Kernel Density Estimation

Rohan Shiloh Shah

In Classification and Regression, the primary goal is the estimation of a prediction function. The likelihood or conditional density is one such function; for regression $p(\vec{y}|\vec{x}) = p(\vec{y}, \vec{x}) / \int p(\vec{y}, \vec{x}) d\vec{y}$ and similarly for classification $p(c|\vec{x}) = p(c, \vec{x}) / \sum_c p(c, \vec{x})$ where c is a class label from the set of labels $\mathfrak{C}$. These are supervised learning tasks since each training example is paired with a corresponding label or annotation; for regression $\vec{y} \in \mathbb{R}^n$ and for classification $c \in \mathfrak{C}$.

Given an unannotated training data set, we seek to build a model, specifically an unconditional probability density function, that delineates the essential information contained in the observation space $\mathcal{X}$. This is *un-supervised learning* since it is performed in the abscence of annotations (and hence without any cost or loss function) through direct interaction with 'new experiences'.

One common approach is to assume the density has a fixed parametric form and then to estimate the parameters (using a maximum likelihood approach) associated with this form; for example using a mixture model we can decompose the unknown density as follows:

$$\hat{f}_{(\mu,\Sigma)}(\vec{x}) = \sum_{i=1}^{m} \zeta_i \; P_{(\mu_i,\Sigma_i)}(\vec{x}), \; \zeta_i \geq 0, \; \sum \zeta_i = 1 \tag{1}$$

where the mixing coefficients ζ_i quantify the contribution of the i^{th} model in the mixture to the generation of the estimate $\hat{f}_{(\mu,\Sigma)}$. However, in many instances *parametric* mixture approximations do not converge in probability to the true density. The following sections present a regularized, non-parametric estimate that is a mixture of convolved kernel functions and is asymptotically both unbiased and consistent.

1 Non-Parametric Density Estimation

Provided with n discrete observations of a random variable

$$\mathcal{S} = \{\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_n\} \subseteq \mathcal{X}$$

all of which are identically and independently distributed (iid) according to some *unknown* probability distribution $F(\vec{x})$, we seek an estimate $\hat{f}(\vec{x})$ of the true *probability density function* $f(\vec{x})$. The search for $\hat{f}(\vec{x})$ is usually performed in a restricted functional space $\mathcal{H}_f$; which is infact a Reproducing Kernel Hilbert Space (see Section **??**). The functional space is further restricted to only those functions that are non-negative and integrate to one. The probability density function (PDF) is simply the derivative of the cumulative distribution function (CDF):

$$f(\vec{x}) = \frac{\partial F(\vec{x})}{\partial \vec{x}} \iff F(\vec{x}) = \int_{-\infty}^{\vec{x}} f(\vec{\psi}) d\vec{\psi} \tag{2}$$

We can rewrite 2 as a linear mapping [WGS$^+$99]:

$$\int_{-\infty}^{+\infty} \mathbb{I}_{\vec{\psi}<\vec{x}} f(\vec{\psi}) d\vec{\psi} = F(\vec{x}) \iff \mathcal{A} f(\vec{x}) = F(\vec{x}) \tag{3}$$

where both integrals in 2 and 3 are vector integrations and $\mathcal{A}$ is an injective mapping from $\mathcal{H}_f$ to the Hilbert Space where F is defined; $\mathcal{H}_F$. Neither f or F are known (whereas the operator $\mathcal{A}$ and its inverse are well defined) so we begin by estimating F using samples $\mathcal{S}$ generated by the random process and then proceed to deriving $\hat{f}$ from our estimate $\hat{F}$ using an approximation of the inverse of the linear transformation $\mathcal{A}$.

