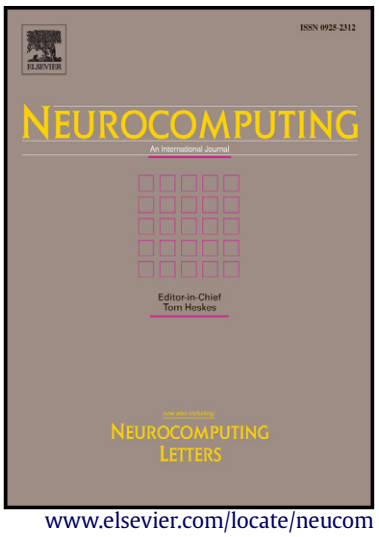
Sparse density estimator with tunable kernels



www.elsevier.com/locate/neucom

Sparse Density Estimator with Tunable Kernels

Xia Hong, Sheng Chen, Victor M. Becerra

PII: S0925-2312(15)01173-X
DOI: http://dx.doi.org/10.1016/j.neucom.2015.08.021
Reference: NEUCOM15949

To appear in: *Neurocomputing*

Received date: 30 March 2015
Revised date: 29 June 2015
Accepted date: 3 August 2015

Cite this article as: Xia Hong, Sheng Chen, Victor M. Becerra, Sparse Density Estimator with Tunable Kernels, *Neurocomputing*, http://dx.doi.org/10.1016/j.neucom.2015.08.021

Sparse Density Estimator with Tunable Kernels

Xia Hong, Sheng Chen, and Victor M. Becerra

Abstract

A new sparse kernel density estimator with tunable kernels is introduced within a forward constrained regression framework whereby the nonnegative and summing-to-unity constraints of the mixing weights can easily be satisfied. Based on the minimum integrated square error criterion, a recursive algorithm is developed to select significant kernels one at time, and the kernel width of the selected kernel is then tuned using the gradient descent algorithm. Numerical examples are employed to demonstrate that the proposed approach is effective in constructing very sparse kernel density estimators with competitive accuracy to existing kernel density estimators.

Index Terms

Probability density function, kernel density estimator, sparse modeling, minimum integrated square error

I. Introduction

The probability density function (PDF) estimation, e.g., the Parzen window (PW) and finite mixture model, is of fundamental importance to many data analysis and pattern recognition applications [1]–[8]. There is a considerable interest into research on sparse PDF estimation which can be summarized into two categories. The first category is based on constrained optimization. For example, the support vector machine (SVM) density estimation was researched [9], [10], in which the density estimation problem is formulated as a supervised learning mode whilst the mean absolute deviation between the empirical cumulative distribution function (CDF) calculated from the training data and the CDF based on the PDF estimator also calculated from the training data are minimized. This yields the sparsity inducing property, i.e., at the optimality,

X. Hong and V.M. Becerra are with School of Systems Engineering, University of Reading, Reading, RG6 6AY, UK (E-mails: x.hong@reading.ac.uk, v.m.becerra@reading.ac.uk).

S. Chen is with Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK (E-mail: sqc@ecs.soton.ac.uk), and also with King Abdulaziz University, Jeddah 21589, Saudi Arabia.

many kernels' weights are driven to zero. The desirable property of sparsity inducing also happens in the interesting approach of reduced set density estimator (RSDE) [11], which is based on the minimization of the integrated square error (ISE) between the estimator and the true density evaluated on the training data [2], [11], [12], and two efficient optimization algorithms were introduced. Alternatively, by exploiting the first and second order Riemannian geometry of the multinomial manifold, the Riemannian trust-region algorithm [13] was recently applied to find the set of sparse mixing coefficients based on the minimum ISE (MISE), referred to as the RTR-MISE algorithm [14].

The second category of sparse kernel density estimators construct the PDF estimator in a forward regression manner. A regression-based PDF estimation method was introduced [15], in which the empirical CDF is constructed and used as the desired response. In order to automatically determine the model structure with the improved model generalization, the regression-based idea of [15] and the approach of [16] were extended to yield an orthogonal forward regression (OFR) based sparse density estimation algorithm [17] which is capable of automatically constructing a very sparse kernel density estimate, with comparable performance to that of the PW estimate. A simpler and viable alternative approach was proposed to use kernels directly as regressors by adopting the PW estimate as the target response [18]. A sparse kernel density estimator [19] was introduced based on the MISE and the forward constrained regression (FCR) [20] to select significant kernels one at time, which has very low computational cost and is referred to as the FCR-MISE algorithm.

With the exception of [4], in all the above-mentioned sparse kernel estimators, including those based on the MISE approach [11], [14], [19], the PDF kernels involve a single and *fixed* kernel bandwidth parameter that needs to be empirically predetermined. By contrast, this paper introduces a new sparse kernel density estimator with *tunable* kernels also based on the MISE. Specifically, a new recursive algorithm is developed to select significant kernels one at time, followed by tuning the kernel width of the selected kernel using the gradient descent algorithm. This means that there is no need to determine the bandwidth parameters empirically outside the algorithm loop. Numerical examples are employed to demonstrate that the proposed approach can construct very sparse kernel density estimates with competitive accuracy, compared to the existing kernel density estimators.

