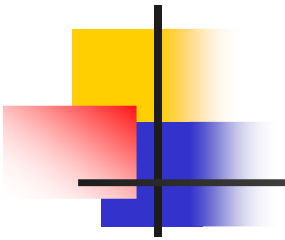
WCCI 2008 Presentation

Sparse Kernel Density Estimator Using Orthogonal Regression Based on D-Optimality Experimental Design

Sheng Chen[†], Xia Hong[‡] and Chris J. Harris[†]

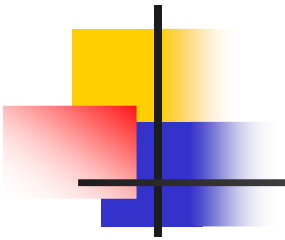
[†] **School of Electronics and Computer Science**
University of Southampton
Southampton SO17 1BJ, UK

[‡] **School of Systems Engineering**
University of Reading
Reading RG6 6AY, UK



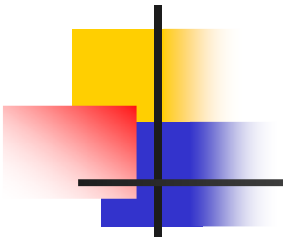
Outline

- Overview of existing **density estimation** methods
- Proposed sparse kernel density estimator:
  - Convert **unsupervised** density learning into **constrained regression** by adopting **Parzen window estimate** as desired response
  - Unsupervised **orthogonal forward regression** based on D-optimality experimental design to determine structure
  - **Multiplicative nonnegative quadratic programming** to calculate kernel weights
- Empirical investigation and performance comparison



Overview of Existing Density Estimators

- Parametric **Gaussian mixture model**, GMM
  - Nonlinear optimisation by EM algorithm to determine all parameters
  - Need to determine number of components
- Non-parametric **Parzen window estimator**, PWE
  - Extremely simple and accurate, non-sparse with high test complexity
  - Need to determine kernel width
- **Sparse kernel density estimators** by making some weights zeros
  - **SVM**, estimating in CDF space with EDF as desired response
  - Reduced set density estimator, **RSDE**, (Girolami and He, 2003), similar to SVM with different criterion
  - Need to determine kernel width



Existing Sparse Estimators (continue)

- Select SKDEs by **orthogonal forward regression**
- Estimating in CDF space with EDF as desired response
  - Selection by minimising training MSE (Choudhury, 2003)
  - Selection by minimising **leave-one-out** MSE with local regularisation, LOO-MSE-LR, (Chen *et al*, 2004)
  - Need to determine kernel width, *ad hoc* mechanisms to ensure nonnegative and unity constraints for kernel weights (increase computation)
- Estimating in original PDF space with PWE as desired response
  - Selection by LOO-MSE-LR and MNQP algorithm for kernel weights, **LOO-MSE-LR+MNQP**, (Chen *et al*, 2008)
  - Need to determine kernel width

Problem Formulation

- Given a realisation sample $D_N = \{\mathbf{x}_k\}_{k=1}^N$, drawn from unknown density $p(\mathbf{x})$, provide a **kernel density estimate**

$$\hat{p}(\mathbf{x}; \boldsymbol{\beta}_N, \rho) = \sum_{k=1}^{N} \beta_k K_\rho(\mathbf{x}, \mathbf{x}_k)$$

subject to: $\beta_k \geq 0,\ 1 \leq k \leq N$, and $\boldsymbol{\beta}_N^T \mathbf{1}_N = 1$

- **Unsupervised** learning, no desired response $y_k = p(\mathbf{x}_k)$ for estimator
- **Parzen window estimate** $\hat{p}(\mathbf{x}; \mathbf{1}_N/N, \rho_{\text{Par}})$:
  - Place a "conditional" unimodal PDF $K_{\rho_{\text{Par}}}(\mathbf{x}, \mathbf{x}_k)$ at each $\mathbf{x}_k$ and average over all samples with equal weighting
  - Kernel width ρ_{Par} has to be determined via cross validation
  - **Remarkably simple and accurate** but non-sparse

