Distributed Self-Monitoring Sensor Networks via Markov Switching Dynamic Linear Models

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

Juan Ye
School of Computer Science
University of St Andrews, UK
Email: ye.juan@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 empowered with low-cost sensing devices and wireless communications present an opportunity to enable continuous, fine-grained data collection over a wide environment. However, the quality of data collected is susceptible to the hardware conditions and also adversarial external factors such as high variance in temperature and humidity. Over time, the sensors report erroneous readings, which deviate from true readings. To tackle the problem, we propose an efficient self-monitoring, self-managing and self-adaptive sensing framework based on a dynamic hybrid Bayesian network that combines Hidden Markov Model and Dynamic Linear Model. The framework does not only enable automatic on-line inference of true readings robustly but also monitor the working status of sensor nodes at the same time, which can uncover important insights on hardware management. The whole process also benefits from the derived approximation algorithm and thus supports on-line one-pass computation with minimum human intervention, which make the accurate formal inference affordable for distributed edge processing.**

***Index Terms*—self-management, sensor networks, machine learning, DLM, Markov switching model, state space model, hybrid dynamic network**

I. INTRODUCTION

Wireless sensor network (WSN) presents an unprecedented opportunity for many scientific disciplines to explore scientific questions on a collection of fine-grained, detailed observations. One of the most critical challenges in WSN is sensor errors; that is, sensors produce readings that deviate from normal patterns exhibited by true readings [1]. Sensor errors are a prevalent problem in deployed WSNs, which can be caused by malfunction in the sensor hardware, low battery, or environmental interference. All these sensor errors can degrade performance of a WSN, affect its monitoring performance, and result in loss of information fidelity and wrong decision making.

To make sure that sensor data are trustworthy representations of the physical process, a range of control decisions need to be made. For example, from a decentralised self-management perspective, each node needs to decide an appropriate sampling frequency to match the temporal involvement of the phenomenon and also possibly the energy budget limit. To make informed decisions, each node needs to form a sound "understanding" over the physical process. Our previous work demonstrates the value of Bayesian dynamic linear model (DLM) in achieving this goal [2]. DLMs essentially provide such an understanding by making formal probabilistic inference: posterior probability distributions over the physical process are recursively updated based on the accumulating sensor data, which provides the required information for decision making.

However, understanding the physical process alone is not enough. Other context information, like the sensor hardware status, is also important. Sensor nodes are known to be unreliable and volatile. Depending on the hardware status, such as low battery or connection failure, sensors often experience various types of faults. Among them, SHORT, NOISE and CONSTANT are the most common types of faults observed across real world deployments [1]. They exhibit distinctive statistical patterns and provide important insight into further hardware failure diagnosis. Being able to identify and classify them therefore is of great importance. At a surface level, simply filtering the faults can help clean the data, but uncovering the categories of the faults can be more important, which may lead to informed decision making on remedy strategies. For example, transient spike errors probably can be safely ignored; but a brief period of noisy readings, still providing some vague information about the physical process, should be dealt with caution rather than discarded completely; whereas long-lasting or repetitive constant or noise faults should signal sensor replacement.

Ideally, the fault monitoring process should be carried out on-line and on-site; *i.e.*, classify new sensor readings as they arrive locally at each node. This is challenging for both the complicated nature of the task and the physical constraints of the hardware. As the physical process is dynamic and hidden (the sensors readings are noisy observations rather than the process itself), there is limited ground truth or labels for supervised learning techniques. On the other hand, unsupervised clustering solutions usually require fixed-sized times series segmentation for feature extraction, which is inherently unpopular for on-line inference; and finding the optimal window size introduces additional difficulty.

To tackle the problem, we extend our previous work on DLMs and propose a hybrid model that combines the Bayesian Dynamic Linear Models (DLMs) and Hidden Markov Models (HMMs); *i.e.* Hidden Markov Switching Dynamic Linear Models (HMS-DLMs). The proposed HMS-DLMs, falling into a general framework of dynamic hybrid Bayesian network [3], can make inference on both the physical process and other context information at the same time. The key contributions are summarised below:

- A model that can infer both the physical process and also the sensor fault status at the same time;
- A model that can automatically filter out the data faults without any *ad hoc* intervention;
- An efficient approximate on-line inference algorithm derived for local processing;
- The one-pass algorithm that requires no storage of training data;
- The solution that achieves competitive inference results compared to the state-of-the-art techniques.

In the following, we first present the background on DLMs and sensor faults in Section III. The proposed model and approximate inference is then presented in Section IV. We evaluate the method and present the results in Section V. Section VI concludes the paper with discussion and points to future work.

II. RELATED WORK

Researchers have looked into automatic detection and management of sensor errors. The most common approach is to use neighbouring sensors' values, from which their spatial and temporal correlations are explored and then used to predict the ground truth values to detect and measure sensor errors [4], [5], [6]. Miluzzo et al. [7] have designed a semi-blind approach to calibrate sensors with the aid of infrequent true readings from high-fidelity sources; e.g., a calibrated sensor. Kumar et al. have used kriging to correlate readings from sensors deployed at different locations [5].

Hybrid dynamic bayesian networks have been widely used in many disciplines from econometrics to machine learning [8], [9], [10], [11]. Recently, a more flexible non-parametric Bayesian framework that builds on hierarchical dirichlet process is put forward to solve the hidden space size problem [12], where the hidden states are adapted to the dynamics of the data. Most of these existing solutions rely on off-line learning procedures, like EM based procedure or sampling techniques, to train the model. Approximate inference and learning algorithms have also been extensively studied, assumed density filtering [13], [14], sequential sampling [15], [16] and MCMC [17] are popular candidates.

III. BACKGROUND

A. Bayesian dynamic linear model

A dynamic linear model is formed by a hidden Markov process $\{\boldsymbol{\theta}_t\}$ and a data generation process $\{y_t\}$. $\boldsymbol{\theta}_t$ evolves over time based on its previous state $\boldsymbol{\theta}_{t-1}$ but subject to some random turbulence; while y_t is a noisy observation of $\boldsymbol{\theta}_t$. A probabilistic graphical model representation (PGR) of the DLM is listed in Fig. 1. Formally, a DLM can be defined as follows.

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

$$y_t = \boldsymbol{F}_t\boldsymbol{\theta}_t + v_t, \qquad v_t \sim \mathcal{N}(\mu_t, \sigma^2); \tag{1a}$$

$$\boldsymbol{\theta}_t = \boldsymbol{G}_t\boldsymbol{\theta}_{t-1} + \boldsymbol{w}_t, \ \ \boldsymbol{w}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}_t\sigma^2), \tag{1b}$$

together with a prior for $\boldsymbol{\theta}_0 \sim \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{C}_0)$*; where* $\boldsymbol{G}_t$ *and* $\boldsymbol{F}_t$ *are fixed matrices,* $\mathcal{N}(\cdot,\cdot)$ *denotes a Gaussian distribution.*

A DLM model matches the sensor context well. The hidden process can be viewed as the physical process under monitoring, say temperature, while $\{y_t\}$ are the sensor readings. To be more specific, by setting $\boldsymbol{F}_t = \boldsymbol{F} = [1, 0, \ldots, 0]'$, $\mu_t = 0$ and $\boldsymbol{G}_t$ as a $p \times p$ Jordan form matrix with 1 on both diagonal and super diagonal entries, the physical process essentially models an evolving p-order polynomial function of time t and y_t is an unbiased observation of the function [18]. By calculus, the model translates to assuming the physical signal is evolving contentiously and smoothly over time [1]. In practice, the first and second order models are adequate, *i.e.* $p = 1, 2$. For example, when $p = 1$, $\boldsymbol{F} = \boldsymbol{G} = 1$ and $\mu_t = 0$, the physical process is assumed as an evolving constant model and y_t is a sample subject to some sampling noise v_t.

