A fast algorithm for sparse probability density function construction

Xia Hong
School of Systems Engineering
University of Reading
UK

Sheng Chen
School of Electronics and Computer Science
University of Southampton
UK

***Abstract*—A new sparse kernel density estimator is introduced. Our main contribution is to develop a recursive algorithm for the selection of significant kernels one at time using the minimum integrated square error (MISE) criterion for both kernel selection. The proposed approach is simple to implement and the associated computational cost is very low. Numerical examples are employed to demonstrate that the proposed approach is effective in constructing sparse kernel density estimators with competitive accuracy to existing kernel density estimators.**

***Index Terms*—probability density function, sparse modelling, minimum integrated square error**

I. Introduction

The probability density function (PDF) estimation problem is fundamental to many pattern recognition, data analysis and other engineering applications [1]–[5]. The celebrated Parzen window (PW) estimate [6] can be regarded as a special case of the finite mixture model [1], in which the number of mixtures is equal to that of the training data samples and all the mixing weights are equal. The point density estimate using the PW estimator for a future data sample can be computationally expensive if the number of training data samples is very large. The finite mixture model is based on fixing the number of mixtures, and applying the expectation-maximisation (EM) algorithm [7] to provide the maximum likelihood (ML) estimate of the mixture model's parameters. This associated ML optimisation, in general, is a highly nonlinear optimisation process requiring extensive computation. The EM algorithm for Gaussian mixture model enjoys an explicit iterative form [8]. However, it is known that this EM algorithm based ML estimation to be ill posed and has a slow convergence speed. To tackle the associated numerical difficulties, it is often required to apply resampling techniques [9], [10].

There is a considerable interest into research on the sparse PDF estimation. The support vector machine (SVM) density estimation technique has been proposed [11], [12]. The optimisation in the SVM method is to solve a constrained quadratic optimisation problem. This yields the sparsity inducing property, i.e. at the optimality, many kernels' weights are driven to zeros. Alternatively a novel regression-based PDF estimation method has been introduced [13], in which the empirical CDF is constructed, in the same manner as in the SVM density estimation approach, to be used as the desired response. The orthogonal forward regression (OFR) approach is an efficient supervised regression model construction method [14]. The OFR method has been combined with a leave-one-out test score and local regularisation [15], [16]. The regression-based idea of [13] and the approach in [15], [16] have been extended to yield a new OFR based sparse density estimation algorithm [17] with comparable performance to that of the PW estimate. In [13], [17], the regressors are the CDFs of the kernels and the target response is the empirical CDF. A simple and viable alternative approach has been proposed to use kernels directly as regressors by adopting the PW estimate as the target response [18].

The desirable property of sparsity inducing also happens in the interesting approach of reduced set density estimator (RSDE) [19], based on the minimisation of the integrated square error (ISE) between the estimator and the true density [2], [19], [20]. Two efficient optimisation algorithms were introduced for the RSDE that has a complexity of $\mathcal{O}(N^2)$ per iteration, where N is the number of data samples and $\mathcal{O}(M)$ denotes the order of M, compared to a standard quadratic optimisation solver at $\mathcal{O}(N^3)$. The complexity of the sparse density estimators [17], [18] is also $\mathcal{O}(N^2)$ scaled by the number of regressors selected, which is generally very small. Our extensive experience has shown that all the sparse density estimators [11], [12], [17]–[19] discussed here are capable of automatically producing sparse PDF estimates with comparable performance to that of the PW estimate, but the density estimators of [17]–[19] produce much sparser estimates than the SVM based density estimator.

Against this background, this paper introduces a new algorithm for sparse kernel density estimation based on the MISE and the forward constrained regression (FCR) [21]. In our proposed new sparse kernel density estimator, referred to as the FCR-MISE algorithm, a kernel term is selected one at a time which has the minimum ISE value among all the candidate kernels formed from the data points. Within the FCR framework, the mixing weights are computed using a recursion linking the weight for the newly selected kernel and the set of the mixing weights of the previous stages [21]. The proposed density estimation algorithm is very efficient due to the recursive computation and the closed-form solution of only one parameter per step. Specifically, the complexity of our proposed new algorithm is $\mathcal{O}(N)$ scaled by the squared number of kernels selected. Numerical examples are employed

to demonstrate that our new sparse kernel density estimator is capable of producing very sparse PDF estimates with comparable accuracy to those of the PW estimator and some other existing sparse kernel density estimators.