II. Forward construction of tunable sparse kernel density estimator

Given the finite data set $D_N = \{\boldsymbol{x}_j\}_{j=1}^N$ consisting of N data samples, where the data $\boldsymbol{x}_j \in \mathbb{R}^m$ follows an unknown PDF $p(\boldsymbol{x})$, the problem under study is to find a sparse approximation of $p(\boldsymbol{x})$ by forward construction based on the subset $D_M = \left\{\boldsymbol{x}_1^{'}, \boldsymbol{x}_2^{'}, \cdots, \boldsymbol{x}_M^{'}\right\}$ of M data samples selected from D_N. For example, if $\boldsymbol{x}_6$ from D_N is selected to form the first kernel, it is denoted as $\boldsymbol{x}_1^{'}$ in D_M. A general kernel based density estimate of $p(\boldsymbol{x})$ is given by

$$\widehat{p}^{(M)}\left(\boldsymbol{x}; \boldsymbol{\beta}_M, \boldsymbol{\sigma}_M\right) = \sum_{i=1}^{M} \beta_i K_{\sigma_i}\left(\boldsymbol{x}, \boldsymbol{x}_i^{'}\right) \tag{1}$$

subject to

$$\beta_i \geq 0, \quad \text{and } \boldsymbol{\beta}_M^{\mathrm{T}} \mathbf{1}_M = 1, \tag{2}$$

where $K_{\sigma_i}\left(\boldsymbol{x}, \boldsymbol{x}_i^{'}\right)$ is the Gaussian kernel with the kernel center vector $\boldsymbol{x}_i^{'}$ and an *adjustable* kernel width σ_i given by

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

and β_i is the ith kernel weight, while $\boldsymbol{\sigma}_M = \left[\sigma_1\ \sigma_2 \cdots \sigma_M\right]^{\mathrm{T}}$, $\boldsymbol{\beta}_M = \left[\beta_1\ \beta_2 \cdots \beta_M\right]^{\mathrm{T}}$, and $\mathbf{1}_M$ is the M-dimensional vector whose elements are all equal to one.

We form the kernel density estimator (1) from the subset D_M in a forward construction manner. Specifically given the initial condition $\sigma_i = \sigma_0$, $\forall i$, and starting from an empty model set, our proposed sparse kernel density estimation algorithm selects the kernel functions $K_{\sigma_0}\left(\boldsymbol{x}, \boldsymbol{x}_i^{'}\right)$ into the model set one at a time from D_N. At each forward step, the associated kernel width σ_i is then optimized to obtain $K_{\sigma_i}\left(\boldsymbol{x}, \boldsymbol{x}_i^{'}\right)$.

Let the superscript $^{(l)}$ denote the lth forward selection step. At the lth forward selection step, further denote the intermediate kernel density estimator $\widehat{p}^{(l)}\left(\boldsymbol{x}; \boldsymbol{\beta}_l^{(l)}, \boldsymbol{\sigma}_l\right)$ as $\widehat{y}^{(l)}(\boldsymbol{x})$, where $\boldsymbol{\sigma}_l^{(l)} = \left[\sigma_1\ \sigma_2 \cdots \sigma_l\right]^{\mathrm{T}}$ and $\boldsymbol{\beta}_l^{(l)} = \left[\beta_1^{(l)}\ \beta_2^{(l)} \cdots \beta_l^{(l)}\right]^{\mathrm{T}}$, with $\beta_i^{(l)}$, $1 \leq i \leq l$, as the kernels weights at the lth forward selection step, i.e.,

$$\widehat{y}^{(l)}(\boldsymbol{x}) = \sum_{i=1}^{l} \beta_i^{(l)} K_{\sigma_i}\left(\boldsymbol{x}, \boldsymbol{x}_i^{'}\right). \tag{4}$$

The proposed algorithm integrates the FCR procedure [20] described below:

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

$$\widehat{y}^{(1)}(\boldsymbol{x}) = K_{\sigma_1}\left(\boldsymbol{x}, \boldsymbol{x}_1^{'}\right), \tag{5}$$

where $K_{\sigma_1}\left(\boldsymbol{x}, \boldsymbol{x}_1^{'}\right)$ is obtained by adjusting the kernel width from σ_0 to σ_1 based on the selected kernel center $\boldsymbol{x}_1^{'}$. Clearly $\beta_1^{(1)} = 1$.

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

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

where $K_{\sigma_l}\left(\boldsymbol{x}, \boldsymbol{x}_l^{'}\right)$ is obtained by adjusting the kernel width from σ_0 to σ_l based on the selected kernel center $\boldsymbol{x}_l^{'}$, while $0 \leq \lambda_l \leq 1$, $\forall l$, and $\lambda_1 = 0$.

It can be straightforwardly verified that the model constructed using the FCR procedure satisfies the convex constraint conditions of (2), namely, $\beta_i^{(l)} \geq 0$, $1 \leq i \leq l$, and $\sum_{i=1}^{l} \beta_i^{(l)} = 1$, $\forall l \geq 1$, see [20]. Moreover, given λ_l and $\boldsymbol{\beta}_{l-1}^{(l-1)}$, $\boldsymbol{\beta}_l^{(l)}$ 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 at each forward selection step l are 1) how to initially select the kernel center vector $\boldsymbol{x}_l^{'}$ with the kernel width $\sigma_l = \sigma_0$, followed by adjusting the kernel width σ_l for the selected kernel; and 2) how to compute λ_l and hence the kernel weight $\boldsymbol{\beta}_l^{(l)}$.