The online inference task is to find $P(\boldsymbol{\theta}_t|y_{1:t})$, the distribution of the signal given data up to t; where $y_{1:t} \triangleq \{y_1 \ldots y_t\}$. If both σ^2 and $\boldsymbol{W}_t$ are known, the inference can be carried out by a Kalman Filter [18]. On the other hand, an online one-pass Bayesian inference can be derived when σ^2 is unknown and $\boldsymbol{W}_t$ is specified by a discount factor $\delta \in (0, 1]$ [2], [18]. The inference is on both σ^2 and $\boldsymbol{\theta}_t$ together, *i.e.* $P(\boldsymbol{\theta}_t, \phi|y_{1:t})$, where $\phi \triangleq 1/\sigma^2$ is the sensing precision. The algorithm is summarised here.

1) Online discount factor learning for a DLM: Assume a conjugate Gaussian Gamma prior for $\boldsymbol{\theta}_0, \phi$ at $t = 0$:

$$P(\boldsymbol{\theta}_0, \phi) = \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{C}_0\phi)\mathcal{G}(n_0, s_0) \triangleq \mathcal{NG}(\boldsymbol{m}_0, \boldsymbol{C}_0, n_0, s_0)$$

with $\boldsymbol{m}_0, \boldsymbol{C}_0, n_0, s_0$ as initial parameters; the recursive update procedure is: for $t > 0$,

$$P(\boldsymbol{\theta}_t, \phi|y_{1:t}) = \mathcal{NG}(\boldsymbol{m}_t, \boldsymbol{C}_t, n_t, s_t) \tag{2}$$

[1] By Taylor's expansion, a continuous and differentiable function $f(t)$ can be locally approximated arbitrarily well by polynomials.

with

$$\begin{aligned} \boldsymbol{m}_t &= \boldsymbol{G}_t\boldsymbol{m}_{t-1} + \boldsymbol{K}_t e_t, & n_t &= n_{t-1} + 1/2, \\ \boldsymbol{C}_t &= \delta^{-1}\boldsymbol{G}_t\boldsymbol{C}_{t-1}\boldsymbol{G}'_t - \boldsymbol{K}_t\boldsymbol{K}'_t Q_t, & s_t &= s_{t-1} + e_t^2/2Q_t \end{aligned}$$

where $\boldsymbol{K}_t = \boldsymbol{R}_t\boldsymbol{F}'_t/Q_t$, $e_t = y_t - f_t$, $f_t = \boldsymbol{F}_t\boldsymbol{G}_t\boldsymbol{m}_{t-1}$, $Q_t = \boldsymbol{F}_t\boldsymbol{R}_t\boldsymbol{F}'_t + 1$, and $\boldsymbol{R}_t = \delta^{-1}\boldsymbol{G}_t\boldsymbol{C}_{t-1}\boldsymbol{G}'_t$. It can be shown the one step ahead forecasting distribution is a student T distribution: $P(y|y_{1:t-1}) = \mathcal{T}_{2n_{t-1}}(f_t, Q_t s_{t-1}/n_{t-1})$ [18].

B. Sensor data faults

Sensor data faults are frequently observed in real deployments. Among them, the most common ones are SHORT, NOISE and CONSTANT [1], [19]. The fault types can also provide insights into the hardware status [1]. For example, the NOISE fault is usually associated with low batteries. The definitions of the three faults are listed below.

- SHORT: A sharp momentary jump between normal readings, hardware failures like fault in the analog-to-digital convert board can be associated with this fault;
- NOISE: Sensor readings exhibit large unexpected variation for a period of time and low batteries can lead to this fault;
- CONSTANT: Also known as "stuck at" faults. Sensor readings remain constant for a period of time, and the reported value can be out of the possible range of the normal readings and uncorrelated to the physical process.

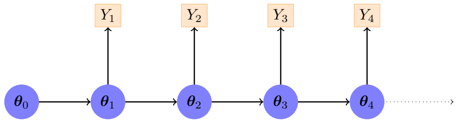

Fig. 1. Probabilistic graphical model representation (PGR) of DLMs for sensing, where the following notational convention is adopted: square nodes are observed data, and circular nodes represent hidden random variables.

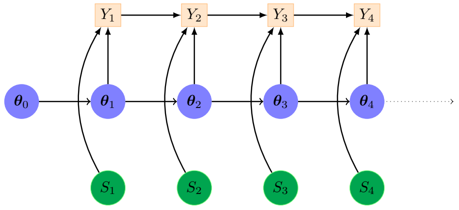

Fig. 2. PGR of Mixture-DLMs for sensing; where the sensor statuses S_t are independent.

IV. THE PROPOSED MODEL

In this section, we give a comprehensive treatment of the proposed solution. We first (in Section IV-A) discuss the limit of the singular DLMs model, which motivates the extended hybrid HMS-DLMs model introduced next. In Section IV-B, we give the specification of the proposed model; the online inference algorithm is presented in Section IV-C.

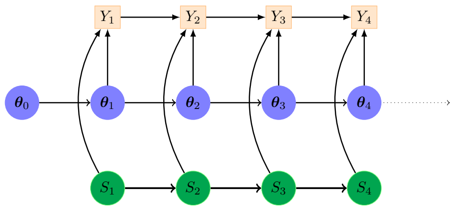

Fig. 3. PGR of HMS-DLMs for sensing; where the sensor statuses S_t are serially dependent.

A. Motivation and overview

Singular DLMs can successfully model a continuously evolving physical process and "normal" functioning sensing model. However, it is not enough to accommodate the rich dynamics induced by the hidden sensor status. A natural choice is to include an extra layer of sensing status on top of the DLM model. To be more specific, to model the sensor status, we introduce hidden process $\{S_t, t > 0\}$, where each $S_t = k \in \{1, 2, \dots, K\}$ is a discrete random variable indicating a particular status of the sensor at time t: for example, a normal state or abnormal states that generate SHORT, NOISE and so on.

A simple probabilistic model for categorical random variables is the multinomial distribution: assume $S_t \sim Multi(\boldsymbol{\pi})$ for all t, where $\boldsymbol{\pi}$ is a probability distribution over $1 \dots K$; the model assumes all S_t are independent draws from one multinomial distribution and $P(S_t = k) = \pi_k$. However, this assumption ignores the possible serial correlations between S_t. When the current sensing status is NORMAL, it is reasonable to assume it might stay NORMAL at the next step; and such transitions should be different depending on the current S_t and possibly its previous history. In light of this, we model S_t as a 1st-order Markov chain: *i.e.* $P(S_t|S_{1:t-1}) = P(S_t|S_{t-1} = k) = \boldsymbol{\pi}_k$. The associated transition matrix is written as $\boldsymbol{\Gamma} = [\boldsymbol{\pi}'_1, \dots, \boldsymbol{\pi}'_K]'$, where $P(S_t = j|S_{t-1} = i) = \boldsymbol{\Gamma}_{ij}$ for $i, j \in \{1 \dots, K\}$. The formal definition of this hybrid model can be written as follows.

