Leveraging Spatial Correlation for Sensor Drift Calibration in Smart Building

Tinghuan Chen, *Graduate Student Member, IEEE*, Bingqing Lin, Hao Geng, *Graduate Student Member, IEEE*, Shiyan Hu, *Senior Member, IEEE*, and Bei Yu, *Member, IEEE*

***Abstract*—Sensor drift is an intractable obstacle to practical temperature measurement in smart building. In this article, we propose a sensor spatial correlation model. Given prior knowledge, maximum *a posteriori* (MAP) estimation is performed to calibrate drifts. MAP is formulated as a nonconvex problem with three hyper-parameters. An alternating-based method is proposed to solve this nonconvex formulation. Cross-validation, Gibbs expectation-maximization (EM) and variational Bayesian EM (VB-EM) are further exploited to determine hyper-parameters. Experimental results on widely used benchmarks from the simulator `EnergyPlus` demonstrate that compared with state-of-the-art methods, the proposed framework can achieve a robust drift calibration and a better tradeoff between accuracy and runtime. On average, compared with state-of-the-art, the proposed framework can achieve about 3× accuracy improvement. In order to attain the same drift calibration accuracy with VB-EM, Gibbs EM needs 10 000 samples, which will incur a 30× runtime overhead.**

***Index Terms*—Bayesian inference, optimization, sensor calibration.**

I. Introduction

IN MODERN smart building, the temperature measurement is a key step for smart temperature management implemented by a cyber-physical system (CPS) [1]–[4]. CPS is a complex, heterogeneous distributed system with seamlessly integrated and closely interacted cyber components (e.g., sensors, sink nodes, control centers, and actuators) and physical processes (e.g., temperature) to achieve stability, high performance, robustness [5]. The physical world is sensed by corresponding sensors and the acquired data is sent to a sink node or control center. The sink node or control center will then send an instruction to actuators to control the physical world after the data is analyzed [6]. In a smart building, the in-building temperatures are monitored by several spatially distributed and immovable temperature sensors.

Part Number	Temp. Range	Accuracy	Price [9]
SMT172	$-45 \sim 130$ °C	± 0.25 °C	\$ 35.13
AD590JH	$-50 \sim 150$ °C	± 0.50 °C	\$ 17.91
TMP100	$-55 \sim 125$ °C	± 2.00 °C	\$ 1.79
MCP9509	$-40 \sim 125$ °C	± 4.50 °C	\$ 0.88
LM335A	$-40 \sim 100$ °C	± 5.00 °C	\$ 0.75

(a)

(b)

(c)

Fig. 1. (a) Comparison of different temperature sensors. (b) Sensor MCP9509. (c) Sensor LM335A.

Although advanced technologies in the semiconductor industry and micro-electromechanical systems are developed in recent years, in practice, sensors outputs exist errors, which are one of the major barriers to the use of sensor networks. There are three main types of errors: 1) gain; 2) drift; and 3) noise [7]. Compared with gain and noise, the sensor drift is considered with vital importance since it has a significantly negative effect on measurement accuracy [8]. Although sensors with high accuracy can be deployed, these sensors always have expensive prices. As shown in Fig. 1(a), the temperature sensor AD590JH with ± 0.5 °C accuracy is sold at more than tenfold price of TMP100 with ± 2 °C accuracy.

The sensor calibration is studied in many previous research works. These works can be classified into micro- and macro-calibration or nonblind, semi-blind, and blind calibration [10], [11]. The micro-calibration is a cumbersome method that each sensor is individually tuned so that the ground-truth data can be recovered from measurements [12].

In the macro-calibration schemes, data from uncalibrated sensors are collected to optimize the overall system performance. Therefore, macro-calibration schemes are widely used in practice. For macro-calibration schemes, according to information whether they need, there are three categories of sensor calibration: 1) nonblind; 2) semi-blind; and 3) blind. In the nonblind calibration, one or more than one *prior* knowledge is adopted to calibrate measurements (e.g., the ground-truth data measured by some of the sensors with high accuracy [13] and the distances between sensors and the sink node [14]). However, extra costs are required to obtain this information. For example, several sensors with high accuracy

Manuscript received March 4, 2020; revised June 11, 2020; accepted July 12, 2020. Date of publication August 12, 2020; date of current version June 18, 2021. This work was supported in part by the Research Grants Council of Hong Kong SAR under Project CUHK24209017. The preliminary version, titled "Sensor Drift Calibration via Spatial Correlation Model in Smart Building," has been presented at the ACM/IEEE Design Automation Conference (DAC) in 2019. This article was recommended by Associate Editor D. Atienza. *(Corresponding author: Bei Yu.)*

Tinghuan Chen, Hao Geng, and Bei Yu are with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong (e-mail: byu@cse.cuhk.edu.hk).

Bingqing Lin is with the College of Mathematics and Statistics, Shenzhen University, Shenzhen 518060, China.

Shiyan Hu is with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K.

Digital Object Identifier 10.1109/TCAD.2020.3015438

have to be deployed to obtain ground truth data, and a Global Position System module is installed in sensors.

In order to reduce extra costs, some works focused on semi-blind calibration schemes. In [15], only partial position information and the ground-truth data are required to calibrate all measurements. In order to further decrease extra costs, blind calibration schemes were proposed.

In literature about the blind calibration, there are two models to estimate measurement errors: 1) the first-order autoregression (AR) model and 2) the signal space projection (SSP) model. For the AR model, assuming that errors are a time series following Gaussian distribution with zero-mean and constant variance, adaptive filters, such as Kalman filter [16], [17], unscented Kalman [18] filter, and particle filter are adopted to track the error in each time slot so that the measurement can be calibrated [16], [18]. In the AR model, however, it is assumed that only one sensor has measurement error in each time instant. In fact, this assumption is hard to satisfy in practice.

The calibration problem is naturally studied extensively to be a sparse reconstruction problem, where a sparse set of sensors are assumed to have significant drifts. Balzano and Nowak [19] first proposed SSP, in which the measured data are over-sampled by sensors. In other words, the sensor number is more than the variance source number. Therefore, the ground-truth vector has one or more than one null-space. Assuming that sensors are free-error in the initial time, this null space can be obtained by principal component analysis or independent component analysis [20]. In addition, assuming that measurement errors are sparse, the error can be estimated by sparse regression techniques, such as the maximum likelihood estimation [21] and deep learning [22]. Wang *et al.* [21] adopted temporal sparse Bayesian learning (TSBL) [23] to calibrate time-variant and incremental drifts for the sparse set of sensors. However, due to the sparsity assumption, not all sensors can be calibrated. In addition, since the observation matrix is directly determined by drift-free measurement, the method cannot calibrate drifts if signals lie in a time-variant subspace.

Very recently, in order to calibrate all sensors, Ling and Strohmer [24] presented three models, which are formulated as bilinear inverse problems. However, these models heavily rely on partial information about the sensing matrix. For the temperature sensor calibration in a smart building, the sensing matrix depends on the weather, the position of sensors and parameters of the building, e.g., material characteristics, geometry, and equipment power per area [1], [2], [25]. In practice, it is hard to obtain this complex and tedious information. As a result, these models cannot be directly used to calibrate temperature sensors in a smart building.

Bayesian inference is a very useful theoretical tool in machine learning and modeling. In recent years, Bayesian inference has been introduced for sensor calibration [15], [21], [26]–[30]. The extra location information, as prior knowledge, is obtained by the Global Position System and then used to perform Bayesian inference [15]. In [27] and [29], Bayesian inference with Gibbs sampling and particle filtering is developed to calibrate the sensor gain and drift. A Cal-AMP algorithm is proposed to perform blind calibration for unknown gains on the sensors [28]. A sensor spatial correlation model was proposed to calibrate drift in smart building [26], [30]. In this method, maximum *a posteriori* (MAP) is formulated as a nonconvex problem, which is handled by alternating-based optimization. Cross-validation and Gibbs expectation-maximization (EM) are used to determine hyper-parameters. Note that we also study stochastic Gibbs EM, but it cannot achieve stable and accurate performance.

In this article, we make the following contributions.

1) Due to high sensor spatial correlation, we build a sensor spatial correlation model whose coefficients only depend on measurements.
2) Given the prior information from drift-free measurement model coefficients, based on MAP estimation, we formulate sensor drift calibration problem to be a nonconvex problem with three hyper-parameters.
3) We propose an alternating-based optimization algorithm to handle the nonconvex formulation efficiently and effectively.
4) Cross-validation, a grid search method, is adopted to determine hyper-parameters.
5) In order to make a tradeoff between accuracy and runtime, we propose two EM-based methods to determine hyper-parameters. Compared with Gibbs EM, a stochastic method, our proposed variational Bayesian EM (VB-EM), a deterministic method, can achieve a better tradeoff between runtime and accuracy.
6) Experimental results show that on benchmarks simulated from `EnergyPlus`, compared with state-of-the-art methods, the proposed framework with the VB-EM method can achieve a better tradeoff between accuracy and runtime. In order to achieve the same drift calibration accuracy with VB-EM, Gibbs EM requires 10 000 samples, which will lead to a $30\times$ runtime overhead.

The remainder of this article is organized as follows. In Section II, we provide a problem formulation about sensor drift calibration. In Section III, we build a drift calibration model based on sensor spatial correlation and deliver mathematical formulation with three hyper-parameters. In Section IV, we propose a more efficient alternating-based method to handle the mathematical formulation. In Section V, three hyper-parameters are determined by cross-validation, Gibbs EM and VB-EM. In Section VI, we broadly introduce our proposed whole flow. Section VII presents experimental results with comparison and discussion, followed by the conclusion in Section VIII.

II. Preliminary

In this section, we will introduce some backgrounds in sensor drift calibration.

In smart building, several low-cost sensors are deployed to sense in-building temperatures. Furthermore, all sensors are nonremovable once they are deployed in smart building. Besides, due to a slow-aging effect, all sensors have unknown time-invariant drifts. As shown in Fig. 2, unlike communication channels [23], for a sensor signal to be output, i.e., current (I_{out}), it is contaminated by a time-invariant drift. Sensor 1 has

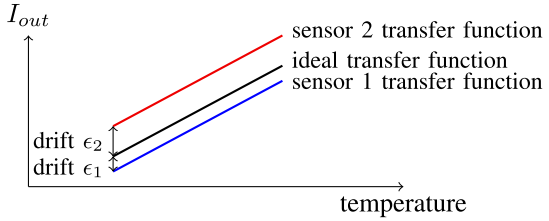

Fig. 2. Drift versus temperature [31].

a drift ϵ_1 so that its transfer function shifts downward denoted by the blue line in Fig. 2. While Sensor 2 has a drift ϵ_2 so that its transfer function shifts upward denoted by the red line in Fig. 2. In order to achieve high-accurate measurements, drifts need to be estimated and calibrated. Based on the above description, we define the sensor drift calibration problem as follows.

Problem 1 (Sensor Drift Calibration): Given the measurement values sensed by all sensors during several time-instants, drifts will be accurately estimated and calibrated.

III. Mathematical Formulation

We assume that n sensors are deployed to sense in-building temperatures in a smart building. During a short time after new sensors are deployed, drifts are assumed to be insignificant in all sensors. Furthermore, as in [21], we assume that all sensors are drift-free during m_0 initial time-instants. Due to over-sampling, as illustrated in [21] and [23], signals measured by sensors lie in a low dimensional subspace. Furthermore, in a smart building, all actual temperatures measured by sensors have a high correlation, for example, the dense deployment of sensors. Therefore, we build a linear model among all actual temperatures as follows:

$$x_i^{(k)} \approx \sum_{j=1,j\neq i}^{n} a_{i,j}x_j^{(k)} + a_{i,0}, \qquad k = 1, 2, \ldots, m_0 \tag{1}$$

where $x_i^{(k)}$ is the ground-truth temperature sensed by ith sensor at kth time-instant. $a_{i,j}$ is the drift-free model coefficient. We define $\boldsymbol{x} \triangleq [x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}, \ldots, x_n^{(m_0)}]^\top$, $\boldsymbol{a}_i \triangleq [a_{i,0}, \ldots, a_{i,i-1}, a_{i,i+1}, \ldots, a_{i,n}]^\top \in \mathbb{R}^n$, $\boldsymbol{a} \triangleq [\boldsymbol{a}_1^\top, \boldsymbol{a}_2^\top, \ldots, \boldsymbol{a}_n^\top]^\top \in \mathbb{R}^{n^2}$.

