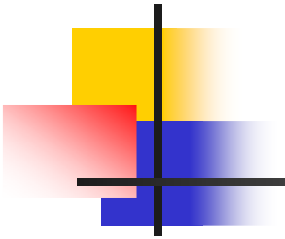
IJCNN 2007 Presentation

Probability Density Function Estimation Using Orthogonal Forward Regression

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

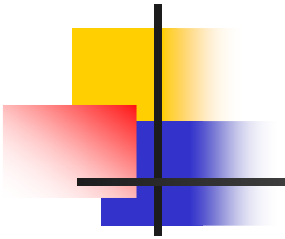
[†] **School of Electronics and Computer Science**
University of Southampton
Southampton SO17 1BJ, UK

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



Outline

- Motivations/overview for **sparse kernel density** estimation
- Proposed sparse kernel density estimator:
  - Convert **unsupervised** density learning into **constrained regression** by adopting **Parzen window estimate** as desired response
  - **Orthogonal forward regression** based on leave-one-out test mean square error and regularisation to determine structure
  - **Multiplicative nonnegative quadratic programming** to calculate kernel weights
- Empirical investigation and performance comparison



Motivations

- For most scientific and engineering problems, understand underlying **probability distributions** is the key to **fully** understand them

  Knowing probability density function $\Leftrightarrow$ fully understanding problem

- For **regression**, knowing PDF $\Rightarrow$ describe underlying process at any operating condition, i.e. **completely specify** data generating machanism

  Least squares approach, for example, is simply based on second-order momnets or statistics of the PDF

- For **classification**, knowing class conditional PDFs $\Rightarrow$ produce optimal Bayes' classifier, i.e. achieves **minimum classification error rate**

  Most classification methods can only approximate this optimal classification solution

Motivations (continue)

- Specific engineering topic: **control**
  - Various optimal controlls are based on controlling certain **moments**
  - Researchers have realised potential of directly controlling **probability distributions** (Prof Wang of University of Manchester)
  - If one can **control** PDF, one can control any moments, i.e. implementing any "optimal" control
- One of my home topics: **communication receiver detector**
  - State-of-the-art is **minimum mean square error** design, but it is detection error probability or bit error rate that really matters
  - By focusing on PDF of detector's output, we arrive at **minimum bit error rate** optimal design

Problem Formulation

- **PDF estimation**: given a realisation sample $D_N = \{\mathbf{x}_k\}_{k=1}^N$, drawn from unknown density $p(\mathbf{x})$, provide an estimate $\hat{p}(\mathbf{x})$ of $p(\mathbf{x})$
- PDF estimation is **difficult**
  - Unlike **regression** or **classification**, this is **unsupervised** learning, no teacher to provide desired response $y_k = p(\mathbf{x}_k)$ for estimator
- Density estimation methods can be classified as
  - **Parametric** approach: assume specific known PDF form of $\hat{p}(\mathbf{x}; \boldsymbol{\gamma})$, and the problem becomes one of fitting unknown parameters $\boldsymbol{\gamma}$
  - **Non-parametric** approach: does not impose any assumption on specific PDF form $\Rightarrow$ approach we adopted

Kernel Density Estimation

- Generic **kernel density estimate** is formulated as

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

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

- Classic solution **Parzen window estimate**: minimise divergence criterion between $p(\mathbf{x})$ and $\hat{p}(\mathbf{x}; \boldsymbol{\beta}_N, \rho)$ on D_N, leads to $\beta_k = \frac{1}{N}$, $1 \leq k \leq N$
  - Place a "conditional" unimodal PDF $K_\rho(\mathbf{x}, \mathbf{x}_k)$ 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 computational cost of calculating a PDF value scales directly with sample size N

Existing State-of-the-Art

- From **full kernel set** to make as many kernel weights to (near) zero as possible based on relevant criteria, yielding a sparse representation

  [1] Support vector machine based kernel density estimator

  Convert kernels into cumulative distribution functions and use empirical distribution function calculated on D_N as desired response, some hyperparameters to tune

  [2] Reduced set kernel density estimator

  Minimise integrated squared error on D_N, require certain types of kernels

- **Orthogonal forward regression** to select subset of significant kernels based on appropriate criteria, yielding a sparse kernel density estimate

  [3] OFR minimising training mean square error

  [4] OFR minimising leave-one-out mean square error with regularisation

  Both [3] and [4] convert kernels into CDFs and use EDFs as desired response, only select kernels do not cause negative kernel weights and normalise kernel weight vector

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 $\mathbf{A}\boldsymbol{\beta}_N = \mathbf{g}_N$