Definition 2. *A K state Hybrid DLM model is:*

$$y_t = \boldsymbol{F}_t^{[S_t]}\boldsymbol{\theta}_t + v_t^{[S_t]}, \qquad \boldsymbol{v}_t^{[S_t]} \sim \mathcal{N}(\mu_t^{[S_t]}, \sigma^{2[S_t]}); \tag{3a}$$

$$\boldsymbol{\theta}_t = \boldsymbol{G}_t^{[S_t]}\boldsymbol{\theta}_{t-1} + \boldsymbol{w}_t^{[S_t]}. \quad \boldsymbol{w}_t^{[S_t]} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}_t^{[S_t]}\sigma^{2[S_t]}), \tag{3b}$$

where $S_t \in 1 \dots K$ is a discrete valued stochastic process.

For the Markovian case of S_t, the model is named as Hidden Markov Switching DLMs (HMS-DLMs). And for the constant multinomial case, the model actually degenerates to a mixture of DLMs, denoted as M-DLMs thereafter. The probabilistic graphical representations (PGR) of the two models are listed in Fig. 2 and 3 respectively. Note their differences over the singular DLMs in Fig. 1 and the different specifications over S_t between them.

B. Proposed HMS-DLMs model for sensor

In this section, we give detailed description on the proposed HMS-DLMs for sensor nodes. We first discuss the component DLMs individually; and then the transition distribution on S_t.

1) A four states DLMs for sensor: Based on the fault categories, we propose a four state HMS-DLMs to model sensor's behaviours comprehensively. They are NORMAL, SHORT, NOISE and CONSTANT, corresponding to the named working status of a sensor. Extensions to accommodate more states, say expanding NOISE into LOW NOISE and HIGH NOISE, is also straightforward. The proposed model is an observation switching HMS-DLMs, in which the process model, defined in (3b), is assumed to be the same among the four regimes but the process for y_t differs. This assumption matches the problem context: the physical process is always a smoothly evolving function irrespective of how sensors measure it. As some of the four models are not standard DLMs; for example, non-Gaussian noises are used to model outlier, we also find their approximate DLM representations such that the standard inference procedure can still be applied [2].

a) NORMAL: We model the normal state as the DLM introduced in Section III-A, *i.e.* $\boldsymbol{F}_t = \boldsymbol{F} = [1, 0, \ldots, 0]'_{p\times 1}$, $\mu_t = 0$, $p \in \{1, 2\}$ and $\boldsymbol{G}_t$ as a $p \times p$ Jordan form matrix with unit entries on the diagonal and upper diagonal entries. To achieve efficient online inference over $(\sigma^2, \boldsymbol{\theta}_t)$, $\boldsymbol{W}_t$ is specified by a discount factor $\delta \in (0, 1]$, the same procedure used in our previous work [2] (see Section III-A for details). This process model is shared by the following three regimes.

b) SHORT: A SHORT fault is modelled as a transient error which does not correlate with the signal [1], therefore we use the following sensing model:

$$y_t = \boldsymbol{F}_0\boldsymbol{\theta}_t + v_t,\ v_t \sim \mathcal{U}(L, U), \tag{4}$$

where $\boldsymbol{F}_0 = [0]'_{p\times 1}$ vector, and $\mathcal{U}$ denotes a uniform distribution with the range $[L, U]$. The zero linear transformation implies the observation is independent of the signal.

The model assumes the reading is a random draw over the sensing range. Note that L and U can either be set by checking the sensor's hardware specification or learnt by data; and the estimation procedure is

$$L = min(L_0, y_{1:t}),\ \ U = max(U_0, y_{1:t}),$$

where L_0 and U_0 are some initial guess of the range. It is easy to show that the procedure finds the maximum a posteriori (MAP) estimate of a uniform distribution.

This model can be cast into a standard DLM as:

$$y_t = \boldsymbol{F}_0\boldsymbol{\theta}_t + v_t,\ v_t \sim \mathcal{N}(0, V_S), \tag{5}$$

where we approximate the uniform distribution by a Gaussian. One can minimise the Kullack-Leibler (KL) divergence between the two distributions to find the optimal V_S:

$$V_S = \operatorname*{argmin}_{V_S} D_{KL}(\mathcal{N}(0, V_S)||\mathcal{U}(L, U))$$

which leads to the following approximation:

$$V_S = \frac{1}{3}(U^2 + UL + L^2).$$

c) NOISE: NOISE is similar to the NORMAL state except it samples with a larger noise:

$$y_t = \boldsymbol{F}\boldsymbol{\theta}_t + v_t,\ v_t \sim \mathcal{N}(0, V_N\sigma^2) \tag{6}$$

where we have introduced a parameter $V_N \gg 1$ to denote the larger noise. A value between $[5, 10]$ is a good choice. Note that depending on the application, the user can introduce more than one noise model to accommodate various scales of noises, say $V_{\text{LN}} = 5$ and $V_{\text{HN}} = 10$, which extends to a 5 state model.

d) CONSTANT: We model CONSTANT as follows:

$$y_t = \boldsymbol{F}_0\boldsymbol{\theta}_t + y_{t-1} + v_t,\ v_t \sim \delta_0(\cdot) \tag{7}$$

where $\delta_0(\cdot)$ is a *Dirac* delta function on zero, which means a noise free observation of the previous reading y_{t-1}. Note that in practice, this model can be cast into a DLM as:

$$y_t = \boldsymbol{F}_0\boldsymbol{\theta}_t + v_t,\ v_t \sim \mathcal{N}(y_{t-1}, V_c),$$

where $\mu_t = y_{t-1}$ and $V_c = \epsilon$ is some very small value. It is easy to show the two distributions converge as $V_c \to 0$.

2) Transition matrix specification: In this section, we explain how the transition matrix $\boldsymbol{\Gamma}$ can be set by exploiting the sensor faults' properties. We explain each state based transition probability individually and the full transition matrix is given at the end.

a) SHORT: To model the transient behaviour, the self-transition probability $\boldsymbol{\Gamma}_{ii}$ is set to a small value, where $i =$ SHORT. The transition vector is set as follows $\boldsymbol{\Gamma}_{ii} = p_\epsilon$, the rest three states equally share the rest probability mass $\boldsymbol{\Gamma}_{ij} = 1/3(1 - p_s)$ for $i \neq j$. In practice, $p_\epsilon = .0001$ is used [3]. The equal sharing of the exit probability represents our ignorance over its transition preference.

b) CONSTANT (CONS.): To model the temporal correlation of CONS., *i.e.* a sensor might "stuck" for a while, we set the self transition probability to a large value: $\boldsymbol{\Gamma}_{ii} = p_s$ where $i =$ CONS. and $p_s \geq 0.8$. We find any value in the range $(0.8, 0.95)$ for p_s works well. The rest three states equally share the rest probability mass $\boldsymbol{\Gamma}_{ij} = 1/3(1 - p_s)$ for $i \neq j$.

---

[2] It is not strictly necessary but convenient to derive the general inference algorithm.

[3] p_ϵ is not set as zero for numerical stability

c) NORMAL and NOISE: For NOISE and NORMAL, according to their definitions [1], they both exhibit strong serial correlations; *i.e.*, a sensor is working fine (or reporting noisy readings) at time t, it is more likely to work properly (or stay noisy) at the next time step. Like the CONS. model, the self-transition probability is set to a large value: $\boldsymbol{\Gamma}_{ii} = p_s$ where $i =$ NORMAL or NOISE to reflect this behaviour.

