Data-Driven Linear Parameter-Varying Model Identification Using Transfer Learning

Yajie Bao, *Graduate Student Member, IEEE*, and Javad Mohammadpour Velni, *Member, IEEE*

***Abstract*—This letter proposes transfer learning methods to address a challenge in state-space linear parameter-varying (LPV-SS) model identification/learning using kernelized machine learning, when the distributions of the training and testing sets are different. Kernel mean matching is first employed to correct sample bias by resampling the data in the training set before the states in state-space model are estimated. Moreover, transfer component analysis is adopted to find a state-space basis transformation such that the transformed states follow similar distributions. The proposed methods are validated by testing on an ideal continuous stirred tank reactor (CSTR) model. Simulation results show that the proposed learning methods can enhance the accuracy of model identification and reduce the efforts involved in hyperparameters tuning.**

***Index Terms*—Nonparametric identification, linear parameter-varying models, kernels, transfer learning.**

I. Introduction

DATA-DRIVEN methods have been shown to provide accurate and low-complexity state-space linear parameter-varying (LPV-SS) models of nonlinear systems for observer and controller design purposes [1]. Experimental results show that these methods work well but under the assumption that training and future data share the same feature space and distribution [1], [2]. However, this assumption can be violated when applying the identified models for control, due to the differences between the training and application environments. Since it is time consuming and cost intensive to collect necessary training data and rebuild models for each application environment, *transfer learning* has proven effective to provide a feasible approach to employ previously learned models to facilitate the model identification of a similar but different (not-identical) environment.

For global identification of LPV-SS models using input/output data, existing methods can be categorized into parametric and non-parametric methods. Parametric methods assume that scheduling dependencies of the model coefficients are known *a priori* [3] while non-parametric methods provide a reconstruction of the scheduling dependencies without an explicit declaration of these often unknown dependencies [4]. Moreover, parametric methods require an appropriate selection of basis functions [4] while non-parametric methods require the selection of nonlinear kernel functions and the tuning of the associated hyperparameters [5]. Furthermore, most existing parametric methods assume an affine scheduling dependency with predefined basis functions, which restricts the complexity of a representation [1]. Examples of parametric LPV-SS identification include direct prediction-error minimization (PEM) methods and global subspace and realization-based techniques (SID) (see [6] and references therein). In particular, the nonlinear optimization of the PEM methods depends heavily on a proper initial seeding while SID methods suffer from the curse of dimensionality [1]. For non-parametric methods, authors in [5] used kernelized canonical correlation analysis (CCA) to estimate state sequence and a least-squares support vector machine (LS-SVM) to capture the dependency structure, which presents an attractive bias–variance trade-off. In this letter, we build on the kernelized methods in [5] using transfer learning for LPV-SS model identification.

Transfer learning is a machine learning technique that aims at applying knowledge learned from previous tasks (a.k.a. source tasks) to new tasks (a.k.a. target tasks), and has been extensively studied in machine learning community (see [7] for a comprehensive survey). The previous knowledge can be represented by reusable instances, feature representations, parameters and relational knowledge [2]. One approach for transfer learning using kernelized methods is to use predefined kernel functions and find a latent subspace where the distribution discrepancy between source and target domains is small, the property of target domain is maintained, and the accuracy on labeled data is maximized. Authors in [8] proposed transfer component analysis (TCA) which matches the distribution of source and target domains by minimizing the maximum mean discrepancy (MMD) and preserves the locality by the manifold regularizer.

Contribution of this letter is to develop two transfer learning approaches for LPV-SS model identification: sample bias correction and latent space learning. Sample bias arises when the environment for collecting data changes. For example, operating conditions of a system change such that the scheduling variables (of underlying LPV model) run into a range with a distribution that is different from the data used to build the model. This difference can affect the model accuracy since the optimization problems in the underlying learning algorithms are typically solved over training data. To tackle this problem, we propose to resample the training data before estimating

Manuscript received September 14, 2020; revised November 8, 2020; accepted November 23, 2020. Date of publication November 30, 2020; date of current version December 22, 2020. This work was supported by the United States National Science Foundation under Award #1762595 and Award #1912757. Recommended by Senior Editor G. Cherubini. *(Corresponding author: Yajie Bao.)*

The authors are with the School of Electrical and Computer Engineering, University of Georgia, Athens, GA 30602 USA (e-mail: yajie.bao@uga.edu; javadm@uga.edu).

Digital Object Identifier 10.1109/LCSYS.2020.3041407

the state sequence and use the kernel mean matching (KMM) approach in [9] to estimate the resampling weights directly from data without density estimation. However, when the testing set is collected from a similar but not-identical system, the difference between training and testing sets cannot be explained away by sample bias. We assume there is a latent space where the transformed states from training and testing sets follow similar distributions and adopt transfer component analysis (TCA) in [8] to learn a transformation from the original state space to the latent space. To the best of the authors' knowledge, this letter presents the first attempt to explore transfer learning methods for model identification of nonlinear systems (in LPV-SS setting) using kernelized machine learning methods.

The rest of this letter is organized as follows. Section II gives the problem statement and introduces the state and matrix function estimation using kernelized machine learning methods. Transfer learning approaches, including kernel mean matching and transfer component analysis, are introduced in Section III. Section IV presents our experiments on two "similar but not-identical" continuous stirred tank reactor (CSTR) models to evaluate the performance of the proposed methods. Concluding remarks are finally provided in Section V.

II. Problem Statement and Related Preliminaries

In machine learning, *domain* $\mathcal{D} = \{\mathcal{X}, P_X(x)\}$[1] is used to describe the input space $\mathcal{X}$ and the associated distribution P_X on the dataset; *task* $\mathcal{T} = \{\mathcal{Y}, h\}$ consists of the output space $\mathcal{Y}$ and the mathematical model $h : \mathcal{X} \to \mathcal{Y}$ to approximate an oracle that knows the correct answers to all questions. Model h can be a deterministic function $h : y = h(x)$ or a distribution $P_{XY}(x, y)$. In this letter, we use *environment* to refer to the oracle. Environment changes when one system changes operating conditions or switches to another system.

