State-of-charge dependent equivalent circuit model identification for batteries using sparse Gaussian process regression

Kesen Fan[a], Yiming Wan[a,b,*], Benben Jiang[c,d]

[a]*School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China*
[b]*Key Laboratory of Image Processing and Intelligent Control, Huazhong University of Science and Technology, Ministry of Education, China*
[c]*Center for Intelligent and Networked Systems, Department of Automation, Tsinghua University, Beijing 100084, China*
[d]*Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China*

Abstract

Due to their ease of implementation, equivalent circuit models (ECMs) of batteries are widely used in battery management systems. Generally, ECM parameters vary with operating conditions, thus how such parameter dependencies are addressed substantially influences the accuracy of an ECM over a wide operating range. In this paper, we identify an ECM whose parameters have nonlinear dependence on state-of-charge (SOC). By transforming the SOC-dependent ECM into a linear parameter varying (LPV) input-output model, we propose a non-parametric sparse Gaussian process regression (GPR) approach, which alleviates the difficulty of specifying parametric functional SOC-dependencies of model parameters. The proposed approach derives the posterior distributions of ECM parameters, thus is capable to provide both parameter estimates and their associated uncertainties. The computational cost over large datasets is significantly reduced by adopting the sparse GPR. The proposed approach is applied to the above LPV model with two noise model structures, i.e., white and colored noises. Identification results using experimental data illustrate the

*Corresponding author
Email addresses: kesenfan@hust.edu.cn (Kesen Fan), ywan@hust.edu.cn (Yiming Wan), bbjiang@tsinghua.edu.cn (Benben Jiang)

efficacy of the proposed approach. The use of colored noise enhances robustness under different noise levels, and achieves higher output prediction accuracy over experimental datasets.

Keywords: Equivalent circuit model; state-of-charge dependent parameters; linear parameter varying system; sparse Gaussian process regression

1. Introduction

Batteries are essential components in various applications such as portable devices, power grids, and hybrid or electric vehicles [1, 2, 3]. To ensure safety and reliability, an advanced battery management system (BMS) is needed to monitor battery states and optimize its performance. All these functions rely on accurate battery models.

Existing battery models in literature can be classified into three categories: electrochemical model, data-driven model, and equivalent circuit model (ECM) [4, 5]. The ECM relies on connected circuit elements to represent the dominating electrochemical processes within a battery cell. Due to its structural simplicity and computational efficiency, the ECM has been widely used by BMSs. To capture nonlinear battery dynamics over a wide operating range, a nonlinear ECM is developed in [6, 7, 8, 9, 10] by allowing its parameters depend on state-of-charge (SOC), temperature, or current load.

To account for the above parameter dependencies on time-varying operating conditions, one popular approach adopts the linear ECM and performs online adaptive parameter estimation via recursive least-squares or Kalman filtering, see [11, 12, 13] for a non-exhaustive list of references. Another line of research aims at identifying functional relations of the above parameter dependencies to construct a nonlinear ECM. For this purpose, a straightforward approach is to identify multiple linear ECMs at a finite number of operating conditions, and then interpolates these linear ECM parameters to construct lookup tables or parametric functional relations over a wide operating range [6, 14]. In contrast, the alternative approach directly estimates the parametric functional

relations of the above ECM parameter dependencies by formulating an identification problem that minimizes the averaged squared output prediction error [7, 8, 9]. With the nonlinear parameter dependencies, the resulting identification problem is solved by resorting to metaheuristic algorithms [9] or nonlinear least-squares optimization [10]. When restricting the ECM parameter dependencies to be linear or affine, the obtained identification problem can be more efficiently solved by the least-squares or subspace methods [7, 8]. A common limitation of the above approaches is that it is non-trivial to determine suitable parametrized structures of the ECM parameter dependencies on operating conditions, as they may vary with the battery chemistry or age. An improper parametrized structure or under-parameterization results in parameter estimation bias, while over-parametrization may increase the variance of the identified model.

To address the above limitations of the parametric approaches in [7, 8, 9], we propose a non-parametric Gaussian process regression (GPR) approach in this paper to identify the SOC-dependent ECM. The proposed approach avoids or alleviates the difficulty of specifying the parametrized SOC-dependency structures, and provides a measure of uncertainty for its parameter estimates. Firstly, two linear parameter varying (LPV) models are derived from the SOC-dependent ECM: a LPV-ARX (autoregressive with exogenous inputs) model with a white noise term, and a LPV-ARMAX (autoregressive moving average with exogenous inputs) model with a first-order colored noise term. Inspired by [15], such SOC-dependencies of LPV model coefficients are described by non-parametric Gaussian processes (GPs), which are different from the parametrized functional forms adopted in [7, 8, 9]. With the GPs' hyperparameters learned from data, the posterior Gaussian distributions of the LPV model coefficients are computed to determine the ECM parameter estimates and their associated uncertainties. To be computationally tractable for large datasets, our proposed GPR based identification approach employs the sparse GPR in [16] that introduces a sparse approximation to the GP prior. The comparison between the use of LPV-ARX and LPV-ARMAX models is made by using experimental data. With the pro-

posed sparse GPR approach, the LPV-ARMAX model identification achieves smaller output prediction errors over different test datasets, and is more robust at a high noise level, compared to using the LPV-ARX model.

This journal paper extends the authors' previous conference paper [17] in the following aspects: 1) instead of using the exact GPR in [17], it adopts the sparse GPR approach which reduces computation complexity for large datasets; 2) instead of simply assuming white noises in [17], it additionally takes into account colored noises, which results in better identification performance at a high noise level; 3) instead of using only simulation data in [17], it uses experimental data to demonstrate the efficacy of the proposed approach.

To the best of our knowledge, there have been limited studies so far on using GPR for battery ECM identification, although GPR has been recently applied to SOC estimation, state-of-health estimation, and lifetime prediction for batteries [18, 19, 20, 21].

The rest of this paper is organized as follows. Section 2 states the SOC-dependent ECM identification problem. Section 3 briefly reviews sparse GPR. In Section 4, the sparse GPR approach to identify the SOC-dependent ECM is presented. Identification results via experimental data are discussed in Section 5. The concluding remarks are given in Section 6.

Notations. For a matrix $\boldsymbol{A}$, diag$\{\boldsymbol{A}\}$ represents a diagonal matrix that has the same diagonal elements as $\boldsymbol{A}$. A matrix denoted by diag $\begin{bmatrix}\boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_n\end{bmatrix}$ is block diagonal, with $\{\boldsymbol{A}_i\}_{i=1}^n$ being its diagonal blocks. It becomes a diagonal matrix if all $\boldsymbol{A}_i$'s are scalars. For a square matrix $\boldsymbol{A}$, $|\boldsymbol{A}|$ denotes its determinant. The identity matrix of dimension n is $\boldsymbol{I}_n$.

2. Model description and problem statement

In this section, we describe the continuous-time first-order ECM, and derive the corresponding discrete-time LPV model. Then, we state the SOC-dependent ECM parameter identification problem to be solved.

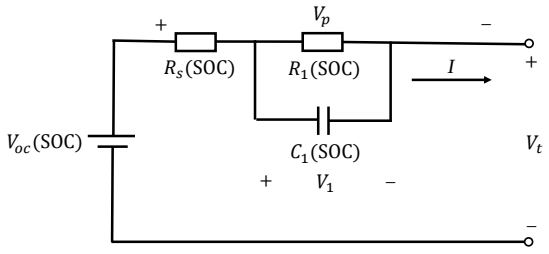

Figure 1: SOC-dependent first-order ECM

2.1. SOC-dependent equivalent circuit model

Due to its simplicity and accuracy, the first-order ECM in Fig. 1 is widely adopted in battery applications. The open circuit voltage (OCV) is denoted by V_{oc}. The internal resistance R_s is used to represent the ohmic polarization phenomenon. The polarization resistance R_1 and polarization capacitance C_1 are used to describe electrochemical polarization and concentration polarization, respectively, which reflect the transient dynamics under current excitation. Based on the circuit theory, the first-order ECM is expressed as:

$$\frac{dV_1}{dt} = \frac{I}{C_1(z)} - \frac{V_1}{R_1(z)C_1(z)}, \tag{1a}$$

$$V_t = V_{oc}(z) - R_s(z)I - V_1, \tag{1b}$$

where I is the load current with a positive value at discharge, V_t is the terminal voltage, V_1 denotes the overpotential voltage across R_1, and z represents SOC. In the above ECM, both model parameters and OCV have nonlinear dependence on the SOC z, as denoted by $R_s(z)$, $R_1(z)$, $C_1(z)$, and $V_{oc}(z)$. Such dependence of ECM parameters on SOC has been reported in [7, 9, 22], and will be illustrated later in Section 5.

Assumption 1. *Both the SOC z and the OCV-SOC relationship $V_{oc}(z)$ are available, which is a common assumption in the ECM identification literature such as [6, 11]. The SOC z is estimated with sufficient accuracy, using Coulomb*

counting with an accurate initial SOC. The OCV-SOC relationship $V_{oc}(z)$ is obtained from the OCV-SOC test [23].

Remark 1. *Without loss of generality, we assume a constant temperature regulated by the thermal management system or temperature chamber, and do not consider a higher-order ECM. If the considered battery is operated under highly dynamic load changes and a wide range of temperature, the proposed GPR based identification method can be slightly modified to address a high-order ECM with dependence on both SOC and temperature.*

To identify ECM parameters from sampled measurements, the continuous-time ECM in (1a) is transformed into a discrete-time model. With a sufficiently small sampling interval T_s, the current input and SOC between samples are assumed constant [7, 11, 24]. Then, it is derived from (1a) that

$$V_{1,k} = R_1(z_k)\left(1 - e^{-\frac{T_s}{\tau(z_k)}}\right) I_{k-1} + e^{-\frac{T_s}{\tau(z_k)}} V_{1,k-1}, \tag{2}$$

with $\tau(z_k) = R_1(z_k)C_1(z_k)$, $V_{1,k} = V_1(kT_s)$, $I_k = I(kT_s)$ and $z_k = z(kT_s)$. With $V_{t,k} = V_t(kT_s)$ representing the sampled terminal voltage, define

$$V_{p,k} = V_{oc}(z_k) - V_{t,k} = V_{1,k} + R_s(z_k)I_k. \tag{3}$$

Note that the second equation in (3) is derived from (1b). By substituting (3) into (2), we obtain the following LPV model:

$$\begin{aligned} V_{p,k} &= \theta_1(z_k)V_{p,k-1} + \theta_2(z_k, z_{k-1})I_{k-1} + \theta_3(z_k)I_k \\ &\approx \theta_1(z_k)V_{p,k-1} + \theta_2(z_k)I_{k-1} + \theta_3(z_k)I_k, \end{aligned} \tag{4}$$

where the above SOC-dependent coefficients are defined as

$$\theta_1(z_k) = e^{-\frac{T_s}{\tau(z_k)}},\ \theta_3(z_k) = R_s(z_k), \tag{5a}$$

$$\theta_2(z_k, z_{k-1}) = R_1(z_k)\left(1 - e^{-\frac{T_s}{\tau(z_k)}}\right) - R_s(z_{k-1})e^{-\frac{T_s}{\tau(z_k)}}, \tag{5b}$$

$$\theta_2(z_k) = R_1(z_k)\left(1 - e^{-\frac{T_s}{\tau(z_k)}}\right) - R_s(z_k)e^{-\frac{T_s}{\tau(z_k)}}. \tag{5c}$$

In the second line of (4), the approximation $\theta_2(z_k, z_{k-1}) \approx \theta_2(z_k)$ is adopted, since the difference between z_k and z_{k-1} is assumed negligible due to a sufficiently small sampling interval ($|z_k - z_{k-1}| \leq 6 \times 10^{-4}$ for the experimental datasets in Section 5).