III. Joint kernel selection and kernel width optimization based on the MISE

We now introduce our new algorithm integrating the kernel term selection, the kernel width optimization and the kernel weight calculation based on MISE [2], [11], [12] and the FCR framework described in the previous section. In particular, we detail the joint kernel selection, the tunable kernel width optimization and kernel weight estimation at the lth forward selection stage. Specifically, based on the ISE criterion, we formulate initially the kernel weight estimation problem for a given kernel per forward selection step, and then the kernel width optimization using the gradient descent algorithm for the selected kernel. Joint kernel selection together with the kernel width/weights optimization are finally presented.

A. Kernel weight estimation

At the lth forward selection stage, $K_{\sigma_i}\left(\boldsymbol{x}, \boldsymbol{x}_i^{'}\right)$ are given for $1 \leq i \leq l-1$, and we consider the problem of determining λ_l and σ_l for a fixed $\boldsymbol{x}_l^{'}$ based on the ISE which is the global accuracy

measure for density estimate and is given by [11]

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

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 $\{\lambda_l, \sigma_l\}$ for the last term $Q^{(l)}\big(\lambda_l, \sigma_l\big)$, which becomes our objective function. We point out that since our algorithm is based on the FCR framework, only these two parameters need to be estimated at the lth selection stage. Note that $\boldsymbol{\beta}_l^{(l)}$ 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)). Similarly $\{\sigma_1, \sigma_2, \cdots, \sigma_{l-1}\}$ are also obtained from the previous forward selection steps.

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

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

as well as noting the result of $\int K_{\sigma_i}\big(\boldsymbol{x}, \boldsymbol{x}_i^{'}\big) K_{\sigma_j}\big(\boldsymbol{x}, \boldsymbol{x}_j^{'}\big) d\boldsymbol{x}$ given in Appendix yield

$$\begin{aligned}
Q^{(l)}\big(\lambda_l, \sigma_l\big) \triangleq & \sum_{i=1}^{l}\sum_{j=1}^{l} \beta_i^{(l)}\beta_j^{(l)} K_{\sigma_{i,j}}\big(\boldsymbol{x}_i^{'}, \boldsymbol{x}_j^{'}\big) \\
& - \frac{2}{N}\sum_{i=1}^{l} \beta_i^{(l)} \sum_{k=1}^{N} K_{\sigma_i}\big(\boldsymbol{x}_k, \boldsymbol{x}_i^{'}\big), \qquad (10)
\end{aligned}$$

where $\sigma_{i,j} = \sqrt{\sigma_i^2 + \sigma_j^2}$. Using matrix expression, we easily obtain the recursive form of $Q^{(l)}(\lambda_l, \sigma_l)$ which is given by

$$Q^{(l)}\big(\lambda_l, \sigma_l\big) = \mu^{(l)} - 2\nu^{(l)} \tag{11}$$

where

$$\begin{cases} \mu^{(l)} = \big(\boldsymbol{\beta}_l^{(l)}\big)^{\mathrm{T}} \boldsymbol{C}_l^{(l)} \boldsymbol{\beta}_l^{(l)}, \\ \nu^{(l)} = \big(\boldsymbol{\beta}_l^{(l)}\big)^{\mathrm{T}} \boldsymbol{p}_l^{(l)}, \end{cases} \tag{12}$$

in which $\boldsymbol{p}_l^{(l)}$ and $\boldsymbol{C}_l^{(l)}$ can be computed recursively as

$$\boldsymbol{p}_l^{(l)} = \Big[\big(\boldsymbol{p}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \ \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_l}\big(\boldsymbol{x}_k, \boldsymbol{x}_l^{'}\big)\Big]^{\mathrm{T}}, \tag{13}$$

$$\boldsymbol{C}_l^{(l)} = \begin{bmatrix} \boldsymbol{C}_{l-1}^{(l-1)} & \boldsymbol{b}_{l-1}^{(l)} \\ \big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} & \gamma_l \end{bmatrix}, \tag{14}$$

and

$$\begin{cases} \gamma_l = 1/\big(4\pi\sigma_l^2\big)^{m/2}, \\ \boldsymbol{b}_{l-1}^{(l)} = \big[K_{\sigma_{1,l}}\big(\boldsymbol{x}_1^{'}, \boldsymbol{x}_l^{'}\big) \cdots K_{\sigma_{l-1,l}}\big(\boldsymbol{x}_{l-1}^{'}, \boldsymbol{x}_l^{'}\big)\big]^{\mathrm{T}}. \end{cases} \tag{15}$$

This recursion is initialized at the first step ($l = 1$) as

$$\boldsymbol{C}_1^{(1)} = K_{\sqrt{2}\sigma_1}\big(\boldsymbol{x}_1^{'}, \boldsymbol{x}_1^{'}\big) = \gamma_1 \tag{16}$$

and

$$\boldsymbol{p}_1^{(1)} = \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_1}\big(\boldsymbol{x}_k, \boldsymbol{x}_1^{'}\big). \tag{17}$$