Assuming that we have an identified model (in this letter, an LPV-SS model) $\mathcal{M}_1$ of a dynamic system $\mathcal{P}_1$ using a large dataset $\mathcal{D}_1$ collected from $\mathcal{P}_1$, the main goal of this letter is to adapt the model $\mathcal{M}_1$ to obtain a good model $\mathcal{M}_2$ for a similar but different environment $\mathcal{P}_2$ using a small dataset $\mathcal{D}_2$ collected from $\mathcal{P}_2$. It is assumed that $\mathcal{D}_2$ is not sufficient to train a good model for $\mathcal{P}_2$. Additionally, we assume that the new system $\mathcal{P}_2$ and $\mathcal{P}_1$ are similar but have different parameters. Due to the differences in the parameters of these two environments, the distributions of $\mathcal{D}_1$ and $\mathcal{D}_2$ are different, which can cause performance degradation of the previous model $\mathcal{M}_1$ on the data $\mathcal{D}_2$ of the new environment $\mathcal{P}_2$. In the machine learning literature, this problem is known as *transductive learning* or domain adaptation where the domain of source task $\mathcal{D}_S$ and that of target task $\mathcal{D}_T$ are different but related while $\mathcal{Y}_S = \mathcal{Y}_T$ [2].

A. Formulation of the LPV-SS Model Identification Problem

A discrete-time LPV-SS model with innovation-type noise can be expressed as

$$\begin{aligned} x_{k+1} &= A(p_k)x_k + B(p_k)u_k + K(p_k)e_k, \\ y_k &= C(p_k)x_k + D(p_k)u_k + e_k, \end{aligned} \tag{1}$$

[1]We use $\mathcal{X}$, X, x and $P_X(x)$ to respectively denote the space, the variable, the sample and the distribution, and hence $x \in X \subseteq \mathcal{X}$.

where $p_k \in \mathbb{P} \subset \mathbb{R}^{n_p}$, $u_k \in \mathbb{R}^{n_u}$, $x_k \in \mathbb{R}^{n_x}$, $e_k \in \mathbb{R}^{n_y}$, and $y_k \in \mathbb{R}^{n_y}$ denote the scheduling variables, inputs, states, stochastic white noise process, and output measurements of the system at time instant $k \in \mathbb{Z}$,[2] respectively, and A, B, C, D, and K are smooth matrix functions of p_k. The LPV-SS representation (1) can be transformed into

$$\begin{aligned} x_{k+1} &= \tilde{A}(p_k)x_k + \tilde{B}(p_k)u_k + K(p_k)y_k, \\ y_k &= C(p_k)x_k + D(p_k)u_k + e_k, \end{aligned} \tag{2}$$

where $\tilde{A}(p_k) = A(p_k) - K(p_k)C(p_k)$ and $\tilde{B}(p_k) = B(p_k) - K(p_k)D(p_k)$. We note that (2) must be asymptotically stable in the deterministic sense for identification of (1) [5]. The LPV-SS model identification problem is to estimate states x_k, as well as the matrix functions $\tilde{A}(p_k)$, $\tilde{B}(p_k)$, $C(p_k)$, $D(p_k)$ and $K(p_k)$ given the measurements $\mathcal{D} = \{u_k, y_k, p_k\}_{k=1}^N$.

B. State Estimation in LPV-SS Identification Using KCCA

Considering that states are the interface between the past and future behavior of a system, the authors in [5] show that a state sequence that is compatible with the system can be estimated by determining the maximum correlation (using canonical correlation analysis (CCA)) between $\varphi_{\mathrm{p}}(\bar{p}_k^d)\bar{z}_k^d$ and $\varphi_{\mathrm{f}}(\bar{p}_{k+d}^d)\bar{z}_{k+d}^d$, where φ_{p} and φ_{f} represent the past and future state maps, $\bar{p}_k^d := \begin{bmatrix} p_{k-d}^{\mathrm{T}} & \cdots & p_{k-1}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ and $\bar{p}_{k+d}^d := \begin{bmatrix} p_k^{\mathrm{T}} & \cdots & p_{k+d-1}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ denote past and future scheduling variables, and $\bar{z}_k^d := \begin{bmatrix} \bar{u}_k^{d\mathrm{T}} & \bar{y}_k^{d\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ and $\bar{z}_{k+d}^d := \begin{bmatrix} \bar{u}_{k+d}^{d\mathrm{T}} & \bar{y}_{k+d}^{d\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ concatenate past and future inputs and outputs which are denoted similarly to $\bar{p}_k^d$ and $\bar{p}_{k+d}^d$. Since φ_{p} and φ_{f} are unknown nonlinear dynamic functions of p_k, [5] used kernelized CCA to estimate states by solving

$$\begin{aligned} \max_{v,w} \mathcal{J}(v, w, s, r) = \gamma \sum_{k=1}^{N} \left(s_k r_k - v_{\mathrm{f}} \frac{1}{2} s_k^2 - v_{\mathrm{p}} \frac{1}{2} r_k^2 \right) \\ - \frac{1}{2} v^{\mathrm{T}} v - \frac{1}{2} w^{\mathrm{T}} w \\ \text{s.t.} \quad s_k = v^{\mathrm{T}} \varphi_{\mathrm{f}}(\bar{p}_{k+d}^d)\bar{z}_{k+d}^d, k = 1, \ldots, N, \\ r_k = w^{\mathrm{T}} \varphi_{\mathrm{p}}(\bar{p}_k^d)\bar{z}_k^d, k = 1, \ldots, N, \end{aligned}$$