Both the continuous-time ECM (1) and its transformed discrete-time LPV model (4) do not include any uncertainties. Considering noises in the measurement of $V_{t,k}$ and inexactness of the OCV-SOC relation $V_{oc}(z_k)$, $V_{p,k}$ determined by $V_{p,k} = V_{oc}(z_k) - V_t(z_k)$ in (3) is contaminated with inevitable errors. To describe such uncertainties, let $V_{p,k}$ and $\widetilde{V}_{p,k}$ denote the true value and its estimate, and (3) is modified as

$$\widetilde{V}_{p,k} = V_{oc}(z_k) - V_{t,k} = V_{p,k} + e_k \tag{6}$$

with e_k being the lumped error in $\widetilde{V}_{p,k}$. Then, the LPV model (4) becomes

$$\widetilde{V}_{p,k} = \theta_1(z_k)\widetilde{V}_{p,k-1} + \theta_2(z_k)I_{k-1} + \theta_3(z_k)I_k + \epsilon_k \tag{7}$$

with the noise term

$$\epsilon_k = e_k - \theta_1(z_k)e_{k-1}. \tag{8}$$

2.2. Problem statement

Due to their dependence on SOC, the ECM parameters vary with time. Various recursive identification methods have been reported in literature to adaptively estimate these time-varying ECM parameters [11, 12, 13]. However, such methods do not quantify the functional relations between these ECM parameters and SOC. In this paper, we aim at identifying the nonlinear SOC-dependencies of the internal resistance R_s, the polarization resistance R_1, and the time constant τ. Depending on how the noise term in (8) is addressed, the following two LPV models are considered:

1) As a common practice in the ECM identification literature [12], ϵ_k is approximated as a zero-mean Gaussian white noise denoted by v_k, and its variance σ^2 is unknown. In this case, (7) turns into the LPV-ARX model

$$\widetilde{V}_{p,k} = \theta_1(z_k)\widetilde{V}_{p,k-1} + \theta_2(z_k)I_{k-1} + \theta_3(z_k)I_k + v_k. \tag{9}$$

2) As inspired by (8), ϵ_k is approximated as a zero-mean first-order colored noise expressed by $\epsilon_k = v_k – cv_{k-1}$, where v_k is a zero-mean Gaussian white noise with an unknown variance σ^2, and c is a constant to be determined. The use of colored noise to improve model accuracy has been reported in literature such as [13, 25]. In this case, (7) becomes the LPV-ARMAX model

$$\widetilde{V}_{p,k} = \theta_1(z_k)\widetilde{V}_{p,k-1} + \theta_2(z_k)I_{k-1} + \theta_3(z_k)I_k + v_k - cv_{k-1}. \tag{10}$$

Since z_k and $V_{oc}(z_k)$ are available according to Assumption 1, $\widetilde{V}_{p,k}$ is computed from the measured $V_{t,k}$ according to (6). In this paper, we use $\{I_k, \widetilde{V}_{p,k}, z_k\}$ to identify the LPV-ARX model (9) or the LPV-ARMAX model (10), and then derive the nonlinear dependence of R_s, R_1, and τ on the SOC z.

3. Overview of sparse Gaussian Process Regression

A GP model is a probability distribution over functions, such that any finite number of function values follow the joint Gaussian distribution [26]. As a non-parametric Bayesian approach, GPR computes a posterior distribution over models, which in turn implies predicted output distributions. With a training dataset of size N, the exact GPR performs training at the computational cost $\mathcal{O}(N^3)$, and computes each predicted mean and variance at the costs of $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$, respectively [27]. These computations become intractable as the data size N grows to a few thousands. To address this issue, the sparse GPR proposed in [16] adopts an approximated GP prior, i.e., the Fully Independent Training Conditional (FITC) approximation, which will be utilized in this paper and concisely reviewed in this section.

Mathematically, a GP model can be written as [26]

$$f(x) \sim \mathcal{GP}(m_f(x), \kappa(x, x')), \tag{11a}$$

$$t_k = f(x_k) + n_k,\ n_k \sim \mathcal{N}(0, \sigma^2), \tag{11b}$$

where x_k is the input, t_k is the target output, n_k is the Gaussian white noise independent from x_k, and $\mathcal{N}(\cdot, \cdot)$ denotes a Gaussian distribution with the spec-

ified mean and covariance. In (11a), $f(x)$ is described as a GP with mean function $m_f(x)$ and covariance function (i.e., kernel function) $\kappa(x, x')$. The mean function $m_f(x)$ can be determined according to the a priori knowledge, or set to zero if no a priori knowledge is available. For the kernel function $\kappa(x, x')$, a typical choice is the squared exponential kernel function $\kappa(x, x') = \lambda e^{-\frac{(x-x')^2}{2\delta^2}}$. Then, the hyperparameters for (11b) are $\boldsymbol{\Theta} := [\lambda, \delta, \sigma]$.

For the sake of brevity, only the case of zero mean, i.e., $m_f(x) = 0$, is reviewed in the rest of this section. In a training dataset of size N, let $\boldsymbol{x} = \{x_i\}_{i=1}^N$, $\boldsymbol{t} = \{t_i\}_{i=1}^N$, and $\boldsymbol{f} = \{f(x_i)\}_{i=1}^N$ represent the inputs, observed target outputs, and the corresponding function values. The GPR problem aims at predicting the value of $f(x_*)$ at a new input x_*. Instead of using the exact GP prior, the sparse GPR uses an approximated GP prior by introducing the pseudo inputs $\boldsymbol{x_u} = \{x'_i\}_{i=1}^M$ and their associated pseudo targets $\boldsymbol{u} = \{f(x'_i)\}_{i=1}^M$, with $M \ll N$. Note that the pseudo targets $\boldsymbol{u}$ are not necessarily a subset of the available observations $\boldsymbol{t}$.

With the pseudo targets $\boldsymbol{u}$, the exact joint distribution of $\boldsymbol{f}$ and $f_* = f(x_*)$ can be expressed as

$$p(\boldsymbol{f}, f_*) = \int p(\boldsymbol{f}, f_*|\boldsymbol{u})p(\boldsymbol{u})\mathrm{d}\boldsymbol{u} \tag{12}$$

with $p(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K_{u,u}})$ and $[\boldsymbol{K_{u,u}}]_{ij} = \kappa(x'_i, x'_j)$. Assume that $\boldsymbol{f}$ and f_* are conditionally independent given $\boldsymbol{u}$. Then, the joint distribution in (12) can be approximated as

$$p(\boldsymbol{f}, f_*) \approx q(\boldsymbol{f}, f_*) = \int q(\boldsymbol{f}|\boldsymbol{u})q(f_*|\boldsymbol{u})p(\boldsymbol{u})\mathrm{d}\boldsymbol{u} \tag{13}$$

where $q(\boldsymbol{f}|\boldsymbol{u})$ and $q(f_*|\boldsymbol{u})$ are two approximate conditional distributions. In particular, the FITC approximation assumes [16, 27]

$$q(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{K_{f,u}}\boldsymbol{K_{u,u}^{-1}}\boldsymbol{u},\, \mathrm{diag}\{\boldsymbol{K_{f,f}} - \boldsymbol{Q_{f,f}}\}), \tag{14}$$

$$q(f_*|\boldsymbol{u}) = p(f_*|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{K_{*,u}}\boldsymbol{K_{u,u}^{-1}}\boldsymbol{u},\, K_{*,*} - Q_{*,*}), \tag{15}$$

with $[\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{f}}]_{ji} = [\boldsymbol{K}_{\boldsymbol{f},\boldsymbol{u}}]_{ij} = \kappa(x_i, x'_j)$, $[\boldsymbol{K}_{\boldsymbol{f},\boldsymbol{f}}]_{ij} = \kappa(x_i, x_j)$, $K_{*,*} = \kappa(x_*, x_*)$,

$$\boldsymbol{K}_{\boldsymbol{*},\boldsymbol{u}}^{\top} = \boldsymbol{K}_{\boldsymbol{u},*} = \begin{bmatrix} \kappa(x'_1, x_*) & \kappa(x'_2, x_*) & \dots & \kappa(x'_M, x_*) \end{bmatrix}^{\top},$$

$$\boldsymbol{Q}_{\boldsymbol{b},\boldsymbol{a}}^{\top} = \boldsymbol{Q}_{\boldsymbol{a},\boldsymbol{b}} = \boldsymbol{K}_{\boldsymbol{a},\boldsymbol{u}} \boldsymbol{K}_{\boldsymbol{u},\boldsymbol{u}}^{-1} \boldsymbol{K}_{\boldsymbol{u},\boldsymbol{b}},\ \boldsymbol{a} \text{ and } \boldsymbol{b} \text{ denote } \boldsymbol{f} \text{ or } *.$$

From (13)–(15), the joint distribution of $\boldsymbol{t} = \boldsymbol{f} + \boldsymbol{n}$ and f_* and the marginal distribution of $\boldsymbol{t}$ are approximated as

$$q(\boldsymbol{t}, f_*) = \mathcal{N}\left(\boldsymbol{0},\ \begin{bmatrix} \boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma} & \boldsymbol{Q}_{\boldsymbol{f},*} \\ \boldsymbol{Q}_{*,\boldsymbol{f}} & K_{*,*} \end{bmatrix}\right), \tag{16}$$

$$q(\boldsymbol{t}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma}), \tag{17}$$

with

$$\boldsymbol{\Gamma} = \mathrm{diag}\{\boldsymbol{K}_{\boldsymbol{f},\boldsymbol{f}} - \boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}}\} + \sigma^2 \boldsymbol{I}_N.$$

The hyperparameters $\boldsymbol{\Theta}$ are learned from data by maximizing the marginal likelihood $q(\boldsymbol{t})$ in (17). The corresponding objective function is

$$\begin{aligned} g(\boldsymbol{\Theta}) &= -\frac{1}{2}\boldsymbol{t}^{\top}(\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma})^{-1}\boldsymbol{t} - \frac{1}{2}\log|\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma}| \\ &= -\frac{1}{2}\boldsymbol{t}^{\top}(\boldsymbol{\Gamma}^{-1} - \boldsymbol{\Gamma}^{-1}\boldsymbol{K}_{\boldsymbol{f},\boldsymbol{u}}\boldsymbol{\Omega}^{-1}\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{f}}\boldsymbol{\Gamma}^{-1})\boldsymbol{t} \\ &\quad - \frac{1}{2}\log|\boldsymbol{\Omega}| + \frac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{u}}| - \frac{1}{2}\log|\boldsymbol{\Gamma}|, \end{aligned} \tag{18}$$

where $\boldsymbol{\Omega}$ is defined as

$$\boldsymbol{\Omega} = \boldsymbol{K}_{\boldsymbol{u},\boldsymbol{f}}\boldsymbol{\Gamma}^{-1}\boldsymbol{K}_{\boldsymbol{f},\boldsymbol{u}} + \boldsymbol{K}_{\boldsymbol{u},\boldsymbol{u}}, \tag{19}$$

and $|\cdot|$ denotes the determinant of a square matrix. The second equality in (18) is obtained by applying the matrix inversion lemma to $(\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma})^{-1}$ and applying the matrix determinant lemma to $|\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma}|$ [28].