II. Fast sparse kernel density estimator construction algorithm

Given the finite data set $D_N = \{\boldsymbol{x}_j\}_{j=1}^N$ consisting of N data samples, where the data vector $\boldsymbol{x}_j \in \mathbb{R}^m$ follows an unknown PDF $p(\boldsymbol{x})$, the problem under study is to find a sparse approximation of $p(\boldsymbol{x})$ based on D_N. A general kernel based density estimate of $p(\boldsymbol{x})$ is given by

$$\widehat{p}^{(N)}\big(\boldsymbol{x};\boldsymbol{\beta}_N,\rho\big) = \sum_{j=1}^{N} \beta_j K_\rho\big(\boldsymbol{x},\boldsymbol{x}_j\big), \tag{1}$$

subject to

$$\beta_j \geq 0,\ 1 \leq j \leq N,\ \text{and}\ \boldsymbol{\beta}_N^{\mathrm{T}}\mathbf{1}_N = 1, \tag{2}$$

where β_js are the kernel weights, $\boldsymbol{\beta}_N = \big[\beta_1\ \beta_2 \cdots \beta_N\big]^{\mathrm{T}}$, and $\mathbf{1}_N$ is the N-dimensional vector whose elements are all equal to one, while $K_\rho\big(\boldsymbol{x},\boldsymbol{x}_j\big)$ is a chosen kernel function with the kernel centre vector $\boldsymbol{x}_j$ and a suitable kernel width ρ. In this study, we use the Gaussian kernel of

$$K_\rho\big(\boldsymbol{x},\boldsymbol{x}_j\big) = \frac{1}{\big(2\pi\rho^2\big)^{m/2}} \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}_j\|^2}{2\rho^2}\right) \tag{3}$$

but many other kernels can also be used. The sparse kernel density estimation involves the determination of the model structure of (1) where most elements in $\boldsymbol{\beta}_N$ become zeros. This can be achieved either by solving the constrained quadratic optimisation problem which initially works on the full model set of all the N kernels [11], [12], [19], or alternatively by selecting significant model terms one at a time forwardly which initially works on an empty model set [13], [17], [18].

The proposed sparse kernel density estimation algorithm also initially works on an empty model set, as in the cases of [13], [17], [18]. Specifically, in our proposed algorithm, the kernel functions $K_\rho\big(\boldsymbol{x},\boldsymbol{x}_j\big)$ with nonzero weights β_j are included into the model set selected in a forward regression manner. The final sparse kernel density estimator are based on the kernels formed from the subset $D_s = \big\{\boldsymbol{x}_1^{'}, \boldsymbol{x}_2^{'}, \cdots, \boldsymbol{x}_s^{'}\big\}$ of s data samples selected from D_N in this way. For example, if $\boldsymbol{x}_6$ is selected to form the first kernel, it is denoted as $\boldsymbol{x}_1^{'}$ in the selected data subset. Let the superscript $^{(l)}$ denote the lth forward selection step. At the lth forward selection step, further denote the intermediate kernel density estimator $\widehat{p}^{(l)}\big(\boldsymbol{x};\boldsymbol{\beta}_l^{(l)},\rho\big)$ as $\widehat{y}^{(l)}(\boldsymbol{x})$, that is,

$$\widehat{y}^{(l)}(\boldsymbol{x}) = \sum_{j=1}^{l} \beta_j^{(l)} K_\rho\big(\boldsymbol{x},\boldsymbol{x}_j^{'}\big), \tag{4}$$

where $\beta_j^{(l)}$, $1 \leq j \leq l$, are the kernels weights at the lth forward selection step, and $\boldsymbol{\beta}_l^{(l)} = \big[\beta_1^{(l)}\ \beta_2^{(l)} \cdots \beta_l^{(l)}\big]^{\mathrm{T}}$.

The proposed algorithm uses the FCR procedure [21] described below:

(i) At the first step, the PDF estimator is simply the first selected kernel

$$\widehat{y}^{(1)}(\boldsymbol{x}) = K_\rho\big(\boldsymbol{x},\boldsymbol{x}_1^{'}\big). \tag{5}$$

This means that $\beta_1^{(1)} = 1$.

(ii) At the lth step, where $l \geq 2$, the PDF estimator is constructed by adding the lth selected kernel $K_\rho\big(\boldsymbol{x},\boldsymbol{x}_l^{'}\big)$ to $\widehat{y}^{(l-1)}(\boldsymbol{x})$ via

$$\widehat{y}^{(l)}(\boldsymbol{x}) = \lambda_l \widehat{y}^{(l-1)}(\boldsymbol{x}) + (1-\lambda_l) K_\rho\big(\boldsymbol{x},\boldsymbol{x}_l^{'}\big), \tag{6}$$

where $0 \leq \lambda_l \leq 1$, $\forall l$, and $\lambda_1 = 0$.

It is a straightforward matter to verify that the model constructed using the FCR procedure satisfies the convex constraint conditions of (2), namely, $\beta_j^{(l)} \geq 0$, $1 \leq j \leq l$, and $\sum_{j=1}^{l} \beta_j^{(l)} = 1$, $\forall l \geq 1$, see [21]. If λ_l and $\boldsymbol{\beta}_{l-1}^{(l-1)}$ are given, $\boldsymbol{\beta}_l^{(l)}$ can be recursively computed via

$$\boldsymbol{\beta}_l^{(l)} = \begin{bmatrix} \lambda_l \boldsymbol{\beta}_{l-1}^{(l-1)} \\ 1-\lambda_l \end{bmatrix}, \tag{7}$$

where $l > 1$ and $\boldsymbol{\beta}_1^{(1)} = \beta_1^{(1)} = 1$.

