Probability Density Estimation With Tunable Kernels Using Orthogonal Forward Regression

Sheng Chen, *Fellow, IEEE*, Xia Hong, *Senior Member, IEEE*, and Chris J. Harris

***Abstract*—A generalized or tunable-kernel model is proposed for probability density function estimation based on an orthogonal forward regression procedure. Each stage of the density estimation process determines a tunable kernel, namely, its center vector and diagonal covariance matrix, by minimizing a leave-one-out test criterion. The kernel mixing weights of the constructed sparse density estimate are finally updated using the multiplicative nonnegative quadratic programming algorithm to ensure the nonnegative and unity constraints, and this weight-updating process additionally has the desired ability to further reduce the model size. The proposed tunable-kernel model has advantages, in terms of model generalization capability and model sparsity, over the standard fixed-kernel model that restricts kernel centers to the training data points and employs a single common kernel variance for every kernel. On the other hand, it does not optimize all the model parameters together and thus avoids the problems of high-dimensional ill-conditioned nonlinear optimization associated with the conventional finite mixture model. Several examples are included to demonstrate the ability of the proposed novel tunable-kernel model to effectively construct a very compact density estimate accurately.**

***Index Terms*—Leave-one-out (LOO) cross validation, multiplicative nonnegative quadratic programming (MNQP), orthogonal forward regression (OFR), Parzen window (PW) estimate, probability density function (pdf), sparse kernel density (KD) estimate, tunable kernels.**

I. Introduction

ESTIMATING the probability density function (pdf) based on a realization sample drawn from the underlying density distribution is an important and recurrent theme in machine learning and all fields of engineering [1]–[5]. A powerful approach for density estimation is the finite mixture model (FMM) [6]. If there exists *a priori* information with regard to the functional form of the pdf, namely, the number of mixture components in the FMM is known, the problem becomes one of determining the FMM's functional parameters. The maximum-likelihood (ML) estimates of the parameters can be obtained using the expectation–maximization (EM) algorithm [7]. The associated ML optimization, in general, is a highly nonlinear optimization process requiring extensive computation, but for the Gaussian mixture model (GMM), the EM algorithm can be derived in an explicit and simple iterative form, e.g., [8]. However, this ML estimation is well known to be ill posed, and to tackle the associated numerical difficulties, it is often required to apply resampling techniques [9]–[12]. In general, the correct number of mixture components is unknown, and simultaneously determining the required number of mixture components and estimating the associated parameters of the FMM is a challenging problem. Alternatively, nonparametric techniques are routinely used for density estimation, which do not assume a particular functional form for the pdf. The classical Parzen window (PW) estimate [13] is a well-known nonparametric density estimation technique, which is remarkably simple and accurate. As the PW estimate, which is also known as the kernel density (KD) estimate, employs the full data sample set in defining the density estimate for a subsequent observation, its computational cost for testing directly scales with the sample size. This has motivated the research on the sparse KD (SKD) estimation techniques.

The support vector machine (SVM) method was applied for SKD estimation in [14]–[16], based on its promising ability to perform functional approximations in high-dimensional spaces using sparse representations. An SKD estimation technique, which is referred to as the reduced set density estimator (RSDE), was developed in [17]. Similar to the SVM method, this technique employs the full data sample set as the kernel set and obtains a sparse model by making as many kernel weights to (near) zero as possible. The difference with the SVM approach is that it is based on the minimization of the integrated squared error (ISE) between the unknown underlying density and the KD estimate, calculated on the training set, which can be shown to be equivalent to the ISE between the KD estimator considered and the PW estimate. A regression-based SKD estimation method was derived in [18]. By converting the kernels into the associated cumulative distribution functions (cdf's) and using the empirical distribution function (EDF) calculated on the training data set as the desired response, just like the SVM density estimation [14]–[16], this technique selects SKD estimates based on an orthogonal forward regression (OFR) algorithm that incrementally minimizes the training mean square error (MSE). An efficient SKD construction algorithm was proposed in [19] using the OFR based on the leave-one-out (LOO) test MSE and local regularization (LR), which is capable of constructing very sparse KD estimates with excellent generalization capability. Moreover, the density construction procedure is automatic, and the user is not required to specify an additional terminating criterion [19].

The OFR-based SKD estimation methods of [18] and [19] perform sparse kernel model selection on the associated cdf space, and they also adopt some *ad hoc* mechanisms to ensure the nonnegative and unity constraints for the kernel weights at the cost of increased computation in the model construction

Manuscript received March 24, 2009; revised June 23, 2009, September 18, 2009, and September 24, 2009; accepted October 11, 2009. Date of publication December 15, 2009; date of current version July 16, 2010. This paper was recommended by Associate Editor Q. Zhao.

S. Chen and C. J. Harris are with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (e-mail: sqc@ecs.soton.ac.uk; cjh@ecs.soton.ac.uk).

X. Hong is with the School of Systems Engineering, University of Reading, Reading RG6 6AY, U.K. (e-mail: x.hong@reading.ac.uk).

Digital Object Identifier 10.1109/TSMCB.2009.2034732

procedure. Recently, an interesting regression-based SKD estimation alternative has been proposed [20]. Using the PW estimate as the desired response, this method directly performs SKD estimation in the pdf space, and it automatically selects an SKD representation using the OFR algorithm based on the LOO test MSE and LR (OFR-LOO-LR). The nonnegative and unity constraints required for the kernel weights are met by updating the kernel weights of the selected SKD estimate using a modified multiplicative nonnegative quadratic programming (MNQP) algorithm of [21]. The MNQP algorithm has an additional desired property of further reducing the model size, yielding an even sparser density estimate. Extensive results reported in [20] demonstrate that this SKD estimation method compares favorably with other existing SKD estimation methods, in terms of model generalization capability and model sparsity, as well as computational complexity of the model construction process.

All the existing SKD estimation methods, including the SVM-based SKD estimate [14]–[16], the RSDE [17], and the OFR-based SKD estimation techniques [18]–[20], adopt a *fixed*-kernel model, which places kernel centers on the training data samples, and employ a single common kernel variance for every kernel, which is not provided by the learning algorithms and must be determined via cross validation. In this paper, we propose a *tunable*-kernel model for density estimation. Unlike the fixed-kernel approach, the center vectors of the tunable-kernel model are not restricted to the training data points, and each tunable kernel also has an individually adjusted diagonal covariance matrix. Thus, the proposed tunable-kernel model has an enhanced modeling ability and is capable of producing smaller density estimates, as compared with the fixed-kernel approach. On the other hand, we do not attempt to optimize all the parameters of the tunable-kernel model together as the FMM would, which could be a large and ill-posed nonlinear optimization task. Rather, we construct tunable kernels one by one in an OFR procedure. At each stage of the construction process, a kernel unit is tuned by determining its center vector and diagonal covariance matrix via the minimization of the LOO test MSE. This optimization can be performed using global optimization algorithms, such as the genetic algorithm (GA) [22], [23], the adaptive simulated annealing (ASA) [24], [25], or the particle swarm optimization (PSO) [26], [27]. In this paper, we adopt a simple guided search algorithm, which is referred to as the repeated weighted boosting search (RWBS) [28], to carry out the optimization.

The training complexity of this tunable-kernel algorithm can be significantly higher than those of the fixed-kernel methods [17]–[20]. However, these fixed-kernel SKD estimation methods have to determine the kernel variance, typically via a grid-search-based cross validation. Thus, the total complexity of these SKD estimation methods is equal to their training complexity given the kernel variance multiplied by the number of grid search points used, and their computational advantage over the tunable-kernel approach may be diminished, as, by contrast, the tunable-kernel approach does not require such a cross validation. It should also be pointed out that, apart from the kernel variance, the SVM-based SKD estimation methods [14]–[16] require tuning of additional hyperparameters via cross validation. Because the LOO test MSE has a property similar to "local convexity" with respect to the kernel model size [29], [30], the OFR construction procedure for tunable-kernel models will automatically terminate after constructing a small number of kernel units. This determines the structure of the density estimate. As a pdf estimate, the kernel weights of the constructed tunable-kernel model must meet the nonnegative and unity constraints, and this is guaranteed by updating the kernel weights using the MNQP algorithm [17], [21]. The extra computation involved in this weight computation is very small, since the number of kernel units constructed for the density estimate is very small owing to the enhance modeling capability of the tunable-kernel model. Moreover, during the iterative process of weight updating, some of the kernel weights may be driven to (near) zero values, and the corresponding kernel units can then be removed from the final model, further reducing the model size. This property of the MNQP algorithm is well known [17], [20], [21]. Numerical examples are used to demonstrate the ability of the combined OFR-LOO and MNQP algorithm for constructing accurate density estimates with very small tunable-kernel models.

II. Generalized Kernel Density Estimation

Given the data sample set $D_N = \{\mathbf{x}_k\}_{k=1}^N$ drawn from a density $p(\mathbf{x})$, where the data samples $\mathbf{x}_k = [x_{1,k} \quad x_{2,k} \quad \cdots \quad x_{m,k}]^T \in \mathcal{R}^m$, the task is to estimate the unknown $p(\mathbf{x})$ using the tunable-kernel model of the form

$$\hat{p}(\mathbf{x}) = \sum_{l=1}^{N_K} \beta_l K_l(\mathbf{x}) = \sum_{l=1}^{N_K} \beta_l K_{\mathbf{\Gamma}_l}(\mathbf{x}, \mathbf{c}_l) \tag{1}$$

with the constraints

$$\beta_l \geq 0, \qquad 1 \leq l \leq N_K \tag{2}$$

$$\boldsymbol{\beta}_{N_K}^T \mathbf{1}_{N_K} = 1 \tag{3}$$

where $\boldsymbol{\beta}_{N_K} = [\beta_1 \quad \beta_2 \quad \cdots \quad \beta_{N_K}]^T$ is the kernel weight vector, $\mathbf{1}_{N_K}$ denotes the vector of ones with dimension N_K, N_K is the number of tunable kernels, $\mathbf{c}_l = [c_{1,l} \quad c_{2,l} \quad \cdots \quad c_{m,l}]^T$ denotes the lth kernel unit's center vector, and the lth kernel's covariance matrix takes a diagonal form $\mathbf{\Gamma}_l = \text{diag}\{\gamma_{1,l}^2, \gamma_{2,l}^2, \ldots, \gamma_{m,l}^2\}$. In this paper, the kernel function $K_{\mathbf{\Gamma}}(\bullet, \mathbf{c})$ is chosen to be the Gaussian function of the form

$$\begin{aligned} K_{\mathbf{\Gamma}}(\mathbf{x}, \mathbf{c}) &= G_{\mathbf{\Gamma}}(\mathbf{x}, \mathbf{c}) \\ &= \frac{1}{(2\pi)^{\frac{m}{2}} \det^{\frac{1}{2}}[\mathbf{\Gamma}]} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{c})^T \mathbf{\Gamma}^{-1}(\mathbf{x}-\mathbf{c})} \end{aligned} \tag{4}$$

but any other kernel functions, satisfying $K_{\mathbf{\Gamma}}(\mathbf{x}, \mathbf{c}) \geq 0 \; \forall \; \mathbf{x} \in \mathcal{R}^m$ and

$$\int_{\mathcal{R}^m} K_{\mathbf{\Gamma}}(\mathbf{x}, \mathbf{c}) d\mathbf{x} = 1$$

can also be used.

