and is a good candidate for use with an autoregressive model. The autoregressive model then predicts the cumulative flow used by the building on a minute-by-minute basis, 24 hours into the future. The results of the predictions made by the autoregressive model will be presented in Section 4.2.

The minute-by-minute data, while useful to the autoregressive model because of its high information content, is not required for the purposes of constructing the cost function. For the water softener application, we do not need to know the exact moment when it is optimal to regenerate. Instead, we are looking for a time window—typically on the order of several hours in length—when a regeneration would be appropriate. For this reason, we downsample the cumulative flow predictions from 1-minute sampling frequency to 1-hour sampling frequency. The downsampling has the effect of lowpass filtering the flow forecasts.

The final step in our forecasting process is to differentiate the downsampled cumulative volume predictions. This gives us the predicted *flow rates* on an hour-by-hour basis for a 24-hour time horizon. In the next steps of our algorithm, we will search this forecast for time periods when flow is minimal—times when it will be least costly to bypass the water softener and regenerate.

Hardness Forecasting

In addition to forecasting flow, we also need to predict water hardness readings, since these will inform our decision to regenerate. If we do not predict that the water softener's filtration medium will be depleted in the near future (i.e. water will not be hard), there is no point in regenerating.

Our hardness forecasting technique, like our flow forecasting technique, uses an autoregressive model to predict future sensor input values. Unlike the flow rate signal, the hardness signal is sparse. Most of the hardness readings are zero or nearly zero—this is expected because when the water softener is working properly, it removes all hardness from the water. It is only when the water softener's filtration medium depletes that we get nonzero hardness readings. The sparse nature of the hardness readings and the inherent nonlinearity of the hardness with respect to the cumulative water flow through the softener poses a challenge for learning linear models, such as, autoregressive models.

To account for the inherent nonlinearity so described, we use two operating point models (for low/high cumulative flow since the last regeneration). This is justified on the basis that the nonlinearity of the hardness measurement only manifests itself in the high flow region. A parallel can be drawn between our approach here and the general practice of linearizing nonlinear systems near typical operating points for the design of closed-loop controllers.

Additionally, since the hardness measurements depend on the cumulative flow through the softener since the last regeneration, in addition to the history of the hardness measurements themselves, the model we learn can be expressed in the following discrete-time state space notation ($flow(\dots)$ refers to the cumulative flow since the last regeneration):

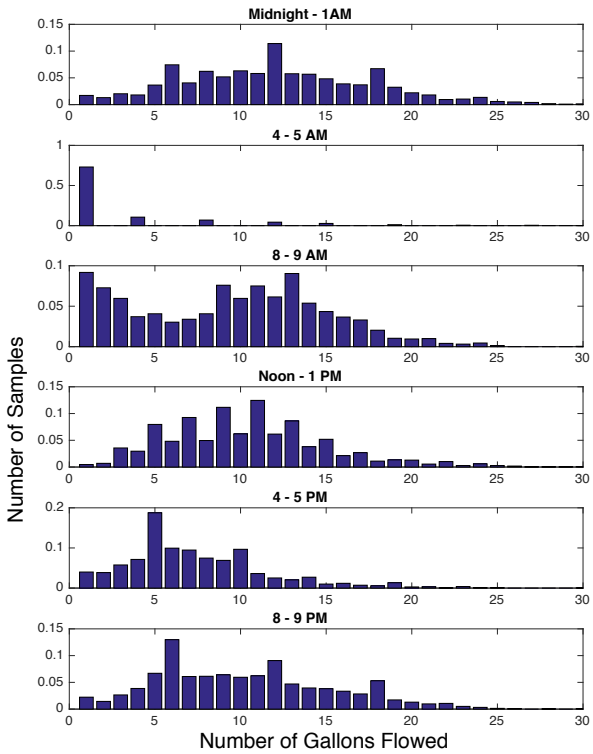

Figure 3: A histogram of flow rate measurements taken in a residential building across several months in 2015. The distribution of water flow rate changes depending on hour of the day, so the flow rate function is not stationary.