Due to a slow-aging effect, all sensors have unknown time-invariant drifts. For multiple measurements during a short period, all sensors are assumed to suffer a time-invariant drift. As shown in Fig. 2, unlike communication channels [23], electric signal output, e.g., current (I_{out}), by electronic devices causes a time-invariant drift. During m time-instants, (1) is naturally extended as

$$\hat{x}_i^{(k)} + \epsilon_i \approx \sum_{j=1,j\neq i}^{n} \hat{a}_{i,j}\left(\hat{x}_j^{(k)} + \epsilon_j\right) + \hat{a}_{i,0}, \quad k = 1, 2, \ldots, m \tag{2}$$

where $\hat{x}_i^{(k)}$ is the measurement value sensed by ith sensor at kth time-instant. In particular, in order to obtain enough information, we assume $m_0, m > n$. For ith sensor, ϵ_i is a time-invariant drift calibration, which is independent of time-instant k. $\hat{a}_{i,j}$ is the model coefficient when all sensors have unknown time-invariant drifts. We vectorize these variables as $\hat{\boldsymbol{x}} \triangleq [\hat{x}_1^{(1)}, \hat{x}_2^{(1)}, \ldots, \ldots, \hat{x}_n^{(m)}]^\top$, $\hat{\boldsymbol{a}}_i \triangleq [\hat{a}_{i,0}, \ldots, \hat{a}_{i,i-1}, \hat{a}_{i,i+1}, \ldots, \hat{a}_{i,n}]^\top \in \mathbb{R}^n$, $\hat{\boldsymbol{a}} \triangleq [\hat{\boldsymbol{a}}_1^\top, \hat{\boldsymbol{a}}_2^\top, \ldots, \hat{\boldsymbol{a}}_n^\top]^\top \in \mathbb{R}^{n^2}$ and $\boldsymbol{\epsilon} \triangleq [\epsilon_1, \epsilon_2, \ldots, \epsilon_n]^\top \in \mathbb{R}^n$.

Note that (2) is the essential in our proposed sensor spatial correlation model. Furthermore, the model error in (2) is assumed to follow identical independent Gaussian distribution with zero-mean and unknown precision (inverse variance) δ_0. Therefore, the likelihood function $\mathcal{P}(\hat{\boldsymbol{x}}|\hat{\boldsymbol{a}}, \boldsymbol{\epsilon})$ is defined as follows:

$$\mathcal{P}\left(\hat{\boldsymbol{x}}|\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}\right) \propto \exp\left(-\frac{\delta_0}{2} \sum_{i=1}^{n}\sum_{k=1}^{m}\Big[\hat{x}_i^{(k)} + \epsilon_i - \sum_{j=1,j\neq i}^{n} \hat{a}_{i,j}\left(\hat{x}_j^{(k)} + \epsilon_j\right) - \hat{a}_{i,0}\Big]^2\right). \tag{3}$$

However, the likelihood function $\mathcal{P}(\hat{\boldsymbol{x}}|\hat{\boldsymbol{a}}, \boldsymbol{\epsilon})$ cannot be directly used to calibrate drifts using maximum-likelihood-estimation (MLE) since it has not enough information. Therefore, we need give two priors in development.

For all sensors, drifts are assumed to follow identical independent Gaussian distribution with zero-mean and unknown precision $\delta_{\boldsymbol{\epsilon}}$ as follows:

$$\mathcal{P}(\boldsymbol{\epsilon}) \propto \exp\left(-\frac{\delta_{\boldsymbol{\epsilon}}}{2}\sum_{i=1}^{n}\epsilon_i^2\right). \tag{4}$$

In addition, we assume that the model coefficient $\hat{a}_{i,j}$ follows identical independent Gaussian distribution. Intuitively, $\hat{a}_{i,j}$ has high dependency on $a_{i,j}$ in statistics. Furthermore, the probability density function of $\hat{a}_{i,j}$ is assumed to take a maximum value at $a_{i,j}$. Therefore, the prior mean of $\hat{a}_{i,j}$ is $a_{i,j}$. In addition, in order that each model coefficient $\hat{a}_{i,j}$ is provided with a relatively equal probability to deviate from the corresponding drift-free model coefficient $a_{i,j}$, the precision of model coefficient $\hat{a}_{i,j}$ is defined to be $\lambda a_{i,j}^{-2}$, where λ is a nonnegative hyper-parameter to control the precision of $\hat{a}_{i,j}$. Therefore, each model coefficient $\hat{a}_{i,j}$ follows identical independent Gaussian distribution with $a_{i,j}$ mean and $\lambda a_{i,j}^{-2}$ precision [32]–[34]. For all model coefficients, we have

$$\mathcal{P}\left(\hat{\boldsymbol{a}}\right) \propto \exp\left(-\sum_{i=1}^{n}\sum_{j=0,j\neq i}^{n}\frac{\lambda}{2a_{i,j}^2}\left(\hat{a}_{i,j} - a_{i,j}\right)^2\right). \tag{5}$$

This prior manner is named Bayesian model fusion, which was developed to combine the early-stage information and the late-stage information using Bayesian inference in Computer-Aided Design applications [32]–[34].

In order to calibrate drifts for all sensors, the posterior $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}})$ needs to be maximized in MAP estimation manner. According to Bayes' rule, the posterior $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}})$ can be expressed by two priors and the likelihood function as follows:

$$\mathcal{P}\left(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}\right) \propto \mathcal{P}\left(\hat{\boldsymbol{x}}|\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}\right) \cdot \mathcal{P}\left(\hat{\boldsymbol{a}}\right) \cdot \mathcal{P}(\boldsymbol{\epsilon}). \tag{6}$$

Taking the logarithm, MAP can be transferred to be the equivalent formulation as follows:

$$\min_{\hat{\boldsymbol{a}},\boldsymbol{\epsilon}} \ \delta_0 \sum_{i=1}^{n}\sum_{k=1}^{m}\left[\hat{x}_i^{(k)}+\epsilon_i-\sum_{j=1,j\neq i}^{n}\hat{a}_{i,j}\left(\hat{x}_j^{(k)}+\epsilon_j\right)-\hat{a}_{i,0}\right]^2 + \lambda\sum_{i=1}^{n}\sum_{j=0,j\neq i}^{n}\frac{1}{a_{i,j}^2}\left(\hat{a}_{i,j}-a_{i,j}\right)^2+\delta_{\boldsymbol{\epsilon}}\sum_{i=1}^{n}\epsilon_i^2. \tag{7}$$

To calibrate these unknown drifts in all sensors, Formulation (7) will be optimized efficiently. Next, we will propose an efficient alternating-based method to optimize Formulation (7).

IV. Alternating-Based Optimization

There are two challenges for Formulation (7): how to handle Formulation (7) and how to induce hyper-parameters λ, δ_0, and $\delta_{\boldsymbol{\epsilon}}$. Formulation (7) is a nonconvex problem, thus it is difficult to obtain an optimal solution. In this section, we propose a fast and efficient alternating-based optimization methodology to optimize Formulation (7) by alternatively updating in each iteration.

According to the alternating-based methodology, at each iteration, the values of $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$ are updated by optimizing Formulation (7) with respect to (w.r.t.) $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$. Furthermore, note that with fixed drift calibration variable $\boldsymbol{\epsilon}$, Formulation (7) w.r.t. $\hat{\boldsymbol{a}}$ is regarded as a convex unconstrained quadratic programming (QP) problem, which can be solved by Gaussian elimination [35]. However, the computational complexity of Gaussian elimination is $\mathcal{O}(n^6)$ (n is the sensor number) if all model coefficients $\hat{\boldsymbol{a}}$ are calculated in one subformulation [35]. Consider that Formulation (7) w.r.t. $\hat{\boldsymbol{a}}$ can be decomposed into n independent subformulations w.r.t. $\hat{\boldsymbol{a}}_i$. In order to reduce computational complexity, instead of calculating all model coefficients $\hat{\boldsymbol{a}}$ in one subformulation, $\hat{\boldsymbol{a}}_i$ will be calculated in ith subformulation ($i = 1, 2, \ldots, n$). As a result, the computational complexity of Gaussian elimination is $\mathcal{O}(n^4)$ in total for n subformulations. Formulation (7) w.r.t. $\hat{\boldsymbol{a}}$ is decomposed into n independent subformulations w.r.t. $\hat{\boldsymbol{a}}_i$ as follows:

$$\min_{\hat{\boldsymbol{a}}_i} \ \delta_0 \sum_{k=1}^{m}\left[\hat{x}_i^{(k)}+\epsilon_i-\sum_{j=1,j\neq i}^{n}\hat{a}_{i,j}\left(\hat{x}_j^{(k)}+\epsilon_j\right)-\hat{a}_{i,0}\right]^2 + \lambda\sum_{j=0,j\neq i}^{n}\frac{1}{a_{i,j}^2}\left(\hat{a}_{i,j}-a_{i,j}\right)^2 \tag{8}$$

with the first-order optimality condition

$$\delta_0\sum_{k=1}^{m}\left(\hat{x}_t^{(k)}+\epsilon_t\right)\left[\sum_{j=1}^{n}\hat{a}_{i,j}\left(\hat{x}_j^{(k)}+\epsilon_j\right)+\hat{a}_{i,0}\right] + \lambda\frac{\left(\hat{a}_{i,t}-a_{i,t}\right)}{a_{i,t}^2}=0 \tag{9}$$

where $t = 0, 1, \ldots, i-1, i+1, \ldots, n$. In particular, we define $\hat{a}_{i,i} \triangleq -1$ and $\hat{x}_0^{(k)}+\epsilon_0 \triangleq 1$. The system of linear equations (9) can be addressed by Gaussian elimination [35].

Algorithm 1 Alternating-Based Method

Require: Sensor measurements $\hat{\boldsymbol{x}}$, prior $\boldsymbol{a}$ and hyper-parameters λ, δ_0, $\delta_{\boldsymbol{\epsilon}}$.

```
Initialize â ← a and ε ← 0;
repeat
    for i ← 1 to n do
        Fix ε, solve the system of linear equations (9) using Gaussian elimination to update â_i;
    end for
    Fix â, solve the system of linear equations (11) using Gaussian elimination to update ε;
until Convergence
return â and ε.
```

In the same manner, with fixed model coefficients $\hat{\boldsymbol{a}}$, Formulation (7) w.r.t. the drift calibration $\boldsymbol{\epsilon}$ can also be regarded to be a convex unconstrained QP problem as follows:

$$\min_{\boldsymbol{\epsilon}} \ \delta_0 \sum_{i=1}^{n}\sum_{k=1}^{m}\left[\hat{x}_i^{(k)}+\epsilon_i-\sum_{j=1,j\neq i}^{n}\hat{a}_{i,j}\left(\hat{x}_j^{(k)}+\epsilon_j\right)-\hat{a}_{i,0}\right]^2 + \delta_{\boldsymbol{\epsilon}}\sum_{i=1}^{n}\epsilon_i^2 \tag{10}$$

with the corresponding first-order optimality condition

$$\delta_0\sum_{i=1}^{n}\sum_{k=1}^{m}\left[\hat{a}_{i,t}\left(\sum_{j=1}^{n}\hat{a}_{i,j}\left(\hat{x}_j^{(k)}+\epsilon_j\right)+\hat{a}_{i,0}\right)\right]+\delta_{\boldsymbol{\epsilon}}\epsilon_t=0 \tag{11}$$

where $t = 1, 2, \ldots, n$.

A local optimum can be obtained by the proposed alternating-based method while the convergence speed and solution quality depend on the initialization of variables. In our proposed framework, two priors are given for model coefficients $\hat{\boldsymbol{a}}$ and drift calibration $\boldsymbol{\epsilon}$. Therefore, in order to achieve a better convergence speed and solution quality, the prior means $\boldsymbol{a}$ and $\mathbf{0}$ are used to initialize variables $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$. We continue to update $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$ until convergence. The convergence condition is that the relative difference of drift calibration $\boldsymbol{\epsilon}$ between current and previous iterations is less than a threshold. In summary, our proposed alternating-based method is shown in Algorithm 1. In fact, if the quadratic objective functions (7) w.r.t. $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$ are strictly convex with lower-bounded Hessians, the proposed alternating-based method can achieve convergency of Formulation (7) [36], [37].

V. Estimation of Hyper-Parameters

It is important to induce the aforementioned three hyper-parameters so that drifts can be accurately calibrated and meanwhile the over-fitting can be avoided [38]. In this section, cross-validation, Gibbs EM, and VB-EM are utilized to induce hyper-parameters, respectively.

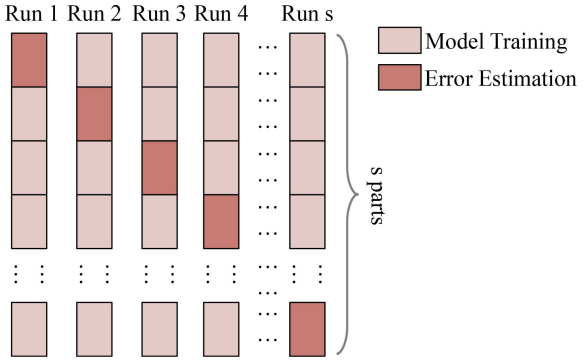

Fig. 3. Unsupervised cross-validation.

