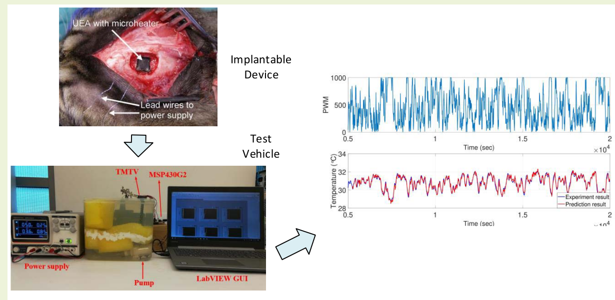
Online Thermal Effect Modeling and Prediction of Implantable Devices

Ruizhi Chai, *Member, IEEE*, and Ying Zhang, *Senior Member, IEEE*

***Abstract*—The overheating caused by the operation of implantable device can cause damage to the surrounding tissue. In applications like neural prosthesis, 1 °C of temperature increase could lead to irreversible damage to the subject. Predicting the overheating effect is therefore critical to maintain safe operation. This work proposes a Bayesian recursive multi-step prediction method for implantable device to predict the overheating effect. The method proposed in this article achieves accurate prediction within a horizon with low complexity by model updating that iteratively minimizes a function of the *j*-step-ahead prediction error. At each time instant, the new available input output data are stored in a First In First Out (FIFO) queue of fixed length, and the model parameters are updated by iteratively minimizing the *j*-step-ahead prediction error of the new data. Moreover, the regularization methods are introduced to improve the prediction performance by taking the Bayesian interpretation of the parameters into consideration. Monte Carlo simulation studies indicate that the developed method is able to estimate the fundamental dynamics of the system when the prediction model is underparametered, and is robust to measurement noise. For time varying systems, the developed method can capture the system dynamics during the system variation. The proposed method is demonstrated via an in-vitro test vehicle, which shows that the temperature increase can be predicted with high accuracy and low complexity.**



Thermal Effect Prediction of Implantable Device

***Index Terms*— Prediction error minimization, multi-step prediction, thermal effect, implantable devices.**

I. Introduction

WITH implantable device becoming more and more powerful, the temperature increase caused by its operation has drawn growing concern. Within human body, even a few degree Celsius above the normal body temperature could cause detrimental effect to the subject. It is reported that a patient with an implanted deep brain stimulator (DBS) suffered significant brain damage after diathermy treatment, and subsequently died [1], [2]. Postmortem examinations indicated that the tissue near the lead electrodes of the DBS deteriorated due to overheating. Researchers have shown that a temperature increase greater than 1 °C could have long-term damage to the brain tissue [3]. It is considered safer to maintain a maximum temperature increase of 1 °C for brain implants [4]. For visual implants, where the stimulating circuitry is in close contact with the retina, even small temperature increase could have potentially deleterious effects on retinal integrity [5].

In many practical applications, the implantable devices only have very limited power consumption and comunication with external world, or stay in sleep mode most of the time. The thermal effect is not a significant problem in these cases. Blood perfusion in the human body often helps to disperse the heat accumulated around the implantable device. The thermal safety can be guaranteed by limiting the functionality of the implantable device during the design phase.

For applicaitons like neural prosthesis, where implantable devices need to constantly stimulate the body and its neural tissues with a large number of electrodes and are in continuous communication with external devices, the heat accumulated around the implantable device can be dangerous to the subject body [6]. With the incorporation of high-density, functional electronic components and as the number of stimulaiton channels increases, this problem become more and more significant. For these applications, an accurate real-time temperature model is crucial to guarantee the safe operation of the implantable device.

In the neural prothesis applications, however, it can be difficult to obtain a parametric model for the thermal effect due to the complex internal structure of the human brain and the dimension of bio-implants. A heat transfer model based on Pennes bioheat equation [7] can capture the temperature increase of the surrounding tissue, but is infeasible for real time applications [8]. Numerical methods [9]–[13] that have been developed to solve the Pennes bioheat equation all rely on sampling the temperature value in the simulatio domain as they evlove in time. The time complexity and space complexity make them unsuitable for real-time applications.

Manuscript received August 8, 2020; revised September 10, 2020; accepted September 13, 2020. This work was supported in part by the National Science Foundation under Grant ECCS-1711447. The associate editor coordinating the review of this article and approving it for publication was Prof. Subhas C. Mukhopadhyay. *(Corresponding author: Ying Zhang.)*

The authors are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: rchai3@gatech.edu; yzhang@gatech.edu).

Digital Object Identifier 10.1109/JSEN.2020.3025874

Researchers in the field have been evaluating the thermal effect during the design phase [14], but real-time thermal effect prediciton is critical to achieve the full potential of the implantable device.

In our previous work [8], the thermal modeling problem has been studied based on the system identificaiton methods [15], [16], or more specifically the mutli-step prediction error method [17]. However, the performance of the proposed method gradually degrade through time with the thermal model deviates from the actual system. Recently, Bayesian estimation based techniques has also been introduced to the system identification problem [18]–[22]. In particular, prior information is introduced to the identification process by designing a covariance, which is also known as kernel in the machine learning literature.

In this article, based on the previous results [23], we study the online prediction of thermal effect of an implantable device, with a focus on neural prosthesis. The Bayesian estimation techniques is incorporated to generate a more robust and accurate thermal effect prediction model that is suitable for real-time applications. More specifically, a Bayesian recursive multi-step prediction error method (Bayesian RMSPEM) is developed based on iteratively minimizing a function of the j-step ahead prediction error. The data used for the estimation is stored in a FIFO queue with fixed length. At each time instant, when the new output enters the queue, the estimator uses the data within the j prior steps to calculate the prediction error of the new data and iteratively update the model parameter for $j = 1, \dots, k$ in the prediction horizon. Output Error (OE) type model is employed as the predictor, as it is able to capture the low-frequency fundamental dynamics even when the predictor order is lower than the underlying system. Moreover, with the Bayesian estimation techniques, the proposed method is able to capture the system dynamics when the predictor order is greater than that of the system. By employing the forgetting factor and taking advantage of the iterative updating procedure, time-varying thermal dynamics can be modeled with the developed method regardless of noises. The developed method is shown to have low complexity and is therefore appropriate for realtime implementation on implantable devices with limited computational power.

The remainder part of this article is organized as follows. Following Section I, Section II introduces the system model used in the paper. The regularized batch preprocessing procedure is described in Section III. The proposed Bayesian recursive identification method is presented in Section IV. Section V extends the algorithm with forgetting factor and discusses about the computational complexity of the algorithm. Simulation investigations are presented in Section VI and the UEA thermal effect prediction is presented in Section VII. At last, the paper is concluded in Section VIII.

II. System Model and Identification Criterion

Let's assume the underlying thermal dynamics $\mathcal{F}$ can be defined as a general Single Input Single Output (SISO) linear discrete-time equation of the type:

$$y(t) = G(q, \theta^o)u(t) + H(q, \theta^o)e(t), \tag{1}$$

where the true parameters of the thermal dynamics are denoted by θ^o. $G(q, \theta^o)$ represents the transfer function from input (power consumption) to output (temperature in the surrounding tissue), and $H(q, \theta^o)$ is the transfer function from a white noise source e to output additive disturbances. Both $G(q, \theta^o)$ and $H(q, \theta^o)$ are asymptotically stable transfer functions. q denotes the shift operator $qy(t) = y(t + 1)$.

Depending on how to parameterize G and H, many model structures have been proposed, such as the autoregressive with exogenous terms (ARX) model, the autoregressive-moving average with exogenous terms (ARMAX) model, and the Box-Jenkins model. In this article, the estimation model with the output error (OE) structure are considered, since the thermal dynamics around the implantable device can be highly complicated and time varying. With a OE model structure, low frequency dynamics can still be captured even when the model order is lower than that of the thermal dynamics.