Regression-Based Approach

- View PW estimate as "observation" of true density contaminated by some "observation noise" and use it as **desired response**

$$\hat{p}(\mathbf{x}; \mathbf{1}_N/N, \rho_{\text{Par}}) = \sum_{k=1}^{N} \beta_k K_\rho(\mathbf{x}, \mathbf{x}_k) + \epsilon(\mathbf{x})$$

- Let $y_k = \hat{p}(\mathbf{x}_k; \mathbf{1}_N/N, \rho_{\text{Par}})$ at $\mathbf{x}_k \in D_N$, this model is expressed as

$$y_k = \hat{y}_k + \epsilon(k) = \boldsymbol{\phi}^T(k)\boldsymbol{\beta}_N + \epsilon(k)$$

where $\boldsymbol{\phi}(k) = [K_{k,1}\ K_{k,2} \cdots K_{k,N}]^T$ with $K_{k,i} = K_\rho(\mathbf{x}_k, \mathbf{x}_i)$, $\epsilon(k) = \epsilon(\mathbf{x}_k)$

- This is standard **regression** model, which over D_N can be written as

$$\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\beta}_N + \boldsymbol{\epsilon}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1\ \boldsymbol{\phi}_2 \cdots \boldsymbol{\phi}_N]$ with $\boldsymbol{\phi}_k = [K_{1,k}\ K_{2,k} \cdots K_{N,k}]^T$, $\boldsymbol{\epsilon} = [\epsilon(1)\ \epsilon(2) \cdots \epsilon(N)]^T$, $\mathbf{y} = [y_1\ y_2 \cdots y_N]^T$

Orthogonal Decomposition

- An **orthogonal decomposition** of regression matrix is $\mathbf{\Phi} = \mathbf{WA}$, where

$$\mathbf{W} = [\mathbf{w}_1 \ \mathbf{w}_2 \cdots \mathbf{w}_N]$$

with orthogonal columns satisfying $\mathbf{w}_i^T \mathbf{w}_j = 0$, if $i \neq j$, and

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

- **Regression model** can alternatively be expressed as

$$\mathbf{y} = \mathbf{W}\mathbf{g}_N + \boldsymbol{\epsilon}$$

where new weight vector $\mathbf{g}_N = [g_1 \ g_2 \cdots g_N]^T$ satisfies $\mathbf{A}\boldsymbol{\beta}_N = \mathbf{g}_N$

D-Optimality Based Construction

- *D*-**optimality** criterion: select N_s-term SKDE such that determinant of resulting subset **design matrix**, $\det\left(\mathbf{\Phi}_{N_s}^T \mathbf{\Phi}_{N_s}\right)$, is maximised
- Note

$$\log\left(\det\left(\mathbf{\Phi}^T\mathbf{\Phi}\right)\right) = \log\left(\det\left(\mathbf{W}^T\mathbf{W}\right)\right) = \sum_{i=1}^{N} \log\left(\mathbf{w}_i^T\mathbf{w}_i\right)$$

  Selected N_s terms corresponding to N_s largest **eigenvalues** of $\mathbf{\Phi}^T\mathbf{\Phi}$
- **Unsupervised** procedure depending on $D_N = \{\mathbf{x}_k\}_{k=1}^N$ only
- **Fast algorithm** of modified Gram-Schmidt orthogonalisation procedure can be used to select N_s kernels using *D*-optimality based OFR
- $N_s \ll N$, resulting **very sparse** kernel density estimate

Proposed Algorithm

- Fast algorithm based on *D*-**optimality** criterion selects N_s significant kernels, $\mathbf{\Phi}_{N_s}$
- Kernel weight vector $\boldsymbol{\beta}_{N_s}$ is calculated using **multiplicative non-negative quadratic programming** to solve constrained nonnegative quadratic programming

$$\min_{\boldsymbol{\beta}_{N_s}} \{ \tfrac{1}{2} \boldsymbol{\beta}_{N_s}^T \mathbf{B}_{N_s} \boldsymbol{\beta}_{N_s} - \mathbf{v}_{N_s}^T \boldsymbol{\beta}_{N_s} \}$$

$$\text{s.t. } \boldsymbol{\beta}_{N_s}^T \mathbf{1}_{N_s} = 1 \text{ and } \beta_i \geq 0,\ 1 \leq i \leq N_s,$$

where $\mathbf{B}_{N_s} = \mathbf{\Phi}_{N_s}^T \mathbf{\Phi}_{N_s}$ is selected subset design matrix, $\mathbf{v}_{N_s} = \mathbf{\Phi}_{N_s}^T \mathbf{y}$

