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

Towards Scalable Kernel-Based Regularized System Identification

Lujing Chen[1], Tianshi Chen[2], Utkarsh Detha[3], and Martin S. Andersen[1]

Abstract— **This paper proposes a methodology for scalable kernel-based regularized system identification based on indirect methods. It leverages stochastic trace estimation methods and an iterative solver such as LSQR for the efficient evaluation of hyperparameter selection criteria. It also uses a derivative-free optimization approach to hyperparameter estimation, which avoids the need for computing gradients or Hessians of the objective function. Moreover, the method is matrix-free, which means it only relies on a matrix-vector oracle and exploits fast routines for various structured matrix-vector products. Our preliminary numerical experiments indicate that the methodology scales significantly better than direct methods, especially when dealing with large datasets and slowly decaying impulse responses.**

I. INTRODUCTION

System identification is a fundamental problem in control theory with important applications in several domains. A popular approach is to use kernel-based regularization to introduce prior information about stability and smoothness [18]. Kernels such as the SS kernel [19] and the TC and DC kernels [6] have been thoroughly studied and often work very well in practice. These kernels lead to kernel matrices that can be efficiently factorized and enjoy linear time and space complexity; see, e.g., [5], [2]. However, when combined with the input matrix, which is a Toeplitz matrix, it is generally difficult to concurrently exploit the Toeplitz structure and the efficient representation of the kernel matrix when employing a direct method based on, e.g., a QR factorization. Thus, kernel-based regularization methods can be computationally expensive, especially when dealing with large datasets.

This paper proposes a methodology for scalable kernel-based regularized system identification based on *indirect* methods. The performance of kernel-based regularization methods is highly dependent on the choice of hyperparameters, which control the trade-off between fitting the training data and avoiding overfitting. Our approach leverages stochastic trace estimation methods and an iterative solver such as the conjugate gradient (CG) method [11] or LSQR [17] for the efficient (but inexact) evaluation of hyperparameter selection criteria such as generalized cross-validation (GCV) and profile marginal likelihood (PML). We combine this with a derivative-free optimization method for hyperparameter estimation, which avoids the need for computing gradients or Hessians of the objective function. The method is also matrix-free in the sense that it only relies on the availability of a matrix-vector oracle rather than an explicit representation of a matrix. This means that the memory footprint is low and that fast routines for various structured matrix-vector products are readily exploited, e.g., products with Toeplitz, band, and/or semiseparable matrices. Using techniques from randomized linear algebra, we construct efficient preconditioners that significantly improve the computational efficiency of the overall methodology.

Our work builds on several existing techniques in the field. For instance, we leverage the use of trace estimation for hyperparameter selection, which was first explored by Hutchinson [12]. Since then, Hutchinson's trace estimator has been extended to the estimation of the trace of functions of large matrices using stochastic Lanczos quadrature [21]. We also draw upon sketching and randomized linear algebra techniques for trace and/or log-determinant estimation, which have been extensively studied in the literature, including in [20] and [4]. Additionally, our methodology relies on preconditioning, which has recently been explored in the context of Gaussian process regression [22]. These techniques are crucial components of our proposed methodology.

The paper is organized into the following sections. In Section II, we provide a brief review of the model and hyperparameter selection criteria that form the basis of our proposed methodology. In Section III, we outline our methodology, including iterative solver techniques, stochastic trace estimation methods, and derivative-free optimization methods that we employ. We present some numerical results in Section IV, where we showcase the efficiency and scalability of our methodology. Finally, in Section V, we summarize our contributions and conclude the paper.

II. BACKGROUND

A. Model

Consider the linear observation model

$$y = \Phi\theta + e \tag{1}$$

where $y \in \mathbb{R}^m$ is a vector of observations, $\Phi \in \mathbb{R}^{m\times n}$ is a Toeplitz matrix constructed from a given input signal $u \in \mathbb{R}^m$, $\theta \in \mathbb{R}^n$ is a vector of unknown parameters, and $e \in \mathbb{R}^m$ represents measurement noise. We will assume that θ and e are independent random variables satisfying

$$\theta|\nu,\beta \sim \mathcal{N}(0,\nu K(\beta)), \quad e|\sigma^2 \sim \mathcal{N}(0,\sigma^2 I),$$

*This work is funded by the Novo Nordisk Foundation (no. NNF20OC0061894), the Villum Foundation (no. 25893), the National Natural Science Foundation of China (no. 62273287), the Shenzhen Science and Technology Innovation Council (no. JCYJ20220530143418040), and the Thousand Youth Talents Plan funded by the central government of China.

[1]Lujing Chen and Martin S. Andersen are with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark. Email: {lujc,mskan}@dtu.dk

[2]Tianshi Chen is with the School of Data Science and Shenzhen Research Institute of Big Data, The Chinese University of Hong Kong, Shenzhen, 518172, China. Email: tschen@cuhk.edu.cn

[3]Utkarsh Detha is with MOSEK ApS, Fruebjergvej 3, Symbion Science Park, 2100 Copenhagen, Denmark. Email: utkarsh.detha@mosek.com

where $\sigma^2 > 0$ is the noise variance and the covariance matrix $\nu K(\beta)$ associated with θ is parameterized by $\nu > 0$ and a vector of kernel parameters $\beta \in \mathbb{R}^d$. We will focus on the case where there is a small number of hyperparameters, which is true for kernels such as the DC, TC, and SS kernels.