$$X(k+1) = \begin{bmatrix} A_{\text{flow-flow}} & 0_{\text{flow-hardness}} \\ C_{\text{hardness-flow}} & D_{\text{hardness-hardness}} \end{bmatrix} X(k)$$

$$X(k) = \begin{bmatrix} \text{flow}(k) \\ \dots \\ \text{flow}(k-20) \\ \text{hardness}(k) \\ \dots \\ \text{hardness}(k-20) \end{bmatrix}$$

$A_{\text{flow-flow}}$ captures the nature of the water usage by the residents of the building. Since this quantification is subject to rapid, unpredictable changes, we re-learn the flow forecasting model after each regeneration cycle thereby using the most recent water usage patterns for the forecasting, eliminating forecasting bias due to historical data.

The dependence of hardness measurements on their own history and the flow is quantified by the coefficient matrices of the mode $C_{\text{hardness-flow}}$ and $D_{\text{hardness-hardness}}$.

The model so described is learned from recorded data for two operating regions (low/high cumulative flow since the last regeneration). When the actual forecasting is done, one of the models is chosen based on the flow since the last regeneration at the time of forecasting.

3.2.2 Cost Function Setup

At the most basic level, there are really only two actions a water softener controller can take at any moment: regenerate the water softener now or wait until later to regenerate the water softener. After we have generated predictions of the flow and hardness signals for the next 24-hour timeframe, we use them to generate cost functions that capture the costs of either (1) regenerating the water softener or (2) not regenerating the water softener[4].

C_{diff} **Range**	Indicative Of
Large +ve	High Flow, Low Hardness
≈ 0	Low Flow, Low Hardness
Large -ve	High Flow, High Hardness

Table 1: Ranges of the cost function C_{diff}, and the corresponding sensor readings that cause them.

The cost of regenerating a water softener is largely measured in terms of the salt and water consumed in a regeneration. We know, however, that we will have to regenerate the softener occasionally, and our objective is to reduce the number of regenerations per gallon of water treated. The cost of not regenerating the water softener is captured by the potential damage to building infrastructure caused by flowing hard water to the building after the filtration medium has depleted. For this reason, we will use

$$C_{Regeneration} = af(t)$$

$$C_{NotRegenerating} = bf(t)H(t)$$

where a and b are normalizing constants, $f(t)$ is our forecast of the instantaneous flow rate through the softener, and $H(t)$ is our forecast of the hardness of the water after being treated by the water softener. These two cost functions are generated from our forecasts, which are based on historical sensor readings. Normalizing constants a and b are introduced to make $C_{Regeneration}(t)$ and $C_{NotRegenerating}(t)$ similar in magnitude. They can be used as knobs to tune the system to be more conservative or more aggressive at saving resources.

3.2.3 Cost Function Processing

To determine when to regenerate the softener, we compare the cost functions $C_{Regeneration}$ and $C_{NotRegenerating}$

$$C_{diff} = C_{Regeneration} - C_{NotRegenerating}$$

An example trace of C_{diff} is shown in Figure 4 in the bottom graph. Corresponding measurements of hardness and flow are shown in the plot on top. Ranges of C_{diff} and their corresponding meanings in terms of water hardness and flow are given in Table 1.

Our goal is to regenerate the softener before the hardness gets too high. However, we want to schedule the regeneration cycle during a period of low flow, because we will need to bypass the water softener during the regeneration cycle, and we do not want to flow a large volume of untreated water to the building.

Our algorithm for choosing a regeneration time based on a computed cost function is as follows:

1. Find minima of C_{diff}, that satisfy $C_{diff}(t_{min}) < T$, where T is a pre-chosen threshold. This represents a

[4]Here we use the term *cost* to mean not the monetary cost of taking one action or another, but the strain on infrastructure and resources.