- Since $N_s \ll N$, MNQP algorithm requires little extra computation and it may set some kernel weights to (near) zero, further reduce model size

Simulation Set Up

- For **density estimation**, N-sample training set for estimation, and test set of $N_{\text{test}} = 10,000$ samples for calculating L_1 test error

$$L_1 = \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} \left| p(\mathbf{x}_k) - \hat{p}(\mathbf{x}_k; \boldsymbol{\beta}_{N_s}, \rho) \right|$$

Kullback-Leibler divergence was also approximated for 1 or 2-D cases

$$D_{\text{KL}}(p|\hat{p}) = \int_{\mathcal{R}^m} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{\hat{p}(\mathbf{x}_k; \boldsymbol{\beta}_{N_s}, \rho)} \, d\mathbf{x}$$

Experiment was repeated N_{run} random runs

- For two-class **classification**, $\hat{p}(\mathbf{x}; \boldsymbol{\beta}_{N_s}, \rho|C0)$ and $\hat{p}(\mathbf{x}; \boldsymbol{\beta}_{N_s}, \rho|C1)$, two class conditional PDF estimates, were estimated, and Bayes' decision

$$\left.\begin{array}{ll} \text{if } \hat{p}(\mathbf{x}; \boldsymbol{\beta}_{N_s}, \rho|\text{C0}) \geq \hat{p}(\mathbf{x}; \boldsymbol{\beta}_{N_s}, \rho|\text{C1}), & \mathbf{x} \in \text{C0} \\ \text{else}, & \mathbf{x} \in \text{C1} \end{array}\right\}$$