Let $G(q, \theta^o)$ be defined as

$$G(q, \theta^o) = \frac{N_G(q)}{D_G(q)} = \frac{b_1^o q^{-1} + \cdots + b_{n_b^o}^o q^{-n_b^o}}{1 + a_1^o q^{-1} + \cdots + a_{n_a^o}^o q^{-n_a^o}}, \tag{2}$$

and

$$\theta^o(\mathbf{a}^o, \mathbf{b}^o) = \begin{bmatrix} \mathbf{a}^o \\ \mathbf{b}^o \end{bmatrix} = \begin{bmatrix} -a_{n_a^o}^o \\ \vdots \\ -a_1^o \\ b_1^o \\ \vdots \\ b_{n_b^o}^o \end{bmatrix} \tag{3}$$

represents the system parameter in G. The numerator $N_G(z)$ and the denominator $D_G(z)$ are assumed to be coprime.

The OE prediction model can be represented as

$$\hat{y}(t|\theta) = \hat{G}(q, \theta)u(t), \tag{4}$$

which doesn't explicitly model the noise. In (4),

$$\hat{G}(q, \theta) = \frac{\hat{N}_G(q)}{\hat{D}_G(q)} = \frac{b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b}}{1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}}, \tag{5}$$

and

$$\theta = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} -a_{n_a} \\ \vdots \\ -a_1 \\ b_1 \\ \vdots \\ b_{n_b} \end{bmatrix}. \tag{6}$$

In general, $n_a \neq n_a^o$ and $n_b \neq n_b^o$.

Suppose the input output data of $\mathcal{F}$ are recorded in sequential in the time domain as

$$\mathcal{L} = \{u(1), y(1), \dots, u(N), y(N)\}. \tag{7}$$

Then the criterion of fit can be defined for the j-step-ahead prediction as

$$J_P^N(j) = \frac{1}{N - n_b - j + 1} \sum_{t=n_b+j}^{N} [y(t) - \hat{y}(t|t-j)]^2 + \gamma \theta^T P^{-1} \theta. \tag{8}$$

In which, $\hat{y}(t|t-j)$ denotes the prediction of $y(t)$ given the output data up to $t-j$ and input data up to t. Prior information of the system parameters are taken into account by introducing a regularization term $\theta^T P^{-1}\theta$ into (8) with P^{-1} represents the covariance information of the parameter prior distribution. γ denotes the relative weight of the regularization term.

The multi-step prediction error criterion is defined as the average of the j-step-ahead criterion with $j = 1, \dots, k$:

$$J_{MP}^N = \frac{1}{k}\sum_{j=1}^{k} J_P^N(j). \tag{9}$$

Gianluigi Pillonetto *et al.* [18], [24] summarized the regularization options for system identification, such as Diagonal/Correlated (DC) kernel, Tuned/Correlated (TC) kernel and Stable Spline (SS) kernel, whose frequency properties are summarized in [25]. The kernels can also be derived for system identification purposes [26]. The regularized cost function can then be solved by the regularized least square estimation.

III. Regularized Batch Pre-Processing

To determine the hyperparameters of the regularization and choose a good starting point for searching the minimum of the cost function presented in the previous section, a batch of data is used to initialize the Bayesian RMSPEM algorithm. This batch of data is called the pre-processing data and represented as $\mathcal{L}^o = \{u^o(1), y^o(1), \dots, u^o(N_0), y^o(N_0)\}$. In practice, this procedure helps to generate a reliable model estimation in the initial phase. In this section, we present how the regularization technique can be used for the multi-step prediction under the batch setting.

First, (4) can be converted into the linear regression form as

$$\hat{y}(t|\theta) = \phi(t)^T\theta. \tag{10}$$

Given the data set $\mathcal{L}^o$, the one-step predictions can be concatenated into the vector form

$$Y = \Phi\theta, \tag{11}$$

in which

$$Y = \begin{bmatrix} \hat{y}(1|\theta) \\ \vdots \\ \hat{y}(N_0|\theta) \end{bmatrix}, \tag{12}$$

and

$$\Phi = \begin{bmatrix} \phi(1)^T \\ \vdots \\ \phi(N_0)^T \end{bmatrix}. \tag{13}$$

Let the regularization matrix P be parameterized in terms of the hyperparameter η. The hyperparameter can be determined through

$$\hat{\eta} = \arg\min_{\eta}\; Y^T Z(\eta)^{-1} Y + \log|Z(\eta)|, \tag{14}$$

$$Z(\eta) = \Phi P(\eta)\Phi^T + \gamma^2 I_{N_0}, \tag{15}$$

which represents the maximization of the negative log likelihood function for estimating η from Y.

Let's then derive the multi-step prediction error cost function and its optimization procedure for a batch of data. Let $\hat{y}(t+j|t)$ be the output value predicted by iterating j times the recursive equation of (4). It can be represented as [16], [17]:

$$\hat{y}(t+j|t) = R_j(q)y(t) + E_j(q)\hat{N}_G(q)u(t+j), \tag{16}$$

where $R_j(q)$ and $E_j(q)$ can be calculated as

$$R_j(q) = \mathbf{C}\mathbf{A}^j \begin{bmatrix} q^{-n_a+1} \\ \vdots \\ 1 \end{bmatrix}, \tag{17}$$

and

$$E_j(q) = \mathbf{C}\sum_{i=0}^{j-1}\mathbf{A}^i\mathbf{B}q^{-i}. \tag{18}$$

In which,

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \dots & 1 \\ -a_{n_a} & -a_{n_a-1} & \dots & -a_1 \end{bmatrix}, \tag{19}$$

$$\mathbf{B} = \mathbf{C}^T, \tag{20}$$

and

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & \dots & 1 \end{bmatrix}. \tag{21}$$

Let

$$Y_t = \begin{bmatrix} y^o(t-n_a+1) \\ \vdots \\ y^o(t) \end{bmatrix}, \tag{22}$$

and

$$U_{t,j} = \begin{bmatrix} u^o(t+j-1) \\ u^o(t+j-2) \\ \vdots \\ u^o(t-n_b+1) \end{bmatrix}. \tag{23}$$

Then define

$$\phi_j(t) = \begin{bmatrix} Y_t \\ U_{t,j} \end{bmatrix}, \tag{24}$$

and

$$\Theta_j(\theta) = \begin{bmatrix} (\mathbf{C}\mathbf{A}^j)^T \\ M_{j,n_b}\mathbf{b} \end{bmatrix}, \tag{25}$$

in which

$$M_{j,n_b} = \begin{bmatrix} \mathbf{CB} & \dots & \mathbf{C}\mathbf{A}^{j-1}\mathbf{B} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \mathbf{CB} & \dots & \mathbf{C}\mathbf{A}^{j-1}\mathbf{B} \end{bmatrix}^T. \tag{26}$$

The j-step-ahead predictor can be reformulated as a linear regression form:

$$\hat{y}(t+j|t) = \phi_j(t)^T\Theta_j(\theta). \tag{27}$$

$\Theta_j(\theta)$ is the j-step-ahead mapping of the predictor parameter θ.

For the batch pre-processing data, define

$$Y_{N_0}^{j} = \begin{bmatrix} y^o(n_b + j) \\ \vdots \\ y^o(N_0) \end{bmatrix}, \tag{28}$$

and

$$\Phi_j = \begin{bmatrix} \phi_j(n_b)^T \\ \vdots \\ \phi_j(N_0 - j)^T \end{bmatrix}. \tag{29}$$

Let $R_j^0 = \Phi_j^T \Phi_j$ and $K_j^0 = \Phi_j^T Y_{N_0}^j$.