where γ, v_{p}, and v_{f} are hyperparameters. This problem can be simplified to a regularized generalized eigenvalue problem and solved via the following economical singular value decomposition (SVD):

$$\begin{aligned} &\left[\begin{array}{c:c} v_{\mathrm{f}} K_{\mathrm{ff}} + I & 0 \\ \hdashline 0 & v_{\mathrm{p}} K_{\mathrm{pp}} + I \end{array}\right]^{-1} \left[\begin{array}{c:c} 0 & K_{\mathrm{pp}} \\ \hdashline K_{\mathrm{ff}} & 0 \end{array}\right] \\ &\quad = W \Sigma \left[\begin{array}{c} V_1 \\ \hdashline V_2 \end{array}\right]^{\mathrm{T}}. \end{aligned} \tag{3}$$

We note that $\eta = [V_1]_1$ and $\kappa = [V_2]_1$ are Lagrange multipliers of the dual problem with $[\cdot]_j$ denoting the j-th column. In (3), $[K_{\mathrm{pp}}]_{l,m} = \bar{z}_l^{d\mathrm{T}} \bar{k}(\bar{p}_l^d, \bar{p}_m^d)\bar{z}_m^d$, $[K_{\mathrm{ff}}]_{l,m} = \bar{z}_{l+d}^{d\mathrm{T}} \bar{k}(\bar{p}_{l+d}^d, \bar{p}_{m+d}^d)\bar{z}_{m+d}^d$ where $\bar{k}$ is a kernel function, and I is the identity matrix. Then, we can obtain the state estimate as $\hat{x}_k = \mathrm{K}^{\mathrm{T}} \left[\bar{k}(\bar{p}_k^d, \bar{p}_1^d)\bar{z}_1^d \cdots \bar{k}(\bar{p}_k^d, \bar{p}_N^d)\bar{z}_N^d \right]^{\mathrm{T}} \bar{z}_k^d$ where $\mathrm{K} = [V_2]_{1:n_x}$ consists of the first n_x columns of V_2.

[2]$\mathbb{Z}$ denotes the set of integers.

C. Matrix Function Estimation Using LS-SVM

With the estimated state sequence $\{\hat{x}_k\}_{k=1}^N$ and using the extended dataset $\breve{\mathcal{D}} = \{u_k, y_k, \hat{x}_k, p_k\}_{k=1}^N$, the matrix functions can be estimated by solving the following LS-SVM problem[3] [5]:

$$\begin{aligned}\min_{W_x, W_y, \epsilon, \zeta} \quad & \mathcal{I}(W_x, W_y, \epsilon, \zeta) = \frac{1}{2}\Big(\|W_x\|_F^2 + \|W_y\|_F^2 \\ & + \sum_{k=1}^{N} \epsilon_k^{\mathrm{T}} \Gamma \epsilon_k + \zeta_k^{\mathrm{T}} \Xi \zeta_k\Big) \\ \text{s.t.} \quad & \epsilon_k = \hat{x}_k - W_x \varphi_x^{\mathrm{T}}(p_k), \\ & \zeta_k = \hat{y}_k - W_y \varphi_y^{\mathrm{T}}(p_k),\end{aligned}$$

where $\varphi_x(p_k)^{\mathrm{T}} := \left[(\Phi_{\tilde{A}}(p_k)\hat{x}_k)^{\mathrm{T}} \quad (\Phi_{\tilde{B}}(p_k)u_k)^{\mathrm{T}} \quad (\Phi_K(p_k)y_k)^{\mathrm{T}}\right]^{\mathrm{T}}$ and $\varphi_y(p_k)^{\mathrm{T}} := \left[(\Phi_C(p_k)\hat{x}_k)^{\mathrm{T}} \quad (\Phi_D(p_k)u_k)^{\mathrm{T}}\right]^{\mathrm{T}}$ with $\Phi_i(\cdot), i = \tilde{A}, \tilde{B}, K, C, D$, being implicit feature maps induced by the kernel functions $\bar{k}_i$, which represent the matrix functions. Polynomial kernels and radial basis functions (RBF) are the commonly used ones. Assuming that α_j and β_j are the Lagrange multipliers associated with the Lagrangian function of the LS-SVM problem at time j, KKT conditions are then used to compute the optimal α_j^* and β_j^*. Finally, the estimation of the matrix functions can be calculated by $\tilde{A}_e(\cdot) = \sum_{j=1}^N \alpha_j^* \hat{x}_j^{\mathrm{T}} \bar{k}_{\tilde{A}}(p_j, \cdot)$, $\tilde{B}_e(\cdot) = \sum_{j=1}^N \alpha_j^* u_j^{\mathrm{T}} \bar{k}_{\tilde{B}}(p_j, \cdot)$, $K_e(\cdot) = \sum_{j=1}^N \alpha_j^* y_j^{\mathrm{T}} \bar{k}_K(p_j, \cdot)$, $C_e(\cdot) = \sum_{j=1}^N \beta_j^* \hat{x}_j^{\mathrm{T}} \bar{k}_C(p_j, \cdot)$, and $D_e(\cdot) = \sum_{j=1}^N \beta_j^* u_j^{\mathrm{T}} \bar{k}_D(p_j, \cdot)$.

III. Transfer Learning for LPV-SS Model Identification

In this section, we introduce two approaches of transfer learning for system identification from the perspectives of instances and features. In particular, we consider the impact of the difference between source and target tasks on state estimation and use sample bias correction and common latent space learning to minimize such a difference for transfer.