was then applied to test data set

One-Dimension Example

- True density was **mixture** of Gaussian and Laplacian distributions

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

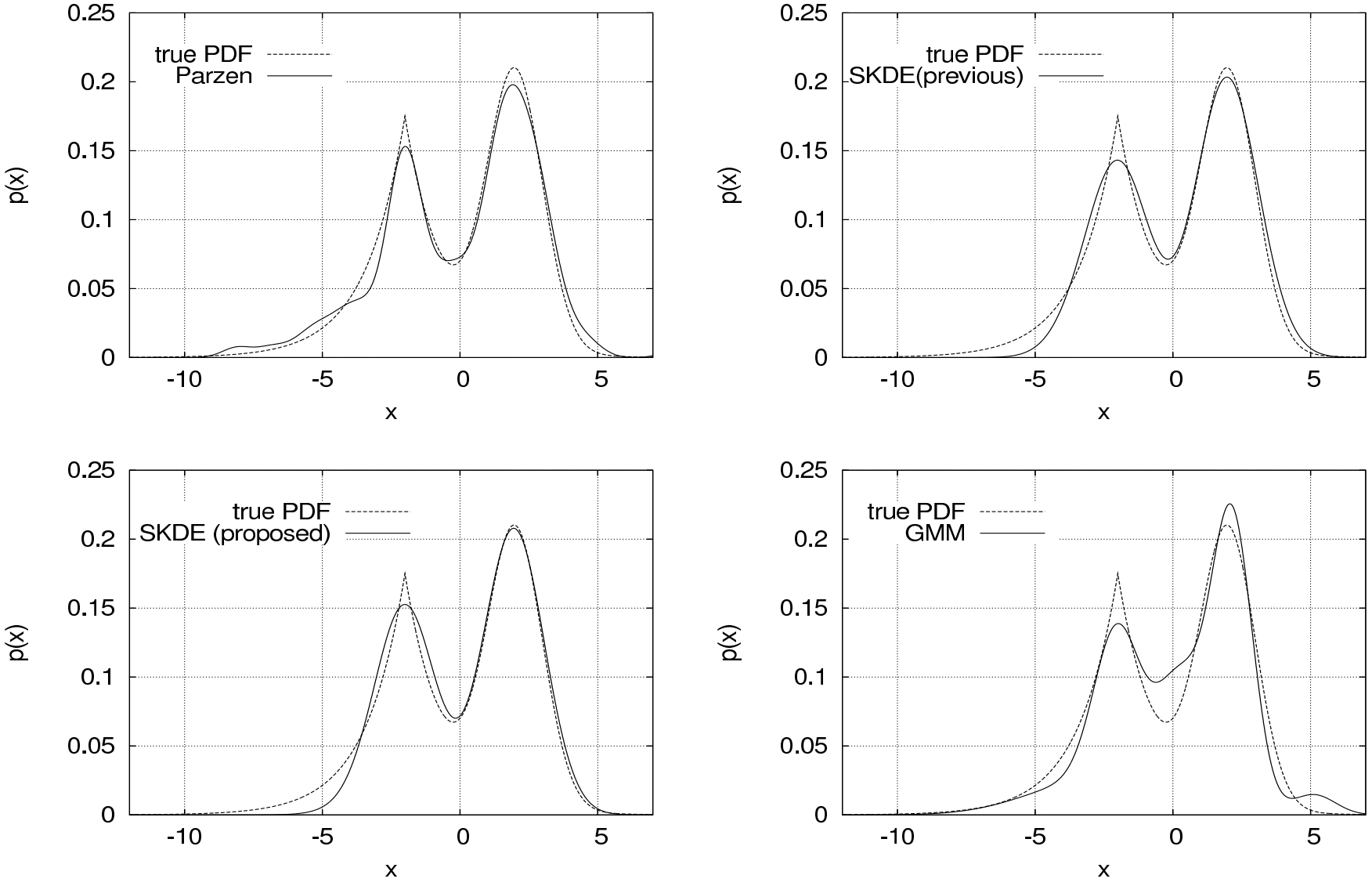
$N = 100$ and $N_{\mathrm{run}} = 1000$

- Performance comparison in terms of **KL divergence**, L_1 **test error** and **number of kernels** required, quoted as mean $\pm$ standard deviation

estimator	KL divergence	L_1 test error	kernel no.
GMM	$(12.074 \pm 7.885) \times 10^{-2}$	$(2.511 \pm 0.904) \times 10^{-2}$	5 ± 0
PWE	$(8.090 \pm 5.198) \times 10^{-2}$	$(2.011 \pm 0.621) \times 10^{-2}$	100 ± 0
Previous	$(8.657 \pm 5.122) \times 10^{-2}$	$(2.010 \pm 0.649) \times 10^{-2}$	5.2 ± 1.2
proposed	$(8.308 \pm 3.931) \times 10^{-2}$	$(1.945 \pm 0.644) \times 10^{-2}$	4.6 ± 0.8

SKDE: **Previous** (LOO-MSE-LR+MNQP), **proposed** (D-optimality+MNQP)

One-D Example (continue)



Two-Dimension Example

- True density was **mixture** of five Gaussian distributions

$$p(x,y) = \sum_{i=1}^{5} \frac{1}{10\pi} e^{-\frac{(x-\mu_{i,1})^2}{2}} e^{-\frac{(y-\mu_{i,2})^2}{2}}$$

with means $(\mu_{i,1}, \mu_{i,2})$: $(0,-4)$, $(0,-2)$, $(0,0)$, $(-2,0)$ and $(-4,0)$. $N = 500$ and $N_{\mathrm{run}} = 100$

- Performance comparison in terms of **KL divergence**, L_1 **test error** and **number of kernels** required, quoted as mean $\pm$ standard deviation

estimator	KL divergence	L_1 test error	kernel no.
GMM	$(3.392 \pm 0.870) \times 10^{-2}$	$(3.675 \pm 0.672) \times 10^{-3}$	8 ± 0
PWE	$(3.422 \pm 0.548) \times 10^{-2}$	$(3.620 \pm 0.439) \times 10^{-3}$	500 ± 0
Previous	$(3.664 \pm 0.920) \times 10^{-2}$	$(3.610 \pm 0.502) \times 10^{-3}$	13.2 ± 2.9
proposed	$(3.474 \pm 1.298) \times 10^{-2}$	$(3.236 \pm 0.558) \times 10^{-3}$	7.9 ± 0.8

