Sensor Network Data Fault Detection using Hierarchical Bayesian Space-Time Modeling

Kevin Ni and Greg Pottie

Abstract

We present a new application of hierarchical Bayesian space-time (HBST) modeling: data fault detection in sensor networks primarily used in environmental monitoring situations. To show the effectiveness of HBST modeling, we develop a rudimentary tagging system to mark data that does not fit with given models. Using this, we compare HBST modeling against first order linear autoregressive (AR) modeling, which is a commonly used alternative due to its simplicity. We show that while HBST is more complex, it is much more accurate than linear AR modeling as evidenced in greatly reduced false detection rates while maintaining similar, if not better detection rates. HBST modeling reduces false detection rates 41.5% to 96.5% when paired with our simple fault detection method. We also see that HBST modeling is more robust to model mismatches and unmodeled dynamics than linear AR modeling.

I. Introduction

Sensor networks in environmental monitoring have matured significantly since their inception in [1] and [2]. With this maturity comes an increased focus on ensuring data quality for scientific use. The detection of faulty sensors is key in maintaining data integrity throughout the sensor network. Once a sensor is discovered to produce anomalous and likely faulty data, quick action may then be taken to discover the cause of the fault and perhaps repair or replace a sensor. If a faulty sensor is not replaced or repaired, then the amount of usable data over the lifetime of a sensor deployment is reduced significantly. Hence, reliable detection of faults is vital for maintaining data yield during a sensor network deployment.

We utilize the hierarchical Bayesian space-time (HBST) modeling found in [3] and apply it to the problem of fault detection. While HBST modeling is much more complex than techniques such as linear autoregressive (AR) modeling, we validate its use in fault detection by applying HBST modeling to both simulated data and real data collected from two different experiments. While not perfect, HBST modeling is more accurate and robust than linear AR modeling to unmodeled dynamics.

Several previous experiments and deployments exhibited difficulties in collecting sensor data [4] [5] [6] [7]. While creating a simple to use sensor network application, [5] reports that accurate sensor data is difficult to obtain. As an example, the system developed in [5] was deployed in [6] with the goal of examining the microclimate over the volume of a redwood tree. The authors discovered that there were many data anomalies that needed to be discarded post deployment, thereby reducing the amount of usable data. Additionally, [7] evaluates a sensor network in a volcano monitoring environment with high data quality requirements as measured by yield and data fidelity, concluding that sensor networks must still improve. Post deployment analysis of a wireless sensor network in [4] exposed many network packet losses in addition to several node level data problems.

[8] uses models of real-world processes based on sensor readings to answer queries to a sensor network for data. Using time-varying multivariate Gaussians to model data, the authors respond to a predetermined set of query types, treating the sensor network like a database. To some extent this shields the user from faulty sensors. However, the authors point out that more complex models should be used to detect faulty sensors and give reliable data in the presence of faults.

The field of fault detection techniques in sensor networks has been growing as the use of sensor networks gains traction, but none have used HBST modeling to model the expected behavior of sensor data. Sensor network faults commonly seen in environmental monitoring situations are detailed in [9] and [10]. [10] presents a few fault types and evaluates different approaches to detect these faults. [9] lays out modeling issues, features and indications of sensor data faults emphasizing that modeling expected behavior is crucial in detecting faults. Taking advantage of expected spatial and temporal correlations is a key component of effective detection systems, but this is not easily exploited. We will apply HBST modeling to detect faults more effectively than existing methods.

[11] discusses a cross-validation method of detecting in the presence of faults using a minimization of multiple unspecified sensor fusion functions. This requires heavy calculations with each additional datum. [12] uses a basic approach by dividing samples into temporal granules and co-located sensors into spatial granules. Each granule is then expected to be measuring similar data and anything outside of this is considered to be a fault. [13] uses linear autoregressive models to characterize data for error correction, targeting transient “soft” failures. With these linear models, the authors develop a predictive error correction technique using current data to decide past data. We will compare the traditional linear autoregressive models with the new application of HBST to fault detection.

Bayesian techniques are not new in the fault detection application. In [14], Bayesian updating for the distribution of an individual sensor’s readings using prior distributions is employed. However, the prior

knowledge of the phenomenon that is to provide the prior distribution is not given in detail, and the method does not explicitly take advantage of any spatial or temporal modeling. [15] uses spatio-temporal correlations to learn contextual information statistically. This contextual information is used in a Bayesian framework to detect faults. [16] uses Bayesian techniques to select a subset of trusted sensors with which to compare other sensors. Assumptions on the correlation between sensors and across time are made limiting the effectiveness of the Bayesian selection method. The limited ability of linear models in [16] hamstring the overall method of the selection of a trusted subset of sensors.

However none of the Bayesian fault detection methods are used to directly model the data and learn the parameters of the data model. We will show that the use of better modeling by way of HBST modeling improves the performance of the system. We will use the modeling framework for hierarchical Bayesian space-time models of [3] which is used to model space-time data "ubiquitous in the environmental sciences." This introduces better spatial and temporal modeling than has been previously utilized in this application. To get around the issue of the complexity and computation costs involved with Bayesian modeling, we will show how such modeling can be used in a "semi-realtime" fault detection scheme to improve fault detection results.

In section II, we discuss some preliminaries before defining our HBST model. We define the system setup and assumptions for which we develop our approach. We also detail how we synchronize sensor network data for use in spatial statistics tools using a binning approach. Section III first presents the model used in [3]. Then, we adapt this model for its new application to fault detection and detail specific assumptions on the model structure. In section IV we discuss how we determine the parameters of this HBST model using Bayesian estimation with Markov Chain Monte Carlo (MCMC) methods and Gibbs sampling. Section V presents the simple fault tagging method that we use to compare HBST modeling with linear AR models.

Results are presented in section VI, where we apply HBST and linear AR modeling to simulated data and real data. The results show that HBST modeling in comparison to linear AR modeling excels at reducing false detection while maintaining good detection rates. Further discussion about the advantages and disadvantages of HBST modeling for use in fault detection is in section VII. Following the conclusion (section VIII), we include an appendix where full conditional distributions for use in the Gibbs sampler are derived.

II. System model and setup

The sensor network system setup, context, and application significantly influences how one should model the target phenomenon. For demonstrative purposes, we will model temperature data as this is one of the most common measurements. We will apply our method to two sets of real world data to show the versatility. The first set measures cold air drainage across a canyon in James Reserve, California. The second set of data is temperature at the surface of Lake Fulmor, in the James Reserve. In both of these scenarios, sensors are deployed in a close to linear manner, therefore the spatial dimension we will use will consist of only one axis.

As is currently done in most sensor deployments, all sensor data is forwarded to a central fusion center without modification. We assume that corrupted or missing data communication packets are simply unavailable data points which have no bearing on data faults. The fusion center will perform relevant modeling computations and make decisions regarding faults of individual sensors.

Additionally, real world data from all sensors are not usually synchronized so the data arriving to the fusion center cannot be easily vectorized. Most common space-time statistical tools assume that samples occur at regularly defined time intervals in a synchronized manner so that they may be easily placed in vectors at each time instant. In order to adapt real world data to such a scheme we "bin" the data by time instances for each sensor.

For one sensor, examining the regularly defined time instant at time t_i, we look at the interval surrounding this which is of the size r, where r is the difference between time instances t_i and t_{i+1}. If within the interval $t_i - \frac{r}{2}$ and $t_i + \frac{r}{2}$ there is one sensor value for this sensor then the output at time t_i for this sensor is exactly this sensor value. If there are multiple sensor values within this interval, then the output at time t_i is the mean of all these values. However, if there is no data point, then a line between the two nearest surrounding data points is used to interpolate all values in between.