From a Bayesian viewpoint, we have that

$$\begin{aligned} y|\theta, \sigma^2 &\sim \mathcal{N}(\Phi\theta, \sigma^2 I) \\ y|\sigma^2, \nu, \beta &\sim \mathcal{N}(0, \Sigma_y) \\ \theta|y, \sigma^2, \nu, \beta &\sim \mathcal{N}(\hat{\theta}, \Sigma_{\theta|y}) \end{aligned}$$

where $\hat{\theta} = \sigma^{-2}\Sigma_{\theta|y}\Phi^T y$ is the posterior mean, and the covariance matrices Σ_y and $\Sigma_{\theta|y}$ are defined as

$$\Sigma_y = \nu\left(\frac{\sigma^2}{\nu}I + \Phi K(\beta)\Phi^T\right)$$
$$\Sigma_{\theta|y} = \sigma^2\left(\Phi^T\Phi + \frac{\sigma^2}{\nu}K(\beta)^{-1}\right)^{-1}.$$

Letting $K(\beta) = LL^T$ be a Cholesky factorization of $K(\beta)$ and defining $\tilde{\Phi} = \Phi L$ and $\lambda = \sigma^2/\nu$, we can express the posterior mean $\hat{\theta}$ as

$$\begin{aligned} \hat{\theta} &= \underset{\theta}{\operatorname{argmin}}\left\{\|\Phi\theta - y\|_2^2 + \lambda\,\theta^T K(\beta)^{-1}\theta\right\} \\ &= L\ \underset{\tilde{\theta}}{\operatorname{argmin}}\left\{\|\tilde{\Phi}\tilde{\theta} - y\|_2^2 + \lambda\|\tilde{\theta}\|_2^2\right\} = L\tilde{\theta}^\star, \end{aligned} \tag{2}$$

where $\tilde{\theta}^\star = W_\eta^{-1}\tilde{\Phi}^T y = \tilde{\Phi}^T C_\eta^{-1} y$ with $\eta = (\lambda, \beta)$ and

$$W_\eta = \lambda I + \tilde{\Phi}^T\tilde{\Phi}, \quad C_\eta = \lambda I + \tilde{\Phi}\tilde{\Phi}^T. \tag{3}$$

It is straightforward to verify that $C_\eta = \nu^{-1}\Sigma_y$ and $\sigma^2\Sigma_{\theta|y}^{-1} = L^T W_\eta L$. The model output defined by the parameter estimate $\hat{\theta}$ can be expressed as

$$\hat{y} = \Phi\hat{\theta} = \tilde{\Phi}\tilde{\theta}^\star = H_\eta y \tag{4}$$

where $H_\eta = \tilde{\Phi}W_\eta^{-1}\tilde{\Phi}^T$ is the so-called influence matrix. Both $\hat{\theta}$ and $\hat{y}$ depend on the hyperparameters λ and β, which must somehow be selected.

B. Hyperparameter Selection

Next, we outline two popular hyperparameter selection criteria: generalized cross-validation (GCV) and marginal likelihood maximization.

1) Generalized cross-validation: The GCV approach to the hyperparameter selection problem is to minimize the criterion [10]

$$\mathrm{GCV}(\eta) = \frac{\frac{1}{m}\|(I - H_\eta)y\|_2^2}{(\mathrm{tr}(I - H_\eta)/m)^2}.$$

It follows from the Sherman–Morrison–Woodbury identity that $\mathrm{tr}(I - H_\eta) = \lambda\,\mathrm{tr}(C_\eta^{-1})$. Moreover, if we let $\tilde{\sigma}_1, \ldots, \tilde{\sigma}_r$ denote the $r = \min(m, n)$ singular values of $\tilde{\Phi}$, then

$$\begin{aligned} \mathrm{tr}(I - H_\eta) = m - \mathrm{tr}(H_\eta) &= m - \sum_{i=1}^r \frac{\tilde{\sigma}_i^2}{\lambda + \tilde{\sigma}_i^2} \\ &= m - \sum_{i=1}^r \frac{\lambda + \tilde{\sigma}_i^2}{\lambda + \tilde{\sigma}_i^2} + \sum_{i=1}^r \frac{\lambda}{\lambda + \tilde{\sigma}_i^2} \\ &= m - n + \lambda\,\mathrm{tr}(W_\eta^{-1}). \end{aligned}$$

This implies that we can evaluate the GCV criterion by computing $\hat{y}$ and either $\mathrm{tr}(H_\eta)$, $\mathrm{tr}(C_\eta^{-1})$, or $\mathrm{tr}(W_\eta^{-1})$. The quantity $\mathrm{d_{eff}}(\eta) = \mathrm{tr}(H_\eta)$ is referred to as the effective degrees of freedom. We will define $\psi_{\mathrm{GCV}}(\eta) = \ln \mathrm{GCV}(\eta)$, which can also be expressed as

$$\psi_{\mathrm{GCV}}(\eta) = \ln(\|y - \hat{y}\|_2^2) - 2\ln(\lambda\,\mathrm{tr}(C_\eta^{-1})) + \ln(m). \tag{5}$$

2) Marginal likelihood maximization: The marginal likelihood maximization approach amounts to minimizing $y^T\Sigma_y^{-1}y + \ln\det(\Sigma_y)$ with respect to η and ν, or, equivalently, minimizing

$$\mathrm{ML}(\eta, \nu) = \nu^{-1}y^T C_\eta^{-1}y - m\ln(\nu^{-1}) + \ln\det(C_\eta).$$