By substituting (7) and (12)-(14) into (11), we have

$$\begin{aligned} &Q^{(l)}\big(\lambda_l, \sigma_l\big) = \\ &\begin{bmatrix} \lambda_l \boldsymbol{\beta}_{l-1}^{(l-1)} \\ 1 - \lambda_l \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{C}_{l-1}^{(l-1)} & \boldsymbol{b}_{l-1}^{(l)} \\ \big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} & \gamma_l \end{bmatrix} \begin{bmatrix} \lambda_l \boldsymbol{\beta}_{l-1}^{(l-1)} \\ 1 - \lambda_l \end{bmatrix} \\ &- 2\big[\lambda_l\big(\boldsymbol{\beta}_{l-1}^{(l-1)}\big)^{\mathrm{T}} \ 1 - \lambda_l\big] \begin{bmatrix} \boldsymbol{p}_{l-1}^{(l-1)} \\ \frac{1}{N}\sum\limits_{k=1}^{N} K_{\sigma_l}\big(\boldsymbol{x}_k, \boldsymbol{x}_l^{'}\big) \end{bmatrix} \\ &= \lambda_l^2 \mu^{(l-1)} + \big(1 - \lambda_l\big)^2 \gamma_l + 2\lambda_l\big(1 - \lambda_l\big)\big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)} \\ &- 2\lambda_l \nu^{(l-1)} - \frac{2\big(1 - \lambda_l\big)}{N}\sum_{k=1}^{N} K_{\sigma_l}\big(\boldsymbol{x}_k, \boldsymbol{x}_l^{'}\big). \end{aligned} \tag{18}$$

For $l > 1$, $Q^{(l)}\big(\lambda_l, \sigma_l\big)$ is a quadratic function with respect to λ_l. Hence there exits a unique minimum of $Q^{(l)}\big(\lambda_l, \sigma_l\big)$ for a given σ_l, which can be found by setting $\frac{\partial}{\partial \lambda_l} Q^{(l)}\big(\lambda_l, \sigma_l\big) = 0$, followed by the constraint satisfaction operation. This yields the closed-form solution for λ_l for the given σ_l as

$$\lambda_l = \min\left\{\max\left\{u_l, 0\right\}, 1\right\}, \tag{19}$$

with

$$u_l = \frac{\gamma_l - \big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)} + \nu^{(l-1)} - \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_l}\big(\boldsymbol{x}_k, \boldsymbol{x}_l^{'}\big)}{\mu^{(l-1)} + \gamma_l - 2\big(\boldsymbol{b}_{l-1}^{(l)}\big)^{\mathrm{T}} \boldsymbol{\beta}_{l-1}^{(l-1)}}. \tag{20}$$

It is easy to verify that the constraint satisfaction operator

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

Therefore, $0 \le \lambda_l \le 1$ is guaranteed. By plugging λ_l back to (18), we obtain the MISE value $Q^{(l)}\big(\lambda_l, \sigma_l\big)$ for this given kernel. The computational cost of parameter estimation for a kernel with fixed width is in the order of $\mathsf{O}(l)$, which is extremely low, owing to the recursive computation and the closed-form solution for the parameter λ_l when σ_l is fixed.

B. Kernel width optimization with MISE criterion

We now consider the problem of optimizing $K_{\sigma_l}\big(\boldsymbol{x}, \boldsymbol{x}_l'\big)$ by adjusting σ_l, also based on the MISE, when λ_l is fixed. Express (18) as

$$Q^{(l)}\big(\lambda, \sigma_l\big) = \lambda_l^2 \mu^{(l-1)} - 2\lambda_l \nu^{(l-1)} + S^{(l)}\big(\lambda_l, \sigma_l\big), \tag{22}$$

where

$$\begin{aligned} S^{(l)}\big(\lambda_l, \sigma_l\big) &= 2\lambda_l\big(1 - \lambda_l\big) \sum_{i=1}^{l-1} \beta_i^{(l-1)} K_{\sigma_{i,l}}(\boldsymbol{x}_i^{'}, \boldsymbol{x}_l^{'}) \\ &+ \big(1 - \lambda_l\big)^2 \gamma_l - \frac{2\big(1 - \lambda_l\big)}{N} \sum_{k=1}^{N} K_{\sigma_l}\big(\boldsymbol{x}_k, \boldsymbol{x}_l^{'}\big) \end{aligned} \tag{23}$$

which excludes all the components independent of σ_l. The gradient descent algorithm for minimizing $S^{(l)}\big(\lambda_l, \sigma_l\big)$ and hence $\mathrm{ISE}^{(l)} = \mathrm{ISE}\big(\lambda_l, \sigma_l\big)$ for the selected $\boldsymbol{x}_l^{'}$ and the fixed λ_l is given as follows.

Starting with $\sigma_l^{\text{old}} = \sigma_0$, repeat the following iterations for a sufficiently large number of times Iter, e.g., Iter $= 20$

$$\begin{cases} \sigma_l^{\text{new}} = \sigma_l^{\text{old}} - \eta \frac{\partial S^{(l)}\left(\lambda_l, \sigma_l^{\text{old}}\right)}{\partial \sigma_l}, \\ \sigma_l^{\text{old}} \leftarrow \max\{\sigma_l^{\text{new}}, \sigma_{\min}\}, \end{cases} \tag{24}$$

where $\eta > 0$ is a small positive learning rate, $\sigma_{\min}$ is a small positive value representing the lower bound of the kernel width parameter, and the gradient is given by