The transition probability to CONS. from NORMAL or NOISE needs some explanation. Note that according to CONS.'s definition, it is essentially a SHORT error (or a jump) followed by a period of "stuck at" readings at that specific erroneous value. In other words, the previous stuck at value y_{t-1} can only be generated from a SHORT state but not the other two. In light of this, the transition probability to CONS. is set as zero, *i.e.* $\boldsymbol{\Gamma}_{ij} = p_\epsilon$ for $j =$ CONS.. The transition probabilities to the other states share the rest probability mass: $\boldsymbol{\Gamma}_{ij} = 1/2(1 - p_s - p_\epsilon)$ for $j \notin \{i, \text{CONS.}\}$.

d) Transition matrix: The full transition matrix $\boldsymbol{\Gamma}$ is listed in (8). Note that the only parameter left to be specified is the self-transition probability p_s and the matrix can be viewed as a function of p_s (p_ϵ is introduced for stability reasons, therefore can be safely set as a small value).

$$\boldsymbol{\Gamma} = \begin{array}{ccccc} & \text{NR} & \text{SH} & \text{NS} & \text{CN} \\ \text{NR} & p_s & \frac{(1-p_s-p_\epsilon)}{2} & \frac{(1-p_s-p_\epsilon)}{2} & p_\epsilon \\ \text{SH} & \frac{1-p_\epsilon}{3} & p_\epsilon & \frac{1-p_\epsilon}{3} & \frac{1-p_\epsilon}{3} \\ \text{NS} & \frac{(1-p_s-p_\epsilon)}{2} & \frac{(1-p_s-p_\epsilon)}{2} & p_s & p_\epsilon \\ \text{CN} & \frac{1-p_\epsilon}{3} & \frac{1-p_\epsilon}{3} & \frac{1-p_\epsilon}{3} & p_s \end{array} \tag{8}$$

This method greatly relieves human input and the further optimization procedure. For example, one can update p_s based on the inference results on S_t, which is similar to the E step procedure of an EM algorithm. Note that a full analysis of the matrix involves $K \times K$ parameters, and it can only be done either by a heavy iterative optimization algorithm or sampling algorithm [17]. Such a procedure also requires storage of the full dataset for off-line learning, which invalidates our on-line inference requirement.

C. Approximate on-line inference

In this section, we discuss how the inference on the given HMS-DLMs is done. We first give the problem statement of the inference task; then an approximate on-line one-pass inference algorithm is derived and presented.

1) Inference task: When a HMS-DLMs is specified, the unknown parameters of interests are: the hidden physical process signal $\boldsymbol{\theta}_{1:t}$, the sensor status $S_{1:t}$, and sensor noise for the NORMAL state σ^2 or precision ϕ. From an on-line inference perspective, the task is to calculate the contemporary posterior over the unknowns given sensor data up to t:

$$P(\boldsymbol{\theta}_t, S_t, \phi | y_{1:t}), \text{ for } t > 0, \tag{9}$$

based on its previous result $P(\boldsymbol{\theta}_{t-1}, S_{t-1}, \phi | y_{1:t-1})$. Comparing with singular DLMs (see Section III-A), we have to make additional inference on S_t.

Unfortunately, this problem has been shown to be NP-hard [20]. To see this, note that a HMS-DLMs with known hidden statues trace $S_{1:t}$ is a standard DLM (with switching parameters: $F_t = F^{[S_t]}, v_t = V^{[S_t]} \ldots$ and so on): therefore the on-line algorithm in Section III-A can be used to make exact inference for this case: $P(\boldsymbol{\theta}_t, \phi | y_{1:t}, S_{1:t})$. However, unconditionally, by probability theory, the inference becomes

$$P(\boldsymbol{\theta}_t, \phi | y_{1:t}) = \sum_{S_{1:t}} P(\boldsymbol{\theta}_t, \phi | S_{1:t}, y_{1:t}) P(S_{1:t} | y_{1:t}).$$

The posterior distribution can be considered as a mixture with K^t components. And the difficulty originates from the exponentially growing size of $S_{1:t}$, at a scale of $K \times \ldots \times K = K^t$.

2) Online inference algorithm: Approximate inference algorithms therefore have been widely studied. We employ a technique called Assumed Density Filter (ADF) here to achieve efficient inference [21], [14], [13]. Intuitively, this algorithm works by approximating the large K^t mixture as a fix-sized, say K^h mixtures ($h = 1, 2$ is usually good enough). The model essentially assumes $\boldsymbol{\theta}_t, \phi$ only depends on the past h steps' history rather than the whole trace $S_{1:t}$:

$$P(\boldsymbol{\theta}_t, \phi | y_{1:t}) \approx \sum_{S_{t-h:t}} P(\boldsymbol{\theta}_t, \phi | S_{t-h:t}, y_{1:t}) P(S_{t-h:t} | y_{1:t}).$$

The difference of our algorithm lies in integrating discount factor learning into the framework to facilitate on-line inference on ϕ at the same time.

In an overview, the derived algorithm consists of three steps: it first expands and updates the posterior belief on $\boldsymbol{\theta}_t^{S_{t-1}}$ given y_t, which leads to a $K \times K$ hypothesis conditional on S_{t-1}, S_t; the algorithm then updates the posterior on S_t, S_{t-1} given y_t; the last step is a collapsing step which reduces the model size from $K \times K$ to K (the collapsing step is based on minimising the KL divergence between the K mixture and the collapsed singular distribution [18]), making the model ready to be expanded again when a new observation arrives, which repeats the first step. Note that for notational convenience, we write the four states NORMAL, SHORT, NOISE, CONS. as $1, 2, 3, 4$ respectively. The algorithm is summarised below.

Step 0 initialisation: at $t = 0$, assume prior distributions on $\boldsymbol{\theta}_0^i$ for each $S_0 = i \neq 1$:

$$P(\boldsymbol{\theta}_0^i) = \mathcal{N}(\boldsymbol{m}_0^i, \boldsymbol{C}_0^i),$$

and a Gaussian Gamma prior for $S_0 = i = 1$ on $(\boldsymbol{\theta}_0^i, \phi)$:

$$P(\boldsymbol{\theta}_0^i, \phi) = \mathcal{NG}(\boldsymbol{m}_0^i, \boldsymbol{C}_0^i, n_0, s_0);$$

and a prior distribution on S_0: $P(S_0 = i)$ for $i = 1 \ldots K$. For $t > 1$, repeat the following steps

Step 1 (update on $\boldsymbol{\theta}_t^i$) For each $S_{t-1} = i, S_t = j$ pair, where $i, j = 1 \ldots K$ Update the posterior mean and variance on $\boldsymbol{\theta}_t^{(ij)}$:

$$\boldsymbol{m}_t^{(ij)} = \boldsymbol{a}_t^{(ij)} + \boldsymbol{K}_t^{(ij)} e_t^{(ij)},$$
$$\boldsymbol{C}_t^{(ij)} = \boldsymbol{R}_t^{(ij)} - \boldsymbol{K}_t^{(ij)} \boldsymbol{K}_t^{(ij)'} Q_t^{(ij)},$$

where

$$\begin{aligned} \boldsymbol{a}_t^{(ij)} &= \boldsymbol{G}_t \boldsymbol{m}_{t-1}^i, & \boldsymbol{R}_t^{(ij)} &= \delta^{-1} \boldsymbol{G}_t \boldsymbol{C}_{t-1}^i \boldsymbol{G}_t', \\ f_t^{(ij)} &= \mu_t^{[j]} \boldsymbol{F}_t^{[j]} \boldsymbol{a}_t^{(ij)}, & Q_t^{(ij)} &= \boldsymbol{F}_t^{[j]} \boldsymbol{R}_t^{(ij)} \boldsymbol{F}_t^{[j]'} + V_t^{[j]} \\ \boldsymbol{K}_t^{(ij)} &= \boldsymbol{R}_t^{ij} \boldsymbol{F}_t^{[j]'} / Q_t^{(ij)}, & e_t^{(ij)} &= y_t - f_t^{(ij)}. \end{aligned}$$