With the above learned hyperparameters, f_* at the new input x_* is predicted from $\boldsymbol{t}$ by deriving the conditional distribution $q(f_*|\boldsymbol{t})$ from the joint distribution

$q(\boldsymbol{t}, f_*)$ in (16), i.e.,

$$\begin{aligned}\mathbb{E}[f_*|\boldsymbol{t}] &= \boldsymbol{Q}_{*,\boldsymbol{f}}(\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma})^{-1}\boldsymbol{t} \\ &= \boldsymbol{K}_{*,\boldsymbol{u}}\boldsymbol{\Omega}^{-1}\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{f}}\boldsymbol{\Gamma}^{-1}\boldsymbol{t}, \qquad (20a) \\ \mathrm{cov}[f_*|\boldsymbol{t}] &= K_{*,*} - \boldsymbol{Q}_{*,\boldsymbol{f}}(\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma})^{-1}\boldsymbol{Q}_{\boldsymbol{f},*} \\ &= K_{*,*} - Q_{*,*} + \boldsymbol{K}_{*,\boldsymbol{u}}\boldsymbol{\Omega}^{-1}\boldsymbol{K}_{\boldsymbol{u},*}, \qquad (20b)\end{aligned}$$

where the second equations in (20a) and (20b) are also obtained by applying the matrix inversion lemma to $(\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}} + \boldsymbol{\Gamma})^{-1}$, and $\boldsymbol{\Omega}$ is defined in (19).

With (18) and (20), the M-by-M matrix $\boldsymbol{\Omega}$, instead of the N-by-N matrix $\boldsymbol{Q}_{\boldsymbol{f},\boldsymbol{f}}+\boldsymbol{\Gamma}$, is inverted. In the training phase that involves (18), the computational cost for the sparse GPR is $\mathcal{O}(NM^2)$, which is much lower than $\mathcal{O}(N^3)$ for the exact GPR. As for the prediction phase using (20), $\boldsymbol{\Omega}^{-1}\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{f}}\boldsymbol{\Gamma}^{-1}\boldsymbol{t}$ and $\boldsymbol{\Omega}^{-1}$ can be precomputed, given the learned hyperparameters and the available data $\boldsymbol{t}$. Then, the computational costs for the predicted mean (20a) and variance (20b) are reduced to $\mathcal{O}(M)$ and $\mathcal{O}(M^2)$, respectively, which are much lower than $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ when using the exact GPR. In conclusion, compared to the exact GPR, using the sparse GPR results in significantly reduced computational costs if $M \ll N$.

4. Identification of SOC-dependent ECM parameters

In this section, a sparse GPR approach is proposed to identify the SOC-dependent ECM parameters in two steps: the LPV-ARX or LPV-ARMAX model coefficients $\{\theta_i(z)\}_{i=1}^3$ are first identified, and then used to derive the ECM parameters $R_s(z)$, $R_1(z)$ and $\tau(z)$ according to (5).

Inspired by [15], the sparse GPR approach describes the model coefficients $\{\theta_i(z)\}_{i=1}^3$ in (9) or (10) as three independent GPs. Rather than using zero-mean priors, we would like to use non-zero mean functions $\{m_i(z)\}_{i=1}^3$ which are estimates from conventional least-squares identification. This helps reduce parameter estimation errors and model prediction errors, as indicated by our previous results in [17]. With these above non-zero means $\{m_i(z)\}_{i=1}^3$, we derive

the following LPV-ARX and LPV-ARMAX models from (9) and (10) for the identification purpose:

$$\overline{y}_k = \overline{\theta}_1(z_k)\overline{x}_{1,k} + \overline{\theta}_2(z_k)\overline{x}_{2,k} + \overline{\theta}_3(z_k)\overline{x}_{3,k} + v_k, \tag{21}$$

$$\overline{y}_k = \overline{\theta}_1(z_k)\overline{x}_{1,k} + \overline{\theta}_2(z_k)\overline{x}_{2,k} + \overline{\theta}_3(z_k)\overline{x}_{3,k} + v_k - cv_{k-1}, \tag{22}$$

with

$$\overline{\theta}_1(z_k) = \theta_1(z_k) - m_1(z_k),\ \overline{\theta}_i(z_k) = \beta_i[\theta_i(z_k) - m_i(z_k)],\ i = 2, 3, \tag{23}$$

$$\overline{y}_k = \widetilde{V}_{p,k} - m_1(z_k)\widetilde{V}_{p,k-1} - m_2(z_k)I_{k-1} - m_3(z_k)I_k, \tag{24}$$

$$\overline{x}_{1,k} = \widetilde{V}_{p,k-1},\ \overline{x}_{2,k} = \frac{I_{k-1}}{\beta_2},\ \overline{x}_{3,k} = \frac{I_k}{\beta_3}. \tag{25}$$

By doing so, $\{\overline{\theta}_i(z)\}_{i=1}^3$ become zero-mean GPs, which can be addressed by the reviewed derivations in (14)–(20). Moreover, since the original model coefficients in (9) and (10) take values at vastly different orders of magnitudes, the positive scale factors $\{\beta_i\}_{i=2}^3$ are introduced such that the transformed model coefficients $\{\overline{\theta}_i(z)\}_{i=1}^3$ are at the same order of magnitude, which helps improve numerical condition for solving the identification problem.

4.1. Sparse GPR approach to LPV-ARX model identification

In this subsection, we consider the transformed LPV-ARX model (21) whose model coefficients $\{\overline{\theta}_i(z_k)\}_{i=1}^3$ are three independent GPs with zero means and covariance functions

$$\operatorname{cov}\left[\,\overline{\theta}_i(z_j), \overline{\theta}_i(z_l)\right] = \kappa^i(z_j, z_l) = \lambda_i e^{-\frac{(z_j - z_l)^2}{2\delta_i^2}},\ i = 1, 2, 3. \tag{26}$$

Suppose we have the training dataset

$$\boldsymbol{\mathcal{D}} = \{\overline{x}_{1,j}, \overline{x}_{2,j}, \overline{x}_{3,j}, z_j, \overline{y}_j\}_{j=1}^N.$$

In the following, the FITC approximation in Section 3 is applied to the three GPs $\{\overline{\theta}_i(z_k)\}_{i=1}^3$. First, the pseudo dataset is selected as $\{\boldsymbol{z}_{\overline{\boldsymbol{u}}}, \overline{\boldsymbol{u}}_1, \overline{\boldsymbol{u}}_2, \overline{\boldsymbol{u}}_3\}$:

$$\boldsymbol{z}_{\overline{\boldsymbol{u}}} = \{z'_j\}_{j=1}^M,\ \overline{\boldsymbol{u}}_i = \{\overline{\theta}_i(z'_j)\}_{j=1}^M,\ i = 1, 2, 3,$$

with $M \ll N$. Here, we use the same pseudo inputs $\boldsymbol{z}_{\overline{\boldsymbol{u}}}$ for the three GPs $\{\overline{\theta}_i(z_k)\}_{i=1}^3$, and choose them to be equally spaced over the considered SOC range. With the same procedure that derives (17) from (14)–(16), the FITC approximation to the a priori distribution of each coefficient vector

$$\overline{\boldsymbol{\theta}}_i = \begin{bmatrix} \overline{\theta}_i(z_1) & \overline{\theta}_i(z_2) & \cdots & \overline{\theta}_i(z_N) \end{bmatrix}^\top$$

is

$$q(\overline{\boldsymbol{\theta}}_i) = \mathcal{N}\left(\mathbf{0}, \boldsymbol{Q}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} + \mathrm{diag}\left\{\boldsymbol{K}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} - \boldsymbol{Q}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\right\}\right),\ i = 1, 2, 3,$$

with

$$\begin{aligned} &[\boldsymbol{K}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}]_{jl} = \kappa^i(z_j, z'_l),\ [\boldsymbol{K}^i_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}]_{jl} = \kappa^i(z'_j, z'_l), \\ &\boldsymbol{Q}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} = \boldsymbol{K}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}(\boldsymbol{K}^i_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2 \boldsymbol{I}_M)^{-1} \boldsymbol{K}^i_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}. \end{aligned} \tag{27}$$

The regularization factor $\omega \neq 0$ is added to ensure the invertibility of $\boldsymbol{K}^i_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2 \boldsymbol{I}_M$. The value of ω will be determined together with hyperparameters.

Next, we proceed to the measured output. According to (21), the output vector $\overline{\boldsymbol{y}} = \begin{bmatrix} \overline{y}_1 & \overline{y}_2 & \cdots & \overline{y}_N \end{bmatrix}^\top$ is expressed as

$$\overline{\boldsymbol{y}} = \overline{\boldsymbol{X}}\,\overline{\boldsymbol{\theta}} + \boldsymbol{v} \tag{28}$$

with

$$\overline{\boldsymbol{\theta}} = \begin{bmatrix} \overline{\boldsymbol{\theta}}_1^\top & \overline{\boldsymbol{\theta}}_2^\top & \overline{\boldsymbol{\theta}}_3^\top \end{bmatrix}^\top,\ \boldsymbol{v} = \begin{bmatrix} v_1 & \cdots & v_N \end{bmatrix}^\top, \tag{29}$$

$$\overline{\boldsymbol{X}} = \begin{bmatrix} \overline{\boldsymbol{X}}_1 & \overline{\boldsymbol{X}}_2 & \overline{\boldsymbol{X}}_3 \end{bmatrix},\ \overline{\boldsymbol{X}}_{\boldsymbol{i}} = \mathrm{diag}\begin{bmatrix} \overline{x}_{i,1} & \cdots & \overline{x}_{i,N} \end{bmatrix}. \tag{30}$$