Proposed Algorithm

- Use OFR algorithm based on leave-one-out mean square error and regularisation to **automatically** select N_s significant kernels $\mathbf{\Phi}_{N_s}$
- Associated kernel weight vector $\boldsymbol{\beta}_{N_s}$ is calculated using **multiplicative nonnegative 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**, data set of N samples was used to construct kernel density estimate, and separate test data set of $N_{\text{test}} = 10,000$ samples was used to calculate L_1 test error for resulting estimate

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

  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

- Density to be estimated 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 $N_{\text{run}} = 200$

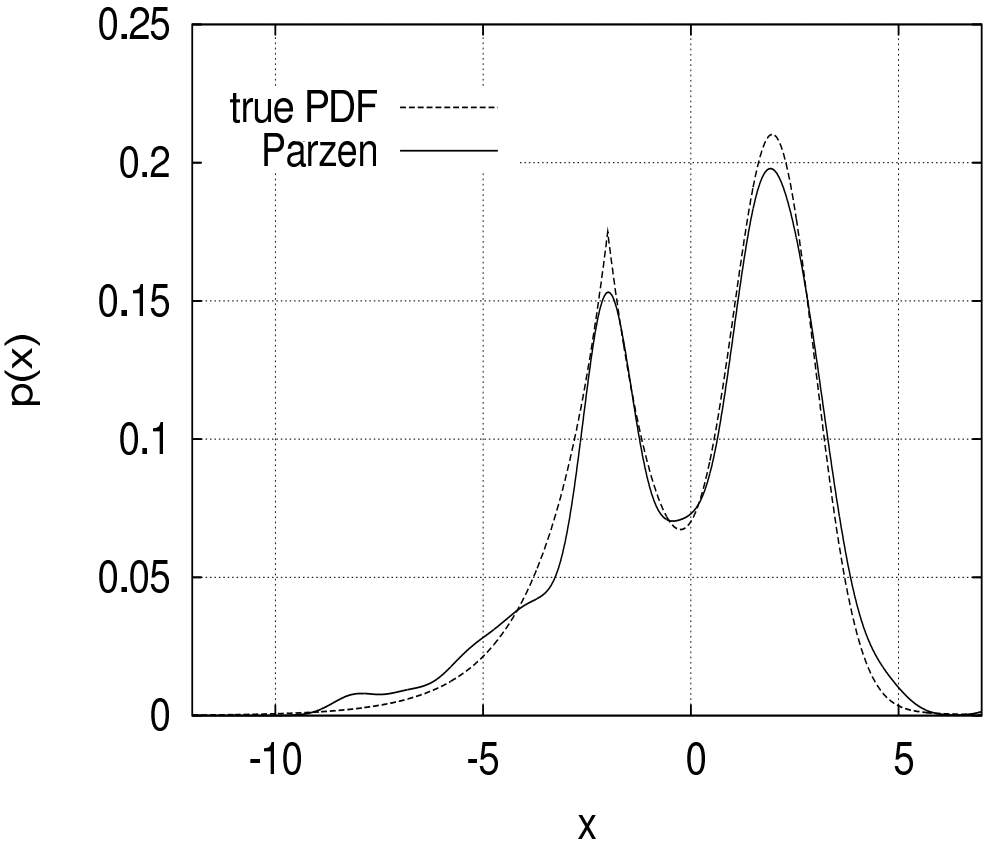
- Performance comparison in terms of L_1 test error and number of kernels required, quoted as mean $\pm$ standard deviation over 200 runs

method	L_1 test error	kernel number
PW estimator	$(1.9503 \pm 0.5881) \times 10^{-2}$	100 ± 0
SKD estimator [4]	$(2.1785 \pm 0.7468) \times 10^{-2}$	4.8 ± 0.9
proposed SKD estimator	$(1.9436 \pm 0.6208) \times 10^{-2}$	5.1 ± 1.3

