Entropy **2013**, *15*, 4540-4552; doi:10.3390/e15104540

OPEN ACCESS

entropy

ISSN 1099-4300

www.mdpi.com/journal/entropy

Article

A Kernel-Based Calculation of Information on a Metric Space

R. Joshua Tobin [1] **and Conor J. Houghton** [2,*]

[1] School of Mathematics, Trinity College Dublin, Dublin 2, Ireland; E-Mail: tobinrj@tcd.ie

[2] Department of Computer Science, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol BS8 1UB, UK

* Author to whom correspondence should be addressed; E-Mail: conor.houghton@bristol.ac.uk; Tel.: +44-117-954-5140.

Received: 24 July 2013; in revised form: 14 October 2013 / Accepted: 14 October 2013 / Published: 22 October 2013

Abstract: Kernel density estimation is a technique for approximating probability distributions. Here, it is applied to the calculation of mutual information on a metric space. This is motivated by the problem in neuroscience of calculating the mutual information between stimuli and spiking responses; the space of these responses is a metric space. It is shown that kernel density estimation on a metric space resembles the k-nearest-neighbor approach. This approach is applied to a toy dataset designed to mimic electrophysiological data.

Keywords: mutual information; neuroscience; electrophysiology; metric spaces; kernel density estimation

---

1. Introduction

This paper is concerned with the calculation of mutual information for spike trains using the data that are available in a typical *in vivo* electrophysiology experiment in the sensory system. It uses a kernel-based estimation of probability distributions.

In particular, this paper is concerned with computing the mutual information, $I(R;S)$, between two random variables, R and S. The motivating neuroscience example is a typical sensory pathway electrophysiology experiment in which the corpus of sensory stimuli are presented over multiple trials, so there is a set of recorded responses for each of a number of stimuli. The stimuli are drawn from a discrete space, the corpus, but the responses are spike trains. The space of spike trains is peculiar;

locally, it is like a smooth manifold, with the spike times behaving like coordinates; but, globally, it is foliated into subspaces, each with a different number of spikes. The space of spike trains does, however, have a metric. As such, S takes values in a discrete set, $\mathcal{S}$, and models the stimulus, and R takes values in a metric space, $\mathcal{R}$, and models the response.

$\mathcal{R}$ is not a discrete space, and so, to calculate the mutual information between S and R, it is necessary to either discretize $\mathcal{R}$ or to use differential mutual information. In the application of information theory to electrophysiological data, it is common to take the former route and discretize the data. Here, the latter alternative is chosen, and the differential mutual information is estimated.

The mutual information between two random variables, R and S, is a measure of the average amount of information that is gained about S from knowing the value of R. With S, a discrete random variable taking values in $\mathcal{S}$ and R, a continuous random variable, the mutual information is:

$$I(R;S) = \sum_{s\in\mathcal{S}} \int_{\mathcal{R}} p(r,s) \log_2 \frac{p(r,s)}{p(r)p(s)} dr \tag{1}$$

where dr is the measure on $\mathcal{R}$: computing the differential mutual information between R and S requires integration over $\mathcal{R}$. Integration requires a measure, and when there are coordinates on a space, it is common to use the integration measure derived from these coordinates.

The space of spike trains has no system of coordinates, and so, there is no coordinate-based measure. This does not mean that the space has no measure. As a sample space, it has an intrinsic measure corresponding to the probability distribution; thus, there is a measure, just not one derived from coordinates. The probability of an event occurring in a region of sample space gives a volume for that region. In other words, the volume of a region, $\mathcal{D}$, can be identified with $P(\mathbf{x} \in \mathcal{D})$. This is the measure that will be used throughout this paper; it does not rely on coordinates, and so, can be applied to the case of interest here.

Of course, in practice, the probability density is not usually known on the space of spike trains, but $P(\mathbf{x} \in \mathcal{D})$ can be estimated from the set of experimental data. A Monte-Carlo-like approach is used: the volume of a region is estimated by counting the fraction of data points that lie within it:

$$\text{vol}(\mathcal{D}) = P(\mathbf{x} \in \mathcal{D}) \approx \frac{\text{number of data points in } \mathcal{D}}{\text{total number of points}} \tag{2}$$

This is exploited in this paper to estimate the volume of square kernels, making it possible to estimate conditional probabilities using kernel density estimation.

The classical approach to the problem of estimating $I(R;S)$ is to map the spike trains to binary words using temporal binning [1,2], giving a histogram approximation for $p(r,s)$. This approach is very successful, particularly when supplemented with a strategically chosen prior distribution for the underlying probability distribution of words [3,4]. This is sometimes called the plug-in method, and that term is adopted here. One advantage of the plug-in method is that the mutual information it calculates is correct in the limit: in the limit of an infinite amount of data and an infinitesimal bin size, it gives the differential mutual information.

Nonetheless, it is interesting to consider other approaches, and in this spirit, an alternate approach is presented here. This new method exploits the inherent metric structure of the space of spike trains,

it is very natural and gives an easily implemented algorithm, which is accurate on comparatively small datasets.

