Towards data-centric control of sensor networks through Bayesian dynamic linear modelling

Lei Fang
School of Computer Science
University of St Andrews, UK
Email: lf28@st-andrews.ac.uk

Simon Dobson
School of Computer Science
University of St Andrews, UK
Email: simon.dobson@st-andrews.ac.uk

***Abstract*—Wireless sensor networks usually operate in dynamic, stochastic environments. While the behaviour of individual nodes is important, they are better seen as contributors to a larger mission, and managing the sensing quality and performance of these missions requires a range of online decisions to adapt to changing conditions. In this paper we propose an self-adaptive, self-managing and self-optimising sensing framework grounded in Bayesian dynamic linear models. Experimental results show that this solution can make sound scheduling decisions while also minimising energy usage.**

***Index Terms*—self-management, adaptive sampling, sensor networks, machine learning, energy efficiency**

I. Introduction

Wireless sensor networks (WSNs) offer several motivating challenges for self-organising systems. A typical objective of a sensor deployment is to sample an environment at a desired pace and send the observations back to a "sink" location. The whole process needs to take energy efficiency into account. Centralised control has been the dominating approach in the literature where either management decisions are made at the sink or some top-down co-ordination is used. Although these solutions have only lightweight local computation, they suffer from scalability problems and potentially a single point failure. More importantly, the power used in node-server communication might outweigh the power saved through minimising local computation.

A bottom-up, self-organising solution is therefore desirable. Ideally one would like local decisions to be made independently, with little (or no) communication overhead. A globally optimal result might not be achieved this way: but "good" results should be expected if sound local decisions are made.

As a data-driven application, sensor nodes need to understand a certain extent of the physics which they are tasked to measure in order to achieve well-founded local control. This is not always straightforward. Firstly, the physics is usually very hard to model. Specific models do exist for some applications, such as flooding or meteorology. However, these models are usually too complicated to be included at sensor level, and also too specific to be reusable in other more general settings. Secondly, the physics is fundamentally uncertain and is further confused by measurement error, outside perturbations that go unobserved, the (often unavoidable) degradation of sensor capabilities over time, and the impact of management decisions such as energy-saving on sensor fidelity.

Understanding and addressing these issues requires that we adopt a management approach that is grounded both in the engineering of sensor networks and in the behaviour being observed. One approach is to use machine learning to track observations and make control decisions in response. While machine learning is often too heavyweight to be deployed on compute-constrained nodes, there are approaches that are appropriate for such settings.

In this paper, we explore Bayesian Dynamic Linear Models (DLM) as an approach to local decision-making that can take place at an affordable cost. DLMs tailored to general sensor platforms together with the efficient model inference procedures are derived. The benefits of employing DLMs are their formal basis in Bayesian inference; their generality; their robustness to missing and perturbed observations; and their distribution efficiency that allows for purely local implementation. To put the model into use, we investigate the practical-sampling scheduling problem where each sensor node decides autonomously when to sample and how the data can be recovered. We show how DLM inference allows sensor nodes to decide the sampling time locally, and that a globally "good" result can be achieved out of this bottom-up, self-organising design: all the results are achieved without any form of centralised liaison. The paper concludes with further applications and benefits of using DLMs, and some notes for the future.

II. Related Work

Model based solutions are popular for WSNs control. The solutions employ formal or ad hoc models maintained at either the server side or in the network. Centralised modelling solutions, like [1], control the sensors by a server-side model. However, this category of solution suffers from scalability problem and single point failure. Another paradigm also called replicated models solution features keeping synchronised models both at the sink and network. PAQ [2], [3], for example, employs an autoregressive (AR) model for sensor data modelling. However, ARIMA model usually requires computational intensive learning phase; moreover, the learning cannot be done on-line. Low-Energy Adaptive Clustering Hierarchy (LEACH) [4] and Adaptive Sampling Approach to Data Collection (ASAP) [5] fall into the category of cluster-based in-network aggregation method. The cluster based solutions usually require some form of intra-cluster co-ordination, and the overhead on cluster head node could be significant.

Regarding sampling control, some similar solutions have been put forward. An decentralised adaptive sampling control method is proposed in [6] where Fisher Information and Gaussian process (GP) regression are used. The solution targets at a specific application with specialised hardware platform that can handle intensive learning task of GP. Other similar solutions include [7], where the authors present a utility-based adaptive sampling solution. A piecewise linear function is used to model the sensor data. But the model learning requires storing of the learning data locally, and the model update is not straightforward. A prediction-based geometric data stream monitoring solution is proposed in [8]. The solution aims at reducing sink-node communication. However, the solution targets at triggering event detection where a non-linear function of the distributed data source is of interest.

III. Bayesian Dynamic Modelling

Bayesian Dynamical Linear Models (DLMs) have been successfully applied in analysing financial time series [9], [10], and some others like medicine, ecology data [11], [12]. However, the close linkage between DLMs and sensor platforms has not been well discovered. In this section, the backbone of the proposed solution: DLMs is introduced with a focus on its connection with sensors.

A. Model Definition

Sensor data can be viewed as a collection of (multivariate) time series; let $\boldsymbol{y}_t^i$, a p-variate vector, denote the sensor data measured at time t by node i; then $\boldsymbol{Y}^i = \{\boldsymbol{y}_t^i, t \geq 1\}$ represents the time series collected by node i. A dynamic linear model is a probabilistic time series model which can be formulated as follows (node id i is dropped for convenience):

Definition 1 (Bayesian Dynamic Linear Model)**.**

$$\textit{Sensor model:}\quad \boldsymbol{y}_t = \boldsymbol{F}_t\boldsymbol{\theta}_t + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{\Sigma}_t) \tag{1a}$$

$$\textit{Process model:}\ \boldsymbol{\theta}_t = \boldsymbol{G}_t\boldsymbol{\theta}_{t-1} + \boldsymbol{w}_t, \boldsymbol{w}_t \sim \mathcal{N}_m(\boldsymbol{0}, \boldsymbol{\Omega}_t), \tag{1b}$$

together with a prior for $\boldsymbol{\theta}_0$

$$\boldsymbol{\theta}_0 \sim \mathcal{N}_m(\boldsymbol{m}_0, \boldsymbol{C}_0); \tag{1c}$$

where $\boldsymbol{G}_t$ *(of order* $m \times m$*) and* $\boldsymbol{F}_t$ *(*$p \times m$*) are scalar matrices,* $\mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *denotes a* p*-variate Gaussian distribution with mean and covariance matrix* $\boldsymbol{\mu}$ *and* $\boldsymbol{\Sigma}$ *respectively, and* $\boldsymbol{v}_t, \boldsymbol{w}_t$ *are independent to each other for all* t*.*

The dimensions of the relevant parameters are summarised in Table I.

TABLE I
Parameters of a DLM model

Parameters	Dimensions
$\boldsymbol{y}_t, \boldsymbol{f}_t, \boldsymbol{e}_t$	$p \times 1$
$\boldsymbol{m}_t, \boldsymbol{a}_t$	$m \times 1$
$\boldsymbol{C}_t, \boldsymbol{R}_t, \boldsymbol{W}, \boldsymbol{G}_t$	$m \times m$
$\boldsymbol{Q}_t, \boldsymbol{V}$	$p \times p$
$\boldsymbol{F}_t$	$p \times m$

A DLM can be completely specified by a quadruple:

$$\{\boldsymbol{F}_t, \boldsymbol{G}_t, \boldsymbol{\Sigma}_t, \boldsymbol{\Omega}_t\},$$

plus the initial prior parameters $\{\boldsymbol{m}_0, \boldsymbol{C}_0\}$ for $\boldsymbol{\theta}_0$, where $\boldsymbol{F}_t$ and $\boldsymbol{G}_t$ are usually fixed matrices that depend on the model of choice; however the two covariance matrices $\boldsymbol{\Sigma}_t$ and (especially) $\boldsymbol{\Omega}_t$, the unobserved state evolution variance, are unknown.