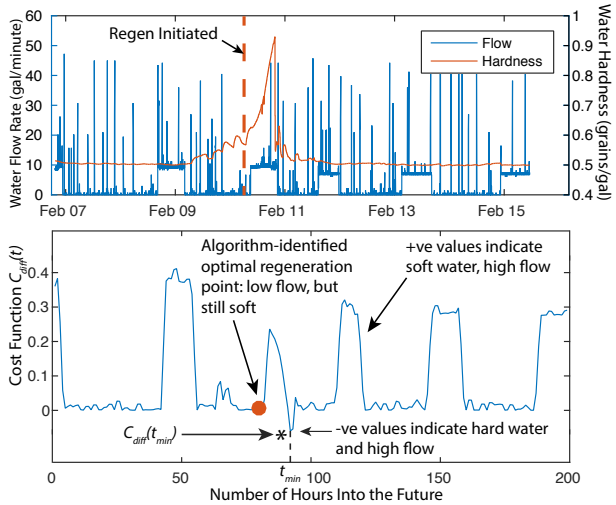

Figure 4: Top: Water flow and hardness measurements (y-axis) as a function of time (x-axis). Bottom: our cost function (y-axis) as a function of time (x-axis). Large positive values of the cost function are indicative of high flow but low hardness. Large negative values of the cost function are indicative of high hardness and high flow. Our algorithm chooses a point on the cost function to regenerate when forecasts of the cost function are close to zero, but before the cost function becomes negative.

time when the hardness has increased so that the cost of not regenerating is higher than the cost of regenerating (see Figure 4 (*)).

2. Search the cost function $C_{diff}(t)$ for values of $t < t_{min}$ that satisfy $C_{diff} \approx 0$ and $\frac{dC_{diff}}{dt} \approx 0$. This is the optimal time to regenerate.

If, when processing $C_{diff}(t)$, we find a minimum that is significantly less than zero, we know that our forecasts indicate that the water will become hard at the time the minimum occurs. If the threshold T is too close to zero, this algorithm may initiate a regeneration too early, perhaps in response to a spurious increase in hardness readings which results in elevated hardness forecasts. For this evaluation, we chose T by trial and error. Our algorithm is not highly sensitive to the choice of T, but we have observed that setting it too close to zero can cause early regenerations, which would reduce efficiency. We are calling that time t_{min}. However, by the time that minimum occurs, it will be too late to regenerate—we should have regenerated before the water became hard, or, in terms of the cost function, before C_{diff} became negative.

We want to schedule a regeneration when C_{diff} is close to zero. According to the way we've defined our cost function, values near zero indicate that we have low flow and low hardness. To find such a point, we will trace our cost function back from t_{min} to find a point on the cost function where C_{diff} is close to zero just before it becomes negative.

4. EVALUATION

In this section, we evaluate our forecasting methods and compare them to existing techniques for choosing when to regenerate water softeners. We will also compare our methods to an oracle forecaster, which knows exactly what the flow and hardness will be in the future.

4.1 Dataset

The dataset analyzed in this section consists of flow and hardness data collected from a single building by an AWESOME sensor during the months of January-June 2016. The building is a research lab on the University of Wisconsin campus that employs about 100 people.

4.2 Sensor Value Forecasting

Flow Forecasting

We first evaluate our flow forecasting algorithm by analyzing its relative prediction error as a function of the number of hours ahead we are predicting. We define relative prediction error as

$$e_{p_r}(h) = \left| \frac{p(t-h) - s(t-h)}{s(t-h)} \right|$$

where h is the number of hours in advance our algorithm is predicting, $p(t)$ is our prediction as a function of time, and $s(t)$ is the actual measured sensor value as a function of time. In this evaluation, we use roughly 14 days of historical data to make forecasts of 1-1.5 days in advance. Our computations of relative prediction error for cumulative flow are shown in Figure 5. Computations of relative prediction error for instantaneous flow are shown in Figure 6.

Relative prediction error for cumulative flow is on average below 30% for all time horizons we studied, and it actually decreases for longer time horizons. The instantaneous flow signal tends to be very bursty—as fixtures turn on and turn off in the building, water flow starts and stops sporadically. If these bursts do not arrive at the exact moments that our algorithm expects them to, but instead arrive several minutes before or after, then the near-term cumulative flow error will be higher. As long as the bursts of flow eventually arrive, the long-term cumulative flow is low because it captures all the flow that has happened over a long period of time, and the errors in the prediction cancel each other out.

