2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

An Efficient Implementation for Kernel-based Regularized System Identification with Periodic Input Signals

Zhuohua Shen[1], Yu Xu[1], Martin S. Andersen[2], and Tianshi Chen[1]

***Abstract*— Efficient implementation of algorithms for kernel-based regularized system identification is an important issue. The state of art result is based on semiseparable kernels and a class of commonly used test input signals in system identification and automatic control, and with such input signals, the output kernel is semiseparable and exploring this structure gives rise to very efficient implementation. In this paper, we consider instead the periodic input signals, which is another class of commonly used test input signals. Unfortunately, with periodic input signals, the output kernel is NOT semiseparable. Nevertheless, it can be shown that the output kernel matrix is hierarchically semiseparable (HSS). Moreover, it is possible to develop efficient implementation of algorithms by exploring the HSS structure of the output kernel matrix and the periodic structure of the regression matrix. The efficiency of the proposed implementation of algorithms is demonstrated by Monte Carlo simulations.**

I. Introduction

In the past decade, kernel-based regularized system identification has been one of the major advances in system identification, achieved many important results and become an emerging new system identification paradigm, see e.g., the survey papers [1], [2] and the book [3]. The key difference between this new paradigm and the classical paradigm based on the maximum likelihood/prediction error methods [4] is two fold. Firstly, the new paradigm finds a systematic way to embed the prior knowledge on the underlying system to be identified in the model structure through a well designed kernel. Secondly, the model complexity can be tuned through the hyper-parameter used to parameterize the kernel in a continuous and more reliable way.

The recent advance on kernel-based regularized system identification includes the kernel design and analysis [5], [6], [7], [8], efficient implementation [9], asymptotic theory [10], [11], [12], and its application in various contexts, e.g., spatial temporal data processing [13] and iterative learning control [14]. In particular, efficient implementation has been an important issue, because it is the key to apply this emerging new system identification in the engineering practice. The most widely used implementation [15] has a computational complexity of $\mathcal{O}(Nn^2+n^3)$, where N is the number of data and n is the order of the finite impulse response (FIR) model, and thus is not efficient if n is large, i.e., the underlying system to be identified has a slow dynamics. Recently, an efficient implementation was proposed in [9], based on semiseparable kernels and a class of commonly used test input signals in system identification and automatic control. With such input signals, the output kernel is semiseparable and exploring this structure gave rise to implementation with computational complexity of $\mathcal{O}(Np^2 + p^3)$, where p is the semiseparability rank of the output kernel.

Unfortunately, the implementation proposed in [9] cannot be applied to general periodic input signals, which is another class of commonly used test input signals, because the output kernel with general periodic input signals is NOT semiseparable. In this paper, we consider the case with semiseparable kernels and periodic input signals. It will be shown that the output kernel matrix is hierarchically semiseparable (HSS). Moreover, it is possible to develop efficient implementation of algorithms by exploring the HSS structure of the output kernel matrix and the periodic and Toeplitz structure of the regression matrix. The efficiency of the proposed implementation of algorithms is demonstrated by Monte Carlo simulations.

The remaining parts of this paper are organized as follows. In Section II, we introduce some background materials and the problem statement. In Section III, we give the details of our proposed implementation and then in Section IV, we illustrate our implementation by Monte Carlos simulations. Finally, we conclude this paper in Section V.

II. Background and Problem Statement

A. Kernel-based Regularized System Identification with periodic Inputs and Semiseparable Kernels

In this paper, we consider the identification of linear time-invariant (LTI), discrete-time, causal, and bounded-input bounded-ouptut (BIBO) stable systems described by

$$y(t) = G^0(q)u(t) + v(t), t = 1, \cdots, M, \tag{1}$$

where t is the time index, $M \in \mathbb{N}$ is the number of data, $y(t), u(t)$ and $v(t)$ are the measurement output, input and measurement noise of the system at time t, respectively, and $G^0(q)$ is the unknown transfer function of the system with q being the forward shift operator such that $qu(t) = u(t+1)$, and $v(t)$ is assumed to be white noise with mean zero and variance σ^2.

Since the system is LTI, causal and BIBO stable, its transfer function $G^0(q)$ has the following expansion

$$G^0(q) = \sum_{k=1}^{\infty} g_k^0 q^{-k}, \tag{2}$$

[1] Zhuohua Shen, Yu Xu, and Tianshi Chen are with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, 518172, Shenzhen, China. {zhuohuashen,yuxu19}@link.cuhk.edu.cn, tschen@cuhk.edu.cn

[2]Martin S. Andersen (mskan@dtu.dk) are with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark.

where g_k^0, $k = 1, \cdots, \infty$, are the so-called impulse response coefficients of $G^0(q)$ and absolutely summable, i.e., $\sum_{k=1}^{\infty} |g_k^0| < \infty$. Therefore, the identification of $G^0(q)$ is equivalent to the estimation of the impulse response g_k^0, $k = 1, \cdots, \infty$, which is, however, an intrinsically ill-conditioned problem with finite number of data. One way to overcome this problem is to first propose a parametric model $G(q, \theta)$ with the parameter $\theta \in \mathbb{R}^n$ and then estimate $G(q, \theta)$ based on the data $\{y(t), u(t)\}_{t=1}^N$ with $M \geq n$.