A. GMM Estimate

For the GMM with the given number of kernels N_K, we can determine all the mixture weights β_l, center vectors $\mathbf{c}_l$, and diagonal covariance matrices $\mathbf{\Gamma}_l$ together using the

EM algorithm, which takes an explicit iterative form [8]. Denote the parameters of the GMM by $\boldsymbol{\Omega} \triangleq \{\beta_l, \mathbf{c}_l, \boldsymbol{\Gamma}_l\}_{l=1}^{N_K}$. Given a value of $\boldsymbol{\Omega}$, labeled $\boldsymbol{\Omega}^{\rm old}$, define

$$P(l|\mathbf{x}_k, \boldsymbol{\Omega}^{\rm old}) = \frac{\beta_l^{\rm old} K_{\boldsymbol{\Gamma}_l^{\rm old}}\left(\mathbf{x}_k, \mathbf{c}_l^{\rm old}\right)}{\sum_{i=1}^{N_K} \beta_i^{\rm old} K_{\boldsymbol{\Gamma}_i^{\rm old}}\left(\mathbf{x}_k, \mathbf{c}_i^{\rm old}\right)} \tag{5}$$

for $1 \le l \le N_K$ and $1 \le k \le N$. Then, a new value of $\boldsymbol{\Omega}$ is obtained according to [8]

$$\beta_l^{\rm new} = \frac{1}{N}\sum_{k=1}^{N} P(l|\mathbf{x}_k, \boldsymbol{\Omega}^{\rm old}) \tag{6}$$

$$\mathbf{c}_l^{\rm new} = \frac{\sum_{k=1}^{N} \mathbf{x}_k P(l|\mathbf{x}_k, \boldsymbol{\Omega}^{\rm old})}{\sum_{k=1}^{N} P(l|\mathbf{x}_k, \boldsymbol{\Omega}^{\rm old})} \tag{7}$$

$$\boldsymbol{\Gamma}_l^{\rm new} = \frac{1}{\sum_{k=1}^{N} P(l|\mathbf{x}_k, \boldsymbol{\Omega}^{\rm old})} \sum_{k=1}^{N} P(l|\mathbf{x}_k, \boldsymbol{\Omega}^{\rm old}) \times {\rm diag}\left\{(x_{1,k} - c_{1,l}^{\rm new})^2, \cdots, (x_{m,k} - c_{m,l}^{\rm new})^2\right\} \tag{8}$$

where $x_{i,k} - c_{i,l}^{\rm new}$ denotes the ith element of $\mathbf{x}_k - \mathbf{c}_l^{\rm new}$.

This simple EM algorithm is generally ill posed. In particular, the updating equation (8) may cause numerical problems, leading to divergence. Often, more complicated robust techniques such as the bootstrap [9], [10] may need to be used to overcome numerical difficulties. The choice of the initial $\boldsymbol{\Omega}$ is also critical, as whether the algorithm converges to local minima may depend on the initial parameter value. Our previous experience in [10] suggests that it is necessary to impose a minimum bound $\gamma_{\rm min}^2$ for all the variances $\gamma_{i,l}^2$, $1 \le i \le m$, to alleviate the numerical problem and to improve the chance of convergence. Appropriate $\gamma_{\rm min}^2$ can only be found by experiment.

B. Fixed-Kernel Density Estimate

The standard fixed-KD estimate can be viewed as a special case of this generalized KD estimate by choosing $N_K = N$, using every data sample $\mathbf{x}_k \in D_N$ as kernel center $\mathbf{c}_k$ and setting $\gamma_{i,k}^2 = \gamma^2$, $\forall\, i, k$, where γ^2 is assumed to be given, for example, via cross validation. The density estimation model (1) in this case is expressed as

$$\hat{p}(\mathbf{x}; \gamma) = \sum_{k=1}^{N} \beta_k K_\gamma(\mathbf{x}, \mathbf{x}_k) \tag{9}$$

where γ is also known as the fixed-kernel width.

In particular, the PW estimate is obtained by further setting all the kernel weights to $\beta_k = 1/N$, $1 \le k \le N$, yielding

$$\hat{p}_{\rm Parz}(\mathbf{x}; \gamma_{\rm Parz}) = \frac{1}{N}\sum_{k=1}^{N} K_{\gamma_{\rm Parz}}(\mathbf{x}, \mathbf{x}_k) \tag{10}$$

where $\gamma_{\rm Parz}^2$ denotes the kernel variance for the PW estimate, which is obtained based on cross validation. The PW estimate, in fact, can be derived as the ML estimator using the divergence-based criterion [6]. The negative cross entropy or divergence between the true density $p(\mathbf{x})$ and the estimate $\hat{p}(\mathbf{x}; \gamma)$ of (9), which is calculated on the training set, is defined as

$$\int_{\mathcal{R}^m} p(\mathbf{u}) \log\, \hat{p}(\mathbf{u}; \gamma) d\mathbf{u} \approx \frac{1}{N}\sum_{k=1}^{N} \log\, \hat{p}(\mathbf{x}_k; \gamma) = \frac{1}{N}\sum_{k=1}^{N} \log\left(\sum_{n=1}^{N} \beta_n K_\gamma(\mathbf{x}_k, \mathbf{x}_n)\right). \tag{11}$$

Minimizing this divergence subject to the constraints (2) and (3) leads to $\beta_n = 1/N$ for $1 \le n \le N$, i.e., the PW estimate. The PW estimate (10) is known to process a mean ISE convergence rate on the order of N^{-1} [13], but it is nonsparse.

The RSDE [17] derives an SKD estimate by making as many kernel weights to near zero as possible in (9) based on the ISE criterion. Specifically, with the fixed Gaussian kernel

$$K_\gamma(\mathbf{x}, \mathbf{x}_k) = G_\gamma(\mathbf{x}, \mathbf{x}_k) = \frac{1}{(2\pi\gamma^2)^{\frac{m}{2}}} e^{-\frac{\|\mathbf{x}-\mathbf{x}_k\|^2}{2\gamma^2}} \tag{12}$$

the kernel weight vector of the RSDE is obtained by solving the constrained nonnegative quadratic programming

$$\min_{\boldsymbol{\beta}_N}\left\{\frac{1}{2}\boldsymbol{\beta}_N^T \mathbf{G}_N \boldsymbol{\beta}_N - \hat{\mathbf{p}}_N^T \boldsymbol{\beta}_N\right\} \quad \text{s.t. } \boldsymbol{\beta}_N^T \mathbf{1}_N = 1 \text{ and } \beta_i \ge 0, \qquad 1 \le i \le N \tag{13}$$

where $\mathbf{G}_N = [g_{i,j}] \in \mathcal{R}^{N\times N}$ with

$$g_{i,j} = \int_{\mathcal{R}^m} G_\gamma(\mathbf{x}, \mathbf{x}_i) G_\gamma(\mathbf{x}, \mathbf{x}_j) d\mathbf{x} = G_{\sqrt{2}\gamma}(\mathbf{x}_i, \mathbf{x}_j) \tag{14}$$

$$\hat{\mathbf{p}}_N = [\hat{p}_{\rm Parz}(\mathbf{x}_1; \gamma)\hat{p}_{\rm Parz}(\mathbf{x}_2; \gamma), \ldots, \hat{p}_{\rm Parz}(\mathbf{x}_N; \gamma)]^T \tag{15}$$

i.e., the ith element of $\hat{\mathbf{p}}_N$ is $\hat{p}_{\rm Parz}(\mathbf{x}_i; \gamma)$, the PW estimate at the data point $\mathbf{x}_i$ with the same kernel width γ as the KD estimate to be determined. Note that the ISE between the unknown underlying density and the KD estimate, which is calculated on the training set, is equivalent to the ISE between the KD estimator and the PW estimator, as is illustrated in the following equation:

$$\min_{\boldsymbol{\beta}_N} \int_{\mathcal{R}^m} \left|\hat{p}_{\rm Parz}(\mathbf{x}; \gamma_{\rm Parz}) - \hat{p}(\mathbf{x}; \gamma)\right|^2 d\mathbf{x} = \min_{\boldsymbol{\beta}_N} \int_{\mathcal{R}^m} \hat{p}^2(\mathbf{x}; \gamma) d\mathbf{x} - 2\sum_{i=1}^{N} \beta_i \mathcal{E}_{\hat{p}_{\rm Parz}}\left[K_\gamma(\mathbf{x}, \mathbf{x}_i)\right] \tag{16}$$

where $\mathcal{E}_{\hat{p}_{\rm Parz}}[\bullet]$ denotes the expectation with respect to $\hat{p}_{\rm Parz}(\mathbf{x}; \gamma_{\rm Parz})$. Given (12), the first term in the right-hand side of (16) is the first term of the cost function in (13), whereas the second term in right-hand side of (16) can be expressed as

$$\sum_{i=1}^{N} \beta_i \mathcal{E}_{\hat{p}_{\rm Parz}}\left[K_\gamma(\mathbf{x}, \mathbf{x}_i)\right] \approx \sum_{i=1}^{N} \beta_i \frac{1}{N}\sum_{k=1}^{N} K_\gamma(\mathbf{x}_k, \mathbf{x}_i) = \sum_{i=1}^{N} \beta_i \hat{p}_{\rm Parz}(\mathbf{x}_i; \gamma) \tag{17}$$

which is identical to the second term of the cost function in (13). To solve the constrained optimization (13), in particular to obtain an SKD estimate, the MNQP algorithm [21] can be used. However, because the full kernel matrix $\mathbf{G}_N$ has a very high dimension of $N \times N$, the MNQP algorithm slowly converges. The RSDE [17] uses the alternative sequential minimal optimization [31] to solve (13). Note that the optimization process can only drive many kernel weights to small values, and therefore, an appropriate zero threshold has to empirically be specified to remove these weights.

The SKD estimator of [20] is very different from the RSDE. It uses the OFR-LOO-LR algorithm to select a small subset of significant kernels from the full fixed-kernel set that contains all the training data points. The associated kernel weight vector of the sparse estimate is updated with the MNQP algorithm.

The proposed tunable KD estimator can be viewed as an alternative between the fully "nonlinear" optimization approach optimizing all the kernel parameters together, such as the FMM estimator, and the "linear" optimization approach based on the fixed-kernel model, such as the SKD estimator of [20] and the RSDE of [17].

C. Tunable-Kernel Density Estimate

Density estimation is an unsupervised learning problem. One way of transferring it into a supervised learning problem is to convert the kernels into the associated cdf's and adopt the EDF calculated using the training data as the desired response, as in the SKD estimation methods of [14]–[16], [18], and [19]. The true cdf of the pdf $p(\mathbf{x})$ is defined as

$$F(\mathbf{x}) = \int_{-\infty}^{\mathbf{x}} p(\mathbf{u}) d\mathbf{u} \tag{18}$$

and the cdf associated with kernel $K_l(\mathbf{x})$ is given by

$$q_l(\mathbf{x}) = \int_{-\infty}^{\mathbf{x}} K_{\mathbf{\Gamma}_l}(\mathbf{u}, \mathbf{c}_l) d\mathbf{u} \tag{19}$$

where $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_m]^T \in \mathcal{R}^m$. Further define the EDF on the training set D_N as

$$\hat{F}(\mathbf{x}) = \frac{1}{N} \sum_{k=1}^{N} \prod_{j=1}^{m} \theta(x_j - x_{j,k}) \tag{20}$$

with

$$\theta(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0. \end{cases} \tag{21}$$

Using $\hat{F}(\mathbf{x})$ as the desired response for $F(\mathbf{x})$, the density estimation can be expressed as a regression modeling

$$\hat{F}(\mathbf{x}) = \sum_{l=1}^{N_K} \beta_l q_l(\mathbf{x}) + \hat{\epsilon}(\mathbf{x}) \tag{22}$$

subject to the constraints (2) and (3), where $\hat{\epsilon}(\mathbf{x})$ denotes the modeling error at $\mathbf{x}$. According to the Glivenko–Cantelli theorem [32], the EDF (20) converges to the true cdf almost surely as $N \to \infty$, under the assumption of independent identically distributed observations.

An alternative approach is the direct modeling in the pdf space by using the PW estimate (10) as the desired response of the true pdf $p(\mathbf{x})$, as in the SKD estimation methods of [20]. This is the approach adopted in this paper. The PW estimate can be viewed as the "observation" of the true density contaminated by some "observation noise," namely, $\hat{p}_{\mathrm{Parz}}(\mathbf{x}; \gamma_{\mathrm{Parz}}) = p(\mathbf{x}) + \tilde{\epsilon}(\mathbf{x})$. Thus, the generic density estimation problem (1) can be viewed as the following regression problem with the PW estimate as the desired response:

$$\hat{p}_{\mathrm{Parz}}(\mathbf{x}; \gamma_{\mathrm{Parz}}) = \sum_{l=1}^{N_K} \beta_l K_l(\mathbf{x}) + \epsilon(\mathbf{x}) \tag{23}$$

subject to the constraints (2) and (3), where $\epsilon(\mathbf{x})$ is the modeling error at $\mathbf{x}$.