Here, $\mathrm{diag}\begin{bmatrix} \overline{x}_{i,1} & \cdots & \overline{x}_{i,N} \end{bmatrix}$ is a diagonal matrix whose diagonal elements are $\{\overline{x}_{i,j}\}_{j=1}^N$. Therefore, the approximate joint distribution of $\overline{\boldsymbol{y}}$ in (28) is

$$q(\overline{\boldsymbol{y}}) = \mathcal{N}\left(\mathbf{0}, \overline{\boldsymbol{X}}\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top + \boldsymbol{\Lambda}\right), \tag{31}$$

$$\boldsymbol{\Lambda} = \overline{\boldsymbol{X}}\mathrm{diag}\left\{\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} - \boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\right\}\overline{\boldsymbol{X}}^\top + \boldsymbol{\Sigma}, \tag{32}$$

$$\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} = \mathrm{diag}\begin{bmatrix} \boldsymbol{Q}^1_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} & \boldsymbol{Q}^2_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} & \boldsymbol{Q}^3_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} \end{bmatrix}, \tag{33}$$

$$\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} = \mathrm{diag}\begin{bmatrix} \boldsymbol{K}^1_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} & \boldsymbol{K}^2_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} & \boldsymbol{K}^3_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} \end{bmatrix},$$

where $[\boldsymbol{K}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}]_{jl} = \kappa^i(z_j, z_l)$, and $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}_N$ is the covariance matrix of the noise $\boldsymbol{v}$ in (29). It can be seen that $\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}$ in (33) is a block diagonal matrix whose diagonal blocks are $\left\{\boldsymbol{Q}^i_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\right\}_{i=1}^{3}$. From (27), we rewrite $\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}$ in (33) as

$$\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} = \boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}} \left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2 \boldsymbol{I}_{3M}\right)^{-1} \boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}} \tag{34}$$

with $\boldsymbol{K}^{\top}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}} = \boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}} = \text{diag}\left[\boldsymbol{K}^1_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}} \quad \boldsymbol{K}^2_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}} \quad \boldsymbol{K}^3_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}\right]$, and

$$\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} = \text{diag}\left[\boldsymbol{K}^1_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} \quad \boldsymbol{K}^2_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} \quad \boldsymbol{K}^3_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}\right]. \tag{35}$$

Note that both $\overline{\boldsymbol{X}}\text{diag}\left\{\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} - \boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\right\}\overline{\boldsymbol{X}}^{\top}$ and $\boldsymbol{\Lambda}$ in (32) are diagonal matrices, according to (30) and $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}_N$.

Let the hyperparameters in (26) be denoted by $\boldsymbol{\lambda} = [\lambda_1 \ \lambda_2 \ \lambda_3]$ and $\boldsymbol{\delta} = [\delta_1 \ \delta_2 \ \delta_3]$. With a training dataset, the hyperparameters $\boldsymbol{\lambda}$ and $\boldsymbol{\delta}$, the regularization factor ω, and the noise standard deviation σ are determined by maximizing the marginal likelihood $q(\overline{\boldsymbol{y}})$ in (31). Similarly to (18), the formulated objective function to be optimized is derived as

$$\begin{aligned} & g_1(\boldsymbol{\lambda}, \boldsymbol{\delta}, \omega, \sigma) \\ = & -\frac{1}{2}\overline{\boldsymbol{y}}^{\top}\left(\boldsymbol{\Lambda}^{-1} - \boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^{\top}\boldsymbol{\Lambda}^{-1}\right)\overline{\boldsymbol{y}} \\ & -\frac{1}{2}\left(\log|\alpha_1\boldsymbol{\Pi}| - \log\left|\alpha_2(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2\boldsymbol{I}_{3M})\right| + \log|\alpha_3\boldsymbol{\Lambda}|\right), \end{aligned} \tag{36}$$

with

$$\boldsymbol{\Pi} = \boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^{\top}\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}} + \boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2\boldsymbol{I}_{3M}. \tag{37}$$

In (36), amplification factors $\{\alpha_i\}_{i=1}^3$ are introduced to address the risk of numerical underflow associated with the three log-determinant terms. This just adds a constant to the objective function (36), thus does not affect the optimal solution.

For a new input z_*, the joint probability distribution of the output vector $\overline{\boldsymbol{y}}$ and the model coefficients $\overline{\boldsymbol{\theta}}_* = \left[\overline{\theta}_1(z_*) \quad \overline{\theta}_2(z_*) \quad \overline{\theta}_3(z_*)\right]^{\top}$ is

$$q\left(\begin{bmatrix}\overline{\boldsymbol{y}} \\ \overline{\boldsymbol{\theta}}_*\end{bmatrix}\right) = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix}\overline{\boldsymbol{X}}\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^{\top} + \boldsymbol{\Lambda} & \boldsymbol{Q}_{\overline{\boldsymbol{y}},*} \\ \boldsymbol{Q}_{*,\overline{\boldsymbol{y}}} & \boldsymbol{K}_{*,*}\end{bmatrix}\right), \tag{38}$$

with

$$\boldsymbol{Q}_{*,\overline{\boldsymbol{y}}} = (\boldsymbol{Q}_{\overline{\boldsymbol{y}},*})^\top = \boldsymbol{Q}_{*,\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top, \tag{39a}$$

$$\boldsymbol{Q}_{*,\overline{\boldsymbol{\theta}}} = \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2\boldsymbol{I}_{3M}\right)^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}, \tag{39b}$$

$$\boldsymbol{K}_{*,\overline{\boldsymbol{u}}} = (\boldsymbol{K}_{*,\overline{\boldsymbol{u}}})^\top = \operatorname{diag}\left[\boldsymbol{K}^1_{*,\overline{\boldsymbol{u}}} \quad \boldsymbol{K}^2_{*,\overline{\boldsymbol{u}}} \quad \boldsymbol{K}^3_{*,\overline{\boldsymbol{u}}}\right],$$

$$\boldsymbol{K}^i_{\overline{\boldsymbol{u}},*} = (\boldsymbol{K}^i_{*,\overline{\boldsymbol{u}}})^\top = \left[\kappa^i(z_*, z'_1) \quad \cdots \quad \kappa^i(z_*, z'_M)\right]^\top,$$

$$\boldsymbol{K}_{*,*} = \operatorname{diag}\left[\kappa^1(z_*, z_*) \quad \kappa^2(z_*, z_*) \quad \kappa^3(z_*, z_*)\right].$$

From (38), given the output vector $\overline{\boldsymbol{y}}$, the posterior mean and variance of the model coefficients $\overline{\boldsymbol{\theta}}_*$ are

$$\mathbb{E}\left[\,\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}\right] = \boldsymbol{Q}_{*,\overline{\boldsymbol{y}}}\left(\overline{\boldsymbol{X}}\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top + \boldsymbol{\Lambda}\right)^{-1}\overline{\boldsymbol{y}}, \tag{40a}$$

$$\operatorname{cov}\left[\,\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}\right] = \boldsymbol{K}_{*,*} - \boldsymbol{Q}_{*,\overline{\boldsymbol{y}}}\left(\overline{\boldsymbol{X}}\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top + \boldsymbol{\Lambda}\right)^{-1}\boldsymbol{Q}_{\overline{\boldsymbol{y}},*}. \tag{40b}$$

Theorem 1. *The posterior mean and variance of the transformed model coefficients* $\overline{\boldsymbol{\theta}}_*$ *in* (40) *are equivalently expressed as*

$$\mathbb{E}[\,\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}] = \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Psi}\overline{\boldsymbol{y}}, \tag{41a}$$

$$\operatorname{cov}[\,\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}] = \boldsymbol{K}_{*,*} - \boldsymbol{Q}_{*,*} + \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*}, \tag{41b}$$

with $\boldsymbol{\Pi}$ *defined in* (37)*,*

$$\boldsymbol{\Psi} = \boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}, \tag{42a}$$

$$\boldsymbol{Q}_{*,*} = \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}} + \omega^2\boldsymbol{I}_{3M})^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*}. \tag{42b}$$

The detailed proof is referred to Appendix A.

Thanks to the sparse structure of $\overline{\boldsymbol{X}}$ in (30) and the diagonal structure of $\boldsymbol{\Lambda}$ in (32), the computational costs of solving (36) and computing the posterior mean and variance in (41) are $\mathcal{O}(NM^2)$, $\mathcal{O}(M)$, and $\mathcal{O}(M^2)$, respectively, which is the same as described in the last paragraph of Section 3. Note that $\boldsymbol{\Psi}\overline{\boldsymbol{y}}$ and $\boldsymbol{\Pi}^{-1}$ are precomputed before calculating the posterior prediction in (41).

4.2. Sparse GPR approach to LPV-ARMAX model identification

In this subsection, the sparse GPR approach proposed for the LPV-ARX model identification is extended to the LPV-ARMAX model (22) such that the

colored noise $v_k - cv_{k-1}$ is taken into account.

Still, the derivations in Section 4.1 can be applied to the LPV-ARMAX model (22). However, the noise covariance matrix $\boldsymbol{\Sigma}$ in (32) becomes tridiagonal, i.e.,

$$\boldsymbol{\Sigma} = \sigma^2 \begin{bmatrix} 1+c^2 & -c & & & \\ -c & 1+c^2 & -c & & \\ & \ddots & \ddots & \ddots & \\ & & -c & 1+c^2 & -c \\ & & & -c & 1+c^2 \end{bmatrix} \tag{43}$$

for the colored noise vector $\begin{bmatrix} v_1 - cv_0 & v_2 - cv_1 & \cdots & v_N - cv_{N-1} \end{bmatrix}^\top$, rather than a diagonal matrix $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}_N$ for the white noise vector. In this case, $\boldsymbol{\Lambda}$ in (32) for the LPV-ARMAX model also becomes a tridiagonal matrix, since $\overline{\boldsymbol{X}} \mathrm{diag}\left\{ \boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} - \boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} \right\} \overline{\boldsymbol{X}}^\top$ remains diagonal according to (30). To address the above tridiagonal structure of $\boldsymbol{\Lambda}$ and the additional hyperparameter c in (43), we will revised the training phase as follows. Using the learned hyperparameters, computing the posterior predictions remains the same as in (41a) and (41b).

To exploit the tridiagonal structure of $\boldsymbol{\Lambda}$ in the following computations, we first perform the LU decomposition [28] for $\boldsymbol{\Lambda}$:

$$\boldsymbol{\Lambda} = \boldsymbol{L}_{\boldsymbol{\Lambda}} \boldsymbol{U}_{\boldsymbol{\Lambda}}, \tag{44}$$

$$\boldsymbol{L}_{\boldsymbol{\Lambda}} = \begin{bmatrix} 1 & & & & \\ p_2 & 1 & & & \\ & p_3 & 1 & & \\ & & \ddots & \ddots & \\ & & & p_N & 1 \end{bmatrix}, \; \boldsymbol{U}_{\boldsymbol{\Lambda}} = \begin{bmatrix} q_1 & -c\sigma^2 & & & \\ & q_2 & -c\sigma^2 & & \\ & & \ddots & \ddots & \\ & & & q_{N-1} & -c\sigma^2 \\ & & & & q_N \end{bmatrix}, \tag{45}$$

$$q_1 = \sigma^2(1+c^2) + b_1, \; p_i = -\frac{c\sigma^2}{q_{i-1}},$$
$$q_i = \sigma^2(1+c^2) + b_i + p_i c\sigma^2, \; i = 2, \ldots, N,$$

where b_i is the ith diagonal element of the diagonal matrix $\overline{\boldsymbol{X}} \mathrm{diag}\left\{ \boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} - \boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}} \right\} \overline{\boldsymbol{X}}^\top$. Note that $\boldsymbol{L}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{U}_{\boldsymbol{\Lambda}}$ in (45) are banded matrices whose bandwidths are both 1.