B. DLM for sensors (univariate)

A DLM model suits the sensor context particularly well. Equation (1b), which is a first-order Markovian stochastic process, can be actually viewed as the model for the hidden physical phenomenon of interest (for example temperature). The model assumes the physical process is stochastically evolving over the time and its current state depends on its previous state subject to some linear transformation $\boldsymbol{G}_t$ and some stochastic small changes $\boldsymbol{w}_t$. Intuitively, this process assumption is appropriate for most physical variables: for instance, the current temperature develops upon its previous state plus some small change either positive or negative. Equation (1a) represents the *sensor* or *observation model*, where sensor measurements $\boldsymbol{y}_t$ are formed by "observing" the hidden process $\boldsymbol{\theta}_t$, again plus some unbiased observation errors. Figure 1 shows the DLM definition graphically: the round nodes represent the hidden physical process whose value depends on its previous state; while the square nodes are the sensor observations of the corresponding hidden process.

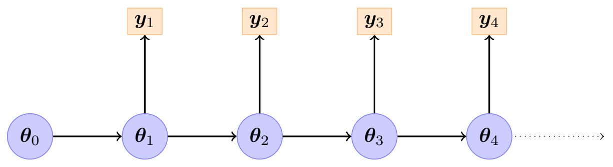


Fig. 1. Graphical representation of a DLM

1) process model assumption for sensors: The legitimacy of the process model can be further established as follows. Assume the physical process can be formed as a real-valued continuous function of time, say, $\phi(t)$. Note the assumption applies to most sensor settings. The process then can be modelled by setting $\boldsymbol{\theta}_t$ as an m-component random vector (with the corresponding $\boldsymbol{F}_t = (1, 0, \ldots, 0)'$ and $\boldsymbol{G}_t$ as a $m \times m$ upper triangular matrix of unit elements), leading to the *m-order polynomial trend* DLM. For example, a second-order polynomial DLM (also called local linear trend model, hereafter referred to as simply trend model) is formed with a hidden state $\boldsymbol{\theta}_t = (\mu_t, \beta_t)'$, and

$$y_t = \mu_t + v_t, \qquad v_t \sim \mathcal{N}(0, \sigma_v^2), \tag{2a}$$

$$\mu_t = \mu_{t-1} + \beta_{t-1} + w_{t,1}, \quad w_{t,1} \sim \mathcal{N}(0, \sigma_{w,1}^2), \tag{2b}$$

$$\beta_t = \beta_{t-1} + w_{t,2}, \qquad w_{t,2} \sim \mathcal{N}(0, \sigma_{w,2}^2), \tag{2c}$$

which is equivalent to a DLM with the following settings:

$$\boldsymbol{F}_t \coloneqq \boldsymbol{E} = [1 \quad 0], \quad \boldsymbol{G}_t \coloneqq \boldsymbol{L} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix},$$

and $\boldsymbol{w}_t = (w_{t,1}, w_{t,2})', \boldsymbol{\Omega}_t = \text{diag}(\sigma_{w,1}^2, \sigma_{w,2}^2)$. The model can be simplified by setting $\boldsymbol{\Omega}_t = \sigma_v^2 \cdot \text{diag}(W_{t,1}, W_{t,2})$, where $W_{t,i} = \sigma_{w,i}^2 / \sigma_v^2$ is the corresponding variance ratio. Define $\boldsymbol{W}_t = \text{diag}(W_{t,1}, W_{t,2})$, the corresponding model quadruple is

$$M_2 : [\boldsymbol{E}, \boldsymbol{L}, \sigma_v^2, \boldsymbol{W}_t]. \tag{2d}$$

Other DLMs with different orders can be formed similarly, although in real world data analysis only the first two orders are relevant. To see the connection between the model and the physical process $\phi(t)$, apply the Taylor expansion to $\phi(t)$ at time t_i: the function at k-unit time step forward can then be written as

$$\phi(t_i + k) \approx \phi(t_i) + \phi'(t_i)k + \ldots = \sum_{n=0}^{\infty} \tilde{\phi}_i^{(n)} k^n,$$

where $\tilde{\phi}_i^{(n)} = \phi^{(n)}(t_i)/i!$[1]. The m-order polynomial DLM actually corresponds to the m-order Taylor expansion of the physical process function with $\tilde{\phi}_i^{(n)}$ matches the $(n+1)$th element of $\boldsymbol{\theta}_t$, where the unknown polynomial coefficients $\tilde{\phi}_i^{(n)}$ are stochastic and subject to random noise rather than being fixed. For most sensor data, linear approximations (i.e. 1st- or 2nd-order DLMs) are sufficient to capture the future evolvement of the process; higher order models with complicated polynomial growth are usually not appropriate unless that specific growth is known *a priori*.

[1]This argument holds as long as f is continuous and differentiable

C. Multivariate DLM extension

For most WSN applications the deployed nodes have more than one type of sensor, and the intra-node variates are usually correlated. Figure 2, for example, shows temperature and humidity observations that are negatively correlated. To model multivariate sensor data, an easy extension could have been treating $y_{i,t}$ for $i \in \{1, \ldots, p\}$ as independent components. However, this extension ignores the inter-variate correlations and also excludes the important "borrowing strength" of the multivariate model.

A more realistic model can be built based upon the univariate DLMs. Assume each univariate sensor data $y_{i,t}$ admits a polynomial DLM model (i.e. local level or trend model):

$$y_{i,t} = \boldsymbol{F}_t \boldsymbol{\theta}_{i,t} + v_{i,t}, \qquad v_{i,t} \sim \mathcal{N}(0, \sigma_{ii}^2); \tag{3}$$

$$\boldsymbol{\theta}_{i,t} = \boldsymbol{G}_t \boldsymbol{\theta}_{i,t-1} + \boldsymbol{w}_{i,t}, \;\; \boldsymbol{w}_{i,t} \sim \mathcal{N}_m(\boldsymbol{0}, \boldsymbol{W}_t \sigma_{ii}^2). \tag{4}$$

The multivariate model can be constructed as follows:

$$\boldsymbol{y}_t = (\boldsymbol{F}_t \otimes \boldsymbol{I}_p)\boldsymbol{\theta}_t + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{\Sigma}); \tag{5a}$$

$$\boldsymbol{\theta}_t = (\boldsymbol{G}_t \otimes \boldsymbol{I}_p)\boldsymbol{\theta}_{t-1} + \boldsymbol{w}_t. \;\; \boldsymbol{w}_t \sim \mathcal{N}_{mp}(\boldsymbol{0}, \boldsymbol{W}_t \otimes \boldsymbol{\Sigma}), \tag{5b}$$

where $\boldsymbol{y}_t = (y_{1,t}, \ldots, y_{p,t})'$, $\boldsymbol{v}_t = (v_{1,t}, \ldots, v_{p,t})'$, $\boldsymbol{I}_n$ is a n-order identity matrix, $\otimes$ is the Kronecker product, $\boldsymbol{\Sigma} = (\sigma_{ij}^2)_{p \times p}$ is the (symmetric, positive-definite) covariance matrix of $\boldsymbol{v}_t$; and $\boldsymbol{\theta}_t = \text{vec}(\boldsymbol{\Theta}_t')$, $\boldsymbol{w}_t = \text{vec}(\boldsymbol{\Omega}_t')$, where

$$\boldsymbol{\Theta}_t = (\boldsymbol{\theta}_{1,t}, \ldots, \boldsymbol{\theta}_{p,t}), \; \boldsymbol{\Omega}_t = (\boldsymbol{w}_{1,t}, \ldots, \boldsymbol{w}_{p,t}),$$

and vec is the standard vec-operator which stacks the columns of a $m \times p$ matrix into a mp column vector. This multivariate DLM model still conforms to the general DLM definition, with a modified quadruple