2. Methods

This section describes the proposed method for calculating mutual information. Roughly, the conditional probability is approximated using kernel density estimation and, by using the unconditioned probability distribution as a measure, integration is approximated by the Monte-Carlo method of summing over data points.

Since this is a kernel-based approach, a review of kernel density estimation is given in Section 2.1. This also serves to establish notation. The two key steps used to derive the kernel-based estimate are a change of measure and a Monte-Carlo estimate. The change of measure, described in Section 2.2, permits the estimation of probabilities by a simple Monte-Carlo method. The new measure also simplifies the calculation of $I(R;S)$, resulting in a formula involving a single conditional distribution. This conditional distribution is estimated using a Monte-Carlo estimate in Section 2.3.

2.1. Kernel Density Estimation

The non-parametric kernel density estimation (KDE) method [5–7] is an approach to estimating probability densities. In KDE, a probability density is estimated by filtering the data with a kernel. This kernel is normalized with an integral of one and is usually symmetric and localized. For an n-dimensional distribution with outcome vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ and a kernel, $k(\mathbf{x})$, the estimated distribution is usually written:

$$\tilde{p}(\mathbf{x}) = \frac{1}{m}\sum_i k(\mathbf{x} - \mathbf{x}_i) \tag{3}$$

where, because the argument is $\mathbf{x} - \mathbf{x}_i$, there is a copy of the kernel centered at each data point. In fact, this relies on the vector-space structure of n-dimensional space; in the application considered here, a more general notation is required, with $k(\mathbf{x};\mathbf{y})$ denoting the value at $\mathbf{x}$ of the kernel when it is centered on $\mathbf{y}$. In this situation, the estimate becomes:

$$\tilde{p}(\mathbf{x}) = \frac{1}{m}\sum_i k(\mathbf{x};\mathbf{x}_i) \tag{4}$$

The square kernel is a common choice. For a vector space, this is:

$$k(\mathbf{x};\mathbf{y}) = \begin{cases} \frac{1}{V} & \|\mathbf{x}-\mathbf{y}\| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where V is chosen, so that the kernel integrates to one. The kernel is usually scaled to give it a bandwidth:

$$k(\mathbf{x};\mathbf{y},h) = \begin{cases} \frac{1}{hV} & \|\mathbf{x}-\mathbf{y}\| < h \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

This bandwidth, h, specifies the amount of smoothing. The square kernel is the most straight-forward choice of kernel mathematically, and so, in the construction presented here, a square kernel is used.

In the case that will be of interest here, where $\mathbf{x}$ and $\mathbf{y}$ are not elements of a vector space, the condition $\|\mathbf{x}-\mathbf{y}\| < h$ must be replaced by $d(\mathbf{x},\mathbf{y}) < h$, where $d(\mathbf{x},\mathbf{y})$ is a metric measuring the distance between $\mathbf{x}$ and $\mathbf{y}$. Calculating the normalization factor, V, is more difficult, since this requires integration. This problem is discussed in the next subsection.

2.2. Change of Measure

Calculating the differential mutual information using KDE requires integration, both the integration required by the definition of the mutual information and the integration needed to normalize the kernel. As outlined above, these integrals are estimated using a Monte-Carlo approach; this relies on a change of measure, which is described in this section.

For definiteness, the notation used here is based on the intended application to spike trains. The number of stimuli is n_s, and each stimulus is presented for n_t trials. The total number of responses, n_r, is then $n_r = n_s n_t$. Points in the set of stimuli are called s and in the response space, r; the actual data points are indexed, r_i, and (r_i, s_i) is a response-stimulus pair. As above, the random variables for stimulus and response are S and R, whereas the set of stimuli and the space of responses are denoted by a calligraphic $\mathcal{S}$ and $\mathcal{R}$, respectively. It is intended that when the method is applied, the responses, $r \in \mathcal{R}$, will be spike trains.

The goal is to calculate the mutual information between the stimulus and the response. Using the Bayes theorem, this is:

$$I(R;S) = \sum_{s\in\mathcal{S}} \int_{\mathcal{R}} p(r,s) \log_2 \frac{p(r|s)}{p(r)} dr \tag{7}$$

Unlike the differential entropy, the differential mutual information is invariant under the choice of measure. Typically, differential information theory is applied to examples where there are coordinates, $(x_1, x_2, \ldots, x_n)$, on the response space and the measure is given by $dr = dx_1 dx_2 \ldots dx_n$. However, here, it is intended to use the measure provided by the probability distribution, $p(r)$. Thus, for a region, $\mathcal{D} \subset \mathcal{R}$, the change of measure is:

$$\mathrm{vol}(\mathcal{D}) = \int_{\mathcal{D}} p(r) dr = \int_{\mathcal{D}} d\beta \tag{8}$$

so:

$$d\beta = p(r) dr \tag{9}$$

The new probability density relative to the new measure, $p_\beta(r)$, is now one:

$$p_\beta(r) = \frac{p(r)}{d\beta/dr} = 1 \tag{10}$$

Furthermore, since $p(r|s)$ and $p(r)$ are both densities, $p(r|s)/p(r)$ is invariant under a change of measure and:

$$I(R;S) = \sum_{s} \int_{\mathcal{R}} p_\beta(r,s) \log_2 \frac{p_\beta(r|s)}{p_\beta(r)} d\beta = \sum_{s} \int_{\mathcal{R}} p_\beta(r,s) \log_2 p_\beta(r|s) d\beta \tag{11}$$

where, again, $p_\beta(r,s)$ and $p_\beta(r|s)$ are the values of the densities, $p(r,s)$ and $p(r|s)$, after the change of measure.