The *empirical distribution* at $\vec{x}$ can be estimated from the data by taking the ratio of the number of samples that are less than or equal to $\vec{x}$ to the total number of samples:

$$\hat{F}(\vec{x}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{(-\infty,\, \vec{x}_i]}(\vec{x}) \tag{4}$$

and is an unbiased maximum likelihood estimate that is piece-wise constant. For the density to exist, the estimated distribution $\hat{F}$ must be differentiable and hence continuous and so to smooth out the estimate $\hat{F}$; a non-linear regression (Figure 1) is used to approximate the distribution in the regions where training samples are unavailable; specifically the regression is performed on the set of pairs

$$\left\{ \vec{x}_i, \hat{F}(\vec{x}_i); \quad i = 1, \cdots, n \right\}$$

and is parametrized by the vector $\vec{\rho}$ which leads to our new estimate for the distribution: $\hat{F}_{\vec{\rho}}(\vec{x})$. Support Vector Regression techniques may also be used to derive accurate regressions since regularization (see Section **??**) is then possible.

Now to estimate the density function at $\vec{x}$ we need to estimate the derivative of $\hat{F}_{\vec{\rho}}(\vec{x})$ which can roughly be done by taking the difference between two evaluations of the distribution function at fixed lengths $+h$ and $-h$ from $\vec{x}$:

$$\begin{aligned}
\hat{f}(\vec{x}) &= \mathcal{A}^{-1} \hat{F}_{\vec{\rho}}(\vec{x}) \\
&= \frac{1}{2h} \int_{\vec{x}-h}^{\vec{x}+h} d\hat{F}_{\vec{\rho}}(\psi) \\
&= \frac{1}{2h} \left(\hat{F}_{\vec{\rho}}(\vec{x}+h) - \hat{F}_{\vec{\rho}}(\vec{x}-h) \right) \\
&= \frac{1}{2h} \left(\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\vec{x}_i \le \vec{x}+h} - \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\vec{x}_i \le \vec{x}-h} \right) \\
&= \frac{1}{2h} \left(\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\vec{x}_i \le \vec{x}+h} - \mathbb{I}_{\vec{x}_i \le \vec{x}-h} \right) \\
&= \frac{1}{2h} \left(\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\vec{x}-h \,\le\, \vec{x}_i \,\le\, \vec{x}+h} \right) \\
&= \frac{1}{V} \times \frac{k(\vec{x})}{n}
\end{aligned} \tag{5}$$

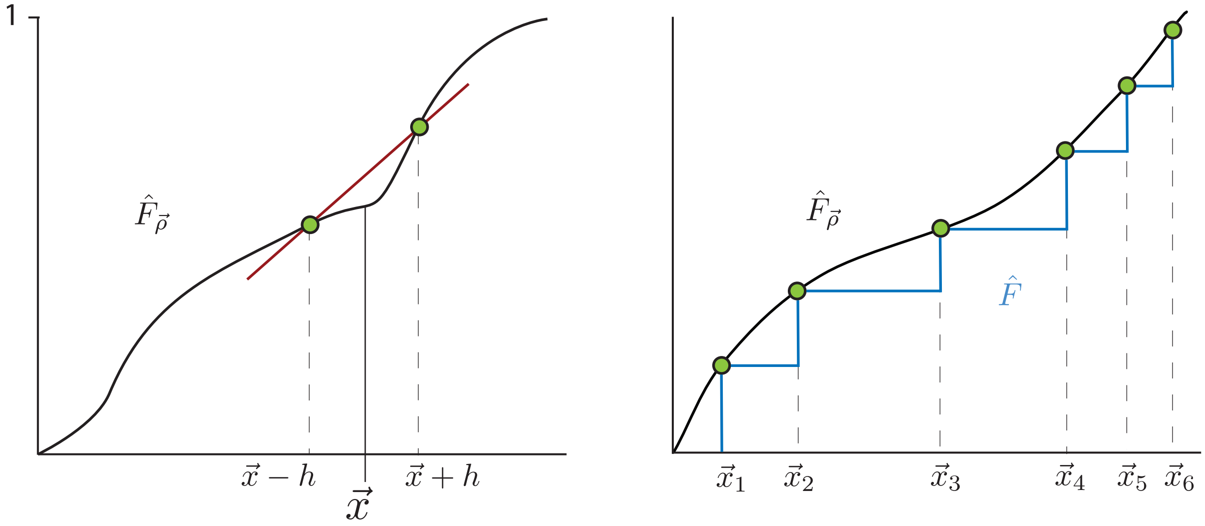

Figure 1: Estimating the Density and Distribution Functions: [left] A linear interpolation on $\hat{F}_{\vec{\rho}}$ between the end points of the region $\mathcal{R} = [\vec{x} - h, \vec{x} + h]$ gives us an estimate for the slope $\hat{f}(\vec{x})$. [right] A non-linear regression on evaluations of 4 at all training samples, in this case on $(\vec{x}_1, \hat{F}(\vec{x}_1)), (\vec{x}_2, \hat{F}(\vec{x}_2)), \cdots, (\vec{x}_6, \hat{F}(\vec{x}_6))$, yields a smooth estimate for the distribution function.