For the kernel-based regularization method (KRM), it chooses $G(q, \theta)$ as the finite impulse response (FIR) model, which is obtained by truncating the infinite impulse response of $G^0(q)$ at a sufficiently high order n as follows

$$G(q, \theta) = \sum_{k=1}^{n} g_k q^{-k}, \theta = [g_1, \cdots, g_n]^T, \tag{3}$$

where g_k, $k = 1, \cdots, n$, are called the finite impulse response coefficients, and then yields

$$y(t) = \sum_{k=1}^{n} g_k u(t-k) + v(t), t = 1, \cdots, N. \tag{4}$$

More specifically, (4) can be rewritten in the following linear regression form

$$Y_N = \Phi_N \theta + V_N \tag{5}$$

where $N = M - n$, $Y_N = [y(n+1), \cdots, y(M)]^T$, $\Phi_N^T = [\phi(n)^T, \cdots, \phi(M)^T]$ with $\phi(k) = [u(k), \cdots, u(k-n+1)]$, $k = n, n+1, \cdots, M$, and $V_N = [v(n+1), \cdots, v(M)]^T$. Then θ can be estimated by minimizing the following kernel-based regularized least squares (RLS) criterion:

$$\begin{aligned} \hat{\theta}_N &= \arg\min_{\theta} \|Y_N - \Phi_N \theta\|^2 + \gamma \theta^T K(\alpha)^{-1} \theta \\ &= \left(\Phi_N^T \Phi_N + \sigma^2 K(\alpha)^{-1}\right)^{-1} \Phi_N^T Y_N \end{aligned} \tag{6}$$

where $\|\cdot\|$ is the Euclidean norm, $\gamma > 0$ is the regularization parameter, $K(\alpha) \in \mathbb{R}^{n \times n}$ is the so-called *kernel* matrix and defined through a positive semidefinite kernel [16] $\kappa(t, s; \alpha) : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$, and $\alpha \in \mathbb{R}^p$ is the so-called hyper-parameter used to parameterize the kernel $\kappa(t, s; \alpha)$.

In practice, one needs to first design a suitable kernel $\kappa(t, s; \alpha)$, then estimate both γ and α, and finally, get the RLS estimate $\hat{\theta}_N$. By far the most effective method to estimate α is the so-called empirical Bayes method. This method further assumes that the measurement noise $v(t)$ is Gaussian, $\theta \sim N(0, K(\alpha))$, and moreover, θ is independent of $v(t)$, $t = 1, \cdots, N$, and $\gamma = \sigma^2$. Under these assumptions, it is easy to verify that the RLS estimate $\hat{\theta}_N$ in (6) is also the maximum a posterior estimate of θ and the hyper-parameter α is estimated by maximizing the marginal likelihood of α, i.e., $\hat{\alpha} \triangleq \arg\max_{\alpha} p(Y_N \mid \alpha)$ or equivalently,

$$\begin{aligned} \hat{\alpha} &= \arg\min_{\alpha} Y_N^T H(\alpha)^{-1} Y_N + \log |H(\alpha)| \\ H(\alpha) &= \Phi_N K(\alpha) \Phi_N^T + \sigma^2 I_N, \end{aligned} \tag{7}$$

where $|\cdot|$ is the determinant of a matrix, I_N is the N-dimensional identity matrix, and $H(\alpha)$ is often called the output kernel matrix.

The straightforward computation of $\hat{\theta}_N$ in (6) and $\hat{\alpha}$ in (7) has computational complexity of $\mathcal{O}(N^3)$. In order to apply the KRM in practice with large N, several efficient implementations have been developed over the past decade, e.g., [17], [15], [9]. In this paper, we focus on this topic under the following assumption.

Assumption 1. *Assume that the input signal $u(t)$ is periodic with period $p \in \mathbb{N}$, and the kernel matrix $K(\alpha)$ is extended $\{k\}$-generator representable semiseparable with $k \in \mathbb{N}$, and moreover, $N \geq n \geq p$.*

1) periodic Input Signals: On the one hand, it is worth to mention that periodic input signals are one class of most widely used test signals in system identification and control, see e.g., [4]. Under Assumption 1, Φ_N in (5) has a periodic structure with period $p \in \mathbb{N}$, i.e.,

$$\Phi_N(i, j) = \Phi_N(i + t_1 p, j + t_2 p) \tag{8}$$

for all $1 \leq i, i + t_1 p \leq N$, $1 \leq j, j + t_2 p \leq n$, and $t_1, t_2 \in \mathbb{Z}$, and Φ_N can be rewritten as follows

$$\Phi_N = \begin{bmatrix} \Phi_b & \Phi_b & \dots & \Phi_{bc} \\ \Phi_b & \Phi_b & \ddots & \Phi_{bc} \\ \vdots & \vdots & \vdots & \vdots \\ \Phi_{br} & \Phi_{br} & \dots & \Phi_{bb} \end{bmatrix} \tag{9}$$

where $\Phi_b = \Phi_N(1 : p, 1 : p)$, $\Phi_{br} = \Phi_N(1 : N\%p, 1 : n\%p)$, $\Phi_{bc} = \Phi_N(1 : p, 1 : n\%p)$, $\Phi_{bb} = \Phi_N(1 : N\%p, 1 : n\%p)$, "%" represents the modulo operation, and $\Phi_N(1 : a, 1 : b)$ with $a, b \in \mathbb{N}$ the submatrix of Φ_N consisting of the first a rows and b columns of Φ_N.