Define $\boldsymbol{\phi}_{N_K}(k) = [K_1(\mathbf{x}_k) \quad K_2(\mathbf{x}_k) \quad \cdots \quad K_{N_K}(\mathbf{x}_k)]^T$ and $t_k = \hat{p}_{\mathrm{Parz}}(\mathbf{x}_k; \gamma_{\mathrm{Parz}})$. Then, the regression model (23) for the data point $\mathbf{x}_k \in D_N$ can be expressed as

$$t_k = \hat{t}_k + \epsilon(k) = \boldsymbol{\phi}_{N_K}^T(k) \boldsymbol{\beta}_{N_K} + \epsilon(k) \tag{24}$$

where $\epsilon(k) = \epsilon(\mathbf{x}_k)$. Thus, the regression model (23) over the training data set D_N can be expressed in the matrix form

$$\mathbf{t} = \mathbf{\Phi}_{N_K} \boldsymbol{\beta}_{N_K} + \boldsymbol{\epsilon} \tag{25}$$

where $\mathbf{t} = [t_1 \quad t_2 \quad \cdots \quad t_N]^T$, $\boldsymbol{\epsilon} = [\epsilon(1) \quad \epsilon(2) \quad \cdots \quad \epsilon(N)]^T$, and the regression matrix $\mathbf{\Phi}_{N_K} = [\boldsymbol{\phi}_1 \quad \boldsymbol{\phi}_2 \quad \cdots \quad \boldsymbol{\phi}_{N_K}]$ with $\boldsymbol{\phi}_l = [K_l(\mathbf{x}_1) \quad K_l(\mathbf{x}_2) \quad \cdots \quad K_l(\mathbf{x}_N)]^T$, $1 \leq l \leq N_K$. Note that $\boldsymbol{\phi}_l$ is the lth column of $\mathbf{\Phi}_{N_K}$, whereas $\boldsymbol{\phi}_{N_K}^T(k) = [\phi_{k,1} \quad \phi_{k,2} \quad \cdots \quad \phi_{k,N_K}]$ is the kth row of $\mathbf{\Phi}_{N_K}$. Let an orthogonal decomposition of the regression matrix $\mathbf{\Phi}_{N_K}$ be

$$\mathbf{\Phi}_{N_K} = \mathbf{W}_{N_K} \mathbf{A}_{N_K} \tag{26}$$

where

$$\mathbf{A}_{N_K} = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,N_K} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{N_K-1,N_K} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{27}$$

and $\mathbf{W}_{N_K} = [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \cdots \quad \mathbf{w}_{N_K}]$ with columns satisfying $\mathbf{w}_i^T \mathbf{w}_j = 0$, if $i \neq j$. The regression model (25) can alternatively be expressed as

$$\mathbf{t} = \mathbf{W}_{N_K} \mathbf{g}_{N_K} + \boldsymbol{\epsilon} \tag{28}$$

where the weight vector $\mathbf{g}_{N_K} = [g_1 \quad g_2 \quad \cdots \quad g_{N_K}]^T$ associated with the orthogonal regression matrix $\mathbf{W}_{N_K}$ satisfies the triangular system $\mathbf{A}_{N_K} \boldsymbol{\beta}_{N_K} = \mathbf{g}_{N_K}$. The space spanned by the original model bases $\boldsymbol{\phi}_l$, $1 \leq l \leq N_K$, is identical to the space spanned by the orthogonal model bases $\mathbf{w}_l$, $1 \leq l \leq N_K$, and the model output $\hat{t}_k$ is equivalently expressed by

$$\hat{t}_k = \mathbf{w}_{N_K}^T(k) \mathbf{g}_{N_K} \tag{29}$$

where $\mathbf{w}_{N_K}^T(k) = [\, w_{k,1} \quad w_{k,2} \quad \cdots \quad w_{k,N_K} \,]$ is the kth row of $\mathbf{W}_{N_K}$.

The regression framework (24) does not imply that we use the density estimate (1) to approximate the PW estimate. Rather, the objective is to estimate or approximate the underlying density $p(\mathbf{x})$ whose noisy "observation" is considered to be the PW estimate. This is exactly as in the standard regression where the underlying data-generating mechanism is unknown but whose noisy observation is available as the target of the regression model output. The objective of regression is not to approximate the noisy observation or desired output but to uncover or approximate the underlying data-generating mechanism. If an estimate can only approximate the desired response, it is known as overfitting and cannot be regarded as an adequate estimate. Similarly, the EDF (20) is nondifferentiable, but the true density and cdf are differentiable. Using the framework (22) does not imply a nondifferentiable estimate. In addition, considering the PW estimate as the "desired" response is not a strange idea at all. The RSDE [17] is said to be based on the ISE criterion. Using the same argument of [17], we have just shown in the previous section that this ISE criterion is equivalent to the ISE criterion between the KD estimate considered and the PW estimate.

Reformulating the density estimation as a regression problem by using the PW estimate as the target function of the true pdf has some advantages over the regression approach based on using the EDF as the target function of the true cdf. The former approach can use many types of kernel function, and it is computationally simpler, as it does not need to compute the values of regressors (19) on the training data set D_N. Computing the associated cdf's for the kernels can be inconvenient and may be difficult for certain types of kernels. Computing the values of the PW estimator on D_N is no more complex than calculating the values of $\hat{F}(\mathbf{x})$ on D_N. The only drawback of using the PW estimate is that the kernel variance for the PW estimator must be determined. Although we developed the tunable-kernel model using the PW estimate as the target function, the developed construction algorithm is equally applicable for both approaches.

III. Generalized Kernel Density Construction

We present the combined OFR-LOO and MNQP algorithm for constructing the generalized KD estimate (1) with excellent generalization capability. The structure of this density estimate is determined using the OFR procedure based on the LOO test MSE, which automatically constructs the set of tunable kernels $\{\mathbf{c}_l, \mathbf{\Gamma}_l\}_{l=1}^{N_K}$ one by one. The kernel weights of the constructed generalized KD estimate are then tuned by the MNQP algorithm to ensure the nonnegative and unity constraints (2) and (3). Denote the augmented training data set by $\bar{D}_N = \{\mathbf{x}_k, t_k\}_{k=1}^N$. The concept of LOO cross validation is explained in detail, for example, in [29] and [30], and therefore, it will not be repeated here.

A. OFR With the LOO Test Criterion

Consider the modeling process after the nth stage, which produces the n-unit model identified using $\bar{D}_N$, which is written in the following form for notational convenience:

$$\hat{\mathbf{t}}^{(n)} = \mathbf{\Phi}_n \boldsymbol{\beta}_n = \mathbf{W}_n \mathbf{g}_n \tag{30}$$

where $\mathbf{\Phi}_n = [\boldsymbol{\phi}_1 \quad \boldsymbol{\phi}_2 \quad \cdots \quad \boldsymbol{\phi}_n]$ is the n-unit regression matrix, $\boldsymbol{\beta}_n = [\beta_1 \quad \beta_2 \quad \cdots \quad \beta_n]^T$ is the associated model's weight vector, $\mathbf{W}_n = [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \cdots \quad \mathbf{w}_n]$, and $\mathbf{g}_n = [g_1 \quad g_2 \quad \cdots \quad g_n]^T$. The kth element of $\hat{\mathbf{t}}^{(n)}$ is

$$\hat{t}^{(n)}(k) = \boldsymbol{\phi}_n^T(k)\boldsymbol{\beta}_n = \mathbf{w}_n^T(k)\mathbf{g}_n \tag{31}$$

where $\boldsymbol{\phi}_n^T(k)$ denotes the kth row of $\mathbf{\Phi}_n$, and $\mathbf{w}_n^T(k)$ denotes the kth row of $\mathbf{W}_n$. The modeling error of this n-unit model is simply

$$\epsilon^{(n)}(k) = t_k - \hat{t}^{(n)}(k). \tag{32}$$

Now, "remove" the kth data point from $\bar{D}_N$ and use the remaining $N-1$ data points to identify the n-unit model. The "test" error of the resulting LOO n-unit model can be calculated on the data point removed from training. This LOO modeling error, which is denoted by $\epsilon^{(n,-k)}(k)$, is given by [33]

$$\epsilon^{(n,-k)}(k) = \frac{\epsilon^{(n)}(k)}{\eta^{(n)}(k)} \tag{33}$$

where $\eta^{(n)}(k)$ is the associated LOO error weighting. The LOO MSE for the n-unit generalized KD model is then defined by

$$J_n = \frac{1}{N}\sum_{k=1}^{N}\left(\epsilon^{(n,-k)}(k)\right)^2. \tag{34}$$

This LOO MSE is a measure of the model generalization capability [33]–[36]. For the model (28) with the orthogonal regression matrix, the computation of the LOO test MSE criterion J_n is very efficient because $\epsilon^{(n)}(k)$ and $\eta^{(n)}(k)$ can recursively be computed using [29], [30]

$$\epsilon^{(n)}(k) = \epsilon^{(n-1)}(k) - g_n w_{k,n} \tag{35}$$

$$\eta^{(n)}(k) = \eta^{(n-1)}(k) - \frac{w_{k,n}^2}{\mathbf{w}_n^T\mathbf{w}_n + \lambda} \tag{36}$$

respectively, where $\lambda \geq 0$ is a small regularization parameter. The regularization parameter λ enters (36) if the regularized orthogonal least-squares solution for the weights is adopted [30]. Note that, in the current OFR-LOO algorithm, λ can simply be set to zero (no regularization) or a very small value (e.g., 10^{-6}). This is very different from our previous OFR-LOO-LR algorithm for selecting SKD estimates with the fixed-kernel model [20], where multiple regularizers are employed and an iterative evidence procedure has to be used to update the regularization parameters.

The proposed OFR-LOO algorithm constructs the generalized kernel units one by one by minimizing the LOO MSE J_n. Specifically, at the nth stage of the construction procedure, the nth tunable-kernel unit is determined by minimizing J_n with respect to the kernel's center vector $\mathbf{c}_n$ and the diagonal covariance matrix $\mathbf{\Gamma}_n$, i.e.,

$$\min_{\mathbf{c}_n, \mathbf{\Gamma}_n} J_n(\mathbf{c}_n, \mathbf{\Gamma}_n). \tag{37}$$

The construction procedure is automatically terminated when

$$J_{N_K} \leq J_{N_K+1} \tag{38}$$

yielding an N_K-term generalized kernel model. Note that, for the LOO criterion J_n, there always exists an "optimal" N_K such that, for $n \leq N_K, J_n$ decreases as the model size n increases while the condition (38) holds [29], [30]. The search space of the optimization (37) can simply be set as

$$\min\{x_{i,k}, 1 \leq k \leq N\} \leq c_{i,n} \leq \max\{x_{i,k}, 1 \leq k \leq N\}$$
$$0 < \tilde{\gamma}^2_{\min} \leq \gamma^2_{i,n} \leq \tilde{\gamma}^2_{\max}, \qquad 1 \leq i \leq m.$$

Since J_n is nonconvex with respect to $\mathbf{c}_n$ and $\mathbf{\Gamma}_n$, a gradient-based algorithm may become trapped at a local minimum. Alternatively, global search optimization methods, such as the GA [22], [23], the ASA [24], [25], or the PSO [26], [27], may be used to perform the optimization task (37). We adopt the RWBS algorithm [28] to determine $\mathbf{c}_n$ and $\mathbf{\Gamma}_n$. The motivation and analysis of the RWBS algorithm as a general global optimizer are detailed in [28]. A comparative study given in [28] shows that the RWBS algorithm achieves a similar convergence speed as the GA and ASA for several global optimization applications while offering advantages of minimum programming effort and fewer algorithmic parameters to tune. The detailed procedure for determining the nth kernel unit based on the RWBS algorithm is given in the Appendix.