where $k(\vec{x}) = \sum_{i=1}^{n} \mathbb{I}_{\vec{x}-h \,\leq\, \vec{x}_i \,\leq\, \vec{x}+h}$ is the number of samples that fall in the region $\mathcal{R} = [\vec{x}-h, \vec{x}+h]$ and V is the volume of $\mathcal{R}$ which in this case is simply $(2h)^{-1}$. The shape of the region $\mathcal{R}$ and hence its volume V can be adjusted as more random samples become availible; it has been shown [DHS01] that as the regions $\mathcal{R}$ get smaller ($\lim_{n\to\infty} V = 0$) and the samples in the region increases ($\lim_{n\to\infty} k = \infty$), the estimated density $\hat{f}(\vec{x})$ will converge to $f(\vec{x})$ provided that $\lim_{n\to\infty} k/n = \infty$ (that is to say the proportion of samples falling within the region to those outside it is very small); in the limit it will be a smooth density function.

2 Kernel Density Estimation: Parzen Windows

In the previous section we decomposed the CDF into regions or windows $\mathcal{R}$ and estimated the PDF for each window separately. There are several ways to choose the placement (i.e. the center) of the windows; for example they can be *disjoint* and cover the entire domain as is the case with Frequency Histogram Estimation (Figure 2); the benefit of using such a method is that the structure of the windows are independant of the number of random samples (although the estimation procedure is not). Alternatively, the approach we will consider henceforth will associate a window with each random sample $\vec{x}_i$ so that we have a sequence of windows $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \cdots$ centered at $\vec{x}_1, \vec{x}_2, \vec{x}_3, \cdots$. Since the number of windows (and their volume) is now dependent on the sample size we will re-write the result (5) as