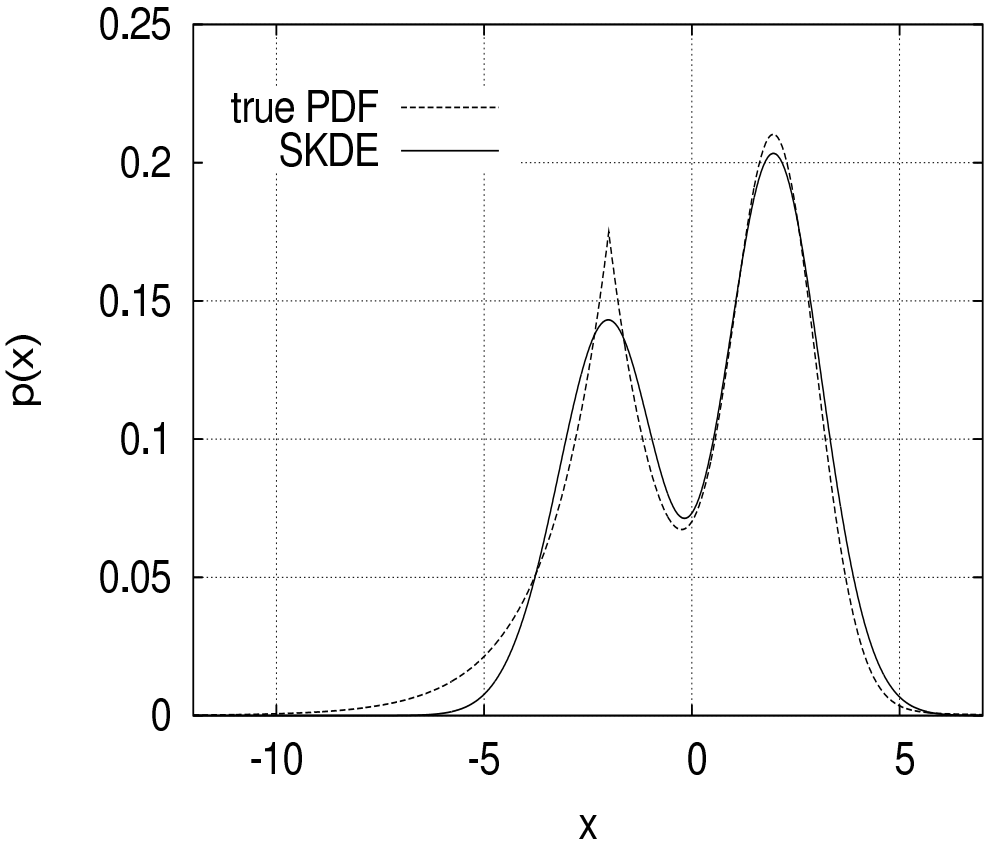
One-D Example (continue)

True density (dashed), (a) a PW estimate (solid) and (b) a proposed SKD estimate (solid)

(a)



(b)



Two-Class Two-Dimension Example

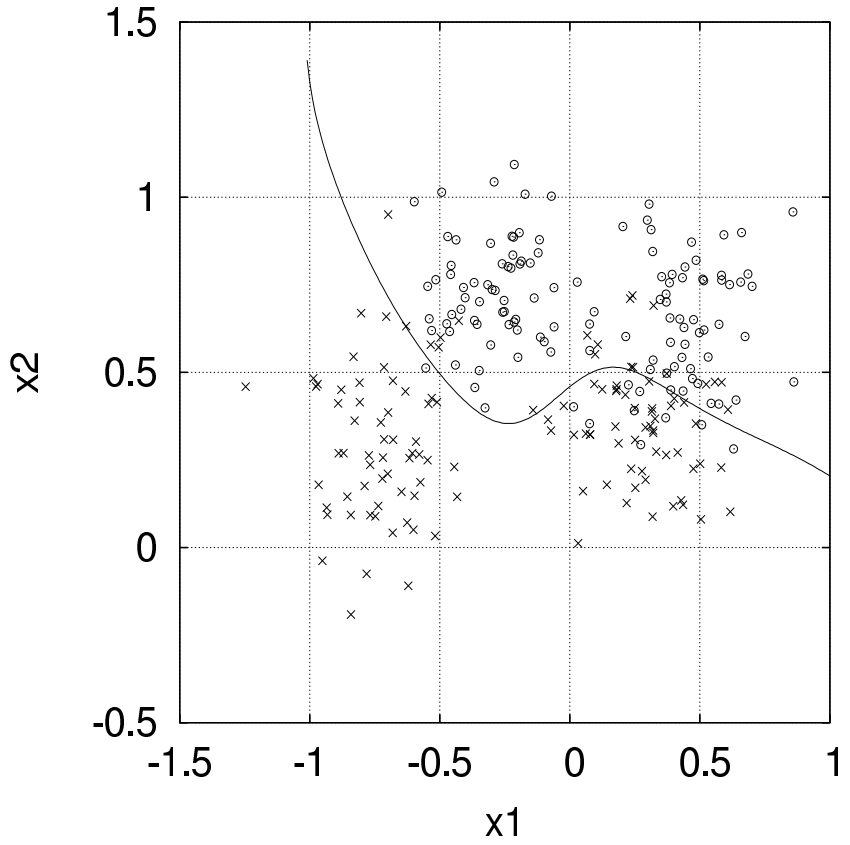
- `http://www.stats.ox.ac.uk/PRNN/`: two-class **classification** problem in two-dimensional feature space
- Training set contained 250 samples with 125 points for each class, test set had 1000 points with 500 samples for each class, and optimal Bayes test error rate based on true probability distribution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