$$\begin{aligned} \frac{\partial S^{(l)}(\lambda_l, \sigma_l)}{\partial \sigma_l} &= 2\lambda_l(1-\lambda_l) \sum_{i=1}^{l-1} \beta_i^{(l-1)} \frac{\partial K_{\sigma_{i,l}}(\boldsymbol{x}_i', \boldsymbol{x}_l')}{\partial \sigma_l} \\ &- \frac{m(1-\lambda_l)^2 \gamma_l}{\sigma_l} - \frac{2(1-\lambda_l)}{N} \sum_{k=1}^{N} \frac{\partial K_{\sigma_l}(\boldsymbol{x}_k, \boldsymbol{x}_l')}{\partial \sigma_l} \\ &= 2\lambda_l(1-\lambda_l) \sum_{i=1}^{l-1} \beta_i^{(l-1)} K_{\sigma_{i,l}}(\boldsymbol{x}_i', \boldsymbol{x}_l') \Big(- \frac{m\sigma_l}{\sigma_i^2 + \sigma_l^2} + \\ &\frac{\|\boldsymbol{x}_i' - \boldsymbol{x}_l'\|^2 \sigma_l}{(\sigma_i^2 + \sigma_l^2)^2} \Big) - \frac{m(1-\lambda_l)^2 \gamma_l}{\sigma_l} - \\ &\frac{2(1-\lambda_l)}{N} \sum_{k=1}^{N} K_{\sigma_l}(\boldsymbol{x}_k, \boldsymbol{x}_l') \Big(- \frac{m}{\sigma_l} + \frac{\|\boldsymbol{x}_k - \boldsymbol{x}_l'\|^2}{\sigma_l^3} \Big). \end{aligned} \tag{25}$$

C. Joint kernel selection and parameter estimation algorithm

At the lth forward selection stage, a data sample is to be selected from the remaining $(N-l+1)$ candidate data samples based on the fixed kernel width σ_0, while the associated kernel width σ_l is optimized, and the l kernel weights are adjusted. More specifically, we initially review the contribution of each candidate data sample according to its associated MISE value, based on the fixed kernel width σ_0, and decide which is to be added to the model. The data point producing the smallest MISE value amongst all the candidate data samples is selected as $\boldsymbol{x}_l'$. With the kernel weights being fixed, we then adjust the kernel width σ_l using the gradient descent algorithm described in Section III-B. Finally, the optimal kernel weights are recalculated for the given σ_l as described in Section III-A.

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

$$\boldsymbol{X}_N^{(l-1)} = \Big[\boldsymbol{x}_1' \cdots \boldsymbol{x}_{l-1}' \ \boldsymbol{x}_l^{(l-1)} \cdots \boldsymbol{x}_N^{(l-1)} \Big], \tag{26}$$

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

$$\boldsymbol{q}_N^{(l-1)} = \Big[\frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_1^{'}\big) \cdots \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_{l-1}^{'}\big) \\ \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_l^{(l-1)}\big) \cdots \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_N^{(l-1)}\big)\Big], \tag{27}$$

with

$$\boldsymbol{X}_N^{(0)} = \Big[\boldsymbol{x}_1^{(0)}\ \boldsymbol{x}_2^{(0)} \cdots \boldsymbol{x}_N^{(0)}\Big] = \Big[\boldsymbol{x}_1\ \boldsymbol{x}_2 \cdots \boldsymbol{x}_N\Big], \tag{28}$$

$$\boldsymbol{q}_N^{(0)} = \Big[\frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_1\big)\ \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_2\big) \cdots \\ \frac{1}{N}\sum_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_N\big)\Big]. \tag{29}$$

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 $q^{(l-1)}(j) = \frac{1}{N}\sum\limits_{k=1}^{N} K_{\sigma_0}\big(\boldsymbol{x}_k, \boldsymbol{x}_j^{(l-1)}\big)$ 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 $\lambda_1 = 0$.

Step 1). For $1 \le j \le N$, based on σ_0, compute

$$Q^{(1,j)} = \gamma - 2\boldsymbol{p}_1^{(1,j)}, \tag{30}$$

where $\gamma = \frac{1}{(4\pi\sigma_0^2)^{m/2}}$ and $\boldsymbol{p}_1^{(1,j)} = q^{(0)}(j)$.

Step 2). Find

$$Q^{(1,j_1)} = \min\left\{Q^{(1,j)}, 1 \le j \le N\right\}. \tag{31}$$

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.

Step 3). Apply (24) to find σ_1.

Step 4). Calculate $\mu^{(1)} = \boldsymbol{C}_1^{(1)}$ and $\nu^{(1)} = \boldsymbol{p}_1^{(1)}$ using (16) and (17). Update $Q^{(1)} = \mu^{(1)} - 2\nu^{(1)}$.

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

Step 1). For $l \leq j \leq N$, set $\sigma_j = \sigma_0$, compute

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

and

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

Step 2): Find

$$Q^{(l,j_l)} = \min\left\{ Q^{(l,j)},\ l \leq j \leq N \right\}. \tag{32}$$

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.

Step 3): With $\lambda_l = \lambda_l^{(j_l)}$, calculate $\boldsymbol{\beta}_l^{(l)}$ using (7). Then apply (24) to find σ_l.

Step 4). Update $\boldsymbol{p}_l^{(l)}$, $\boldsymbol{C}_l^{(l)}$. Recalculate λ_l using (19) and (20). Recalculate $\boldsymbol{\beta}_l^{(l)}$ using (7). Update $\mu^{(l)}$, $\nu^{(l)}$ and $Q^{(l)}(\lambda_l, \sigma_l)$ using (11)-(14).

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

$$\left| Q^{(M+1)} - Q^{(M)} \right| \leq \delta Q,$$

where δQ is a predetermined very small positive number, and this produces a subset model with the M significant kernels.

TABLE I

COMPUTATIONAL COST OF THE PROPOSED ALGORITHM IN COMPARISON TO THE FCR-MISE ALGORITHM AT THE lTH FORWARD STEP.