This process requires three assumptions. First, there are no large gaps in sensor data, otherwise interpolation will fail to effectively capture data. Second, along these lines, when there is a gap in data, a linear interpolation between data is effective and accurate. Finally, depending on the time scale between the regular intervals, the process variation between time instances is insignificant so that interpolation is accurate.

To show that this process is effective, we examined the energy difference between a binned data set and the original data by calculating the area under the curve of the two sets. We tested the cold air drainage data and from day to day. On average, the percentage difference between the areas is insignificant. For

example, we pick one node to illustrate. Over the course of four days, the day to day difference in area averages an insignificant 0.27%. Visually there is little difference from the original data set and the binned set, as shown in Figure 1.

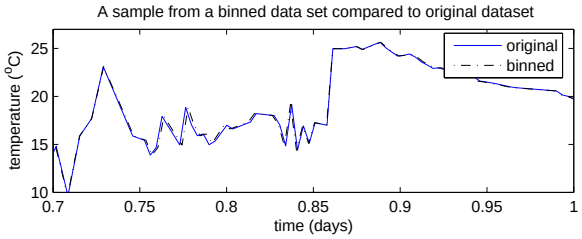


Fig. 1. Sample data and a binned version. This figure focuses on a small portion of the data to show that binning has no real effect on any analysis.

III. Hierarchical Bayesian Space-Time Model

We first provide an overview of the method presented in [3], as the adaptation of this approach for fault detection is the basis of our work. Wikle et. al. detail a flexible hierarchical model for space-time data using a multi-stage approach. We use this approach as a guideline to model sensor network data for the purposes of fault detection and make modifications to fit our needs. The flexibility, robustness, and systematic approach of this method makes it suitable for fault detection. Also its direct application to modeling climate data is ideal for the environmental monitoring that sensor networks perform.

In [3], the hierarchical space-time model consists of five stages of modeling. In the first stage, a statistical measurement error model is defined. Assuming $Y(s,t)$ is the process for sensor s at a location l_s and time t, then the observed (measured) data $Z(s,t)$ is distributed by some error distribution $P(Z(s,t)|Y(s,t),\theta_1)$ where θ_1 is a collection of parameters for the distribution. The next stage models the process Y. Based on the relevant processes of interest in [3], Y consists of several components. These components are a site-specific mean $\mu(s)$, a large-scale temporal model with site specific parameters $M(t;\beta(s))$, a short-time scale dynamic process $X(s,t)$, and a zero mean random variable that models noise $\gamma(s,t)$. The specification of the joint distribution of $\gamma(s,t)$ is simplified to avoid modeling a $ST \times ST$ covariance matrix. The hierarchical approach allows this since other modeled features of $Y(s,t)$ will explain the space-time structure, X(s,t).

The third stage defines these spatial structures and dynamics for the Y process. In the example presented in [3], μ is defined to be a Markov random field, and X is modeled as a one step space-time autoregressive

moving average. We will deviate from these assumptions in our own modeling. The fourth stage defines the prior distributions on the model parameters. The fifth stage defines hyperprior distributions on the prior parameters of the fourth stage. With simplifications made in our modeling, we do not define any hyperprior distributions.

[3] presents an example in which this approach is applied to monthly averaged maximum temperature data in the midwestern United States. For the application of sensor data fault detection we make several adjustments and deviate from the example presented. Also, because of the type of system as well as the much smaller scale we are observing, we detail further restrictions on the data when defining our model.

Given at time t a set of observations from S sensors, Z_t is a $S \times 1$ vector of the observations. We begin by modeling the measurement process, Z_t as simply the phenomenon process with additive noise, ϵ_Z.

$$Z_t = Y_t + \epsilon_Z$$

Assuming the measurements $Z_t(s) \forall\, s = 1, \ldots, S$ are all independent and the noise is normal, then we represent Z_t as:

$$Z_t|\{Y_t, \sigma_Z^2\} \sim \mathcal{N}(Y_t, \sigma_Z^2 \mathbf{I}) \tag{1}$$

The phenomenon process can be modeled as a combination of three main components and the noise component ϵ_Y. This noise component requires the assumption that the noise $\gamma(s,t)$ is normal and independent and identically distributed for all Y_t.

$$Y_t = \mu + M_t + X_t + \epsilon_Y$$

As in [3] we will assume that all $Y_t(s)$ are normally distributed and conditionally independent such that:

$$Y_t|\{\mu, M_t, X_t, \sigma_Y^2\} \sim \mathcal{N}(\mu + M_t + X_t, \sigma_Y^2 \mathbf{I}) \tag{2}$$

The spatial structures and dynamics consist of site specific means μ, a "long term" trend M_t, and a time dynamic process X_t accounting for day to day variations.

We make several departures from [3] in how these spatial structures are modeled and defined in order to decrease complexity and also to better match our system. Instead of defining a Markov random field, in order to decrease run time, we first define the site specific mean to be a simple first order spatial regression:

$$\mu(s) = \mu_1 + \mu_2 l_s$$

where l_s is the physical position of sensor s. μ_1 in this case is the overall mean of the phenomenon and μ_2 represents small corrections according to spatial trends. If there is no strong spatial trend, or the

trend is not linear along s then μ_2 will tend to zero and the site specific means will tend to the overall phenomenon mean at μ_1. This will make the system more robust when a linear model is not accurate.

These two parameters of μ are modeled as independent normal random variables with fixed and specified priors.

$$\mu_1 \sim \mathcal{N}(\bar{\mu_1}, \sigma^2_{\mu_1}) \tag{3}$$

$$\mu_2 \sim \mathcal{N}(\bar{\mu_2}, \sigma^2_{\mu_2}) \tag{4}$$

We model the "long term" trend as a daily harmonic with spatially varying amplitudes and phases with an additional linear trend:

$$M_t(s) = (f_1 + f_2 l_s)\cos(\omega t) + (g_1 + g_2 l_s)\sin(\omega t) + h_1 t$$

where $\omega = 2\pi$ for a daily harmonic (when t is defined in units of days). f_1,f_2,g_1,g_2 define how the harmonic varies spatially. We add the h_1 term to account for the day to day weather trend over the modeling window; this is different from [3] as their long term trend is annual which has no year to year trend.

We assume all of the parameters in M_t to be independent normal random variables with fixed and specified priors.

$$f_1 \sim \mathcal{N}(\bar{f_1}, \sigma^2_{f_1}) \tag{5}$$

$$f_2 \sim \mathcal{N}(\bar{f_2}, \sigma^2_{f_2}) \tag{6}$$

$$g_1 \sim \mathcal{N}(\bar{g_1}, \sigma^2_{g_1}) \tag{7}$$

$$g_2 \sim \mathcal{N}(\bar{g_2}, \sigma^2_{g_2}) \tag{8}$$

$$h_1 \sim \mathcal{N}(\bar{h_1}, \sigma^2_{h_1}) \tag{9}$$

We model the time dynamic term as a "diagonal" vector autoregressive process:

$$X_t = \mathbf{H}X_{t-1} + \epsilon_X \tag{10}$$

where

$$\mathbf{H} = a\mathbf{I}$$

giving

$$X_t | \{X_{t-1}, \mathbf{H}, \sigma_X\} \sim \mathcal{N}(\mathbf{H}X_{t-1}, \sigma^2_X \mathbf{I}) \tag{11}$$

We assume that a is the same for all locations and it is normally distributed:

$$a \sim \mathcal{N}(\bar{a}, \sigma_a^2) \tag{12}$$

Note that the description for X_t is much simpler than in [3] in order to decrease run time and speed convergence of the Gibbs sampler. We will discuss the simulation using Gibbs sampling in section IV. Adding off-diagonals and allowing elements in $\mathbf{H}$ to vary quickly transforms equation 10 into a space-time autoregressive moving average (STARMA) model, see [17]. This increases complexity and the resultant model is over-parameterized and sensitive to the initial conditions in our case. Using such a model also requires us to restrict sensors to be fixed or assigned to a regularly spaced lattice position which is not commonly true.