Two-Class Two-Dimension Example

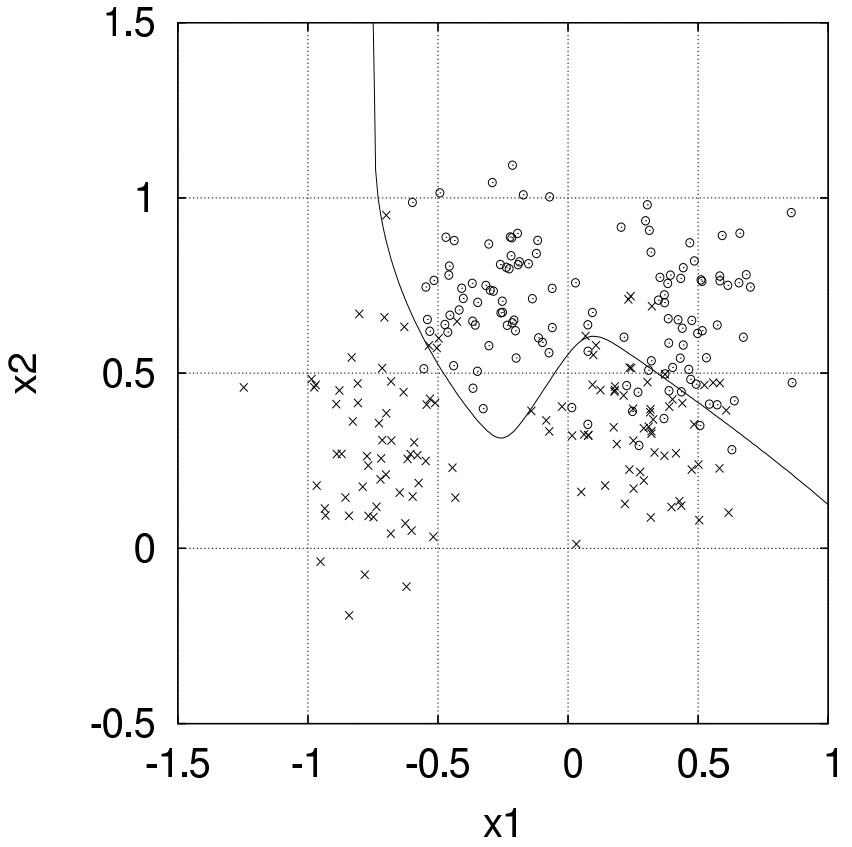
- `http://www.stats.ox.ac.uk/PRNN/`: two-class **classification** problem in two-dimensional feature space
- Training set contained 250 samples with 125 points for each class, test set had 1000 points with 500 samples for each class, and optimal Bayes test error rate based on true probability distribution was 8%
- Performance comparison in terms of test error rate and number of kernels

method	$\hat{p}(\bullet\|C0)$	$\hat{p}(\bullet\|C1)$	test error rate
GMM	2 components	2 components	9.0%
PWE	125 kernels	125 kernels	8.0%
Previous SKDE	6 kernels	5 kernels	8.0%
Proposed SKDE	2 kernels	2 kernels	8.0%