It can be seen that the key issues are how to select the kernel $K_\rho\big(\boldsymbol{x},\boldsymbol{x}_l^{'}\big)$ as well as how to compute λ_l and hence the kernel weights $\boldsymbol{\beta}_l^{(l)}$, which are addressed in the next section.

III. Joint kernel selection and weight estimation based on the MISE

In the following, we introduce a new algorithm integrating the kernel term selection and the kernel weight estimation based on the MISE measure [2], [19], [20], within the general FCR framework described in the previous section. More specifically, the joint kernel selection and weight estimation at the lth forward selection stage is detailed in this section. We initially formulate the kernel weight estimation problem using the MISE criterion for a given kernel per forward selection step, and following this, we present the full algorithm including the kernel selection also based on the MISE.

A. Kernel weight estimation

Assuming that at the lth forward selection stage $K_\rho\big(\boldsymbol{x},\boldsymbol{x}_l^{'}\big)$ has been selected, we consider the problem of determining λ_l based on the global accuracy measure for density estimate, the integrated square error (ISE) which is given as (see for

example [19])

$$
\begin{aligned}
I\big(\boldsymbol{\beta}_l^{(l)}\big) &= \int \Big(p(\boldsymbol{x}) - \sum_{j=1}^{l} \beta_j^{(l)} K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big)\Big)^2 d\boldsymbol{x} \\
&= \int p^2(\boldsymbol{x}) d\boldsymbol{x} + \int \Big(\sum_{j=1}^{l} \beta_j^{(l)} K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big)\Big)^2 d\boldsymbol{x} \\
&\quad - 2E\Big[\sum_{j=1}^{l} \beta_j^{(l)} K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big)\Big] = \int p^2(\boldsymbol{x}) d\boldsymbol{x} \\
&\quad + \sum_{i=1}^{l}\sum_{j=1}^{l} \beta_i^{(l)} \beta_j^{(l)} \int K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_i^{'}\big) K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big) d\boldsymbol{x} \\
&\quad - 2\sum_{j=1}^{l} \beta_j^{(l)} E\Big[K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big)\Big] \\
&= \int p^2(\boldsymbol{x}) d\boldsymbol{x} + Q^{(l)}\big(\lambda_l\big), \qquad (8)
\end{aligned}
$$

in which $E[\bullet]$ denotes the expectation with respect to the true density $p(\boldsymbol{x})$. Since the unknown term $\int p^2(\boldsymbol{x}) d\boldsymbol{x}$ is independent of $\boldsymbol{\beta}_l^{(l)}$, it can be dropped from the objective function. We write the argument directly as λ_l for the last term $Q^{(l)}\big(\lambda_l\big)$, which becomes our objective function. We point out that since our algorithm is based on the FCR framework, this is the only parameter that needs to be estimated at the lth selection stage. $\boldsymbol{\beta}_l^{(l)}$ depends on λ_l and $\boldsymbol{\beta}_{l-1}^{(l-1)}$, i.e. the sequence $\{\lambda_1, \lambda_2, \cdots, \lambda_{l-1}\}$, that have already been obtained from the previous forward selection steps (see (7)).

Using the following unbiased estimator of $E\big[K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big)\big]$

$$
E\big[K_\rho\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big)\big] \approx \frac{1}{N}\sum_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_j^{'}\big) \qquad (9)
$$

as well as noting the Gaussian kernel yield

$$
\begin{aligned}
Q^{(l)}\big(\lambda_l\big) = &\sum_{i=1}^{l}\sum_{j=1}^{l} \beta_i^{(l)} \beta_j^{(l)} K_{\sqrt{2}\rho}\big(\boldsymbol{x}_i^{'}, \boldsymbol{x}_j^{'}\big) \\
&- \frac{2}{N}\sum_{j=1}^{l} \beta_j^{(l)} \sum_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_j^{'}\big). \qquad (10)
\end{aligned}
$$

For the first forward selection step, since we only have one kernel with $\lambda_1 = 0$, the only problem is to do with kernel selection but not with parameter estimation. For the convenience of derivation, we specifically write $Q^{(1)}\big(\lambda_1\big)$ as

$$
Q^{(1)}\big(\lambda_1\big) = \boldsymbol{C}_1^{(1)} - 2\boldsymbol{p}_1^{(1)}, \qquad (11)
$$

with

$$
\boldsymbol{p}_1^{(1)} = \frac{1}{N}\sum_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_1^{'}\big), \qquad (12)
$$

$$
\boldsymbol{C}_1^{(1)} = K_{\sqrt{2}\rho}\big(\boldsymbol{x}_1^{'}, \boldsymbol{x}_1^{'}\big) = \gamma, \qquad (13)
$$

where $\gamma = 1/\big(4\pi\rho^2\big)^{m/2}$. Using matrix expression, we can easily obtain the general recursive form of $Q^{(l)}(\lambda_l)$ for $l \geq 2$ given by

$$
Q^{(l)}\big(\lambda_l\big) = \Big(\boldsymbol{\beta}_l^{(l)}\Big)^{\mathrm{T}} \boldsymbol{C}_l^{(l)} \boldsymbol{\beta}_l^{(l)} - 2\Big(\boldsymbol{\beta}_l^{(l)}\Big)^{\mathrm{T}} \boldsymbol{p}_l^{(l)}, \qquad (14)
$$