A. Unsupervised Cross-Validation

Cross-validation is a simple method to select hyper-parameters [39]. Although there are three hyper-parameters λ, δ_0, and $\delta_{\boldsymbol{\epsilon}}$ in Formulation (7), instead of determined by cross-validation individually, we design two ratios λ/δ_0 and $\delta_{\boldsymbol{\epsilon}}/\delta_0$ which demand certain values. As shown in Fig. 3, we partition temperature measurements during m time-instants into s nonoverlapping parts. Given each combination of ratios candidates λ/δ_0 and $\delta_{\boldsymbol{\epsilon}}/\delta_0$, in each run, one of the s parts is exploited to estimate the model error and the rest $s-1$ parts are used to calculate model coefficients and drift calibration. In addition, different groups will be selected for error estimation in different runs. In the same manner, each run gives a model error e_r $(r = 1, 2, \ldots, s)$ estimated from a part of temperature measurements. The model error is defined as follows:

$$e_r \triangleq \sum_{k=\frac{(r-1)m}{s}+1}^{\frac{rm}{s}} \left[\hat{x}_i^{(k)} + \epsilon_i - \sum_{j=1, j\neq i}^{n} \hat{a}_{i,j}\left(\hat{x}_j^{(k)} + \epsilon_j\right) - \hat{a}_{i,0} \right]^2. \tag{12}$$

The final model error is computed as the average $\bar{e} = \sum_{r=1}^{s} e_r/s$. Then two ratios λ/δ_0 and $\delta_{\boldsymbol{\epsilon}}/\delta_0$ corresponding to the minimum average model error are chosen.

The pseudocode of the unsupervised cross-validation is shown in Algorithm 2. We input sensor measurements $\hat{\boldsymbol{x}}$, the drift-free model coefficients $\boldsymbol{a}$, the number of folds for cross-validation s and several hyper-parameters candidates. Then the sensor measurements are split into s nonoverlapping parts as illustrated in Fig. 3. Formulation (7) on model training set is optimized by Algorithm 1 and the model error defined in (12) on the validation set is calculated for each run and each candidate of hyper-parameters. After the grid search on all hyper-parameters and sensor measurements $\hat{\boldsymbol{x}}$ is finished, the model error is averaged on s runs. At last, the hyper-parameters with the least average model error are chosen to output. The unsupervised cross-validation flow is summarized in Fig. 4.

Note that unlike conventional cross-validation [1], [2], [25], [32]–[34], not any golden values are used in metrics to choose hyper-parameters in model fitting stage. Therefore, in our proposed framework, cross-validation is adopted in an unsupervised-learning-like fashion.

Cross-validation is time-consuming since Algorithm 1 has to be performed for multiple times. Thus, we propose two fast and efficient EM algorithms to determine hyper-parameters in the statistical model.

Algorithm 2 Unsupervised Cross-Validation

Require: Sensor measurements $\hat{\boldsymbol{x}}$, prior $\boldsymbol{a}$, number of folds for cross-validation s, hyper-parameters candidates $\{(\lambda/\delta_0)_1, (\lambda/\delta_0)_2, \cdots, (\lambda/\delta_0)_{d_{\lambda/\delta_0}}\}$ and $\{(\delta_{\boldsymbol{\epsilon}}/\delta_0)_1, (\delta_{\boldsymbol{\epsilon}}/\delta_0)_2 \cdots, (\delta_{\boldsymbol{\epsilon}}/\delta_0)_{d_{\delta_{\boldsymbol{\epsilon}}/\delta_0}}\}$.

1: **for** $r \leftarrow 1$ to s **do**
2: **for** $i \leftarrow 1$ to d_{λ/δ_0} **do**
3: **for** $j \leftarrow 1$ to $d_{\delta_{\boldsymbol{\epsilon}}/\delta_0}$ **do**
4: Obtain the model coefficients $\hat{\boldsymbol{a}}$ and calibration $\boldsymbol{\epsilon}$ by Algorithm 1 from sensor measurements with rth part removed.
5: Compute the model error on rth part of sensor measurements by Equation (12).
6: **end for**
7: **end for**
8: **end for**
9: Average the computed modeling error for each pair of hyper-parameters candidates $(\lambda/\delta_0)_i$ and $(\delta_{\boldsymbol{\epsilon}}/\delta_0)_j$, *i.e.*, $\bar{e} \leftarrow \sum_{r=1}^{s} e_k/s$.
10: Select $(\lambda/\delta_0)_i$ and $(\delta_{\boldsymbol{\epsilon}}/\delta_0)_j$ with the smallest modeling error, *i.e.*, $(\lambda/\delta_0)_{opt}, (\delta_{\boldsymbol{\epsilon}}/\delta_0)_{opt} \leftarrow \operatorname{argmin} \bar{e}$.
11: **return** The optimal hyper-parameters $(\lambda/\delta_0)_{opt}$ and $(\delta_{\boldsymbol{\epsilon}}/\delta_0)_{opt}$.

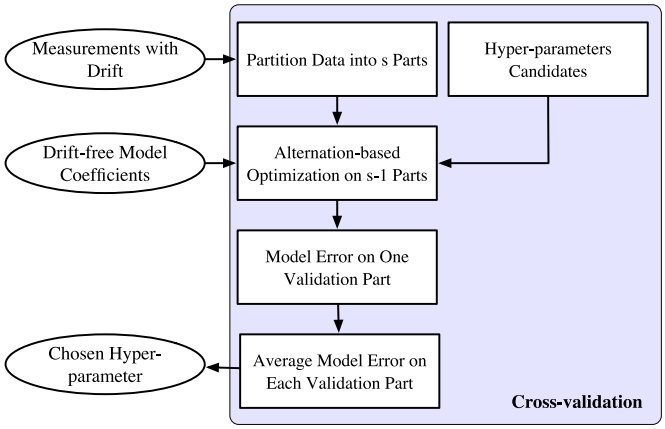

Fig. 4. Unsupervised cross-validation flow.

B. Gibbs Expectation Maximization

In this section, MLE is used to determine individual hyper-parameters δ_0, λ, and $\delta_{\boldsymbol{\epsilon}}$. MLE of hyper-parameters is formulated as follows:

$$\max_{\delta_{\boldsymbol{\epsilon}}, \delta_0, \lambda} \quad \mathcal{P}\left(\hat{\boldsymbol{x}}; \delta_0, \lambda, \delta_{\boldsymbol{\epsilon}}\right). \tag{13}$$

However, in the likelihood function $\mathcal{P}(\hat{\boldsymbol{x}}; \delta_0, \lambda, \delta_{\boldsymbol{\epsilon}})$, the integral is intractable. EM algorithm is leveraged to efficiently find a solution to Formulation (13), [40]. According to the EM algorithm, Formulation (13) can be taken the logarithm and transferred to be its auxiliary lower bound function [41]. Then the auxiliary lower-bound function is optimized by **E-step**

and **M-step** iteratively after the term independent of hyper-parameters is omitted. The detailed derivation can be found in [42]. For convenience, all hyper-parameters are collected as a set $\Omega \triangleq \{\delta_0, \lambda, \delta_{\boldsymbol{\epsilon}}\}$.

1) Expectation Step With Gibbs Sampling: Since $\mathcal{P}(\hat{\boldsymbol{x}}; \delta_0, \lambda, \delta_{\boldsymbol{\epsilon}}) = \mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda) + \mathcal{KL}(\mathcal{Q}||\mathcal{P})$ and $\mathcal{KL}(\mathcal{Q}||\mathcal{P}) \geq 0$, $\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda)$ is a lower-bound function defined as follows:

$$\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda) = \mathbb{E}_{\mathcal{Q}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon})} \ln \frac{\mathcal{P}\left(\hat{\boldsymbol{X}}, \hat{\boldsymbol{a}}, \boldsymbol{\epsilon}; \delta_\epsilon, \delta_0, \lambda\right)}{\mathcal{Q}\left(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}\right)} \tag{14}$$

where $\mathbb{E}$ indicates the expectation operator. $\mathcal{Q}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon})$ is an arbitrary joint distribution for $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$. Instead of maximizing the marginal likelihood directly, the EM maximizes the lower bound function. Let's assume that we can find $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$ analytically. Then, we can simply substitute $\mathcal{Q}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}) = \mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$. The lower-bound function can be represented as follows:

$$\begin{aligned}\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda) = {} & \mathbb{E}_{\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})} \ln \mathcal{P}\left(\hat{\boldsymbol{X}}, \hat{\boldsymbol{a}}, \boldsymbol{\epsilon}; \delta_\epsilon, \delta_0, \lambda\right) \\ & - \mathbb{E}_{\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})} \ln \mathcal{P}\left(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}}\right). \end{aligned} \tag{15}$$

The second term $\mathbb{E}_{\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})} \ln \mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$ is a constant w.r.t. δ_ϵ and δ_0 and λ, and we do not take the term into account when maximizing the lower-bound function. Therefore, we define a quantity as follows:

$$Q\left(\Omega|\Omega^{\text{old}}\right) = \mathbb{E}_{\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})} \ln \mathcal{P}\left(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}, \boldsymbol{\epsilon}; \Omega\right) \tag{16}$$

where Ω^{old} denotes estimated hyper-parameters in the previous iteration.

However, the posterior $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\boldsymbol{x}; \Omega^{\text{old}})$ is intractable since it is hard to calculate the integral in the likelihood function $\mathcal{P}(\hat{\boldsymbol{x}}; \delta_0, \lambda, \delta_{\boldsymbol{\epsilon}})$. There are two main methods to approximate the posterior $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\boldsymbol{x}; \Omega)$: variational inference and Markov Chain Monte Carlo (MCMC) [42]. Compared with variational inference, MCMC has the advantage of being nonparametric and asymptotically exact [43]. Therefore, Monte Carlo method is utilized to approximate the quantity defined in (17) as follows:

$$Q\left(\Omega|\Omega^{\text{old}}\right) \approx \frac{1}{L} \sum_{l=1}^{L} \ln \mathcal{P}\left(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}^{(l)}, \boldsymbol{\epsilon}^{(l)}; \Omega\right) \tag{17}$$

where samples $\hat{\boldsymbol{a}}^{(l)}$ and $\boldsymbol{\epsilon}^{(l)}$ are sampled from the distribution $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$. L is the total number of samples. In MCMC, there are two main algorithms to obtain samples from the desired distribution $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$: Metropolis–Hastings algorithm and Gibbs sampling. Since the rejection rate will be high in complex problems, the Metropolis–Hastings algorithm has a very slow convergence rate [42]. Therefore, Gibbs sampling is harnessed to obtain samples $\hat{\boldsymbol{a}}^{(l)}$ and $\boldsymbol{\epsilon}^{(l)}$.