We specify the variances of the X, Y and Z to have an inverse gamma distribution, which is the conjugate prior to the normal distribution:

$$\sigma_Z^2 \sim \Gamma^{-1}(\alpha_Z, \beta_Z) \tag{13}$$

$$\sigma_Y^2 \sim \Gamma^{-1}(\alpha_Y, \beta_Y) \tag{14}$$

$$\sigma_X^2 \sim \Gamma^{-1}(\alpha_X, \beta_X) \tag{15}$$

The prior parameters of these inverse gamma distributions are fixed and specified.

IV. Model simulation

Once this model has been established and given the data collected over a period of time, we determine the parameters of this model using Bayesian estimation. To do this we use Markov chain Monte Carlo methods, and more specifically the Gibbs sampler, for stochastic simulation [18]. Instead of drawing samples of all the parameters from one massive and difficult to calculate joint distribution, Gibbs sampling draws subsets of parameters conditioned on the value of the other parameters. This allows for quicker computation and simple derivations of conditional distributions. The derivations of the conditional distributions for our model for the Gibbs sampler are provided in the Appendix.

Choices need to be made for the starting point values and the length of the simulation to run. More information on discarding and thinning sections of a Gibbs sampling run can be found in [18].

As is required when using Gibbs samplers, we tested on several real world data sets with a few initial pilot simulations using different starting value sets. One of the starting value sets was the estimated means of the parameters from exploratory analysis. Visual assessment of convergence was seen to appear by 4000 iterations for all cases. Thus for use in our algorithm, the parameters were estimated using a single

long simulation (10000 iterations) with the estimated mean value starting sets. We discard the first half of the data where the sequence is converging.

The final result of the simulation is a number of random draws for each of the parameters as well as the X_t and Y_t dynamic processes. With these, we can then apply a fault detection method.

V. Fault detection

The primary weakness of HBST modeling is that the posterior simulation of the model parameters using MCMC techniques and Gibbs sampling is computationally expensive. Thus, we seek to minimize the frequency that we calculate parameters by specifying a semi-realtime detection system. By having this semi-realtime system, we can exploit the capabilities of HBST modeling while minimizing the impact of the high computation cost. Instead of performing calculations with each new incoming data value as is done in systems such as the ones in [16], [11], and [13], calculations are to be performed at regular time intervals at a time scale larger than the sensing intervals.

That is, sensor data integrity audits occur much less frequently than sensor samples are taken. For example, while the sensor data used in this paper measures the phenomenon on a scale of every 5 minutes, we will audit sensors every one day. This reflects logistical realities, in that it is unlikely for sensor replacement to be on the sensing time scale in the environmental sensing context, e.g. a person would likely wait for the next day to replace a sensor that failed while they were sleeping that night. Also it is common for a sensor to temporarily report questionable data and then return to normal [9]. Therefore, by having the audit occur at larger intervals, a sensor that returns to normal operating conditions will not be as frequently tagged.

To test the abilities of HBST modeling, our goal is to tag data from sensors which are believed to be behaving outside modeled behavior. With this tagging, one can use the results in a more complex memory based method for fault identification. For example, the rate at which a sensor is tagged may be thresholded and identified as cooperative or non-cooperative and used in the reputation based framework as described in [19]. Alternatively, the tag rate may be used as a prior in a Bayesian decision method to select a subset of trusted sensors such as in [16]. Going further, a Bayesian network [20] may be implemented with the tagging rate influencing the probabilities. However all of these possibilities are beyond the scope of this paper. When paired with more sophisticated fault detection methods, such as those in [16] and [10], HBST modeling can boost performance.

We use a simplistic thresholding technique based upon nearby sensors for detection. Consider a single data point for sensor s. For sensors $s+1$ and $s-1$ we calculate 95% confidence intervals of the time

dynamic term $X(s_{i\pm1}, t)$ using the variance, $\hat{\sigma}_X^2$. We denote $\hat{}$ to be the sample mean across all samples from the simulated posterior parameters. We add these to the site specific mean and estimated long term trend for sensor s. Effectively, we assume that time dynamic X_t for sensor s, which is without any spatial effects as those are carried in M_t and μ, is similar for surrounding sensors. This bounds the space time dynamic performance of each sensor by the adjacent sensor's worst case performance, then adds these bounds to the site specific mean and long term trend. In this way, sensor s is assumed, at worst, to report location adjusted values of its neighboring sensors. We will see how this assumption holds in the results.

That is, if we consider sensor s at location l_s at a particular time time t, we define the lower bound and upper bounds of the expected time dynamic term to be:

$$X_l(s,t) = \min(\hat{X}(s-1,t) - 2\hat{\sigma}_X, \hat{X}(s+1,t) - 2\hat{\sigma}_X)$$

$$X_u(s,t) = \max(\hat{X}(s-1,t) + 2\hat{\sigma}_X, \hat{X}(s+1,t) + 2\hat{\sigma}_X)$$

Then we use the estimated terms for μ_1, μ_2, f_1, f_2, g_1, g_2, and h_1 to calculate:

$$\widetilde{\mu}(s) = \hat{\mu}_1 + \hat{\mu_2} l_s$$

$$\widetilde{M_t}(s) = (\hat{f}_1 + \hat{f}_2 l_s) cos(\omega t) + (\hat{g}_1 + \hat{g}_2 l_s) sin(\omega t) + \hat{h}_1 t$$

Finally, the lower and upper bounds are:

$$Z_l(s,t) = \widetilde{\mu}(s) + \widetilde{M_t}(s) + X_l(s,t) - 2(\hat{\sigma}_Y + \hat{\sigma}_Z)$$

$$Z_u(s,t) = \widetilde{\mu}(s) + \widetilde{M_t}(s) + X_u(s,t) + 2(\hat{\sigma}_Y + \hat{\sigma}_Z)$$

We extend the bounds using the estimated standard deviations of the phenomenon and measurement processes, $(\hat{\sigma}_Y + \hat{\sigma}_Z)$, because we compare to measured data $Z(s,t)$. If $Z(s,t)$, the actual measurement from sensor s at time t, exceeds these bounds, then it is marked as faulty.

In our results we compare this approach to modeling with an analogous method without HBST modeling. The basis of the analogous method is using first order linear autoregressive (AR) models over a window of the previous data. Examples of linear AR modeling in fault detection can be seen in [16] and [13], where both use first order linear models. Also [12] and [21] smooth data using a moving average window resulting in an expected mean, which is a less complex operation than the linear modeling done in the previously mentioned works.

For the comparison modeling technique, we create bounds similar to the HBST modeling case. The estimate of the standard deviation is derived from the linear AR model for each individual sensor. For

sensor s, two surrounding sensors' readings and standard deviations are used to provide a lower and upper bound. Similar tagging is used to identify readings that exceed these bounds.