The cost function of batch pre-processing is

$$J_{MP}^{N_0}(k) = \frac{1}{k} \sum_{j=1}^{k} J_P^{N_0}(j), \tag{30}$$

in which

$$J_P^{N_0}(j) = \frac{1}{N_0 - n_b - j + 1} \| Y_{N_0}^j - \Phi_j \Theta_j(\theta) \|^2 + \gamma \theta^T P^{-1} \theta. \tag{31}$$

This cost function can be minimized using the standard Newton method. The optimal model parameters can be estimated iteratively as:

$$\theta_{j+1} = \theta_j - \Big(\frac{\partial^2}{\partial \theta^2} J_{MP}^{N_0}(j)\Big|_{\theta=\theta_j}\Big)^{-1} \nabla_\theta J_{MP}^{N_0}(j)\Big|_{\theta=\theta_j}, \quad j = 1, \dots, k-1. \tag{32}$$

The Hessian of $J_{MP}^{N_0}(j)$ can be calculated as:

$$\frac{\partial^2}{\partial \theta^2} J_{MP}^{N_0}(j)\Big|_{\theta=\theta_j} = \frac{1}{j} \sum_{s=1}^{j} \frac{\partial^2}{\partial \theta^2} J_P^{N_0}(s), \tag{33}$$

and

$$\frac{\partial^2}{\partial \theta^2} J_P^{N_0}(s) \approx \frac{2}{N_0 - n_b - s + 1} \nabla_\theta \Theta_s(\theta) R_s^0 \nabla_\theta \Theta_s(\theta)^T + 2\gamma P^{-1}. \tag{34}$$

The gradient of $J_{MP}^{N_0}(j)$ can be expressed as follows:

$$\nabla_\theta J_{MP}^{N_0}(j)\Big|_{\theta=\theta_j} = \frac{1}{j} \sum_{s=1}^{j} \nabla_\theta J_P^{N_0}(s) \tag{35}$$

and

$$\nabla_\theta J_P^{N_0}(s) = \frac{2}{N_0 - n_b - s + 1} \nabla_\theta \Theta_s(\theta)(R_s^0 \Theta_s(\theta) - K_s^0) + 2\gamma P^{-1}\theta. \tag{36}$$

In (34) and (36), Θ_j and its gradient $\nabla_\theta \Theta_j$ can be updated iteratively over j:

$$\Theta_{j+1} = W_j \Theta_j, \tag{37}$$

$$\nabla_\theta \Theta_{j+1} = \nabla_\theta \Theta_j W_j^T + ([\mathbf{C}\ \mathbf{0}_{1,n_b+j-1}]\Theta_j) H_j^T. \tag{38}$$

in which,

$$W_j = \left[\begin{bmatrix} \mathbf{A}^T \\ \mathbf{0}_{j,n_a} \\ \mathbf{bC} \end{bmatrix}, \begin{bmatrix} \mathbf{0}_{n_a,n_b-1+j} \\ I_{n_b-1+j} \\ \mathbf{0}_{1,n_b-1+j} \end{bmatrix} \right], \tag{39}$$

and

$$H_j = \left[\begin{bmatrix} I_{n_a} \\ \mathbf{0}_{n_b+j,n_a} \end{bmatrix}, \begin{bmatrix} \mathbf{0}_{n_a+j,n_b} \\ I_{n_b} \end{bmatrix} \right]. \tag{40}$$

The initial value can be set as $\Theta_1 = \theta$, $\nabla_\theta \Theta_1 = I_{n_a+n_b}$.

During the pre-processing, R_s^0 and K_s^0 are stored for $s = 1, \dots, k$ and used for initialization of the RMSPEM. The calculated θ also provides an initial start to accelerate the convergence of RMSPEM. In practical applications, the model order selection procedure can be incorporated into the pre-processing. The model order that minimizes the cost function (30) can be used for the following Bayesian RMSPEM. This saves the computational cost during the run time. Due to the robustness to model order, many system variations during the run time can still be captured. Moreover, as will be demonstrated below, the computational cost of the Bayesian RMSPEM is proportional to the model order. In many cases, a low model order can be generated in the pre-processing procedure, which helps to reduce the complexity of the proposed algorithm.

IV. Bayesian Recursive MSPEM

After the regularization hyperparameters and the RMSPEM initial parameters R_s^0, K_s^0 and θ_0 are determined through the pre-processing, the Bayesian RMSPEM algorithm updates the parameter estimation at each time instant when there is new data available. Assume the test data is represented as $\mathcal{L} = \{u(1), y(1), \dots, u(N), y(N), \dots\}$, the update procedure of the proposed algorithm will be demonstrated by assuming that the current time instant is N, and the new available data are the input $u(N)$ and output $y(N)$. The model parameters are then updated iteratively using the Newton method based on the prediction error of $y(N)$.

As shown in the batch pre-processing, to determine the Hessian matrix and the gradient of the cost function, the R matrix and K matrix must be calculated for $j = 1, \dots, k$. Let's define

$$R_j^N = R_j^0 + \Phi_j^T \Phi_j \tag{41}$$

and

$$K_j^N = K_j^0 + Y_N^{j\,T} \Phi_j. \tag{42}$$

At time N, R_j^N and K_j^N must be stored for $j = 1, \dots, k$.

R_j^N can be represented as

$$R_j^N = R_j^0 + \sum_{s=n_b}^{N-j} \phi_j(s)\phi_j(s)^T, \tag{43}$$

which can then be calculated recursively as

$$R_j^N = R_j^{N-1} + \phi_j(N-j)\phi_j(N-j)^T. \tag{44}$$

On the right hand side of (44), R_j^{N-1} can be determined with all the data up to $(N-1)$th discrete time instant, and $\phi_j(N-j)$ contains all the data up to Nth discrete time instant.

Similarly, K_j^N can be represented as

$$K_j^N = K_j^0 + \sum_{s=n_b}^{N-j} \phi_j(s) y(s+j), \tag{45}$$

and it can be calculated recursively as

$$K_j^N = K_j^{N-1} + \phi_j(N-j)y(N). \tag{46}$$

In which, K_j^{N-1} can be determined with all the data up to $(N-1)$th discrete time instant. $y(N)$ is the output measurement available at Nth discrete time instant. $\phi_j(N-j)$ requires the data up to Nth discrete time instant.

Furthermore, given the saved R_j^{N-1} and K_j^{N-1}, it only requires the input and output values within a finite time window to calculate R_j^N and K_j^N. More specifically, it requires the output measurements from time $N-k-n_a+1$ to time $N-1$ and input values from time $N-k-n_b+1$ to time $N-1$ to calculate $\phi_j(N-j)$, $j=1,\ldots,k$. To calculate K_j^N, the newest output measurement $y(N)$ is also needed. In practice, all the output measurements and input values necessary to update the R matrix and K matrix can be saved in a FIFO queue as

$$Input\ Queue: \begin{bmatrix} u(N-k-n_b+1) \\ \vdots \\ u(N-1) \end{bmatrix},$$

$$Output\ Queue: \begin{bmatrix} y(N-k-n_a+1) \\ \vdots \\ y(N) \end{bmatrix}. \tag{47}$$

For each $j=1,\ldots,k$, the corresponding R matrix and K matrix have to be saved separately. Each time, when the new input and output values are recorded, R matrix and K matrix are to be updated using (44) and (46) for each j.