2) Semiseparable Kernels: On the other hand, most of the existing kernels proposed in KRM are semiseparable, see e.g., [9]. As a result, the kernel matrix $K(\alpha)$ is extended $\{k\}$-generator representable semiseparable for some $k \in \mathbb{N}$. Recall that a symmetric matrix $K \in \mathbb{R}^{n \times n}$ is said to be extended $\{k\}$-generator representable semiseparable [18, p.304] if there exists $k \in \mathbb{N}, k \leq n$ such that

$$K = S(U, V) = \text{tril}(UV^T) + \text{triu}(VU^T, 1) \tag{10}$$

where $U, V \in \mathbb{R}^{n \times k}$ are called *generators* of K, $\text{tril}(UV^T)$ denotes the lower-triangular matrix generated by UV^T by zeroing all its (i, j) entries with $j > i$, and $\text{triu}(VU^T, 1)$ the upper-triangular matrix generated by VU^T by zeroing all its (i, j) entries with $i > j - 1$.

B. Problem Statement

In this paper, we study the problem of how to develop more efficient implementation algorithms for the computation of $\hat{\alpha}$ in (7) (and also $\hat{\theta}_N$ in (6), but it is skipped, because the space is limited and its computation is a byproduct of the computation of $\hat{\alpha}$ in (7)) with lower computational complexity than the existing ones, e.g., [17], [15], [9], under Assumption 1 by making use of the structural properties of Φ_N and $K(\alpha)$ as sketched in Sections II-A.1 and II-A.2, and exploring the structural properties of $H(\alpha)$ defined in (7).

Remark II.1. *As can be seen from [9], the computation of the cost function of the empirical Bayes method* (7) *shares some common elements as that of the other hyper-parameter estimation methods, such as the Stein's unbiased risk estimation (SURE) method and the generalized cross validation (GCV) method. Therefore, the proposed implementation algorithms can be used to develop efficient implementation algorithms also for the SURE and GCV methods.*

C. Fundamentals of Efficient Implementation by Exploring HSS Matrix

In this subsection, we introduce some fundamentals of efficient implementation by exploring the structure of HSS matrix.

As will be shown shortly in Proposition 2, $H(\alpha)$ in (7) is actually an HSS matrix. HSS matrices are a generalization of semi-separable matrices, and also a special case of hierarchically off-diagonal low-rank (HODLR) matrices [19], [20], [21]. HODLR matrices have low-rank off-diagonal blocks, while the diagonal blocks can be subdivided into the same form recursively, i.e., they also have low-rank off-diagonal blocks. The HSS matrices have the additional property that the off-diagonal blocks can depend on the off-diagonal blocks in the deeper level's diagonal blocks [21]. More specifically, if $A \in \mathbb{R}^{N\times N}$ is a two-level HODLR matrix, then A can be represented as

$$A = \begin{bmatrix} A_1^{(1)} & U_1^{(1)}K_{12}^{(1)}V_2^{(1)^T} \\ U_2^{(1)}K_{21}^{(1)}V_1^{(1)^T} & A_2^{(1)} \end{bmatrix} \tag{11}$$

where the off-diagonal blocks $U_1^{(1)}K_{12}^{(1)}V_2^{(1)^T}$ and $U_2^{(1)}K_{21}^{(1)}V_1^{(1)^T}$ are of low-rank, and $A_1^{(1)}, A_2^{(1)}$ can be further divided into

$$\begin{aligned} A_1^{(1)} &= \begin{bmatrix} A_1^{(2)} & U_1^{(2)}K_{12}^{(2)}V_2^{(2)^T} \\ U_2^{(2)}K_{21}^{(2)}V_1^{(2)^T} & A_2^{(2)} \end{bmatrix} \\ A_2^{(1)} &= \begin{bmatrix} A_3^{(2)} & U_3^{(2)}K_{34}^{(2)}V_4^{(2)^T} \\ U_4^{(2)}K_{43}^{(2)}V_3^{(2)^T} & A_4^{(2)} \end{bmatrix} \end{aligned} \tag{12}$$

which is in the same structure as A until it reaches the size or rank threshold [22]. For HSS matrices, the matrices $U_1^{(1)}, U_2^{(1)}, V_1^{(1)}, V_2^{(1)}$ can further depend on $U_i^{(2)}$ and $V_j^{(2)}$ recursively through the so-called *translation operators*. For example, there are translation operators $R_k^{(2)}, W_k^{(2)}, k = 1,2,3,4$ such that

$$U_i^{(1)} = \begin{bmatrix} U_{2i-1}^{(2)}R_{2i-1}^{(2)} \\ U_{2i}^{(2)}R_{2i}^{(2)} \end{bmatrix},\ V_i^{(i)} = \begin{bmatrix} V_{2i-1}^{(2)}W_{2i-1}^{(2)} \\ V_{2i}^{(2)}W_{2i}^{(2)} \end{bmatrix},\ i = 1,2 \tag{13}$$

It is similar for other U_i, V_i's at each level [19].

The HODLR and HSS matrices allow efficieent implementation of algorithms for arithmetical operations [23], e.g., addition, inversion, and QR decomposition.

III. How to explore the structure

We show how to develop efficient algorithms by exploring the structure of $H(\alpha)$ to compute $Y_N^T H(\alpha)^{-1} Y_N$ and $\log|H(\alpha)|$ efficiently.