If $j = 1$, update the posterior on ϕ:

$$n_t = n_{t-1} + 1/2, \; s_t^{(i)} = s_{t-1} + \left(e_t^{(i1)} \right)^2 / 2Q_t^{(i1)}.$$

Step 2 (update on S_t) For each $S_{t-1} = i, S_t = j$, $i, j = 1 \ldots K$, update

$$P(S_t | y_{1:t}) = \sum_{s_{t-1}=1:K} P(S_t, S_{t-1} | y_{1:t}) \qquad (10)$$
$$P(S_{t-1} | y_{1:t}) = \sum_{s_t=1:K} P(S_t, S_{t-1} | y_{1:t}), \qquad (11)$$

where

$$\begin{aligned} P(S_t, S_{t-1} | y_{1:t}) &\propto P(S_t, S_{t-1} | y_{1:t-1}) \cdot \\ & f_j(y_t | f_t^{(ij)}, Q_t^{(ij)}, n_{t-1}, s_{t-1}); \\ P(S_t, S_{t-1} | y_{1:t-1}) &= \boldsymbol{\Gamma}_{ij} P(S_{t-1} | y_{1:t-1}) \end{aligned}$$

and f_j is the likelihood associated with each state j. For $j = 1, 3$, f_j are Student T distributions:

$$\mathcal{T}_{2n_{t-1}}(f_t^{(ij)}, Q_t^{(ij)} s_{t-1}/n_{t-1});$$

For $j = 2, 4$, f_j are Gaussians: $\mathcal{N}(f_t^{(ij)}, Q_t^{(ij)})$.

Step 3 (collapse) For each $j = 1, \ldots, K$ calculate the collapsed posterior mean and variance of $\boldsymbol{\theta}_t^j$:

$$\boldsymbol{m}_t^j = \sum_{i=1}^K p_{ij} \boldsymbol{m}_t^{(ij)}$$
$$\boldsymbol{C}_t^j = \sum_{i=1}^K p_{ij} (\boldsymbol{C}_t^{(ij)} + (\boldsymbol{m}_t^{(ij)} - \boldsymbol{m}_t^j)(\boldsymbol{m}_t^{(ij)} - \boldsymbol{m}_t^j)') \qquad (12)$$

Collapse on ϕ: $s_t = \left(\sum_{i=1}^K p_{ij} / s_t^{(i)} \right)^{-1}$, where

$$\begin{aligned} p_{ij} &= P(S_{t-1} = i | S_t = j, y_{1:t}) \\ &= \frac{P(S_t = i, S_{t-1} = j | y_{1:t})}{P(S_t = j | y_{1:t})} \end{aligned}$$

Repeat to step 1 when a new data sample is observed.

A few observations can be made from the algorithm.

- The inference result on the sensor state S_t is summarised in

  $$P(S_t | y_{1:t}) \text{ as defined in (10);}$$

  note that

  $$P(S_{t-1} | y_{1:t}) \text{ as defined in (11)}$$

  provides an alternative one step smoothed estimation which is a by-product of our h step approximation. The smoothed distribution is supposed to perform better than the filtered distribution as it includes one more sensor observation. See V-C2 for some empirical comparisons.
- The inference on the physical process $\boldsymbol{\theta}_t$ is summarised as

  $$P(\boldsymbol{\theta}_t | y_{1:t}) = \sum_{j=1}^K P(S_t = j | y_{1:t}) \mathcal{N}(\boldsymbol{m}_t^j, \boldsymbol{C}_t^j), \qquad (13)$$

  a mixture of K Gaussians. An intuitive explanation is: the distribution weights each hypothesis of the four states to give a final result. The mean and variance of this mixture distribution can be found in the same way as the collapsing step (12) if summary statistics are needed.
- The inference is completely on-line which requires no storage of historic sensor data and it scales at $O(K^h \times T)$ in time complexity; for a finite K (4 in our case) and $h = 2$, the complexity is linear.

3) Learning on discount factor δ: The discount factor $\delta \in (0, 1]$ dictates how the physical process $\boldsymbol{\theta}_t$ evolves. According to the conjugate learning algorithm listed in III-A, it quantifies the evolution noise by $\boldsymbol{W}_t = (1 - \delta)/\delta \boldsymbol{G}_t \boldsymbol{C}_{t-1} \boldsymbol{G}_t'$. Therefore, it essentially provides a signal-noise-ratio decomposition between $\boldsymbol{W}_t$ and σ^2, which can greatly affect the inference result. To see this, when $\delta \to 1$, $\boldsymbol{\theta}_t$ degenerates to a constant vector, and most of the variance of y_t will be explained by the sensor noise σ^2; and the opposite applies when $\delta \to 0$.

To resolve this problem, we introduce an initialisation step on δ based on the first batch of samples. In practice, it can either be historic data or some initial samples at the beginning of the deployment.

The idea is to find the best δ that explains the observed data, and we adopt a Bayesian inference procedure to find the best δ. Given a finite choices on $\delta \in \{\delta_d, d = 1 \ldots D\}$. And a prior on $P(\delta = \delta_d)$, the posterior distribution can be obtained as follows:

$$\begin{aligned} P(\delta | y_{1:J}) \propto P(\delta) P(y_{1:J} | \delta) &= P(\delta) \prod_{t=1}^J P(y_t | y_{1:t-1}, \delta) \\ &= P(\delta) \prod_{t=1}^J \mathcal{T}_{2n_{t-1}}(y_t | f_t^\delta, Q_t s_{t-1}^\delta / n_{t-1}) \qquad (14) \end{aligned}$$

where J denotes the initial sample size and the the required parameters $\{f_t, Q_t, n_{t-1}, s_{t-1}\}$ are readily available through the on-line inference algorithm (see III-A). For computational efficiency, one can simply use the sum of the squared errors $\sum_{t=1}^{J}(y_t - f_t)^2 n_{t-1}/Q_t s_{t-1}$ to replace the (log-)likelihood term, where we use the fact that a Gaussian approximates the student T distribution when the degree of freedom increases. The best $\delta = \text{argmax}_d\, P(\delta = \delta_d|y_{1:T})$ is then used for further analysis, the chosen δ is actually the maximum a posteriori (MAP) estimate. The whole initialisation step is still a one-pass algorithm, and scales linearly with J as long as D is finite. In practice, δ_d can be set as a sequence from 0.5 to 0.9 inclusive with an equal step of 0.1 and the prior can either be uniform or simply a posterior from some previous analysis.

V. EVALUATION

In this section, we evaluate the proposed solution on synthetically generated data and real sensor data. The synthetic analysis aims to empirical access the accuracy of the proposed inference algorithm; while the real sensor data analysis compare our solution with some popular baselines.

A. Implementation and Baselines

The proposed solution is implemented in R and the code is made publicly available[4]. To compare the performance on time series clustering on S_t, we compare with the following baselines:

- Hidden Markov Models (HMM) with Gaussian emissions, which is similar to the solution proposed in [1] [5]; depmixS4 package in R is used [22].
- K-means: sensor series are segmented into fixed sizes and the mean, variance, minimum, and maximum are extracted for clustering;
- Hierarchical Clustering (H-Clust): the complete linkage is used on the same data generated in the K-means method.

Note that all the above methods are off-line which requires a model training step. And the clustering size is set as the true size: 4.