With the recursive updates of (44) and (46), the estimation of $\frac{\partial^2}{\partial\theta^2}J_P^N(s)$ and $\nabla_\theta J_P^N(s)$ can be formulated as:

$$\frac{\partial^2}{\partial\theta^2}J_P^N(s) \approx \frac{2}{N-n_b-s+1}\nabla_\theta\Theta_s(\theta)R_s^N\nabla_\theta\Theta_s(\theta)^T + 2\gamma P^{-1}, \tag{48}$$

$$\nabla_\theta J_P^N(s) = \frac{2}{N-n_b-s+1}\nabla_\theta\Theta_s(\theta)(R_s^N\Theta_s(\theta)-K_s^N) + 2\gamma P^{-1}\theta. \tag{49}$$

Again, Θ_j and its gradient $\nabla_\theta\Theta_j$ can be updated using (39) and (40). Let

$$Q_N^j = \frac{1}{j}\sum_{s=1}^{j}\nabla_\theta\Theta_s(\theta)R_s^N\nabla_\theta\Theta_s(\theta)^T + 2\gamma P^{-1}, \tag{50}$$

$$-P_N^j = \frac{1}{j}\sum_{s=1}^{j}\nabla_\theta\Theta_s(\theta)(R_s^N\Theta_s(\theta)-K_s^N) + 2\gamma P^{-1}\theta. \tag{51}$$

Then Q_N^j and P_N^j can be calculated iteratively for $j=1,\ldots,k$ as:

$$Q_N^j = \frac{j-1}{j}Q_N^{j-1} + \frac{1}{j}\nabla_\theta\Theta_j(\theta)R_N^j\nabla_\theta\Theta_j(\theta)^T + \frac{2}{j}\gamma P^{-1}, \tag{52}$$

$$P_N^j = \frac{j-1}{j}P_N^{j-1} + \frac{1}{j}\nabla_\theta\Theta_j(\theta)(K_s^N - R_s^N\Theta_s(\theta)) - \frac{2}{j}\gamma P^{-1}\theta. \tag{53}$$

The parameter update procedure can be represented as

$$\theta_{j+1} = \theta_j + \mu(Q_N^j)^{-1}P_N^j. \tag{54}$$

Algorithm 1 Bayesian RMSPEM Method

Require: Previously obtained parameter θ_{pre}, Input Queue U, Output Queue Y, R, K

$\theta = \theta_{pre}$;
$\Theta = \theta_{pre}$;
$\nabla_\theta\Theta = I_{n_a+n_b}$;
Set μ;
$Q_N = 0$;
$P_N = 0$;

for `j=1:k` **do**
- Initialize ϕ_j;
- $R[j] = R[j] + \phi_j\phi_j^T$;
- $K[j] = K[j] + \phi_j Y[k]$;
- $Q_N = \frac{j-1}{j}Q_N + \frac{1}{j}\nabla_\theta\Theta R[j]\nabla_\theta\Theta^T + \frac{2}{j}\gamma P^{-1}$;
- $P_N = \frac{j-1}{j}P_N + \frac{1}{j}\nabla_\theta\Theta(K[j]-R[j]\Theta) - \frac{2}{j}\gamma P^{-1}\theta$;
- $\theta = \theta + \mu Q_N^{-1}P_N$;
- Calculate W_j, H_j;
- $\nabla_\theta\Theta = \nabla_\theta\Theta W_j^T + ([C, \mathbf{0}_{1\times n_b+j-1}]\Theta)H_j^T$;
- $\Theta = W_j\Theta$;

end for

The computation procedure of the proposed method is summarized in Algorithm 1.

V. Algorithm Extension and Analysis

A. Forgetting Factor

For identification of time-varying systems, the aforementioned method can be modified so that past data become less relevant for the current estimation. In this subsection, we propose a routine that use the forgetting factor to weight the past data.

Following a classical practice in parametric time-varying system identification [27], we introduce a forgetting factor $\lambda \in (0,1]$ into the update procedure in order to base the estimation mainly on the more recent data. Specifically, we modify the j-step-ahead cost function to be

$$J_P^N(j) = \frac{1}{N-n_b-j+1}\|\Lambda_N^j(Y_N^j - \Phi_j\Theta_j(\theta))\|^2 + \gamma\theta^T P^{-1}\theta, \tag{55}$$

in which,

$$\Lambda_N^j = \begin{bmatrix} \lambda^{\frac{N-n_b-j}{2}} & & & \\ & \lambda^{\frac{N-n_b-j-1}{2}} & & \\ & & \ddots & \\ & & & \lambda^0 \end{bmatrix} \tag{56}$$

and λ can often chose from 0.98 to 0.995. By using this forgetting factor, measurements older than $T_0 = \frac{1}{1-\lambda}$ samples are included in the criterion with a weight that is $e^{-1} \approx 36\%$ of that of the most recent measurement.

With the modified cost function, the algorithm update procedure remains the same while the update of data matrix R_s^N and K_s^N in Hessian matrix (52) and Gradient (53) can be modified as

$$\bar{R}_j^N = \lambda\bar{R}_j^{N-1} + \phi_j(N-j)\phi_j(N-j)^T, \tag{57}$$

$$\bar{K}_j^N = \lambda\bar{K}_j^{N-1} + \phi_j(N-j)y(N). \tag{58}$$

B. Performance Analysis

The time complexity of Algorithm 1 can be analyzed in terms of the number of flops (floating-point operation). For each $j = 1, \ldots, k$, the calculation requires order $(n_a + n_b)^3$ and j^2 flops. Therefore, the entire algorithm for $j = 1, \ldots, k$ requires order $(n_a + n_b)^3$ and k^3. More specifically, the time complexity is in an order of magnitude similar to square matrix multiplication. In real applications, benefiting from the property of robustness to model orders, the computational cost can be reduced with a lower model order and shorter prediction range.

In terms of the space complexity, besides the input queue of length $k + n_b - 1$ and the output queue of length $k + n_a$, the algorithm needs to store $R[j]$ and $K[j]$ for each $j = 1, \ldots, k$. $R[j]$ is a matrix in $\mathcal{R}^{(n_a+n_b+j-1)\times(n_a+n_b+j-1)}$ and $K[j]$ is a vector in $\mathcal{R}^{n_a+n_b+j-1}$. The previous parameter estimation $\theta \in \mathcal{R}^{n_a+n_b+j-1}$ also needs to be stored.

Online system identification methods like RPEM (recursive prediction error method) are often less computationally demanding. However it cannot guarantee long term prediction performance, especially in the case where complex noise models are involved. Therefore, it is not very suitable for practical applications like adaptive MPC, where the prediction accuracy within a certain horizon is crucial to the performance of the closed-loop control.

The method proposed in [19] has higher computational cost compared to the method proposed in this article, as it maintains a high order model and rely on hyper-parameter updating at each execution to select the appropriate model. The hyper-parameter calculation process is both computationally demanding and difficult to implement for embedded systems, like what is used in the bioimplants. Moreover, the method in [19] requires a sampling rate that is several times of the model updating rate, which is also a challenge for many applications.

Compared to the online identification techniques likes RPEM, the developed algorithm falls between the online identification and the batch identification. It uses the pre-processing to determine the kernel hyperparameters and initialize RMSPEM. In applications where the prediction accuracy is crucial to the performance, this prevents the bad performance in the initial phase of the algorithm. During the operation, the developed method has the advantage of low complexity and robustness to different noise models. Even when the predetermined model structure is underparameterized, the developed method still captures the low-frequency fundamental dynamics. With the forgetting factor incorporated, the developed method is able to track a time varying system and provide a k-step-ahead prediction based on the history information within the k prior steps.

C. Practical Application

The method presented above assume the input to be the power consumption of implantable device. However that can be hard to estimate for a practical system. Instead, we can choose the input of the model to be the controllable system operating status, and the relationship between the input and output of the model can be learned online during the operation.