B. Determining Weights of the Generalized Kernel Model

After the structure determination using the aforementioned OFR-LOO algorithm, an N_K-term generalized kernel model, i.e., $\{K_{\mathbf{\Gamma}_l}(\mathbf{x}, \mathbf{c}_l)\}_{l=1}^{N_K}$, is obtained, where N_K is typically very small. The kernel weight vector $\boldsymbol{\beta}_{N_K}$, which is computed from $\mathbf{A}_{N_K}\boldsymbol{\beta}_{N_K} = \mathbf{g}_{N_K}$, may not satisfy the constraints (2) and (3). However, we can use a modified version of the MNQP algorithm [21] to calculate $\boldsymbol{\beta}_{N_K}$. Since N_K is very small, the extra computation involved is small. This task is defined as follows: finding $\boldsymbol{\beta}_{N_K}$ for the generalized kernel model (25), subject to the constraints (2) and (3). More specifically, the kernel weight vector can be obtained by solving the following constrained nonnegative quadratic programming:

$$\min_{\boldsymbol{\beta}_{N_K}} \left\{ \frac{1}{2}\boldsymbol{\beta}^T_{N_K}\mathbf{B}_{N_K}\boldsymbol{\beta}_{N_K} - \mathbf{v}^T_{N_K}\boldsymbol{\beta}_{N_K} \right\}$$
$$\text{s.t. } \boldsymbol{\beta}^T_{N_K}\mathbf{1}_{N_K} = 1 \text{ and } \beta_i \geq 0, \qquad 1 \leq i \leq N_K \quad (39)$$

where $\mathbf{B}_{N_K} = \mathbf{\Phi}^T_{N_K}\mathbf{\Phi}_{N_K} = [b_{i,j}] \in \mathcal{R}^{N_K \times N_K}$ is the related design matrix, and $\mathbf{v}_{N_K} = \mathbf{\Phi}^T_{N_K}\mathbf{t} = [\, v_1 \quad v_2 \quad \cdots \quad v_{N_K}\,]^T$. Although there exists no closed-form solution for this optimization problem, the solution can readily be obtained iteratively.

Since the elements of $\mathbf{B}_{N_K}$ and $\mathbf{v}_{N_K}$ are strictly positive, the auxiliary function [21] for the aforementioned problem is given by

$$\frac{1}{2}\sum_{i=1}^{N_K}\sum_{j=1}^{N_K} b_{i,j}\frac{\beta_j^{[\iota]}\left(\beta_i^{[\iota+1]}\right)^2}{\beta_i^{[\iota]}} - \sum_{i=1}^{N_K} v_i\beta_i^{[\iota+1]}$$

and the Lagrangian associated with this auxiliary problem can be formed as [17]

$$\mathcal{L} = \frac{1}{2}\sum_{i=1}^{N_K}\sum_{j=1}^{N_K} b_{i,j}\frac{\beta_j^{[\iota]}\left(\beta_i^{[\iota+1]}\right)^2}{\beta_i^{[\iota]}} - \sum_{i=1}^{N_K} v_i\beta_i^{[\iota+1]} - h^{[\iota]}\left(\sum_{i=1}^{N_K}\beta_i^{[\iota+1]} - 1\right) \quad (40)$$

where the superindex $^{[\iota]}$ denotes the iteration index, and h is the Lagrangian multiplier. Setting

$$\frac{\partial\mathcal{L}}{\partial\beta_i^{[\iota+1]}} = 0 \qquad \frac{\partial\mathcal{L}}{\partial h^{[\iota]}} = 0 \quad (41)$$

leads to the following updating equations:

$$c_i^{[\iota]} = \beta_i^{[\iota]}\left(\sum_{j=1}^{N_K} b_{i,j}\beta_j^{[\iota]}\right)^{-1}, \qquad 1 \leq i \leq N_K \quad (42)$$

$$h^{[\iota]} = \left(\sum_{i=1}^{N_K} c_i^{[\iota]}\right)^{-1}\left(1 - \sum_{i=1}^{N_K} c_i^{[\iota]}v_i\right) \quad (43)$$

$$\beta_i^{[\iota+1]} = c_i^{[\iota]}\left(v_i + h^{[\iota]}\right). \quad (44)$$

It is easy to check that, if $\boldsymbol{\beta}_{N_K}^{[\iota]}$ meets the constraints (2) and (3), $\boldsymbol{\beta}_{N_K}^{[\iota+1]}$ updated according to (42)–(44) also satisfies (2) and (3). The initial condition can be set as $\beta_i^{[0]} = 1/N_K$, $1 \leq i \leq N_K$. During the iterative procedure, some of the kernel weights may be driven to (near) zero. The corresponding kernels can then be removed from the kernel model, leading to a further reduction in the generalized kernel model size.

C. Computational Complexity Comparison

Since the computational complexity of the MNQP algorithm for tuning the kernel weights is negligible, the complexity is dominated by the complexity of the OFR-LOO algorithm for selecting N_K tunable kernels. The computational complexity of one LOO cost function evaluation and the associated model column orthogonalization can be shown to be on the order of $O(N)$ (see the Appendix, (63)–(68)). In fact, the exact complexity expression for $O(N)$ can be derived as in [37]. Thus, the computational requirements of the proposed combined OFR-LOO and MNQP algorithm for constructing the generalized SKD (GSKD) estimate can be expressed as

$$C(\text{GSKD}) = N_{\text{ce}}(\text{GSKD}) \times O(N). \quad (45)$$

Here, $N_{\text{ce}}(\text{GSKD})$ is the total number of the LOO cost function evaluations and the associated model column orthogonalizations. This number can readily be shown to be

$$N_{\text{ce}}(\text{GSKD}) = N_K\left(N_G(P_S + 2M_I) - (N_G - 1)\right)$$
$$\approx N_K N_G(P_S + 2M_I)$$

where the population size P_s, the number of the generations N_G, and the number of the weighted boosting search (WBS) iterations M_I are the algorithmic parameters of the RWBS optimization algorithm, as defined in the Appendix.

For our previous combined OFR-LOO-LR and MNQP algorithm for constructing the fixed-kernel SKD estimate [20], the computational complexity of one LOO cost function evaluation and the associated model column orthogonalization is also on the order of $O(N)$. Assume that, for the given kernel variance γ^2, the algorithm selects N'_K kernels from the full set of N fixed kernels. Then, the computational requirements of this algorithm with the given γ^2 is determined by

$$C(\text{SKD}) = N_{\text{ce}}(\text{SKD}) \times O(N). \tag{46}$$

Here, $N_{\text{ce}}(\text{SKD})$ is the total number of the LOO cost function evaluations and associated model column orthogonalizations with the given γ^2, which can be shown to be

$$N_{\text{ce}}(\text{SKD}) = \sum_{i=1}^{N'_K} (N - (i-1)) \approx N'_K N$$

where the approximation is arrived because the selected model size N'_K is usually much smaller than the training data size N. Basically, $C(\text{SKD}) = O(N^2)$, as the complexity of an OFR-type algorithm for the fixed-kernel model is well known to be on the order of $O(N^2)$ [37].

Typically, $C(\text{SKD}) \ll C(\text{GSKD})$. Consider the 6-D density estimation example of Section IV-D, where $N = 600$. For the GSKD estimation, $P_S = 40$, $N_G = 10$, and $M_I = 400$ were used for the RWBS algorithm, and the resulting average model size was $N_K = 5$. Thus, on average, $C(\text{GSKD}) = 42\,000 \times O(600)$. Since the average $N'_K = 9.4$ for the OFR-LOO-LR algorithm, $C(\text{SKD}) = 5640 \times O(600)$ on average. However, the complexity (46) is for a given kernel variance. This hyperparameter has to be determined by a line search based on cross validation. Let us make an optimistic assumption that, at each point of the line search, the algorithm produces the same model size N'_K. The true complexity of the OFR-LOO-LR algorithm for constructing the SKD estimate with fixed kernels will be $L_S \times C(\text{SKD})$, where L_S is the total points of the line search. Therefore, the training complexity for constructing the tunable-kernel estimate may not necessarily be higher than that for selecting the fixed-kernel estimate.

The computational complexity of the RSDE algorithm is also known to be on the order of $O(N^2)$ [17]. The PW estimator is a plug-in estimator, and its training complexity involves only the determination of the kernel variance via cross validation. The problem of the PW estimator is its high test complexity. For the GMM estimator, each iteration of the EM algorithm involves the computation of (5)–(8). Thus, the computational complexity of this EM algorithm is reasonably low. However, care much be exercised in choosing the initial parameter values; otherwise, the algorithm may not converge or may suffer from the problem of local minima.

IV. Numerical Examples

Five examples were used in the simulation to test the proposed combined OFR-LOO and MNQP algorithm for constructing the GSKD estimator and to compare its performance with the three fixed-kernel estimators, namely, the nonsparse PW estimator, our previous SKD estimator [20], and the RSDE of [17], as well as the tunable-kernel GMM estimator. For each case, a data set of N randomly drawn samples was used to construct the density estimate, and a separate test data set of $N_{\text{test}} = 10\,000$ samples was used to calculate the L_1 test error for the resulting estimate according to

$$L_1 = \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} |p(\mathbf{x}_k) - \hat{p}(\mathbf{x}_k)| \,. \tag{47}$$

The Kullback–Leibler divergence (KLD), which is defined as

$$D_{\text{KL}}(p|\hat{p}) = \int_{\mathcal{R}^m} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{\hat{p}(\mathbf{x})} d\mathbf{x} \tag{48}$$

is a measure of the difference between the two probability distributions $p(\mathbf{x})$ and $\hat{p}(\mathbf{x})$. For the 1-D and 2-D problems, the KLD was also used to test the resulting estimates. For a 1-D problem, the KLD can accurately be approximated by partitioning the integration range $[x_{\min},\ x_{\max}]$ into the N_{par} small equal-length intervals and computing the summation

$$D_{\text{KL}}(p|\hat{p}) \approx \sum_{k=1}^{N_{\text{par}}} p(k) \log \frac{p(k)}{\hat{p}(k)} \Delta x \tag{49}$$

where $\Delta x = (x_{\max} - x_{\min})/N_{\text{par}}$, $p(k) = p(x_{\min} + k\Delta x)$, and $\hat{p}(k) = \hat{p}(x_{\min} + k\Delta x)$. In the experiment, we chose $N_{\text{par}} \geq 10\,000$ to ensure the accuracy of the approximation. Similarly, for a 2-D problem, the KLD is approximated by partitioning the integration range $[x_{1,\min},\ x_{1,\max}] \times [x_{2,\min},\ x_{2,\max}]$ into the $N_{\text{par}} \times N_{\text{par}}$ small equal-area intervals and calculating

$$D_{\text{KL}}(p|\hat{p}) \approx \sum_{k=1}^{N_{\text{par}}} \sum_{l=1}^{N_{\text{par}}} p(k,l) \log \frac{p(k,l)}{\hat{p}(k,l)} (\Delta x)^2 \tag{50}$$

where $\Delta x = (x_{1,\max} - x_{1,\min})/N_{\text{par}} = (x_{2,\max} - x_{2,\min})/N_{\text{par}}$, $p(k,l) = p(x_{1,\min} + k\Delta x, x_{2,\min} + l\Delta x)$, and $\hat{p}(k,l) = \hat{p}(x_{1,\min} + k\Delta x, x_{2,\min} + l\Delta x)$. To ensure the accuracy of the approximation, we chose $N_{\text{par}} > 100$. For higher dimensional problems, calculation of the KLD becomes computationally too expensive. The experiment was repeated by N_{run} different random runs for each example.