Next, we consider training hyperparameters. Unlike the LPV-ARX model, there is an extra unknown constant parameter c in (43) to be determined. To train hyperparameters including c, the objective function

$$g_2(\boldsymbol{\lambda}, \boldsymbol{\delta}, \omega, \sigma, c) = g_1(\boldsymbol{\lambda}, \boldsymbol{\delta}, \omega, \sigma) + \gamma(c - c_0)^2 \tag{46}$$

is modified from $g_1(\boldsymbol{\lambda}, \boldsymbol{\delta}, \omega, \sigma)$ in (36) by including an extra regularization term $\gamma(c-c_0)^2$, where $\gamma > 0$ is a weighting coefficient, and c_0 is predetermined to be the priori approximate of $\theta_1(z)$ according to (8). With the regularization term in (46), the difference between c and c_0 is penalized.

With the LU decomposition of $\boldsymbol{\Lambda}$ in (44), the computational cost of inverting the M-by-M matrix $\boldsymbol{\Pi}$ in (37) is analyzed as follows. According to

$$\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top \boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}} = \boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top \boldsymbol{U}_{\boldsymbol{\Lambda}}^{-1}\boldsymbol{L}_{\boldsymbol{\Lambda}}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}, \tag{47}$$

we first perform the LU decomposition to derive $\boldsymbol{U}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{L}_{\boldsymbol{\Lambda}}$ at the computational cost $\mathcal{O}(N)$, then compute $\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top \boldsymbol{U}_{\boldsymbol{\Lambda}}^{-1}$ and $\boldsymbol{L}_{\boldsymbol{\Lambda}}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}$ by exploiting the banded structure of $\boldsymbol{U}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{L}_{\boldsymbol{\Lambda}}$, both at the cost $\mathcal{O}(MN)$ [29]. The multiplication of $\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top \boldsymbol{U}_{\boldsymbol{\Lambda}}^{-1}$ and $\boldsymbol{L}_{\boldsymbol{\Lambda}}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}$ in (47) is obtained at the cost $\mathcal{O}(NM^2)$, and the inverse of $\boldsymbol{\Pi}$ is calculated at the cost $\mathcal{O}(M^3)$. Therefore, the overall cost of computing $\boldsymbol{\Pi}^{-1}$ is still $\mathcal{O}(NM^2)$ since $M \ll N$. For this reason, the computational cost of training hyperparameters by maximizing (46) is $\mathcal{O}(NM^2)$. As for the posterior prediction, the equations in (41) are still valid for the LPV-ARMAX model, whose computations are divided into two steps. Firstly, $\boldsymbol{\Psi}\overline{\boldsymbol{y}}$ and $\boldsymbol{\Pi}^{-1}$ are precomputed, which needs to utilize the LU decomposition of $\boldsymbol{\Lambda}$ as mentioned before. With such precomputation, calculating the posterior mean and variance for (41a) and (41b) are still at the costs $\mathcal{O}(M)$ and $\mathcal{O}(M^2)$, respectively.

4.3. From LPV model coefficients to ECM parameters

From the transformed model coefficients $\overline{\boldsymbol{\theta}}_* = \begin{bmatrix}\overline{\theta}_1(z_*) & \overline{\theta}_2(z_*) & \overline{\theta}_3(z_*)\end{bmatrix}^\top$ obtained in Section 4.1 or 4.2, the posterior means and variances of the original

model coefficients $\boldsymbol{\theta}_* = \begin{bmatrix}\theta_1(z_*) & \theta_2(z_*) & \theta_3(z_*)\end{bmatrix}^\top$ are

$$\mathbb{E}[\boldsymbol{\theta}_*|\boldsymbol{\mathcal{D}}] = \boldsymbol{m}_* + \mathbf{S}\mathbb{E}[\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}],\ \mathrm{cov}[\boldsymbol{\theta}_*|\boldsymbol{\mathcal{D}}] = \mathbf{S}\mathrm{cov}[\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}]\mathbf{S}^\top \tag{48}$$

according to (23), with $\boldsymbol{m}_* = [m_1(z_*)\ m_2(z_*)\ m_3(z_*)]^\top$ and $\mathbf{S} = \mathrm{diag}[1\ \beta_2^{-1}\ \beta_3^{-1}]$. According to (5), the ECM parameters $\{R_s, R_1, \tau\}$ can be expressed as

$$\begin{aligned} R_s(z_*) &= \theta_3(z_*),\ \tau(z_*) = -\frac{T_s}{\ln\theta_1(z_*)}, \\ R_1(z_*) &= \frac{\theta_2(z_*) + \theta_3(z_*)\theta_1(z_*)}{1-\theta_1(z_*)}. \end{aligned} \tag{49}$$

Together with (48), their posterior means and variances are derived as

$$\mathbb{E}[R_s(z_*)|\boldsymbol{\mathcal{D}}] = \mu_3(z_*), \tag{50a}$$

$$\mathrm{cov}[R_s(z_*)|\boldsymbol{\mathcal{D}}] = \mathrm{cov}[\,\theta_3(z_*)|\boldsymbol{\mathcal{D}}], \tag{50b}$$

$$\mathbb{E}[\tau(z_*)|\boldsymbol{\mathcal{D}}] \approx -\frac{T_s}{\ln\mu_1(z_*)}, \tag{50c}$$

$$\mathrm{cov}[\tau(z_*)|\boldsymbol{\mathcal{D}}] \approx \frac{T_s^2}{[\ln\mu_1(z_*)]^4\mu_1^2(z_*)}\mathrm{cov}[\,\theta_1(z_*)|\boldsymbol{\mathcal{D}}], \tag{50d}$$

$$\mathbb{E}[R_1(z_*)|\boldsymbol{\mathcal{D}}] \approx \frac{\mu_2(z_*) + \mu_3(z_*)\mu_1(z_*)}{1-\mu_1(z_*)}, \tag{50e}$$

$$\mathrm{cov}[R_1(z_*)|\boldsymbol{\mathcal{D}}] \approx \boldsymbol{J}(z_*)\mathrm{cov}[\boldsymbol{\theta}_*|\boldsymbol{\mathcal{D}}]\boldsymbol{J}^\top(z_*), \tag{50f}$$

where $\mu_i(z_*)$ denotes the ith element of $\mathbb{E}[\,\boldsymbol{\theta}_*|\boldsymbol{\mathcal{D}}]$ in (48), $\mathrm{cov}[\,\theta_i(z_*)|\boldsymbol{\mathcal{D}}]$ is the ith diagonal element of the covariance matrix $\mathrm{cov}[\boldsymbol{\theta}_*|\boldsymbol{\mathcal{D}}]$ given in (48), and $\boldsymbol{J}(z_*)$ is defined as

$$\boldsymbol{J}(z_*) = \left.\begin{bmatrix}\frac{\partial R_1}{\partial\theta_1} & \frac{\partial R_1}{\partial\theta_2} & \frac{\partial R_1}{\partial\theta_3}\end{bmatrix}\right|_{\theta_i(z_*)=\mu_i(z_*),i=1,2,3}, \tag{51}$$

with R_1 in (49). Note that $\tau(z_*)$ and $R_1(z_*)$ are actually non-Gaussian distributed due to their nonlinear dependence on $\{\theta_i(z_*)\}_{i=1}^3$ in (49). Here we adopt the first-order Taylor expansions of the functional dependencies $\tau(z_*)$ and $R_1(z_*)$ to derive the approximated means and covariances in (50c)–(50f).

4.4. Summary of the proposed identification algorithm

Based on the OCV-SOC relationship $V_{oc}(z)$ and available data V_t and I over the considered SOC range, the proposed sparse GPR based identification algorithm is summarized below.

Step 1. Prepare identification data $\{I_k, \widetilde{V}_{p,k}, z_k\}_{k=1}^N$

1a) For $1 \leq k \leq N$, estimate the SOC z_k from the current input by Coulomb counting, determine the corresponding OCV $V_{oc}(z_k)$, and compute the LPV model output $\widetilde{V}_{p,k} = V_{oc}(z_k) - V_{t,k}$;

1b) Determine the mean functions $\{m_i(z)\}_{i=1}^3$ from preliminary identification results or the a prior knowledge (how we do it in our case study is described in the first paragraph of Section 5.1), select the scale factors $\{\beta_i\}_{i=2}^3$ in (23), and compute the transformed regressors in (25) and the transformed output in (24).

Step 2. Learn hyperparameters of the sparse GPR model

2a) Choose zero means and covariance functions in (26) for the GPs of LPV model coefficients $\{\overline{\theta}_i(z)\}_{i=1}^3$;

2b) Select the pseudo inputs $\boldsymbol{z}_{\overline{\boldsymbol{u}}} = \{z_i'\}_{i=1}^M$ to be equally spaced over the SOC range in the dataset;

2c) Determine the hyperparameters of the LPV-ARX or LPV-ARMAX model by maximizing (36) or (46), respectively, where $\{\alpha_i\}_{i=1}^3$, γ, and c_0 are properly specified. Multiple initial guesses can be used to alleviate the problem of local maxima.

Step 3. Calculate SOC-dependencies of ECM parameters

3a) Obtain the posterior mean and variance functions of $\{\overline{\theta}_i(z_*)\}_{i=1}^3$ and $\{\theta_i(z_*)\}_{i=1}^3$ according to (41) and (48), respectively.

3b) Obtain the approximated posterior mean and variance functions of $R_s(z_*)$, $\tau(z_*)$, and $R_1(z_*)$ according to (50) and (51).

Note that the above algorithm for the LPV-ARMAX model differs from that for the LPV-ARX model in only Steps 2c) and 3a). Firstly, the objective functions (36) and (46) for learning hyperparameters in Step 2c) are different. Secondly, the matrix $\boldsymbol{\Lambda}$ in (32) is tridiagonal for the LPV-ARMAX model due to $\boldsymbol{\Sigma}$ in (43), whilst $\boldsymbol{\Lambda}$ is diagonal for the LPV-ARX model. The LU decomposition of $\boldsymbol{\Lambda}$ in (44) is additionally performed for the LPV-ARMAX model when

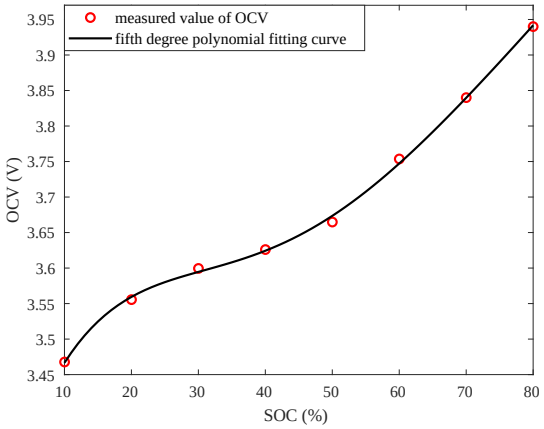

Figure 2: The OCV-SOC curve fitted by a 5-degree polynomial.

learning hyperparameters in Step 2c) and computing the posterior mean and variance in Step 3a).