Due to the small size of an implantable device, the temperature measured by the temperature sensor can be used to approximate the hot spot temperature in most of the cases. In those cases where the temperature sensor is placed far away from the hot spot of an implantable device, we can evaluate the relationship between the measured temperature and the hotspot temperature during the preprocessing phase, then choose the temperature threshold of thermal management more conservatively, so that the actual hot spot will not overheat. Compared to the state-of-the-art approach [14] that limits the functionality of the implantable device by considering the worst case scenario during the design phase, the proposed method can still achieve better overall performance while maintaining safe operation.

VI. Simulation Investigation

In this section, the properties of the developed method are demonstrated with three simulation studies. The first simulation study is a Monte Carlo test with underparameterized prediction models, wherein the order of the prediction model is lower than that of the data generation system. The second simulation study is a Monte Carlo test that features different noise models. The third simulation study demonstrates the performance of the developed method with a linear time varying system.

In these simulation studies, the system generates two kinds of data sets. The first type is the pre-processing data set $\mathcal{L}^o = \{u^o(1), y^o(1), \ldots, u^o(N_0), y^o(N_0)\}$. The second type is the test data set $\mathcal{L} = \{u(1), y(1), \ldots, u(N), y(N)\}$.

The benchmark methods used for comparison are the commonly used online system identification methods, such as Recursive ARX and Recursive OE [15], [16], which are comparable to the proposed method in terms of computational complexity. More specifically, the methods can be implemented on an embedded platform for real time applications and the model is updated at every time step when the new measurements become available. The online Bayesian system identification techniques mentioned in [19], [20] have higher computational cost, as it maintains a high order model and rely on hyper-parameter updating at each execution to select the appropriate model. The hyper-parameter calculation process is both computationally demanding and difficult to implement for embedded systems. Moreover, the method in [19] requires a sampling rate that is several times of the model updating rate, which is also a challenge for many applications.

A. Underparameterized Model

We consider Monte Carlo study of 100 runs regarding identification of discrete-time OE models (4). At each run, a different 30th order transfer function is generated using the procedure described in [18]. A second order input filter is also generated using the similar procedure.

The input in the pre-processing data set $\mathcal{L}^o$ is the realization from white Gaussian noise of unit variance filtered by the input filter. The delay of the input is equal to 1. Starting from zero initial conditions, 1000 input-output data are collected with the output corrupted by an additive white Gaussian noise. The signal-to-noise ratio (SNR) is randomly chosen with in

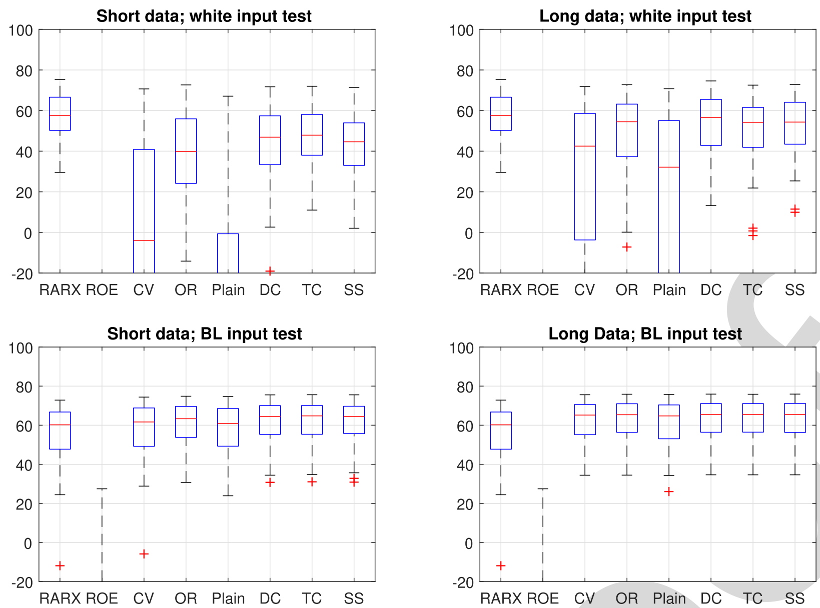


Fig. 1. Identification of discrete-time OE-models.

[1, 10] at every run. In the left two test cases of Figure 1, the preprocessing data set contains the first 150 input-output data while all the 1000 data are used in the right two cases.

Two types of test data sets are generated at every run. The first one contains the white noise corrupted output obtained using a unit variance white Gaussian noise as input. The second one is obtained with a test input generated using the same procedure as in the pre-processing data.

The performance measure (59) as in [18], which represents the variance of the prediction model, is adopted in this article to compare different estimated models. The prediction horizon is chosen to be 20 steps.

$$\mathcal{F}_k(\hat{\theta}) = 100\left(1 - \sqrt{\frac{\sum_{t=k+1}^{N}(y(t) - \hat{y}(t|t-k))^2}{\sum_{t=k+1}^{N}(y(t) - \bar{y})}}\right). \quad (59)$$

The following 7 estimation methods are implemented for comparison:

1) *RecursiveARX*: It implements the recursive PEM approach with ARX model of 8th order. The estimated model is used to predict the output of 20 steps ahead. The estimator is implemented with the *rarx* Matlab routine.
2) *RecursiveOE*: The recursive OE estimator implements the OE model of 20th order and it predicts the output of 20 steps ahead according to the new available data.
3) *RMSPEM+CV*: The RMSPEM algorithm with model order selected via cross validation (CV). Specifically, the pre-processing data are split into two parts $\mathcal{L}_a^o$ and $\mathcal{L}_b^o$, containing the first and last $\frac{N}{2}$ input-output pairs in $\mathcal{L}^o$ respectively. The candidate models have the structure that the polynomials B and F have the same order which varies between 1 and 30. For OE models with different orders, the model parameters are obtained by the batch pre-processing with the estimation data $\mathcal{L}_a^o$. Then the prediction errors are computed for the validation data $\mathcal{L}_b^o$. The model order that maximizing the prediction performance is selected and the final model parameter estimation is calculated with batch pre-processing for the complete data set $\mathcal{L}^o$.
4) *RMSPEM+Or*: The RMSPEM algorithm with an oracle (Or). In particular, for different model orders between 1 and 30, we use the batch pre-procesing to calculate the model parameters with $\mathcal{L}^o$. Then the oracle chooses the model structure that maximizes the fit on the test data. It represents a case that is impractical in general but provides a reference for performance evaluation.
5) *RMSPEM*: The RMSPEM algorithm that uses the OE model of 20th order.
6) *RMSPEM+DC,TC,SS*: The Bayesian RMSPEM algorithm equipped with DC, TC, and SS respectively. The employed model is 20th order. During the pre-processing, the kernel hyperparameters are estimated by solving the marginal likelihood optimization.

Fig. 1 shows the boxplots of the 100 performance measures calculated in the Monte-Carlo study. The left panels are the results that use only the first 150 input output data for preprocessing, and the right panels are the results that use full 1000 input output data during preprocessing. The top panels show the performance measures with the white input signal and the bottom panels are the performance with the filtered input signal like in the pre-processing data set. The vertical axis represent the performance measure for each estimator.

