KERNEL PROBABILITY DENSITY ESTIMATION METHODS [*]

S. Towers
State University of New York at Stony Brook

Abstract
Kernel Probability Density Estimation techniques are fast growing in popularity in the particle physics community. This note gives an overview of these techniques, and compares their signal/background discrimination performance to that of an artificial neural network.

1 Introduction

Over the past ten years, the particle physics community has become conversant with a number of quite sophisticated multivariate techniques that exploit higher-order correlations between variables to achieve optimum separation between signal and background.

Kernel Probability Density Estimation (PDE) methods have recently become part of this arsenal (see, for instance, references [1] and [2]), and are based on the premise that continuous, differentiable functions can be exactly modelled by the infinite sum of some other, appropriately chosen, 'kernel' function [3]. Fourier series are a familiar example of this concept; all periodic functions can be expressed as infinite sums of sine and cosine terms.

We will restrict this overview to non-parametric PDE methods that use a Gaussian kernel. The choice of a Gaussian kernel is a natural one for particle physics applications, since nearly all variables we analyse have been Gaussian smeared by detector resolution, or other effects. 'Non-parametric' simply means that no assumptions are made about the form of the probability density functions (PDF's) from which the samples are drawn.

A typical application of Gaussian kernel PDE's begins with a sample of N Monte Carlo events generated in a k-dimensional parameter space. The Monte Carlo events are distributed according to some (unknown) PDF, $F(\vec{x})$. A Gaussian kernel PDE method estimates the value of the PDF at a point $\vec{x}$ by the sum of Gaussians centred at the Monte Carlo generated points, $\{\vec{y}_i, i=1,N\}$:

$$f(\vec{x}) = \frac{1}{N|V|^{1/2}(2\pi h)^{k/2}} \sum_i^N \exp\left[-\frac{\vec{d}_i^T V^{-1} \vec{d}_i}{2h^2}\right], \tag{1}$$

where $\vec{d}_i = (\vec{x} - \vec{y}_i)$, V is a covariance matrix, and h is an additional scaling factor.

The optimal forms of V and h are a matter of debate. We will discuss two options here:

The static-kernel PDE Method determines V from the covariance matrix of the overall sample. The scale factor h is set to $N^{-1/(k+4)}$[1]. Because the parameters of the resulting Gaussian kernel are the same for all points, this is known as a static kernel method.

The Gaussian Expansion Method (GEM), developed by the author, determines V from the covariance matrix of the N_0 points in the Monte Carlo sample that are spatially closest to $\vec{x}$ in the normalised parameter space (where all variables are normalised to lie between 0 and 1). The parameter h is set to one.

In general, the arbitrary parameter N_0 must be large enough that the elements of V are approximately Gaussian distributed about the true value, but small enough that the local

[*]A colour version of this paper is available at `http://www-d0.fnal.gov/˜smjt/durham/pde.ps`

structure of the original PDF close to $\vec{x}$ is imitated by $f(\vec{x})$. The author has found that $N_0 > 25$ is a reasonable choice, and that the quality of the simulation of $F(\vec{x})$ does not strongly depend on the exact value of N_0.

Since the covariance matrix V depends on the density of points in the parameter space, the parameters of the Gaussian kernel will change for different $\vec{x}$. Thus this technique is known as a dynamic, or adaptive, kernel method.

The statistical variance of the GEM PDF estimate is

$$(\Delta f)^2 = \sum_{ij} \frac{\partial f}{\partial y_i} \frac{\partial f}{\partial y_j} V_{ij},$$

to yield:

$$(\Delta f(\vec{x}))^2 = \frac{4}{N^2 |V| (4\pi)^k} \sum_{i}^{N} (\vec{d}_i^T V^{-1} \vec{d}_i) \exp\left[-\vec{d}_i^T V^{-1} \vec{d}_i \right]. \tag{2}$$

Note that there are only one or two free parameters of both methods, and they are rather intuitive in nature. This is in sharp contrast to a typical neural network, which usually has a number of tuning parameters, the meaning of which is usually not transparent to the average user.

Both static-kernel PDE and GEM have relative advantages and disadvantages, which we will explore in the next section.