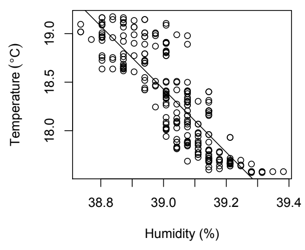
$$[\boldsymbol{F}_t \otimes \boldsymbol{I}_p, \boldsymbol{G}_t \otimes \boldsymbol{I}_p, \boldsymbol{\Sigma}, \boldsymbol{W}_t \otimes \boldsymbol{\Sigma}]. \tag{5c}$$

Among the four elements, only $\boldsymbol{\Sigma}$, and $\boldsymbol{W}_t$ are unknown (whose learning algorithm is discussed in the next section). The inter-variate correlation is introduced by the measurement covariance $\boldsymbol{\Sigma}$: *e.g.* off-diagonal covariances with larger magnitude indicate a stronger linkage between the two variates.

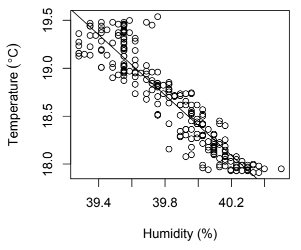
To summarize, with the help of the above formalisms, various sensor data, univariate or multivariate, can be modelled by first selecting the appropriate univariate DLM quadruple and then transforming it to a multivariate model by (5).

IV. Bayesian Conjugate Inference of DLM

For sensor applications, the inference problem revolves around the data. The distribution of *future* observations is of special interest to a sensor node, specifically the h-step lookahead predictive distribution $p(\boldsymbol{y}_{t+h}|\boldsymbol{y}_{1:t})$ for any $h \geq 1$. This predictive distribution provides information about expected future observations based on the data collected so far.



(a) Node 1



(b) Node 2

Fig. 2. The intra-node attributes correlation: humidity versus temperature

Relying on the distributions, informed decisions about future data collection can be made.

For a DLM with a known quadruple, the seminal Kalman filter [13] can be used to estimate the predictive distribution. However, in real sensor applications, the covariance matrices of the quadruple are unknown until the data is actually observed. Traditional maximum-likelihood estimation might be used [14]: however, such estimation requires an iterative optimisation procedure.

Instead, we use conjugate Bayesian learning that treats unknown quantities as random variables to achieve an affordable inference engine. To simplify the illustration, only the multivariate DLM result is presented; however, the univariate case can be easily recovered as a special case. To achieve this computational efficiency, the conjugate prior for $\{\boldsymbol{\Sigma}, \boldsymbol{\theta}_0\}$ (the unknown quantities of a DLM[2]) is assumed. An Inverse-Wishart (IW) distribution is the natural conjugate prior distribution for (matrix valued) variance. There exist different IW distribution functions in the literature [15], [16], [17], [9], [18]: in this paper, we adopt the definition used in [17]: $\boldsymbol{\Sigma} \sim \mathcal{IW}(\nu, \boldsymbol{S})$ with density

$$p(\boldsymbol{\Sigma}) \propto C \times |\boldsymbol{\Sigma}|^{-(\nu+p+1)/2} \exp\left\{-\frac{1}{2}\text{tr}(\boldsymbol{S}\boldsymbol{\Sigma}^{-1})\right\},$$

where $C = \left(2^{\nu p/2}\pi^{p(p-1)/4}\prod_{i=1}^{p}\Gamma\left(\frac{\nu+1-i}{2}\right)\right)^{-1} \cdot |\boldsymbol{S}|^{-\nu/2}$ is a constant that is independent of $\boldsymbol{\Sigma}$[3].

Theorem 1 (Bayesian conjugate learning)**.** *Adopt the following prior distribution for $\boldsymbol{\theta}_0, \boldsymbol{\Sigma}|\emptyset$:*

$$p(\boldsymbol{\theta}_0, \boldsymbol{\Sigma}) = p(\boldsymbol{\theta}_0|\boldsymbol{\Sigma})p(\boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{C}_0 \otimes \boldsymbol{\Sigma})\mathcal{IW}(n_0, \boldsymbol{S}_0) \tag{6}$$

$$\triangleq \mathcal{NIW}_{\otimes}(\boldsymbol{m}_0, \boldsymbol{C}_0, n_0, \boldsymbol{S}_0) \tag{7}$$

for some pre-determined parameters $\boldsymbol{m}_0, \boldsymbol{C}_0, n_0, \boldsymbol{S}_0$ in the multivariate sensor DLM model specified in (5).

for $t > 0$:

1) *Evolution step t:*

$$\boldsymbol{\theta}_t|\boldsymbol{\Sigma}, \boldsymbol{y}_{1:t-1} \sim \mathcal{N}_{mp}(\boldsymbol{a}_t, \boldsymbol{R}_t \otimes \boldsymbol{\Sigma})$$

where

$$\boldsymbol{a}_t = (\boldsymbol{G}_t \otimes \boldsymbol{I}_p)\boldsymbol{m}_{t-1}, \quad \boldsymbol{R}_t = \boldsymbol{G}_t\boldsymbol{C}_{t-1}\boldsymbol{G}'_t + \boldsymbol{W_t} \tag{8a}$$

2) *Conditional predictive:*

$$\boldsymbol{y}_t|\boldsymbol{\Sigma}, \boldsymbol{y}_{1:t-1} \sim \mathcal{N}_p(\boldsymbol{f}_t, Q_t\boldsymbol{\Sigma}),$$

with unconditional predictive

$$\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1} \sim \mathcal{T}_p(\boldsymbol{f}_t, \frac{Q_t\boldsymbol{S}_{t-1}}{n_{t-1}-p+1}, n_{t-1}-p+1)$$

where

$$\boldsymbol{f}_t = (\boldsymbol{F}_t \otimes \boldsymbol{I}_p)\boldsymbol{a}_t, \quad Q_t = \boldsymbol{F}_t\boldsymbol{R}_t\boldsymbol{F}'_t + 1 \tag{8b}$$

3) *Updated Posterior at t:*

$$\boldsymbol{\theta}_t, \boldsymbol{\Sigma}|\boldsymbol{y}_{1:t} \sim \mathcal{NIW}_{\otimes}(\boldsymbol{m}_t, \boldsymbol{C}_t, n_t, \boldsymbol{S}_t),$$

with marginal

$$\boldsymbol{\Sigma}|\boldsymbol{y}_{1:t} \sim \mathcal{IW}(n_t, \boldsymbol{S}_t)$$

where

$$\begin{aligned} \boldsymbol{m}_t &= \boldsymbol{a}_t + (\boldsymbol{K}_t \otimes \boldsymbol{I}_p)\boldsymbol{e}'_t, & \boldsymbol{C}_t &= \boldsymbol{R}_t - \boldsymbol{K}_t\boldsymbol{K}'_tQ_t, \\ n_t &= n_{t-1} + 1, & \boldsymbol{e}_t &= \boldsymbol{y}_t - \boldsymbol{f}_t \\ \boldsymbol{K}_t &= \boldsymbol{R}_t\boldsymbol{F}'_t/Q_t, & \boldsymbol{S}_t &= \boldsymbol{S}_{t-1} + \boldsymbol{e}_t\boldsymbol{e}'_t/Q_t \end{aligned} \tag{8c}$$

Figure 3 shows the Bayesian learning procedures graphically. Without concrete prior knowledge, the prior is usually set diffusive or non-informative. Note the posterior distribution is adapted or learnt when more data is admitted; and the diffusive prior is shrinking as more data is learnt. In this work, non-informative priors are always used. According to Theorem 1, the predictive distribution is a p-variate Student T distributed. A Student T is similar to a Gaussian distribution but with heavier tail. The following useful results are listed. First, any subset of a multivariate T-random vector is still Student T distributed with model parameters immediately available from the joint distribution. This result allows one to infer on each individual sensor observation without any further computational effort.