For a fixed η, this function is convex in ν^{-1}, so we may eliminate ν by setting the partial derivative with respect to ν to zero, which yields $\hat{\nu} = (y^T C_\eta^{-1}y)/m$ and leads to the PML objective

$$\begin{aligned} \psi_{\mathrm{PML}}(\eta) &= (1/m)\mathrm{ML}(\eta, \hat{\nu}) + \ln(m) - 1 \\ &= \ln(y^T C_\eta^{-1}y) + (1/m)\ln\det(C_\eta). \end{aligned} \tag{6}$$

This can also be expressed as

$$\begin{aligned} \psi_{\mathrm{PML}}(\eta) &= \ln(y^T(y - \hat{y})) \\ &\quad + \frac{1}{m}\ln\det(W_\eta) - \frac{n}{m}\ln(\lambda) \end{aligned} \tag{7}$$

which follows from the fact that $y - \hat{y} = \lambda C_\eta^{-1}y$ and the Weinstein–Aronszajn identity

$$\lambda^{-m}\det(C_\eta) = \lambda^{-n}\det(W_\eta).$$

The main benefit of using the PML objective instead of the ML objective is that the number of hyperparameters is reduced by one.

C. Direct Evaluation

To evaluate the GCV criterion (5) for a given η, we need to compute $\hat{y}$ and either $\mathrm{tr}(H_\eta)$, $\mathrm{tr}(C_\eta^{-1})$, or $\mathrm{tr}(W_\eta^{-1})$, and to evaluate PML criterion (6), we need $\hat{y}$ and either $\ln\det(C_\eta)$ or $\ln\det(W_\eta)$. We now outline how this can be done using a direct method, i.e., we explicitly form and factorize the matrix $\tilde{\Phi} = \Phi L$. We will consider *indirect* methods in the next section, i.e., methods that only rely on matrix-vector products with $\tilde{\Phi}$ and its transpose.

To simplify notation, we will assume that $\mathrm{rank}(\tilde{\Phi}) = n$ (and hence $m \geq n$) and that $\tilde{\Phi} = Q_1 R$ is a "thin" QR decomposition of $\tilde{\Phi}$. This can be computed in $O(mn^2)$ arithmetic operations. One way to proceed is to compute a factorization $W_\eta = \lambda I + R^T R = B^T B$ where B is upper triangular. This can be done using another QR factorization or a Cholesky factorization, both of which require $O(n^3)$ operations. We then have $\tilde{\theta}^\star = B^{-1}B^{-T}\tilde{\Phi}^T y$, $\hat{y} = \tilde{\Phi}\tilde{\theta}^\star$, and

$$\ln\det(W_\eta) = 2\sum_{i=1}^n \ln(B_{ii}), \quad \mathrm{tr}(H_\eta) = \|\tilde{\Phi}B^{-1}\|_{\mathrm{F}}^2,$$

where e_i denotes is the ith standard unit vector. It follows that the cost of evaluating the GCV and PML criteria as outlined above grows as $O(mn^2)$.

III. INDIRECT METHOD

To leverage both the Toeplitz structure of Φ and the structure of the kernel matrix $K(\beta)$, we propose an indirect method to compute $\hat{\theta}$ and evaluate ψ_{GCV} and/or ψ_{PML} using only a matrix-vector oracle. We will assume that an implicit factorization $K(\beta) = LL^T$ can be computed in linear time and that matrix-vector products with L and its transpose take $O(n)$ time. For example, this is true for the SS, TC, and DC kernels. Additionally, using the fast Fourier transform (FFT), matrix-vector products with Φ and $\tilde{\Phi}$, as well as their transposes, take $O((m+n)\ln(m+n))$ time. Before introducing two simple preconditioning methods to enhance performance, we start by considering the problem of evaluating ψ_{GCV} and/or ψ_{PML}.

A. Function Evaluation

The posterior mean $\hat{\theta}$ and the hyperparameter selection criteria ψ_{GCV} and ψ_{PML} can be evaluated in several ways. We first consider the problem of computing $\hat{y}$, which appears in the first term of (5) and (7), and then we consider the problem of evaluating or approximating $\mathrm{tr}(C_\eta^{-1})$ and $\ln\det(C_\eta)$ via stochastic trace estimation.

1) Computing $\hat{\theta}$ and $\hat{y}$: The posterior mean $\hat{\theta} = L\tilde{\theta}^\star$ can be computed by solving the regularized least-squares problem in (2), i.e., we need to solve $W_\eta\tilde{\theta}^\star = \tilde{\Phi}^T y$, which can be computed using an indirect method such as CG or LSQR. We note that analytically, LSQR and CG yield the same iterates, but in finite-precision, LSQR is generally preferred over CG for problems of the form (2) [17]. In both cases, a good preconditioner can often significantly improve the performance of iterative solvers such as CG and LSQR; we return to preconditioning techniques later in this section. Alternatively, we can solve $C_\eta z = y$ using CG or LSQR and compute $\tilde{\theta}^\star = \tilde{\Phi}^T z$. Finally, given $\tilde{\theta}^\star$, we can readily compute $\hat{y} = \tilde{\Phi}\tilde{\theta}^\star$.