$$\hat{f}_n(\vec{x}) = \frac{1}{V_n} \times \frac{k_n(\vec{x})}{n} \tag{6}$$

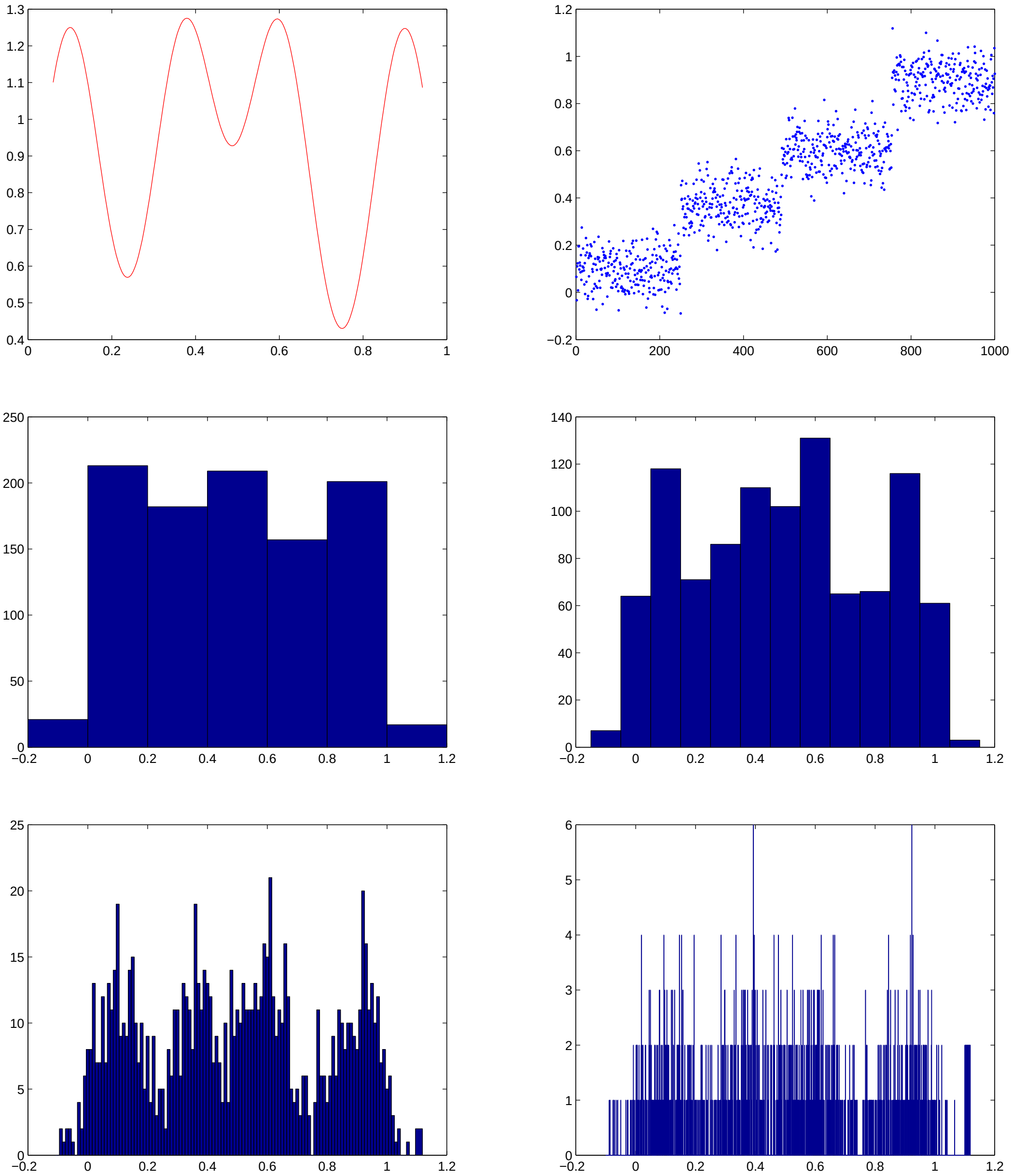

Figure 2: Parametric Frequency Histogram Estimation: The true unknown density (top left) can be estimated by taking random samples (top right, 1000 random samples) and placing them in bins of fixed length to generate a histogram. Histograms with bin-size $h = 0.2$, $h = 0.1$, $h = 0.01$ and $h = 0.001$ are shown; the bin-size or bandwidth (as well as the actual placement of the bins) is an important parameter in estimating the density function; in this case only a bin-size of $h = 0.01$ is able to capture the multi-modality of the true density. In the limit as the bandwidth goes to zero the histogram will converge to the true density provided the number of samples goes to infinity.

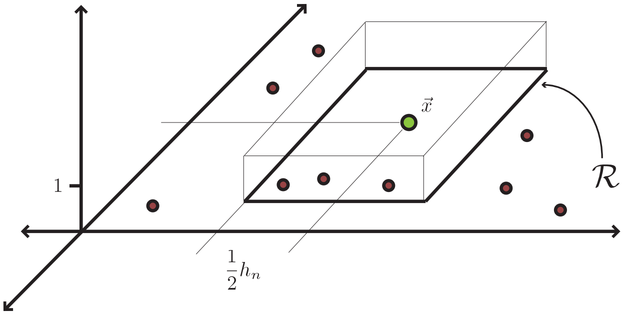

Figure 3: Let the hypercube window $\mathcal{R}_n$ have dimension $d = 2$; then $k_n(\vec{x})$ is simply a count of the random samples that fall in the square with sides of length h_n and centered at $\vec{x}$; in this case $k_n(\vec{x}) = 3$.

where V_n is the volume of the region $\mathcal{R}_n$ and k_n is the number of samples that fall inside it; as more samples are generated the regions can either be refined (as is the case with the Parzen method) or we can reverse this logic and gradually increase the size of the regions starting with an inconspicuously small $\mathcal{R}_1$ (which is the case with the nearest-neighbor estimation method).