The instantaneous flow error, however, does not benefit from this error canceling effect. If flow bursts do not arrive exactly as expected, the instantaneous flow error will be high. This does not actually hurt us though, and our predictions are good enough. As we will see later, it is not the absolute magnitude of forecast error that matters, but the timing of peaks and valleys in the cost function. As long as our algorithm can accurately predict the timing of peaks and valleys in instantaneous water flow forecasts, the later steps will work properly.

Hardness Forecasting

Our hardness forecasting algorithm is intended to predict whether or not water samples will read hard in the next 24-hour time horizon. Since our water hardness sensor readings are sparse—long stretches of zero readings followed by short periods of hard water readings—we are interested in predicting not only the exact value of the hardness sensor readings but also the presence or absence of hard water. Our goal is to correctly predict the value of the sensor reading, if possible. If, during a forecasting horizon, we do not correctly predict the hardness value, but we do correctly identify the

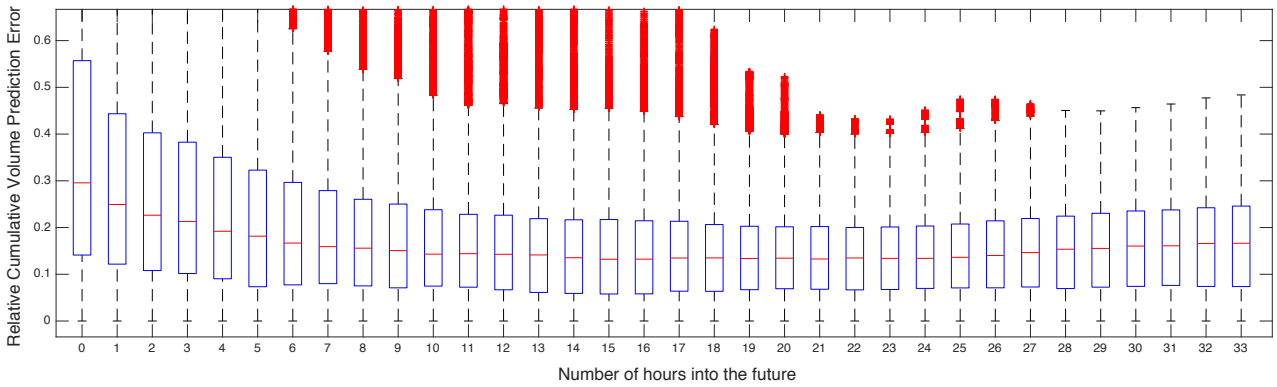

Figure 5: Relative prediction error of cumulative water flow (y-axis) as a function of the forecast horizon (x-axis). The predictions evaluated here are based on 14 days of historical data.

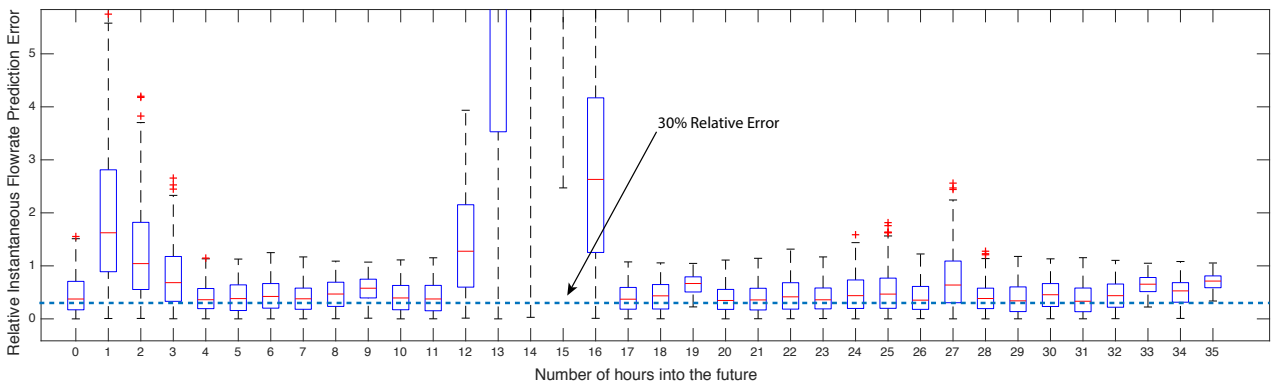

Figure 6: Relative prediction error of instantaneous water flow (y-axis) as a function of the forecast horizon (x-axis). The predictions evaluated here are based on 14 days of historical data.