A. Preamble

The key idea is sketched below. First, we show that when Φ_N is periodic, the output kernel matrix $H(\alpha)$ has a HSS structure. Then by referring to [19], [23], we modify a fast full QR decomposition to factorize $H(\alpha)$ into different levels such that for $k = l-1, l-2, \ldots, 2, 1$, we have

$$H^{(k)}(\alpha) = W^{(k)}P^{(k)}\mathrm{diag}(H^{(k-1)}, \sigma^2 I_{2^k p})P^{(k)^T}W^{(k)^T}, \tag{14}$$

where k is the level index, l is the deepest level chosen for optimal computational complexity, $W^{(k)}$ is an orthogonal matrix, $P^{(k)}$ is a permutation matrix, and $\mathrm{diag}(A, B)$ means the diagonal matrix generated by square blocks A and B. When the factorization is finished at the l^{th} step,

$$H^{(0)} = W^{(0)}P^{(0)}\mathrm{diag}(D(\alpha), \sigma^2 I_p)P^{(0)^T}W^{(0)^T} \in \mathbb{R}^{2p\times 2p} \tag{15}$$

where $D(\alpha) = R^{(0)}K(\alpha)R^{(0)^T} + \sigma^2 I_p \in \mathbb{R}^{p\times p}$, which can be computed efficiently by exploring the semi-separable structure of $K(\alpha)$. In $H(\alpha)$, only $D(\alpha)$ changes as α changes. Since Y_N is fixed, all the computations in $Y_N^T H(\alpha)^{-1} Y_N$ except the one related to $D(\alpha)$ are only needed to be done once; since the determinant of $W^{(\cdot)}$ and $P^{(\cdot)}$ is either 1 or -1, the computation of $\log|H(\alpha)|$ is almost about that of $|D(\alpha)|$. The cost for computing $|D(\alpha)|$ is free of N, which is much smaller when $N \gg n, p$.

B. Efficient implementation by exploring HSS structure of $H(\alpha)$ and periodic structure of Φ_N

1) HSS Structure: We first show that $H(\alpha)$ is an HSS matrix. Since Φ_N is periodic, let the *deepest level* be l (the choice for l will be discussed later), divide it into 2^l parts according to rows:

$$\Phi_N = \begin{bmatrix} \Phi_1^{(l)^T} & \Phi_2^{(l)^T} & \cdots & \Phi_{2^l}^{(l)^T} \end{bmatrix}^T \tag{16}$$

where for $i = 1, \ldots, 2^l - 1$,

$$\Phi_i^{(l)} = \begin{bmatrix} \Phi_b & \cdots & \Phi_{bc} \\ \vdots & \vdots & \vdots \\ \Phi_b & \cdots & \Phi_{bc} \end{bmatrix}_{m\times n}, \Phi_{2^l}^{(l)} = \begin{bmatrix} \Phi_b & \cdots & \Phi_{bc} \\ \vdots & \vdots & \vdots \\ \Phi_{br} & \cdots & \Phi_{bb} \end{bmatrix}_{r\times n},$$

m is a multiple of p, r satisfies $2^{l-1}m + r = N$, normally $m \approx r$, $2^l m \approx N$, and m is not necessarily larger than n. Then H (α is sometimes omitted for simplicity) becomes

$$\begin{aligned} H = &\begin{bmatrix} \Phi_1^{(l)}K\Phi^{(l)^T} & \Phi_1^{(l)}K\Phi_2^{(l)^T} & \cdots & \Phi_1^{(l)}K\Phi_{2^l}^{(l)^T} \\ \Phi_2^{(l)}K\Phi_1^{(l)^T} & \Phi_2^{(l)}K\Phi_2^{(l)^T} & \cdots & \Phi_2^{(l)}K\Phi_{2^l}^{(l)^T} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{2^l}^{(l)}K\Phi_1^{(l)^T} & \Phi_{2^l}^{(l)}K\Phi_2^{(l)^T} & \cdots & \Phi_{2^l}^{(l)}K\Phi_{2^l}^{(l)^T} \end{bmatrix} \\ &+ \sigma^2\mathrm{diag}(I_m, I_m, \ldots, I_r), \end{aligned} \tag{17}$$

which is an HSS matrix, as shown in the proposition below.

Proposition 2. *Suppose that Assumption 1 holds. Then H in* (7) *or* (17) *is a HSS matrix with identity translation operator.*

According to [20], [21], [23], it is possible to develop fast factorization for HSS H based on full QR decomposition for Φ_N. An efficient QR decomposition for Φ_N will be proposed in the next section.

2) Efficient QR Decomposition for periodic Φ_N: We first consider the QR decomposition:

$$\Phi_N(1:N,1:p) = \begin{bmatrix}\Phi_b^T & \cdots & \Phi_b^T & \Phi_{br}^T\end{bmatrix}^T = Q\begin{bmatrix}R_b^T & 0\end{bmatrix}^T, \tag{18}$$

where $Q \in \mathbb{R}^{N\times N}$ is orthogonal, $R_b \in \mathbb{R}^{p\times p}$. Then Q^T can be applied to the whole Φ_N:

$$Q^T\Phi_N = \begin{bmatrix} R_b & .. & R_b & R_{bc} \\ 0_{(N-p)\times p} & .. & 0_{(N-p)\times p} & 0_{(N-p)\times(n\%p)}\end{bmatrix} \triangleq \begin{bmatrix}R \\ 0\end{bmatrix} \tag{19}$$

where $R_{bc} = \begin{bmatrix}R_b(1:n\%p, 1:n\%p) \\ 0_{(p-n\%p)\times(n\%p)}\end{bmatrix}$, $R \in \mathbb{R}^{p\times n}$. In the following, we consider the QR decomposition based on the Householder transformation because of its nice properties when factorizing periodic Φ_N.