The optimal values of the kernel variances γ^2_{Parz} and γ^2 for the PW estimator and the two SKD estimators with the fixed-kernel model, respectively, were empirically found via cross validation. For the GMM, the number of mixing Gaussian components N_K must be determined. Instead of exhaustedly trying different values for the number of mixing components based on cross validation, we used the average model size obtained for the GSKD estimate as the number of mixing components for the GMM. For the EM algorithm, all the initial mixing weights β_l were set to $1.0/N_K$, the initial center vectors $\mathbf{c}_l$ were randomly chosen from the region $[a,\ b]^m \in \mathcal{R}^m$, and all the initial variances $\gamma^2_{i,l}$ were set to the same value γ^2_{ini}. A minimum bound $\gamma^2_{\min}$ for the variances was also assigned. If some runs of the EM algorithm were observed to diverge, the

TABLE I
PERFORMANCE OF THE PW ESTIMATOR, PREVIOUS SKD ESTIMATOR [20], RSDE [17], PROPOSED GSKD ESTIMATOR, AND GMM ESTIMATOR IN TERMS OF L_1 TEST ERROR AND KLD DIVERGENCE, AS WELL AS NUMBER OF KERNELS REQUIRED FOR THE 1-D EXAMPLE OF THE EIGHT-GAUSSIAN MIXTURE, QUOTED AS MEAN ± STANDARD DEVIATION, OVER 200 RUNS

estimator	PW	SKD [20]	RSDE [17]	GSKD	GMM
kernel type	fixed Gaussian, $\gamma_{\mathrm{Parz}} = 0.17$	fixed Gaussian, $\gamma = 0.3$	fixed Gaussian, $\gamma = 0.56$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^2$	4.1194 ± 1.3513	4.1886 ± 1.3457	5.8163 ± 0.8358	3.9531 ± 1.4166	4.9829 ± 1.5762
KLD$\times 10^2$	4.4778 ± 2.7741	8.2111 ± 11.282	6.9556 ± 2.5217	6.1627 ± 4.3911	5.6345 ± 3.3672
kernel no.	200	10.2 ± 1.7	14.0 ± 4.3	6.5 ± 1.4	7
maximum	200	15	32	10	7
minimum	200	5	6	3	7

region $[a,\ b]^m$ and the values of γ^2_{ini} and/or γ^2_{min} were rechosen until all the N_{run} of the EM algorithm were converged.

A. First 1-D Example

In this 1-D example, the density to be estimated was the mixture of eight Gaussian distributions, which is given by

$$p(x) = \frac{1}{8}\sum_{i=0}^{7} \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}} \tag{51}$$

with

$$\sigma_i = \sqrt{\left(\frac{2}{3}\right)^i}, \quad \mu_i = 3\left(\left(\frac{2}{3}\right)^i - 1\right), \qquad 0 \le i \le 7. \tag{52}$$

The number of data points for density estimation was $N = 200$. The experiment was repeated $N_{\mathrm{run}} = 200$ times. The five density estimates were obtained, and they were the proposed GSKD estimate based on the tunable-kernel model, the GMM estimate, the PW estimate, our previous SKD estimate [20], and the RSDE [17]. The last three estimators were based on the fixed-kernel model. The average model size obtained for the GSKD estimate was 6.5, and therefore, we used $N_K = 7$ for the GMM. After considerable experiments, all the $N_{\mathrm{run}} = 200$ runs of the EM algorithm converged with the initialization $[a,\ b] = [-4,\ 3]$, $\gamma^2_{\mathrm{ini}} = 0.1$, and $\gamma^2_{\mathrm{min}} = 0.01$.

Table I compares the performance of the five density estimates, in terms of the L_1 test error and the KLD, as well as the number of kernels required. In addition, the maximum and minimum numbers of kernels over 200 runs are also listed in Table I for each density estimator. For this example, the PW estimate achieved the best test performance in terms of the KLD value, but the PW estimator is a nonsparse method, and the number of kernels was equal to the number of the training data points. It can be seen that the proposed GSKD and GMM estimators did well, both achieving better test performance with a more parsimonious KD estimate, as compared with the other two fixed-kernel SKD estimators of [17] and [20]. Figs. 1–5 depict the PW estimate, our previous SKD estimate [20], the RSDE [17], the proposed GSKD estimate, and the GMM estimate obtained in a typical run, respectively, in comparison with the true density distribution. The RWBS algorithmic parameters were set to $P_S = 10$, $M_I = 200$, and $N_G = 10$. The average complexity of the GSKD estimator is given in Table II, in comparison with that of the previous SKD estimator when the kernel width γ was chosen [20].

B. Second 1-D Example

The density to be estimated was the mixture of Gaussian and Laplacian distributions, which is defined by

$$p(x) = \frac{1}{2\sqrt{2\pi}} e^{-\frac{(x-2)^2}{2}} + \frac{0.7}{4} e^{-0.7|x+2|}. \tag{53}$$

The number of data points for density estimation was $N = 100$, and the experiment was repeated $N_{\mathrm{run}} = 100$ times. The number of Gaussian mixture components for the GMM was set to $N_K = 5$, as the average model size obtained for the GSKD estimate was found to be 4.5. After several tries, the appropriate initialization was found to be $[a,\ b] = [-12,\ 7]$, $\gamma^2_{\mathrm{ini}} = 0.1$, and $\gamma^2_{\mathrm{min}} = 0.01$ for the EM algorithm.

Table III lists the performance of the five density estimators, in terms of the L_1 test error and the KLD, as well as the number of kernels required. Figs. 6–10 plot a typical PW estimate obtained, a typical SKD estimate constructed, a typical RSDE obtained, a typical GSKD estimate obtained, and a typical GMM estimate obtained, in comparison with the true density. For this example, it can be seen that the proposed GSKD estimator and the RSDE achieved the best test performance, whereas the GMM estimator had the worst test performance, which was likely due to the local minimum problem. The proposed GSKD estimator achieved the most compact estimate. The GSKD estimator, however, employed the RWBS with the algorithmic parameters $P_S = 10$, $M_I = 200$, and $N_G = 10$, and its average complexity was significantly higher than that of the previous SKD estimator [20] when the kernel width γ was chosen, as can be seen from Table IV. The computational complexity of the RSDE [17] when the kernel width γ was chosen was of course similar to that of the SKD estimator [20].

C. 2-D Example

The density to be estimated was defined by the mixture of Gaussian and Laplacian distributions, which is given as follows:

$$p(x_1, x_2) = \frac{1}{4\pi} e^{-\frac{(x_1-2)^2}{2}} e^{-\frac{(x_2-2)^2}{2}} + \frac{0.35}{8} e^{-0.7|x_1+2|} e^{-0.5|x_2+2|}. \tag{54}$$

The estimation data set contained $N = 500$ samples, and the experiment was repeated $N_{\mathrm{run}} = 100$ times. Because we had an average model size of 7.2 for the GSKD estimate, $N_K = 8$ was used for the GMM. With considerable efforts, the appropriate initialization was found to be $[a,\ b]^2 = [-8,\ 8]^2$, $\gamma^2_{\mathrm{ini}} = 0.4$, and $\gamma^2_{\mathrm{min}} = 0.01$ for the EM algorithm to converge in all the $N_{\mathrm{run}} = 100$ runs.

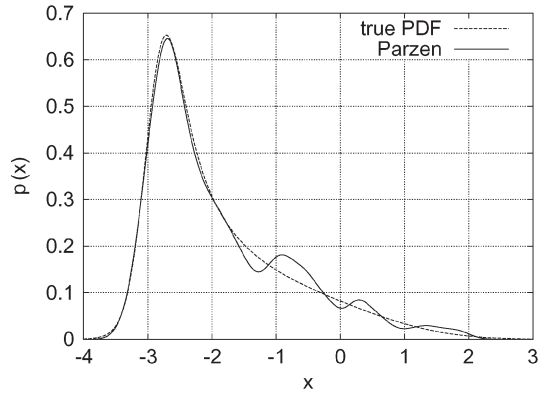

Fig. 1. (Solid) PW estimate in comparison with (dashed) true density for the 1-D example of the eight-Gaussian mixture.

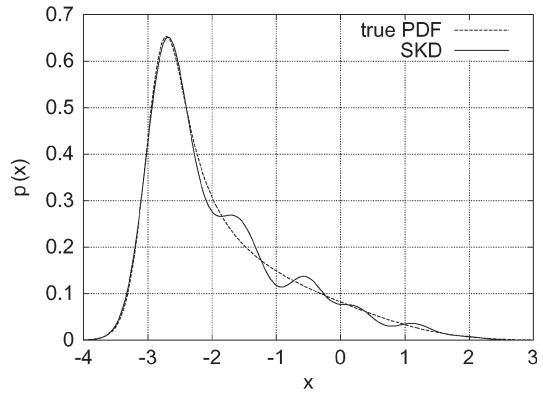

Fig. 2. (Solid) SKD estimate [20] in comparison with (dashed) true density for the 1-D example of the eight-Gaussian mixture.

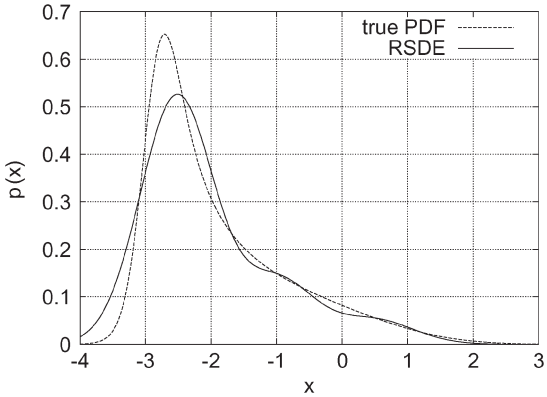

Fig. 3. (Solid) RSDE [17] in comparison with (dashed) true density for the 1-D example of the eight-Gaussian mixture.

Table V lists the L_1 test errors and the KLD values, as well as the numbers of kernels required for the five density estimates, namely, the PW estimate, the SKD estimate [20], and the RSDE [17] with the standard fixed-kernel model, as well as the proposed GSKD estimate and the GMM estimate with the tunable-kernel model. For this example, the GMM estimator achieved the best test performance. The proposed tunable KD estimator and the fixed-kernel RSDE also did well. The proposed GSKD estimator again had the most compact model with an average model size less than half of the RSDE. Table VI compares the average complexity of the GSKD estimator with

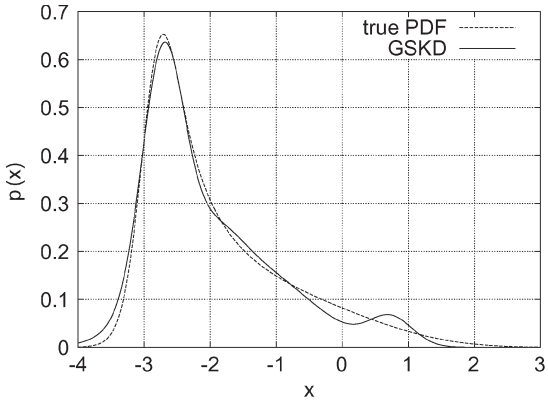

Fig. 4. (Solid) Proposed GSKD estimate in comparison with (dashed) true density for the 1-D example of the eight-Gaussian mixture.

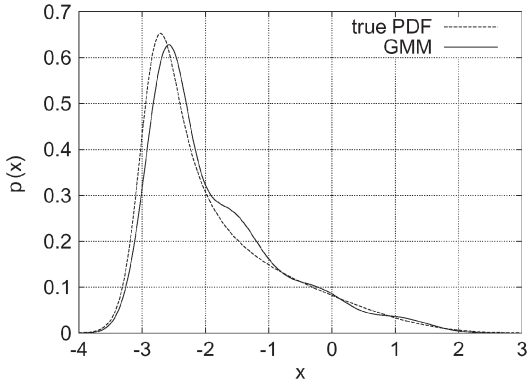

Fig. 5. (Solid) GMM estimate in comparison with (dashed) true density for the 1-D example of the eight-Gaussian mixture.

TABLE II
AVERAGE COMPLEXITY COMPARISON OF THE PREVIOUS SKD ESTIMATOR [20] WITH THE GIVEN γ AND THE PROPOSED GSKD ESTIMATOR FOR THE 1-D EXAMPLE OF THE EIGHT-GAUSSIAN MIXTURE

Data size	SKD [20]	GSKD
$N = 200$	$2040 \times O(200)$	$26650 \times O(200)$

the algorithmic parameters of $P_S = 20$, $M_I = 200$, and $N_G = 10$ with that of the SKD estimator [20] when the kernel width γ was chosen.