Method	Kernel selection	Kernel width tuning	kernel weight re-estimation
FCR-MISE	$(N-l+1)\mathsf{O}(l)$	none	none
The proposed	$(N-l+1)\mathsf{O}(l)$	Iter $\times\ \mathsf{O}(N+l)$	$\mathsf{O}(N)$

D. Remarks:

Remark 1: The reason that the optimisation of (18) with respect to λ_l and σ_l is carried out separately is that the optimal value λ_l can be expressed in closed form for fixed σ_l, thus significantly reducing computational costs. Alternatively both of them can be optimized using gradient descent algorithm simultaneously. Since the relationship with respect to σ_l is not quadratic, the results will not be the same, each only achieving a local minimum. However the property that (18) is quadratic in λ_l cannot be exploited for computational advantage.

Remark 2: In FCR-MISE algorithm [19], each kernel has a common fixed width, and appropriate kernel value can be determined empirically through trial and error based on cross-validation. More specifically, a suitable kernel width value can be found using a line search based on the cross-validation performance. In the proposed algorithm, the kernel width is given as a $\boldsymbol{\sigma}_M$, where each element in $\boldsymbol{\sigma}_M$ is optimized from an initial σ_0 which needs to be preset. Unlike the fixed kernel width in FCR-MISE algorithm [19], the choice of σ_0 is more relaxed, since there is a wide range of suitable values.

E. Computational cost

The proposed algorithm is an extension to the low cost FCR-MISE algorithm [19], with the difference that each kernel is tuned after it has been selected. The FCR-MISE algorithm [19] has a significant advantage in that it offers a much lower complexity in constructing PDF estimate than other existing sparse estimators with $O(N^2)$ complexity. Table I compares the computational cost of the proposed algorithm with that of the FCR-MISE algorithm at the lth forward step. Overall the computational cost is increased at each forward stage, compared to the FCR-MISE algorithm. Since the tuning of the kernel is only applied to the selected kernel, the extra cost is small. In contrast to our proposed algorithm which automatically tunes each kernel width, however, there exists extra computational cost for any estimator based on a pre-set fixed single

kernel width, such as the FCR-MISE algorithm, since this kernel width has to be empirically tuned outside the algorithm loop. Moreover, the total computational cost of an algorithm is dependent on the model size M of the final selected model. Since M is usually much smaller than N, the total computational cost is approximately linear with respect to the model size M. Since our proposed algorithm can produce a much smaller model, its total computational cost can actually be lower than that of the FCR-MISE algorithm.

IV. Simulation Study

Two numerical examples are provided. In each example, we randomly draw a data set of N points from a known distribution $p(\boldsymbol{x})$ to construct the PDF estimate $\widehat{p}^{(M)}\big(\boldsymbol{x}_k; \boldsymbol{\beta}_M, \boldsymbol{\sigma}_M\big)$ based on the proposed approach. A separate test data set of $N_{\text{test}} = 10000$ sample points was used for evaluation according to the L_1 norm

$$L_1 = \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} \big| p\big(\boldsymbol{x}_k\big) - \widehat{p}^{(M)}\big(\boldsymbol{x}_k; \boldsymbol{\beta}_M, \boldsymbol{\sigma}_M\big) \big|. \tag{33}$$

The experiment was repeated for 100 different random runs.

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

$$\begin{aligned} p(\boldsymbol{x}) =& \frac{1}{4\pi} \exp\left(-\frac{(x_1-2)^2}{2}\right) \exp\left(-\frac{(x_2-2)^2}{2}\right) \\ &+ \frac{0.35}{8} \exp(-0.7|x_1+2|) \exp(-0.5|x_2+2|). \end{aligned} \tag{34}$$

The estimation data set had $N = 500$ samples.

Example 2: The density to be estimated for this 6-D example was the mixture of three Gaussians defined by

$$\begin{aligned} p(\boldsymbol{x}) =& \frac{1}{3} \sum_{i=1}^{3} \frac{1}{(2\pi)^3 \sqrt{\det(\boldsymbol{\Gamma}_i)}} \\ &\times \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^{\mathrm{T}} \boldsymbol{\Gamma}_i^{-1} (\mathbf{x}-\boldsymbol{\mu}_i)\right), \end{aligned} \tag{35}$$

with $\boldsymbol{\mu}_1 = [1\ 1\ 1\ 1\ 1\ 1]^{\mathrm{T}}$, $\boldsymbol{\mu}_2 = [-1\ -1\ -1\ -1\ -1\ -1]^{\mathrm{T}}$, $\boldsymbol{\mu}_3 = [0\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}$, $\boldsymbol{\Gamma}_1 = \text{diag}\{1,2,1,2,1,2\}$, $\boldsymbol{\Gamma}_2 = \text{diag}\{2,1,2,1,2,1\}$, and $\boldsymbol{\Gamma}_3 = \text{diag}\{2,1,2,1,2,1\}$. The estimation data set had $N = 600$ samples.

TABLE II

PERFORMANCE COMPARISON OF KERNEL DENSITY ESTIMATORS.