In all the four simulation cases, the *Recursive ARX* method achieves good performance, but is not as good when the input is filtered.The PEM method can't guarantee the k-step-ahead prediction accuracy. The *RMSPEM+CV* approach has good prediction performance for the case with filtered input signal, but for white input signal the performance is unacceptable, especially when there is less pre-processing data. The *RMSPEM+Or* represents the ideal case where the test data is available for determining the model structure. It is shown that *RMSPEM+Or* achieves good prediction performance in

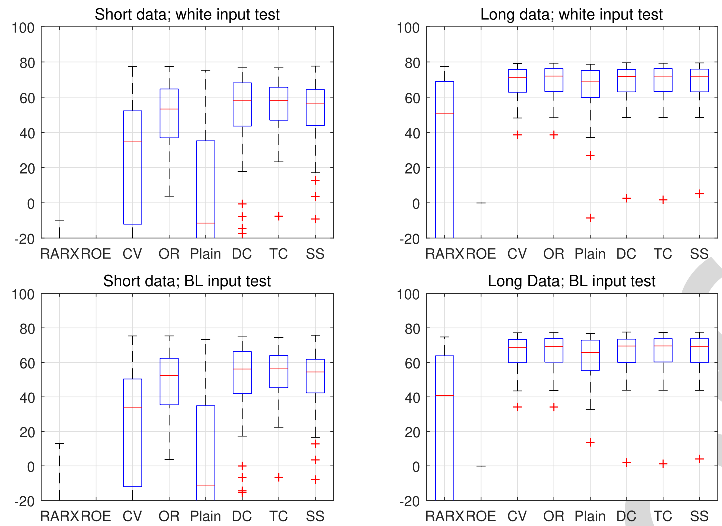


Fig. 2. Identification of discrete-time Box Jenkins models.

all of the four scenarios and is therefore used as reference. The *RMSPEM* algorithm without any regulator is also implemented. When the input signal is white noise, the prediction performance is significantly inferior compared to other estimators. With the incorporated kernels, the Bayesian RMSPEM achieves satisfactory prediction for 20 steps ahead. In the white noise input case, the Bayesian RMSPEM even outperforms the *RMSPEM+Or*.

Additionally, it is obvious that if the data used for pre-processing is similar to the test data, the prediction performance is generally better. This is because the initial estimate obtained through the preprocessing is more likely to be in the neighborhood of the "good" estimate. More preprocessing data helps to improve the prediction performance, but in the Bayesian RMSPEM case the improvement is limited. Therefore, it is shown that the developed Bayesian RMSPEM method is very robust to the pre-processing data.

B. Box-Jenkins System

Let's consider a Box-Jenkins type data-generation system, in which $G(q) = \frac{B(q)}{F(q)}$ and $H(q) = \frac{C(q)}{D(q)}$ are 30th order transfer functions generated using the procedure described in the previous section. The SNR is randomly chosen from 1 to 10. The system is excited by two types of input signal. The first type of input signal is a white Gaussian noise with unit variance. The second type is the realization from white Gaussian noise filtered by a second order filter. The delay of the input is equal to 1. Starting from zero initial conditions, 1000 input-output data are collected.

Two sets of pre-processing data are generated. Both contains the first 200 input output data of the system excited with filtered white Gaussian noise. The first preprocessing data set is generated using only the system process model $G(q)$. The second preprocessing data set is generated using both the process model $G(q)$ and the noise model $H(q)$ in the Box-Jenkins model. Moreover, two sets of test data are used, which include one having the input with the same characteristic of the pre-processing data and another one that use white Gaussian noise as input.

The following estimators are used in this study:

1) *RecursiveARX*: It implements the recursive PEM approach with ARX model of 30th order. The estimated model is used to predict the output 20 steps ahead.
2) *RecursiveOE*: The recursive OE estimator implements the OE model of 30th order. The prediction horizon is 20 steps.
3) *RMSPEM+CV, RMSPEM+Or, RMSPEM*: These three estimators use the same setup as in the previous simulation study.
4) *RMSPEM+DC,TC,SS*: The Bayesian RMSPEM approach with DC, TC, SS kernels respectively. The kernel hyperparameters and the weight γ are determined using the preprocessing data.

The Monte Carlo study runs 80 tests. The prediction performance are plotted in Figure 2.

It is demonstrated in this study that the proposed method generally has a superior performance over the Recursive ARX method despite the type of kernel used. Moreover, generating pre-processing data with only the process model $G(q)$ gives a better initial estimation, thus the Bayesian RMSPEM better captures the underlying process model in the Monte Carlo tests shown in the left panel of Figure 2.

C. LTV System

In this study, the online parameter identification of linear time varying system with unknown order is investigated. The plant has two operating modes. The first mode has a 30th order transfer function generated randomly using a similar process as described in Section VI-A. The transfer function of the second mode is generated by perturbing the transfer function of the first mode with two additional poles and zeros. Thus both order and parameters of the time varying system change when switching from the first mode to the second mode. 100 data sets consisting of 3000 input-output measurement pairs are generated using Monte Carlo simulations. The system switch at time $k = 1001$. The input of the system is generated as the realization of a unit variance Gaussian signal filtered by a randomly generated second order filter.

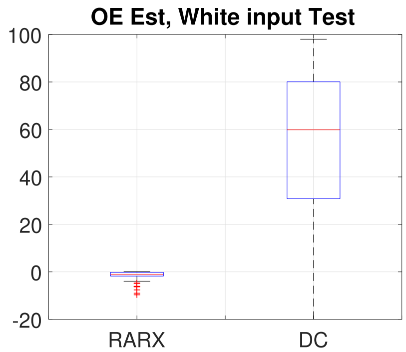


Fig. 3. Identification of linear time varying system.

TABLE I
EXECUTION TIME OF THE TWO ESTIMATORS

	ARX order_select	Bayesian RMSPEM
mean	1.1755 s	0.0506 s
std	11.7552	0.5002

The first 400 input output data are used for pre-processing and the rest of them are used for testing. Two estimators are implemented for comparison. The first one is the recursive ARX method which chooses the model order that minimizes the prediction error at each time instant. The second one is the Bayesian RMSPEM method with DC regularizer whose hyper-parameters are determined during the preprocessing process. Both recursive ARX and the Bayesian RMSPEM use a forgetting factor of 0.98. The prediction window is set to be 20 steps and the order of the prediction model is chosen to be 30.

The prediction performance of the Monte Carlo study is shown in Figure 3 with the y axis representing the performance measure calculated using (59). It is shown that the proposed method is able to track the switch of the time varying system, while the recursive ARX method fails to do so. Moreover, Bayesian RMSPEM is considerably faster than the recursive ARX with order selection. As is shown in Table I,for the 3000 input-output data, the mean cumulative time of the two estimators are 0.0506 seconds and 1.1755 seconds respectively. This result is measured on a computer platform with Intel i7-3770 3.40GHz processor and 12 GB memory.

VII. THERMAL MODELING OF UTAH ELECTRODE ARRAY

In this section, the proposed Bayesian RMSPEM method is employed to predict the thermal effect of Utah electrode array (UEA), which is a 3-D microelectrodes used for deep brain stimulation [28]. The proposed method is suitable for this application is because it provides an accurate temperature prediction with low computational complexity. The performance of the method is demonstrated with a COMSOL simulation and an in-vitro experiment.

A. COMSOL Simulation

In this study, the developed method is used to model the thermal effect of the UEA. A COMSOL Multiphysics model (Figure 4) is implemented for what is demonstrated in [28]. The details of the model is explained in [8]. The UEA is placed on the surface of the brain tissue and a probe is place at $(x, y, z) = (0, 0, 0.042)$ to measure the temperature. The simulation setup is summarized in Table II. The COMSOL

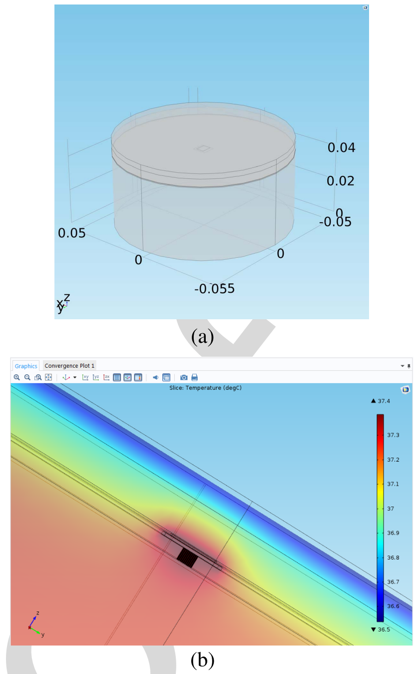


Fig. 4. Illustration of the developed COMSOL model (a) cylinderical human brain model. (b) the UEA model [8].