A few variants of the proposed HMS-DLMs are also compared. In particular,

- Singular Dynamic Linear Model (DLM), which corresponds to our previous work [2];
- Mixture of Dynamic Linear Models (M-DLMs): S_t are assumed serially independent;
- HMS-DLMs ($h = 1$): a simplified approximate inference, where $h = 1$ history is used for approximation;
- HMS-DLMs ($h = 2$): the proposed solution, as presented in Section IV-C.

B. Evaluation Metrics

Inference on S_t: To evaluate the unsupervised learning performance on S_t, we use four commonly used measures for clustering [23]: normalised mutual information (NMI), adjusted Rand Index (ARI) and entropy (ENTR). All of them except entropy ranges from 0 to 1 measuring similarities between two clustering results. To assess the classification accuracy, accuracy (ACC), balanced accuracy (ACC$_b$) and by class averaged F-measures are reported.

Inference on $\boldsymbol{\theta}_t$: Three metrics are used to assess the signal inference accuracy: mean squared error (MSE), mean absolute deviation (MAD) and mean absolute percentage error (MAPE):

$$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2, \text{MAD} = \frac{1}{n}\sum_{t=1}^{n} |e_t|, \text{MAPE} = \frac{1}{n}\sum_{t=1}^{n} \frac{|e_t|}{|\theta_t|}$$

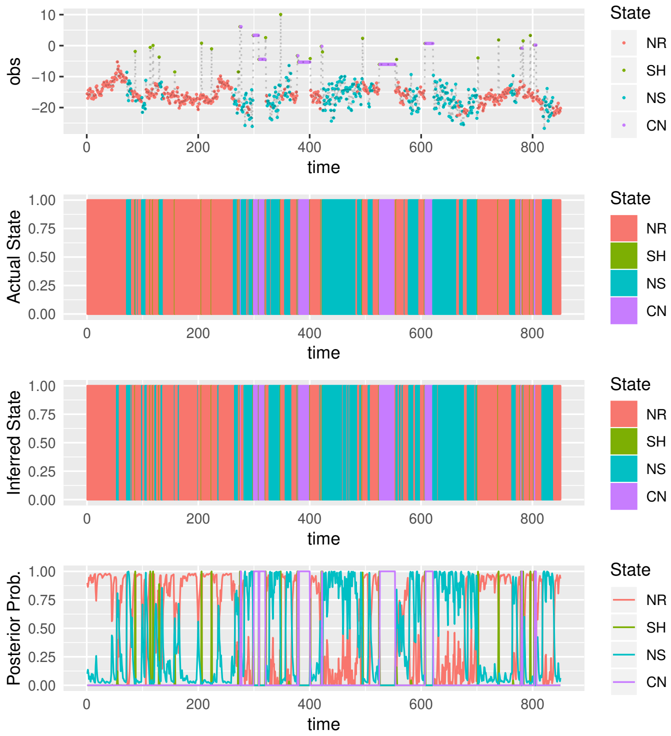

Fig. 4. Online inference results on the sensor status S_t where NR, SJ, NS, and CN represent NORMAL, SHORT, NOISE and CONST. respectively. The four diagrams are the sensor readings y_t, the true state of S_t, online inferred S_t by the proposed solution and the posterior probability $P(S_t|y_{1:t})$. In this case, the discount factor is set as 0.9.

C. Synthetic data analysis

In this section, we run the derived approximate inference algorithm on simulated data. The data is generated by the HSM-DLMs as specified in Section IV [6]. As an example, check the top row in Fig. 4, in which a series of synthetically

---

[4] https://leo.host.cs.st-andrews.ac.uk

[5] In their paper, they used supervised learning to train the HMM, and the labels are provided by artificially injected errors. In reality, the labels are not available in general.

[6] In particular, given S_0 and $\boldsymbol{\theta}_0$; for $t > 0$, S_t is simulated based on S_{t-1} and $\boldsymbol{\Gamma}$; then $\boldsymbol{\theta}_t$ is simulated based on S_t and $\boldsymbol{\theta}_{t-1}$; then finally y_t is simulated based on S_t and $\boldsymbol{\theta}_t$.

TABLE I

ASSESSMENT ON HMS-DLMS WITH THE PROPOSED δ LEARNING PROCEDURE. THE RESULTS ARE MEASURED ON THE INFERENCE ON S_t

Meas.\ Model	True Vars	δ_{MAP}	$\delta = 0.95$	$\delta = 0.9$	$\delta = 0.6$	$\delta = 0.3$	$\delta = 0.05$
ACC	0.871 (0.043)	**0.873 (0.037)**	0.811 (0.092)	0.818 (0.087)	0.851 (0.049)	0.861 (0.031)	0.833 (0.038)
F-measure	**0.856 (0.035)**	0.850 (0.035)	0.763 (0.092)	0.771 (0.079)	0.820 (0.042)	0.829 (0.046)	0.787 (0.077)
ACC$_b$	**0.886 (0.024)**	0.878 (0.024)	0.817 (0.064)	0.824 (0.058)	0.856 (0.032)	0.868 (0.027)	0.854 (0.034)
ARI	0.624 (0.105)	**0.630 (0.095)**	0.519 (0.185)	0.531 (0.178)	0.582 (0.113)	0.601 (0.078)	0.55 (0.069)
NMI	**0.618 (0.059)**	**0.618 (0.059)**	0.576 (0.089)	0.581 (0.086)	0.595 (0.060)	0.595 (0.052)	0.541 (0.055)
ENTR	**0.268 (0.038)**	0.275 (0.043)	0.334 (0.086)	0.327 (0.082)	0.300 (0.050)	0.291 (0.037)	0.314 (0.035)

TABLE II

ASSESSMENT ON HMS-DLMS WITH THE PROPOSED δ LEARNING PROCEDURE.; THE RESULTS ARE MEASURED ON THE INFERENCE ON $\boldsymbol{\theta}_t$

Meas.\ Model	True Vars	δ_{MAP}	$\delta = 0.95$	$\delta = 0.9$	$\delta = 0.6$	$\delta = 0.3$	$\delta = 0.05$
MSE	**2.505 (3.353)**	3.354 (5.14)	8.195 (10.232)	5.975 (7.571)	3.322 (3.859)	3.717 (3.61)	5.648 (3.948)
MAD	**0.828 (0.53)**	0.911 (0.613)	1.671 (1.411)	1.414 (1.145)	1.089 (0.523)	1.165 (0.33)	1.436 (0.316)
MAPE	**1.396 (2.96)**	1.606 (4.075)	1.723 (3.744)	1.716 (3.944)	3.89 (17.865)	6.109 (32.268)	8.164 (44.75)

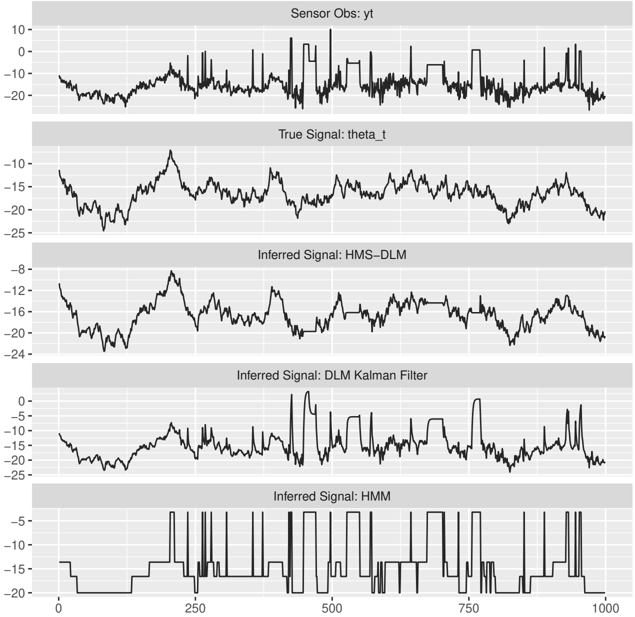

Fig. 5. Online inference results on the physical process $\boldsymbol{\theta}_t$. The five diagrams are the sensor observations, the actual physical process, online inferred signal by HMS-DLMs, singular DLM, and HMM.

generated data is listed. The four states are coloured to make the four regimes distinctive.