Result 1 (Marginal T Distribution)**.** *Let $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix}$ be distributed as $\mathcal{T}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu)$ with $\boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{\mu}_1\\ \boldsymbol{\mu}_2\end{bmatrix}$, $\boldsymbol{\Sigma} = \begin{bmatrix}\boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12}\\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22}\end{bmatrix}$,*

[2]Efficient specification of $\boldsymbol{W}_t$ is detailed in the following section

[3]Quintana [19] and West [12] use a different form of IW which lead to different estimation update rules for ν and $\boldsymbol{S}$. The benefit of our definition is that the estimation algorithm for $\boldsymbol{S}$ is completely recursive and can be calculated without re-scaling.

and $|\boldsymbol{\Sigma}_{22}| > 0$*, where* $\boldsymbol{\mu}_1$ *and* $\boldsymbol{\mu}_2$ *are* p_1*,* p_2 *dimensional random vectors and* $p_1 + p_2 = p$*. Then the marginal distribution,* $\boldsymbol{x}_i$*,* $i = 1, 2$*, is also T distributed; Specifically,*

$$\boldsymbol{x}_i \sim \mathcal{T}_{p_i}(\boldsymbol{x_i}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \nu).$$

Second, the conditional distribution, on the other hand, provides a more informed distribution that takes inter-variate correlations into account. For example, when partial observation $\boldsymbol{x}_2$ is available, the conditional distribution on $\boldsymbol{x}_1$ can be obtained by:

Result 2 (Conditional T Distribution)**.** *The conditional distribution,* $\boldsymbol{x}_1|\boldsymbol{x}_2$*, is* p_1*-variate T distributed:*

$$\boldsymbol{x}_1|\boldsymbol{x}_2 \sim \mathcal{T}_{p_1}(\boldsymbol{x_1}; \boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2}, \nu_{1|2}),$$

where

$$\nu_{1|2} = \nu + p_2 \tag{9a}$$

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) \tag{9b}$$

$$\boldsymbol{\Sigma}_{1|2} = \frac{\nu + (\boldsymbol{x}_2 - \boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)}{\nu + p_2}\left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}\right). \tag{9c}$$

Note the conditional distribution is essentially an update upon the marginal densities where the "update" is provided by the partial observation.

A. Specification of $\mathbf{W}_t$ *by Discount Factor*

Theorem 1 only works when $\boldsymbol{W}_t$ is pre-specified. In reality, the matrix that represents the variance ratios also needs to be learnt from data. Unfortunately, this complicated model (with essentially two unknown variance matrices $\{\boldsymbol{\Sigma}, \boldsymbol{W}_t\}$) has no closed form solution [14]. To resolve the problem, we adopt the common practice of specifying $\boldsymbol{W}_t$ dynamically as a discount factor of $\boldsymbol{P}_t = \boldsymbol{G}_t\boldsymbol{C}_{t-1}\boldsymbol{G}_t'$,

$$\boldsymbol{W}_t = \frac{1-\delta}{\delta}\boldsymbol{P}_t, \tag{10}$$

where $\delta \in (0, 1]$ is the discount factor selected by the modeller [12], [9], [20]. δ actually represents the percentage of the precision lost as t increments. For routine analysis, δ is usually selected between the range $[0.9, 0.99]$ [12]; in this work, a $\delta = 0.9$ is used. We adopt the method not only because it achieves good forecasting results but also there is no need to store $\boldsymbol{W}_t$ specifically, as it is implicitly defined by $\boldsymbol{P}_t$, an existing parameter used in Theorem 1. This "parsimony" property is important for sensor nodes with limited storage capacity. The following theorem shows that the Bayesian conjugate analysis with discount factor specification is efficient in both a time and a space sense.

Theorem 2. *For a DLM model* (5) *with unknown* $\boldsymbol{\Sigma}$ *and discount factor specification of* $\boldsymbol{W}_t$*, the algorithm in Theorem 1 has constant* $O(1)$ *space complexity and linear* $O(n)$ *time complexity, where* n *is the point the update stops or the size of the time series.*

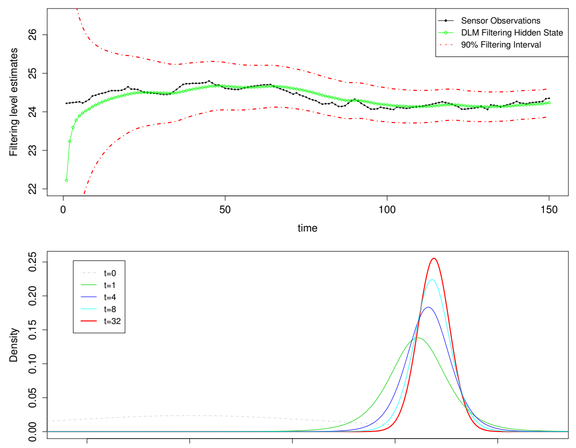


Fig. 3. Bayesian conjugate inference with a noninformative prior

Proof. Note the recurrent relationship between (8a) and (8c): there is no need to store $\boldsymbol{a}_t, \boldsymbol{R}_t$ specifically as they can be updated directly upon $\{\boldsymbol{m}_{t-1}, \boldsymbol{C}_{t-1}\}$. At each time instance t, the procedures require the exact local storage of $\boldsymbol{e}_t, \boldsymbol{f}_t, \boldsymbol{Q}_t, \boldsymbol{K}_t, \boldsymbol{S}_t, n$, and no historic sensor data need to be stored; therefore, the space complexity is constant at each time instance. For time complexity, we calculate the complexity of each step in Theorem 1: at each t, (8a) is of $O((mp)^2 + m^3)$; (8b) is of $O(p \times mp + m^2)$; (8c) is of $O(m^2 + mp \times p + m^2 + p^2)$; the total is $O((mp)^2 + m^3 + mp^2) = O(max(m, p)^4)$. Note when a DLM is chosen, m, p are both constant integers: therefore the time complexity is constant at each t, leading to a linear growth as t increments. □

B. Model update with missing observation

Sensor nodes are unreliable, not only in the sense that sensor observations might be faulty, but that they may not be available from time to time. In both cases, the rational practice is to treat the observation as missing. Fortunately, model update with missing observation can be performed easily with Bayesian learning. To deal with a missing observation at t – so $\boldsymbol{y}_t$ is missing – one simply adopts the following update procedures to replace Equation (8c):

$$\boldsymbol{m}_t = \boldsymbol{a}_t; \boldsymbol{C}_t = \boldsymbol{R}_t; n_t = n_{t-1}; \boldsymbol{S}_t = \boldsymbol{S}_{t-1}; \tag{11a}$$

the procedures follow because $p(\boldsymbol{\theta}_t, \boldsymbol{\Sigma}|\boldsymbol{y}_{1:t-1}, \boldsymbol{y}_t = \texttt{NULL}) = p(\boldsymbol{\theta}_t, \boldsymbol{\Sigma}|\boldsymbol{y}_{1:t-1})$, the new observation provides no real update to the model. At the $(t+1)$th evolution step after the update step, the following evolution procedure is adopted instead of Equation (8a):

$$\boldsymbol{a}_{t+1} = (\boldsymbol{G}_{t+1} \otimes \boldsymbol{I}_p)\boldsymbol{m}_t, \quad \boldsymbol{R}_{t+1} = \boldsymbol{G}_{t+1}\boldsymbol{C}_t\boldsymbol{G}_{t+1}' + \boldsymbol{W}_t, \tag{11b}$$