TABLE II
COMSOL MODEL PARAMETERS

	Thermal Conductivity $(W/(m \cdot K))$	Specific heat capacity $(J/(kg \cdot K))$	Density (kg/m^3)
Brain	0.528	3640	1041
Skull	0.650	1300	1990
Scalp	0.342	3150	1100
Blood	0.530	3840	1060
Silicon	124.0	702	2329

simulation is conducted for 1000 seconds. The power dissipation of the UEA is randomly generated every 10 seconds using a Gaussian distribution, which are then constrained within $[0, 0.02]$ mW. The temperature measurements are recorded and converted into the temperature increase with respect to the body temperature, then stored along with the generated power dissipation at the same time instant.

Bayesian RMSPEM is used to generate a model that predicts the temperature increase of the UEA given its power dissipation. The prediction window of the Bayesian RMSPEM is set to be 10 steps. Each step is 10 seconds. The data of first 200 seconds are used for the pre-processing. The Bayesian RMSPEM updates the parameters of a 5th order prediction model according to the temperature increase obtained by COMSOL. Then the updated model is used to predict the temperature 10 steps later via the j-step-ahead predictor (27). This prediction is then compared with the results obtained from COMSOL. The comparison results are shown in Figure 5.

This comparison result indicates that the thermal dynamics of UEA can be captured by the Bayesian RMSPEM method. The prediction performance is 91.0195. The Mean Square Error of the prediction is about 1.2850×10^{-5} °C.

B. In Vitro Experiment

An in-vitro experiment system [23] is built to emulate the thermal effect of UEA. The system uses a custom designed

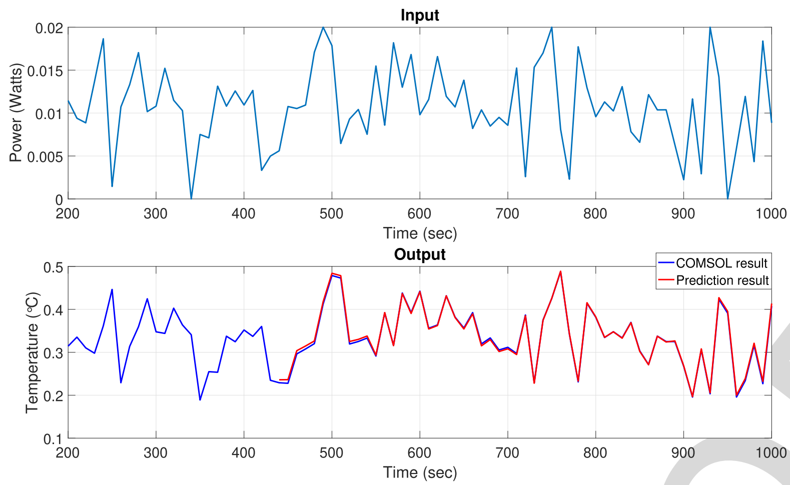


Fig. 5. Modeling the thermal effect of the UEA. Top plot represent the randomly generated power dissipation of the UEA. Bottom plot is the comparison of the simulated UEA temperature and the predicted UEA temperature.

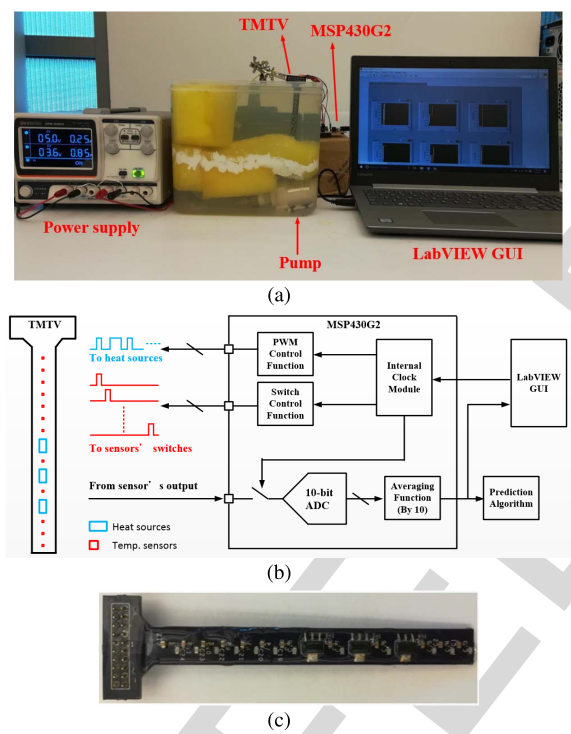


Fig. 6. (a) The developed hardware testing system. (b) Hardware diagram. (c) The developed TMTV system. [23].

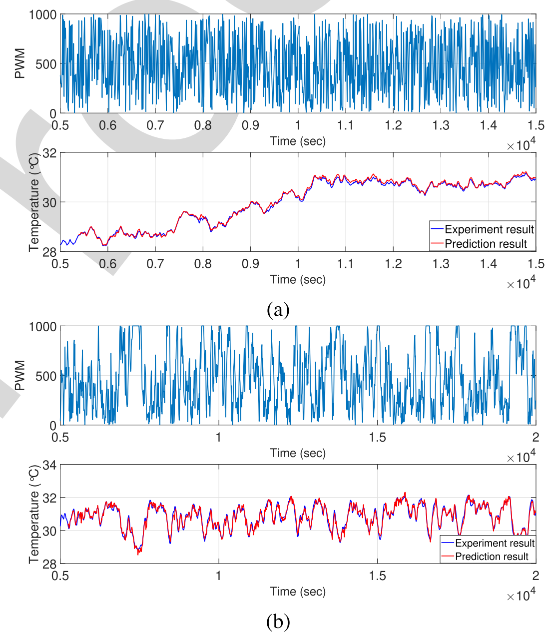


Fig. 7. Experiment results (a) Gaussian input. (b) Filtered Gaussian input.

temperature monitoring and management test vehicle (TMTV) with heat sources and temperature sensors to emulate the implanted electronics and a water circulation system to emulate the blood perfusion effect. A TI MSP430G2 board acts as the middleware between the TMTV and PC. It controls the operation of TMTV via sending PWM signals within the range of [0, 1000] to the heat sources and sends the temperature measurements back to PC, which is then processed by the LabView front end. The PWM signal controls the duty cycle of the heat sources, with 0 being 100% and 1000 being 0%. Figure 6 demonstrates the developed hardware testing system.

We use this testing system to evaluate the prediction accuracy of the simplified thermal model. More specifically, two experiments are conducted. The first experiment randomly generates 2000 PWM signals within the range of [0, 1000] using Gaussian distribution and apply the PWM signals to the heat sources on TMTV with a step size of 10 seconds. The temperature recorded by the onboard sensors are then compared with the temperature predicted by the proposed Bayesian RMSPEM method and the prediction error are used for model updating. The Bayesian RMSPEM method implements a 20th order OE model and predicts the temperature measurements of 10 steps ahead. The results are presented in Figure 7(a). It is demonstrated that the Bayesian RMSPEM accurately predicts the temperature variation despite the varying PWM signal and achieves an overall prediction mean square error of 0.131 °C.