2) Stochastic trace estimation: The trace of a matrix $A \in \mathbb{R}^{n\times n}$ can be expressed as $\mathrm{tr}(A) = \mathbb{E}[v^T A v]$ where v is a random variable such that $\mathbb{E}[vv^T] = I$. This identity underpins Hutchinson's trace estimator [12], which is a sample average approximation of the form

$$\hat{T}_N(A) = \frac{1}{N}\sum_{i=1}^{N} v_i^T A v_i,$$

where the vectors $v_1, \ldots, v_N$ are independent realizations of a Rademacher random variable. Note that the N terms in the sum can be computed in parallel. We observe that $\hat{T}_1(A) = \mathrm{tr}(A)$ if A is diagonal. More generally, $\hat{T}_N(A)$ is unbiased, and its variance is $O(1/N)$. We note that improved, variance-reduced versions exist, e.g., Hutch++ [14], and the more general problem of approximating $\mathrm{tr}(f(A))$ where f is a function of a symmetric matrix A can be tackled using series expansion or stochastic Lanczos quadrature methods; see, e.g., [4], [21], [20].

Stochastic trace estimation can be used to estimate the GCV criterion (5) by computing either $\hat{T}_N(H_\eta)$, $\hat{T}_N(C_\eta^{-1})$, or $\hat{T}_N(W_\eta^{-1})$. Products with H_η, C_η^{-1}, and W_η^{-1} can be evaluated using LSQR or CG, as outlined earlier in this section.

The problem of estimating $\ln\det(W_\eta)$, which appears in the PML criterion (7), can be cast as a trace estimation problem, which follows by noting that $\ln\det(W_\eta) = \mathrm{tr}(\ln(W_\eta))$. Using the approach from [4], we let $0 < \alpha < 2/\|C_\eta\|_2$ be a constant such that $\|I - \alpha W_\eta\|_2 < 1$ and use the Mercator series

$$-\ln(1-x) = \sum_{k=1}^{\infty}\frac{x^k}{k}, \quad |x| < 1,$$

to expand $\mathrm{tr}(\ln(W_\eta))$ as

$$\begin{aligned}\mathrm{tr}(\ln(W_\eta)) &= \mathrm{tr}(\ln(I - (I - \alpha W_\eta))) - m\ln(\alpha)\\ &= -\,\mathrm{tr}\left(\sum_{k=1}^{\infty}\frac{(I-\alpha W_\eta)^k}{k}\right) - m\ln(\alpha). \quad (8)\end{aligned}$$

The sum converges faster the smaller $\|I - \alpha W_\eta\|_2$ is, and the choice

$$\alpha^\star = \operatorname*{argmin}_{\alpha>0}\|I - \alpha W_\eta\|_2 = \frac{2}{\lambda_{\max}(W_\eta) + \lambda_{\min}(W_\eta)}$$

attains the minimum

$$\|I - \alpha^\star W_\eta\|_2 = \frac{\kappa(W_\eta) - 1}{\kappa(W_\eta) + 1},$$

where $\kappa(W_\eta)$ denotes the 2-norm condition number of W_η. This can be improved via preconditioning, which we address later in this section.

Algorithm 1 combines Hutchinson's method with the series expansion in (8) to compute an estimate of $\ln\det(W_\eta)$. The algorithm is closely related to that of [4] but uses only k matrix-vector products with W_η to compute $2k$ terms in the series. Moreover, rather than truncating the sum after a fixed number of terms, Algorithm 1 truncates the sum adaptively based on a user-defined tolerance.

Algorithm 1: Log-determinant estimation.

```
Data: $A \in \mathbb{S}^n$ such that $\|A\|_2 < 1$, $N > 0$, $\varepsilon > 0$.
Result: $s \approx \ln\det(I - A)$.
$s \leftarrow 0$
for $l \leftarrow 1$ to $N$ do
    $v \leftarrow$ Rademacher(n)/$\sqrt{n}$
    $t \leftarrow 0$, $k \leftarrow 1$
    do
        $\bar{v} \leftarrow v$, $v \leftarrow Av$
        $t \leftarrow t + (v^T\bar{v})/(2k-1) + (v^Tv)/(2k)$
        $k \leftarrow k + 1$
    while $v^Tv > 2k\varepsilon$
    $s \leftarrow s - t$
end
$s \leftarrow (n/N)s$
```

B. Preconditioning

It is well known that preconditioning can greatly improve the performance of iterative methods such as LSQR and CG, reducing the number of iterations required to obtain a given accuracy. When applied to the system of equations $W_\eta \tilde{\theta} = \tilde{\Phi}^T y$ with the initial vector $\tilde{\theta}_0$, the iterates $\tilde{\theta}_1, \tilde{\theta}_2, \dots$ produced by the CG method satisfy the worst-case bound [16]

$$\|\tilde{\theta}_k - \tilde{\theta}^\star\|_{W_\eta}^2 \leq 2\left(\frac{\sqrt{\kappa(W_\eta)}-1}{\sqrt{\kappa(W_\eta)}+1}\right)^k \|\tilde{\theta}_0 - \tilde{\theta}^\star\|_{W_\eta}^2.$$

The basic idea behind preconditioning is to apply a transformation to the original problem that makes it easier to solve while preserving the solution. Roughly speaking, we would like the preconditioner M to approximate W_η well, e.g., in the sense that $\kappa(M^{-1/2}W_\eta M^{-1/2}) \ll \kappa(W_\eta)$, and products with $M^{-1/2}$ and/or M^{-1} should be cheap to compute. Preconditioning can also dramatically improve the performance of Algorithm 1. Indeed, we can express $\ln\det(W_\eta)$ as

$$\ln\det(W_\eta) = \ln\det(M^{-1/2}W_\eta M^{-1/2}) + \ln\det(M),$$

and M should be chosen in such a way that $\ln\det(M)$ and products with $M^{-1/2}$ are cheap to evaluate. Alternatively, if we choose to work with C_η instead of W_η, we can instead construct a preconditioner for C_η.