D. 6-D Example

In this 6-D example, the underlying density to be estimated was given by the mixture of three Gaussian distributions

$$p(\mathbf{x}) = \frac{1}{3}\sum_{i=1}^{3} \frac{1}{(2\pi)^{6/2}} \frac{1}{\det^{1/2}|\bar{\mathbf{\Gamma}}_i|} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \bar{\mathbf{\Gamma}}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)} \quad (55)$$

with

$$\boldsymbol{\mu}_1 = [1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0]^T$$
$$\bar{\mathbf{\Gamma}}_1 = \mathrm{diag}\{1.0, 2.0, 1.0, 2.0, 1.0, 2.0\} \quad (56)$$
$$\boldsymbol{\mu}_2 = [-1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0]^T$$
$$\bar{\mathbf{\Gamma}}_2 = \mathrm{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\} \quad (57)$$
$$\boldsymbol{\mu}_3 = [0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0]^T$$
$$\bar{\mathbf{\Gamma}}_3 = \mathrm{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}. \quad (58)$$

TABLE III
PERFORMANCE OF THE PW ESTIMATOR, PREVIOUS SKD ESTIMATOR [20], RSDE [17], PROPOSED GSKD ESTIMATOR, AND GMM ESTIMATOR IN TERMS OF L_1 TEST ERROR AND KLD, AS WELL AS NUMBER OF KERNELS REQUIRED FOR THE 1-D EXAMPLE OF THE GAUSSIAN AND LAPLACIAN MIXTURE, QUOTED AS MEAN $\pm$ STANDARD DEVIATION, OVER 100 RUNS

estimator	PW	SKD [20]	RSDE [17]	GSKD	GMM
kernel type	fixed Gaussian, $\gamma_{\mathrm{Parz}} = 0.54$	fixed Gaussian, $\gamma = 1.1$	fixed Gaussian, $\gamma = 1.3$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^2$	1.9963 ± 0.6179	2.0213 ± 0.6535	2.1377 ± 0.4669	1.9517 ± 0.6702	2.4597 ± 0.8117
KLD$\times 10^2$	8.0003 ± 5.1662	8.1419 ± 5.0102	6.1213 ± 3.1425	7.0446 ± 5.1226	12.7724 ± 9.5317
kernel no.	100	5.1 ± 1.2	7.9 ± 3.2	4.5 ± 0.9	5
maximum	100	7	24	7	5
minimum	100	2	3	2	5

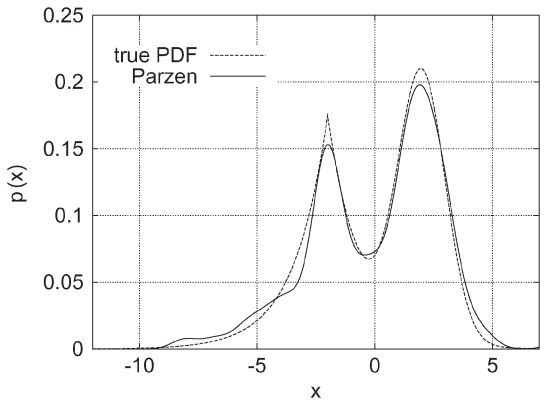

Fig. 6. (Solid) PW estimate in comparison with (dashed) true density for the 1-D example of the Gaussian and Laplacian mixture.

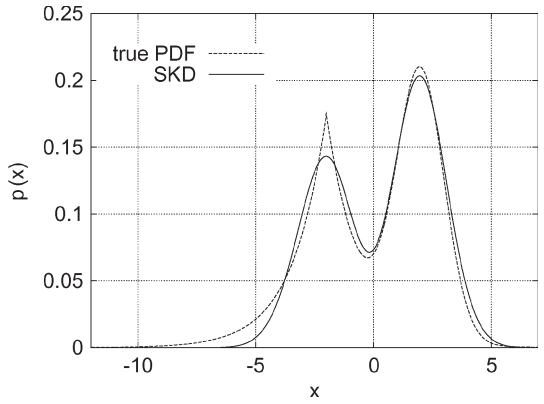

Fig. 7. (Solid) SKD estimate [20] in comparison with (dashed) true density for the 1-D example of the Gaussian and Laplacian mixture.

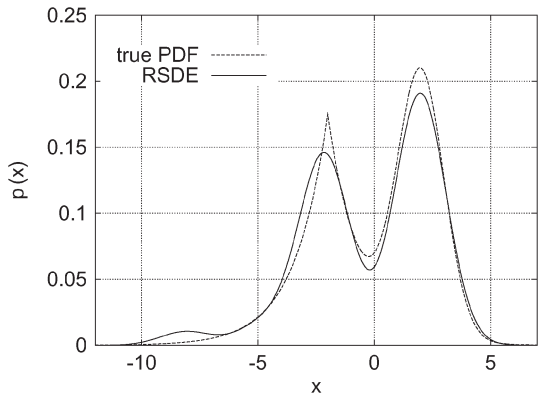

Fig. 8. (Solid) RSDE [17] in comparison with (dashed) true density for the 1-D example of the Gaussian and Laplacian mixture.

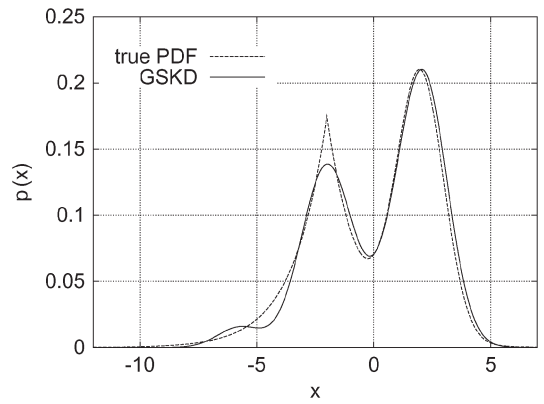

Fig. 9. (Solid) Proposed GSKD estimate in comparison with (dashed) true density for the 1-D example of the Gaussian and Laplacian mixture.

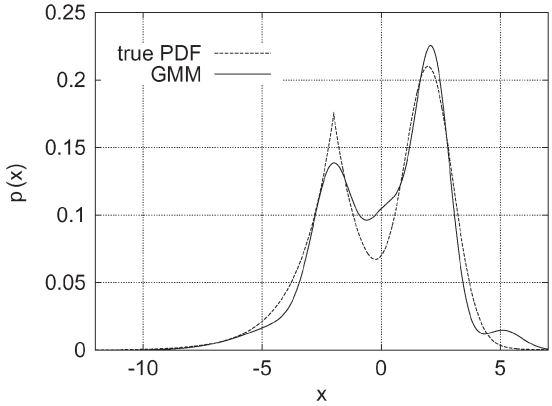

Fig. 10. (Solid) GMM estimate in comparison with (dashed) true density for the 1-D example of the Gaussian and Laplacian mixture.

TABLE IV
AVERAGE COMPLEXITY COMPARISON OF THE PREVIOUS SKD ESTIMATOR [20] WITH THE GIVEN γ AND THE PROPOSED GSKD ESTIMATOR FOR THE 1-D EXAMPLE OF THE GAUSSIAN AND LAPLACIAN MIXTURE

Data size	SKD [20]	GSKD
$N = 100$	$510 \times O(100)$	$18450 \times O(100)$

The estimation data set was set to $N = 50, 600$, and 1000 samples, respectively, and the experiment was repeated $N_{\mathrm{run}} = 100$ times. For the GMM estimator, $N_K = 4$, 5, and 7 were used for the cases of $N = 50, 600$, and 1000, respectively, whereas the appropriate initialization for the EM algorithm was found to be $[a,\ b]^6 = [-5,\ 5]^6$, $\gamma^2_{\mathrm{ini}} = 0.1$, and $\gamma^2_{\mathrm{min}} = 0.01$. The GSKD estimator had the RWBS algorithmic parameters of $P_S = 40$, $M_I = 400$, and $N_G = 10$.

TABLE V
PERFORMANCE OF THE PW ESTIMATOR, PREVIOUS SKD ESTIMATOR [20], RSDE [17], PROPOSED GSKD ESTIMATOR, AND GMM ESTIMATOR IN TERMS OF L_1 TEST ERROR AND KLD, AS WELL AS NUMBER OF KERNELS REQUIRED FOR THE 2-D EXAMPLE OF THE GAUSSIAN AND LAPLACIAN MIXTURE, QUOTED AS MEAN ± STANDARD DEVIATION, OVER 100 RUNS

estimator	PW	SKD [20]	RSDE [17]	GSKD	GMM
kernel type	fixed Gaussian, $\gamma_{\rm Parz} = 0.42$	fixed Gaussian, $\gamma = 1.1$	fixed Gaussian, $\gamma = 1.2$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^3$	4.0358 ± 0.6925	3.8379 ± 0.7797	4.0533 ± 0.4460	3.7238 ± 0.8046	3.1573 ± 0.8909
KLC $\times 10$	1.4661 ± 0.2281	1.4028 ± 0.5337	0.8961 ± 0.4113	1.3628 ± 0.5197	0.4824 ± 0.1297
kernel no.	500	15.3 ± 3.9	16.2 ± 3.4	7.2 ± 1.3	8
maximum	500	25	24	9	8
minimum	500	8	9	4	8

TABLE VI
AVERAGE COMPLEXITY COMPARISON OF THE PREVIOUS SKD ESTIMATOR [20] WITH THE GIVEN γ AND THE PROPOSED GSKD ESTIMATOR FOR THE 2-D EXAMPLE OF THE GAUSSIAN AND LAPLACIAN MIXTURE

Data size	SKD [20]	GSKD
$N = 100$	$7650 \times O(500)$	$30240 \times O(500)$

The results obtained by the five density estimators are summarized in Table VII, where it can be seen that the test performance of the various estimators for the case of $N = 50$ was much poorer than the cases of $N = 600$ and 1000. This was not surprising as the estimation set of $N = 50$ was insufficient to achieve an accurate estimate. For the cases of $N = 600$ and 1000, the GMM estimator achieved the best test performance, and it can also be seen that the proposed GSKD estimator based on the tunable-kernel model structure had clear advantages over the standard SKD estimator based on the fixed-kernel model structure, in terms of achievable test performance and estimator model size.

Average computational requirements of the GSKD estimator and our previous fixed-kernel SKD estimator [20] when the kernel variance γ^2 was chosen are compared in Table VIII. The computational complexity of the RSDE [17] with a given kernel variance γ^2 was well known to be similar to that of the SKD estimator [20].

E. 10-D Example

The underlying density of this 10-D example was the mixture of three Gaussian distributions, which took the form of (55) with

$$\boldsymbol{\mu}_1 = [1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0]^T$$
$$\bar{\boldsymbol{\Gamma}}_1 = \text{diag}\{1.0, 2.0, 1.0, 2.0, 1.0, 2.0, 1.0, 2.0, 1.0, 2.0\} \tag{59}$$
$$\boldsymbol{\mu}_2 = [-1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0 \quad -1.0]^T$$
$$\bar{\boldsymbol{\Gamma}}_2 = \text{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0, 2.0, 1.0, 2.0, 1.0\} \tag{60}$$
$$\boldsymbol{\mu}_3 = [0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0]^T$$
$$\bar{\boldsymbol{\Gamma}}_3 = \text{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}. \tag{61}$$

The estimation set contained $N = 20$ samples, and the experiment was repeated $N_{\rm run} = 100$ times. This example was specifically designed to represent the situation where the number of data samples is very small in comparison with the dimension of the density space, as 20 samples are extremely sparse in the 10-D space. The GMM estimator had $N_K = 4$ mixture components, whereas the EM algorithm had the parameters $[a,\ b]^{10} = [-5,\ 5]^{10}$, $\gamma^2_{\rm ini} = 0.2$, and $\gamma^2_{\rm min} = 0.02$. The GSKD estimator had the RWBS algorithmic parameters of $P_S = 40$, $M_I = 400$, and $N_G = 10$. The results obtained by the five density estimators are compared in Table IX, where it can be seen that the three fixed-kernel estimators and the proposed GSKD estimator based on the tunable-kernel model structure had similar test performance, but the GSKD estimator achieved a smaller estimator model size.