presence or absence of a hard water event, then the algorithm has largely achieved its goal.

A plot of the relative errors of SPOCK's hardness prediction algorithms is shown in Figure 7. On average, the relative prediction errors are close to zero for time horizons up to 24-hours in the future. The variance, however, increases for predictions made further into the future.

A receiver operating curve for our hardness forecasting algorithm is shown in Figure 8. We generated the ROC by varying the threshold at which we identified the water as being hard—higher thresholds resulted in a higher false positive rate. Our dataset did not include hardness values high enough to generate true positive rates above 0.9 because AWESOME, the system used to gather the data, regenerated the water softeners before the hardness reached high values.

4.3 Comparison to Reserve Capacity and Oracle Algorithms

Here, we compare SPOCK to the oracle forecaster and the reserve capacity algorithms. The oracle forecaster algorithm is the same as SPOCK, except that the forecasting algorithm uses real data in the dataset rather than trying to predict future sensor readings. The reserve capacity algorithm is the method popularly used to control water softeners, described in Section 2.1.1.

Algorithm	Improvement over Reserve Capacity
Oracle	11.7%
SPOCK	9.9 %

Table 2: Improvements of the oracle algorithm and SPOCK over the reserve capacity.

Figure 9 shows a bar chart comparing the average number of gallons between regenerations for each of the three algorithms (higher numbers are better). Table 2 gives percentage improvements over the reserve capacity algorithm. SPOCK's autoregressive prediction algorithm outperforms the reserve capacity algorithm by roughly 10%. Furthermore, SPOCK performs only about 2% worse than the oracle prediction algorithm, which has access to much higher quality predictions about future sensor readings. Despite having access to imperfect sensor predictions, SPOCK can still make reasonable control decisions. Since our timescale of interest for scheduling regenerations is on the order of hours, even some errors in flowrate and hardness predictions can have negligible effects on the timing of control events. We acknowledge that this may not be the case for other building automation tasks, such as controlling light levels in rooms. How the timing of critical building automation events is affected by the accuracy of sensor predictions is still an open question.

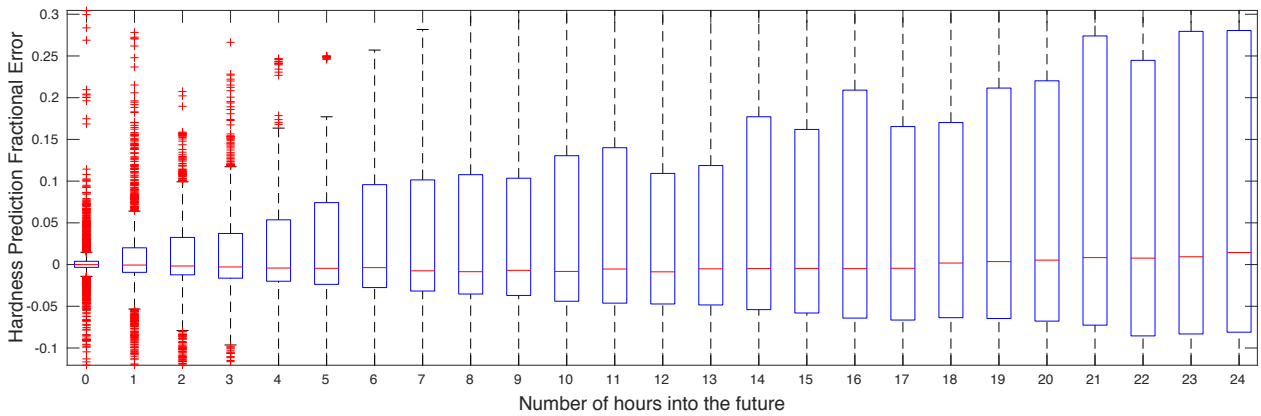

Figure 7: Relative prediction error of water hardness (y-axis) as a function of the forecast horizon (x-axis). The predictions evaluated here are based on 14 days of historical data.