A. Sample Bias Correction

First, we consider the scenario where the distributions of the scheduling variable are different, i.e., $P_{\mathbb{P}_T} \neq P_{\mathbb{P}_S}$. $P_{\mathbb{P}}$ affects state estimation through K_{pp} and K_{ff} in Eq. (3). To avoid the negative impact of the distribution discrepancy (generally called sample bias), one approach is to resample $\mathcal{D}_S$ such that the empirical distributions follow $\hat{P}_{\mathcal{D}_S} \approx \hat{P}_{\mathcal{D}_T}$. The ideal resampling weights are $\beta = \frac{P_{\mathcal{D}_T}}{P_{\mathcal{D}_S}}$. However, estimating both distributions to compute β is overkill. Instead, kernel mean matching (KMM) [9] directly estimates β from data by solving the following quadratic problem:

$$\min_{\beta} \quad \frac{1}{2}\beta^{\mathrm{T}} K_{S,S}\beta - K_S^{\mathrm{T}}\beta \tag{4}$$

$$\text{s.t.} \quad \left|\frac{1}{n_S}\sum_{i=1}^{n_S}\beta_i - 1\right| \leq \epsilon, \tag{5}$$

$$\beta_i \in [0, B], \quad i = 1, \ldots, n_S \tag{6}$$

[3]Here, the scheduling dependency is restricted to be static as in (2) for the sake of simplicity.

where β_i denotes the i-th entry of vector β, $K_{S,S}$ is a kernel matrix, the (i,j)-th entry of which is $[K_{S,S}]_{i,j} = \bar{k}(\mathrm{x}_S^{(i)}, \mathrm{x}_S^{(j)})$ and $[K_S]_i = \frac{n_S}{n_T}\sum_{j=1}^{n_T}\bar{k}(\mathrm{x}_S^{(i)}, \mathrm{x}_T^{(j)})$. Furthermore, $\mathrm{x}_S^{(i)}$ represents the i-th sample in the source domain while $\mathrm{x}_T^{(j)}$ denotes the j-th sample in the target domain. Constraint (6) is to limit the scope of the distribution discrepancy and the influence of individual observations and (5) to ensure that $\beta_i \hat{P}_{\mathcal{D}_S}(\mathrm{x}_S^{(i)})$ is close to a valid probability distribution. Furthermore, the optimal solution β^* matches the average value of the feature vectors of $\mathcal{D}_S$ with the average feature vectors of $\mathcal{D}_T$ and thus can be used to reweight $\mathrm{x}_S^{(i)}$, as the kernel embedding is one-to-one when using a universal kernel [10]. In particular, a kernel k is universal if it induces a strictly positive definite kernel matrix for any set of distinct points [11]. One typical universal kernel is Gaussian kernel on compact subsets of $\mathbb{R}^d$. Additionally, ϵ in (5) should be chosen as $O(\frac{B}{\sqrt{n_S}})$ such that normalizing $\sum_{i=1}^{n_S}\beta_i$ only induces a slight change of (4).

Different from [9] that assumes $P_S(\mathrm{Y}|\mathrm{X}) = P_T(\mathrm{Y}|\mathrm{X})$ and estimates $\beta = \frac{P_{\mathrm{X}_T}}{P_{\mathrm{X}_S}}$, we collect all the related variables into $\mathrm{x}^{(k)} = \left[\bar{p}_{k+d}^{d\mathrm{T}} \quad \bar{z}_{k+d}^{d\mathrm{T}} \quad \bar{p}_k^{d\mathrm{T}} \quad \bar{z}_k^{d\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{2d(n_p+n_u+n_y)}$ and employ KMM to minimize the distribution discrepancy between X_S and X_T for three reasons. Firstly, as shown in Section II-B, state estimation is affected by $\bar{p}_{k+d}^d$, $\bar{z}_{k+d}^d$, $\bar{p}_k^d$ and $\bar{z}_k^d$. Secondly, KCCA finds the optimal linear combination w and v to maximize the correlation of $w^{\mathrm{T}}\mathrm{X}$ and $v^{\mathrm{T}}\mathrm{Y}$ rather than predicting Y using X. Therefore, assuming $P_S(\mathrm{Y}|\mathrm{X}) = P_T(\mathrm{Y}|\mathrm{X})$ is not proper for state estimation using KCCA. Lastly, the generalization bounds based on the kernel embedding for KMM still hold. Additionally, the increase of data dimensions will not result in the curse of dimensionality that befalls high-dimensional density estimation, as we estimate β directly from data. Resampling x_S by $\mathrm{x}_S^{(i)} \sim \beta_i^* \hat{P}_{\mathcal{D}_S}(\mathrm{x}_S^{(i)})$ such that sample bias is corrected, we apply KCCA to the combined x_T and the resampled x_S to estimate the state sequence and then LS-SVM to estimate matrix functions.

B. Latent Space Learning

When the differences between source and target domains are beyond sample bias, there exists a latent space where $P_S(\phi(\mathrm{x}_S)) \approx P_T(\psi(\mathrm{x}_T))$ and (ϕ, ψ) are functions that map data to the features in that space, due to the similarity of source and target tasks. However, $P_S(\phi(\mathrm{x}_S)) \approx P_T(\psi(\mathrm{x}_T))$ is not a sufficient condition for effective transfer learning. For example, let $\phi(\mathrm{x}) = \psi(\mathrm{x}) = 1$, then the extracted feature is useless for system identification. Therefore, the information contained in $\mathcal{D}_S$ and $\mathcal{D}_T$ should be preserved.

Definition 1: The LPV-SS representation (2) with state dimension n_x is said to be structurally observable if there exists a scheduling trajectory $p \in \mathbb{P}^{\mathbb{Z}}$ such that the n-step observability matrix is full (column) rank for all $k \in \mathbb{Z}$.

For a structurally observable LPV-SS representation of a system, $\bar{z}_k^d$ forms a non-minimal state representation of the system, which is illuminated by Lemma 1.