Next, we outline two simple, yet powerful, methods that produce a preconditioner of the form $M_k = V(D+\lambda I)V^T + (\lambda+\delta)(I - VV^T)$ where $V \in \mathbb{R}^{n\times k}$ has orthonormal columns, D is diagonal and positive semidefinite, $k \ll n$, and $\delta \geq 0$. The first one is based on a truncated singular value decomposition (SVD) of $\tilde{\Phi}$, and the second one is based on a randomized Nyström approximation of $\tilde{\Phi}^T\tilde{\Phi}$ (or $\tilde{\Phi}\tilde{\Phi}^T$ if we wish to construct a preconditioner for C_η). The operation $x \leftarrow M_k^{-1/2}x$ can then be implemented as

$$\tilde{x} \leftarrow V^T x, \quad x \leftarrow V(\lambda I + D)^{-1/2}\tilde{x} + \frac{1}{\sqrt{\lambda+\delta}}(x - V\tilde{x}),$$

which costs $O(kn)$ operations.

1) Truncated SVD: A low-rank approximation of $\tilde{\Phi}$ can be obtained using a truncated SVD. Specifically, if $\tilde{\Phi} = USV^T$ is an SVD of $\tilde{\Phi}$ where S is the diagonal matrix with the singular values $\sigma_1 \geq \cdots \geq \sigma_{\min(m,n)}$, then the Eckart–Young–Mirsky theorem states a truncated SVD

$$\tilde{\Phi}_{[k]} = \sum_{i=1}^{k} \sigma_i u_i v_i^T = U_k S_k V_k^T, \quad 1 \leq k \leq \min(m,n),$$

is optimal in the sense that

$$\tilde{\Phi}_{[k]} \in \operatorname*{argmin}_{Z\in\mathbb{R}^{m\times n}} \left\{\|\tilde{\Phi} - Z\| \mid \operatorname{rank}(Z) \leq k\right\}.$$

Here $\|\cdot\|$ is either the Frobenius norm or the spectral norm. For a given $k < \min(m,n)$, we can construct a preconditioner M_k for W_η as

$$M_k = V_k(S_k^2 + \lambda I)V_k^T + (\lambda+\delta)(I - V_kV_k^T), \tag{9a}$$

where $\delta \geq 0$ is a constant. The condition number of the preconditioned matrix $\widetilde{W}_\eta = M_k^{-1/2}W_\eta M_k^{-1/2}$ satisfies

$$\kappa(\widetilde{W}_\eta) = \begin{cases} \frac{\sigma_{k+1}^2+\lambda}{\sigma_n^2+\lambda}, & m \geq n,\ \delta \in [\sigma_n^2, \sigma_{k+1}^2], \\ \frac{\sigma_{k+1}^2+\lambda}{\lambda}, & m < n,\ \delta \in [0, \sigma_{k+1}^2]. \end{cases}$$

The simple choice $\delta = 0$ leads to $\kappa(\widetilde{W}_\eta) = (\sigma_{k+1}^2 + \lambda)/\lambda$. This is much smaller than $\kappa(W_\eta)$ for $k \ll \min(m,n)$ if the singular values of $\tilde{\Phi}$ decay quickly and $\lambda \gtrsim \sigma_r^2$ where σ_r denotes is the smallest nonzeros singular value of $\tilde{\Phi}$. We note that the leading k singular values of $\tilde{\Phi}$ and the corresponding left and/or right singular vectors can be computed using the Arnoldi iteration [3], which only requires matrix-vector products. This is implemented in, e.g., the `svds` routine in MATLAB and Python/SciPy.

2) Randomized Nyström approximation: Randomized techniques such as randomized Nyström approximation [7], [8] is a fast alternative to the truncated SVD. To construct a preconditioner for W_η, we first compute $Y = \tilde{\Phi}^T\tilde{\Phi}\Omega$ where $\Omega \in \mathbb{R}^{n\times k}$ is a random matrix, e.g., with independent standard normal entries. The rank k matrix $Y(Y^T\Omega)^\dagger Y^T$ is then a randomized Nyström approximation of $\tilde{\Phi}^T\tilde{\Phi}$. Computing an eigenvalue decomposition,

$$Y(Y^T\Omega)^\dagger Y^T = \hat{V}_k\hat{D}_k\hat{V}_k^T,$$

which costs $O(nk^2)$, we construct a randomized Nyström preconditioner as

$$\hat{M}_k = \hat{V}_k(\hat{D}_k + \lambda I)\hat{V}_k^T + (\lambda+\delta)(I - \hat{V}_k\hat{V}_k^T).$$

Assuming that $m \geq n$ and $\delta \in [\sigma_n^2, \sigma_{k+1}^2]$, it can be shown that [8, Thm. 5.1]

$$\mathbb{E}[\kappa(\hat{M}_k^{-1/2}W_\eta\hat{M}_k^{-1/2})] < 28$$

if $k = 2\lceil 1.5\,\mathrm{d_{eff}}(\eta)\rceil$. However, the effective degrees of freedom are not known in practice (the preconditioner is used to estimate $\mathrm{d_{eff}}(\eta) = \operatorname{tr}(H_\eta)$), so a more practical approach is to choose k as large as one can afford. Finally, we note that the same procedure can be used to construct a randomized Nyström preconditioner $\hat{P}_k$ for C_η.