By referring to [24, Algorithms 5.1.1 and 5.2.1], we propose an adapted algorithm to compute the QR decomposition in (18):

1) **Algorithm 1** computes the Householder transformation $H_{\mathbf{v}} \in \mathbb{R}^{N\times N}$ of a given periodic vector $\mathbf{x} \in \mathbb{R}^N$. Since the Householder vector $\mathbf{v} \in \mathbb{R}^N$ is periodic except the first entry, we set $\mathbf{v}_b$ and constant β as outputs such that
   - $\tilde{\mathbf{v}}_b = \mathbf{v}_b(2:p+1)$;
   - $\mathbf{v} = [\mathbf{v}_b(1) \underbrace{\tilde{\mathbf{v}}_b^T \cdots \tilde{\mathbf{v}}_b^T}_{m \text{ times}} \tilde{\mathbf{v}}_b^T(1:r)]^T$, where $m = \lfloor (N-1)/p \rfloor$, $r = (N-1)\%p$.
   - The transformation can be expressed as $H_{\mathbf{v}} = I_N - \beta\mathbf{v}\mathbf{v}^T$.
2) **Algorithm 2** computes the Householder QR decomposition of the given periodic matrix $\Phi_N(1:N,1:p) \in \mathbb{R}^{N\times p}$. The outputs are: upper triangular matrix $R_b \in \mathbb{R}^{p\times p}$; vectors $\mathbf{v}_b^{(k)}$ and constants $\beta^{(k)}$, $k = 1,\ldots,p$. Then the Q in (18) can be represented by following:
   - $\tilde{\mathbf{v}}_b^{(k)} = \mathbf{v}_b^{(k)}(2:p+1)$.
   - $\mathbf{v}^{(k)} = [\mathbf{v}_b^{(k)}(1) \underbrace{\tilde{\mathbf{v}}_b^{(k)T} \cdots \tilde{\mathbf{v}}_b^{(k)T}}_{m^{(k)} \text{ times}} \tilde{\mathbf{v}}_b^{(k)T}(1 : r^{(k)})]^T$, where $m^{(k)} = \lfloor (N-k)/p \rfloor$, $r^{(k)} = (N-k)\%p$.
   - $H_{\mathbf{v}_k} = I_{N-k+1} - \beta^{(k)}(\mathbf{v}^{(k)})(\mathbf{v}^{(k)})^T$
   - $Q = \prod_{k=1}^{p} \text{diag}(I_{k-1}, H_{\mathbf{v}_k})$.

All the outputs in both algorithms are represented using their periodic part, as we use Φ_b to represent Φ_N. The computational costs of the two algorithms as well as matrix-vector product are shown in the following proposition.

Proposition 3. *Given a periodic matrix $\Phi_N \in \mathbb{R}^{N\times n}$ with period p, the cost of doing Householder QR decomposition for Φ_N with periodic part-only outputs (i.e., R_b in (19), $\mathbf{v}_b^{(k)}$ and $\beta^{(k)}$ above) is about $2p^2(p-1) = \mathcal{O}(p^3)$ flops. Besides, given a vector $\mathbf{x} \in \mathbb{R}^N$, $Q^T\mathbf{x}$ costs about $3Np = \mathcal{O}(Np)$ flops.*

Algorithm 1 Householder transformation for a periodic vector (House)

Input: periodic part $\mathbf{x}_b = \mathbf{x}(1:p)$, size N.
Output: First $p+1$ elements of Householder vector $\mathbf{v}_b = \mathbf{v}(1:p+1)$, β.

$\mathbf{x}_b' \leftarrow \begin{bmatrix}\mathbf{x}_b(2:p) \\ \mathbf{x}_b(1)\end{bmatrix}$;
$m \leftarrow \left\lfloor \frac{N-1}{p} \right\rfloor$; $r \leftarrow (N-1)\%p$;
$\sigma \leftarrow m(\mathbf{x}_b')^T\mathbf{x}_b' + (\mathbf{x}_b')^T(1:r)\mathbf{x}_b'(1:r)$;
$\mathbf{v}_b \leftarrow \begin{bmatrix}1 \\ \mathbf{x}_b'\end{bmatrix}$;
if $\sigma = 0$ **then**
  $\beta \leftarrow \sigma$;
else
  $\mu \leftarrow \sqrt{(\mathbf{x}_b(1))^2 + \sigma}$;
  if $\mathbf{x}_b(1) \le 0$ **then**
    $\mathbf{v}_b(1) \leftarrow \mathbf{x}_b(1) - \mu$
  else
    $\mathbf{v}_b(1) \leftarrow \frac{-\sigma}{\mathbf{x}_b(1)+\mu}$;
  end if
  $\beta \leftarrow \frac{2(\mathbf{v}_b(1))^2}{\sigma+(\mathbf{v}_b(1))^2}$;
  $\mathbf{v}_b \leftarrow \frac{\mathbf{v}_b}{\mathbf{v}_b(1)}$;
end if
$\tilde{\mathbf{v}}_b \leftarrow \mathbf{v}_b(2:p+1)$;
$\mathbf{v} \leftarrow [\mathbf{v}_b^T(1) \underbrace{\tilde{\mathbf{v}}_b^T \cdots \tilde{\mathbf{v}}_b^T}_{m \text{ times}} \tilde{\mathbf{v}}_b^T(1:r)]^T$.