When using the conventional method, a decision must be made on the size of the window used to calculate the linear model. This window size, as discussed in [16] affects the quality of the fit to the data. For simplicity, we fix this window size to 25 samples because after trial and error this produces the best results in most cases for our simulated data set.

In interpreting the results, we use two measurement metrics: detection rate, and false detection rate. We expect that better modeling will decrease the false detection rate since a well modeled system will have less anomalies. Detection rates are expected to remain similar because questionable data should still be outside of the range of any reasonable model.

This simplistic way of bounding data by neighboring sensors' worst case performances has an additional drawback in the cases of edge sensors. Sensors on the edge are only influenced by one other sensor, greatly reducing the bounds. So it is expected that edge sensors have a higher false detection rate than non-edge sensors. However, as we will see in the results, since HBST modeling adjusts for spatial differences and trends, edge false detection is reduced significantly in comparison to AR modeling.

VI. Results

To show the applicability of HBST modeling to multiple situations, we demonstrate our method using three separate data sets. One data set is artificially generated and used as a toy example to illustrate under ideal conditions the performance of our system. The second data set is the cold air drainage data set from sensors that have been deployed at James Reserve in California. The last set of data is from a series of buoys deployed at Lake Fulmor, also at James Reserve. For this last set of data, we use the temperature measurements that are at the surface of the water.

A. Simulated Data

We use simulated data to show the expected results from both HBST and AR modeling. Spatial structure is well defined and matches very well to the assumptions made in our fault tagging scheme. Simulated data also highlights some of the limitations of our simple tagging scheme for nodes on the edge of the sensing field. We show results from data with no faults as well as injected faults to show the best performance of each system.

The simulated data consists of a daily diurnal long term trend, as well as an extra harmonic that was not modeled by the HBST model we defined in section III. Spatial structure was generated using a

TABLE I

FALSE DETECTION RATES FOR SIMULATED DATA WITH NO FAULTS

	HBST	**Linear AR**
Including Edge Nodes	0.2079	0.2784
Excluding Edge Nodes	0.0014	0.0041
Just Edge Nodes	0.6210	0.8270

similar model to that of section III, by including a spatial trend on the site specific mean and harmonic parameters. Parameters were fixed to rough estimates derived from actual data. We generated simulated data for six sensors all equally spaced. A sample from three sensors over three days is in figure 2. We

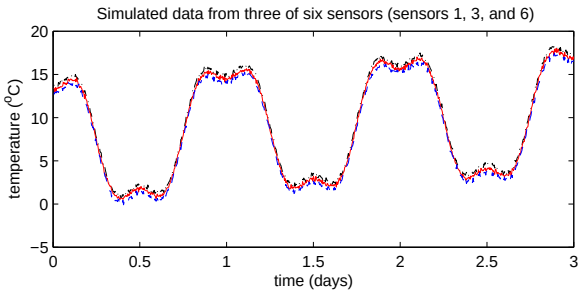


Fig. 2. Simulated data. A sample of three days from three sensors.

apply the fault tagging techniques described in section V to get a baseline for expectations. We average the tag rates for each sensor over three days, and present the overall false detection rates in table I. Also, we show the false detection rate for just the edge cases to show the increased tag rate for edge cases. As noted before, the edge cases show much higher false detection rates.

Examining the results of table I, edge nodes have a much lower false detection rate using HBST modeling than linear AR modeling. This is expected because the HBST modeling approach is capable of modeling and correcting for spatial trends. Overall, in all cases the HBST modeling approach shows significant reductions in false detections. When including edge nodes, HBST modeling gives a 25.3% improvement over linear AR. More significantly, when edge nodes are excluded, HBST modeling gives a 64.9% improvement over linear AR.

To test detection capabilities, one day was selected to have faults injected, and we tested the detection of each fault independently. We inject two types of common faults as defined in [10] and [9] at arbitrary

locations. In figure 3 we show three sensors, two with faults, and one with no faults. One sensor has a “stuck-at” fault injected, and the other has outliers.

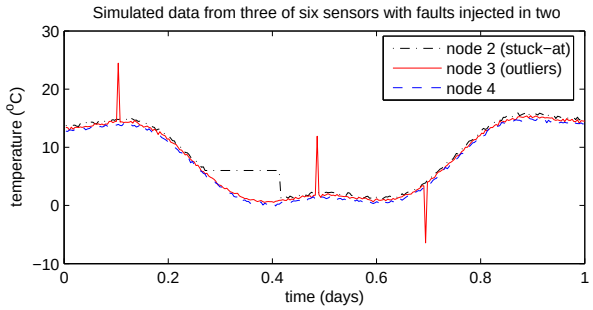


Fig. 3. Simulated data with injected faults.

The results for HBST modeling and linear AR modeling are shown in Figure 4. Outlier detection worked perfectly for both Linear AR modeling and HBST modeling. However, The HBST modeling showed a 95.9% lower false detection in comparison to linear AR modeling. One reason the HBST modeling false detection is so low is because the time dynamic uncertainty is elevated in the presence of faults, and outliers seem to affect this variance more than other faults.

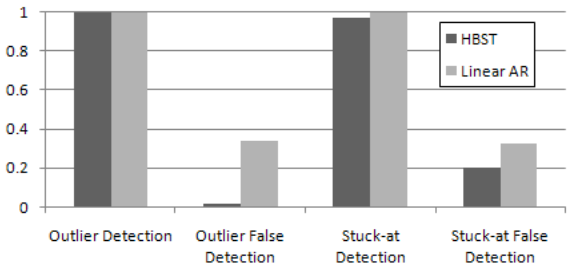


Fig. 4. Fault detection rates for simulated data with injected faults

For the case of the stuck at fault, detection was almost equal for both cases. Although the linear AR modeling performed slightly better, the HBST modeling approach only missed one sample, which is insignificant. More significant is that HBST modeling has a 36.8% lower false detection than linear AR modeling does.

The simulated data results show that HBST modeling is superior in reducing false detection rates in all cases. Detection capability remains virtually the same. Simulated data provides us with a baseline of

expected performance, and shows how the spatial modeling of HBST models is important in the edge nodes for reduction of the false detection rate.

B. Cold Air Drainage Data

Using real world data from a deployment, we examine how HBST modeling affects detection. First we examine the case where data does not exhibit any apparent errors. We examine the false detection rate of six sensors over the course of five days. Figure 5 shows data from the first three sensors starting on September 17, 2005. For the overall results, we look over the course of the 5 days in figure 5 and

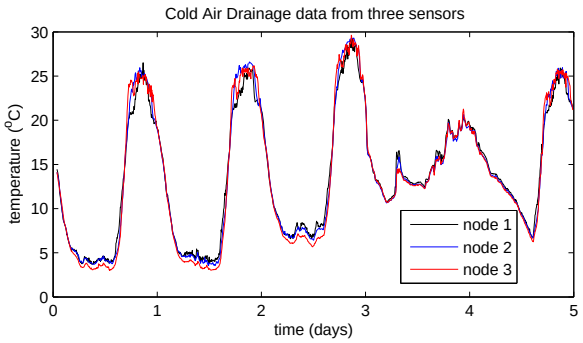


Fig. 5. Data from three deployed sensors

average the day to day tag rate.