Lemma 1 [5]: Let (2) be structurally observable and $d \geq n_x$. Then, there exists a function $f : \mathbb{R}^{n_f} \to \mathbb{R}^{n_y}$ with $n_f = (d+1)(n_u + n_p + n_x) + dn_y$ such that for any trajectories $p \in \mathbb{P}^{\mathbb{Z}}$, $u \in (\mathbb{R}^{n_u})^{\mathbb{Z}}$ and $e \in (\mathbb{R}^{n_y})^{\mathbb{Z}}$

$$y_k = f(u_k, p_k, e_k, \bar{z}_k^d, \bar{p}_k^d, \bar{e}_k^d). \tag{7}$$

The states should summarize the historical information that is useful for predicting the future behavior of a system. Instead of using $\bar{z}_k^d$, the method in Section II-B estimates the states up to a similarity transformation T_0 which can have dynamic scheduling dependency. Furthermore, T_0 is injective when $n_{\hat{x}} \geq n_x$ and $n_{\hat{x}}$ is determined by the rank-revealing property of the SVD in (3). To learn a common latent space to facilitate transfer learning, the objective is to find a state-space basis transformation $(T \diamond p)$[4] such that $P_S((T \diamond p) \cdot x_S) \approx P_T((T \diamond p) \cdot x_T)$ while preserving the state property. Therefore, we propose to respectively estimate state sequences x_S and x_T[5] for x_S and x_T using KCCA and then find the basis transformation $(T \diamond p)$ using transfer component analysis (TCA) proposed by [8].

To measure the distribution discrepancy between P_{x_S} and P_{x_T}, we use the maximum mean discrepancy (MMD) proposed by [12]. Given samples $x_S : \{x_S^{(i)}\}_{i=1}^{n_S} \sim p$ and $x_T : \{x_T^{(j)}\}_{j=1}^{n_T} \sim q$, the empirical estimate of MMD is

$$\widehat{\mathrm{MMD}} = \Big\| \frac{1}{n_S} \sum_{i=1}^{n_S} \phi(x_S^{(i)}) - \frac{1}{n_T} \sum_{j=1}^{n_T} \phi(x_T^{(j)}) \Big\|_{\mathcal{H}}^2 \tag{8}$$

$$= \frac{1}{n_S^2} \sum_{i=1}^{n_S} \sum_{j=1}^{n_S} \bar{k}(x_S^{(i)}, x_S^{(j)}) + \frac{1}{n_T^2} \sum_{i=1}^{n_T} \sum_{j=1}^{n_T} \bar{k}(x_T^{(i)}, x_T^{(j)}) - \frac{2}{n_S n_T} \sum_{i=1}^{n_S} \sum_{j=1}^{n_T} \bar{k}(x_S^{(i)}, x_T^{(j)}) \tag{9}$$

$$= \mathrm{tr}(KL) \tag{10}$$

where $\mathcal{H}$ denotes a reproducing kernel Hilbert space (RKHS) with kernel $\bar{k}$ and tr denotes matrix trace, $\phi : \mathcal{X} \to \mathcal{H}$ is the induced feature function by $\bar{k}$, $K = \begin{bmatrix} K_{x_S,x_S} & K_{x_S,x_T} \\ K_{x_T,x_S} & K_{x_T,x_T} \end{bmatrix}$ is the augmented kernel matrix, $L = \begin{bmatrix} \frac{1}{n_S^2}\mathbf{1}_{n_S}\mathbf{1}_{n_S}^{\mathrm{T}} & -\frac{1}{n_S n_T}\mathbf{1}_{n_S}\mathbf{1}_{n_T}^{\mathrm{T}} \\ -\frac{1}{n_T n_S}\mathbf{1}_{n_T}\mathbf{1}_{n_S}^{\mathrm{T}} & \frac{1}{n_T^2}\mathbf{1}_{n_T}\mathbf{1}_{n_T}^{\mathrm{T}} \end{bmatrix}$ is the weight matrix where $\mathbf{1}_{n_S}$ is the column vectors with all ones and length n_S, and $K, L \in \mathbb{R}^{(n_S+n_T)\times(n_S+n_T)}$. To learn $(T \diamond p)$, inspired by [8], we introduce a transformation matrix $\tilde{W}$ into the empirical kernel map [13] and derive the following parameterized kernel matrix:

$$\tilde{K} = (KK^{-\frac{1}{2}}\tilde{W})(\tilde{W}^{\mathrm{T}}K^{-\frac{1}{2}}K) = KWW^{\mathrm{T}}K, \tag{11}$$

where $W = K^{-\frac{1}{2}}\tilde{W}$. Using (10) and (11), the parameterized MMD between the states x_S and x_T can be expressed as

$$\widehat{\mathrm{MMD}} = \mathrm{tr}((KWW^{\mathrm{T}}K)L) = \mathrm{tr}(W^{\mathrm{T}}KLKW). \tag{12}$$