Algorithm 2 Householder QR decomposition for a periodic matrix

Input: periodic part $\Phi_b = \Phi_{trun}(1:p,1:p)$, size N.
Output: First $p+1$ elements of each Householder vector $\mathbf{v}_b^{(k)} = \mathbf{v}^{(k)}(1:p+1), \beta^{(k)}, R_b$.

$\Phi_b^{(1)} \leftarrow \Phi_b$;
for $k := 1$ to p **do**
  $n^{(k)} \leftarrow p - (k-1)$;
  $m^{(k)} \leftarrow \left\lfloor \frac{N-k}{p} \right\rfloor$; $r^{(k)} \leftarrow (N-k)\%p$;
  $[\mathbf{v}_b^{(k)}, \beta^{(k)}] \leftarrow \text{House}(\Phi_b^{(k)}(1:p,1), N-(k-1))$;
  $\Phi_b'^{(k)} \leftarrow \begin{bmatrix}\Phi_b^{(k)}(2:p, 1:n^{(k)}) \\ \Phi_p^{(k)}(1, 1:n^{(k)})\end{bmatrix}$;
  $S_f^{(k)} \leftarrow \mathbf{v}_b^{(k)}(1)\Phi_b^{(k)}(1, 1:n^{(k)})$;
  $S_m^{(k)} \leftarrow m^{(k)}(\mathbf{v}_b^{(k)})^T(2:p+1)\Phi_b'^{(k)}$;
  $S_r^{(k)} \leftarrow (\mathbf{v}_b^{(k)})^T(1:r+1)\Phi_b'^{(k)}(1:r, 1:n^{(k)})$;
  $S^{(k)} \leftarrow S_f^{(k)} + S_m^{(k)} + S_r^{(k)}$;
  $T_b^{(k)} \leftarrow \beta^{(k)}\mathbf{v}_b^{(k)}S^{(k)}$;
  $\tilde{\Phi}_b'^{(k)} \leftarrow \begin{bmatrix}\Phi_b^{(k)}(1, 1:n^{(k)}) - T_b^{(k)}(1, 1:n^{(k)}) \\ \Phi_b'^{(k)} - T_b^{(k)}(2:p+1, 1:n^{(k)})\end{bmatrix}$;
  $R_b(k, k:p) \leftarrow \tilde{\Phi}_b'^{(k)}(1, 1:n^{(k)})$;
  if $k \neq p$ **then**
    $\Phi_b^{(k+1)} \leftarrow \tilde{\Phi}_b'^{(k)}(2:p+1, 2:n^{(k)})$;
  end if
end for

3) Computational Complexity by Exploring the HSS structure: By [20], [21], [23], the cost for computing $\hat{\alpha}$ in (7) by exploring the HSS structure and Householder QR decomposition using **Algorithm 1** and **Algorithm 2** is given in the theorem below.

Theorem 4. *Suppose that Assumption 1 holds and the deepest level of $H(\alpha)$ is l. Then the computational complexity of computing both $Y_N^T H(\alpha)^{-1} Y_N$ and $\log|H(\alpha)|$ is*

$$\underbrace{[3Np + (\frac{7}{3} + \frac{5}{3}l)p^3 + (12 \cdot 2^l - 2l - 8)p^2]}_{\text{irrespective of } \alpha} + \underbrace{f(n,p,k)}_{\text{dependent on } \alpha} \tag{20}$$

where $f(n,p,k)$ is the cost of computing the part dependent on α, and the lower-order terms such as $\mathcal{O}(p)$ are ignored. Moreover, the computational complexity (20) *is minimized at $l = 0$, and the cost becomes $[3Np + \frac{7}{3}p^3 + 4p^2] + f(n,p,k)$.*

Theorem 4 shows that to have the most efficient implementation, it is enough to keep $H(\alpha)$ as a whole and just compute the full Householder QR decomposition for Φ_N. As a result, we propose to compute $Y_N^T H(\alpha)^{-1} Y_N$ and $\log|H(\alpha)|$ in the following four steps:

i) Compute full QR decomposition of $\Phi_N = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$ using **Algorithm 2**, $Q \in \mathbb{R}^{N \times N}$, $R \in \mathbb{R}^{p \times n}$ is periodic. Then

$$H(\alpha) = Q[\mathrm{diag}(D(\alpha), \sigma^2 I_{N-p})]Q^T \tag{21}$$

where $D(\alpha) = RK(\alpha)R^T + \sigma^2 I_p$.

ii) Compute $\bar{Y}_N = Q^T Y_N$. Let $\bar{Y}_N = \begin{bmatrix} Y_1^T & Y_2^T \end{bmatrix}^T$, $Y_1 = \bar{Y}_N(1:p)$, $Y_2 = \bar{Y}_N(p+1:N)$. Then compute $S_2 = \sigma^{-2} Y_2^T Y_2$.

iii) Compute explicit $D(\alpha)$ using the semiseparable structure of $K(\alpha)$:
   - Compute KR^T, each matrix-vector multiplication of $KR^T(1:n,i), i = 1,\ldots,p$ is obtained by [25, Algorithm 4.1.], which provides an efficient way to calculate semiseparable matrix-vector product.
   - Compute $R(KR^T) + \sigma^2 I_p$ using the normal computation way.