Gibbs sampling has the behavior that one or batch variables are cyclically and repeatedly updated in some particular order at random from conditional distribution. Sampling order is arranged to be $\hat{a}_{1,0}^{(l)}, \ldots, \hat{a}_{1,n}^{(l)}, \hat{a}_{2,0}^{(l)}, \ldots, \hat{a}_{n,n-1}^{(l)}, \epsilon_1^{(l)}, \ldots, \epsilon_n^{(l)}$. In Gibbs sampling, one of the key points is the derivation of the conditional distribution for each variable. Note that according to Formulation (7), the logarithm conditional distribution w.r.t. individual variable is quadratic. Therefore, the conditional distribution of each variable is Gaussian distribution as follows:

$$\begin{aligned}\hat{a}_{p,q} &\sim \mathcal{P}\left(\hat{a}_{p,q}|\boldsymbol{\epsilon}, \hat{\boldsymbol{a}}_{/\hat{a}_{p,q}}, \hat{\boldsymbol{x}}; \delta_\epsilon, \delta, \lambda\right) = \mathcal{N}\left(\hat{a}_{p,q}; \mu_{\hat{a}_{p,q}}, \sigma_{\hat{a}_{p,q}}^{-1}\right) \\ \epsilon_t &\sim \mathcal{P}\left(\epsilon_t|\boldsymbol{\epsilon}_{/\epsilon_t}, \hat{\boldsymbol{a}}, \hat{\boldsymbol{x}}; \delta_\epsilon, \delta, \lambda\right) = \mathcal{N}\left(\epsilon_t; \mu_{\epsilon_t}, \sigma_{\epsilon_t}^{-1}\right) \end{aligned} \tag{18}$$

in agreement with (4) and (5). μ is mean and σ is precision. $\hat{\boldsymbol{a}}_{/\hat{a}_{p,q}}$ and $\boldsymbol{\epsilon}_{/\epsilon_t}$ denote $\hat{\boldsymbol{a}}$ but with $\hat{a}_{p,q}$ omitted and $\boldsymbol{\epsilon}$ but with ϵ_t omitted, respectively. In particular, we define $\hat{a}_{i,i}^{(s)} \triangleq -1$, $\hat{x}_0^{(k)} + \epsilon_0^{(s)} \triangleq 1$ and $p \neq q$. The mean and precision of each variable are given as follows:

$$\sigma_{\epsilon_t} = m\delta_0 \sum_{i=1}^{n} \hat{a}_{i,t}^{(s)2} + \delta_{\boldsymbol{\epsilon}}, \tag{19}$$

$$\sigma_{\hat{a}_{p,q}} = \delta_0 \sum_{k=1}^{m} \left(\hat{x}_q^{(k)} + \epsilon_q^{(s)}\right)^2 + \frac{\lambda}{a_{p,q}^2}, \tag{20}$$

$$\mu_{\epsilon_t} = \frac{\delta_0}{\sigma_{\epsilon_t}} \sum_{k=1}^{m} \sum_{i=1}^{n} \hat{a}_{i,t}^{(s)} \left[\sum_{j=0}^{n} \hat{a}_{i,j}^{(s)} \left(\hat{x}_j^{(k)} + \epsilon_j^{(s)}\right) - \hat{a}_{i,t}^{(s)} \epsilon_t^{(s)}\right] \tag{21}$$

and

$$\begin{aligned}\mu_{\hat{a}_{p,q}} = {} & \frac{\delta_0}{\sigma_{\hat{a}_{p,q}}} \sum_{k=1}^{m} \left(\hat{x}_q^{(k)} + \epsilon_q^{(s)}\right) \left[\sum_{j=0}^{n} \hat{a}_{p,j}^{(s)} \left(\hat{x}_j^{(k)} + \epsilon_j^{(s)}\right)\right. \\ & \left. - \hat{a}_{p,q}^{(s)} \left(\hat{x}_q^{(k)} + \epsilon_q^{(s)}\right)\right] + \frac{\lambda}{\sigma_{\hat{a}_{p,q}} a_{p,q}}. \end{aligned} \tag{22}$$

Before Gibbs sampling, in order to converge to the desired posterior, the warm-start has to be performed if there is no reasonable initialization for samples. Furthermore, it is very hard to judge whether the warm-start is enough [42]. In order to waive the warm-start, a reasonable initialization for samples is adopted in Gibbs sampling. Note that Gibbs sampling is used to obtain samples from the desired posterior $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$ (6). As we discussed in Section II, Formulation (7) is equivalent to MAP estimation of $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$. Thus, given hyper-parameters Ω^{old} and measurement values $\hat{\boldsymbol{x}}$, Gibbs sampling can be initialized by handling Formulation (7) to obtain initial samples $\hat{\boldsymbol{a}}^{(0)}$ and $\boldsymbol{\epsilon}^{(0)}$ which satisfy the distribution $\mathcal{P}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}|\hat{\boldsymbol{x}}; \Omega^{\text{old}})$. As a result, the warm-start can be totally waived.

2) Maximization Step: After L samples are obtained by Gibbs sampling, in **M-step**, we will maximize the approximated quantity as follows:

$$\max_{\Omega} \quad \frac{1}{L} \sum_{l=1}^{L} \ln \mathcal{P}\left(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}^{(l)}, \boldsymbol{\epsilon}^{(l)}; \Omega\right). \tag{23}$$

Algorithm 3 Gibbs EM

Require: Sensor measurements $\hat{\boldsymbol{x}}$, prior $\boldsymbol{a}$;
1: Initialize hyper-parameters Ω;
2: **repeat**
3: Initialize samples $\Psi^{(0)}$ by Algorithm 1;
4: **for** $l \leftarrow 1$ to L **do**
5: **for** $i \leftarrow 1$ to $n^2 + n$ **do**
6: Sample $\psi_i^{(l)}$ from the desired conditional distribution $\mathcal{N}(\psi_i; \mu_{\psi_i}, \sigma_{\psi_i})$ (18) with $\psi_1^{(l)}, \cdots, \psi_{i-1}^{(l)}$, $\psi_{i+1}^{(l-1)}, \cdots, \psi_{n^2+n}^{(l-1)}$;
7: **end for**
8: **end for**
9: Update hyper-parameters Ω by Equations (24) to (26);
10: **until** Convergence
11: **return** hyper-parameters Ω.

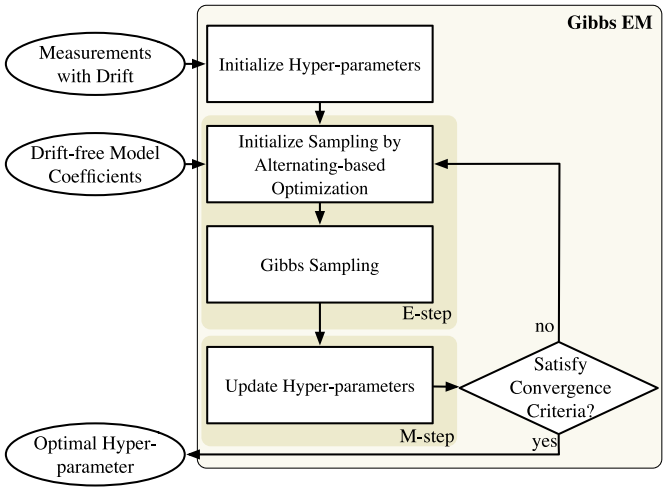

Fig. 5. Gibbs EM flow.

With the first-order optimality condition, namely, $dQ/d\Omega = 0$, hyper-parameters λ, δ_0, $\delta_{\boldsymbol{\epsilon}}$ can be updated as follows:

$$\lambda = \frac{n^2 L}{\sum_{i=1}^{n} \sum_{j=0, j\neq i}^{n} \sum_{l=1}^{L} \frac{\left(\hat{a}_{i,j}^{(l)} - a_{i,j}\right)^2}{a_{i,j}^2}} \tag{24}$$

$$\delta_0 = \frac{Lmn}{\sum_{l=1}^{L} \sum_{i=1}^{n} \sum_{k=1}^{m} \left[\sum_{j=1}^{n} \hat{a}_{i,j}^{(l)} \left(\hat{x}_j^{(k)} + \epsilon_j^{(l)}\right) + \hat{a}_{i,0}^{(l)}\right]^2} \tag{25}$$

$$\delta_{\boldsymbol{\epsilon}} = \frac{nL}{\sum_{l=1}^{L} \sum_{i=1}^{n} \epsilon_i^{(l)2}}. \tag{26}$$

Here, $\hat{a}_{i,i}^{(l)} \triangleq -1$ and $\hat{x}_0^{(k)} + \epsilon_0^{(l)} \triangleq 1$. We continue to alternate between E-step and M-step until convergence. The convergence condition is that the relative difference of three hyper-parameters between current and previous iterations is less than a threshold. Then hyper-parameters λ, δ, $\delta_{\boldsymbol{\epsilon}}$ can be determined.

For convenience, all variables are collected as a set $\Psi \triangleq \{\psi_1, \psi_2, \ldots, \psi_{n^2+n}\} = \{\hat{a}_{1,0}, \ldots, \hat{a}_{1,n}, \ldots, \hat{a}_{n,n-1}, \epsilon_1, \ldots, \epsilon_n\}$. The pseudocode of the Gibbs EM is concluded in Algorithm 3. Given $\hat{\boldsymbol{x}}$, $\boldsymbol{a}$, and initialized hyper-parameters as inputs, the Gibbs EM is performed iteratively. In E-step, samples are initialized by Algorithm 1 to waive warm start. Then in Gibbs sampling, each samples are cyclically and repeatedly obtained by the desired conditional distributions defined in (18). Once there are enough samples acquired by Gibbs sampling, hyper-parameters are updated by (24) to (26) in M-step. E-step and M-step are iteratively performed until the convergence criteria is satisfied. At last, we have the optimized hyper-parameters. For a better understanding, our proposed Gibbs EM is displayed in Fig. 5.

C. Variational Bayesian Expectation Maximization

In practice, MCMC sampling method is computationally demanding. As a result, it often limits its use to small-scale problems [42]. Besides, it can be difficult to judge whether the sampling method is generating independent samples from the desired distribution. In this section, we develop a deterministic approximation scheme which scales well to large applications.

We can even enforce full independence between the model parameters $\hat{\boldsymbol{a}}$ and the calibration $\boldsymbol{\epsilon}$ given measurements $\hat{\boldsymbol{x}}$. This assumption, known as the mean-field approximation [44], allows us to compute the update rules for $\hat{\boldsymbol{a}}$ and the calibration $\boldsymbol{\epsilon}$ in isolation.

According to the mean-field approximation, in (14), let $\mathcal{Q}(\hat{\boldsymbol{a}}, \boldsymbol{\epsilon}) = \mathcal{Q}(\hat{\boldsymbol{a}})\mathcal{Q}(\boldsymbol{\epsilon})$. Then, the lower-bound function $\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda)$ can be factorized into $\hat{\boldsymbol{a}}$ and $\boldsymbol{\epsilon}$

$$\begin{aligned}
&\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda) \\
&= -\mathcal{KL}\Big(\mathcal{Q}(\hat{\boldsymbol{a}}) || \tilde{\mathcal{P}}(\boldsymbol{\epsilon})\Big) - \mathbb{E}_{\mathcal{Q}(\boldsymbol{\epsilon})} \ln \mathcal{Q}(\boldsymbol{\epsilon}) + c \qquad (27)\\
&= -\mathcal{KL}\Big(\mathcal{Q}(\boldsymbol{\epsilon}) || \tilde{\mathcal{P}}(\hat{\boldsymbol{a}})\Big) - \mathbb{E}_{\mathcal{Q}(\hat{\boldsymbol{a}})} \ln \mathcal{Q}(\hat{\boldsymbol{a}}) + c \qquad (28)
\end{aligned}$$

where c is a constant to adjust $\tilde{\mathcal{P}}(\hat{\boldsymbol{a}})$ or $\tilde{\mathcal{P}}(\boldsymbol{\epsilon})$ to become a proper probability density function as follows:

$$\tilde{\mathcal{P}}(\hat{\boldsymbol{a}}) \triangleq \frac{1}{Z} \exp\left[\mathbb{E}_{\mathcal{Q}(\boldsymbol{\epsilon})} \ln \mathcal{P}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}; \lambda, \delta_0, \delta_{\boldsymbol{\epsilon}})\right] \tag{29}$$

$$\tilde{\mathcal{P}}(\boldsymbol{\epsilon}) \triangleq \frac{1}{Z} \exp\left[\mathbb{E}_{\mathcal{Q}(\hat{\boldsymbol{a}})} \ln \mathcal{P}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}; \lambda, \delta_0, \delta_{\boldsymbol{\epsilon}})\right] \tag{30}$$

where Z is a normalized constant. Since Kullback–Leibler divergence $\mathcal{KL}(\mathcal{Q}(\hat{\boldsymbol{a}}) || \tilde{\mathcal{P}}(\boldsymbol{\epsilon})) \geq 0$, the lower-bound function $\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda)$ is maximized w.r.t. $\mathcal{Q}(\hat{\boldsymbol{a}})$ when $\mathcal{KL}(\mathcal{Q}(\hat{\boldsymbol{a}}) || \tilde{\mathcal{P}}(\boldsymbol{\epsilon})) = 0$, which happens when $\mathcal{Q}(\hat{\boldsymbol{a}}) = \tilde{\mathcal{P}}(\boldsymbol{\epsilon})$. Similarity, because $\mathcal{KL}(\mathcal{Q}(\boldsymbol{\epsilon}) || \tilde{\mathcal{P}}(\hat{\boldsymbol{a}})) \geq 0$, the lower-bound function $\mathcal{L}(\mathcal{Q}; \delta_\epsilon, \delta_0, \lambda)$ is maximized w.r.t. $\mathcal{Q}(\boldsymbol{\epsilon})$ if $\mathcal{KL}(\mathcal{Q}(\boldsymbol{\epsilon}) || \tilde{\mathcal{P}}(\hat{\boldsymbol{a}})) = 0$, which happens when $\mathcal{Q}(\boldsymbol{\epsilon}) = \tilde{\mathcal{P}}(\hat{\boldsymbol{a}})$.