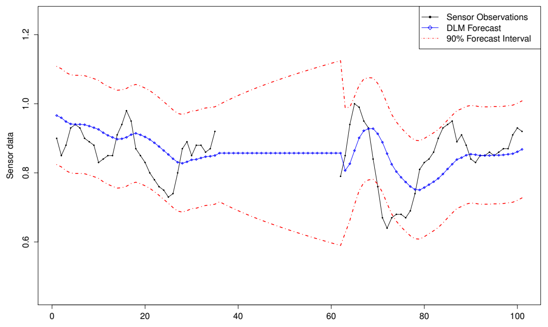


Fig. 4. Filtering when observation are missing

where $\boldsymbol{W}_t = \frac{1}{\delta}\boldsymbol{P}_t$ is the previous evolution matrix specified at t. Note the update actually maintains a constant step-ahead evolution matrix, $\boldsymbol{W}_{t+1} = \boldsymbol{W}_t$. This modification prevents the explosion of the evolution variance $\boldsymbol{R}_t$ when multiple k consecutive observations are missing. To see this, if $\{\boldsymbol{y}_{t+1:t+k}\}$ are missing, the ordinary evolution step (8a) would lead to

$$\boldsymbol{R}_{t+k} = \boldsymbol{G}^k \boldsymbol{C}_t \boldsymbol{G}^{k\prime}/\delta^k,$$

where $\boldsymbol{G}^k \triangleq \boldsymbol{G}_{t+k}\boldsymbol{G}_{t+k-1}\cdots\boldsymbol{G}_{t+1}$. Since δ lies in $(0,1]$, $\boldsymbol{R}_{t+k}$ will increase exponentially as k accumulates, which contradicts the linear form of DLM [12]. Figure 4 demonstrates graphically the model update procedures with missing observations. Note the forecast interval (model uncertainty) automatically adjusts when data is missing.

h-step-ahead forecast: Theorem 1 provides the algorithm for calculating the one-step-ahead forecast distribution. The general h-step-ahead forecast distribution with $h > 1$ can actually be easily obtained by treating the future observations $\{\boldsymbol{y}_{t+1:t+h-1}\}$ as missing. The result is true because the following identity

$$p(\boldsymbol{y}_{t+h}|\boldsymbol{y}_{1:t}) = p(\boldsymbol{y}_{t+h}|\boldsymbol{y}_{1:t}, \boldsymbol{y}_{t+1:t+h-1} = \texttt{null}). \quad (12)$$

V. Sensor Control with DLM

To show the practical use of DLM, we investigate the sampling scheduling problem of WSNs. An DLM based data collection framework featuring adaptive sampling and guaranteed sampling accuracy is proposed and evaluated in this section. The framework forms a self-organizing distributed solution where each node determines locally the sampling decisions with the help of DLMs and global good results can be achieved out of this self-managed scheme.

A. Problem Statement

Let ϵ be the precision required of the sensor data. For example, if the true signal is $s \in \mathbb{R}$, then any value within the range of $[s-\epsilon, s+\epsilon]$ is satisfactory. The problem is to determine the sampling time for each sensor node such that the sampled and collected data meet the precision requirement with small computational overhead. As pointed out by Alippi *et al.* [21], the energy spent on sensing is comparable to other intensive sensor activities including RF communication. However, blindly reducing sensing will lead to insufficient samples, which in turn results in inadequate data collection. Therefore, the trade-off between accuracy (resolution) and energy efficient needs to be resolved by the autonomic control.

B. Overview

The proposed solution starts with a one-off *model learning phase* in which each sensor node learns its local DLM model according to Theorem 1 until a pre-determined N_l number of sensor observations have been learnt or the model parameters have stabilised. To ease the computation burden for the sensor nodes, only univariate DLM models are maintained for each sensor data series. For the convergence test, we examine the prediction's variance

$$\boldsymbol{\Lambda}^t = \frac{Q_t \boldsymbol{S}_{t-1}}{n_{t-1} - p + 1}$$

and stop the learning phase if

$$\frac{|\boldsymbol{\Lambda}^t - \boldsymbol{\Lambda}^{t-1}|}{\boldsymbol{\Lambda}^{t-1}} < 0.01.$$

The objective of this phase is to obtain a functioning DLM model with learnt and stabilised model parameters. Note that to communicate the learned result, the nodes are not required to send back the learning data to the sink but only the posterior model parameters. After receiving the model parameters, the sink has the synchronised DLM models maintained in the network.

The operational phase commences after the learning phase. Figure 5 shows the procedures in the operational phase graphically. Each node first decides the next sampling time, T, by making inference on the DLM model. The sampling schedule decision is made such that only those "valuable" or "interesting" samples are taken, and the rest is lost during the period in between. The nodes then act according to the sampling schedule: upon time T, the node takes a series of I_{update} consecutive samples and updates its local model accordingly. For data collection applications, the node may also send the sensor data to the sink to update both the local and server DLM models such that the DLM model is synchronised again. Any missing elements (which should have been collected during the sleeping period) can be easily restored at the server-side by running a DLM inference. The details on the scheduling algorithm and missing value inferences are presented in the following sections.

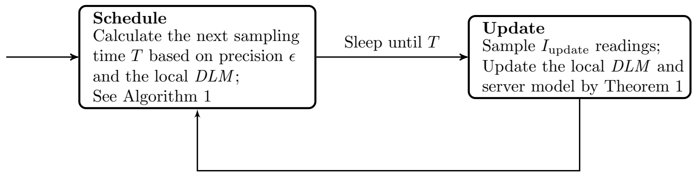

Schedule
Calculate the next sampling time T based on precision ϵ and the local DLM;
See Algorithm 1
Sleep until T
Update
Sample I_{update} readings;
Update the local DLM and server model by Theorem 1

Fig. 5. Dynamical linear model based data collection framework flow chart

C. DLM Based Sampling Scheduling

According to Theorem 1 and (12), an h-step ahead prediction distribution is still Student T distributed:

$$y_{t+h}|y_{1:t} \sim \mathcal{T}(f(h), Q(h), \nu(h)).$$

Therefore, a prediction interval can be derived:

$$\mathbb{P}\left(f(h) - t_{\alpha,\nu(h)}\sqrt{Q(h)} < y_{t+h} < f(h) + t_{\alpha,\nu(h)}\sqrt{Q(h)}\right) = 1 - 2\alpha, \quad (13)$$

where $t_{\alpha,\nu(h)}$ is the critical percentile value for the Student T random variable. For example, when $\alpha = 2.5\%$ is used, the future observation y_{t+h} will fall in the envelope $\left[f(h) - t_{0.025,\nu(h)}\sqrt{Q(h)}, f_t(h) + t_{0.025,\nu(h)}\sqrt{Q_t(h)}\right]$ with a 95% level of confidence, conditional on all the historic data.

As h rolls forward, the uncertainty accumulates so the interval expands. In view of this, a sensor node applies a greedy algorithm to decide when to sample again: as long as the confidence interval is smaller than the user-specified precision interval $[f(h) - \epsilon, f(h) + \epsilon]$, there is no motivation to sample the environment as the future observations are likely with high confidence to be within the forecast interval; meanwhile its precision is also satisfactory. Algorithm 1 summarises this scheduling decision making procedure.