5. Identification results using experimental datasets

This section presents the identification results of the proposed GPR approach using experimental data of 18650 LiNiMnCoO2/Graphite lithium-ion cells released by Center for Advanced Life Cycle Engineering at University of Maryland [23]. All experimental data used in this section were collected at the room temperature 25°C. The cell capacity is 2Ah. With data from the incremental-current OCV test, the OCV-SOC curve is approximated by the 5-degree polynomial $V_{oc}(z) = 6.77z^5 - 21.6z^4 + 25.9z^3 - 13.6z^2 + 3.51z + 3.23$, which is shown in Fig. 2.

The Federal Urban Driving Schedule (FUDS) dataset is used to identify the SOC-dependent ECM parameters. To test the robustness of identification algorithms, two FUDS datasets at different noise levels are prepared: the one at a low noise level is the original FUDS dataset of size $N = 9700$, and the other one at a high noise level is generated by artificially adding a white noise

onto the measured terminal voltage V_t in the original FUDS dataset, where the added white noise has zero mean and standard deviation 2×10^{-3} volt.

For comparison, the following four algorithms are implemented to identify the SOC-dependent ECM over the 10–80% SOC range:

1) Multi-ARX algorithm, i.e., the multiple ARX model based approach. Firstly, we divide the 10–80% SOC range into a number of intervals of equal length. Over each SOC interval, a time-invariant ARX model in the form of (9) is identified by using the MATLAB function `arx`, and its model coefficients are used to derive the corresponding linear ECM parameters according to (49) [6, 14]. Then, these linear ECMs over all SOC intervals are linearly interpolated to given the estimates of the functional SOC-dependencies $R_s(z)$, $R_1(z)$, and $C_1(z)$.
2) Multi-ARMAX algorithm, i.e., the multiple ARMAX model based approach. The same procedures in a) are used, but with the MATLAB function `armax` applied to identify a time-invariant ARMAX model in the form of (10) over each SOC interval [25, 13].
3) SGPR-LPV-ARX algorithm, i.e., the proposed sparse GPR approach using the LPV-ARX model (21), as presented in Sections 4.1 and 4.4.
4) SGPR-LPV-ARMAX algorithm, i.e., the proposed sparse GPR approach using the LPV-ARMAX model (22), as presented in Sections 4.2 and 4.4.

Since the true ECM parameters of the experimental lithium-ion cells are unknown, the associated parameter estimation errors cannot be computed to evaluate the identification performance of the above algorithms. Instead, we consider the following two aspects for performance evaluation:

Firstly, for each algorithm, we examine the consistency of its obtained ECM parameter estimates at different noise levels. For this purpose, each algorithm is applied to two FUDS datasets contaminated with low and high levels of voltage measurement noises, as mentioned in the second paragraph of this section.

Secondly, for comparison among all algorithms, we evaluate the accuracy of the identified ECMs by root mean square errors (RMSEs) of one-step pre-

dicted terminal voltages using test datasets. Specifically, three test datasets under different dynamic current profiles are used, i.e., the US06 Highway Driving Schedule, the Dynamic Stress Test (DST), and the Beijing Dynamic Stress Test (BJDST) [23]. The one-step predicted terminal voltage is computed as $\widehat{V}_{t,k} = V_{oc}(z_k) - \widehat{V}_{p,k}$, $\widehat{V}_{p,k} = \theta_1(z_k)\widetilde{V}_{p,k-1} + \theta_2(z_k)I_{k-1} + \theta_3(z_k)I_k$ and $\widetilde{V}_{p,k-1} = V_{oc}(z_{k-1}) - V_{t,k-1}$, where $V_{t,k}$ is the true terminal voltage, z_k is determined by Coulomb counting according to Assumption 1. The corresponding RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{L}\sum_{k=1}^{L}\left(V_{t,k} - \widehat{V}_{t,k}\right)^2},$$

with L being the number of samples in a test dataset. A smaller RMSE indicates higher model accuracy.

5.1. Results of multi-ARX and SGPR-LPV-ARX algorithms

In this subsection, the above two ARX-based algorithms multi-ARX and SGPR-LPV-ARX are compared, with the following parameter settings. For the multi-ARX algorithm, the 10-80% SOC range is divided into 10 intervals of equal length. The SGPR-LPV-ARX algorithm summarized in Section 4.4 is implemented as follows. In Step 1b), the a priori mean functions $\{m_i(z)\}_{i=1}^3$ are assumed to be constant, and they are determined by identifying a time-invariant ARX model using the entire FUDS dataset over the 10-80% SOC range. Specifically, the a priori means obtained at the low noise level are $m_1(z) = 0.95, m_2(z) = -0.0668, m_3(z) = 0.0716,$ while those obtained at the high noise level are $m_1(z) = 0.92, m_2(z) = -0.0644, m_3(z) = 0.0716$. In addition, $\beta_2 = \beta_3 = 50$ is set. In Step 2b), the pseudo inputs $\boldsymbol{z}_{\overline{\boldsymbol{u}}}$ consist of $M = 36$ equally spaced SOCs over the 10-80% SOC range. Since the pseudo inputs $\boldsymbol{z}_{\overline{\boldsymbol{u}}}$ have a substantially reduced size compared to $N = 9700$ input samples in the FUDS dataset, the computational cost of the sparse GPR approach is significantly lower than that of the exact GPR approach, as pointed out at the end of Section 3. In Step 2c), the amplification factors in (36) are $\alpha_1 = \alpha_2 = 10^3$ and $\alpha_3 = 1$. The optimization solver `fmincon` in MATLAB is used to maximize

(36) by adopting the Newton method. The hyperparameters learned at the low noise level are

$$\lambda_1 = 0.06,\ \lambda_2 = 5.27,\ \lambda_3 = 6.57,\ \sigma = 5.45 \times 10^{-4},$$
$$\delta_1 = 0.40,\ \delta_2 = 0.42, \delta_3 = 0.63,\ \omega = 2.79 \times 10^{-3},$$

while those learned at the high noise level are

$$\lambda_1 = 0.68,\ \lambda_2 = 4.43,\ \lambda_3 = 3.35,\ \sigma = 2.82 \times 10^{-3},$$
$$\delta_1 = 0.76,\ \delta_2 = 0.07, \delta_3 = 0.43,\ \omega = 9.73 \times 10^{-3}.$$

The ECM parameter estimates obtained by the multi-ARX algorithm at low and high noise levels are depicted in Fig. 3. With this algorithm, one set of ECM parameters is identified over each SOC interval, and their values indeed vary with SOC. The estimates of R_1 and τ at the low noise level are inconsistent with those at the high noise level. The reason is that the ARX model does not explicitly consider the time correlation of the colored noise ϵ_k in (8). Due to this reason, the ECM parameter estimates at the high noise level are more biased, since the simplified white noise assumption becomes less valid. As a result, the ECM identified at the high noise level gives much larger RMSEs of the one-step predicted terminal voltages under test datasets, as listed in Table 1.

Due to the same reason as above, the SGPR-LPV-ARX algorithm also gives inconsistent estimates at different noise levels. This can be seen from the posterior means of the LPV-ARX model coefficients $\{\theta_i(z)\}_{i=1}^3$ and the ECM parameters $\{R_s(z), R_1(z), \tau(z)\}$ in Fig. 4. Moreover, compared to the results at the low noise level, the larger confidence regions of $\theta_1(z)$, $\theta_2(z)$, $R_1(z)$, and $\tau(z)$ at the high noise level indicate significant increase in their estimation uncertainties. Similarly to the multi-ARX algorithm, the SGPR-LPV-ARX algorithm also results in more biased estimates as the noise level increases. Consequently, the SGPR-LPV-ARX algorithm gives much larger RMSEs of the one-step predictions from the ECM identified at the high noise level, as shown in Table 1.

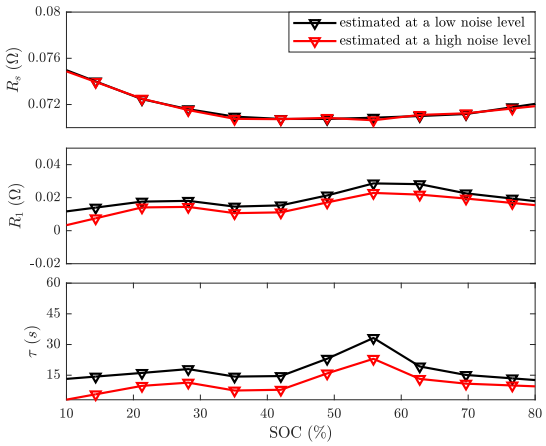

Figure 3: Identification results of the multi-ARX algorithm at low and high noise levels. Each triangle represents an estimate at one SOC interval, and linearly interpolation is used to construct the functional SOC-dependence of each ECM parameter.

Table 1: Using ECMs identified at different noise levels, RMSEs (unit: volt) of one-step predicted terminal voltages are compared under BJDST, US06, and DST conditions.

Identification method	Test datasets		
	BJDST	US06	DST
With identification dataset at a low noise level			
Multi-ARX	1.191×10^{-3}	1.391×10^{-3}	8.691×10^{-4}
SGPR-LPV-ARX	1.167×10^{-3}	1.418×10^{-3}	8.722×10^{-4}
Multi-ARMAX	1.190×10^{-3}	1.389×10^{-3}	8.684×10^{-4}
SGPR-LPV-ARMAX	7.063×10^{-4}	9.926×10^{-4}	6.968×10^{-4}
With identification dataset at a high noise level			
Multi-ARX	2.187×10^{-3}	1.463×10^{-3}	1.453×10^{-3}
SGPR-LPV-ARX	2.002×10^{-3}	1.476×10^{-3}	1.386×10^{-3}
Multi-ARMAX	1.053×10^{-3}	1.159×10^{-3}	7.649×10^{-4}
SGPR-LPV-ARMAX	7.176×10^{-4}	9.863×10^{-4}	6.945×10^{-4}

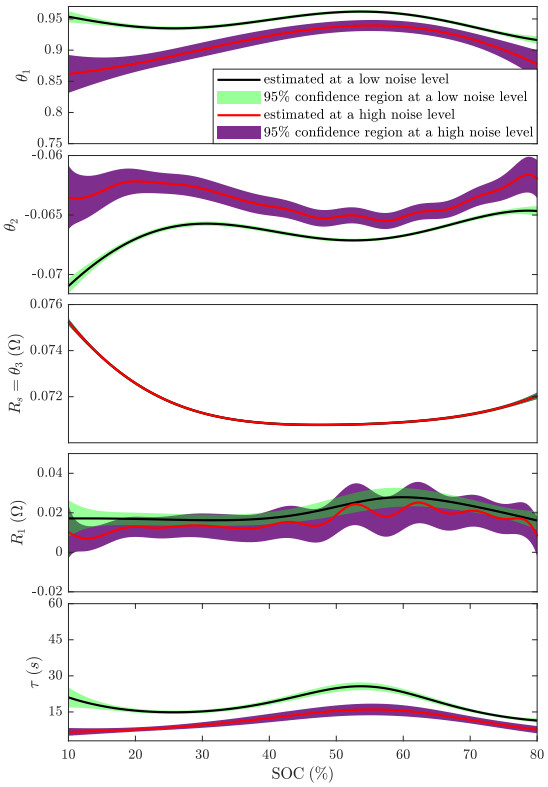

Figure 4: Identification results of the SGPR-LPV-ARX algorithm at low and high noise levels.