Therefore, in *variational E-step*, let

$$\ln \mathcal{Q}(\hat{\boldsymbol{a}}) = \mathbb{E}_{\mathcal{Q}(\boldsymbol{\epsilon})} \ln \mathcal{P}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}, \boldsymbol{\epsilon} | \lambda, \delta_0, \delta_{\boldsymbol{\epsilon}}) \tag{31}$$

$$\ln \mathcal{Q}(\boldsymbol{\epsilon}) = \mathbb{E}_{\mathcal{Q}(\hat{\boldsymbol{a}})} \ln \mathcal{P}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}, \boldsymbol{\epsilon} | \lambda, \delta_0, \delta_{\boldsymbol{\epsilon}}) \tag{32}$$

where according to Formulation (7), the logarithm conditional distribution w.r.t. individual variable is quadratic. Therefore, the variational distribution of each variable is Gaussian distribution as follows:

$$\mathcal{Q}(\hat{\boldsymbol{a}}) \sim \mathcal{N}(\hat{\boldsymbol{a}}; \mathbb{E}(\hat{\boldsymbol{a}}), \boldsymbol{\Sigma}_{\hat{\boldsymbol{a}}}) \tag{33}$$

$$\mathcal{Q}(\boldsymbol{\epsilon}) \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbb{E}(\boldsymbol{\epsilon}), \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}) \tag{34}$$

where $\mathbf{\Sigma}_{\hat{\boldsymbol{a}}}$ is the covariance matrix. $\hat{\boldsymbol{a}}_i$ is independent to each other, since Formulation (7) w.r.t. $\hat{\boldsymbol{a}}$ can be decomposed into n independent subFormulations w.r.t. $\hat{\boldsymbol{a}}_i$. Therefore, $\mathbf{\Sigma}_{\hat{\boldsymbol{a}}} \triangleq \text{diag}[\mathbf{\Sigma}_{\hat{\boldsymbol{a}}_1}, \mathbf{\Sigma}_{\hat{\boldsymbol{a}}_2}, \ldots, \mathbf{\Sigma}_{\hat{\boldsymbol{a}}_n}]$. $\mathbb{E}(\hat{\boldsymbol{a}}) \triangleq [\mathbb{E}(\hat{\boldsymbol{a}}_1^\top), \mathbb{E}(\hat{\boldsymbol{a}}_2)^\top, , \ldots, \mathbb{E}(\hat{\boldsymbol{a}}_n)^\top]^\top$. By combining coefficients of $\hat{\boldsymbol{a}}_i$ quadratic term in (32), we can obtain

$$\mathbf{\Sigma}_{\hat{\boldsymbol{a}}_i}^{-1} = \delta_0\Big(\hat{\boldsymbol{X}}_{/i}^\top \hat{\boldsymbol{X}}_{/i} + \mathbb{E}\big(\boldsymbol{E}_{/i}\big)^\top \hat{\boldsymbol{X}}_{/i} + \hat{\boldsymbol{X}}_{/i}^\top \mathbb{E}\big(\boldsymbol{E}_{/i}\big) + \mathbb{E}\Big(\boldsymbol{E}_{/i}^\top \boldsymbol{E}_{/i}\Big)\Big) + \lambda \boldsymbol{A}_i^\top \boldsymbol{A}_i \tag{35}$$

where $\boldsymbol{A}_i \triangleq \text{diag}[a_{i,0}, a_{i,1}, \ldots, a_{i,i-1}, a_{i,i+1}, \ldots, a_{i,n}]$

$$\hat{\boldsymbol{X}}_{/i} \triangleq \begin{bmatrix} 1 & \hat{x}_1^{(1)} & \cdots & \hat{x}_{i-1}^{(1)} & \hat{x}_{i+1}^{(1)} & \cdots & \hat{x}_n^{(1)} \\ 1 & \hat{x}_1^{(2)} & \cdots & \hat{x}_{i-1}^{(2)} & \hat{x}_{i+1}^{(2)} & \cdots & \hat{x}_n^{(2)} \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \hat{x}_1^{(m)} & \cdots & \hat{x}_{i-1}^{(m)} & \hat{x}_{i+1}^{(m)} & \cdots & \hat{x}_n^{(m)} \end{bmatrix} \tag{36}$$

$$\boldsymbol{E}_{/i} \triangleq \begin{bmatrix} 0 & \epsilon_1 & \cdots & \epsilon_{i-1} & \epsilon_{i+1} & \cdots & \epsilon_n \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\ 0 & \epsilon_1 & \cdots & \epsilon_{i-1} & \epsilon_{i+1} & \cdots & \epsilon_n \end{bmatrix}. \tag{37}$$

Hence, according to Equations (36) and (37), $\mathbb{E}(\boldsymbol{E}_{/i})$ depends on the mean vector $\mathbb{E}(\boldsymbol{\epsilon})$ and $\mathbb{E}(\boldsymbol{E}_{/i}^\top \boldsymbol{E}_{/i})$ depends on the autocorrelation matrix $\mathbb{E}(\boldsymbol{\epsilon}^\top \boldsymbol{\epsilon})$ and the mean vector $\mathbb{E}(\boldsymbol{\epsilon})$.

Combining coefficients of $\hat{\boldsymbol{a}}_i$ linear term in (32), we can obtain the mean vector as follows:

$$\mathbb{E}\big(\hat{\boldsymbol{a}}_i\big)^\top = \delta_0\Big(\hat{\boldsymbol{x}}_i^\top + \mathbb{E}(\boldsymbol{\epsilon}_i)^\top\Big)\Big(\hat{\boldsymbol{X}}_{/i} + \mathbb{E}\big(\boldsymbol{E}_{/i}\big)\Big)\mathbf{\Sigma}_{\hat{\boldsymbol{a}}_i} \tag{38}$$

where $\hat{\boldsymbol{x}}_i \triangleq [\hat{x}_i^{(1)}, \hat{x}_i^{(2)}, \ldots, \hat{x}_i^{(m)}]^\top$ and $\mathbb{E}(\boldsymbol{\epsilon}_i) \triangleq [\mathbb{E}(\epsilon_i), \mathbb{E}(\epsilon_i), \ldots, \mathbb{E}(\epsilon_i)]^\top \in \mathbb{R}^m$. Once the covariance matrix $\mathbf{\Sigma}_{\hat{\boldsymbol{a}}_i}^{-1}$ and mean vector $\mathbb{E}(\hat{\boldsymbol{a}}_i)$ are obtained, we can obtain the autocorrelation matrix $\mathbb{E}(\hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^\top) = \mathbf{\Sigma}_{\hat{\mathbf{a}}_i} + \mathbb{E}(\hat{\mathbf{a}}_i)\mathbb{E}(\hat{\mathbf{a}}_i)^\top$.

Likewise, by combining coefficients of $\boldsymbol{\epsilon}$ quadratic term in (31), we can get the covariance matrix $\mathbf{\Sigma}_{\boldsymbol{\epsilon}}$ and the mean vector $\mathbb{E}(\boldsymbol{\epsilon})$ in (39) and (40), as shown at the bottom of the page, which depend on the mean vector $\mathbb{E}(\hat{\boldsymbol{a}}_i)$ and the autocorrelation matrix $\mathbb{E}(\hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^\top)$. Then the autocorrelation matrix $\mathbb{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top)$ can be obtained by $\mathbb{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top) = \mathbf{\Sigma}_{\boldsymbol{\epsilon}} + \mathbb{E}(\boldsymbol{\epsilon})\mathbb{E}(\boldsymbol{\epsilon})^\top$.

Since model coefficients $\mathbf{a}$ and calibration $\boldsymbol{\epsilon}$ variational distributions are dependent on each other, by iteratively updating them, the variational distributions $\mathcal{Q}(\hat{\mathbf{a}})$ and $\mathcal{Q}(\boldsymbol{\epsilon})$ can be obtained to maximize the lower-bound function defined in (28).

Once the optimal variational distributions $\mathcal{Q}(\hat{\mathbf{a}})$ and $\mathcal{Q}(\boldsymbol{\epsilon})$ can be obtained, unlike Formulation (23), in **variational M-step**, the objective function is shown as follows:

$$\max_{\Omega} \quad \mathbb{E}_{\mathcal{Q}(\hat{\boldsymbol{a}})\mathcal{Q}(\boldsymbol{\epsilon})} \ln \mathcal{P}\big(\hat{\boldsymbol{x}}, \hat{\boldsymbol{a}}, \boldsymbol{\epsilon}; \Omega\big). \tag{41}$$

With the first-order optimality condition, that is $dQ/d\Omega = 0$, hyper-parameters λ, δ_0, $\delta_{\boldsymbol{\epsilon}}$ can be updated as follows:

$$\lambda = \frac{n^2}{\sum_{i=1}^n \sum_{j=0, j\neq i}^n \frac{\mathbb{E}\left(\hat{a}_{i,j}^2\right) + a_{i,j}^2 - 2a_{i,j}\mathbb{E}(\hat{a}_{i,j})}{a_{i,j}^2}} \tag{42}$$

$$\delta_0 = mn / \sum_{i=1}^n \sum_{k=1}^m \sum_{p=0}^n \sum_{q=0}^n \mathbb{E}\big(\hat{a}_{i,p}\hat{a}_{i,q}\big)\Big[\hat{x}_p^{(k)}\hat{x}_q^{(k)} + \hat{x}_p^{(k)}\mathbb{E}\big(\epsilon_q\big) + \hat{x}_q^{(k)}\mathbb{E}\big(\epsilon_p\big) + \mathbb{E}\big(\epsilon_p\epsilon_q\big)\Big] \tag{43}$$

$$\delta_{\boldsymbol{\epsilon}} = \frac{n}{\sum_{i=1}^n \mathbb{E}\big(\epsilon_i^2\big)}. \tag{44}$$

Here we can obtain the variational distributions' mean vector and autocorrelation matrix replace samples i.e., $\mathbb{E}(\epsilon_i) = \sum_{l=1}^L \epsilon_i^{(l)}/L$, $\mathbb{E}(\hat{a}_{i,j}) = \sum_{l=1}^L \hat{a}_{i,j}^{(l)}/L$, $\mathbb{E}(\epsilon_i\epsilon_j) = \sum_{l=1}^L \epsilon_i^{(l)}\epsilon_j^{(l)}/L$ and $\mathbb{E}(\hat{a}_{i,p}\hat{a}_{i,q}) = \sum_{l=1}^L \hat{a}_{i,p}^{(l)}\hat{a}_{i,q}^{(l)}/L$. Compared with Gibbs EM, the variational Bayesian EM is more stable as it is a deterministic method while Gibbs EM is stochastic method.

Like Gibbs EM, we continue to alternate between variational E-step and variational M-step until convergence. The convergence condition is that the relative differences of three hyper-parameters between current and previous iterations are less than thresholds. Then hyper-parameters λ, δ, $\delta_{\boldsymbol{\epsilon}}$ can be determined.

The pseudocode of variational Bayesian EM is listed in Algorithm 4. Given $\hat{\boldsymbol{x}}, \boldsymbol{a}$, and initialized hyper-parameters, variational Bayesian EM is performed iteratively. In variational E-step, to initialize calibration variational distribution, the calibration mean and the autocorrelation matrix are obtained by Algorithm 1. Then calibration and model coefficients variational distributions are iteratively updated by Equations (35) and (38)–(40) until the convergence criteria is satisfied. In variational M-step, hyper-parameters are updated according

$$\mathbf{\Sigma}_{\boldsymbol{\epsilon}}^{-1} = m\sum_{i=1}^n \begin{bmatrix} \mathbb{E}(\hat{a}_{i,1}\hat{a}_{i,1}) & \cdots & \mathbb{E}(\hat{a}_{i,1}\hat{a}_{i,i-1}) & -\mathbb{E}(\hat{a}_{i,1}) & \mathbb{E}(\hat{a}_{i,1}\hat{a}_{i,i+1}) & \cdots & \mathbb{E}(\hat{a}_{i,1}\hat{a}_{i,n}) \\ \mathbb{E}(\hat{a}_{i,2}\hat{a}_{i,1}) & \cdots & \mathbb{E}(\hat{a}_{i,2}\hat{a}_{i,i-1}) & -\mathbb{E}(\hat{a}_{i,2}) & \mathbb{E}(\hat{a}_{i,2}\hat{a}_{i,i+1}) & \cdots & \mathbb{E}(\hat{a}_{i,2}\hat{a}_{i,n}) \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots \\ \mathbb{E}(\hat{a}_{i,i-1}\hat{a}_{i,1}) & \cdots & \mathbb{E}(\hat{a}_{i,i-1}\hat{a}_{i,i-1}) & -\mathbb{E}(\hat{a}_{i,i-1}) & \mathbb{E}(\hat{a}_{i,i-1}\hat{a}_{i,i+1}) & \cdots & \mathbb{E}(\hat{a}_{i,i-1}\hat{a}_{i,n}) \\ -\mathbb{E}(\hat{a}_{i,1}) & \cdots & -\mathbb{E}(\hat{a}_{i,i-1}) & 1 & -\mathbb{E}(\hat{a}_{i,i+1}) & \cdots & -\mathbb{E}(\hat{a}_{i,n}) \\ \mathbb{E}(\hat{a}_{i,i+1}\hat{a}_{i,1}) & \cdots & \mathbb{E}(\hat{a}_{i,i+1}\hat{a}_{i,i-1}) & -\mathbb{E}(\hat{a}_{i,i+1}) & \mathbb{E}(\hat{a}_{i,i+1}\hat{a}_{i,i+1}) & \cdots & \mathbb{E}(\hat{a}_{i,i+1}\hat{a}_{i,n}) \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots \\ \mathbb{E}(\hat{a}_{i,n}\hat{a}_{i,1}) & \cdots & \mathbb{E}(\hat{a}_{i,n}\hat{a}_{i,i-1}) & -\mathbb{E}(\hat{a}_{i,n}) & \mathbb{E}(\hat{a}_{i,n}\hat{a}_{i,i+1}) & \cdots & \mathbb{E}(\hat{a}_{i,n}\hat{a}_{i,n}) \end{bmatrix} \tag{39}$$

$$\mathbb{E}(\boldsymbol{\epsilon}) = \left(m\sum_{i=1}^n \big[\mathbb{E}\big(\hat{a}_{i,0}\hat{a}_{i,1}\big) \cdots \mathbb{E}\big(\hat{a}_{i,0}\hat{a}_{i,i-1}\big) - \mathbb{E}\big(\hat{a}_{i,0}\big)\ \mathbb{E}\big(\hat{a}_{i,0}\hat{a}_{i,i+1}\big) \cdots \mathbb{E}\big(\hat{a}_{i,0}\hat{a}_{i,n}\big)\big] + \frac{1}{m}\sum_{k=1}^m \Big(\boldsymbol{x}^{(k)}\Big)^\top \mathbf{\Sigma}_{\boldsymbol{\epsilon}}^{-1}\right)\mathbf{\Sigma}_{\boldsymbol{\epsilon}} \tag{40}$$