The results are summarized in figure 6. HBST modeling gives a 42.7% lower false detection rate than linear AR modeling. Also, as expected, the edge nodes have a much higher false detection rate than the rest of the nodes. The edge cases have a false detection rate of 53.5% and 71.3% for the HBST modeling and linear AR modeling respectively. The HBST modeling is capable of reducing false detection for edge nodes due to the use of spatial means. When we exclude these values, the performance of the HBST modeling outperforms the linear AR modeling by 63.6%.

Deeper examination of the results shows that the HBST modeling tags data predominantly during the peak of the day, where the data is highly variable and dynamic. This is likely due to unmodeled phenomena. While our method is robust to unmodeled dynamics that are spatially correlated, these dynamics are not well correlated.

One possible cause of this is the passage of sunflecks where the sensor may be exposed to sun and shade alternatively due to the forest coverage, wind, clouds, and passage of time during the day; this causes temperature readings to rise and fall in unexpected ways. These sunflecks are highly dynamic

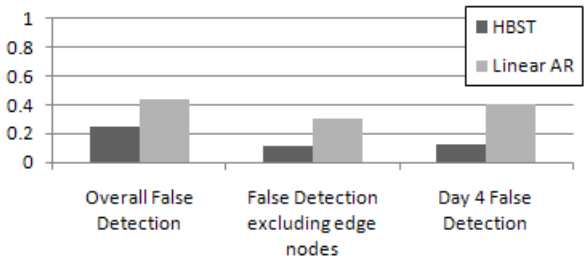


Fig. 6. False detection rates for cold air drainage data in the absence of faults

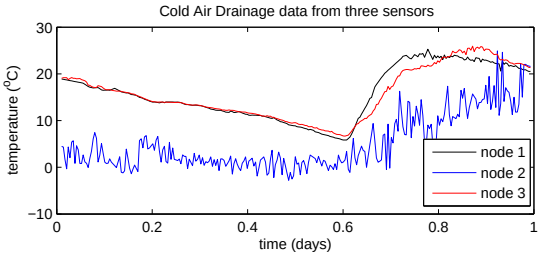
and very difficult to model accurately. More information regarding models of sunflecks and sunlight penetration through a forest canopy can be found in [22] and [23]. So, it is more likely for our modeling to fail in this dynamic period.

If one were to include a model for dynamics such as sunflecks, this would undoubtedly increase the performance of the fault detection system. However, this requires much more sophistication and will likely greatly increase the computation costs of the model. This is because each sensor node is different and will have different dynamics associated with them. Also, in order to model these dynamics, information regarding the forest canopy distance from the ground and the coverage the canopy provides given the time of day and the day of the year must be obtained through more detailed measurements.

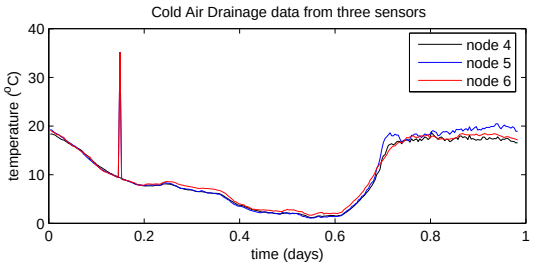
Looking across the days, we see that day 4 does not exhibit these highly variable peak temperatures, and the high temperature of the day is significantly lower than the other days. This suggests that the day may have been overcast or even rainy when sunflecks may not have existed.

If we examine only this day, then the overall false detection rate is greatly reduced when using HBST modeling. HBST modeling gives a 70.6% improvement over linear AR modeling. Linear AR modeling does not improve much since there are many correlated dynamics that are not modeled. HBST modeling is robust to unmodeled dynamics that are spatially correlated. It may be prudent to include prior knowledge in the form of daily weather patterns. If a day was noted to be overcast, then any judgments on sensor reliability may be given more weight than decisions on other days.

We now examine some examples of real data with questionable data that is assumed to be faulty. Figure 7(a) shows data from three sensors for one day, Sept. 25, 2005, with one sensor giving likely faulty data, with high noise and readings distant from other sensors. The other two sensors that are physically located around this sensor are also shown. Figure 7(b) shows data from three neighboring sensors on Sept. 16, 2005 where two independent neighboring sensors exhibit outliers at the same instant. There



(a) Data from a faulty sensor



(b) Data with an outlier

Fig. 7. Two examples of faults in real data

is no conclusive reason for why this happened, but it is important to tag such an anomaly. We test the detection rate for each fault with six sensors to model and examine the results summarized in figure 8. For the case of the faulty sensor in figure 7(a), both HBST modeling and linear AR modeling detect the

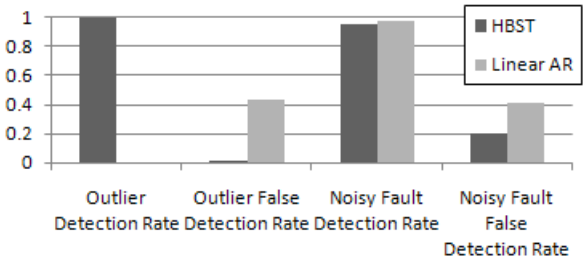


Fig. 8. Detection and false detection rates for cold air drainage data in the presence of faults

fault very well, exceeding 95% detection. However, the false detection rate for HBST modeling is 49.7% lower than linear AR modeling. For the case of the outlier fault in figure 7(b), HBST modeling detects the outliers perfectly while the linear AR modeling completely misses the outliers. The false detection rate is significantly lower for HBST modeling as well, giving a 96.5% lower rate.

These results show that HBST modeling is a significant improvement over linear AR modeling in

both accuracy and robustness. False detection rates are 42.7% to 96.5% lower using HBST modeling in comparison to linear AR modeling. As we will see in the next set of data, this gain is not limited to one specific deployment.

C. Lake Fulmor Data

To show our method can be adapted to multiple types of deployments, we present results from a second set of real data. We use temperature data collected at the surface from sensors deployed on buoys at Lake Fulmor in James Reserve between August 28th and September 1st, 2006. Figure 9 shows data from three of the five sensors used in this test. Nodes 2 and 3 display likely outliers at the beginning, while node 3 shows aberrant behavior starting at approximately day 2.65. This fault at the end of the data set is due to the battery failing on this particular node.

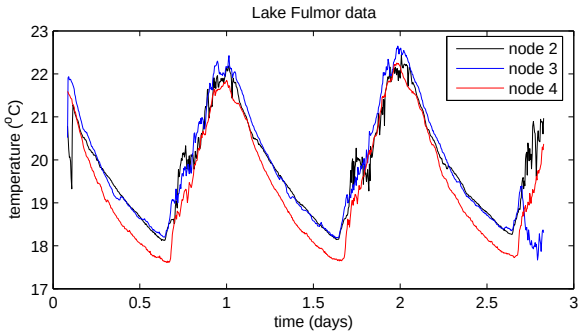


Fig. 9. Data from three buoys at lake Fulmor

Figure 10 presents a summary of the results. We initially exclude the faults from modeling to test the case where faults are not present. The results are similar to the results of the cold air drainage data set of section VI-B. HBST modeling reduces false detections by 44.4%.

The linear AR modeling method is unable to capture the fault at the beginning of the data set because the fault occurs during the delay before being able to tag data that linear AR models must have when starting up. The HBST modeling does correctly identify this outlier. Focusing on the fault for node 3 at the end of the data, we see that HBST modeling outperforms linear AR modeling as expected. HBST modeling has a 41.5% lower false detection rate while having a slight 3% advantage in detection.

These results show that the new application of hierarchical Bayesian space-time modeling can produce similar, if not better, detection rates of faults, while greatly reducing the false detections which are caused by poor modeling.