with

$$
\boldsymbol{p}_l^{(l)} = \Big[\big(\boldsymbol{p}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \ \frac{1}{N}\sum_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_l^{'}\big)\Big]^{\mathrm{T}}, \qquad (15)
$$

$$
\boldsymbol{C}_l^{(l)} = \begin{bmatrix} \boldsymbol{C}_{l-1}^{(l-1)} & \boldsymbol{b}_{l-1}^{(l)} \\ \big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} & \gamma \end{bmatrix}, \qquad (16)
$$

where $\boldsymbol{b}_{l-1}^{(l)} = \big[K_{\sqrt{2}\rho}\big(\boldsymbol{x}_1^{'}, \boldsymbol{x}_l^{'}\big) \cdots K_{\sqrt{2}\rho}\big(\boldsymbol{x}_{l-1}^{'}, \boldsymbol{x}_l^{'}\big)\big]^T$.

By substituting (7), (15) and (16) into (14), we have

$$
\begin{aligned}
Q^{(l)}\big(\lambda_l\big) = &\begin{bmatrix} \lambda_l \boldsymbol{\beta}_{l-1}^{(l-1)} \\ 1-\lambda_l \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{C}_{l-1}^{(l-1)} & \boldsymbol{b}_{l-1}^{(l)} \\ \big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} & \gamma \end{bmatrix} \begin{bmatrix} \lambda_l \boldsymbol{\beta}_{l-1}^{(l-1)} \\ 1-\lambda_l \end{bmatrix} \\
&- 2\big[\lambda_l \big(\boldsymbol{\beta}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \ 1-\lambda_l\big] \begin{bmatrix} \boldsymbol{p}_{l-1}^{(l-1)} \\ \frac{1}{N}\sum\limits_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_l^{'}\big) \end{bmatrix} \\
= &\lambda_l^2 \mu^{(l)} + \big(1-\lambda_l\big)^2 \gamma + 2\lambda_l\big(1-\lambda_l\big)\big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)} \\
&- 2\lambda_l \nu^{(l)} - \frac{2\big(1-\lambda_l\big)}{N}\sum_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_l^{'}\big), \qquad (17)
\end{aligned}
$$

where

$$
\begin{cases} \mu^{(l)} = \big(\boldsymbol{\beta}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \boldsymbol{C}_{l-1}^{(l-1)} \boldsymbol{\beta}_{l-1}^{(l-1)}, \\ \nu^{(l)} = \big(\boldsymbol{\beta}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \boldsymbol{p}_{l-1}^{(l-1)}. \end{cases} \qquad (18)
$$

It happens that $Q^{(l)}\big(\lambda_l\big)$ is a quadratic function with respect to λ_l. Hence there exits a unique minimum of $Q^{(l)}\big(\lambda_l\big)$, which can be found by setting $\frac{\partial}{\partial \lambda_l} Q^{(l)}\big(\lambda_l\big) = 0$, followed by the constraint satisfaction operation. This yields the closed-form solution for λ_l given as

$$
\lambda_l = \min\big\{\max\big\{u_l, 0\big\}, 1\big\}, \qquad (19)
$$

with

$$
u_l = \frac{\gamma - \big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)} + \nu^{(l)} - \frac{1}{N}\sum\limits_{i=1}^{N} K_\rho\big(\boldsymbol{x}_i, \boldsymbol{x}_l^{'}\big)}{\mu^{(l)} + \gamma - 2\big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)}}. \qquad (20)
$$

It is easy to verify that the constraint satisfaction operator

$$
\min\big\{\max\{u, 0\}, 1\big\} = \begin{cases} 1, & u > 1, \\ 0, & u < 0, \\ u, & 0 < u < 1. \end{cases} \qquad (21)
$$

Therefore, $0 \leq \lambda_l \leq 1$ is guaranteed. By plugging λ_l back to (17), we obtain the MISE value $Q^{(l)}\big(\lambda_l\big)$ for this given kernel. The computational cost of parameter estimation for a given kernel per forward selection step is in the order of $\mathcal{O}(l)$, which is extremely low owing to the recursive computation and the closed-form solution for the only parameter λ_l.

B. Joint kernel selection and weight estimation algorithm

The basic idea for kernel selection is to select the subset D_s of s data samples one at a time from the full data set D_N and to form the kernels $K_\rho(\boldsymbol{x}, \boldsymbol{x}_j^{'})$ so that the ISE is minimised sequentially. Specifically, at the lth forward selection stage a data sample is selected from the remaining $(N-l+1)$ candidate data samples. We review the contribution of each candidate data sample according to its associated MISE value to decide if this sample is to be added to the model. The data point producing the smallest MISE value amongst all the candidate data samples is assigned as $\boldsymbol{x}_l^{'}$ and is used to form $K_\rho(\boldsymbol{x}, \boldsymbol{x}_l^{'})$.