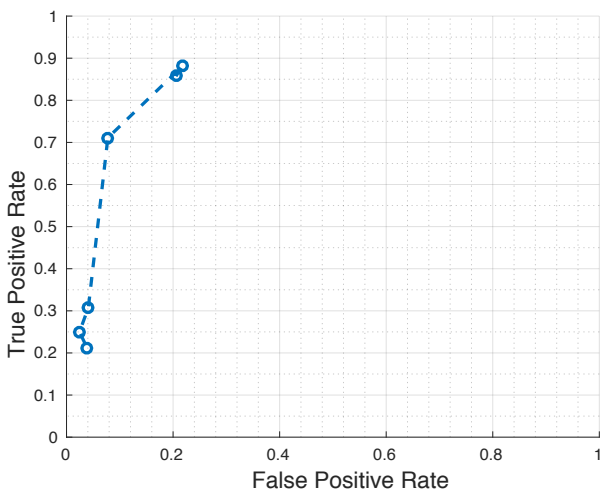

Figure 8: Receiver operating characteristic for SPOCK's hardness forecaster. We could not get the true positive rate above 0.9 because our dataset did not have hardness values high enough.

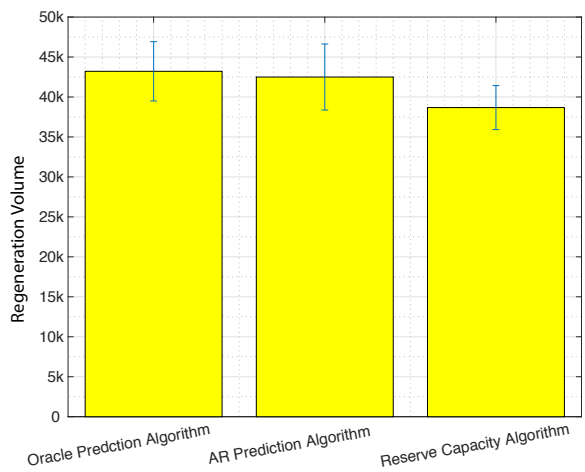

Figure 9: Comparison of tank capacity using (i) oracle regeneration scheduler, (ii) our autoregressive regeneration scheduler, and (iii) the reserve capacity algorithm. Our scheduler performs nearly as well as the oracle predictor, which knows exactly what the sensor readings will be for all future readings. Our autoregressive regeneration scheduler outperforms the reserve capacity scheduler by 9.9%.

5. CONCLUSION

In this work, we demonstrated a sensor value forecasting technique called SPOCK that could predict future sensor readings for a 24-hour time horizon. We developed an algorithm that could make control decisions for a water softener on the basis of our sensor forecasts, improving the efficiency of the water softener controller by roughly 10%.

We found that, despite some error in SPOCK's sensor value forecasts, it still makes near-optimal control decisions for the water softener application. This is because errors in our autoregressive model are not amplified significantly by our cost function postprocessing algorithm. There may be other building automation related applications for which this is not true: we are thinking here of time-sensitive control operations. For those applications, we would likely need to develop highly accurate forecasting algorithms. This could be achieved by using alternative algorithms (eg. artificial neural networks) or by exploiting mutual information from multiple sensor modalities such as electricity, water, and light sensors. We leave as an open question which classes of control problems lend themselves well to sensor value forecasting and what level of accuracy is required.

6. ACKNOWLEDGMENTS

The authors are supported in part by the US National Science Foundation through awards CNS-1555426, CNS-1525586, CNS-1405667, CNS-1345293, CNS-1343363, and CNS-1647152.