The second experiment generates a random second order low pass filter and applies it to the 2000 random PWM signals. The filtered PWM signal is then applied to the TMTV. This is used to emulate the output of a real thermal management system, where the computed control signal is usually a low frequency signal that depends on various inputs. In this experiment, it is shown in Figure 7(b) that the temperature output can be predicted with a 5th order OE model, which is much simpler than the 20th order OE model used in the first experiment. By taking advantage of this low order model, the computational cost of the proposed method can be greatly reduced. The overall prediction mean square error is about 0.024 °C.

VIII. Conclusion

With the development of more powerful implantable devices, especially the neural prosthesis, the overheating of such devices has become a hidden hazard. Accurate long range prediction of thermal effect is critical to maintain safe operation of implantable device. The existing methods can not guarantee long term performance and achieve full potential of device. A Bayesian multi-step prediction method is developed in this article to generate accurate online thermal effect prediction for implantable devices with limited computational power. The developed method iteratively minimize a function of the j-step-ahead prediction error and recently developed system identification techniques relying on regularization is adopted to improve the prediction performance. Specifically, we assume the online setting in that new data become available at each time instant and then saved into a FIFO queue of fixed length. Based on the input output data saved in the queue, the parameters are updated by iteratively minimizing the j-step-ahead prediction error of the new data. Three simulation studies are presented to demonstrate the performance of the developed method. The first Monte Carlo simulation study shows that when the prediction model is underparameterized the developed method can still capture the low frequency dynamics of the system. The second Monte Carlo simulation study shows that the developed method is robust to different noise models and different input signals. The third Monte Carlo simulation study demonstrates that the developed method is able to capture the dynamics of a time varying system. The application of predicting the thermal effect of UEA is demonstrated via both a COMSOL simulation and an in-vitro experiment, which shows that the developed method can capture the complicated thermal dynamics with great accuracy while only requiring a simple thermal model.

References

[1] P. S. Ruggera, D. M. Witters, G. V. Maltzahn, and H. I. Bassen, "In vitroassessment of tissue heating near metallic medical implants by exposure to pulsed radio frequency diathermy," *Phys. Med. Biol.*, vol. 48, no. 17, pp. 2919–2928, Sep. 2003.

[2] J. G. Nutt, V. C. Anderson, J. H. Peacock, J. P. Hammerstad, and K. J. Burchiel, "DBS and diathermy interaction induces severe CNS damage," *Neurology*, vol. 56, no. 10, pp. 1384–1386, May 2001.

[3] J. C. LaManna, K. A. McCracken, M. Patil, and O. J. Prohaska, "Stimulus-activated changes in brain tissue temperature in the anesthetized rat," *Metabolic Brain Disease*, vol. 4, no. 4, pp. 225–237, Dec. 1989.

[4] T. S. Ibrahim, D. Abraham, and R. L. Rennaker, "Electromagnetic power absorption and temperature changes due to brain machine interface operation," *Ann. Biomed. Eng.*, vol. 35, no. 5, pp. 825–834, Apr. 2007.

[5] N. L. Opie, "Thermal safety of a retinal prosthesis," Ph.D. dissertation, Dept. Elect. Eng., Univ. Melbourne, Melbourne, VIC, Australia, 2011.

[6] N. L. Opie, A. N. Burkitt, H. Meffin, and D. B. Grayden, "Heating of the eye by a retinal prosthesis: Modeling, cadaver and *in vivo* study," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 2, pp. 339–345, Feb. 2012.

[7] H. H. Pennes, "Analysis of tissue and arterial blood temperatures in the resting human forearm," *J. Appl. Physiol.*, vol. 1, no. 2, pp. 93–122, Aug. 1948.

[8] R. Chai and Y. Zhang, "Adaptive thermal management of implantable device," *IEEE Sensors J.*, vol. 19, no. 3, pp. 1176–1185, Feb. 2019.

[9] S. C. DeMarco, G. Lazzi, W. Liu, J. D. Weiland, and M. S. Humayun, "Computed SAR and thermal elevation in a 0.25-mm 2-D model of the human eye and head in response to an implanted retinal stimulator. I. Models and methods," *IEEE Trans. Antennas Propag.*, vol. 51, no. 9, pp. 2274–2285, Sep. 2003.

[10] J.-L. Dillenseger and S. Esneault, "Fast FFT-based bioheat transfer equation computation," *Comput. Biol. Med.*, vol. 40, no. 2, pp. 119–123, Feb. 2010.

[11] G. Carluccio, D. Erricolo, S. Oh, and C. M. Collins, "An approach to rapid calculation of temperature change in tissue using spatial filters to approximate effects of thermal conduction," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 6, pp. 1735–1741, Jun. 2013.

[12] V. Singh *et al.*, "On the thermal elevation of a 60-electrode epiretinal prosthesis for the blind," *IEEE Trans. Biomed. Circuits Syst.*, vol. 2, no. 4, pp. 289–300, Dec. 2008.

[13] S. K. Das, S. T. Clegg, and T. V. Samulski, "Computational techniques for fast hyperthermia temperature optimization," *Med. Phys.*, vol. 26, no. 2, pp. 319–328, Feb. 1999.

[14] C. Serrano-Amenos *et al.*, "Thermal analysis of a skull implant in brain-computer interfaces," in *Proc. 42nd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2020, pp. 3066–3069.

[15] L. Ljung and T. Södrestrm, *Theory and Practice of Recursive Identification*. Cambridge, MA, USA: MIT Press, 1983.

[16] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1999.

[17] M. Farina and L. Piroddi, "Simulation error minimization identification based on multi-stage prediction," *Int. J. Adapt. Control Signal Process.*, vol. 25, no. 5, pp. 389–406, May 2011.

[18] G. Pillonetto, F. Dinuzzo, T. Chen, G. De Nicolao, and L. Ljung, "Kernel methods in system identification, machine learning and function estimation: A survey," *Automatica*, vol. 50, no. 3, pp. 657–682, Mar. 2014.

[19] G. Prando, D. Romeres, and A. Chiuso, "Online identification of time-varying systems: A Bayesian approach," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 3775–3780.

[20] D. Romeres, G. Prando, G. Pillonetto, and A. Chiuso, "On-line Bayesian system identification," in *Proc. Eur. Control Conf. (ECC)*, Jun. 2016, pp. 1359–1364.

[21] D. Romeres, M. Zorzi, R. Camoriano, S. Traversaro, and A. Chiuso, "Derivative-free online learning of inverse dynamics models," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 3, pp. 816–830, May 2020.

[22] D. Romeres, D. K. Jha, A. DallaLibera, B. Yerazunis, and D. Nikovski, "Semiparametrical Gaussian processes learning of forward dynamical models for navigating in a circular maze," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 3195–3202.

[23] R. Chai, Y.-P. Lai, W. Sun, M. Ghovanloo, and Y. Zhang, "Online predictive modeling for the thermal effect of implantable devices," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Oct. 2018, pp. 1–4.

[24] G. Pillonetto, T. Chen, A. Chiuso, G. De Nicolao, and L. Ljung, "Regularized linear system identification using atomic, nuclear and kernel-based norms: The role of the stability constraint," *Automatica*, vol. 69, pp. 137–149, Jul. 2016.

[25] M. Zorzi and A. Chiuso, "The harmonic analysis of kernel functions," *Automatica*, vol. 94, pp. 125–137, Aug. 2018.

[26] M. Zorzi and A. Chiuso, "Sparse plus low rank network identification: A nonparametric approach," *Automatica*, vol. 76, pp. 355–366, Feb. 2017.

[27] P. C. Young, *Recursive Estimation Time-Series Analysis: Introduction*. Cham, Switzerland: Springer, 2012.

[28] S. Kim, P. Tathireddy, R. A. Normann, and F. Solzbacher, "Thermal impact of an active 3-D microelectrode array implanted in the brain," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 15, no. 4, pp. 493–501, Dec. 2007.