Algorithm 1 Sampling scheduling greedy algorithm

```
Input: Precision $\epsilon$ ($\epsilon > 0$); Forecast limit $H$; Forecasting
    confidence level $\alpha$
1: $h \leftarrow 1$
2: while $h \leq H$ do
3:     Calculate $Q(h), \nu(h)$ based on Theorem 1 and (12)
                ▷ Calulate the $h$-step ahead prediction interval
4:     if $t_{\alpha,\nu(h)}\sqrt{Q(h)} > \epsilon$ then
5:         break outer while loop
6:     end if
7:     $h \leftarrow h + 1$
8: end while
9: $T \leftarrow h - 1$
```

D. Model Update

Upon the scheduled sampling point T, the sensor node reaches the model update stage, in which the node samples the environment and updates the local model again. This update stage is necessary as the accumulating uncertainty has reached a threshold such that model inference cannot satisfy user's precision requirement.

We denote the sampling size as I_{update}, i.e. at T the sensor node takes I_{update} consecutive samples. These sensor readings need to be reported back to the sink so that both the local node and the sink update their DLM models according to Theorem 1. I_{update} may be defined by the user as a fixed constant. However, it can also be determined dynamically on an on-demand basis: the update stage finishes when the predictive interval shrinks and falls below a threshold, say the user-defined precision ϵ again.

E. Data restoration at the sink

For data collection applications, the sink needs to report all the sensor measurements for end users. This means the sink needs to recover those missing observations that were not taken due to the adaptive sampling policy.

With the DLM formalism, the data restoration can be achieved with little computational effort. Assume observation y_t is missing. Since the local and server models are always synchronized during the update phase; therefore, the sink can simply supply the one-step-ahead point forecast f_t from Theorem 1 as the missing values for each of the univariate DLM it maintains. In other words, the missing values are replaced by $f_t = \mathbb{E}_{y_t|y_{1:t-1}}[y_t]$. Note it can be shown that the point forecast has the smallest expected squared loss,

$$\mathbb{E}_{y_t|y_{1:t-1}}[(y_t - f_t)^2] = \int (y_t - f_t)^2 p(y_t|y_{1:t-1}) dy_t,$$

which means the estimator has the smallest squared discrepancy between the missing observation according to the posterior distribution, making it the best estimator for the missing value y_t.

An alternative method is to train an additional multivariate DLM model out of the received but incomplete data so that the cross sectional correlations can also be used to interpolate the missing data. The multivariate DLM learning result Theorem 1 can be used to learn the model. Since each univariate DLM is operated independently; therefore, the received data might happen to complement each other: at each time instance t:

1) the whole observation vector $\boldsymbol{y}_t$ is missing; or
2) some subset $\boldsymbol{y}_{t,1}$ of it is missing while the rest $\boldsymbol{y}_{t,2}$ are observed, where $\boldsymbol{y}_t = (\boldsymbol{y}_{t,1}, \boldsymbol{y}_{t,2})'$;

the first scenario can be handled by the same way as described above, while for the later case, the conditional student T result can be employed to interpolate the missing partial data. In other words, the missing values are replaced by the mean of the conditional distribution, $\mathbb{E}_{\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}}[\boldsymbol{y}_{t,1}|\boldsymbol{y}_{t,2}]$. Similarly, the estimator also has the smallest expected squared loss with respect to the conditional posterior prediction distribution.

To summarize, apart from the data transmission during the model update phase, there is no other server-sensor communication is required. First, the scheduling decisions are self-determined at each local sensor node; and the data restoration at the server works purely on the data already received without any further query from the sensors. Therefore, the solution is indeed bottom-up and self-organizing.

VI. Evaluation

We present an evaluation in two parts. Firstly we show that the update procedures can all be feasibly implemented on a platform of the capabilities typically used for sensor networks. Secondly, we simulate the behaviour of our system against commonly-used trace data. We defer evaluation of a deployment in the wild to future work.

A. Implementation

To demonstrate the algorithm is a feasible solution for resource constrained sensors, we have implemented the framework in nesC on TinyOS 2.1.0 [22] and evaluated it using IEEE 802.15.4 complaint TMote Sky mote. It consists of an processor running at maximum 8MHz and RAM of 10 KB [23]. The relative small RAM size becomes a major hindrance for the system and application program. The proposed solution with two univariate DLM models embedded (assumed to be temperature and humidity) is implemented. The implementation has a footprint of 584 RAM (in bytes) and 21432 ROM (in bytes), which only account for 5.7% and 43.6% of the total size.

B. Simulation Results

We drive our simulations from sensor data collected as part of a real-world deployment [24]. The simulation is written in R [25] and the results reported are summarized over ten independent runs with sensor data series (temperature, humidity and voltage sensors) randomly selected from the data set. Our method is assessed from two aspects: data collection accuracy, and communication saving. The former assessment shows the quality of the autonomously made decision; the latter demonstrates how sensor nodes can benefit from the sound decisions. We compare the results against the fixed rate sampling schedule which is currently the most widely used scheduling solution regarding the two aspects.

1) Assessment on decision making: To assess the autonomic control, we examine the quality of the collected data that is sampled based on the DLM enhanced scheduling algorithm. Two metrics, *Mean Absolute Difference* (MAD) and *Precision Satisfaction* are used. The metrics are defined as

$$\text{Mean Absolute Difference} = \frac{1}{N}\sum_{i=1}^{N}|\tilde{d}_i - d_i|, \qquad (14)$$

$$\text{Precision Sat.} = \frac{\left|\{\tilde{d}_i : |\tilde{d}_i - d_i| < \epsilon, i \in 1,\dots,N\}\right|}{N}, \qquad (15)$$

where d_i, $\tilde{d}_i$ are the original data collected according to the fixed rate sampling schedule and the data collected as per the sampling scheduling respectively. MAD shows the average difference between the data, while precision satisfaction shows the percentage of the collected data is satisfactory towards the user's requirement.

Tables II to IV show the results for temperature, humidity and voltage data respectively. Two DLM models, 1st (level model) and 2nd (trend model) order polynomial DLMs, are considered. For each type of sensor, we report the simulation results on different levels of precision ϵ requirements. The reported results are the averages together with the standard deviations over the ten independent runs. The following observations can be made based on the tables.

- First, for all the different settings, the mean absolute difference achieved are below their precision requirements: which means on average the decisions made satisfies the application's requirements. It is also interesting to note the solution adapts itself well to the user's precision requirements. When a demanding data precision is required, the solution delivers the performance as required, which can also be seen graphically from Figure 6.
- Second, all the precision satisfaction entries are around 95% (if $\pm$ the standard deviation). The high satisfaction rates imply the decisions made in general are sound and satisfactory. Recall a 95% credible interval is used in the simulation, the results roughly matches our expectation, which also means the DLMs are reliable in delivering the forecasts.
- Third, the trend model outperforms the level model counterparts in data quality for most of the settings. However, the difference on voltage is negligible. This is due to the fact that the voltage data is more stable than the other two data (it does not grow or decline randomly); therefore, a local level model without stochastic trend component is sufficient to model its dynamics.
- Forth, the effect of multivariate DLM based server side data restoration is also apparent from the results. The spatial correlations between the temperature data is exploited to construct a multivariate DLM model at the sink. According to Table II, both the MAD and precision satisfaction are improved by the multivariate extension. Note the multivariate restoration makes use of the same amount of data as the univariate restoration; therefore, they have the same energy saving as the univariate case.
- Last but not least, the proposed solution works on all the three types of sensors, which also means the DLMs are general enough to model different types of data.