Let us further generalize [DHS01] the definition of the region $\mathcal{R}_n$ so that we can estimate multivariate densities; assume the windows $\mathcal{R}_n$ are d-dimensional hypercubes with edges of length h_n; the volume is then simply $V_n = (h_n)^d$. Now that the window has changed we need to revise our definition of $k_n(\vec{x})$ which was previously defined using a simple indicator function $\mathbb{I}_{\vec{x}-h \leq \vec{x}_i \leq \vec{x}+h}$; generalising this from counting random samples within an interval to counting within a hypercube can be done by first using a *window function* ω of the form:

$$\omega(\vec{s}) = \begin{cases} 1, & |s_j| \leq 0.5 \ \ \forall j = 1, \cdots, d \\ 0, & otherwise \end{cases}$$

which defines the boundary for a unit-hypercube centered at $\vec{0}$ and then defining k_n as follows:

$$k_n(\vec{x}) = \sum_{i=1}^{n} \omega \left(\frac{\vec{x} - \vec{x}_i}{h_n} \right)$$

So $\omega \left(\frac{\vec{x}-\vec{x}_i}{h_n} \right) = 1$ if and only if $\frac{\vec{x}-\vec{x}_i}{h_n} \leq (\frac{1}{2}, \cdots, \frac{1}{2})^T$ or $\vec{x} - \vec{x}_i \leq (\frac{1}{2}h_n, \cdots, \frac{1}{2}h_n)^T$, in other words if $\vec{x}_i$ is less than half the length of an edge of the hypercube away from $\vec{x}$ in all dimensions $i = 1, \cdots, d$. So $\omega \left(\frac{\vec{x}-\vec{x}_i}{h_n} \right)$ defines a (d+1)-dimensional hypercube of volume $(h_n)^d$ (since the (d+1)th dimension has edges of length 1), centered at $\vec{x}$ which counts the number of random samples that fall within the d-dimensional hypercube window $\mathcal{R}$. Finally substituting this result into 6 gives:

$$\begin{aligned} \hat{f}_n(\vec{x}) &= \frac{1}{V_n} \times \frac{k_n}{n} \qquad (7) \\ &= \frac{1}{n} \frac{1}{(h_n)^d} \sum_{i=1}^{n} \omega \left(\frac{\vec{x} - \vec{x}_i}{h_n} \right) \end{aligned}$$

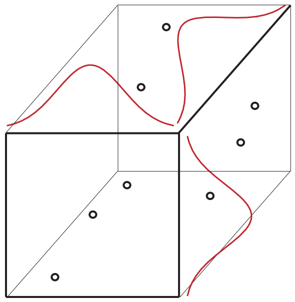

Figure 4: Product Kernel Window Functions: instead of counting the number of random samples within a hypercube centered at $\vec{x}$, we can associate a single-variate kernel function with each dimension and weight the count for each random sample by the product of its kernelized distances from $\vec{x}$ in each dimension. More generally a multi-variate kernel function may be used.

The resulting estimated density is 'jagged' since the window (basis) functions in the above linear combination are hypercubes (Figure 3) with abrupt edges; reducing the width h_n as more samples are generated will smooth out the estimate. Infact if $\lim_{n\to\infty} h_n = 0$ then the estimated density function is proved [Fuk72] to be asymptotically unbiased; $\lim_{n\to\infty} E(\hat{f}_n) = f$.

2.1 Kernel Basis Functions

Instead of simply counting the number of random samples that fall within a fixed volume surronding $\vec{x}$, we can weight the count [DHS01] for each random sample by its *kernelised distance* from $\vec{x}$. This can be achieved by replacing the unit hypercube window function $\omega(\vec{s})$ with a smooth, symmetric *kernel density function* $K(\vec{s})$ satisfying $V_n = \int_{-\infty}^{+\infty} K(\vec{s})\, d\vec{s} = 1$ and $K(\vec{s}) \geq 0$ and then rewriting 7 as:

$$\hat{f}_n(\vec{x}) = \frac{1}{n} \sum_{i=1}^{n} K_{h_n}(\vec{x} - \vec{x}_i) \tag{8}$$

where the bandwidth h_n is shifted into the definition of the kernel as the standard-deviation so that $K_{h_n}(\vec{s}) = K(\vec{s}/h_n)$ and the term involving the volume disappears since $V_n = 1$. The gaussian kernel is most often used;

$$K_\Sigma(\vec{x} - \vec{x}_i) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\vec{x} - \vec{x}_i)^T \Sigma^{-1} (\vec{x} - \vec{x}_i)\right) \tag{9}$$

where Σ is the covariance or *bandwidth matrix*. The key difference between the parametric density estimate 1 and non-parametric kernel density estimation 8 is that in the former the models that define the mixture have means or centers that are estimated from the data, while the latter makes use of kernel functions that are centered at the various samples in the training data.