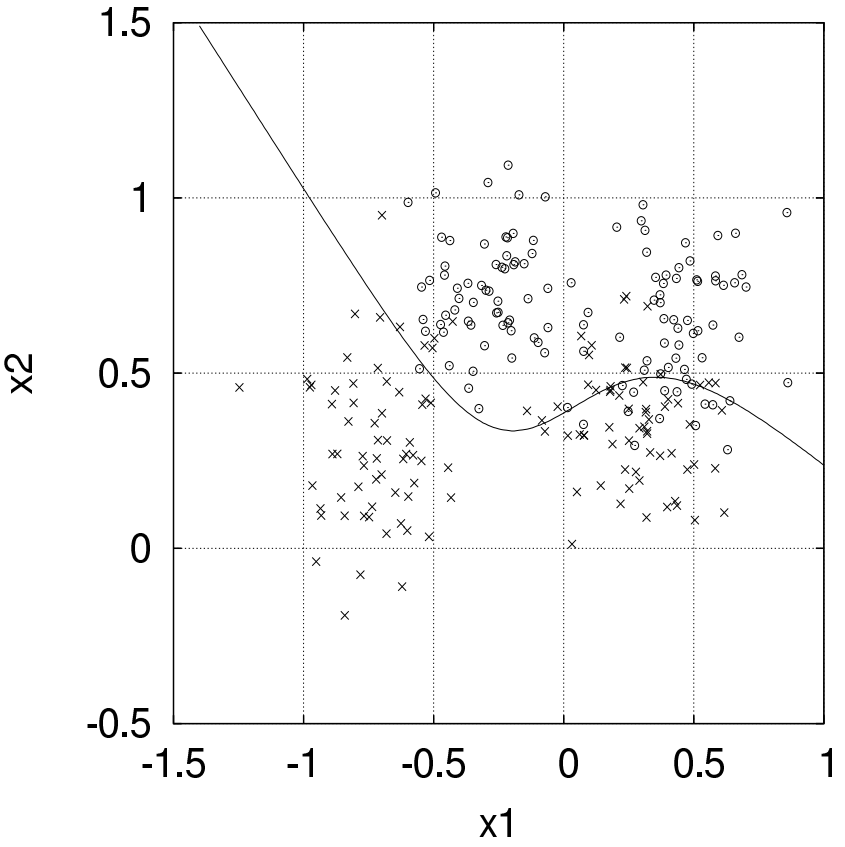
Two-class Two-D Example (continue)

Decision boundary of (a) GMM estimate, and (b) proposed SKD estimate, where circles and crosses represent class-1 and class-0 training data, respectively

(a)



(b)



Six-Dimension Example

- Density to be estimated was **mixture** of three Gaussian distributions

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

$$\boldsymbol{\mu}_1 = [1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0]^T,\ \mathbf{\Gamma}_1 = \mathrm{diag}\{1.0, 2.0, 1.0, 2.0, 1.0, 2.0\}$$

$$\boldsymbol{\mu}_2 = [-1.0\ -1.0\ -1.0\ -1.0\ -1.0\ -1.0]^T,\ \mathbf{\Gamma}_2 = \mathrm{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}$$

$$\boldsymbol{\mu}_3 = [0.0\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0]^T,\ \mathbf{\Gamma}_3 = \mathrm{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}$$

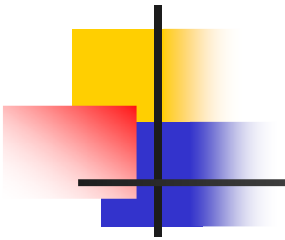
- $N = 600$, performance comparison over $N_{\mathrm{run}} = 100$ runs

method	L_1 test error	kernel number
GMM estimator	$(1.7428 \pm 0.2852) \times 10^{-5}$	8 ± 0
PW estimator	$(3.5195 \pm 0.1616) \times 10^{-5}$	600 ± 0
Previous SKDE	$(3.1134 \pm 0.5335) \times 10^{-5}$	9.4 ± 1.9
Proposed SKDE	$(2.7823 \pm 0.2271) \times 10^{-5}$	8.4 ± 0.9

Titanic Data Set

- `http://ida.first.fhg.de/projects/bench/benchmarks.htm`: two-class three-dimensional **Titanic** data set
- 100 realisations, each realisation contained 150 training samples and 2051 test data samples
- Two-class data samples are **imbalanced**, with class-0 training samples approximately twice of class-1 training samples
- Performance comparison in terms of test error rate and number of kernels

method	kernel no. $\hat{p}(\bullet\|\mathrm{C0}) + \hat{p}(\bullet\|\mathrm{C1})$	test error rate in %
GMM	8 ± 0	23.86 ± 3.22
PWE	150 ± 0	22.48 ± 0.43
Proposed SKDE	7.8 ± 4.4	22.34 ± 0.34



Conclusions

- A regression-based **sparse kernel density estimator** has been proposed
  - Density learning is converted into **constrained regression** using Parzen window estimate as desired response
  - Unsupervised **orthogonal forward regression** based on D-optimality experimental design to determine structure of kernel density estimate
  - **Multiplicative nonnegative quadratic programming** is used to calculate associated kernel weights
- Effectiveness of proposed sparse kernel density estimator has been demonstrated using simulation