C. Optimization

The hyperparameter criteria ψ_{GCV} and ψ_{PML} can be minimized locally using, e.g., derivative-unaware Bayesian optimization [9] or a derivative-free method such as the Nelder–Mead method [15], [13], both of which rely only on function evaluations. On one hand, the Nelder–Mead method is conceptually simple and has minimal overhead, but it can be sensitive to noise when the objective function is stochastic. On the other hand, Bayesian optimization is more robust to noise but has a higher computational overhead.

First-order or second-order methods that rely on partial derivatives can also be used, but the indirect evaluation or estimation of a partial derivative costs as much as a function evaluation. The constraint $\lambda > 0$ can be handled using a change of variables: we express λ as $\lambda = 10^{-\rho}$ and define our optimization variable as $\tilde{\eta} = (\rho, \beta)$. An added benefit of this change of variables is that it is generally more compatible with the default step sizes and stopping criteria used in off-the-shelf optimizers. A similar transformation can be applied to β, e.g., if one or more kernel parameters are more naturally explored on a logarithmic scale or if anticipated hyperparameter ranges differ by orders of magnitude.

IV. Numerical Simulation

A. Implementation

The main steps involved in estimating $\psi_{\text{GCV}}(\eta)$ and/or $\psi_{\text{PML}}(\eta)$ are as follows:

1) Factorize $K(\beta) = LL^T$.
2) Compute randomized Nyström preconditioner $\hat{M}_k$.
3) Compute $\hat{\theta}$ and $\hat{y}$ using preconditioned LSQR.
4) Estimate $\operatorname{tr}(H_\eta)$ using LSQR with Hutchinson's method (GCV) or estimate $\ln\det(\hat{M}_k^{-1/2} W_\eta \hat{M}_k^{-1/2})$ using Algorithm 1 (PML).

We illustrate the methodology using the SS kernel and the implicit $O(n)$ kernel matrix factorization proposed in [2]. The SS kernel can be expressed as

$$K_2^{\text{ss}}(s,t) = \begin{cases} e^{-2\beta s}(e^{-\beta t}/2 - e^{-\beta s}/6), & s \ge t \ge 0, \\ e^{-2\beta t}(e^{-\beta s}/2 - e^{-\beta t}/6), & t \ge s \ge 0, \end{cases}$$

i.e., it is parameterized by a scalar parameter $\beta > 0$. We note that kernels such as the TC and DC kernels possess a similar structure that can readily be exploited in the proposed framework. Our preliminary MATLAB implementation makes use of the FFT to compute products with Φ and its transpose, and we use the EGRSS package [1] to factorize $K(\beta)$ and to compute products with L and L^T. We note that Φ and L, which are stored implicitly, require only $O(m+n)$ memory. The implementation can be improved in several ways, e.g., by caching intermediate FFTs that are used repeatedly and/or by parallelizing the stochastic trace estimation.

The following algorithmic parameters are used in our numerical experiments: In step 2), we use the algorithm from [8] to compute a randomized Nyström preconditioner with $k = 150$ and $\delta = 0$. In step 3), we use MATLAB's built-in `lsqr` routine with the tolerance 10^{-5}. In step 4), we use Algorithm 1 with $N = 50$ and the tolerance $\varepsilon = 10^{-4}$. Moreover, as an approximation to $\alpha^\star = \operatorname{argmin}_{\alpha>0} \|I - \alpha\hat{M}_k^{-1/2} W_\eta \hat{M}_k^{-1/2}\|_2$, we use $\hat{\alpha} = 1.8/(1+\hat{\zeta})$ where $\hat{\zeta} \approx \|\hat{M}_k^{-1/2} W_\eta \hat{M}_k^{-1/2}\|_2$ is estimated using power iteration.

For the hyperparameter optimization, we use MATLAB's `fminsearch` with tolerances `TolX` and `TolFun` set to 0.1 and a change of variables $\tilde{\eta} = (\rho, \tilde{\beta})$ where $\lambda = 10^{-\rho}$ and $\tilde{\beta} = 10^3\beta$. We note that more sophisticated derivative-free optimization methods exist, but `fminsearch` is sufficient for the purpose of our MATLAB-based proof of concept.

B. Test data

To evaluate the indirect approach to the hyperparameter estimation problem, we generated a data bank with ten discrete-time systems (S1-S10) following the procedure outlined in [6]: first, we randomly generated 10 SISO continuous-time systems of order 50 using `rss` in MATLAB, and each of these systems was then discretized by sampling at 3 times the bandwidth of the system. For each system, we generated test data by simulating the system using unit variance Gaussian white noise $u \in \mathbb{R}^m$ with $m = 10^4$ as input, and we added Gaussian white noise $e \in \mathbb{R}^m$ to the output of the simulation $\bar{y} \in \mathbb{R}^m$ to generate a noisy output observation $y = \bar{y} + e$. The variance of the noise e was chosen such that $\operatorname{Var}(e) = 0.1 \cdot \operatorname{Var}(\bar{y})$.