First define $\boldsymbol{X}_N^{(l-1)} \in \mathbb{R}^{m \times N}$ as

$$\boldsymbol{X}_N^{(l-1)} = \Big[\boldsymbol{x}_1^{'} \cdots \boldsymbol{x}_{l-1}^{'} \ \boldsymbol{x}_l^{(l-1)} \cdots \boldsymbol{x}_N^{(l-1)}\Big], \tag{22}$$

and $\boldsymbol{q}_N^{(l-1)} \in \mathbb{R}^{1 \times N}$ as

$$\begin{aligned}\boldsymbol{q}_N^{(l-1)} =& \Big[\frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_1^{'}) \cdots \frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_{l-1}^{'}) \\ & \frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_l^{(l-1)}) \cdots \frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_N^{(l-1)})\Big], \end{aligned} \tag{23}$$

with

$$\boldsymbol{X}_N^{(0)} = \Big[\boldsymbol{x}_1^{(0)} \ \boldsymbol{x}_2^{(0)} \cdots \boldsymbol{x}_N^{(0)}\Big] = \Big[\boldsymbol{x}_1 \ \boldsymbol{x}_2 \cdots \boldsymbol{x}_N\Big], \tag{24}$$

$$\begin{aligned}\boldsymbol{q}_N^{(0)} =& \Big[\frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_1) \ \frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_2) \cdots \\ & \frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_N)\Big]. \end{aligned} \tag{25}$$

If the j_lth column, where $l \le j_l \le N$, and the lth column of $\boldsymbol{X}_N^{(l-1)}$ are interchanged, $\boldsymbol{X}_N^{(l-1)}$ becomes $\boldsymbol{X}_N^{(l)}$. Similarly, if the j_lth column and the lth column of $\boldsymbol{q}_N^{(l-1)}$ are interchanged, $\boldsymbol{q}_N^{(l-1)}$ becomes $\boldsymbol{q}_N^{(l)}$. Further define the jth element of $\boldsymbol{q}_N^{(l-1)}$ as $q^{(l-1)}(j) = \frac{1}{N}\sum_{i=1}^{N} K_\rho(\boldsymbol{x}_i, \boldsymbol{x}_j^{(l-1)})$ for $l \le j \le N$. We are now ready to present our proposed algorithm.

Initialization: At the 1st stage of the selection procedure, set $\boldsymbol{\beta}_1^{(1)} = \beta_1^{(1)} = 1$ and $\lambda_1 = 0$. For $1 \le j \le N$, compute

$$Q^{(1,j)}(\lambda_1) = \gamma - 2\boldsymbol{p}_1^{(1,j)}, \tag{26}$$

where $\boldsymbol{p}_1^{(1,j)} = q^{(0)}(j)$. Next find

$$Q^{(1,j_1)}(\lambda_1) = \min\Big\{Q^{(1,j)}(\lambda_1), 1 \le j \le N\Big\}. \tag{27}$$

Then the j_1th column and the first column of $\boldsymbol{X}_N^{(0)}$ are interchanged to yield $\boldsymbol{X}_N^{(1)}$, and the j_1th column and the first column of $\boldsymbol{q}_N^{(0)}$ are interchanged to yield $\boldsymbol{q}_N^{(1)}$. This effectively selects the first kernel. Update $Q^{(1)}(\lambda_1) = Q^{(1,j_1)}(\lambda_1)$ with $\boldsymbol{C}_1^{(1)} = \gamma$ and $\boldsymbol{p}_1^{(1)} = \boldsymbol{p}_1^{(1,j_1)}$.

The lth stage of the selection procedure, where $l \ge 2$:

Step 1). Calculate $\mu^{(l)}$ and $\nu^{(l)}$ according to (18). Then, for $l \le j \le N$, compute

$$\begin{aligned}\boldsymbol{b}_{l-1}^{(l,j)} =& \big[K_{\sqrt{2}\rho}(\boldsymbol{x}_1^{'}, \boldsymbol{x}_j^{(l-1)}) \cdots K_{\sqrt{2}\rho}(\boldsymbol{x}_{l-1}^{'}, \boldsymbol{x}_j^{(l-1)})\big]^{\mathrm{T}}, \\ d^{(l,j)} =& \big(\boldsymbol{b}_{l-1}^{(l,j)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)}, \\ \lambda_l^{(j)} =& \min\Big\{\max\Big\{\frac{\gamma - d^{(l,j)} + \nu^{(l)} - q^{(l-1)}(j)}{\mu^{(l)} + \gamma - 2d^{(l,j)}}, 0\Big\}, 1\Big\}, \end{aligned}$$

$$\begin{aligned}Q^{(l,j)}(\lambda_l^{(j)}) =& \big(\lambda_l^{(j)}\big)^2 \mu^{(l)} + \big(1-\lambda_l^{(j)}\big)^2 \gamma \\ & + 2\lambda_l^{(j)}\big(1-\lambda_l^{(j)}\big) d^{(l,j)} - 2\lambda_l^{(j)} \nu^{(l)} \\ & - 2\big(1-\lambda_l^{(j)}\big) q^{(l-1)}(j). \end{aligned}$$