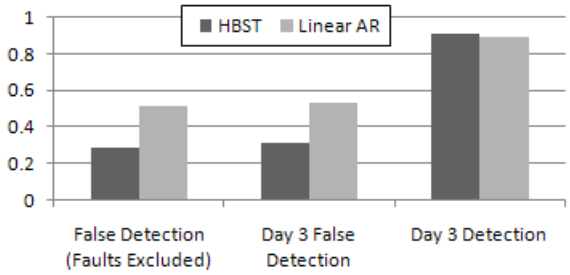


Fig. 10. Detection and false detection for Lake Fulmor data

VII. Discussion

While the performance of our fault detection method has room for improvement, the overall performance of the HBST modeling is much better than the case of standard linear AR models. However, there are trade-offs in accuracy, robustness, and computation where the linear AR modeling may have an advantage. There are different opportunities where each method may be used.

The first advantage of of linear AR models is that they are very simple. They are simple to understand and there are few parameters to determine given the data. On the other hand, the HBST model is much more complex with many more parameters for which we have to solve. It requires the use of posterior simulation techniques such as the Gibbs sampler used here, which in turn requires the derivation of full conditional posterior distributions.

A direct consequence of this is the computational cost. Once a window size is determined, linear AR modeling is computationally much cheaper than the HBST modeling as discussed in section V. Let I be the number of iterations used in Gibbs sampling, and let W be the size of the moving window for linear AR modeling. Asymptotically the performance for the HBST modeling is $O(ITS^3)$ FLOPs while the linear AR modeling performance is $O(W^3TS)$ FLOPs. However this is not descriptive because the HBST method has much more complex calculations that are performed many more times. For the computer we used, to model and process one day's worth of data for six sensors, the conventional method takes less than a half second, while the HBST modeling takes roughly seven to eight minutes.

The issue of window size selection in linear AR modeling may limit its computational advantage. The selection of a good window requires either good prior knowledge or retrospective analysis after acquiring a big data set. In section V, we performed several trials to determine the best window size to use for our data. Trial and error or other more systematic methods will increase the overall cost of linear AR

modeling.

One usually minor disadvantage to linear AR modeling, as we have presented here, is that linear AR modeling requires at least W samples available before it can begin to work. Thus there is a delay before linear AR modeling begins tagging data. This is usually not a big deal as data from a prior day is available to begin modeling. However in the case of the Lake Fulmor data, there is no data prior to the outliers seen, and as such linear AR modeling is unable to detect this fault.

The difference in accuracy of modeling can be seen in our results. Overall, the HBST modeling outperforms the linear AR modeling method. It has lower false detection rates which suggests better modeling capabilities. The linear AR modeling outperformed in the simulated data with injected "stuck-at" and noise faults. This is likely due to the fact that HBST modeling also models uncertainty more than the linear AR modeling. If the data exhibits higher variability throughout the day, then σ_X^2 will be higher because there is less certainty. This results in larger confidence intervals and lower detection rates. However, this type of uncertainty is not captured in linear AR modeling, and is apparent by the significant increase in false detection rate with real data.

Also contributing to the lack of accuracy for linear AR modeling is the fact that spatial structure is not used in modeling expected behavior. The only spatial relationship assumed is in our rudimentary tagging method. This lack of spatial modeling is most apparent in the edge cases where only one other sensor influences the tagging of an edge sensor. The HBST method is able to compensate for this and the standard linear AR modeling more than doubles the false detection rate in the edge cases.

HBST modeling is much more robust than linear AR modeling. Linear AR modeling is simple, but if data does not fit there is no correction made. However, if the model structure we assumed for μ or M_t is not accurate, the time-dynamic term, X_t will compensate for any difference between the assumed structure and the real data. The variability may increase in X_t as a result, but this will be captured in the σ_X^2 term.

It may be more useful for one to use linear AR modeling as a quick way of estimating parameters for priors in the HBST modeling. Also, HBST modeling may not be necessary in cases where sensor deployment is dense since there is likely to be less spatial variation. However, once priors are estimated or given then HBST modeling may be utilized to monitor the network. If the network is also sparsely deployed, then spatial structure is more important to estimate and utilize in fault detection. This is where HBST modeling holds the advantage.

VIII. Conclusion

We have presented a new approach to fault detection by modifying the existing hierarchical Bayesian space-time modeling technique of [3]. While it is much more complex than the first order linear AR modeling method, the results show that additional modeling greatly increases the performance of a fault detection system. It reduces the false detection significantly while maintaining detection ability. In some cases, it is more capable in detecting outliers. While we have paired our modeling with a simple fault tagging system, more complex systems that include historical behavior, as discussed previously, may produce bigger gains than we have seen with our simple system. We will develop such an algorithm to maximize the potential of this modeling for the future.

There are cases where our models break down, as in the case of the peak temperatures during the day time. In both linear AR modeling and HBST modeling, humans must be involved when there are unmodeled dynamics to identify whether or not the data is truly faulty. Additionally, linear AR modeling may require human involvement in the selection of window sizes. In the future, better modeling will be utilized in these cases to increase the performance.

Appendix

APPENDIX: Derivations of full conditional probability distributions

It is computationally efficient to use a Gibbs sampler to obtain draws from the joint posterior distribution. With the conditional independence assumptions afforded by the hierarchical model structure, we can easily derive the conditional distributions needed in the Gibbs sampler.

Many of the derivations are similar to those detailed in the appendix of [3] with some minor changes. Here we detail the derivations for the conditional distributions which have changed due to the modeling differences we make. The distributions that are unchanged (aside from notation) from [3] are $p(X_t|\cdot)$, $p(\sigma_X^2|\cdot)$, $p(\sigma_Y^2|\cdot)$, and $p(\sigma_Z^2|\cdot)$.

The derivations make use of Bayes rule:

$$\begin{aligned} p(A|B) &= \frac{p(B|A)p(A)}{p(B)} \\ &\propto p(B|A)p(A) \end{aligned}$$

Also, as in [3], a "completing the squares" method is extensively used in the derivations which we reproduce here. For a parameter vector θ, if the full conditional distribution is:

$$p(\theta|\cdot) \propto \exp(-\frac{1}{2}[\theta^T A\theta - 2B\theta])$$

then, after completing the square

$$\theta|\cdot \sim \mathcal{N}(A^{-1}B^T, A^{-1})$$

A. $p(Y_t|\cdot)$

From Bayes rule, we start with:

$$p(Y_t|\cdot) \propto p(Z_t|Y_t, \sigma_Z^2)\; p(Y_t|\mu, M_t, X_t, \sigma_Y^2)$$

Using the distributions for Z_t and Y_t as defined in equations 1 and 2 respectively, we get:

$$\begin{aligned} p(Y_t|\cdot) \quad \propto \quad & \exp\{-\frac{1}{2\sigma_Z^2}(Z_t - Y_t)^T(Z_t - Y_t)\} \\ & \times \exp\{-\frac{1}{2\sigma_Y^2}(Y_t - [\mu + M_t + X_t])^T \\ & \times (Y_t - [\mu + M_t + X_t])\} \end{aligned}$$

Define $B = \mu + M_t + X_t$.

$$\begin{aligned} p(Y_t|\cdot) \quad \propto \quad & \exp\{-\frac{1}{2\sigma_Z^2}(Z_t^T Z_t - Z_t^T Y_t - Y_t^T Z_t + Y_t^T Y_t) \\ & -\frac{1}{2\sigma_Y^2}(Y_t^T Y_t - Y_t^T B - B^T Y_t + B^T B)\} \end{aligned}$$