The use of kernel basis functions has several advantages, the most significant of which is that the resulting estimate $\hat{f}(\vec{x})$ is also a smooth density function. It has been shown [Fuk72] that provided that $\lim_{n\to\infty} h_n = 0$ and $lim_{n\to\infty} nh_n = \infty$ the estimated kernel density estimate pointwise converges in probability to the true density - this is asymptotic consistency; uniform convergence in probability is also proved under the additional condition $\lim_{n\to\infty} n(h(n))^2 = \infty$.

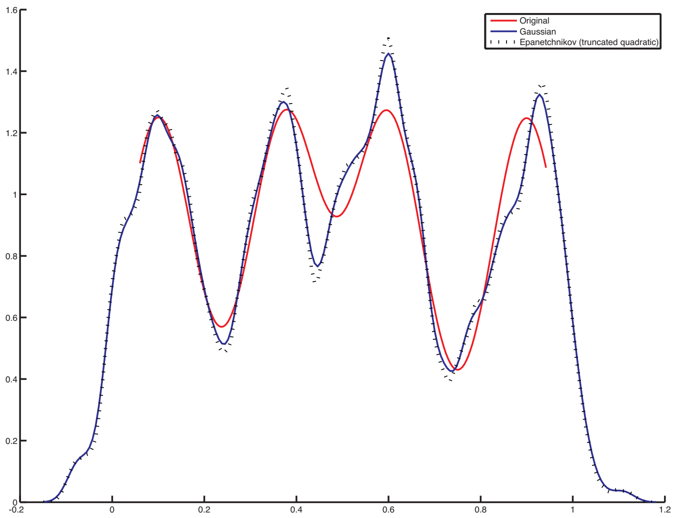

Figure 5: Comparing the Gaussian and Epanetchnikov Kernels [Ihl03]: a bandwidth of 0.0215 is used - the entropy for the Gaussian and Epanetchnikov Kernels are 0.0439 and 0.0430 respectively. Notice how even though the original or true density is defined only on the interval $[0, 1]$ so that random samples are also only generated on this interval, the resulting estimated density extends outside this interval; this can be good if there are regions of missing values so that an implicit non-linear interpolation estimates the density in these regions; it can be bad when the estimation extends into regions for which the density is meant to be undefined.

2.2 Regularization by Convolution

The estimate 8 can be re-written as a convolution of the kernel with the true density function;

$$\begin{aligned} f(\vec{s}) \star K(\vec{s}) &\equiv \int_{-\infty}^{+\infty} K(\vec{s} - \vec{x}) f(\vec{x}) d\vec{x} \\ &= E_{f(\vec{x})} \left(K(\vec{s} - \vec{x}) \right) \\ &\approxeq \frac{1}{n} \sum_{i=1}^{n} K(\vec{s} - \vec{x}_i) \\ &= \hat{f}(\vec{s}) \end{aligned} \tag{10}$$

and so in a sense, the kernel density estimate is approximately a deconvolution from the true density; in [DH73] we see that in the limit as the number of random samples approaches infinity, $\hat{f}(\vec{s})$ converges to $f(\vec{s}) \star K(\vec{s})$[1]. We can also write the density estimate as a smoothing[2] convolution of the *impulsive density function* (which assigns a probability mass of $\frac{1}{n}$ to each of the n random samples; $\hat{f}^i(\vec{x}) = 1/n$) with the kernel function:

$$\begin{aligned} \hat{f}^i(\vec{s}) \star K(\vec{s}) &= \sum_{i=1}^{n} \hat{f}^i(\vec{x}_i)\ K(\vec{s} - \vec{x}_i) \\ &= \sum_{i=1}^{n} \frac{1}{n}\ K(\vec{s} - \vec{x}_i) \\ &= \hat{f}(\vec{s}) \end{aligned} \tag{11}$$

Essentially the convolution is a regularization of the estimate through the addition of 'smoothing' noise in the regions where the impulsive density is undefined.