The main objective for this section is to verify our derived approximation algorithm works when data generated from a HSM-DLMs is given. Note that synthetic data analysis is the only way to obtain the ground truth on both S_t and $\boldsymbol{\theta}_t$. We also want to access how the discount factor conjugate learning, a building block of our method, can fulfil the required inference task.

1) A running example: To better understand the proposed solution, we pick up one particular dataset and show how the proposed solution responds to the two inference tasks. See Table III and IV for repeated experiment results.

Inference on S_t: As an illustrative example, the inference on S_t is shown in Fig 4. The true states are shown in the second row while the inferred state by the one-step smoothed probability: $S_t = \text{argmax}_k\, P(S_t = k|y_{1:t+1})$ is shown below, and the posterior probability $P(S_t = k|y_{1:t+1})$ is shown in the last row. By visual inspection, the approximation on-line inference has done a decent job, even the discount factor δ is not optimised and set as the default value 0.9. Note that there still exist some misclassification, especially between the NOISE and NORMAL.

TABLE III

THE EFFECT OF THE APPROXIMATION SIZE h; AND COMPARISON BETWEEN FILTERING AND SMOOTHING INFERENCE ON S_t

Meas.\Method	**HMS-DLMs,** ($h = 2$)		**HMS-DLMs,**
	$S_t\|y_{1:t}$	$S_t\|y_{1:t+1}$	($h = 1$) $S_t\|y_{1:t}$
ACC	0.854 (0.033)	**0.876 (0.035)**	0.756 (0.129)
F-m	0.802 (0.060)	**0.857 (0.037)**	0.723 (0.107)
ACC$_b$	0.851 (0.032)	**0.883 (0.025)**	0.786 (0.075)
ARI	0.585 (0.083)	**0.636 (0.090)**	0.443 (0.198)
NMI	0.573 (0.055)	**0.625 (0.058)**	0.504 (0.104)
ENTR	0.307 (0.039)	**0.270 (0.043)**	0.359 (0.080)

Inference on $\boldsymbol{\theta}_t$: As the data is simulated, the ground truth signal can be compared with the inferred $P(\boldsymbol{\theta}_t|y_{1:t})$. As an illustrative example, Fig. 5 shows the inference results on the same data that generates Fig 4. The five pictures are the sensor readings, true signal, inferred results by HMS-DLMs, singular Kalman Filter, and HMM respectively. It can be seen HMS-DLMs has automatically ignored the influence of those erroneous observations, which is achieved with no human intervention. The automatic robust inference over the hidden signal is a key feature of HSM-DLMs. The results on Kalman Filter and HMM, however, deviate significantly from the ground truth. To achieve a more robust estimation, one needs to device extra rules, say setting some *ad hoc* confidence interval range, to specifically tell the Kalman Filter or HMM to ignore some values.

2) Approximation size h : Our approximate algorithm assumes the exponentially growing mixture can be summarised by a K^h mixtures. Therefore, h can affect the inference accuracy. To show $h = 2$ is a good approximation, we compare two approximation settings: $h = 1$ and 2. The results are shown in Table III and IV. As expected, $h = 2$ achieves better results for both inference tasks. More details are presented below.

Filtering and smoothing inference on S_t: For the $h = 2$ case, we listed the filtering and smoothing results in Table III: the smoothing results are based on $P(S_t|y_{1:t+1})$ (See (11)), which is a by-product of the $h = 2$ approximation (note that there is no equivalent smoothing distribution for the $h = 1$ case as it only keep one step expansion in the online inference procedure). Unsurprisingly, the one step smoothed results are better than the filtering version $P(S_t|y_{1:t})$.

Effects on $\boldsymbol{\theta}_t$: Similarly, the inference results on the hidden physical process $\boldsymbol{\theta}_t$ under the same setting are shown in Table IV. The gap between the $h = 1$ and $h = 2$ cases are even larger than the S_t case. It is because the one step approximation $h = 1$ is not enough to capture all the dynamics of the hidden space of S_t. We also list the results of HMM and Kalman Filter for reference: when the approximation is too crude, the performance degenerates to homogeneous models.

3) Learning on discount factor δ : To access the effect of δ on the inference, we simulate datasets based on different signal-noise-ratios (varies from 0.1 to 10). The objective is to check whether the conjugate inference via the discount factor technique can cope with the complexity, especially when it is employed in our approximate inference algorithm. For each instance, we compare the following different settings: a baseline method with the true variances (denoted as “True Vars” in the table); δ_{MAP} denotes our proposed on-line δ learning solution; also results with δ in the set of $\{0.05, 0.3, 0.6, 0.9, 0.95\}$ are compared. Note that the “True Var” case is only presented here for reference as both the variances are unknown in reality. For each signal-noise-ratio, we run 10 independent experiments, and the average and standard deviations of the inference results on S_t over all settings and repetitions are listed in Table I. It can be observed that all different δs work well in terms of the classification accuracy which means they all can be used to do on-line faults classification, although our discount factor learning method consistently outperforms the others and achieves similar performance with the truth parameter case. The inference results on $\boldsymbol{\theta}_t$ under the same experiment setting are listed Table II. The results show a similar pattern on the signal inference task.

D. Real sensor data analysis

Now we use real world sensor data to access our solution. The objective here is to access whether our proposed model, especially the model assumption, meets the reality. As there is no ground truth, the only way to perform large scale repetitive experiments is to generate and inject artificial errors. And this has been a common approach to measure the accuracy of a fault detection algorithm in WSNs community [1][24][25], although results vary greatly subjects to the choice of injection parameters. And there is no uniformly accepted injection method. That’s the main reason we focus on synthetic data analysis, which actually provides a more objective assessment.

In this paper, we adopt exactly the same injection method and parameters as used in [1]. The methods are listed below:

- SHORT: $\hat{y}_t = y_t + f \times v_i$, $f \in \{1.5, 2.5, 10\}$;
- NOISE: $\hat{y}_{t:T} = y_t + n$, $n \sim \mathcal{N}(0, V\sigma^2)$, where $V \in \{0.5, 1.5, 3\}$ and σ^2 is the variance of $y_{t:T}$;
- CONSTANT: $\hat{y}_{t:T} = c$, where $c = y_t + f \times y_t$, $f \in \{1.5, 2.5, 10\}$.

The parameters to generate the faults are randomly picked from the given choices. Temperature sensor data from a local deployment in Grangemouth is used. The results are listed in Table V. We compare HSM-DLMs with its variants and also some state of art clustering algorithms, like K-means and HMM. It is obvious the proposed solution with δ learning dominates all metrics especially comparing with HMM, K-means and H-clust. Comparing with M-DLMs, our proposed hidden markov state model is proved to be better; while the $h = 2$ approximation strikes the balance between efficiency and accuracy when compared with the crude $h = 1$ approximation method.

VI. CONCLUSION

In this paper, we have proposed a hybrid Bayesian network based model to solve an on-line inference problem for wireless sensor networks. The model is rich enough to capture the distinctive statistical patterns of the sensor faults which are widely found in real world deployments. The specification of the model is built upon the statistical properties of the faults but also can adapt itself to the data when required, which greatly minimise the learning and modelling efforts. A very efficient on-line one pass algorithm is derived to make formal inference on the proposed model. The inference results on both simulation and real sensor data studies are very promising.