We drop the terms that do not involve Y_t as they can be extracted as constants for normalization.

$$\begin{aligned} p(Y_t|\cdot) \quad \propto \quad & \exp\{-\frac{1}{2}\, Y_t^T(\frac{1}{\sigma_Z^2}\mathbf{I} + \frac{1}{\sigma_Y^2}\mathbf{I})Y_t \\ & +(\frac{1}{\sigma_Z^2}Z_t^T + \frac{1}{\sigma_Y^2}B^T)Y_t\} \\ \propto \quad & \exp\{-\frac{1}{2}(Y_t^T(\frac{1}{\sigma_Z^2}\mathbf{I} + \frac{1}{\sigma_Y^2}\mathbf{I})Y_t \\ & -2(\frac{1}{\sigma_Z^2}Z_t^T + \frac{1}{\sigma_Y^2}B^T)Y_t)\} \end{aligned}$$

Thus:

$$\begin{aligned} Y_t|\cdot \quad \sim \quad & \mathcal{N}((\frac{1}{\sigma_Z^2}\mathbf{I} + \frac{1}{\sigma_Y^2}\mathbf{I})^{-1}(\frac{1}{\sigma_Z^2}Z_t^T + \frac{1}{\sigma_Y^2}B^T)^T, \\ & (\frac{1}{\sigma_Z^2}\mathbf{I} + \frac{1}{\sigma_Y^2}\mathbf{I})^{-1}) \end{aligned} \tag{16}$$

for all $t = 1, \ldots, T$.

B. $p(\mu_1, \mu_2|\cdot)$

Define the design matrix for all sensors s, $s = 1, \ldots, S$ with positions l_s to be:

$$\mathbf{P} = \begin{bmatrix} \vdots & \vdots \\ \mathbf{1} & l_s \\ \vdots & \vdots \end{bmatrix} \tag{17}$$

And define

$$\mu_L = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$$

such that $\mu = \mathbf{P}\mu_L$. Also define

$$\Sigma_\mu = \begin{bmatrix} \sigma^2_{\mu_1} & 0 \\ 0 & \sigma^2_{\mu_2} \end{bmatrix}$$

Furthermore, define $\bar{\mu_L} = [\bar{\mu_1}\ \bar{\mu_2}]^T$. From the formulations of the distributions for μ_1, μ_2, and Y_t from equations 3, 4, and 2 respectively:

$$\begin{aligned} p(\mu_L|\cdot) &\propto p(\mu_L|\bar{\mu_1}, \bar{\mu_2}, \sigma^2_{\mu_1}, \sigma^2_{\mu_2}) \prod_{t=1}^{T} p(Y_t|\mu, M_t, X_t, \sigma^2_y) \\ &\propto \exp\big(-\frac{1}{2}(\mu_L - \bar{\mu_L})^T \Sigma_\mu^{-1}(\mu_L - \bar{\mu_L})\big) \\ &\quad \times \exp\big(-\frac{1}{2\sigma^2_Y}\sum_{t=1}^{T}(Y_t - (\mathbf{P}\mu_L + M_t + X_t))^T \\ &\quad \times (Y_t - (\mathbf{P}\mu_L + M_t + X_t))\big) \\ &\propto \exp\big(-\frac{1}{2}(\mu_L^T(\Sigma_\mu^{-1} + \frac{1}{\sigma^2_Y}\sum_{t=1}^{T}\mathbf{P}^T\mathbf{P})\mu_L) \\ &\quad -2(\mu_L^T\Sigma_\mu^{-1} + \frac{1}{\sigma^2_Y}\sum_{t=1}^{T}(Y_t - M_t - X_t)^T\mathbf{P})\mu_L)\big) \end{aligned}$$

This gives:

$$\begin{aligned} \mu_L|\cdot \ &\sim \ \mathcal{N}((\Sigma_\mu^{-1} + \frac{T}{\sigma^2_Y}\mathbf{P}^T\mathbf{P})^{-1}(\mu_L^T\Sigma_\mu^{-1} \\ &\quad + \frac{1}{\sigma^2_Y}\sum_{t=1}^{T}(Y_t - M_t - X_t)^T\mathbf{P}), \\ &\quad (\Sigma_\mu^{-1} + \frac{T}{\sigma^2_Y}\mathbf{P}^T\mathbf{P})^{-1}) \end{aligned} \tag{18}$$

C. $p(f|\cdot)$, $p(g|\cdot)$, and $p(h|\cdot)$

These derivations follow closely to [3] with some changes to accommodate h_1. First we define $f_L = [f_1\ f_2]^T$, and similarly define $g_L = [g_1\ g_2]^T$. Then we can write M_t as:

$$M_t = \mathbf{P}f_L\cos(\omega t) + \mathbf{P}g_L\cos(\omega t) + h_1 t\vec{\mathbf{1}}$$

where $\mathbf{P}$ is the design matrix defined in equation 17.

Let $\bar{f_L} = [\bar{f_1}\ \bar{f_2}]^T$ and $\bar{g_L} = [\bar{g_1}\ \bar{g_2}]^T$. Also, define $\Sigma_f = [\sigma^2_{f_1}\ \sigma^2_{f_2}]\mathbf{I}$ and $\Sigma_g = [\sigma^2_{g_1}\ \sigma^2_{g_2}]\mathbf{I}$.

We first derive $p(f_L|\cdot)$:

$$\begin{aligned}
p(f_L|\cdot) &\propto p(f_L|\bar{f_L},\Sigma_f)\prod_{t=1}^{T} p(Y_t|\mu, X_t, f_L, g_L, h_1, \sigma^2_Y) \\
&\propto \exp(-\frac{1}{2}(f_L-\bar{f_L})^T\Sigma_f^{-1}(f_L-\bar{f_L})) \\
&\quad \times\exp\Big(-\frac{1}{2\sigma^2_Y}\sum_{t=1}^{T}[Y_t-(\mu+\mathbf{P}f_L\cos(\omega t) \\
&\quad +\mathbf{P}g_L\sin(\omega t)+h_1 t\vec{\mathbf{1}}+X_t)]^T[Y_t-(\mu \\
&\quad +\mathbf{P}f_L\cos(\omega t)+\mathbf{P}g_L\sin(\omega t)+h_1 t\vec{\mathbf{1}}+X_t)]\Big) \\
&\propto \exp\Big(-\frac{1}{2}(f_L^T(\Sigma_f^{-1}+\frac{1}{\sigma^2_Y}\sum_{t=1}^{T}\cos(\omega t)^2\mathbf{P}^T\mathbf{P}) \\
&\quad \times f_L - 2(\bar{f_L}^T\Sigma_f^{-1}+\frac{1}{\sigma^2_Y}\sum_{t=1}^{T}(Y_t-(\mu \\
&\quad +\mathbf{P}g_L\sin(\omega t)+h_1 t\vec{\mathbf{1}}+X_t))^T\mathbf{P}\cos(\omega t))f_L)\Big)
\end{aligned}$$

This gives:

$$\begin{aligned}
f_L|\cdot &\sim \mathcal{N}\Big((\Sigma_f^{-1}+\frac{1}{\sigma^2_Y}\sum_{t=1}^{T}\cos(\omega t)^2\mathbf{P}^T\mathbf{P})^{-1}(\bar{f_L}^T\Sigma_f^{-1} \\
&\quad +\frac{1}{\sigma^2_Y}\sum_{t=1}^{T}(Y_t-(\mu+\mathbf{P}g_L\sin(\omega t)+h_1 t\vec{\mathbf{1}}+X_t))^T \\
&\quad \times\mathbf{P}\cos(\omega t))^T,(\Sigma_f^{-1}+\frac{1}{2}\sum_{t=1}^{T}\cos(\omega t)^2\mathbf{P}^T\mathbf{P})^{-1}\Big)
\end{aligned}$$