Algorithm 4 Variational Bayesian EM

Require: Sensor measurements $\hat{\boldsymbol{x}}$, prior $\boldsymbol{a}$;

Initialize hyper-parameters Ω;
repeat
  Initialize calibration mean $\mathbb{E}(\boldsymbol{\epsilon}) = \boldsymbol{\epsilon}$ obtained by Algorithm 1 and the autocorrelation matrix $\mathbb{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top) = \mathbb{E}(\boldsymbol{\epsilon})\mathbb{E}(\boldsymbol{\epsilon})^\top$;
  repeat
    Update variational distribution $\mathcal{Q}(\hat{\boldsymbol{a}})$ with mean vector $\mathbb{E}(\hat{\boldsymbol{a}})$ and covariance matrix $\boldsymbol{\Sigma}_{\hat{\boldsymbol{a}}}$ by Equations (35) and (38);
    Update variational distribution $\mathcal{Q}(\boldsymbol{\epsilon})$ with mean vector $\mathbb{E}(\boldsymbol{\epsilon})$ and covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$ by Equations (39) and (40);
  until Convergence
  Update hyper-parameters Ω by Equations (42) and (44);
until Convergence
return hyper-parameters Ω.

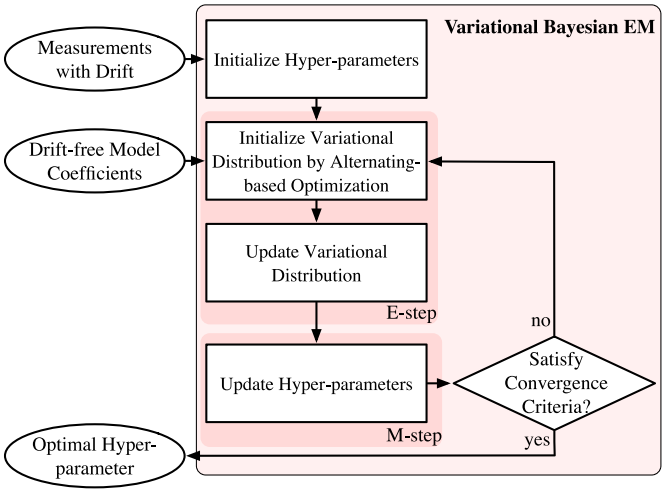

Fig. 6. Variational Bayesian EM flow.

to Equations (42)–(44). The variational E-step and the variational M-step are iteratively performed until the convergence criteria is satisfied. Finally, the optimal hyper-parameters are obtained. For a further illustration, our proposed variational Bayesian EM flow is shown in Fig. 6.

VI. Overall Flow

The overall flow of our proposed sensor drift calibration is shown in Fig. 7, which consists of four parts: 1) model optimization; 2) cross-validation; 3) Gibbs EM; and 4) variational Bayesian EM. With drift-free measurements model coefficients and several temperature measurements with drifts as inputs, an alternating-based optimization algorithm is proposed to resolve sensor drift calibration formulation in model optimization. Additionally, cross-validation, Gibbs EM, and variational Bayesian EM are adopted to induce hyper-parameters, respectively.

Compared with our previous work [30], our proposed VB-EM is used to replace cross-validation, Gibbs EM in our

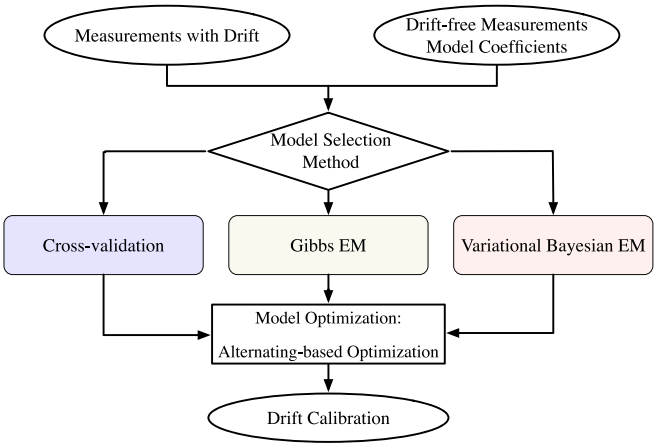

Fig. 7. Proposed sensor drift calibration flow.

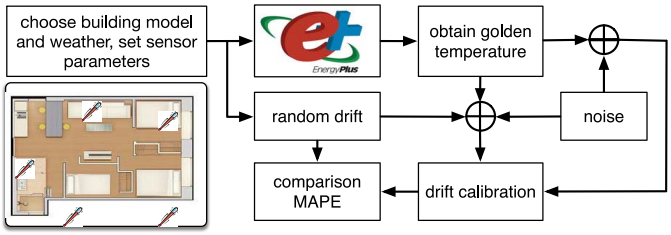

Fig. 8. Generated simulation data.

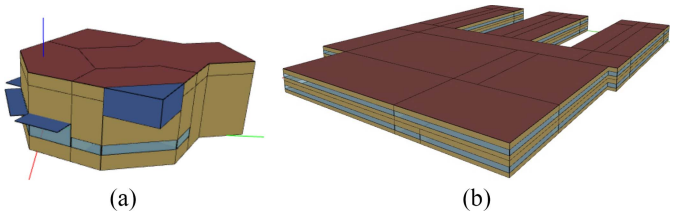

Fig. 9. Benchmark. (a) Hall. (b) Secondary School.

proposed sensor drift calibration flow as shown in Fig. 7. Gibbs EM is a stochastic approximation methodology while our proposed VB-EM is a deterministic approximation methodology. Therefore, the whole flow with our proposed VB-EM can achieve a better tradeoff between accuracy and runtime. Based on the aforementioned techniques, the proposed flow is expected to accurately calibrate sensor drifts.

VII. Experimental Results

The in-building temperature data are selected to test our proposed framework. We use several sensors for calibrate drifts. All data is directly generated from `EnergyPlus` as shown in Fig. 8. As illustrated in Fig. 9, the two building benchmarks, Hall [45] with Washington, D.C weather and Secondary School [46] with Chicago weather, are simulated by `EnergyPlus` to generate the ground-truth in-building temperatures, which are used to evaluate our proposed framework. The temperature sampling period is set to be one hour.

Practically, both drift and measurement noises need to be carefully considered and reasonably set to get close to real temperature measurement. Because of a slow-aging effect, time effects on sensor performance is not considered

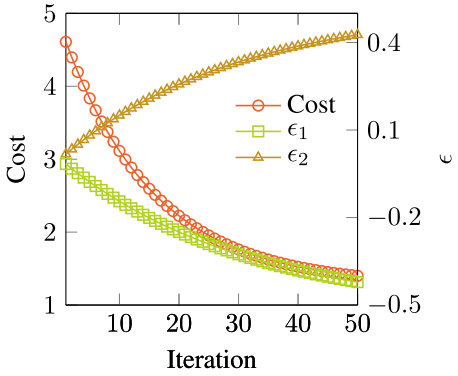

Fig. 10. Convergence of alternating-based method.

in our experiments. Drift is set to be time-invariant while measurement noise is configured as time-variant. According to the sensors' performance shown in Fig. 1(a), two low-cost temperature sensors, MCP9509 with accuracy ±4.5 °C and LM335A with accuracy ±5 °C as shown in Fig. 1(b) and (c), are chosen to set drift variance, respectively. According to the triple standard deviation, we set two drift variances to be $\sigma^2 = (4.5/3)^2 = 2.25$ and $\sigma^2 = (5/3)^2 = 2.78$. In addition, according to our survey, the noise variance is set to be 0.001 °C. All temperature measurements are generated by adding noise.

The time-instant number needs to be reasonably set to mimic practical application and accurately calibrate sensor drifts. We assume the temperature measurements are drift-free during first $m_0 = 240$ time-instants (first 10 days). And during $m = 60$ time-instants (60 h), the temperature measurements with drifts are utilized to test our proposed framework.

TSBL [21] and the proposed framework with cross-validation, Gibbs EM, and VB-EM are exploited to calibrate sensor drifts, respectively. All methods are implemented by `Python 2.7` on 12-core Linux machine with 256 G RAM and 2.80 GHz. 100 combinations of hyper-parameters ratios and $s = 5$ folds are set in cross-validation. In Gibbs EM, since the warm-start is waived in Gibbs sampling, to achieve better tradeoff between accuracy and runtime, only $L = 10$ samples are generated to perform Monte Carlo approximation (17), and three hyper-parameters λ, δ_0, δ_ϵ are initialized with 10^3, 10^{-4} and 10^{-3}, respectively. The threshold values of convergence criterion are set as 10^{-8}, 10^{-2}, and 10^{-3} in Algorithms 1, 3 and 4.

To verify the optimization and convergence of our proposed alternating-based algorithm as shown in Algorithm 1 more clearly, two sensors are used to calibrate sensor drifts and sense school temperatures in the second benchmark. In Fig. 10, we can obviously see that, by using our proposed alternating-based algorithm, the cost of Formulation (7), estimated calibrations ϵ_1 and ϵ_2 converge quickly to the stationary points.

The drift calibration accuracy is quantified by mean absolute percent error (MAPE) defined as follows:

$$\text{MAPE} = \frac{1}{nm}\sum_{k=1}^{m}\sum_{i=1}^{n}\left|\frac{\hat{\epsilon}_i^{(k)} - \epsilon_i}{\epsilon_i}\right| \tag{45}$$

where $\hat{\epsilon}_i^{(k)}$ is estimated calibration. Specifically, in our proposed framework, $\hat{\epsilon}_i^{(k)} = \hat{\epsilon}_i$. The sensor drift calibration performances of accuracy and runtime are shown in Figs. 11 and 12.

As shown in Fig. 12, TSBL has acceptable computational overhead even if its computational complexity is dominated by multiple matrix inversion operations. However, as displayed in Fig. 11, TSBL has the worst performance and robustness for drifts calibration. In fact, temperature signals lie in time-variant subspace since in-building temperatures are influenced by multiple time-variant factors, e.g., weather. As a result, TSBL cannot achieve an obvious drift calibration.

Unlike TSBL, the proposed spatial correlation model can calibrate drifts even if temperature signals lie in a time-variant subspace. Therefore, as shown in Fig. 11, the proposed framework with cross-validation or Gibbs EM or VB-EM outperforms TSBL in terms of accuracy. On average, the proposed framework can achieve about 3× accuracy improvement. Moreover, the proposed drift calibration framework with cross-validation can achieve the best accuracy. However, as shown in Fig. 12, cross-validation has a heavy computational overhead since we need to run Algorithm 1 for multiple times. Compared with cross-validation and TSBL, Gibbs EM and VB-EM have lower computation complexity. Furthermore, in order to make a better tradeoff between accuracy and runtime, Gibbs EM uses fewer samples to perform Monte Carlo approximation so that its accuracy is worse than VB-EM. However, as shown in Fig. 11, the proposed framework with Gibbs EM cannot achieve the best accuracy since Gibbs sampling is a stochastic method and VB-EM ignores the correlations between model coefficients $\mathbf{a}$ and calibration $\boldsymbol{\epsilon}$.