Step 2): Find

$$Q^{(l,j_l)}(\lambda_l^{(j_l)}) = \min\Big\{Q^{(l,j)}(\lambda_l^{(j)}), \ l \le j \le N\Big\}. \tag{28}$$

Then the j_lth column and the lth column of $\boldsymbol{X}_N^{(l-1)}$ are interchanged to yield $\boldsymbol{X}_N^{(l)}$. Also the j_lth column and the lth column of $\boldsymbol{q}_N^{(l-1)}$ are interchanged to yield $\boldsymbol{q}_N^{(l)}$. This effectively selects the lth kernel. Update $\lambda_l = \lambda_l^{(j_l)}$ and $Q^{(l)}(\lambda_l) = Q^{(l,j_l)}(\lambda_l^{(j_l)})$ as well as

$$\boldsymbol{\beta}_l^{(l)} = \begin{bmatrix} \lambda_l^{(j_l)} \boldsymbol{\beta}_{l-1}^{(l-1)} \\ 1 - \lambda_l^{(j_l)} \end{bmatrix},$$

$$\boldsymbol{p}_l^{(l)} = \big[\big(\boldsymbol{p}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \ q^{(l)}(l)\big]^{\mathrm{T}},$$

and

$$\boldsymbol{C}_l^{(l)} = \begin{bmatrix} \boldsymbol{C}_{l-1}^{(l-1)} & \boldsymbol{b}_{l-1}^{(l,j_l)} \\ \big(\boldsymbol{b}_{l-1}^{(l,j_l)}\big)^{\mathrm{T}} & \gamma \end{bmatrix}.$$

Termination: The selection procedure is terminated at the $(s+1)$th stage when the following condition is detected

$$\big|Q^{(s+1)}(\lambda_{s+1}) - Q^{(s)}(\lambda_s)\big| \le \delta Q$$

where δQ is a predetermined very small positive number, and this produces a subset model with the s significant kernels. The computational cost of our proposed algorithm is extremely low. In fact, the lth stage of the selection procedure has the complexity of $2l(N-l+1)$. Therefore, the overall computational complexity of our proposed algorithm is approximately $s^2 N$, that is, $\mathcal{O}(N)$ scaled by s^2, where s is the number of kernels selected, which in general will not necessarily increase with the data set size. Note that for large data sets $s \ll N$. This computation complexity compares very favorably with the existing efficient sparse kernel density estimators at $\mathcal{O}(N^2)$.

IV. Simulation Study

The first two examples are pure PDF estimation examples. In each of these two examples, a data set of N samples was randomly drawn from a distribution $p(\boldsymbol{x})$ and used to construct the PDF estimator $\widehat{p}^{(s)}(\boldsymbol{x}; \boldsymbol{\beta}_s, \rho)$ using the proposed FCR-MSIE approach. A separate test data set of $N_{\text{test}} = 10000$

samples was used for evaluating the density estimate according to the L_1 test error

$$L_1 = \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} \left| p(\boldsymbol{x}_k) - \widehat{p}^{(s)}(\boldsymbol{x}_k; \boldsymbol{\beta}_s, \rho) \right|. \quad (29)$$

The experiment was repeated for 100 different random runs. The benchmark PDF estimators used for comparison include the non-sparse PW estimator as well as the three efficient existing sparse PDF estimators, the SKD estimator of [17], the SKD estimator of [18], and the RSDE of [19]. The Gaussian kernel was used for all the algorithms.

Example 1: The density to be estimated for this 2-dimensional (2-D) example was given by the mixture of two densities of a Gaussian and a Laplacian, as defined by

$$p(\boldsymbol{x}) = \frac{1}{4\pi} \exp\left(-\frac{(x_1-2)^2}{2}\right) \exp\left(-\frac{(x_2-2)^2}{2}\right) + \frac{0.35}{8} \exp(-0.7|x_1+2|) \exp(-0.5|x_2+2|). \quad (30)$$

The estimation data set contained $N = 500$ points.

Example 2: The density to be estimated for this 6-D example was defined by

$$p(\boldsymbol{x}) = \frac{1}{3} \sum_{i=1}^{3} \frac{1}{(2\pi)^3 \sqrt{|\boldsymbol{\Gamma}_i|}} \exp\Big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}} \boldsymbol{\Gamma}_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)\Big), \quad (31)$$

with

$$\begin{aligned} \boldsymbol{\mu}_1 &= [1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0]^{\mathrm{T}}, \\ \boldsymbol{\Gamma}_1 &= \text{diag}\{1.0, 2.0, 1.0, 2.0, 1.0, 2.0\}, \end{aligned}$$

$$\begin{aligned} \boldsymbol{\mu}_2 &= [-1.0\ -1.0\ -1.0\ -1.0\ -1.0\ -1.0]^{\mathrm{T}}, \\ \boldsymbol{\Gamma}_2 &= \text{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}, \end{aligned}$$

$$\begin{aligned} \boldsymbol{\mu}_2 &= [0.0\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0]^{\mathrm{T}}, \\ \boldsymbol{\Gamma}_2 &= \text{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}, \end{aligned}$$

where $|\boldsymbol{\Gamma}|$ denotes the determinant of $\boldsymbol{\Gamma}$. The estimation data set contained $N = 600$ points.