Similarly for $p(g_L|\cdot)$ we have:

$$\begin{aligned}
g_L|\cdot &\sim \mathcal{N}\Big((\Sigma_g^{-1}+\frac{1}{\sigma^2_Y}\sum_{t=1}^{T}\sin(\omega t)^2\mathbf{P}^T\mathbf{P})^{-1}(\bar{g_L}^T\Sigma_g^{-1} \\
&\quad +\frac{1}{\sigma^2_Y}\sum_{t=1}^{T}(Y_t-(\mu+\mathbf{P}f_L\cos(\omega t)+h_1 t\vec{\mathbf{1}}+X_t))^T
\end{aligned}$$

$$\times \mathbf{P}\sin(\omega t))^T, (\Sigma_g^{-1} + \frac{1}{2}\sum_{t=1}^{T}\sin(\omega t)^2 \mathbf{P}^T\mathbf{P})^{-1}\Big)$$

And finally for $p(h_1|\cdot)$ we have:

$$\begin{aligned} h_1|\cdot \;\; \sim \;\; & \mathcal{N}\Big((\frac{1}{\sigma_{h_1}^2} + \frac{S}{\sigma_Y^2}\sum_{t=1}^{T} t^2)^{-1}(\frac{\bar{h_1}}{\sigma_{h_1}^2} + \frac{1}{\sigma_Y^2}\sum_{t=1}^{T}(Y_t \\ & -(\mu + \mathbf{P}f_L\cos(\omega t) + \mathbf{P}g_L\sin(\omega t) + X_t))^T\vec{\mathbf{1}}t) \\ & , (\frac{1}{\sigma_{h_1}^2} + \frac{S}{\sigma_Y^2}\sum_{t=1}^{T} t^2)^{-1}\Big) \end{aligned}$$

Where S is the number of sensors.

D. $p(a|\cdot)$

Recall:

$$\begin{aligned} \mathbf{H}X_{t-1} &= a\mathbf{I}X_{t-1} \\ &= aX_{t-1} \end{aligned}$$

Then:

$$\begin{aligned} p(a|\cdot) &\propto p(a|\bar{a}, \sigma_a^2)\prod_{t=1}^{T} p(X_t|X_{t-1}, a, \sigma_X^2) \\ &\propto \exp(-\frac{1}{2\sigma_a^2}(a-\bar{a})^2)\exp(-\frac{1}{2\sigma_X^2} \\ &\quad \times \sum_{t=1}^{T}(X_t - aX_{t-1})^T(X_t - aX_{t-1})) \\ &\propto \exp(-\frac{1}{2}(a^2(\frac{1}{\sigma_X^2}\sum_{t=1}^{T} X_{t-1}^T X_{t-1} + \frac{1}{\sigma_a^2}) \\ &\quad -2(\frac{1}{\sigma_X^2}\sum_{t=1}^{T} X_t^T X_{t-1} + \frac{\bar{a}}{\sigma_a^2})a)) \end{aligned}$$

Thus we have that:

$$\begin{aligned} a|\cdot \;\; \sim \;\; & \mathcal{N}\Big((\frac{1}{\sigma_X^2}\sum_{t=1}^{T} X_{t-1}^T X_{t-1} + \frac{1}{\sigma_a^2})^{-1} \\ & \times(\frac{1}{\sigma_X^2}\sum_{t=1}^{T} X_t^T X_{t-1} + \frac{\bar{a}}{\sigma_a^2}) \\ & , (\frac{1}{\sigma_X^2}\sum_{t=1}^{T} X_{t-1}^T X_{t-1} + \frac{1}{\sigma_a^2})^{-1}\Big) \end{aligned}$$

ACKNOWLEDGMENTS

This material is based upon work supported by the NSF under award #CNS-0520006. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

REFERENCES

[1] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, May 2000.

[2] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, Jun. 2001.

[3] C. K. Wikle, L. M. Berliner, and N. Cressie, "Hierarchical bayesian space-time models," *Environmental and Ecological Statistics*, vol. 5, no. 2, pp. 117–154, Feb. 1998.

[4] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *Proc. of the 1st European Workshop on Sensor Networks (EWSN)*, Jan. 2004.

[5] P. Buonadonna, D. Gay, J. M. Hellerstein, W. Hong, and S. Madden, "Task: Sensor network in a box," Intel Research Berkeley, Tech. Rep. IRB-TR-04-021, Jan. 2005.

[6] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *Proc. 3rd international conference on Embedded networked sensor systems (SenSys '05)*, 2005.

[7] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *7th USENIX Symposium on Operating System Design and Implementation*, Nov. 2006.

[8] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proc. of Very Large Databases*, 2004.

[9] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, G. Pottie, M. Hansen, , and M. Srivastava, "Sensor network data fault types," *ACM Transactions on Sensor Networks*, submitted for publication 2008.

[10] A. Sharma, L. Golubchik, and R. Govindan, "On the prevalence of sensor faults in real-world deployments," in *IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Jun. 2007.

[11] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli, "On-line fault detection of sensor measurements," in *Proc. of IEEE Sensors*, 2003.

[12] S. R. Jeffery, G. Alonso, M. J. Franklin, W. Hong, and J. Widom, "Declarative support for sensor data cleaning," in *4th International Conference on Pervasive Computing*, 2006.

[13] S. Mukhopadhyay, D. Panigrahi, and S. Dey, "Model based error correction for wireless sensor networks," in *Proc. Sensor and Ad Hoc Communications and Networks SECON 2004.*, Oct 2004, pp. 575–584.

[14] E. Elnahrawy and B. Nath, "Cleaning and querying noisy sensors," in *Proc. of International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2003.

[15] ——, "Context aware sensors," in *Proc. of the First European Workshop on Wireless Sensor Networks (EWSN 2004)*, Jan. 2004.

[16] K. Ni and G. Pottie, "Bayesian selection of non-faulty sensors," in *IEEE International Symposium on Information Theory*, Jun. 2007.

[17] N. A. C. Cressie, *Statistics for Spatial Data*. Wiley-Interscience, 1993.

[18] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, 2nd ed. Chapman & Hall/CRC, 2004.

[19] S. Ganeriwal and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," in *ACM workshop on Security in Ad-hoc & Sensor Networks (SASN) 2004*, Oct. 2004.

[20] D. Heckerman, "A tutorial on learning with bayesian networks," Microsoft Research, Tech. Rep. MSR-TR-95-06, Mar. 1995.

[21] M. Mourad and J.-L. Bertrand-Krajewski, "A method for automatic validation of long time series of data in urban hydrology," *Water Science & Technology*, vol. 45, no. 4–5, pp. 263–270, 2002.

[22] W. Smith, A. K. Knapp, and W. A. Reiners, "Penumbral effects on sunlight penetration in plant communities," *Ecology*, vol. 70, no. 6, pp. 1603–1609, 1989.

[23] J. Ross, M. Sulev, and P. Saarelaid, "Statistical treatment of the par variability and its application to willow coppice," *Agricultural and Forest Meteorology*, vol. 91, no. 1-2, pp. 1–21, 1998.