TABLE II
SIMULATION RESULTS ON TEMPERATURE SENSOR DATA

Model	Mean Absolute Difference (°C) Univariate	Mean Absolute Difference (°C) Multivariate	Precision Satisfaction (%) Univariate	Precision Satisfaction (%) Multivariate	# of Data Messages	Data Saving (%)
DLM-Level $\epsilon = 0.3$	0.086(±0.01)	0.0682(±0.01)	93.94(±1.39)	95.4(±2.39)	5841.4(±196.5)	59.4(±1.36)
DLM-Level $\epsilon = 0.5$	0.114(±0.017)	0.07(±0.01)	95.96(±0.59)	97.6(±2.51)	5364.4(±260.8)	62.8(±1.81)
DLM-Level $\epsilon = 1.0$	0.217(±0.037)	0.15 (±0.051)	96.5(±1.16)	97.7 (±2.50)	3776.7(±69.12)	73.8(±0.48)
DLM-Trend $\epsilon = 0.3$	0.062(±0.009)	0.047 (±0.007)	95.7(±1.13)	96.7 (±3.93)	6274.3(±270.78)	56.4(±1.88)
DLM-Trend $\epsilon = 0.5$	0.079(±0.015)	0.054 (±0.012)	96.9(±0.91)	97.9 (±2.52)	5899.4(±121.4)	59.0(±0.84)
DLM-Trend $\epsilon = 1.0$	0.167 (±0.033)	0.125 (±0.21)	96.8(±0.82)	98.1 (±2.43)	5295.5(±74.03)	63.2(±0.51)

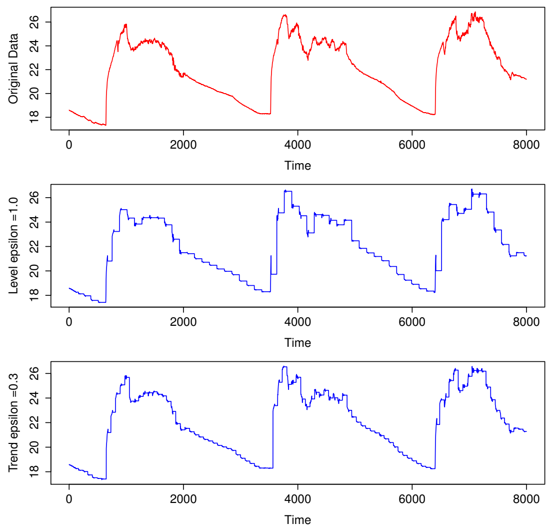


Fig. 6. Evaluation of the DLM-based solution on temperature sensor data. The top figure shows the original data; the middle frame is the data collected by the level model with $\epsilon = 1$°C; the bottom frame is the result for level model with $\epsilon = 0.3$°C.

TABLE III
SIMULATION RESULTS ON HUMIDITY SENSOR DATA

Model	Mean Absolute Difference (%)	Precision Satis. (%)	Data Saving (%)
Level $\epsilon = 2.5$	0.564(±0.208)	94.07(±2.39)	44.2(±15.43)
Level $\epsilon = 5.0$	1.178(±0.171)	95.58(±0.98)	63.90(±7.83)
Level $\epsilon = 10.0$	2.005(±0.534)	98.2(±2.16)	83.01(±0.0)
Trend $\epsilon = 2.5$	0.301(±0.136)	97.36(±1.54)	32.2(±13.6)
Trend $\epsilon = 5.0$	0.819(±0.136)	97.23(±0.96)	52.15(±8.06)
Trend $\epsilon = 10.0$	1.827(±0.318)	97.6(±1.46)	73.76(±4.72)

TABLE IV
SIMULATION RESULTS ON VOLTAGE DATA

Model	Mean Absolute Difference (%)	Precision Satisf. (%)	Data Saving (%)
Level $\epsilon = 0.01$	0.002(±0.002)	94.3(±3.19)	47.5(±25.86)
Level $\epsilon = 0.03$	0.007(±0.001)	96.3(±1.63)	91.5(±0.76)
Level $\epsilon = 0.05$	0.007(±0.001)	98.5(±1.54)	91.8(±0.0)
Trend $\epsilon = 0.01$	0.002(±0.002)	94.6(±2.71)	47.9(±25.3)
Trend $\epsilon = 0.03$	0.007(±0.001)	96.4(±1.81)	91.5(±0.79)
Trend $\epsilon = 0.05$	0.007(±0.001)	98.4(±1.6)	91.8(±0.0)

2) Assessment on communication saving: According to Tables II to IV, it is clear that for all three types of sensors, the proposed solution significantly reduces the data communication in comparison with the original solution. For example, for the temperature sensor at a precision requirement level of $\epsilon = 0.5$°C, the amount of actual data communication of the local level model is 5364.4 messages, and the corresponding saving is over 62.8%, which means over 62% of the original data is exempted from sending back to the sink. Similar results can be found for the other two sensors. It is interesting to see how the solution adaptively responds to the different settings of the precision requirement. In general, when the less precise data is required, the solution saves more on data communication.

VII. CONCLUSION

In this paper, we put forward the use of Bayesian dynamic linear models for local sensor node control. DLMs are general models that can capture the physical context, which provides valuable information for the local node to make informed decisions. More importantly, lightweight learning algorithms exist such that all the model inferences can be done independently at local sensor node level without any form of inter-node coordination or centralized control.

To prove the effectiveness of DLMs, we consider the adaptive sampling problem: a practical problem facing most data collection WSN deployment. Simulation results show that autonomic local decisions can be made with the help of DLMs. Although each sensor node acts independently according to its local DLM model, the overall performance can be further boosted by exploiting the cross-sectional correlations still with the assistance of DLM. For future work, we are going to examine the solution by real world experiments. How the global performance can be further improved is also an interesting question. We are going to examine the application of Bayesian smoothing technique in data restoration. When the variable size grows, the multivariate DLM model might face computation difficulty. Employing sparse covariance structure will be explored to solve higher dimensional problems.

Appendix A
Proof of Theorem 2

Proof. An indirect proof is to show the equivalence of (5) and the Matrix Normal Dynamic Linear Model (MNDLM) developed in [18]. However, due to the different specifications of IW, the estimation rules would diverge slightly; a direct proof is given below. By Kalman Filter theory, the estimation rules for $\boldsymbol{a}_t, \boldsymbol{f}_t$ follows, as (5) is still a DLM.

(1) Evolution step: $\mathrm{Var}[\boldsymbol{\theta}_t|\boldsymbol{\Sigma}, \boldsymbol{y}_{1:t-1}] = (\boldsymbol{G}_t \otimes \boldsymbol{I}_p)(\boldsymbol{C}_{t-1} \otimes \boldsymbol{\Sigma})(\boldsymbol{G}_t \otimes \boldsymbol{I}_p)' + \boldsymbol{W}_t \otimes \boldsymbol{\Sigma} = (\boldsymbol{G}_t\boldsymbol{C}_{t-1}\boldsymbol{G}_t' \otimes \boldsymbol{\Sigma}) + \boldsymbol{W}_t \otimes \boldsymbol{\Sigma} = (\boldsymbol{G}_t\boldsymbol{C}_{t-1}\boldsymbol{G}_t' + \boldsymbol{W}_t) \otimes \boldsymbol{\Sigma} = \boldsymbol{R}_t \otimes \boldsymbol{\Sigma}$.

(3) Prediction step: the variance of the conditional distribution, $Q_t\boldsymbol{\Sigma}$, can be proved by the same way as step one except $Q_t \otimes \boldsymbol{\Sigma} = Q_t\boldsymbol{\Sigma}$ as Q_t is a scalar. The unconditional result follows due to the property of Normal Inverse Wishart [17].