In Fig. 11, it can be seen that because of incremental correlation, the more sensors can achieve the higher accuracy of drift calibration by using our proposed framework. In practice, when fewer sensors need to be calibrated, in order to achieve better accuracy, cross-validation can be used to determine hyper-parameters within a reasonable response time, e.g., 1 min. While more sensors need to be calibrated, Gibbs EM can be exploited to determine hyper-parameters so that sensor measurement accuracy can be improved to a tolerable level within acceptable runtime. The proposed calibration framework with Gibbs EM can achieve a robust drift calibration and a better tradeoff between accuracy and runtime.

In order to illustrate the fact that the warm-start can be waived and the initialization is reasonable for samples in Gibbs sampling, 100 Gibbs sampling traces, including the first 10 sampling traces, are shown in Fig. 13. We can observe that the first 10 samples have the same shape with the 100 samples. It indicates that the first 10 samples have the same statistics with the 100 samples. Consequently, the first 10 samples can be used to represent the 100 samples.

In this application, $L = 10$ can make a better tradeoff between runtime and accuracy. More samples can improve the calibration accuracy. For illustrating the relationships among the number of samples, accuracy, and runtime, we show the experimental results in Fig. 14. We set the sampling number to be 10, 100, 1000, and 10 000. According to Fig. 11, because of the incremental correlation discussed above, the

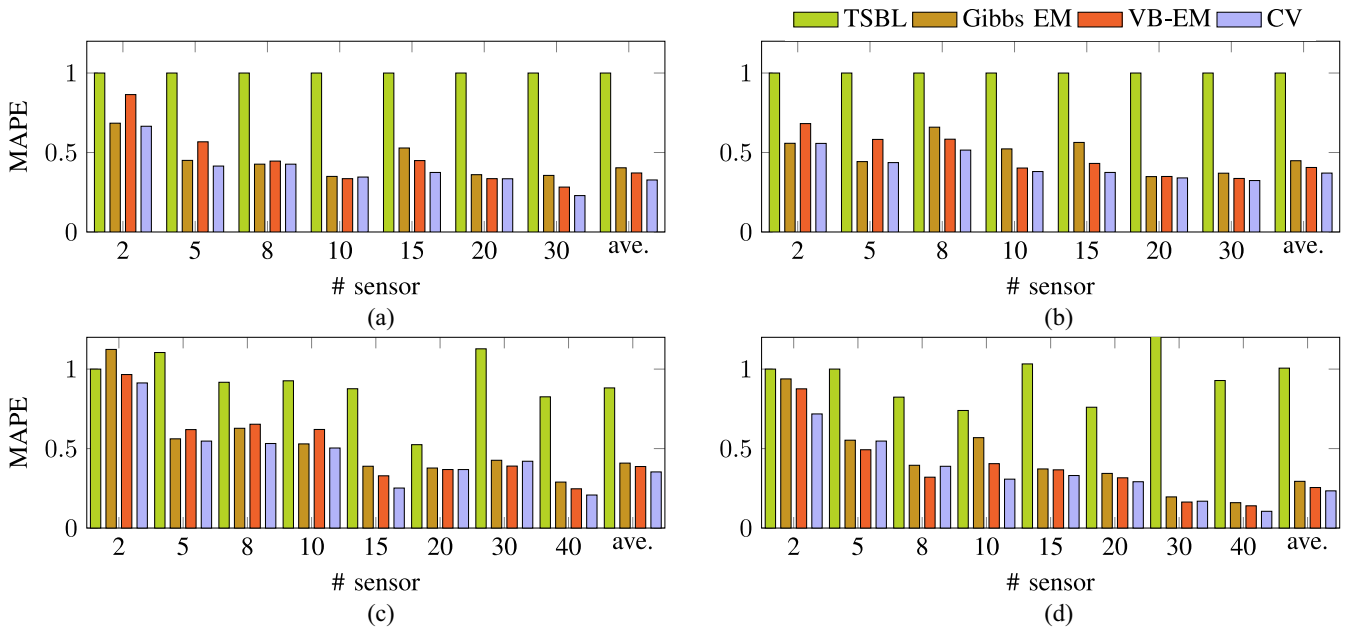

Fig. 11. Drift variance is set to (a) and (c) 2.25; (b) and (d) 2.78; Benchmark: (a) and (b) Hall; (c), and (d) Secondary School.

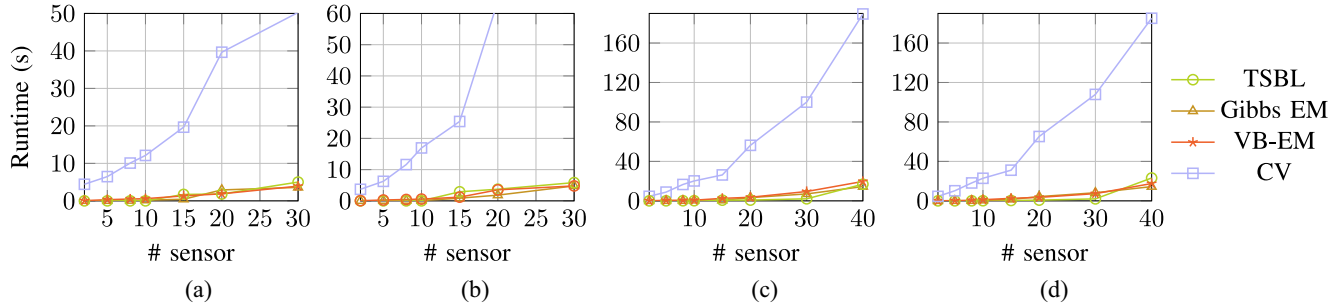

Fig. 12. Runtime versus # sensor: (a) and (c) 2.25; (b) and (d) 2.78; Benchmark: (a) and (b) Hall; (c) and (d) Secondary School.

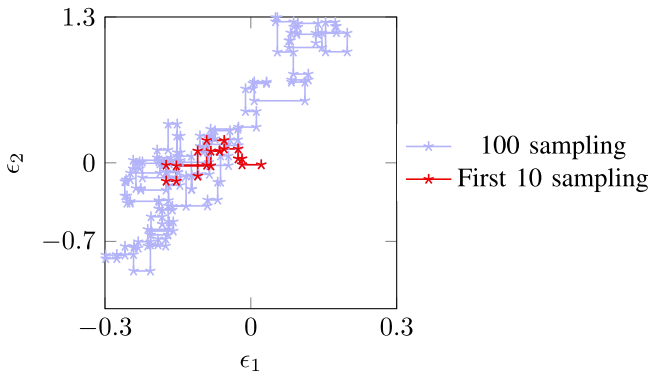

Fig. 13. Gibbs sampling traces.

drift calibration performance is more stable when the number of sensors is more than 15. Therefore, we average MAPE and runtime on different numbers of sensors when the number of sensors is more than 15. As the experimental results demonstrated in Fig. 14, the more number of samples can improve the calibration accuracy while it has more runtime. In addition, the more number of samples cannot achieve significant accuracy improvement while it leads to a significant runtime cost. Therefore, $L = 10$ is enough to make a better tradeoff between runtime and accuracy. Besides, based on the experimental results shown in Fig. 14, to achieve the same drift calibration accuracy with VB-EM, Gibbs EM needs 10 000 samples, which will incur a 30× runtime overhead. Therefore, our proposed VB-EM can achieve better a tradeoff between accuracy and runtime.

Furthermore, considering that the temperature sensors are used in a severe environment, in order to verify the robustness for noise, we set another noise variance to be 0.01 °C. We define $\Delta\text{MAPE} = \text{MAPE}_{0.001} - \text{MAPE}_{0.01}$, where $\text{MAPE}_{0.001}$ is MAPE under 0.001 noise variance and $\text{MAPE}_{0.01}$ is MAPE under 0.01 noise variance. $\Delta\text{MAPE} > 0$ means that noise brings a positive effect on drift calibration accuracy while $\Delta\text{MAPE} < 0$ means that noise affects negatively on drift calibration accuracy. Experimental results are shown in Fig. 15. Although it is uncertain that which effect the noise will bring on drift calibration accuracy under different sensor numbers in each individual experiment, noise brings a negative effect on drift calibration accuracy on average, i.e., $\sum \Delta\text{MAPE}$. What's more, the larger noise variance triggers drift calibration degradation by using our proposed methods, i.e., CV, Gibbs EM, and VB-EM. Our experimental results as shown in Fig. 15

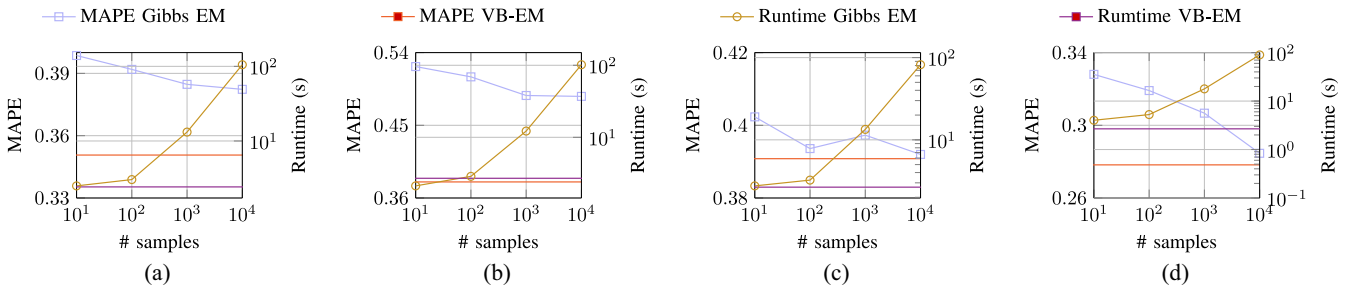

Fig. 14. # samples versus acc. versus runtime: (a) and (b) hall; (c) and (d) school; (a) and (c) drift variance 2.25; (b) and (d) drift variance 2.78.

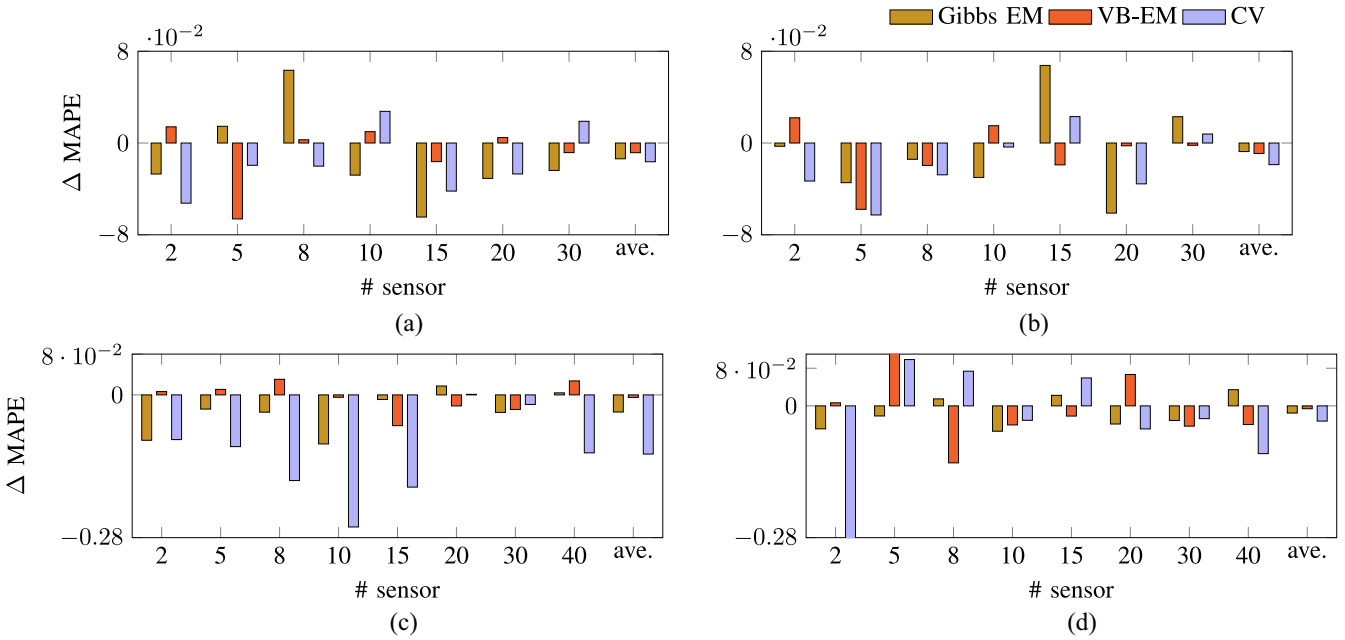

Fig. 15. Δ MAPE when noise variance is set 0.01, drift variance (a) and (c) 2.25; (b) and (d) 2.78; Benchmark: (a) and (b) Hall; (c) and (d) Secondary School.

indicate that compared with CV and Gibbs EM, our proposed framework with VB-EM is robust for noise.