(a) *Example 1*

Method	L_1 test error (mean $\pm$ STD)	Kernel number (mean $\pm$ STD)
PW	$(4.18 \pm 0.8) \times 10^{-3}$	500 ± 0
SDC [17]	$(3.83 \pm 0.8) \times 10^{-3}$	11.9 ± 2.6
SKD [18]	$(3.84 \pm 0.8) \times 10^{-3}$	15.3 ± 3.9
RSDE-MNQP [11]	$(4.24 \pm 0.8) \times 10^{-3}$	129.4 ± 35.7
FCR-MISE [19]	$(3.33 \pm 0.8) \times 10^{-3}$	25.1 ± 2.7
RTR-MISE [14]	$(3.13 \pm 0.7) \times 10^{-3}$	36.7 ± 11.3
The proposed	$(3.57 \pm 0.7) \times 10^{-3}$	7.6 ± 1.4

(b) *Example 2*

Method	L_1 test error (mean $\pm$ STD)	Kernel number (mean $\pm$ STD)
PW	$(3.18 \pm 0.13) \times 10^{-5}$	600 ± 0
SDC [17]	$(4.48 \pm 1.2) \times 10^{-5}$	14.9 ± 2.1
SKD [18]	$(3.11 \pm 0.5) \times 10^{-5}$	9.4 ± 1.9
RSDE-MNQP [11]	$(3.67 \pm 0.7) \times 10^{-5}$	29.4 ± 10.1
FCR-MISE [19]	$(2.82 \pm 0.1) \times 10^{-5}$	19.4 ± 0.9
RTR-MISE [14]	$(2.53 \pm 0.1) \times 10^{-5}$	81.2 ± 20
The proposed	$(2.64 \pm 0.2) \times 10^{-5}$	2.9 ± 0.2

Six methods were used for comparison: (a) the well known PW estimate; (b) the sparse density construction (SDC) algorithm [17]; (c) the sparse kernel density construction (SKD) algorithm [18]; (d) the reduced set density estimator with multiplicative nonnegative quadratic programming (RSDE-MNQP) [11]; (e) the FCR-MISE algorithm [19]; and (f) the RTR-MISE algorithm [14].

We briefly explain these six algorithms. Both the SDC algorithm [17] and the SKD algorithm [18] are regression-based PDF estimation methods that construct sparse PDF forwardly. For the SDC algorithm, the empirical CDF is constructed and used as the desired response, but for the SKD algorithm the PW estimate is constructed and used as the desired response. The RSDE-MNQP [11], the FCR-MISE [19] and the RTR-MISE [14] are all based on the MISE, but employ different optimization algorithms. Specifically, the RSDE-MNQP algorithm uses the MNQP algorithm, the FCR-MISE algorithm formulates the density estimation in a forward constrained regression manner by selecting one kernel at a time forwardly, and the RTR-MISE algorithm is based on the Riemannian trust-region algorithm [13]. We also point out that the MISE cost function is used in PW estimate using grid search for an optimal kernel width. However, the single kernel width for the other five algorithms needs to be preset empirically.

The algorithmic parameters of the proposed approach were set to $\sigma_{\min} = 0.1$ and $\sigma_{\min} = 1$ for *Example 1* and *Example 2*, respectively, Iter $= 20$ and $\eta = 0.02$ for the both examples, while δQ was set to 10^{-4} and 10^{-5}, respectively for the two example. The results obtained by the seven kernel density estimators are listed in Table II (a) and (b), respectively, for the two examples, where the results of the SDC, SKD, FCR-MISE and RTR-MISE are quoted from [14], [17]–[19], respectively. The results of Table II clearly show that our proposed algorithm can construct much sparser kernel density estimates than the five state-of-the-art benchmark sparse kernel density estimators compared, with competitive accuracy. Compared to the low cost FCR-MISE algorithm, the proposed algorithm increases the computational complexity per forward step of Iter $\times \mathsf{O}(N+l)$ due to the tunable kernel calculation. However it is clear that the resultant models are much sparser leading to fewer forward regression steps for computational cost reduction. Note that the computational costs of [19] have already been shown to be better than the other algorithms.

V. Conclusions

We have introduced a new sparse kernel density estimator with tunable kernels based on the idea of forward constrained regression by adding one kernel at a time based on the minimum ISE criterion. Our main contribution has been to develop a new recursive algorithm which selects a significant kernel at each forward construction stage, and then optimizes the kernel width of the selected kernels based on the gradient descent algorithm. The significant advantages of the proposed method are that it is able to obtain very sparse PDF estimates due to the individually tunable kernel width parameters, and it requires no empirically predetermined parameters outside the algorithm. Numerical examples have been employed to demonstrate that the proposed approach can construct *very sparse* kernel density estimators with competitive accuracy to the existing state-of-the-art sparse kernel density estimators.

APPENDIX

INTEGRATING $\int K_{\sigma_i}(\boldsymbol{x}, \boldsymbol{x}_i^{'}) K_{\sigma_j}(\boldsymbol{x}, \boldsymbol{x}_j^{'}) d\boldsymbol{x}$