Then, the transformation matrix can be learned by solving the following optimization problem:

$$\min_{W} \ \mathrm{tr}(W^{\mathrm{T}}W) + \mu \mathrm{tr}(W^{\mathrm{T}}KLKW)) \tag{13}$$

$$\text{s.t.} \quad W^{\mathrm{T}}KHKW = I, \tag{14}$$

where $\mathrm{tr}(W^{\mathrm{T}}W)$ is a regularization term with a trade-off parameter μ determined by cross-validation [14] to control the complexity of W. Constraint (14) is to avoid trivial solution ($W = 0$) and $H = I_{n_S+n_T} - \frac{1}{n_S+n_T}\mathbf{1}\mathbf{1}^{\mathrm{T}}$ is the centering matrix. As shown in [8], this problem can be solved by the SVD of $(I+\mu KLK)^{-1}KHK$ and W^* consists of the eigenvectors corresponding to the $\hat{n}_{TCA}$ leading eigenvalues. Therefore, the final dimension of $\hat{x}_T = [KW]_{n_S+1:,1:\hat{n}_{TCA}}$ is $\hat{n}_{TCA}$ after state estimation via KCCA and transfer learning by TCA. Moreover, both CCA and TCA can be employed as dimensionality reduction methods, which makes it possible to use a large d to better model the dynamics but a small order $\hat{n}$ to describe the system. Since TCA only requires an eigenvalue decomposition, the computational complexity for this transfer learning approach is $O(\hat{n}_{TCA}(n_S + n_T)^2)$, according to [15]. It is noted that since $\hat{x}_S = [KW]_{1:n_S,1:\hat{n}_{TCA}}$ and $\hat{x}_T$ have similar distributions, we use only $\hat{x}_S$ to estimate matrix functions via LS-SVM and evaluate the model accuracy on $\hat{x}_T$.

[4]The notation $(T \diamond p)$ is used as a shorthand to express dynamic dependence of the state transformation T on $p_k, \ldots, p_{k-d+1}$.

[5]x_S and x_T have the same dimension.

TABLE I
CSTR MODEL SPECIFICATIONS

Description (Unit)	Value
V: volume (m^3)	5
C_1: concentration of inflow (kg/m^3)	800
C_2: concentration in reactor (kg/m^3)	213.69
Q_1: input flow (m^3/s)	0.01
Q_2: output flow (m^3/s)	0.01
k_0: pre-exponential term $(./s)$	25
E_A: activation energy of reaction (J/kg)	30e3
T_1: inflow temperature (K)	353
T_2: temperature in the reactor (K)	428.5
T_c: coolant temperature (K)	300
ρ: density (kg/m^3)	800
c_ρ: specific heat $(J/kg.s)$	1e3
ΔH: heat of reaction (J/kg)	125e3
U_{HE}: heat transfer coefficient $(J/kg.s)$	1e3
A_{HE}: surface area of heat exchange (m^2)	2
h: liquid level (m)	5
R: gas constant $(J/mol.K)$	8.31

IV. Experimental Results and Validation

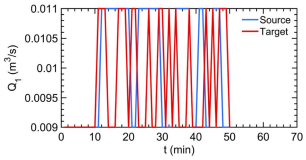
The proposed learning methods of this letter are evaluated using the model of an ideal *continuous stirred tank reactor* (CSTR). A first principles-based model of CSTR is described by (see Table I for the description of variables)

$$\dot{T}_2 = \frac{Q_1}{V}(T_1 - T_2) - \frac{U_{HE}}{A_{HE}}(T_2 - T_c) + \frac{\Delta H k_0}{\rho c_\rho} e^{-\frac{E_A}{RT_2}} C_2,$$

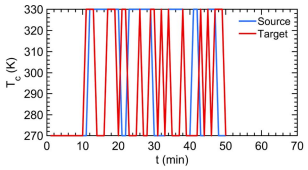
$$\dot{C}_2 = \frac{Q_1}{V}(C_1 - C_2) - k_0 e^{-\frac{E_A}{RT_2}} C_2.$$

A. Validation of Sample Bias Correction Method

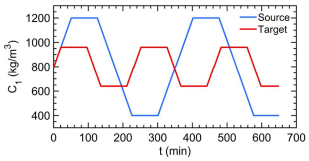
1) Experimental Setting: We use the same specifications as in [5] for the source task (denoted by $\mathcal{T}_S$). The temperature T_2 is considered to be the regulated output, Q_1 and T_c are used as manipulable signals, and raw material concentration C_1 is taken as the scheduling variable p while the internal state C_2 is assumed to be not measurable. *Pseudo random binary sequences* (PRBS) of the two inputs with $\pm 10\%$ of the nominal values are used as the exciting signals and Gaussian white noise is added to the measured output T_2 such that SNR= 25 db is maintained. The scheduling signal for C_1 is slowly varying with limits at $\pm 50\%$ of the nominal value, as shown in Fig. 1(c). Additionally, the sampling time is 60 s and the total simulation time is 120,000 s. Therefore, 2,000 samples



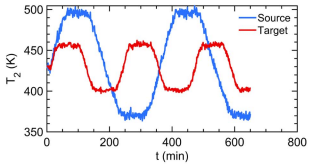
(a) The first 50 values of the input Q_1.



(b) The first 50 values of the input T_c.

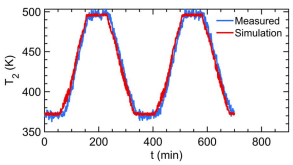


(c) The first 650 values of the scheduling variable.

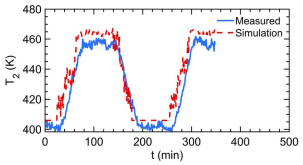


(d) The first 650 values of the measurement output.

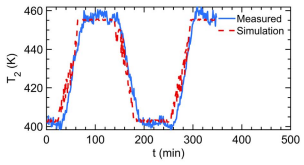
Fig. 1. Data comparison between $\mathcal{T}_S$ and $\mathcal{T}_T$ for KMM.



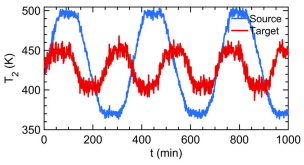
(a) Validation results for $\mathcal{T}_S$ with SNR=25 db resulting in BFR= 84.48%.



(b) Validation results for $\mathcal{T}_T$ using the hyperparameters from $\mathcal{T}_S$ resulting in BFR= 60.21%.



(c) Validation results for $\mathcal{T}_T$ with sample bias correction resulting in BFR= 75.45%.



(d) The first $1,000$ values of the measurement output.