7. REFERENCES

[1] Nest, 2016. http://www.nest.com.

[2] N. Batra, A. Singh, and K. Whitehouse. If you measure it, can you improve it? exploring the value of energy disaggregation. In *Proceedings of the 2Nd ACM*

International Conference on Embedded Systems for Energy-Efficient Built Environments, BuildSys '15, pages 191–200, New York, NY, USA, 2015. ACM.

[3] A. Beltran, V. L. Erickson, and A. E. Cerpa. Thermosense: Occupancy thermal based sensing for hvac control. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys'13, pages 11:1–11:8, New York, NY, USA, 2013. ACM.

[4] W. Charytoniuk and M.-S. Chen. Very short-term load forecasting using artificial neural networks. *IEEE transactions on Power Systems*, 15(1):263–268, 2000.

[5] D. Chen, S. Barker, A. Subbaswamy, D. Irwin, and P. Shenoy. Non-intrusive occupancy monitoring using smart meters. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys'13, pages 9:1–9:8, New York, NY, USA, 2013. ACM.

[6] H. Chen, J. Wu, and S. Gao. A study of autoregressive conditional heteroscedasticity model in load forecasting. In *2006 International Conference on Power System Technology*, pages 1–8. IEEE, 2006.

[7] V. L. Erickson, M. Á. Carreira-Perpiñán, and A. E. Cerpa. Observe: Occupancy-based system for efficient reduction of hvac energy. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 258–269. IEEE, 2011.

[8] D. C. Harris. *Exploring Chemical Analysis*. W. H. Freeman, 2012.

[9] L. Hernández, C. Baladrón, J. M. Aguiar, B. Carro, A. Sánchez-Esguevillas, and J. Lloret. Artificial neural networks for short-term load forecasting in microgrids environment. *Energy*, 75:252–264, 2014.

[10] L. Hernandez, C. Baladron, J. M. Aguiar, B. Carro, A. J. Sanchez-Esguevillas, J. Lloret, and J. Massana. A survey on electric power demand forecasting: future trends in smart grids, microgrids and smart buildings. *IEEE Communications Surveys & Tutorials*, 16(3):1460–1495, 2014.

[11] S. Iyengar, S. Kalra, A. Ghosh, D. Irwin, P. Shenoy, and B. Marlin. iprogram: Inferring smart schedules for dumb thermostats. In *Proceedings of the 2Nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, BuildSys '15, pages 211–220, New York, NY, USA, 2015. ACM.

[12] M. Jin, R. Jia, Z. Kang, I. C. Konstantakopoulos, and C. J. Spanos. Presencesense: Zero-training algorithm for individual presence detection based on power monitoring. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pages 1–10, New York, NY, USA, 2014. ACM.

[13] N. Klingensmith, A. Sridhar, Z. LaVallee, and S. Banerjee. Water or slime?: A platform for automating water treatment systems. In *Proceedings of the 2Nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, BuildSys '15, pages 75–84, New York, NY, USA, 2015. ACM.

[14] N. Klingensmith, D. Willis, and S. Banerjee. A distributed energy monitoring and analytics platform and its use cases. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys'13, pages 36:1–36:2, New York, NY, USA, 2013. ACM.

[15] A. V. Oppenheim and R. W. Schafer. *Discrete-time signal processing*. Pearson Higher Education, 2010.

[16] D. Pisharoty, R. Yang, M. W. Newman, and K. Whitehouse. Thermocoach: Reducing home energy consumption with personalized thermostat recommendations. In *Proceedings of the 2Nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, BuildSys '15, pages 201–210, New York, NY, USA, 2015. ACM.

[17] J. Ranjan, E. Griffiths, and K. Whitehouse. Discerning electrical and water usage by individuals in homes. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pages 20–29, New York, NY, USA, 2014. ACM.

[18] J. Scott, A. Bernheim Brush, J. Krumm, B. Meyers, M. Hazas, S. Hodges, and N. Villar. Preheat: Controlling home heating using occupancy prediction. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, UbiComp '11, pages 281–290, New York, NY, USA, 2011. ACM.