V. CONCLUSION

A novel algorithm has been proposed for constructing the GSKD estimator with tunable-kernel units. Unlike the existing sparse SKD estimators, the kernel center vectors are not restricted to the training data samples, and each kernel unit has an individually adjusted diagonal covariance matrix. On the other hand, we do not optimize all the kernel model's parameters together using nonlinear optimization, as the conventional FMM would. Rather, we optimize the kernel units one by one by minimizing the LOO test MSE based on an OFR procedure. A modified version of the MNQP algorithm is used to compute the kernel weights of the constructed GSKD estimator to guarantee the nonnegative and unity constraints for the mixture coefficients. Five illustrative examples have been included to demonstrate the advantages of the proposed GSKD estimator based on the tunable-kernel model over the standard SKD estimator based on the fixed-kernel model, in terms of achievable test performance and estimator model size. In terms of computational requirements, the SKD estimator with the fixed-kernel model typically imposes several times lower complexity than the GSKD estimator with the tunable-kernel model. However, when the complexity of tuning the kernel variance required by the fixed-kernel model is taken into account, the computational advantage of the fixed-kernel approach over the proposed tunable-kernel approach may be diminished. The results obtained have also suggested that the proposed GSKD estimator has similar test performance to that of the GMM.

APPENDIX
POSITIONING AND SHAPING GENERALIZED KERNEL UNITS

Let $\mathbf{u}$ be the vector that contains $\mathbf{c}_n$ and $\boldsymbol{\Gamma}_n$. Give the following initial conditions:

$$\left.\begin{array}{ll} \epsilon^{(0)}(k) = t_k \text{ and } \eta^{(0)}(k) = 1, & 1 \le k \le N \\ J_0 = \frac{1}{N}\mathbf{t}^T\mathbf{t} = \frac{1}{N}\sum_{k=1}^{N} t_k^2 & \end{array}\right\}. \tag{62}$$

TABLE VII
PERFORMANCE OF THE PW ESTIMATOR, PREVIOUS SKD ESTIMATOR [20], RSDE [17], PROPOSED GSKD ESTIMATOR, AND GMM ESTIMATOR IN TERMS OF L_1 TEST ERROR AND NUMBER OF KERNELS REQUIRED FOR THE 6-D EXAMPLE OF THE THREE-GAUSSIAN MIXTURE, QUOTED AS MEAN $\pm$ STANDARD DEVIATION, OVER 100 RUNS

estimator	PW	SKD [20]	RSDE [17]	GSKD	GMM
$N = 50$ case					
kernel type	fixed Gaussian, $\gamma_{\text{Parz}} = 0.9$	fixed Gaussian, $\gamma = 1.2$	fixed Gaussian, $\gamma = 1.2$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^5$	5.0431 ± 0.4071	4.7748 ± 0.4973	4.6236 ± 0.5680	4.8211 ± 0.4346	5.1407 ± 0.9425
kernel no.	50	12.8 ± 2.4	14.9 ± 4.5	3.2 ± 1.3	4
maximum	50	21	30	10	4
minimum	50	8	7	1	4
$N = 600$ case					
kernel type	fixed Gaussian, $\gamma_{\text{Parz}} = 0.65$	fixed Gaussian, $\gamma = 1.2$	fixed Gaussian, $\gamma = 1.2$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^5$	3.5195 ± 0.1616	3.1134 ± 0.5335	2.7388 ± 0.4998	2.5624 ± 0.2945	1.5309 ± 0.2995
kernel no.	600	9.4 ± 1.9	14.2 ± 3.6	5.0 ± 1.0	5
maximum	600	16	25	8	5
minimum	600	7	8	3	5
$N = 1000$ case					
kernel type	fixed Gaussian, $\gamma_{\text{Parz}} = 0.65$	fixed Gaussian, $\gamma = 1.2$	fixed Gaussian, $\gamma = 1.2$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^5$	3.2710 ± 0.1503	2.9484 ± 0.4472	2.4523 ± 0.3712	2.4209 ± 0.1882	1.2294 ± 0.2035
kernel no.	1000	13.5 ± 2.9	13.9 ± 3.7	6.9 ± 0.9	7
maximum	1000	23	28	8	7
minimum	1000	8	6	4	7

TABLE VIII
AVERAGE COMPLEXITY COMPARISON OF THE PREVIOUS SKD ESTIMATOR [20] WITH THE GIVEN γ AND THE PROPOSED GSKD ESTIMATOR FOR THE 6-D EXAMPLE OF THE THREE-GAUSSIAN MIXTURE

Data size	SKD [20]	GSKD
$N = 50$	$640 \times O(50)$	$26880 \times O(50)$
$N = 600$	$5640 \times O(600)$	$42000 \times O(600)$
$N = 1000$	$13500 \times O(1000)$	$57960 \times O(1000)$

Specify the RWBS algorithmic parameters: the population size P_S, the number of generations in the repeated search N_G, and the number of WBS iterations M_I.

Outer loop: generations For $(l = 1; l \le N_G; l = l + 1)\{$

- *Generation initialization:* Initialize the population by setting $\mathbf{u}_1^{[l]} = \mathbf{u}_{\text{best}}^{[l-1]}$ and randomly generating the rest of the population members $\mathbf{u}_i^{[l]}$, $2 \le i \le P_S$, where $\mathbf{u}_{\text{best}}^{[l-1]}$ denotes the solution found in the previous generation. If $l = 1$, $\mathbf{u}_1^{[l]}$ is also randomly chosen.
- *WBS initialization:* Assign the initial distribution weightings $\delta_i(0) = 1/P_S$, $1 \le i \le P_S$, for the population. Then
  1) For $1 \le i \le P_S$, generate $\boldsymbol{\phi}_n^{i)}$ from $\mathbf{u}_i^{[l]}$, the candidates for the nth model column, and orthogonalize them using the Gram–Schmidt orthogonalization procedure [38]:

$$\alpha_{j,n}^{i)} = \frac{\mathbf{w}_j^T \boldsymbol{\phi}_n^{i)}}{\mathbf{w}_j^T \mathbf{w}_j}, \qquad 1 \le j < n \tag{63}$$

$$\mathbf{w}_n^{i)} = \boldsymbol{\phi}_n^{i)} - \sum_{j=1}^{n-1} \alpha_{j,n}^{i)} \mathbf{w}_j \tag{64}$$

$$g_n^{i)} = \frac{\left(\mathbf{w}_n^{i)}\right)^T \mathbf{t}}{\left(\mathbf{w}_n^{i)}\right)^T \mathbf{w}_n^{i)} + \lambda}. \tag{65}$$

  2) For $1 \le i \le P_S$, calculate the LOO cost function value of each $\mathbf{u}_i^{[l]}$:

$$\epsilon_i^{(n)}(k) = \epsilon^{(n-1)}(k) - w_{k,n}^{i)} g_n^{i)}, \qquad 1 \le k \le N \tag{66}$$

$$\eta_i^{(n)}(k) = \eta^{(n-1)}(k) - \frac{\left(w_{k,n}^{i)}\right)^2}{\left(\mathbf{w}_n^{i)}\right)^T \mathbf{w}_n^{i)} + \lambda}, \qquad 1 \le k \le N \tag{67}$$

$$J_n^{i)} = \frac{1}{N} \sum_{k=1}^{N} \left(\frac{\epsilon_i^{(n)}(k)}{\eta_i^{(n)}(k)} \right)^2 \tag{68}$$

where $w_{k,n}^{i)}$ is the kth element of $\mathbf{w}_n^{i)}$.

Inner loop: WBS For $(t = 1; t \le M_I; t = t + 1)\{$

- *Step 1: Boosting*
  1) Find

$$i_{\text{best}} = \arg \min_{1 \le i \le P_S} J_n^{i)} \quad \text{and} \quad i_{\text{worst}} = \arg \max_{1 \le i \le P_S} J_n^{i)}.$$

  2) Denote $\mathbf{u}_{\text{best}}^{[l]} = \mathbf{u}_{i_{\text{best}}}^{[l]}$ and $\mathbf{u}_{\text{worst}}^{[l]} = \mathbf{u}_{i_{\text{worst}}}^{[l]}$.
  3) Normalize the cost function values

$$\bar{J}_n^{i)} = \frac{J_n^{i)}}{\sum_{j=1}^{P_S} J_n^{j)}}, \qquad 1 \le i \le P_S.$$

  4) Compute a weighting factor μ_t according to

$$\xi_t = \sum_{i=1}^{P_S} \delta_i(t-1) \bar{J}_n^{i)} \quad \mu_t = \frac{\xi_t}{1 - \xi_t}.$$

  5) Update the distribution weightings for $1 \le i \le P_S$

$$\tilde{\delta}_i(t) = \begin{cases} \delta_i(t-1) \mu_t^{\bar{J}_n^{i)}}, & \text{for } \mu_t \le 1 \\ \delta_i(t-1) \mu_t^{1-\bar{J}_n^{i)}}, & \text{for } \mu_t > 1 \end{cases}$$

  and normalize them

$$\delta_i(t) = \frac{\tilde{\delta}_i(t)}{\sum_{j=1}^{P_S} \tilde{\delta}_j(t)}, \qquad 1 \le i \le P_S.$$

TABLE IX
PERFORMANCE OF THE PW ESTIMATOR, PREVIOUS SKD ESTIMATOR [20], RSDE [17], PROPOSED GSKD ESTIMATOR, AND GMM ESTIMATOR IN TERMS OF L_1 TEST ERROR AND NUMBER OF KERNELS REQUIRED FOR THE 10-D EXAMPLE OF THE THREE-GAUSSIAN MIXTURE, QUOTED AS MEAN $\pm$ STANDARD DEVIATION, OVER 100 RUNS

estimator	PW	SKD [20]	RSDE [17]	GSKD	GMM
kernel type	fixed Gaussian, $\gamma_{\text{Parz}} = 1.1$	fixed Gaussian, $\gamma = 1.1$	fixed Gaussian, $\gamma = 1.1$	tunable Gaussian	tunable Gaussian
L_1 test error$\times 10^7$	1.9488 ± 0.0735	1.9273 ± 0.0660	1.8818 ± 0.0852	1.8953 ± 0.0513	2.3641 ± 0.6243
kernel no.	20	10.0 ± 1.2	19.4 ± 2.8	3.7 ± 0.9	4
maximum	20	14	20	6	4
minimum	20	7	5	3	4

- *Step 2: Parameter updating*
  1) Construct the $(P_S + 1)$th point using the formula

$$\mathbf{u}_{P_S+1} = \sum_{i=1}^{P_S} \delta_i(t)\mathbf{u}_i^{[l]}.$$

  2) Construct the $(P_S + 2)$th point using the formula

$$\mathbf{u}_{P_S+2} = \mathbf{u}_{\text{best}}^{[l]} + \left(\mathbf{u}_{\text{best}}^{[l]} - \mathbf{u}_{P_S+1}\right).$$

  3) Calculate $\boldsymbol{\phi}_n^{P_S+1)}$ and $\boldsymbol{\phi}_n^{P_S+2)}$ from $\mathbf{u}_{P_S+1}$ and $\mathbf{u}_{P_S+2}$, orthogonalize these two candidate model columns [as in (63)–(65)], and compute their corresponding LOO cost function values $J_n^{i)}$, $i = P_S + 1, P_S + 2$ [as in (66)–(68)]. Then, find

$$i_* = \arg \min_{i=P_S+1,P_S+2} J_n^{i)}.$$

  The pair $(\mathbf{u}_{i_*}, J_n^{i_*)})$ then replaces $(\mathbf{u}_{\text{worst}}^{[l]}, J_n^{i_{\text{worst}})})$ in the population.

} **End of inner loop**

The solution found in the lth generation is $\mathbf{u} = \mathbf{u}_{\text{best}}^{[l]}$.