C. Simulation results

We report our numerical results from two experiments based on the PML criterion. The first experiment assesses the precision of stochastic function evaluations using the indirect method whereas the second experiment compares runtimes and fits obtained using MATLAB's `impulseest` and the proposed direct and indirect methods for different values of the FIR order n. The experiments were conducted in MATLAB R2022b on a MacBook Pro with a 2.3 GHz Quad-Core Intel Core i7 CPU and 16 GB of memory.

1) Function evaluation: To assess the precision of the PML criterion evaluations using the indirect method, we analyzed 144 hyperparameter pairs $\tilde{\eta} = (\rho, \tilde{\beta})$ on a regular grid where ρ ranges from 1 to 6 and $\tilde{\beta}$ ranges from 1 to 12. We estimated the mean relative error of the PML objective using the direct method as the reference. We performed this analysis for all ten systems based on 12 function evaluations for each hyperparameter pair with the FIR order $n = 10^3$ and using the algorithmic parameters described at the beginning of this section.

Our results showed that the mean relative error was consistently below $3.6 \cdot 10^{-4}$ for every hyperparameter pair and all ten systems. This level of precision is orders of magnitude smaller than the `fminsearch` tolerances used in our experiments. Overall, these findings support the reliability and robustness of the methodology.

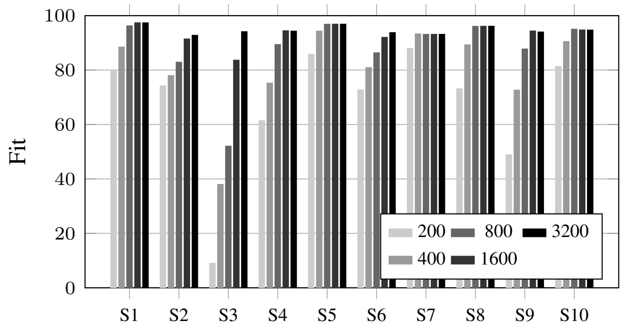

Fig. 1. Fits obtained via the PML criterion for each of the ten systems and different FIR orders.

2) Runtime and fit: To validate the proposed methodology and to evaluate the runtime as a function of n, we applied it to the ten systems from our data bank. To assess the quality of an estimate $\hat{\theta} \in \mathbb{R}^n$, we compute the model fit, defined as

$$\operatorname{Fit}(\hat{\theta}_{\text{e}}) = 100 \cdot \left[1 - \frac{\|\theta_{\text{e}} - \hat{\theta}_{\text{e}}\|_2}{\|\theta_{\text{e}} - \mu\mathbf{1}\|_2}\right], \quad \mu = \frac{1}{n_{\text{e}}}\mathbf{1}^T\theta_{\text{e}},$$

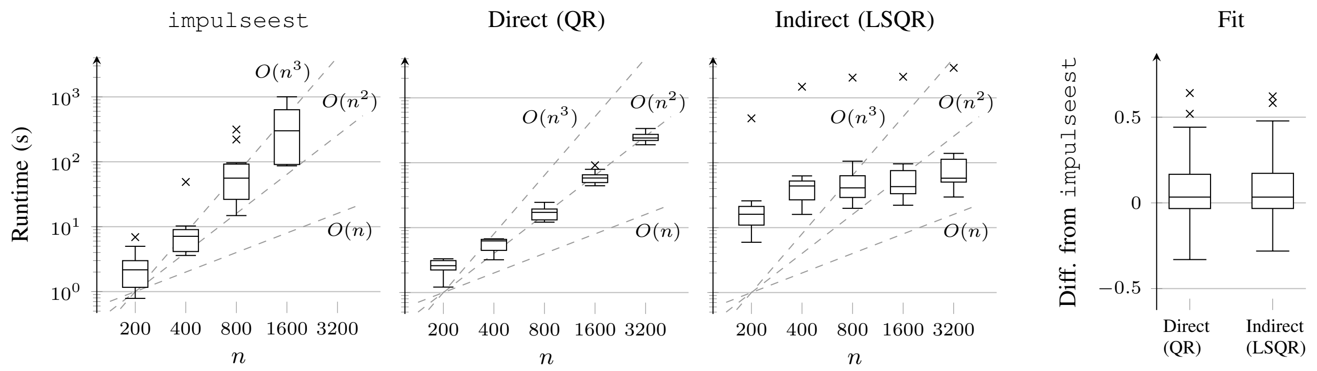

Fig. 2. Comparison of runtime and fit for the PML criterion: the runtime for both MATLAB's `impulseest` and our direct method grows quadratically with the FIR order n. In contrast, we observe that the runtime for the indirect method exhibits sublinear growth. The computation time for `impulseest` at $n = 3200$ was prohibitively long, and therefore no results were obtained for this parameter value. The right-most boxplot, which summarizes the difference in fit from the `impulseest` fit to those obtained with the direct/indirect method for n up to 1600, shows that our implementation predominantly finds hyperparameters that match or outperform those obtained with `impulseest` in terms of fit.

where $\theta_\mathrm{e} \in \mathbb{R}^{n_\mathrm{e}}$ is the ground truth impulse response truncated at length $n_\mathrm{e} = 10^4$, and $\hat{\theta}_\mathrm{e} \in \mathbb{R}^{n_\mathrm{e}}$ is the estimated impulse responses extended to length n_e.