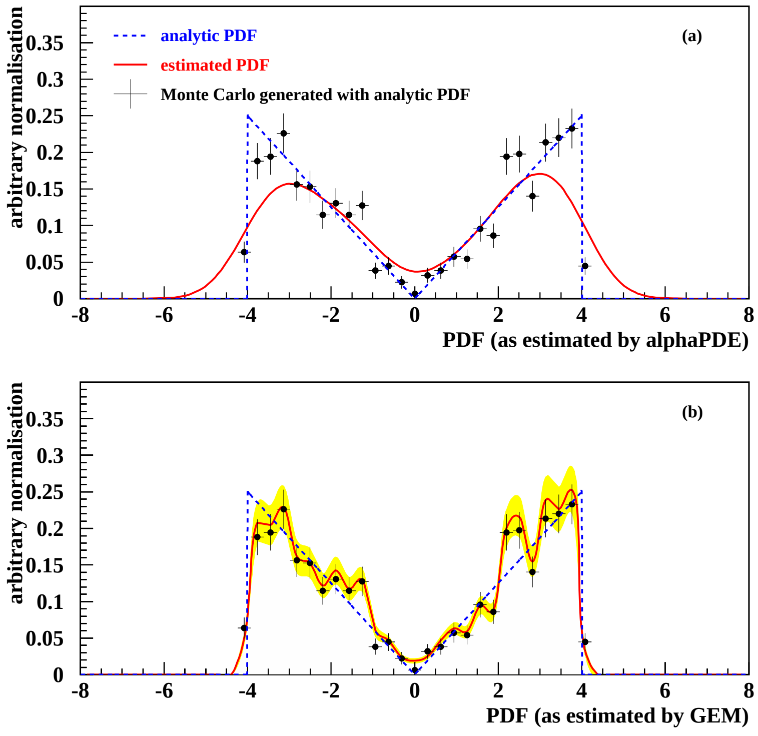

Fig. 1: A comparision of the performance of the static-kernel PDE method to that of GEM. Both figures display an analytic PDF (dashed line), which is used to generate 1000 events randomly sampled from the distribution (points). (a) and (b) show the static-kernel PDE and GEM estimates of the PDF, respectively (solid lines), based on the randomly generated sample. The estimate from the static-kernel PDE method has a smaller variance than that of the GEM method, but is more biased. The shaded region in (b) indicates the statistical variance of the GEM estimate, as provided by Equation 2. Note that neither method uses binned data to estimate the PDF. The data are binned here for display purposes only.

2 Comparison of static-kernel PDE and GEM

A contrived example best shows the primary differences between the static-kernel PDE method and GEM. Figure 1 displays an analytic PDF (indicated by the dashed line), that both the static-kernel PDE method and GEM estimate using the same sample of 1000 events randomly sampled from the PDF (the points). The static-kernel PDE method clearly gives a much smoother estimate than that of GEM. However, the static-kernel PDE estimate is biased, whereas the GEM estimate is nearly unbiased. The old adage "you can't get something for nothing" applies well here; the static-kernel PDE estimate of any PDF will always be biased (although in the limit of infinite Monte Carlo statistics the bias disappears), but will always have a smaller statistical variance than the GEM estimate. The bias is most prevalent in cases where the true PDF is non-differentiable, or has valleys. The variance of the GEM estimate, indicated by the shaded region in (b), is comparable to the variance of the binned Monte Carlo generated events. In the limit of infinite statistics, the variance goes to zero, and both GEM and the static-kernel PDE method then perform equally.

While we know *a priori* the form of the true PDF in this case, in general it is unknown. Thus, in a real analysis situation, when Monte Carlo samples of limited size are used for training, it may be difficult to assess the bias inherent in the static-kernel PDE method. However, the speed of application of the static-kernel PDE method relative to that of GEM may make its use desirable if very large samples of Monte Carlo are involved (the GEM method is slower than static-kernel PDE, because the local covariance matrix must be calculated for each point in the Monte Carlo samples).

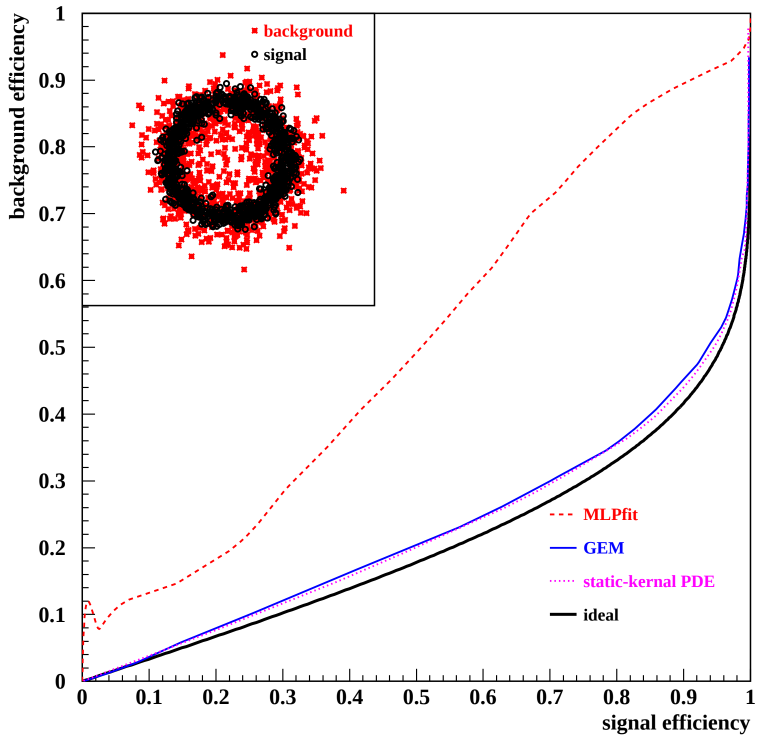

Fig. 2: A comparision of the performance of PDE methods to that of an artificial neural network for the 'signal' and 'background' distributions displayed in the vignette. The training of the neural network fails in this case because the correlations in the parameter space are highly non-linear, and the number of events used in the training samples is relatively small (500 events each for signal and background). Based on the same small training samples, however, the performance of GEM and the static-kernel PDE method are similar, and close to the ideal (shown in black).