It can be further seen from Table 1 that the multi-ARX and SGPR-LPV-ARX algorithms have similar prediction performance at either low or high noise level, although they both result in much larger RMSEs of one-step predictions when they are applied at the high noise level.

It should be noted in Fig. 4 that the confidence regions of $R_1(z)$ and $\tau(z)$ are much wider than those of $\theta_1(z)$ and $\theta_2(z)$. Since the estimation uncertainties of $\{\theta_i(z)\}_{i=1}^3$ shown in Fig. 4 are relatively small, the wider confidence regions of $R_1(z)$ and $\tau(z)$ are attributed to the inevitable transformation (49) from $\{\theta_i(z)\}_{i=1}^3$ to the ECM parameters. The estimates of $R_1(z)$ and $\tau(z)$ derived from (49) are more sensitive to the estimation uncertainties of $\{\theta_i(z)\}_{i=1}^3$. In particular, since $\theta_1(z_*)$ is close to 1, $\ln\theta_1(z_*)$ and $1-\theta_1(z_*)$ are both close to zero, hence the estimation uncertainties are magnified when they propagate from the estimates of $\{\theta_i(z)\}_{i=1}^3$ to $R_1(z)$ and $\tau(z)$ via (49).

5.2. Results of multi-ARMAX and SGPR-LPV-ARMAX algorithms

In this subsection, it will be shown that by explicitly addressing the colored noise ϵ_k in (8), the multi-ARMAX and SGPR-LPV-ARMAX algorithms give better identification performance than their ARX counterparts in Section 5.1, and the SGPR-LPV-ARMAX algorithm achieves the best performance.

The parameter settings for implementing the multi-ARMAX and SGPR-LPV-ARMAX algorithms are the same as described in the first paragraph of Section 5.1, except that the tuning parameters presented in (46) are different, i.e., $\alpha_1=\alpha_2=10^4$, $\alpha_3=1$, $c_0=0.9$ and $\gamma=10^4$. The obtained hyperparameters for the SGPR-LPV-ARMAX algorithm are

$$\lambda_1=0.07,\ \lambda_2=0.64,\ \lambda_3=0.21,\ \sigma=2.06\times10^{-3},$$
$$\delta_1=0.68,\ \delta_2=0.95,\delta_3=0.71,\ \omega=1.3\times10^{-3},\ c=0.92,$$

at the low noise level, and

$$\lambda_1=0.08,\ \lambda_2=0.66,\ \lambda_3=0.89,\ \sigma=2.85\times10^{-3},$$
$$\delta_1=0.64,\ \delta_2=0.73,\delta_3=0.97,\ \omega=1.5\times10^{-3},\ c=0.91,$$

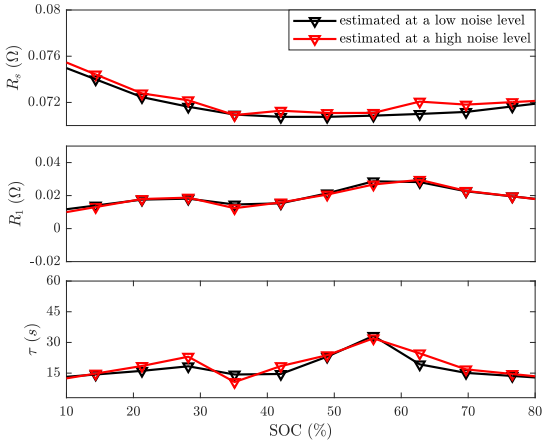

Figure 5: Identification results of the multiple ARMAX model approach at low and high noise levels. Each triangle represents an estimate at one SOC interval, and linearly interpolation is used to construct the functional SOC-dependence of each ECM parameter.

at the high noise level.

As expected, the multi-ARMAX algorithm gives more consistent estimates at different noise levels in Fig. 5, compared to the results of the multi-ARX algorithm depicted in Fig. 3. This is because that the multi-ARMAX algorithm achieves less biased estimation by taking into account the colored noise ϵ_k. As a result, it can be seen from Table 1 that with the identification dataset at a high noise level, the ECM identified by the multi-ARMAX algorithm gives much smaller prediction errors than those given by the multi-ARX algorithm.

The above observations and conclusions are also true for the comparison between the SGPR-LPV-ARMAX and SGPR-LPV-ARX algorithms. The posterior means at different noise levels in Fig. 6 are more consistent and less biased than those given by the SGPR-LPV-ARX algorithm in Fig. 4. Moreover, as the noise level becomes higher, the estimation uncertainties indicated by the confidence regions in Fig. 6 have only a mild increase, compared to the results given by the SGPR-LPV-ARX algorithm in Fig. 4. All these improvements are again attributed to coping with the colored noise ϵ_k.

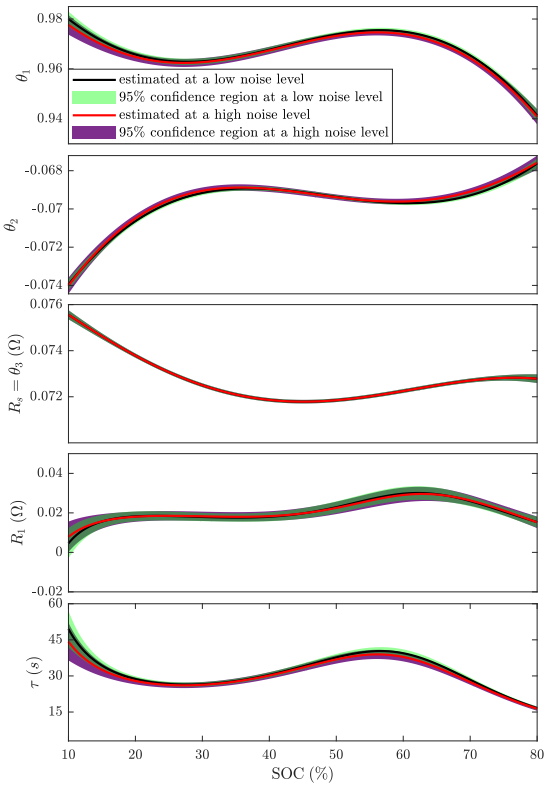

Figure 6: Identification results of the SGPR-LPV-ARMAX algorithm at low and high noise levels.

It can be also seen from Table 1 that with identification datasets at both two noise levels, the ECM identified by the SGPR-LPV-ARMAX algorithm achieves the smallest prediction errors, compared to all the other three algorithms.

As shown in Fig. 6, the estimation uncertainties of $R_1(z)$ and $\tau(z)$ are larger than those of $\theta_2(z)$ and $\theta_3(z)$. This is also due to the increased sensitivity of $R_1(z)$ and $\tau(z)$ in (49) to the variations of $\{\theta_i(z)\}_{i=1}^3$, which has been already explained in the last paragraph of Section 5.1.

5.3. Comparison between exact and sparse GPR based algorithms

In this subsection, the proposed SGPR-LPV-ARX and SGPR-LPV-ARMAX algorithms are compared with their exact GPR based versions, i.e., GPR-LPV-ARX and GPR-LPV-ARMAX, using datasets at a low noise level. Note that GPR-LPV-ARX was reported in [17]. Firstly, we compare their averaged computation time of 1) one iteration of optimizing (36) or (46) in hyperparameter training and 2) making the posterior prediction (50) at a new SOC condition z_*. Secondly, we compare their prediction accuracy in terms of one-step predicted terminal voltages using the same test datasets of Table 1. All algorithms are implemented in MATLAB R2020b, on a computer with 2.90GHz CPU and 16GB RAM. It takes more than one hours to implement just one iteration in optimizing (36) or (46) for the exact GPR over the entire training dataset consisting of 9700 data samples. To avoid such intractable computational load in the comparison, we use one data segment of 2923 samples over the 50–70% SOC range instead.

The parameter settings of the SGPR-LPV-ARX and SGPR-LPV-ARMAX algorithms are the same as described in Sections 5.1 and 5.2, except that the pseudo inputs $\boldsymbol{z}_{\overline{\boldsymbol{u}}}$ consist of $M = 11$ equally spaced SOCs over 50–70% SOC range. The parameter settings of the GPR-LPV-ARX and GPR-LPV-ARMAX algorithms are the same as their sparse versions. The obtained hyperparameters are as follows:

- SGPR-LPV-ARX

$$\lambda_1 = 2.6 \times 10^{-4},\ \lambda_2 = 4.9 \times 10^{-2},\ \lambda_3 = 8.3 \times 10^{-4},\ \sigma = 4.1 \times 10^{-4},$$
$$\delta_1 = 4.6 \times 10^{-2},\ \delta_2 = 2.4 \times 10^{-1},\ \delta_3 = 7.7 \times 10^{-3},\ \omega = 6.3 \times 10^{-3};$$

- SGPR-LPV-ARMAX

$$\lambda_1 = 0.31,\ \lambda_2 = 0.53,\ \lambda_3 = 0.82,\ \sigma = 3.3 \times 10^{-3},$$
$$\delta_1 = 0.92,\ \delta_2 = 0.51,\ \delta_3 = 0.41,\ \omega = 5.0 \times 10^{-3},\ c = 0.9;$$

- GPR-LPV-ARX

$$\lambda_1 = 3.54,\ \lambda_2 = 3.96 \times 10^{-4},\ \lambda_3 = 58.5,\ \sigma = 5.08 \times 10^{-4},$$
$$\delta_1 = 8.87,\ \delta_2 = 1.76 \times 10^{-7},\ \delta_3 = 4.91;$$

- GPR-LPV-ARMAX

$$\lambda_1 = 0.069,\ \lambda_2 = 0.45,\ \lambda_3 = 0.23,\ \sigma = 5 \times 10^{-3},$$
$$\delta_1 = 0.75,\ \delta_2 = 0.084,\ \delta_3 = 0.91,\ c = 0.92.$$

Table 2: Averaged computation time (unit: second) for exact and sparse GPR based algorithms.