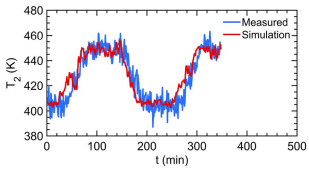
Fig. 2. BFR comparison between: (a) $\mathcal{T}_S$, (b) $\mathcal{T}_T$ using the same hyperparameters as $\mathcal{T}_S$, and (c) $\mathcal{T}_T$ with sample bias correction. Subplot (d) compares $\mathcal{T}_S$ and $\mathcal{T}_T$ for TCA.

are collected and split in training and validation set with the ratio of 65%/35%. We reproduced the results in [5] using the reported hyperparameters[6] except for $d = 2$, which saves computational time and gives a larger BFR= 84.48% than reported BFR= 83.23%.

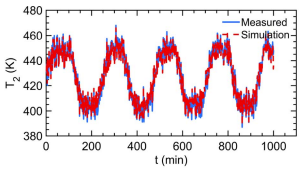
For the target task (denoted by $\mathcal{T}_T$), we decrease the limits of the scheduling trajectory from ±50% to ±20% and reduce the number of samples from 2,000 to 1,000 (see Fig. 1 for comparison). In this way, we construct a scenario where sample bias exists, i.e., $P_{\mathrm{x}_T} \neq P_{\mathrm{x}_S}$. Moreover, the distribution discrepancy can be indicated by the BFR decreases shown in Fig. 2.

2) Results and Discussion: We estimated the resampling weights β using the method described in Section III-A. For KMM, we used Gaussian kernel with $\sigma = 1$ and $B = 1$. Since the range of β (0.2923) is large, instead of resampling the training set of $\mathcal{T}_S$, we discarded the samples with $\beta_i < 0.98$

[6] $d = 2$, $n_{\hat{x}} = 2$, $\sigma_c = 470$, $v_{\mathrm{f}} = 1000$, $v_{\mathrm{p}} = 1000$, $\{\sigma_{s,i}\}_{i=1}^{4} = \{360, 2600, 260, 7000\}$, $\{\gamma_i\}_{i=1}^{2} = \{500, 500\}$ and $\psi = 1.2 \times 10^5$.



(a) Validation results for $\mathcal{T}_T$ using the hyperparameters from $\mathcal{T}_S$ with SNR=10 db resulting in BFR= 56.56%.



(b) Validation results for $\mathcal{T}_T$ via latent space learning with SNR=10 db resulting in BFR= 90.76%.

Fig. 3. BFR comparison for $\mathcal{T}_T$ with and without latent space learning.

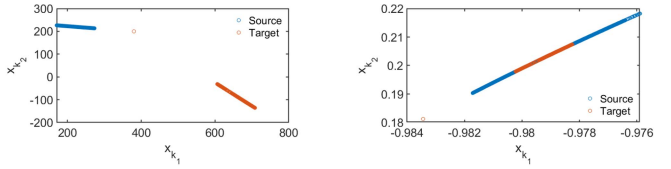
and selected $\mathcal{D}_{S_1}$ with 1,187 transferable samples that are similar to data in the training set of $\mathcal{T}_T$. Then, we estimated a state sequence for $\mathcal{T}_T$ by applying KCCA to the combined dataset of $\mathcal{D}_{S_1}$ and the training set of $\mathcal{T}_T$. With the estimated sequence, we used the extended dataset $\breve{\mathcal{D}}_T$ to estimate the matrix functions. The KMM increases BFR by over 15%, as shown in Fig. 2(c).

To further validate the proposed method, we repeated the experiment 100 times in a Monte Carlo study and the measurement noises were independently generated each time but SNR= 25 db was maintained. However, similar to the adversarial noise in deep learning, fixed hyperparameters of KCCA and LS-SVM cannot provide good performance for all the datasets with different i.i.d. noises. In this letter, we focus on the benefits of transfer learning for system identification. Therefore, only the datasets on which the hyperparameters of $\mathcal{T}_S$ give acceptable BFRs are examined to see whether the sample bias correction can boost the accuracy of identification. The maximal BFR without sample bias correction for the 100 repeats is 74.63%. The BFRs on 35 out of 100 repeats using KMM are greater than 74.63% with the mean of BFRs $\hat{\mu}_{KMM} = 75.30\%$ and standard deviation $\hat{\sigma}_{KMM} = 0.44\%$ while $\hat{\mu} = 62.32\%$ and $\hat{\sigma} = 17.18\%$ is obtained for the same 35 datasets without transfer learning.

B. Validation of Latent Space Learning Method

1) Experimental Setting: For the target task $\mathcal{T}_T$, besides decreasing the limits of the scheduling trajectory and reducing the number of samples as in Section IV-A, we change A_{HE} (which is a typical design parameter for CSTR) from 1 to 2 and thus $V = A_{HE} \cdot h$ from 5 to 10, to have similar but different systems. Moreover, we decrease SNR from 25 db to 10 db (see Fig. 2(d) for comparison). In this way, we increase the distribution discrepancy between P_{x_T} and P_{x_S}, which is indicated by the significant decrease of BFR in Fig. 3(a).

2) Results and Discussion: The method described in Section III-B was implemented, where we first estimated state sequences for $\mathcal{T}_S$ and $\mathcal{T}_T$ with $n_{\hat{x}} = 2$ and then used TCA to learn the state-space basis transformation ($T \diamond p$). Moreover, we normalized the states with the ℓ_2 norm for each dimension to facilitate computation of TCA. For TCA hyperparameters, we used Gaussian RBF kernel with $\sigma = 1$, set the trade-off parameter as $\mu = 1$ and determined $\hat{n}_{TCA} = 2$. After transformation, the state distributions of $\mathcal{T}_S$ and $\mathcal{T}_T$ in the latent space are more similar, as shown in Fig. 4(a)–(b). Therefore, the transformed data in $\mathcal{D}_S$ can facilitate learning $\mathcal{T}_T$. We applied LS-SVM approach in Section II-C to the transformed data



(a) Distributions of the estimated states x_S and x_T using KCCA.