The results of the five density estimators for Examples 1 and 2 are listed in Table I (a) and (b), respectively. For the PW PDF estimator, the kernel width was determined by the MSIE criterion (see for example [2]). For the RSDE and the proposed FCR-MSIE estimator, the kernel widths were empirically set through trial and error. The results for the other two SKD estimators are quoted from [17], [18], respectively. It is seen that the proposed algorithm can construct sparse kernel density estimates with the competitive accuracy to the PW estimator and the other three existing SKD estimators. Our proposed FCR-MSIE estimator has a significant advantage in that it offers a much lower complexity in constructing PDF estimate than the three existing SKD estimators of [17]–[19].

To illustrate the application of the proposed method, the two two-class classification examples are also presented. The training data set is divided into the two-class training data sets, C_0 and C_1, respectively. The proposed method can readily be applied to estimate the two conditional PDFs, $\widehat{p}(\boldsymbol{x}; \boldsymbol{\beta}_{C_0}, \rho_{C_0} | C_0)$ and $\widehat{p}(\boldsymbol{x}; \boldsymbol{\beta}_{C_1}, \rho_{C_1} | C_1)$, based on the data sets C_0 and C_1, respectively. The Bayes decision rule given by

$$\begin{cases} \boldsymbol{x} \in C_0, \text{ if } \widehat{p}(\boldsymbol{x}; \boldsymbol{\beta}_{C_0}, \rho_{C_0} | C_0) \geq \widehat{p}(\boldsymbol{x}; \boldsymbol{\beta}_{C_1}, \rho_{C_1} | C_1), \\ \boldsymbol{x} \in C_1, \text{ otherwise}, \end{cases} \quad (32)$$

can be applied to the test data set to obtain the corresponding classification error rate. The Gaussian kernel was adopted in all the following two examples.

Example 3: The breast cancer data, taken from [23], has the input dimension of $m = 9$. The data set contained 100 realizations, each having 200 training patterns and 77 test patterns. In [22], six state-of-the-arts classifiers were applied to the data set, and we quote the results of [22] in Table II. For the first five classifiers studied in [22], the nonlinear Gaussian radial basis function (RBF) network with five optimised RBF units was used. For the SVM classifier with Gaussian kernel, no average model size was reported in [22], but our experience with the SVM classifier suggests that it could likely contains around 100 or more kernels.

The classification results obtained by the proposed FCR-MISE algorithm are also listed in Table II for comparison. For the FCR-MISE algorithm, the two widths in the two conditional PDF estimates were set empirically as $\rho_{C_0} = 1.8$ and $\rho_{C_1} = 1.9$, respectively, for all the 100 realizations of

TABLE I
PERFORMANCE COMPARISON OF FIVE KERNEL DENSITY ESTIMATORS FOR EXAMPLES 1 AND 2.

(a) Example 1.

Method	L_1 test error (mean ± STD)	Kernel number (mean ± STD)
PW	$(4.18 \pm 0.8) \times 10^{-3}$	500 ± 0
SKD estimator [17]	$(3.83 \pm 0.8) \times 10^{-3}$	11.9 ± 2.6
SKD estimator [18]	$(3.84 \pm 0.8) \times 10^{-3}$	15.3 ± 3.9
RSDE [19]	$(4.24 \pm 0.8) \times 10^{-3}$	129.4 ± 35.7
Proposed FCR-MISE	$(3.33 \pm 0.8) \times 10^{-3}$	25.1 ± 2.7

(b) Example 2.

Method	L_1 test error (mean ± STD)	Kernel number (mean ± STD)
PW	$(3.18 \pm 0.13) \times 10^{-5}$	600 ± 0
SKD estimator [17]	$(4.48 \pm 1.2) \times 10^{-5}$	14.9 ± 2.1
SKD estimator [18]	$(3.11 \pm 0.5) \times 10^{-5}$	9.4 ± 1.9
RSDE [19]	$(3.67 \pm 0.7) \times 10^{-5}$	29.4 ± 10.1
Proposed FCR-MISE	$(2.82 \pm 0.1) \times 10^{-5}$	19.4 ± 0.9

TABLE II
AVERAGE MISCLASSIFICATION RATE IN % OVER THE 100 REALIZATIONS OF THE BREAST CANCER TEST DATA SET AND MODEL SIZE.

Method	Misclassification rate	Model Size
RBF	27.6 ± 4.7	5
Adaboost with RBF	30.4 ± 4.7	5
AdaBoost-Reg	26.5 ± 4.5	5
LP-Reg-AdaBoost	26.8 ± 6.1	5
QP-Reg-AdaBoost	25.9 ± 4.6	5
SVM with RBF kernel	26.0 ± 4.7	not available
Proposed FCR-MISE	26.1 ± 4.7	92 ± 0

TABLE III
AVERAGE MISCLASSIFICATION RATE IN % OVER THE 100 REALIZATIONS OF THE TITANIC TEST DATA SET AND MODEL SIZE.