There are a few possible extensions to consider as future work. We have presented a uni-variate sensor model in this work, and it would be interesting to accommodate the multivariate case. The complexity however might be too much to handle for local sensors as the sensor status space grow exponentially with the dimensionality of the sensor variates.

TABLE IV

THE EFFECT OF THE APPROXIMATION SIZE h ON INFERENCE OF THE PHYSICAL PROCESS $\boldsymbol{\theta}_t$

Meas.\Method	**HMS-DLMs** ($h = 2$)	**HMS-DLMs** ($h = 1$)	**DLM (Kalman Filter)**	**HMMs**
MSE	**3.148 (4.64)**	84.364 (187.774)	87.755 (191.333)	65.623 (160.774)
MAD	**0.934 (0.553)**	3.787 (5.197)	3.996 (3.709)	2.539 (2.182)
MAPE	**1.378 (3.672)**	1.954 (4.391)	2.026 (3.235)	2.198 (4.356)

TABLE V

INFERENCE RESULTS ON S_t ON REAL WORLD SENSOR DATA.

	HMS-DLMs ($\delta_{\mathbf{MAP}}$)	**HMS-DLMs** ($\delta = 0.9$)	**HMS-DLMs**, $h = 1$	**M-DLMs**	**HMMs**	**K-means**	**H-Clust**
ACC	**0.871 (0.029)**	0.79 (0.064)	0.716 (0.114)	0.789 (0.027)	0.4 (0.051)	0.482 (0.066)	0.489 (0.064)
F-m	**0.869 (0.035)**	0.764 (0.074)	0.709 (0.124)	0.749 (0.063)	0.359 (0.085)	0.37 (0.052)	0.387 (0.057)
ACC$_b$	**0.888 (0.027)**	0.809 (0.048)	0.782 (0.079)	0.801 (0.036)	0.589 (0.057)	0.61 (0.052)	0.62 (0.052)
ARI	**0.618 (0.077)**	0.454 (0.119)	0.373 (0.156)	0.441 (0.067)	0.052 (0.042)	0.077 (0.055)	0.09 (0.065)
NMI	**0.616 (0.058)**	0.534 (0.078)	0.467 (0.09)	0.472 (0.063)	0.127 (0.063)	0.117 (0.052)	0.134 (0.053)
ENTR	**0.275 (0.042)**	0.363 (0.056)	0.379 (0.071)	0.377 (0.043)	0.578 (0.06)	0.598 (0.052)	0.584 (0.051)

On the other hand, how to handle fault correlations within the multivariate is challenging. We also plan to apply the method in wild to see how it works in reality.

REFERENCES

[1] A. B. Sharma, L. Golubchik, and R. Govindan, “Sensor faults: Detection methods and prevalence in real-world datasets,” *ACM Transactions on Sensor Networks*, vol. 6, no. 3, p. 23, 2010.

[2] L. Fang and S. Dobson, “Towards data-centric control of sensor networks through bayesian dynamic linear modelling,” in *2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems*, Sep. 2015, pp. 61–70.

[3] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. Cambridge, MA: MIT press, 2009.

[4] R. Tan, G. Xing, Z. Yuan, X. Liu, and J. Yao, “System-level calibration for data fusion in wireless sensor networks,” *ACM Trans. Sen. Netw.*, vol. 9, no. 3, pp. 28:1–28:27, Jun. 2013. [Online]. Available: http://doi.acm.org/10.1145/2480730.2480731

[5] D. Kumar, S. Rajasegarar, and M. Palaniswami, “Geospatial estimation-based auto drift correction in wireless sensor networks,” *ACM Trans. Sen. Netw.*, vol. 11, no. 3, pp. 50:1–50:39, Apr. 2015. [Online]. Available: http://doi.acm.org/10.1145/2736697

[6] L. Fang and S. Dobson, “Data Collection with In-network Fault Detection Based on Spatial Correlation,” in *Proceedings of the 2nd International Conference on Cloud and Autonomic Computing (CAC 2014)*, ser. CAC ’14. IEEE, 2014.

[7] E. Miluzzo, N. D. Lane, A. T. Campbell, and R. Olfati-Saber, “Calibree: A self-calibration system for mobile sensor networks,” in *Proceedings of the 4th IEEE International Conference on Distributed Computing in Sensor Systems*, ser. DCOSS ’08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 314–331. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-69170-9_21

[8] U. Lerner, R. Parr, D. Koller, G. Biswas *et al.*, “Bayesian fault detection and diagnosis in dynamic systems,” in *Aaai/iaai*, 2000, pp. 531–537.

[9] S. Chib and M. Dueker, “Non-markovian regime switching with endogenous states and time-varying state strengths,” 2004.

[10] A. T. Cemgil, H. J. Kappen, and D. Barber, “A generative model for music transcription,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 2, pp. 679–694, 2006.

[11] B. Mesot and D. Barber, “Switching linear dynamical systems for noise robust speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 6, pp. 1850–1858, 2007.

[12] E. B. Fox, E. B. Sudderth, M. I. Jordan, A. S. Willsky *et al.*, “A sticky HDP-HMM with application to speaker diarization,” *The Annals of Applied Statistics*, vol. 5, no. 2A, pp. 1020–1056, 2011.

[13] C.-J. Kim, “Dynamic linear models with markov-switching,” *Journal of Econometrics*, vol. 60, no. 1-2, pp. 1–22, 1994.

[14] T. P. Minka, “A family of algorithms for approximate bayesian inference,” Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[15] A. Doucet, N. De Freitas, K. Murphy, and S. Russell, “Rao-blackwellised particle filtering for dynamic bayesian networks,” in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2000, pp. 176–183.

[16] C. M. Carvalho, M. S. Johannes, H. F. Lopes, N. G. Polson *et al.*, “Particle learning and smoothing,” *Statistical Science*, vol. 25, no. 1, pp. 88–106, 2010.

[17] S. Frühwirth-Schnatter, *Finite Mixture and Markov Switching Models: Modeling and Applications to Random Processes*. New York, NY, USA: Springer, 2006.

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

[19] L. Fang and S. Dobson, “In-Network Sensor Data Modelling Methods for Fault Detection,” in *Evolving Ambient Intelligence*. Springer, 2013, pp. 176–189. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-319-04406-4_17

[20] U. Lerner and R. Parr, “Inference in hybrid networks: Theoretical limits and practical algorithms,” in *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2001, pp. 310–318.

[21] X. Boyen and D. Koller, “Tractable inference for complex stochastic processes,” in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1998, pp. 33–42.

[22] I. Visser and M. Speekenbrink, “depmixS4: An R package for hidden markov models,” *Journal of Statistical Software*, vol. 36, no. 7, pp. 1–21, 2010. [Online]. Available: http://www.jstatsoft.org/v36/i07/

[23] D. M. Christopher, R. Prabhakar, and S. Hinrich, “Introduction to information retrieval,” *An Introduction To Information Retrieval*, vol. 151, no. 177, p. 5, 2008.

[24] J. Gupchup, A. Sharma, A. Terzis, A. Burns, and A. Szalay, “The Perils of Detecting Measurement Faults in Environmental Monitoring Networks,” in *Proceedings of DCOSS*, 2008.

[25] L. Fang and S. Dobson, “Unifying sensor fault detection with energy conservation,” in *Self-Organizing Systems*, ser. IWSOS ’13. Berlin Heidelberg: Springer, 2013, pp. 176–181. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-54140-7_18