Identification method	One iteration in training hyperparameter	Posterior prediction
GPR-LPV-ARX	83.49	2.67×10^{-3}
SGPR-LPV-ARX	0.88	4.26×10^{-5}
GPR-LPV-ARMAX	83.40	2.63×10^{-3}
SGPR-LPV-ARMAX	0.84	4.80×10^{-5}

The averaged computation time for the above four algorithms are shown in Table 2, while their prediction RMSEs under test datasets are listed in Table 3. It can be seen that using the sparse GPR results in a huge reduction in the computation time, with only a slight loss in prediction performance.

Table 3: RMSEs (unit: volt) of one-step predicted terminal voltages of exact and sparse GPR based algorithms under BJDST, US06, and DST conditions.

Identification method	Test datasets		
	BJDST	US06	DST
GPR-LPV-ARX	1.020×10^{-3}	1.392×10^{-3}	8.635×10^{-4}
SGPR-LPV-ARX	1.048×10^{-3}	1.384×10^{-3}	8.909×10^{-4}
GPR-LPV-ARMAX	6.775×10^{-4}	9.674×10^{-4}	6.956×10^{-4}
SGPR-LPV-ARMAX	7.276×10^{-4}	9.794×10^{-4}	6.903×10^{-4}

6. Conclusions

The ECM parameters generally vary with battery operating conditions. This imposes a non-trivial difficulty for the ECM identification problem, if the dependencies of ECM parameters on the operating conditions need to be learned from data. In this paper, the non-parametric sparse GPR approach is proposed for the identification of SOC-dependent ECM. It avoids the difficulty of specifying the parametric functional SOC-dependencies of ECM parameters, and quantifies uncertainties associated with its obtained parameter estimates. Instead of using the exact GPR, the sparse GPR is adopted in our proposed identification approach to significantly reduce the computation complexity over large datasets. With different noise model structures, both the LPV-ARX and LPV-ARMAX models are considered. Identification results show that using the LPV-ARMAX model with a colored noise term enhances robustness to the noise level, and achieves smaller output prediction errors over test datasets.

Appendix A. Proof of Theorem 1

According to the matrix inversion lemma, (34), and (37), we have

$$\left(\overline{\boldsymbol{X}}\boldsymbol{Q}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^{\top}+\boldsymbol{\Lambda}\right)^{-1}=\boldsymbol{\Lambda}^{-1}-\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^{\top}\boldsymbol{\Lambda}^{-1}. \tag{A.1}$$

From (37), (39), (40a), and (A.1), it is derived that

$$\begin{aligned}
\mathbb{E}[\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}] &= \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M}\right)^{-1}\Big(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1} \\
&\quad -\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\Big)\,\overline{\boldsymbol{y}} \\
&= \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M}\right)^{-1}\left(\boldsymbol{I}_{3M}-\left(\boldsymbol{\Pi}-\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}-\omega^2\boldsymbol{I}_{3M}\right)\boldsymbol{\Pi}^{-1}\right) \\
&\quad \cdot\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{y}} \\
&= \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M}\right)^{-1}(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M})\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{y}} \\
&= \boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{y}} \qquad \text{(A.2)}
\end{aligned}$$

which can be expressed as (41a) with $\boldsymbol{\Psi}$ in (42a).

With the derivations in (A.2), $\operatorname{cov}\left[\,\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}\right]$ in (40b) is rewritten as

$$\begin{aligned}
\operatorname{cov}\left[\,\overline{\boldsymbol{\theta}}_*|\boldsymbol{\mathcal{D}}\right] &= \boldsymbol{K}_{*,*}-\boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\boldsymbol{Q}_{\overline{\boldsymbol{y}},*} \\
&= \boldsymbol{K}_{*,*}-\boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{\theta}}}\overline{\boldsymbol{X}}^\top\boldsymbol{\Lambda}^{-1}\overline{\boldsymbol{X}}\boldsymbol{K}_{\overline{\boldsymbol{\theta}},\overline{\boldsymbol{u}}}\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M}\right)^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*} \\
&= \boldsymbol{K}_{*,*}-\boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}(\boldsymbol{\Pi}-\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}-\omega^2\boldsymbol{I}_{3M})\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M}\right)^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*} \\
&= \boldsymbol{K}_{*,*}-\boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\left(\boldsymbol{K}_{\overline{\boldsymbol{u}},\overline{\boldsymbol{u}}}+\omega^2\boldsymbol{I}_{3M}\right)^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*}+\boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*} \\
&= \boldsymbol{K}_{*,*}-\boldsymbol{Q}_{*,*}+\boldsymbol{K}_{*,\overline{\boldsymbol{u}}}\boldsymbol{\Pi}^{-1}\boldsymbol{K}_{\overline{\boldsymbol{u}},*}
\end{aligned}$$

using $\boldsymbol{Q}_{*,*}$ defined in (42b).

Acknowledgment

This work is supported by the National Natural Science Foundation of China, Grant No. 61803163 and No. 61603024.

References

[1] M. T. Lawder, B. Suthar, P. W. Northrop, S. De, C. M. Hoff, O. Leitermann, M. L. Crow, S. Santhanagopalan, V. R. Subramanian, Battery energy storage system and battery management system for grid-scale applications, Proceedings of IEEE 102 (6) (2014) 1014–1030.

[2] L. Lu, X. Han, J. Li, J. Hua, M. Ouyang, A review on the key issues for lithium-ion battery management in electric vehicles, Journal of Power Sources 226 (2013) 272–288.

[3] R. Wang, X. Zhang, J. Bao, A self-interested distributed economic model predictive control approach to battery energy storage networks, Journal of Process Control 73 (2019) 9–18.

[4] U. Krewer, F. Röder, E. Harinath, R. D. Braatz, B. Bedürftig, R. Findeisen, Dynamic models of Li-ion batteries for diagnosis and operation: a review and perspective, Journal of The Electrochemical Society 165 (16) (2018) A3656–A3673.

[5] X. Lin, Y. Kim, S. Mohan, J. B. Siegel, A. G. Stefanopoulou, Modeling and estimation for advanced battery management, Annual Review of Control, Robotics, and Autonomous Systems 2 (2019) 393–426.

[6] T. Huria, M. Ceraolo, J. Gazzarri, R. Jackey, High fidelity electrical model with thermal dependence for characterization and simulation of high power lithium battery cells, in: Proceedings of 2012 IEEE International Electric Vehicle Conference, Greenville, SC, USA, 2012, pp. 1–8.

[7] Y. Hu, S. Yurkovich, Linear parameter varying battery model identification using subspace methods, Journal of Power Sources 196 (5) (2011) 2913–2923.

[8] X. Hua, C. Zhang, G. Offer, Finding a better fit for lithium ion batteries: A simple, novel, load dependent, modified equivalent circuit model and parameterization method, Journal of Power Sources 484 (2021) 229117.

[9] M. Lenz, D. Jöst, F. Thiel, S. Pischinger, D. U. Sauer, Identification of load dependent cell voltage model parameters from sparse input data using the mixed integer distributed ant colony optimization solver, Journal of Power Sources 437 (2019) 226880.

[10] N. Tian, Y. Wang, J. Chen, H. Fang, One-shot parameter identification of the Thevenin's model for batteries: Methods and validation, Journal of Energy Storage 29 (2020) 101282.

[11] C. Zhang, W. Allafi, Q. Dinh, P. Ascencio, J. Marco, Online estimation of battery equivalent circuit model parameters and state of charge using decoupled least squares technique, Energy 142 (2018) 678–688.

[12] M. Partovibakhsh, G. Liu, An adaptive unscented Kalman filtering approach for online estimation of model parameters and state-of-charge of lithium-ion batteries for autonomous mobile robots, IEEE Transactions on Control Systems Technology 23 (1) (2014) 357–363.

[13] B. Fridholm, T. Wik, M. Nilsson, Robust recursive impedance estimation for automotive lithium-ion batteries, Journal of Power Sources 304 (2016) 33–41.

[14] C. Antaloae, J. Marco, F. Assadian, A novel method for the parameterization of a li-ion cell model for EV/HEV control applications, IEEE Transactions on Vehicular Technology 61 (9) (2012) 3881–3892.

[15] A. Golabi, N. Meskin, R. Tóth, J. Mohammadpour, A Bayesian approach for LPV model identification and its application to complex processes, IEEE Transactions on Control Systems Technology 25 (6) (2017) 2160–2167.

[16] E. Snelson, Z. Ghahramani, Sparse Gaussian processes using pseudo-inputs, Advances in Neural Information Processing Systems 18 (2005) 1257–1264.

[17] K. Fan, Y. Wan, State-of-charge dependent battery model identification using Gaussian process regression, IFAC-PapersOnLine 54 (3) (2021) 647–652.

[18] Z. Deng, X. Hu, X. Lin, Y. Che, L. Xu, W. Guo, Data-driven state of charge estimation for lithium-ion battery packs based on Gaussian process regression, Energy 205 (2020) 118000.

[19] X. Li, Z. Wang, J. Yan, Prognostic health condition for lithium battery using the partial incremental capacity and Gaussian process regression, Journal of power sources 421 (2019) 56–67.

[20] K. Liu, Y. Li, X. Hu, M. Lucu, W. D. Widanage, Gaussian process regression with automatic relevance determination kernel for calendar aging prediction of lithium-ion batteries, IEEE Transactions on Industrial Informatics 16 (6) (2019) 3767–3777.

[21] R. R. Richardson, C. R. Birkl, M. A. Osborne, D. A. Howey, Gaussian process regression for in situ capacity estimation of lithium-ion batteries, IEEE Transactions on Industrial Informatics 15 (1) (2018) 127–138.

[22] A. Fotouhi, D. Auger, K. Propp, S. Longo, Accuracy versus simplicity in online battery model identification, IEEE Transactions on Systems, Man, and Cybernetics: Systems 48 (2) (2016) 195–206.

[23] F. Zheng, Y. Xing, J. Jiang, B. Sun, J. Kim, M. Pecht, Influence of different open circuit voltage tests on state of charge online estimation for lithium-ion batteries, Applied Energy 183 (2016) 513–525.

[24] B. Xia, X. Zhao, R. D. Callafon, H. Garnier, T. Nguyen, C. Mi, Accurate lithium-ion battery parameter estimation with continuous-time system identification methods, Applied Energy 179 (2016) 426–436.

[25] T. Feng, L. Yang, X. Zhao, H. Zhang, J. Qiang, Online identification of lithium-ion battery parameters based on an improved equivalent-circuit model and its implementation on battery state-of-power prediction, Journal of Power Sources 281 (2015) 192–203.

[26] C. K. Williams, C. E. Rasmussen, Gaussian Processes for Machine Learning, MIT press Cambridge, MA, 2006.

[27] J. Quiñonero-Candela, C. E. Rasmussen, A unifying view of sparse approximate Gaussian process regression, Journal of Machine Learning Research 6 (Dec) (2005) 1939–1959.

[28] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical recipes 3rd edition: The art of scientific computing, Cambridge university press, 2007.

[29] E. Kılıç, P. Stanica, The inverse of banded matrices, Journal of Computational and Applied Mathematics 237 (1) (2013) 126–135.