Method	Misclassification rate	Model Size
RBF	23.3 ± 1.3	4
Adaboost with RBF	22.6 ± 1.2	4
AdaBoost-Reg	22.6 ± 1.2	4
LP-Reg-AdaBoost	24.0 ± 4.4	4
QP-Reg-AdaBoost	22.7 ± 1.1	4
SVM with RBF kernel	22.4 ± 1.0	not available
Proposed FCR-MISE	22.2 ± 0.4	83.8 ± 6.8

the data set, and the model size for our method is the sum of the kernels in building the two conditional PDFs, selected from a total of 400 training patterns. Clearly the classification accuracy of our FCR-MISE algorithm is competitive, compared with the six state-of-the-arts classifiers studied in [22]. It is worth emphasising that the modelling paradigms of [22] are discriminative models constructed based on both the input and output (class label) information. By contrast, the proposed FCR-MISE algorithm only relies on the input information to construct each conditional PDF, and the total number of the kernels for constructing the Bayes classifier (32) is unavoidably larger than the discriminative classifiers of [22].

Example 4: The Titanic data, also taken from [23], has the input dimension of $m = 3$. The data set contained 100 realizations, each having 150 training patterns and 2051 test patterns. Table III lists the classification results obtained by the proposed FCR-MISE algorithm in comparison with the results of the six classifiers quoted from [22]. The two widths used in the proposed FCR-MISE algorithm were set empirically as $\rho_{C_0} = 1.8$ and $\rho_{C_1} = 1.7$, respectively, for all 100 realizations. The model size for the FCR-MISE algorithm denotes the sum of the kernels used for the two conditional PDF estimates, selected from the total of 300 training patterns. From Table III, it can be seen that the classification accuracy of the FCR-MISE method is competitive, compared with the six state-of-the-arts classifiers studied in [22]. We point out that the size of the Bayes classifier obtained by the FCR-MISE density estimator is most likely to be smaller than the SVM classifier, even though the latter is a discriminative model.

V. CONCLUSIONS

A new sparse kernel density estimator has been introduced. Our main contribution is to derive a recursive algorithm which selects significant kernels one at time based on the minimum integrated square error (MISE) criterion. Since at each forward step, only a single parameter is estimated using a closed form solution developed in this contribution, the proposed approach has a very low the computational complexity. Numerical examples have been employed to demonstrate that the proposed approach can construct sparse kernel density estimators with competitive accuracy to existing kernel density estimators.

REFERENCES

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

[2] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall: London, 1986.

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

[4] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press: Oxford, 1995.

[5] S. Chen, X. Hong, and C J. Harris, "Particle swarm optimization aided orthogonal forward regression for unified data modelling," *IEEE Trans. Evolutionary Computation*, vol. 14, no. 4, pp. 477–499, Aug. 2010.

[6] E. Parzen, "On estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1066–1076, Sept. 1962.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Statistical Society 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," *Technical Report ICSI-TR-97-021*, University of California, Berkeley, 1998.

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

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

[11] 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. MIT Pres: Cambridge, MA, 1999, pp. 293–306.

[12] 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. MIT Press: Cambridge, MA, 2000, pp. 659–665.

[13] A. Choudhury, *Fast Machine Learning Algorithms for Large Data*, Ph.D. dissertation, School of Engineering Sciences, University of Southampton, 2002.

[14] 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, 1989.

[15] X. Hong, P. M. Sharkey, and K. Warwick, "Automatic nonlinear predictive model construction algorithm using forward regression and the PRESS statistic," *IEE Proc. Control Theory Applications*, vol. 150, no. 3, pp. 245–254, 2003.

[16] S. Chen, X. Hong, C. J. Harris, and P. M. Sharkey, "Sparse modelling using forward regression with PRESS statistic and regularization," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol. 34, no. 2, pp. 898–911, 2004.

[17] 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. Systems, Man, and Cybernetics, Part B*, vol. 34, no. 4, pp. 1708–1717, Aug. 2004.

[18] S. Chen, X. Hong, and C. J. Harris, "An orthogonal forward regression techniques for sparse kernel density estimation," *Neurocomputing*, vol. 71, nos. 46, pp. 931–943, Jan. 2008.

[19] M. Girolami and C. He, "Probability density estimation from optimally condensed data samples," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1253–1264, Oct. 2003.

[20] S. W. Scott, "Parametric statistical modeling by minimum integrated square error," *Technometrics*, vol. 43, no. 3, pp. 274–285, Aug. 2001.

[21] X. Hong and C. J. Harris, "A mixture of experts network structure construction algorithm for modelling and control," *Applied Intelligence*, vol. 16, no. 1, pp. 59–69, 2002.

[22] G. Rätsch, T. Onoda, and K. R. Müller, "Soft margins for AdaBoost," *Machine Learning*, vol. 42, no. 3, pp. 287–320, 2001.

[23] G. Rätsch, "http://www.fml.tuebingen.mpg.de/members/raetsch/benchmark,"