method	$\hat{p}(\bullet\|C0)$	$\hat{p}(\bullet\|C1)$	test error rate
PW estimate	125 kernels	125 kernels	8.0%
SKD estimate [4]	5 kernels	4 kernels	8.3%
proposed SKD estimate	6 kernels	5 kernels	8.0%

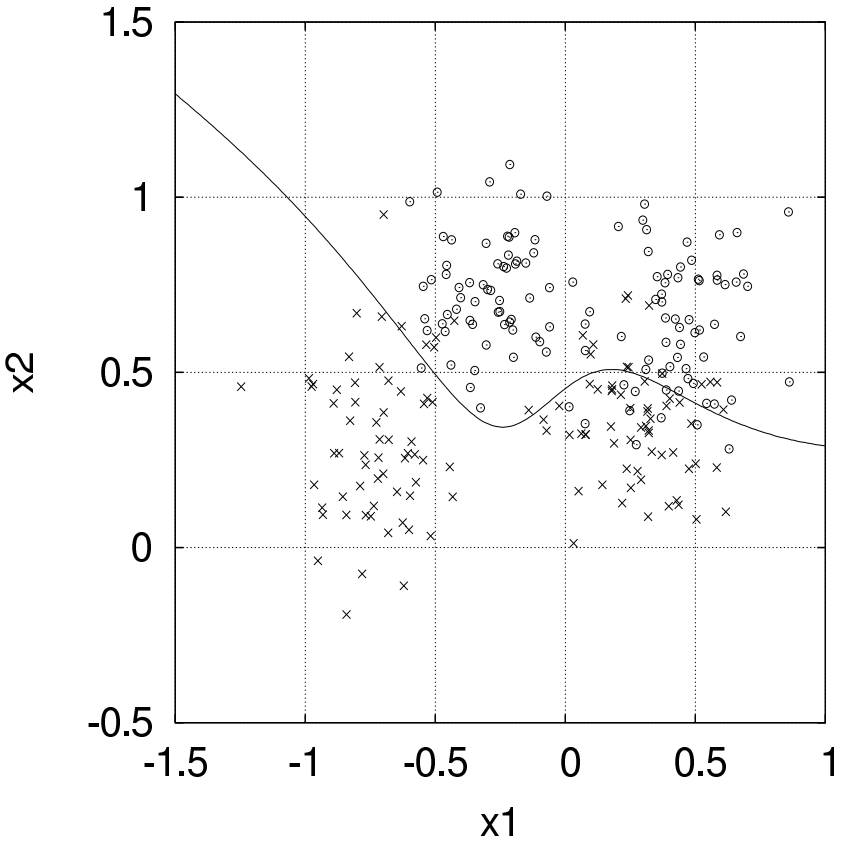
Two-class Two-D Example (continue)

Decision boundary of (a) PW 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

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

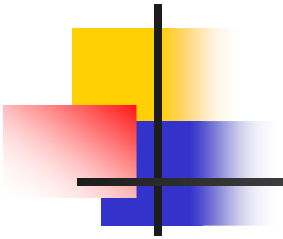
$$\boldsymbol{\mu}_1 = [1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0]^T,\ \mathbf{\Gamma}_1 = \mathrm{diag}\{1.0, 2.0, 1.0, 2.0, 1.0, 2.0\}$$

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

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

- $N = 600$, performance comparison in terms of L_1 test error and number of kernels required, quoted as mean $\pm$ standard deviation over $N_{\mathrm{run}} = 100$ runs

method	L_1 test error	kernel number
PW estimator	$(3.5195 \pm 0.1616) \times 10^{-5}$	600 ± 0
SKD estimator [4]	$(4.4781 \pm 1.2292) \times 10^{-5}$	14.9 ± 2.1
proposed SKD estimator	$(3.1134 \pm 0.5335) \times 10^{-5}$	9.4 ± 1.9



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
  - **Orthogonal forward regression** based on leave-one-out test mean square error and regularisation is employed 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

THANK YOU.

The support of the United Kingdom Royal Academy of Engineering is gratefully acknowledged