VIII. Conclusion

In this article, a sensor spatial correlation model has been proposed to perform drift calibration. Thanks to spatial correlation, the unknown actual temperature measured by each sensor is linearly expressed by all other sensors. The priors for model coefficients and drift calibration are applied to MAP estimation. MAP estimation is then formulated as a nonconvex problem with three hyper-parameters, which is optimized by the proposed alternating-based method. Cross-validation, Gibbs EM, and VB-EM are exploited to determine hyper-parameters, respectively. Experimental results demonstrate that on benchmarks simulated from `EnergyPlus`, the proposed framework with variational Bayesian EM can achieve a robust drift calibration and a better tradeoff between accuracy and runtime. Averagely, compared with state-of-the-art, the proposed framework can achieve about 3× accuracy improvement. In order to achieve the same drift calibration accuracy with VB-EM, Gibbs EM needs 10 000 samples, which will result in a 30× runtime overhead.

Acknowledgment

The authors thank Prof. Xin Li for providing helpful comments.

References

[1] X. Chen, X. Li, and S. X.-D. Tan, "Overview of cyber-physical temperature estimation in smart buildings: From modeling to measurements," in *Proc. INFOCOM Workshops*, 2016, pp. 251–256.

[2] B. Lin and B. Yu, "Smart building uncertainty analysis via adaptive Lasso," *IET Cyber Phys. Syst. Theory Appl.*, vol. 2, no. 1, pp. 42–48, 2017.

[3] F. Farivar, M. S. Haghighi, A. Jolfaei, and M. Alazab, "Artificial intelligence for detection, estimation, and compensation of malicious attacks in nonlinear cyber-physical systems and industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2716–2725, Jan. 2020.

[4] B. Yu, J. Zhou, and S. Hu, "Cyber-physical systems: An overview," in *Big Data Analytics for Cyber-Physical Systems*. Cham, Switzerland: Springer, 2020, pp. 1–11.

[5] Q. Zhu, A. Sangiovanni-Vincentelli, S. Hu, and X. Li, "Design automation for cyber-physical systems," *Proc. IEEE*, vol. 106, no. 9, pp. 1479–1483, Sep. 2018.

[6] S. Adepu *et al.*, "Control behavior integrity for distributed cyber-physical systems," in *Proc. ACM/IEEE Int. Conf. Cyber Phys. Syst. (ICCPS)*, 2020, pp. 30–40.

[7] K. Ni *et al.*, "Sensor network data fault types," *ACM Trans. Sensor Netw.*, vol. 5, no. 3, p. 25, 2009.

[8] (2016). *Engineer's Guide to Accurate Sensor Measurements*. [Online]. Available: http://download.ni.com/evaluation/daq/25188_Sensor_WhitePaper_IA.pdf
[9] (2018). *Findchips*. [Online]. Available: https://www.findchips.com
[10] T. Zhang, Y. Zhang, Z. Li, Z. Song, and H. Liu, "System-level calibration for data fusion in wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 9, no. 3, p. 28, 2013.
[11] M. S. Stanković, S. S. Stanković, and K. H. Johansson, "Distributed blind calibration in lossy sensor networks via output synchronization," *IEEE Trans. Autom. Control*, vol. 60, no. 12, pp. 3257–3262, Dec. 2015.
[12] E. Miluzzo, N. D. Lane, A. T. Campbell, and R. Olfati-Saber, "Calibree: A self-calibration system for mobile sensor networks," in *Proc. Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, 2008, pp. 314–331.
[13] J. Feng, S. Megerian, and M. Potkonjak, "Model-based calibration for sensor networks," in *Proc. IEEE Sensors*, vol. 2, 2003, pp. 737–742.
[14] K. Whitehouse and D. Culler, "Calibration as parameter estimation in sensor networks," in *Proc. ACM Int. Workshop Wireless Sensor Netw. Appl.*, 2002, pp. 59–67.
[15] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for sensor self-calibration," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 3, 2004, pp. 861–864.
[16] Y. Wang, A. Yang, Z. Li, P. Wang, and H. Yang, "Blind drift calibration of sensor networks using signal space projection and Kalman filter," in *Proc. Int. Conf. Intell. Sensors Sensor Netw. Inf. Process. (ISSNIP)*, 2015, pp. 1–6.
[17] Z. Li, Y. Wang, A. Yang, and H. Yang, "Drift detection and calibration of sensor networks," in *Proc. IEEE Int. Conf. Wireless Commun. Signal Process. (WCSP)*, 2015, pp. 1–6.
[18] M. Takruri, S. Rajasegarar, S. Challa, C. Leckie, and M. Palaniswami, "Spatio-temporal modeling-based drift-aware wireless sensor networks," *IET Wireless Sensor Syst.*, vol. 1, no. 2, pp. 110–122, 2011.
[19] L. Balzano and R. Nowak, "Blind calibration of sensor networks," in *Proc. ACM Int. Symp. Inf. Process. Sensor Netw. (IPSN)*, 2007, pp. 79–88.
[20] J. Ding, A. V. Gribok, J. W. Hines, and B. Rasmussen, "Redundant sensor calibration monitoring using independent component analysis and principal component analysis," *Real Time Syst.*, vol. 27, no. 1, pp. 27–47, 2004.
[21] Y. Wang, A. Yang, Z. Li, X. Chen, P. Wang, and H. Yang, "Blind drift calibration of sensor networks using sparse Bayesian learning," *IEEE Sens. J.*, vol. 16, no. 16, pp. 6249–6260, Aug. 2016.
[22] Y. Wang, A. Yang, X. Chen, P. Wang, Y. Wang, and H. Yang, "A deep learning approach for blind drift calibration of sensor networks," *IEEE Sens. J.*, vol. 17, no. 13, pp. 4158–4171, Jul. 2017.
[23] Z. Zhang and B. D. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 5, pp. 912–926, Sep. 2011.
[24] S. Ling and T. Strohmer, "Self-calibration and bilinear inverse problems via linear least squares," *SIAM J. Imag. Sci.*, vol. 11, no. 1, pp. 252–292, 2018.
[25] X. Chen, X. Li, and S. X.-D. Tan, "From robust chip to smart building: CAD algorithms and methodologies for uncertainty analysis of building performance," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2015, pp. 457–464.
[26] T. Chen, B. Lin, H. Geng, and B. Yu, "Smart building sensor drift calibration," in *Big Data Analytics for Cyber-Physical Systems*. Cham, Switzerland: Springer, 2020, pp. 187–202.
[27] J. Yang, X. Zhong, and W. P. Tay, "A dynamic Bayesian nonparametric model for blind calibration of sensor networks," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3942–3953, Oct. 2018.
[28] C. Schulke, F. Caltagirone, F. Krzakala, and L. Zdeborová, "Blind calibration in compressed sensing using message passing algorithms," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2013, pp. 566–574.
[29] J. Yang, W. P. Tay, and X. Zhong, "A dynamic Bayesian nonparametric model for blind calibration of sensor networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2017, pp. 4207–4211.
[30] T. Chen, B. Lin, H. Geng, and B. Yu, "Sensor drift calibration via spatial correlation model in smart building," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2019, pp. 1–6.
[31] (2013). *2-Terminal IC Temperature Transducer*. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/AD590.pdf
[32] F. Wang *et al.*, "Bayesian model fusion: Large-scale performance modeling of analog and mixed-signal circuits by reusing early-stage data," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 8, pp. 1255–1268, Aug. 2016.
[33] Q. Huang, C. Fang, F. Yang, X. Zeng, and X. Li, "Efficient multivariate moment estimation via Bayesian model fusion for analog and mixed-signal circuits," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2015, p. 169.
[34] Q. Huang, C. Fang, F. Yang, X. Zeng, D. Zhou, and X. Li, "Efficient performance modeling via dual-prior Bayesian model fusion for analog and mixed-signal circuits," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2016, pp. 1–6.
[35] G. H. Golub and C. F. Van Loan, *Matrix Computations*. London, U.K.: JHU Press, 2012.
[36] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2009, pp. 689–696.
[37] P. V. Giampouras, A. A. Rontogiannis, and K. D. Koutroumbas, "Alternating iteratively reweighted least squares minimization for low-rank matrix factorization," *IEEE Trans. Signal Process.*, vol. 67, no. 2, pp. 490–503, Jan. 2019.
[38] A. Y. Ng, "Preventing overfitting of cross-validation data," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 97, 1997, pp. 245–253.
[39] M. Feurer, J. T. Springenberg, and F. Hutter, "Initializing Bayesian hyperparameter optimization via meta-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2015, pp. 1128–1135.
[40] Q. Zhang, M. Zhang, T. Chen, J. Fan, Z. Yang, and G. Li, "Electricity theft detection using generative models," in *Proc. IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, 2018, pp. 270–274.
[41] K. Ganchev, B. Taskar, and J. Gama, "Expectation maximization and posterior constraints," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2008, pp. 569–576.
[42] C. Robert, *Machine Learning, a Probabilistic Perspective*. New York, NY, USA: Taylor & Francis, 2014.
[43] T. Salimans, D. Kingma, and M. Welling, "Markov chain Monte Carlo and variational inference: Bridging the gap," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 1218–1226.
[44] J. Palmer, K. Kreutz-Delgado, B. D. Rao, and D. P. Wipf, "Variational EM algorithms for non-Gaussian latent variable models," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2006, pp. 1059–1066.
[45] (2018). *OpenStudio*. [Online]. Available: https://www.openstudio.net
[46] (2018). *National Renewable Energy Laboratory OpenStudio Standards*. [Online]. Available: https://github.com/NREL/openstudio-standards

Tinghuan Chen (Graduate Student Member, IEEE) received the B.Eng. and M.Eng. degrees in electronics engineering from Southeast University, Nanjing, China, in 2017 and 2014, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong.

His research interests include machine learning in cyber-physical systems and VLSI design for reliability.

Bingqing Lin received the Ph.D. degree from Nanyang Technological University, Singapore, in 2014.

He is currently an Assistant Professor with the College of Mathematics and Statistics, Shenzhen University, Shenzhen, China. His current research interests include machine learning and model selection.

Hao Geng (Graduate Student Member, IEEE) received the M.E. degree from the Department of Electronic Engineering and Information Sciences, University of Science and Technology of China, Hefei, China, in 2015, and the M.Sc. degree in machine learning from the Department of Computing, Imperial College London, London, U.K., in 2016. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong.

His research interests include machine learning, deep learning, and optimization methods with applications in VLSI CAD.

Shiyan Hu (Senior Member, IEEE) received the Ph.D. degree in computer engineering from Texas A&M University, College Station, TX, USA, in 2008.

He is a Professor and the Chair of cyber-physical system security with the University of Southampton, Southampton, U.K. His research interests include cyber-physical systems and cyber-physical system security. He has published more than 100 refereed papers in the above areas.

Prof. Hu is a recipient of the 2017 IEEE Computer Society TCSC Middle Career Researcher Award and the 2014 U.S. National Science Foundation CAREER Award. His publications have received a few distinctions, such as the 2018 IEEE Systems Journal Best Paper Award, and the 2017 Keynote Paper in the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the Front Cover Paper in the IEEE TRANSACTIONS ON NANOBIOSCIENCE in March 2014. He is the Chair for IEEE Technical Committee on Cyber-Physical Systems. He is the Editor-in-Chief of *IET Cyber-Physical Systems: Theory & Applications*. He is an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, the *ACM Transactions on Design Automation for Electronic Systems*, and the *ACM Transactions on Cyber-Physical Systems*. He is a Guest Editor for eight IEEE/ACM journals, such as the PROCEEDINGS OF THE IEEE and the IEEE TRANSACTIONS ON COMPUTERS. He has held chair positions in various conferences. He is a Fellow of IET and the British Computer Society. He is an ACM Distinguished Speaker, an IEEE Systems Council Distinguished Lecturer.

Bei Yu (Member, IEEE) received the Ph.D. degree from the University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Assistant Professor with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong.

Dr. Yu received six Best Paper Awards from International Conference on Tools with Artificial Intelligence 2019, *Integration, the VLSI Journal* in 2018, the International Symposium on Physical Design 2017, the SPIE Advanced Lithography Conference 2016, the International Conference on Computer-Aided Design 2013, and the Asia and South Pacific Design Automation Conference 2012, and the six ICCAD/ISPD contest awards. He has served as a TPC Chair of ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He is an Editor of the IEEE TCCPS Newsletter.