With the notations $\boldsymbol{x} = \left[x_1 \ x_2 \cdots x_m\right]^{\mathrm{T}}$ and $\boldsymbol{x}_i^{'} = \left[x_{i,1}^{'} \ x_{i,2}^{'} \cdots x_{i,m}^{'}\right]^{\mathrm{T}}$ for $1 \leq i \leq l$, we have

$$\int K_{\sigma_i}(\boldsymbol{x}, \boldsymbol{x}_i^{'}) K_{\sigma_j}(\boldsymbol{x}, \boldsymbol{x}_j^{'}) d\boldsymbol{x} = \frac{1}{(2\pi\sigma_i\sigma_j)^m} \times \prod_{k=1}^{m} \int \exp\Big(- \frac{(x_k - x_{i,k}^{'})^2}{2\sigma_i^2} - \frac{(x_k - x_{j,k}^{'})^2}{2\sigma_j^2} \Big) dx_k \tag{36}$$

in which

$$\begin{aligned}
&\int \exp\Big(- \frac{(x_k - x_{i,k}^{'})^2}{2\sigma_i^2} - \frac{(x_k - x_{j,k}^{'})^2}{2\sigma_j^2} \Big) dx_k \\
&\qquad (\sigma_i^2 + \sigma_j^2) x_k^2 - 2(x_{i,k}^{'}\sigma_j^2 + x_{j,k}^{'}\sigma_i^2) x_k \\
&= \int \exp\Big(- \frac{+(x_{i,k}^{'})^2\sigma_j^2 + (x_{j,k}^{'})^2\sigma_i^2}{2\sigma_i^2\sigma_j^2} \Big) dx_k \\
&= \exp\Big(- \frac{\frac{(x_{i,k}^{'})^2\sigma_j^2 + (x_{j,k}^{'})^2\sigma_i^2}{\sigma_i^2 + \sigma_j^2} - \big(\frac{x_{i,k}^{'}\sigma_j^2 + x_{j,k}^{'}\sigma_i^2}{\sigma_i^2 + \sigma_j^2}\big)^2}{2\sigma_i^2\sigma_j^2/(\sigma_i^2 + \sigma_j^2)} \Big) \\
&\times \int \exp\Big(- \frac{\big(x_k - (x_{i,k}^{'}\sigma_j^2 + x_{j,k}^{'}\sigma_i^2)/(\sigma_i^2 + \sigma_j^2)\big)^2}{2\sigma_i^2\sigma_j^2/(\sigma_i^2 + \sigma_j^2)} \Big) dx_k \\
&= \exp\Big(- \frac{(x_{i,k}^{'} - x_{j,k}^{'})^2}{2(\sigma_i^2 + \sigma_j^2)} \Big) \\
&\times \int \exp\Big(- \frac{\big(x_k - (x_{i,k}^{'}\sigma_j^2 + x_{j,k}^{'}\sigma_i^2)/(\sigma_i^2 + \sigma_j^2)\big)^2}{2\sigma_i^2\sigma_j^2/(\sigma_i^2 + \sigma_j^2)} \Big) dx_k.
\end{aligned} \tag{37}$$

Noting $\int \frac{1}{\sqrt{2\pi s}} \exp\Big(- \frac{(x-\mu)^2}{2s} \Big) dx = 1$, we have

$$\begin{aligned}
&\int \exp\Big(- \frac{(x_k - x_{i,k}^{'})^2}{2\sigma_i^2} - \frac{(x_k - x_{j,k}^{'})^2}{2\sigma_j^2} \Big) dx_k \\
&= \sqrt{2\pi\sigma_i^2\sigma_j^2/(\sigma_i^2 + \sigma_j^2)} \exp\Big(- \frac{(x_{i,k}^{'} - x_{j,k}^{'})^2}{2(\sigma_i^2 + \sigma_j^2)} \Big)
\end{aligned} \tag{38}$$

so that

$$\begin{aligned}
&\int K_{\sigma_i}(\boldsymbol{x}, \boldsymbol{x}_i^{'}) K_{\sigma_j}(\boldsymbol{x}, \boldsymbol{x}_j^{'}) d\boldsymbol{x} \\
&= \frac{1}{(2\pi\sigma_{i,j}^2)^{m/2}} \exp\Big(- \frac{\|\boldsymbol{x}_i^{'} - \boldsymbol{x}_j^{'}\|^2}{2\sigma_{i,j}^2} \Big) \\
&= K_{\sigma_{i,j}}(\boldsymbol{x}_i^{'}, \boldsymbol{x}_j^{'})
\end{aligned} \tag{39}$$

with $\sigma_{i,j} = \sqrt{\sigma_i^2 + \sigma_j^2}$.

REFERENCES

[1] G. 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] 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.

[5] L. Rutkowski, "Adaptive probabilistic neural networks for pattern classification in time-varying environment," *IEEE Trans. Neural Networks*, vol .15, no. 4, pp 811–827, July 2004.

[6] H. Yin and N. W. Allinson, "Self-organizing mixture networks for probability density estimation," *IEEE Trans. Neural Networks*, vol. 12, no. 2, pp 405–411, March 2001.

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

[9] 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.

[10] 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.

[11] 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.

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

[13] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, "Manopt, a Matlab toolbox for optimization on manifolds," *J. Machine Learning Research*, vol. 15, pp. 1455–1450, April 2014.

[14] X. Hong, J. B. Gao, S. Chen, and T. Zia, "Sparse density estimation on the multinomial manifold," *IEEE Trans. Neural Networks and Learning Systems*, To appear, 2015.

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

[16] S. Chen, X. Hong, C. J. Harris, and P. M. Sharkey, "Sparse modeling using forward regression with PRESS statistic and regularization," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol. 34, no. 2, pp. 898–911, April 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. 4-6, pp. 931–943, Jan. 2008.

[19] X. Hong, S. Chen, A. Qatawneh, K. Daqrouq, M. Sheikh, and A. Morfeq, "Sparse probability density function estimation using the minimum integrated square error," *Neurocomputing*, vol. 115, pp. 122–129, 2013.

[20] 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.