} **End of outer loop**

This yields the solution $\mathbf{u} = \mathbf{u}_{\text{best}}^{[N_G]}$, i.e., $\mathbf{c}_n$ and $\boldsymbol{\Gamma}_n$ of the nth kernel unit, the nth model column $\boldsymbol{\phi}_n$, the orthogonalization coefficients $\alpha_{j,n}$, $1 \le j < n$, the corresponding orthogonal model column $\mathbf{w}_n$, and the weight g_n, as well as the n-term modeling errors $\epsilon^{(n)}(k)$ and associated LOO modeling error weightings $\eta^{(n)}(k)$ for $1 \le k \le N$.

The idea of the RWBS algorithm is remarkably simple. The basic process, i.e., the inner WBS loop, evolves a population of P_S initially randomly chosen solutions by performing a convex combination of the potential solutions to yield $\mathbf{u}_{P_S+1}$ and computing its mirror image $\mathbf{u}_{P_S+2}$. The worst member of the population is then replaced by either $\mathbf{u}_{P_S+1}$ or $\mathbf{u}_{P_S+2}$. This process is repeated M_I times until the process converges. The weightings $\delta_i(t)$ used in the convex combination are adapted to reflect the "goodness" of corresponding potential solutions using the idea from boosting [39]–[42]. The inner iteration loop is designed to efficiently find a minimum point within the convex hull defined by the initial population members. This capability as a local optimizer can be explained by the theory of weak learnability associated with boosting [39], [40]. The outer loop repeats the inner WBS N_G times or "generations" to improve the probability of finding a global optimal solution. An elitist strategy is adopted by retaining the best solution found in the current generation in the initial population of the next generation, which ensures that the information obtained with regard to the previous search region is not lost. By repeating the WBS a number of generations, the algorithm resembles a commonly used random search strategy, which is called the multistart [43]. Note that randomly drawing a number of points adopted by the RWBS algorithm is also the sampling strategy used in a class of global optimization methods, which is referred to as clustering [43].

The appropriate values for the RWBS algorithmic parameters P_S, N_G, and M_I depend on the dimension of $\mathbf{u}$ and how hard the objective function to be optimized. Generally, these algorithmic parameters have to empirically be found, and some general rules are discussed in [28]. In the inner optimization loop, there is no need for every member of the population to converge to a (local) minimum, and it is sufficient to locate where the minimum lies. Thus, the maximum number of iterations M_I for the inner optimization loop can be set to a relatively small value. This makes the search efficient, achieving convergence with a small number of the cost function evaluations. The population size P_S and the number of generations N_G should be set sufficiently large so that the parameter space will sufficiently be sampled.

REFERENCES

[1] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[2] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

[3] B. W. Silverman, *Density Estimation*. London, U.K.: Chapman & Hall, 1996.

[4] H. Wang, "Robust control of the output probability density functions for multivariable stochastic systems with guaranteed stability," *IEEE Trans. Autom. Control*, vol. 44, no. 11, pp. 2103–2107, Nov. 1999.

[5] S. Chen, A. K. Samingan, B. Mulgrew, and L. Hanzo, "Adaptive minimum-BER linear multiuser detection for DS-CDMA signals in multipath channels," *IEEE Trans. Signal Process.*, vol. 49, no. 6, pp. 1240–1247, Jun. 2001.

[6] G. McLachlan and D. Peel, *Finite Mixture Models*. New York: Wiley, 2000.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Stat. Soc. B*, vol. 39, no. 1, pp. 1–38, 1977.

[8] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," Univ. California, Berkeley, Tech. Rep., ICSI-TR-97-021, 1997.

[9] B. Efron and R. J. Tibshirani, *An Introduction to Bootstrap*. London, U.K.: Chapman & Hall, 1993.

[10] Z. R. Yang and S. Chen, "Robust maximum likelihood training of heteroscedastic probabilistic neural networks," *Neural Netw.*, vol. 11, no. 4, pp. 739–747, Jun. 1998.

[11] M. Svensén and C. M. Bishop, "Robust Bayesian mixture modelling," *Neurocomputing*, vol. 64, pp. 235–252, Mar. 2005.

[12] C. Archambeau and M. Verleysen, "Robust Bayesian clustering," *Neural Netw.*, vol. 20, no. 1, pp. 129–138, Jan. 2007.

[13] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Stat.*, vol. 33, no. 3, pp. 1066–1076, 1962.

[14] J. Weston, A. Gammerman, M. O. Stitson, V. Vapnik, V. Vovk, and C. Watkins, "Support vector density estimation," in *Advances in*

Kernel Methods—Support Vector Learning, B. Schölkopf, C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, pp. 293–306.
[15] S. Mukherjee and V. Vapnik, "Support vector method for multivariate density estimation," MIT AI Lab, Cambridge, MA, Tech. Rep., A.I. Memo No. 1653, 1999.
[16] V. Vapnik and S. Mukherjee, "Support vector method for multivariate density estimation," in *Advances in Neural Information Processing Systems*, S. Solla, T. Leen, and K. R. Müller, Eds. Cambridge, MA: MIT Press, 2000, pp. 659–665.
[17] M. Girolami and C. He, "Probability density estimation from optimally condensed data samples," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1253–1264, Oct. 2003.
[18] A. Choudhury, "Fast machine learning algorithms for large data," Ph.D. dissertation, Comput. Eng. Design Center, School Eng. Sci., Univ. Southampton, Southampton, U.K., 2002.
[19] S. Chen, X. Hong, and C. J. Harris, "Sparse kernel density construction using orthogonal forward regression with leave-one-out test score and local regularization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 4, pp. 1708–1717, Aug. 2004.
[20] S. Chen, X. Hong, and C. J. Harris, "An orthogonal forward regression techniques for sparse kernel density estimation," *Neurocomputing*, vol. 71, no. 4–6, pp. 931–943, Jan. 2008.
[21] F. Sha, L. K. Saul, and D. D. Lee, "Multiplicative updates for nonnegative quadratic programming in support vector machines," Univ. Pennsylvania, Philadelphia, PA, Tech. Rep. MS-CIS-02-19, 2002.
[22] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[23] K. F. Man, K. S. Tang, and S. Kwong, *Genetic Algorithms: Concepts and Design*. London, U.K.: Springer-Verlag, 1998.
[24] L. Ingber, "Simulated annealing: Practice versus theory," *Math. Comput. Model.*, vol. 18, no. 11, pp. 29–57, 1993.
[25] S. Chen and B. L. Luk, "Adaptive simulated annealing for optimization in signal processing applications," *Signal Process.*, vol. 79, no. 1, pp. 117–128, Nov. 1999.
[26] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Piscataway, NJ, Nov. 27–Dec. 1, 1995, vol. 4, pp. 1942–1948.
[27] J. Kennedy, R. C. Eberhart, and Y.-H. Shi, *Swarm Intelligence*. San Mateo, CA: Morgan Kaufmann, 2001.
[28] S. Chen, X. X. Wang, and C. J. Harris, "Experiments with repeating weighted boosting search for optimization in signal processing applications," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 4, pp. 682–693, Aug. 2005.
[29] X. Hong, P. M. Sharkey, and K. Warwick, "Automatic nonlinear predictive model construction algorithm using forward regression and the PRESS statistic," *Proc. Inst. Elect. Eng.—Control Theory Appl.*, vol. 150, no. 3, pp. 245–254, May 2003.
[30] S. Chen, X. Hong, C. J. Harris, and P. M. Sharkey, "Sparse modeling using orthogonal forward regression with PRESS statistic and regularization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 898–911, Apr. 2004.
[31] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.
[32] [Online]. Available: http://en.wikipedia.org/wiki/Glivenko-Cantelli_theorem
[33] R. H. Myers, *Classical and Modern Regression With Applications*, 2nd ed. Boston, MA: PWS-KENT, 1990.
[34] M. Stone, "Cross validation choice and assessment of statistical predictions," *J. R. Stat. Soc. B*, vol. 36, pp. 111–147, 1974.
[35] L. K. Hansen and J. Larsen, "Linear unlearning for cross-validation," *Adv. Comput. Math.*, vol. 5, no. 1, pp. 269–280, Dec. 1996.
[36] G. Monari and G. Dreyfus, "Local overfitting control via leverages," *Neural Comput.*, vol. 14, no. 6, pp. 1481–1506, Jun. 2002.
[37] S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset model selection," *IEEE Trans. Signal Process.*, vol. 43, no. 7, pp. 1713–1715, Jul. 1995.
[38] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their applications to non-linear system identification," *Int. J. Control*, vol. 50, no. 5, pp. 1873–1896, Nov. 1989.
[39] R. E. Schapire, "The strength of weak learnability," *Mach. Learn.*, vol. 5, no. 2, pp. 197–227, Jun. 1990.
[40] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.
[41] L. Breiman, "Prediction games and arcing algorithms," *Neural Comput.*, vol. 11, no. 7, pp. 1493–1518, Oct. 1999.
[42] R. Meir and G. Rätsch, "An introduction to boosting and leveraging," in *Advanced Lectures in Machine Learning*, S. Mendelson and A. Smola, Eds. New York: Springer-Verlag, 2003, pp. 119–184.
[43] F. Schoen, "Stochastic techniques for global optimization: A survey of recent advances," *J. Global Optim.*, vol. 1, no. 3, pp. 207–228, Sep. 1991.

Sheng Chen (M'90–SM'97–F'08) received the Ph.D. degree in control engineering from the City University, London, U.K., in 1986 and the D.Sc. degree from the University of Southampton, Southampton, U.K., in 2004.

From October 1986 to August 1999, he held research and academic appointments with the University of Sheffield, Sheffield, U.K., the University of Edinburgh, Edinburgh, U.K., and the University of Portsmouth, Portsmouth, U.K. Since September 1999, he has been with the School of Electronics and Computer Science, University of Southampton. He has published more than 400 research papers. His research interests include wireless communications, adaptive signal processing for communications, machine learning, and evolutionary computation methods.

Dr. Chen is a Fellow of the Institution of Engineering and Technology. In the database of the world's most highly cited researchers, which is compiled by the Institute for Scientific Information of the U.S., he is on the list of the highly cited researchers in the engineering category.

Xia Hong (SM'02) received the B.Sc. and M.Sc. degrees from the National University of Defence Technology, Changsha, China, in 1984 and 1987, respectively, and the Ph.D. degree from the University of Sheffield, Sheffield, U.K., in 1998, all in automatic control.

She was a Research Assistant with Beijing Institute of Systems Engineering, Beijing, China, from 1987 to 1993. She was a Research Fellow with the School of Electronics and Computer Science, University of Southampton, Southampton, U.K., from 1997 to 2001. Since 2001, she has been with the School of Systems Engineering, University of Reading, Reading, U.K., where she currently holds a Readership post. She has authored or coauthored more than 180 research papers and has coauthored a research book. She is actively engaged in research into nonlinear systems identification, data modeling, estimation and intelligent control, neural networks, pattern recognition, learning theory, and their applications.

Dr. Hong was the recipient of the Donald Julius Groen Prize from the Institution of Mechanical Engineers (IMechE) in 1999.

Chris J. Harris received the Ph.D. and D.Sc. degrees from the University of Southampton, Southampton, U.K., in 1972 and 2001, respectively.

He previously held appointments with the University of Hull, Hull, U.K., the University of Manchester Institute of Science and Technology, Manchester, U.K., the University of Oxford, Oxford, U.K., and Cranfield University, Cranfield, U.K., as well as being employed by the U.K. Ministry of Defence. He returned to the University of Southampton as the Lucas Professor of Aerospace Systems Engineering in 1987 to establish the Advanced Systems Research Group and, more recently, the Image, Speech and Intelligent Systems Group. He has authored or coauthored 12 research books and more than 400 research papers. He is the Associate Editor of numerous international journals. His research interests are in the general area of intelligent and adaptive systems theory and its application to intelligent autonomous systems such as autonomous vehicles, management infrastructures such as command and control, intelligent control, and estimation of dynamic processes, multisensor data fusion, and systems integration.

Dr. Harris was elected to the Royal Academy of Engineering in 1996. He was the recipient of the IEE Senior Achievement Medal in 1998 for his work on autonomous systems and the IEE Faraday Medal in 2001 for his work on intelligent control and neurofuzzy systems.