   Then compute the inverse $D(\alpha)^{-1}$. We have

$$S_1(\alpha) = Y_1^T D(\alpha)^{-1} Y_1 \tag{22}$$

   The result of $Y_N^T H(\alpha) Y_N = S_1(\alpha) + S_2$.

iv) Compute

$$\log|H(\alpha)| = (N-p)\log(\sigma^2) + \log|D(\alpha)| \tag{23}$$

directly using explicit $D(\alpha)$.

It is worth to note that the first two steps, i.e., i)-ii), are irrespective of α, but the last two steps depend on α. The computational cost $f(n,p,k)$ in Theorem 4 of the last two steps, i.e., iii)-iv), is shown in the next section.

4) Computation Complexity $f(n,p,k)$ of (22) *and* (23)*:* Each update of α in the solution of (7) requires the recomputation of (22) and (23) with fixed R, where the computational complexity can be reduced by exploring the semiseparable structure of K.

Proposition 5. *Suppose that Assumption 1 holds and let $K \in \mathbb{R}^{n \times n}$ be a semiseparable matrix in the form of* (10) *with the semiseparability rank k, and $R \in \mathbb{R}^{p \times n}$ be the QR decomposition of Φ_N. Then, it holds that*

- *the computation cost of* (22) *is $8npk + np^2 + \frac{4}{3}p^3$;*
- *the computation cost of* (23) *is $\mathcal{O}(p^3)$;*

and hence

$$f(n,p,k) = [8npk + np^2 + \frac{4}{3}p^3] + [\mathcal{O}(p^3)]. \tag{24}$$

5) Overall Computational Complexity:

Theorem 6 (Complexity)**.** *Following Theorem 4, the total cost for both $Y_N^{-1} H(\alpha)^{-1} Y_N$ and $\log|H(\alpha)|$ is*

$$\underbrace{[3Np + \frac{7}{3}p^3 + 4p^2]}_{\text{i), ii): irrespective of } \alpha} + \underbrace{[8npk + np^2 + \frac{4}{3}p^3] + [\mathcal{O}(p^3)]}_{\text{iii), iv): dependent on } \alpha, \text{ i.e., } f(n,p,k)}. \tag{25}$$

IV. Numerical Simulation

In this section, we run numerical simulations to test the efficacy of the proposed method. In particular, we compare the performances between the proposed implementation (denoted by `RFIR-hss`) and Algorithm 2 in [15] (denoted by `RFIR`), which is matrix-inversion-free and based on QR factorization. We test the KRM with the Tuned-Correlated (TC) kernel which is extended 1-semiseparable [9]. In particular, we estimate the hyper-parameters and the noise variance σ^2 by maximizing the marginal likelihood (7), which is computed by `RFIR` and `RFIR-hss`. Then the corresponding regularized impulse response estimates (6) are computed. The numerical optimization of (7) is as follows: first we choose an initial point of the hyper-parameters by grid search and then optimization is implemented by the MATLAB function `fmincon` with the interior-point algorithm.

The input signal is a periodic random Gaussian signal using the entire frequency range with period p and the output additive white Gaussian noise $v(t)$ is generated with variance one tenth of the variance of the noise-free output.

A. Accuracy test

In this test, we run Monte Carlo simulations to test the accuracy of the proposed method. Specifically, we generate 80 discrete-time linear systems of 10th order with the moduli of all the poles within $[0.1, 0.9]$. The number of data points M is chosen to be 600, the FIR model order n is chosen to be 50 and the period p is chosen to be 40. To assess the estimation performance, we define the model fit:

$$\text{fit} = 100\left(1 - \left[\frac{\sum_{k=1}^n |g_k^0 - \hat{g}_k|}{\sum_{k=1}^n |g_k^0 - \bar{g}^0|}\right]^{1/2}\right), \ \bar{g}^0 = \frac{1}{n}\sum_{k=1}^n g_k^0, \tag{26}$$

where g_k^0 and $\hat{g}_k$ are the true and the estimated impulse response at the kth order, respectively. The averaged model fits for `RFIR` and `RFIR-hss` are 86.326354 and 86.326355 respectively. The distribution of the model fit difference between `RFIR` and `RFIR-hss` is shown in Fig. 1. We observe that `RFIR-hss` gives almost the same accuracy

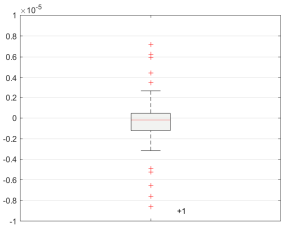

Fig. 1: The model fit difference between `RFIR` and `RFIR-hss`.

	$n = 300$	$n = 600$	$n = 1200$	$n = 2400$	$n = 4800$
`RFIR`	2.471	15.583	75.227	372.419	2327.687
`RFIR-hss`	0.953	1.219	1.761	3.005	7.298

TABLE I: The averaged computation time (in seconds) by `RFIR` and `RFIR-hss` with respect to $n = 300,\ 600,\ 1200,\ 2400$ and 4800.

performance as `RFIR`, which justifies the correctness of our derivations.

B. Efficiency test

In this test, we generate a data set from a fixed system

$$G(q) = \frac{-0.3377q}{(q - 0.9542)(q - 0.9758)}. \tag{27}$$

The number of data points M is chosen to be 10000 and the period p is chosen to be 200 while we choose the FIR model order n to be 300, 600, 1200, 2400 and 4800. For each n, we identify the system for 10 times and measure the corresponding averaged computation time for evaluating the marginal likelihood (7) 567 times (in the initial grid search) by `RFIR` and `RFIR-hss`. The averaged computation time with respect to the FIR model n is shown in TABLE. I, which indicates that `RFIR-hss` is significantly faster than `RFIR` as n grows larger. The averaged model fits by `RFIR` and `RFIR-hss` are 94.1835425 and 94.1835434, respectively.