(b) Distributions of the transformed states $x_{S,1}$ and $x_{T,1}$ using TCA.

(c) Distributions of the estimated states x_S and x_T using KCCA.

(d) Distributions of the transformed states $x_{S,1}$ and $x_{T,1}$ using TCA.

Fig. 4. State distribution comparison before/after transformation. The horizontal axis represents the first dimension of the state while the vertical the second. Subplots (a) and (b) show a successful case, while (c) and (d) show a failed case.

TABLE II
COMPARISON BETWEEN KMM AND TCA

Approach	Mean (BFR %)	Std. (BFR %)	Avg. time (s)
KMM	75.27	0.48	5.21
TCA	87.88	3.62	54.09
$\mathcal{T}_T$	64.00	16.25	-

in $\mathcal{D}_S$ to estimate matrix functions and tested the estimated matrix functions on the transformed data in $\mathcal{D}_T$. Fig. 3 shows that the learned transformation matrix significantly increases BFR by more than 30%.

Similar to Section IV-A, we repeated the experiment 100 times. Since TCA is applied after state estimation, and KCCA with fixed hyperparameters can provide insufficient state estimation due to extreme noise and low SNR, TCA can fail to find a proper latent space for transfer learning. A failed case is shown in Fig. 4(c)–(d). However, by examining the distribution in the latent space, whether to continue transfer learning can be determined in advance of estimating matrix functions. In our experiments, 43 out of 100 repeats show significant performance improvement with the mean BFR of $\hat{\mu}_{TCA} = 87.19\%$ and standard deviation of $\hat{\sigma}_{TCA} = 2.26\%$ while $\hat{\mu} = 50.61\%$ and $\hat{\sigma} = 12.90\%$ for the same 43 datasets without transfer learning.

Additionally, we tested TCA for experimental setting in Section IV-A, as TCA designed for large distribution discrepancy should be able to handle a simple scenario. In our experiments, 56 out of 100 repeats show significant performance improvement with the mean BFR of $\hat{\mu}_{TCA} = 88.12\%$ and standard deviation of $\hat{\sigma}_{TCA} = 2.28\%$ while $\hat{\mu} = 64.60\%$ and $\hat{\sigma} = 15.63\%$ for the same 56 datasets without transfer learning. To compare the performance of TCA and KMM, 24 datasets are selected where both TCA and KMM achieve BFRs greater than 74.63%. Table II summarizes the statistics on those 24 datasets. Note that $\mathcal{T}_T$ refers to identification without transfer learning, and only running time of TCA and KMM are considered to calculate the average time. When using the same hyperparameters for KCCA and LS-SVM, TCA achieves better prediction accuracy than KMM at the expense of higher computational time.

V. CONCLUDING REMARKS

In this letter, transfer learning-based methods (and in particular, sample bias correction and latent space learning) were proposed for LPV-SS model identification using kernelized machine learning. For sample bias, a kernel mean matching method was introduced to estimate the resampling weights directly from data in the source and target domains before state estimation. For latent space learning, transfer component analysis was adopted to learn a state-space transformation matrix such that the transformed data of the source task in the latent space can be used for target task learning. Experiments on two CSTR models with different parameters showed that the proposed methods can boost the accuracy of model identification and moderate the efforts of hyperparameter tuning for LPV-SS model learning using kernelized methods.

REFERENCES

[1] P. B. Cox, "Towards efficient identification of linear parameter-varying state-space models," Ph.D. dissertation, Sch. Dutch Inst. Syst. Control (DISC), Eindhoven Univ. Technol., Eindhoven, The Netherlands, 2018.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[3] B. Bamieh and L. Giarré, "Identification of linear parameter varying models," *Int. J. Robust Nonlinear Control*, vol. 12, no. 9, pp. 841–853, 2002.

[4] R. Tóth, V. Laurain, W. X. Zheng, and K. Poolla, "Model structure learning: A support vector machine approach for LPV linear-regression models," in *Proc. 50th IEEE Conf. Decis. Control Eur. Control Conf.*, 2011, pp. 3192–3197.

[5] S. Z. Rizvi, J. M. Velni, F. Abbasi, R. Tóth, and N. Meskin, "State-space LPV model identification using kernelized machine learning," *Automatica*, vol. 88, pp. 38–47, Feb. 2018.

[6] P. B. Cox, R. Tóth, and M. Petreczky, "Towards efficient maximum likelihood estimation of LPV-SS models," *Automatica*, vol. 97, pp. 392–403, Nov. 2018.

[7] F. Zhuang *et al.*, "A comprehensive survey on transfer learning," *Proc. IEEE*, early access, Jul. 7, 2020, doi: 10.1109/JPROC.2020.3004555.

[8] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[9] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," *Dataset Shift Mach. Learn.*, vol. 3, no. 4, p. 5, 2009.

[10] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample problem," 2008. [Online]. Available: http://arxiv.org/abs/0805.2368.

[11] L. Song, "Learning via Hilbert space embedding of distributions," Ph.D. dissertation, Sch. Inf. Technol., Univ. Sydney, Camperdown, NSW, Australia, 2008.

[12] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, 2006.

[13] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, 1998.

[14] T. Hastie, R. Tibshirani, and J. Friedman, "Model assessment and selection," in *The Elements of Statistical Learning*. New York, NY, USA: Springer, 2009, pp. 219–259.

[15] D. C. Sorensen, "Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations," in *Parallel Numerical Algorithms*. Dordrecht, The Netherlands: Springer, 1997, pp. 119–165.