Fig. 1 shows the model fit for each of the ten systems based on FIR orders ranging from 200 to 3200. Nearly identical fits were obtained with the direct and indirect methods. Observe that the fit increases with the FIR order n until the order is sufficiently high, and for most systems, the fit remains below 90 until the FIR order is relatively high ($n \geq 800$).

Fig. 2 shows boxplots of the runtime for the PML hyperparameter estimation with `impulseest` and the direct and indirect methods for FIR orders ranging from 200 to 3200. The runtime was measured using the `tic` and `toc` routines in MATLAB, and the reported runtime for the indirect method is the mean runtime based on five runs. The results are grouped by the FIR order n, and every group is based on results from all ten systems. For small FIR orders ($n < 800$), `impulseest` and the direct method are faster than our preliminary implementation of the indirect method. However, the runtime for these methods grows at least quadratically. In contrast, the runtime for the indirect method exhibits sublinear growth. We note that the outliers in the boxplot for the indirect method all originate from the same system, which has a relatively large bandwidth. The runtime for this system drops significantly if a larger rank parameter is used for the preconditioner. We will investigate different strategies for choosing this automatically in future work.

The kernel-regularized method is generally much more expensive than simply computing an ordinary least-square (OLS) fit of the form $\hat{\theta}_\mathrm{ls} = \Phi^\dagger y$ where $\Phi^\dagger$ denotes the Moore–Penrose pseudo-inverse of Φ. However, the OLS approach leads to a significantly worse fit in all our experiments (S1-S10): with the FIR order $n = 1600$, the OLS fit is at least 7.5 below that obtained with the kernel-regularized method, and for $n = 3200$, the difference is at least 16.6.

Finally, we would like to mention that we obtained comparable results to those reported in this study when using, e.g., the GCV criterion, the TC kernel, the truncated SVD preconditioner, and/or different data sizes m. However, due to space constraints, we have omitted these results from our presentation.

V. CONCLUSION

We have proposed a methodology for kernel-based regularized system identification based on a matrix-free, indirect solution approach. It relies on stochastic techniques and uses derivative-free optimization for efficient hyperparameter estimation. Our preliminary results demonstrate that the indirect approach scales better than existing methods such as MATLAB's `impulseest`, allowing us to speed up hyperparameter estimation with large FIR orders.

The practical performance of the approach is greatly enhanced by the use of an appropriate preconditioner, as highlighted by our numerical study. To further improve the methodology, future research could investigate a hybrid approach that utilizes a direct method with a low FIR order to quickly establish a suitable initialization and preconditioner for the indirect method with a larger FIR order. Moreover, the pursuit of scalable and robust hyperparameter estimation methods could also include research into the balance between function evaluation speed and precision, e.g., in the context of Bayesian optimization.

REFERENCES

[1] M. S. Andersen. EGRSS v0.1.0 (GitHub), 2023.

[2] M. S. Andersen and T. Chen. Smoothing splines and rank structured matrices: Revisiting the spline kernel. *SIAM Journal on Matrix Analysis and Applications*, 41(2):389–412, January 2020.

[3] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9(1):17–29, 1951.

[4] C. Boutsidis, P. Drineas, P. Kambadur, E.-M. Kontopoulou, and A. Zouzias. A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. *Linear Algebra and its Applications*, 533:95–117, November 2017.

[5] F. P. Carli. On the maximum entropy property of the first-order stable spline kernel and its implications. In *2014 IEEE Conference on Control Applications (CCA)*. IEEE, October 2014.

[6] T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes—revisited. *Automatica*, 48(8):1525–1535, August 2012.

[7] P. Drineas and M. W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6(72):2153–2175, 2005.

[8] Z. Frangella, J. A. Tropp, and M. Udell. Randomized Nyström preconditioning. arXiv:2110.02820v2, 2021.

[9] R. Garnett. *Bayesian Optimization*. Cambridge University Press, Cambridge, England, February 2023.

[10] G. H. Golub, M. Heath, and G. Wahba. Generalized Cross-Validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, May 1979.

[11] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409, December 1952.

[12] M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics - Simulation and Computation*, 19(2):433–450, January 1990.

[13] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder–Mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9(1):112–147, January 1998.

[14] R. A. Meyer, C. Musco, C. Musco, and D. P. Woodruff. Hutch++: Optimal stochastic trace estimation. In *Symposium on Simplicity in Algorithms (SOSA)*, pages 142–155. Society for Industrial and Applied Mathematics, January 2021.

[15] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, January 1965.

[16] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer New York, 2006.

[17] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, March 1982.

[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*, 50(3):657–682, March 2014.

[19] G. Pillonetto and G. De Nicolao. A new kernel-based approach for linear system identification. *Automatica*, 46(1):81–93, January 2010.

[20] A. K. Saibaba, A. Alexanderian, and O. C. F. Ipsen. Randomized matrix-free trace and log-determinant estimators. *Numerische Mathematik*, 137(2):353–395, April 2017.

[21] S. Ubaru, J. Chen, and Y. Saad. Fast estimation of $\mathrm{tr}(f(a))$ via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, January 2017.

[22] J. Wenger, G. Pleiss, P. Hennig, J. Cunningham, and J. Gardner. Preconditioning for scalable Gaussian process hyperparameter optimization. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23751–23780. PMLR, July 2022.