V. Conclusion

In this paper, we proposed an efficient implementation for kernel-based regularized system identification with semiseparable kernels and periodic input signals. The proposed implementation, as illustrated by the simulation results, is more efficient than the existing one and thus offers the users more efficient implementations of algorithms in practice.

Acknowledgment

This work is funded by the National Natural Science Foundation of China (no. 62273287), the Shenzhen Science and Technology Innovation Council (no. JCYJ20220530143418040 and JCY20170411102101881), the Novo Nordisk Foundation (no. NNF20OC0061894), the Villum Foundation (no. 25893), the Thousand Youth Talents Plan funded by the central government of China, and the Undergraduate Research Award funded by the Chinese University of Hong Kong, Shenzhen.

References

[1] 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, 2014.

[2] L. Ljung, T. Chen, and B. Mu, "A shift in paradigm for system identification," *International Journal of Control*, vol. 93, no. 2, pp. 173–180, 2020.

[3] G. Pillonetto, T. Chen, A. Chiuso, G. De Nicolao, and L. Ljung, *Regularized System Identification: Learning Dynamic Models from Data*. Springer Nature, 2022.

[4] L. Ljung, *System identification: theory for the user*. Upper Saddle River, NJ: Prentice Hall, 1999.

[5] T. Chen, "On kernel design for regularized LTI system identification," *Automatica*, vol. 90, pp. 109–122, 2018.

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

[7] M. Zorzi, "A second-order generalization of TC and DC kernels," *arXiv preprint arXiv:2109.09562*, 2022.

[8] M. Bisiacco and G. Pillonetto, "On the mathematical foundations of stable RKHSs," *Automatica*, vol. 118, p. 109038, 2020.

[9] T. Chen and M. S. Andersen, "On semiseparable kernels and efficient implementation for regularized system identification and function estimation," *Automatica*, vol. 132, p. 109682, 2021.

[10] B. Mu, T. Chen, and L. Ljung, "On asymptotic properties of hyperparameter estimators for kernel-based regularization methods," *Automatica*, vol. 94, pp. 381–395, 2018.

[11] ——, "On the asymptotic optimality of cross-validation based hyperparameter estimators for regularized least squares regression problems," *arXiv preprint arXiv:2104.10471*, 2021.

[12] Y. Ju, B. Mu, L. Ljung, and T. Chen, "Asymptotic theory for regularized system identification part I: Empirical Bayes hyper-parameter estimator," *IEEE Transactions on Automatic Control*, pp. 1–16, 2023.

[13] J. Zhang, Y. Ju, B. Mu, R. Zhong, and T. Chen, "An efficient implementation for spatial–temporal Gaussian process regression and its applications," *Automatica*, vol. 147, p. 110679, 2023.

[14] X. Yu, X. Fang, B. Mu, and T. Chen, "Kernel-based regularized iterative learning control of repetitive linear time-varying systems," *arXiv preprint arXiv:2303.03822*, 2023.

[15] T. Chen and L. Ljung, "Implementation of algorithms for tuning parameters in regularized least squares problems in system identification," *Automatica*, vol. 49, no. 7, pp. 2213 – 2220, 2013.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006.

[17] F. P. Carli, A. Chiuso, and G. Pillonetto, "Efficient algorithms for large scale linear system identification using stable spline estimators," *IFAC Proceedings Volumes*, vol. 45, no. 16, pp. 119–124, 2012, 16th IFAC Symposium on System Identification.

[18] R. Vandebril, M. V. Barel, and N. Mastronardi, *Matrix Computations and Semiseparable Matrices: Linear Systems*. Johns Hopkins University Press, Baltimore, 2008.

[19] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, "Fast algorithms for hierarchically semiseparable matrices," *Numerical Linear Algebra with Applications*, vol. 17, no. 6, pp. 953–976, 2010.

[20] S. Chandrasekaran, P. Dewilde, M. Gu, T. Pals, X. Sun, A. J. van der Veen, and D. White, "Some fast algorithms for sequentially semiseparable representations," *SIAM Journal on Matrix Analysis and Applications*, vol. 27, no. 2, pp. 341–364, 2005.

[21] S. Chandrasekaran, M. Gu, and T. Pals, "A fast ULV decomposition solver for hierarchically semiseparable representations," *SIAM Journal on Matrix Analysis and Applications*, vol. 28, no. 3, pp. 603–622, 2006.

[22] S. Ambikasaran, M. O'Neil, and K. R. Singh, "Fast symmetric factorization of hierarchical matrices with applications," *arXiv preprint arXiv:1405. 0223*, 2016.

[23] S. Massei, L. Robol, and D. Kressner, "hm-toolbox: Matlab software for HODLR and HSS matrices," *SIAM Journal on Scientific Computing*, vol. 42, no. 2, pp. C43–C68, 2020.

[24] G. H. Golub and C. F. van Loan, *Matrix Computations*, 4th ed. Johns Hopkins University Press, Baltimore, 2013.

[25] M. S. Andersen and T. Chen, "Smoothing splines and rank structured matrices: Revisiting the spline kernel," *SIAM Journal on Matrix Analysis and Applications*, vol. 41, no. 2, pp. 389–412, 2020.