(3) Update step: denote the precision matrix $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$; by Bayes' theorem:

$$\begin{aligned} p(\boldsymbol{\Sigma}|\boldsymbol{y}_{1:t}) &\propto p(\boldsymbol{\Sigma}|\boldsymbol{y}_{1:t-1})p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\Sigma})) \\ &\propto |\boldsymbol{\Lambda}|^{(n_{t-1}+p+1)/2}\exp\left\{-\mathrm{tr}(\boldsymbol{S}_{t-1}\boldsymbol{\Lambda})/2\right\} \\ &\quad |\boldsymbol{\Lambda}|^{1/2}\exp\left\{-(\boldsymbol{y}_t - \boldsymbol{f}_t)'Q_t^{-1}\boldsymbol{\Lambda}(\boldsymbol{y}_t - \boldsymbol{f}_t)/2\right\} \\ &\propto |\boldsymbol{\Lambda}|^{(n_{t-1}+p+1+1)/2}\exp\left\{\mathrm{tr}\left(\boldsymbol{e}_t\boldsymbol{e}_t'/Q_t \cdot \boldsymbol{\Lambda}\right) + \mathrm{tr}(\boldsymbol{S}_{t-1}\boldsymbol{\Lambda})\right\} \\ &= |\boldsymbol{\Lambda}|^{(n_{t-1}+p+1+1)/2}\exp\left\{\mathrm{tr}\left(\boldsymbol{S}_t\boldsymbol{\Lambda}\right)\right\} \propto \mathcal{IW}(n_t, \boldsymbol{S}_t), \end{aligned}$$

which proves the update procedures for $\boldsymbol{\Sigma}$. The update rules for $\boldsymbol{\theta}_t$ can be proved by noting

$$\left.\begin{bmatrix}\boldsymbol{\theta}_t \\ \boldsymbol{y}_t\end{bmatrix}\right| \boldsymbol{y}_{1:t-1}, \boldsymbol{\Sigma} \sim \mathcal{N}\left[\begin{bmatrix}\boldsymbol{a}_t \\ \boldsymbol{f}_t\end{bmatrix}, \begin{bmatrix}\boldsymbol{R}_t \otimes \boldsymbol{\Sigma} & \boldsymbol{R}_t\boldsymbol{F}_t' \otimes \boldsymbol{\Sigma} \\ \boldsymbol{F}_t\boldsymbol{R}_t' \otimes \boldsymbol{\Sigma} & Q_t \otimes \boldsymbol{\Sigma}\end{bmatrix}\right];$$

apply Gaussian regression theory [26],

$$\boldsymbol{\theta}_t|\boldsymbol{\Sigma}, \boldsymbol{y}_{1:t} \sim \mathcal{N}_{mp}(\boldsymbol{m}_t, \boldsymbol{C}_t \otimes \boldsymbol{\Sigma})$$

is proved.

The above are valid for all $t > 0$, which proves the theory. □

References

[1] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, ser. VLDB '04. VLDB Endowment, 2004, pp. 588–599.

[2] D. Tulone and S. Madden, "PAQ: Time series forecasting for approximate query answering in sensor networks," in *Wireless Sensor Networks*. Springer, 2006, pp. 21–37.

[3] Y.-A. Le Borgne, S. Santini, and G. Bontempi, "Adaptive model selection for time series prediction in wireless sensor networks," *Signal Processing*, vol. 87, no. 12, pp. 3010–3020, 2007.

[4] M. J. Handy, M. Haase, and D. Timmermann, "Low energy adaptive clustering hierarchy with deterministic cluster-head selection," in *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*. IEEE, 2002, pp. 368–372. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1045790

[5] B. Gedik, L. Liu, and P. S. Yu, "ASAP: An Adaptive Sampling Approach to Data Collection in Sensor Networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 18, no. 12, pp. 1766–1783, 2007.

[6] J. Kho, A. Rogers, and N. R. Jennings, "Decentralized control of adaptive sampling in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, no. 3, p. 19, 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1525857

[7] P. Padhy, R. K. Dash, K. Martinez, and N. R. Jennings, "A utility-based sensing and communication model for a glacial sensor network," in *Proceedings of the Fifth International Joint conference on Autonomous Agents and Multiagent Systems*, 2006, pp. 1353–1360.

[8] N. Giatrakos, A. Deligiannakis, M. Garofalakis, I. Sharfman, and A. Schuster, "Distributed geometric query monitoring using prediction models," *ACM Trans. Database Syst.*, vol. 39, no. 2, pp. 16:1–16:42, May 2014. [Online]. Available: http://doi.acm.org/10.1145/2602137

[9] C. M. Carvalho and M. West, "Dynamic matrix-variate graphical models," *Bayesian Analysis*, vol. 2, no. 1, pp. 69–98, 2007.

[10] J. Quintana, V. Lourdes, O. Aguilar, and J. Liu, "Global gambling," *Bayesian Statistics*, 2003.

[11] C. Calder, M. Lavine, P. Müller, and J. S. Clark, "Incorporating multiple sources of stochasticity into dynamic population models," pp. 1395–1402, 2003.

[12] M. West, R. Prado, and A. Krystal, "Evaluation and comparison of EEG traces: Latent structure in nonstationary time series," *Journal of the American Statistical ...*, 1999. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/01621459.1999.10474128

[13] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960. [Online]. Available: http://fluidsengineering.asmedigitalcollection.asme.org/article.aspx?articleid=1430402

[14] G. Petris, S. Petrone, and P. Campagnoli, *Dynamic linear models with R*. New York, NY, USA: Springer, 2009.

[15] A. K. Gupta and D. K. Nagar, *Matrix variate distributions*. Boca Raton, FL: CRC Press, 1999, vol. 104. [Online]. Available: http://books.google.com/books?hl=en&lr=&id=PQOYnT7P1loC&pgis=1

[16] A. P. Dawid, "Some matrix-variate distribution theory: Notational considerations and a Bayesian application," *Biometrika*, vol. 68, no. 1, pp. 265–274, 1981.

[17] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*. New York, NY, USA: Chapman & Hall, 2014.

[18] M. West and J. Harrison, *Bayesian forecasting and dynamic models*. New York, NY, USA: Springer, 1997.

[19] J. M. Quintana, "Multivariate Bayesian forecasting models," Ph.D. dissertation, University of Warwick, Mar. 1987. [Online]. Available: http://webcat.warwick.ac.uk/record=b1452026$\sim$S1

[20] H. Migon and D. Gamerman, "Dynamic models," *Handbook of Statistics*, vol. 25, pp. 553—-588, 2005. [Online]. Available: http://hedibert.org/wp-content/uploads/2013/12/migon-gamerman-lopes-ferreira-2005.pdf

[21] C. Alippi, G. Anastasi, C. Galperti, F. Mancini, and M. Roveri, "Adaptive Sampling for Energy Conservation in Wireless Sensor Networks for Snow Monitoring Applications," in *Proceedings of the IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems*, 2007, pp. 1–6.

[22] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler, "TinyOS: An Operating System for Sensor Networks," in *Ambient Intelligence*, W. Weber, J. Rabaey, and E. Aarts, Eds. Berlin/Heidelberg: Springer Berlin Heidelberg, 2005, ch. 7, pp. 115–148. [Online]. Available: http://dx.doi.org/10.1007/3-540-27139-2_7

[23] Moteiv, *Tmote Sky Datasheet http://www.sentilla.com/pdf/eol/tmote-sky-datasheet.pdf*, 2006.

[24] INTEL, "Intel Berkeley Laboratory sensor data set," http://db.csail.mit.edu/labdata/labdata.html, 2004.

[25] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2014. [Online]. Available: http://www.R-project.org/

[26] R. A. Johnson and D. W. Wichern, *Applied multivariate statistical analysis*. Upper Saddle River, NJ, USA: Prentice hall, 2002.