3 Comparison of PDE performance to that of a Neural Network

The author has found, for the most part, that nearly all signal/background discrimination problems confronted in actual analysis situations are handled equally well by both artificial neural networks and PDE's. One advantage of using PDE's is that it easily provides an intuitive visual means of determining what the PDF looks like in the multi-dimensional parameter space.

There are, in addition, a few types of discrimination problems that are better suited to the PDE approach, rather than that of neural networks. Take, for instance, the contrived example presented in the vignette of Figure 2. The 'signal' is a distributed according to a 2D Gaussian smeared annular ring PDF, while the 'background' is similarly distributed except the width of the smearing is much broader. Because the correlations in the parameter space are highly non-linear in this situation, it can be quite difficult to successfully train a neural network to discriminate between signal and background, especially if the statistics of the signal and background training samples are sparse. The background efficiency versus signal efficiency curves are shown for static-kernel PDE, GEM, and a neural network, MLPfit[4], all based on training samples of 500 events each of signal and background. Both static-kernel PDE and GEM attain discrimination performance that is close to the ideal, but the training of the neural network fails in this case.

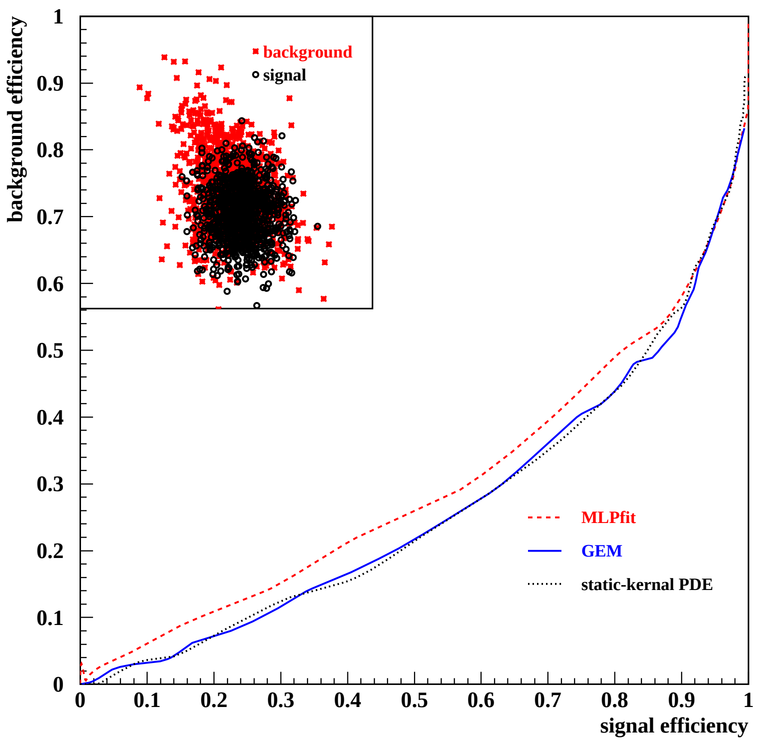

Fig. 3: A comparision of the performance of PDE methods to that of an artificial neural network for the 'signal' and 'background' distributions displayed in the vignette. The correlations in the parameter space are somewhat non-linear for the background, and the size of the training samples is again relatively small, but the discrimination performance of the neural network nonetheless approaches that of the PDE methods. In most real analysis situations, the discrimination performance of artificial neural networks and PDE methods is comparable.

An example of a case where both PDE's and neural networks perform comparably well is given in Figure 3. In this example the 'signal' and 'background' are slightly separated in both dimensions of the parameter space. The two dimensions are uncorrelated for the signal, while the background is slightly kidney shaped (which is perhaps not obvious in the vignette displayed in the figure). The background efficiency versus signal efficiency curves are shown for static-kernel PDE, GEM, and MLPfit. Despite the sparse training statistics, the discrimination performace of the neural network is comparable to that of the PDE methods.

4 Summary

Gaussian kernel PDE methods are quite easy to implement, and yield convenient graphical interpretations of the PDF in the multi-dimensional parameter space. The methods are easy to understand, and the few parameters needed to tune the methods are quite intuitive in nature.

The discrimination performance of the methods is comparable to that of artificial neural networks, and thus they offer an interesting alternative to the use of neural networks in a multivariate analysis.

References

[1] B. Knuteson *et al.*, (2001) physics/0108002.

[2] DØ Collaboration, V.M. Abazov *et al.*, Phys. Rev. Lett. **87** (2001) 231801.

[3] T. Hastie *et al.*, *The Elements of Statistical Learning*, Springer Verlag (2001).

[4] MLPfit, J. Schwindling, `http://schwind.home.cern.ch/schwind/MLPfit.html`