It is interesting to note that the Parzen Method has been shown [ZPR05] to be equivalent to a Regularized Least Squares Method (or Tikhonov Regularization) where the regularizing functional is taken to be the norm of the resulting estimated density. One benefit [MZ00] of 'regularizing by convolving' is that there are no explicit regularization parameters that need to be estimated and then re-trained.

2.3 Bandwidth Matrix Selection

Changes in the bandwidth h_n of the Kernel function, can severely effect the resulting estimate; choosing a large bandwidth will produce biased estimates that hide localised features whereas smaller bandwidths will increase the estimates variability by introducing sharp modulations. The simplest choice of bandwidth is as some function of n, for example $k_n = 1/n$, so that as the number of samples increases the kernel gets smaller. A more complex method minimizes the integrated mean squared error between $\hat{f}_n$ and f_n with respect to k_n. In Figure 6, three bandwidth selection methods are compared.

[1] ok, but don't we want $\hat{f}$ to converge to f?

[2] We refer to smooth and smoothing in two contexts; *smooth functions* have continuous derivatives and *smoothing operations* remove localized features

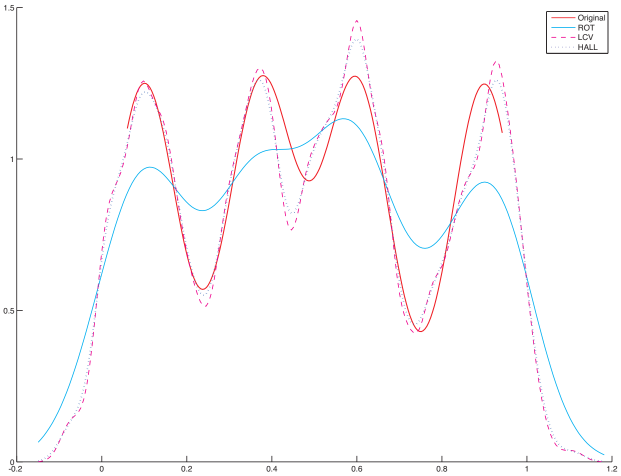

Figure 6: Kernel Bandwidth Selection [Ihl03]: Three bandwitdh selection methods are compared: the entropy and bandwidth of each are: ROT (0.1037, 0.0761), LCV (0.0439, 0.0215), HALL(0.0491, 0.0268). Notice the positive correlation between the entropy and the bandwidth; intuitively as the bandwidth increases the kernel density estimate gets smoother and closer to a uniform distribution which has the maximal entropy.

2.4 Computation and Sparsity

To form an estimation of the kernel density using Parzen Windows, 8 (or 7) must be evaluated for *all* n random samples. The sum in 8 (or 7) is another n evaluations of the window function, which in itself is a function of d the number of dimensions. In subsequent sections on support vector density estimation, an equally robust estimate involving considerably less computation will be considered.

References

[DH73] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[DHS01] Richard O. Duda, Peter E. (Peter Elliot) Hart, and David G. Stork. *Pattern classification.* Wiley, second edition, 2001.

[Fuk72] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 1972.

[Ihl03] Alexander Ihler. Kernel density estimation toolbox for matlab, 2003.

[MZ00] C. Molina and J. Zerubia. Regularisation by convolution in probability density estimation is equivalent to jittering. In *Proc. IEEE International Workshop on Neural Networks for Signal Processing (NNSP)*, Sydney, Australie, December 2000.

[WGS+99] J. Weston, A. Gammerman, M. O. Stitson, V. Vapnik, V. Vovk, and C. Watkins. Support vector density estimation, July 12 1999.

[ZPR05] Peng Zhang, Jing Peng, and Norbert Riedel. Finite sample error bound for parzen windows. In Manuela M. Veloso and Subbarao Kambhampati, editors, *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pages 925–931. AAAI Press; AAAI Press / The MIT Press, 2005.