The expected value of any function, $f(R, S)$, of random variables, R and S, is:

$$\langle f \rangle = \sum_{s \in \mathcal{S}} \int_{\mathcal{R}} p_\beta(r, s) f(r, s) d\beta \tag{12}$$

and this can be estimated on a set of outcomes, $\{(r_i, s_i)\}$, as:

$$\langle f \rangle \approx \frac{1}{n_r} \sum_i f(r_i, s_i) \tag{13}$$

For the mutual information, this gives:

$$I(R; S) \approx \frac{1}{n_r} \sum_i \log_2 p_\beta(r_i | s_i) \tag{14}$$

Now, an estimate for $p_\beta(r_i|s_i)$ is needed; this is approximated using KDE.

2.3. A Monte-Carlo Estimate

One advantage to using $d\beta$ as the measure is that $p_\beta(r) = 1$, and this simplifies the expression for $I(R; S)$. However, the most significant advantage is that under this new measure, volumes can be estimated by simply counting data points. This is used to normalize the kernel. It is useful to define the support of a function: if $f(r)$ is a function, then the support of $f(r)$, supp$[f(r)]$, is the region of its domain where it has a non-zero value:

$$\text{supp}[f(r)] = \{r : f(r) \neq 0\} \tag{15}$$

Typically, the size of a square kernel is specified by the radius of the support. Here, however, it is specified by volume. In a vector space where the volume measure is derived from the coordinates, there is a simple formula relating radius and volume. That is not the case here, and specifying the size of a kernel by volume is not equivalent to specifying it by radius. Choosing the volume over the radius simplifies subsequent calculations and, also, has the advantage that the size of the kernel is related to the number of data points. This also means that the radius of the kernel varies across $\mathcal{R}$.

The term, bandwidth, will be used to describe the size of the kernel, even though here, the bandwidth is a volume, rather than a radius. Since $d\beta$ is a probability measure, all volumes are between zero and one. Let h be a bandwidth in this range. If $k(r'; r, h)$ is the value at r' of a square kernel with bandwidth h centered on r, the support will be denoted as $\mathcal{S}(r; h)$:

$$\mathcal{S}(r; h) = \text{supp}[k(r'; r, h)] \tag{16}$$

and the volume of the support of the kernel is vol$[\mathcal{S}(r; h)]$. The value of the integral is set at one:

$$\int_{\mathcal{S}(r;h)} k(r'; r, h) d\beta = 1 \tag{17}$$

and so, since the square kernel is being used, $k(r'; r, h)$ has a constant value of $1/\text{vol}[\mathcal{S}(r; h)]$ throughout $\mathcal{S}(r; h)$.

Thus, volumes are calculated using the measure, $d\beta$, based on the probability density. However, this density is unknown, and so, volumes need to be estimated. As described above, using $d\beta$, the volume of a region is estimated by the fraction of data points that lie within it. In other words, the change of measure leads to a Monte-Carlo approach to calculating the volume of any region. In the Monte-Carlo calculation, the volume of the support of a kernel is estimated as the fraction of data points that lie within it. A choice of convention has to be made between defining the kernel as containing $\lfloor hn_r \rfloor$ or $\lceil hn_r \rceil$ points, that is, on whether to round hn_r down or up. The former choice is used, so, the kernel around a point, r, is estimated as the region containing the nearest $n_h = \lfloor hn_r \rfloor$ points to r, including r itself. Thus, the kernel around a point, r_i, is defined as:

$$k(r; r_i, n_h) = \begin{cases} \frac{1}{n_h}, & r \text{ is one of the } n_h \text{ closest points to } r_i \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

and the support, $\mathcal{S}(r_i; n_h)$, has $r_j \in \mathcal{S}(r_i; n_h)$ if $k(r_j; r_i, n_h) = 1/n_h$, or, put another way, r_j is one of the n_h nearest data points. In practice, rather than rounding hn_r up or down, the kernel volume in a particular example can be specified using n_h rather than h.

Typically, kernels are balls: regions defined by a constant radius. As such, the kernel described here makes an implicit assumption about the isotropic distribution of the data points. However, in the normal application of KDE, special provision must be made near boundaries, where the distribution of data points is not isotropic [8]. Here, these cases are dealt with automatically.

Since $p_\beta(r_i|s_i) = n_s p_\beta(r_i, s_i)$, here, the conditional distribution, $p_\beta(r_i|s_i)$, is estimated by first estimating $p_\beta(r_i, s_i)$. As described above, a kernel has a fixed volume relative to the measure based on $p_\beta(r)$. Here, the kernel is being used to estimate $p_\beta(r_i, s_i)$:

$$\tilde{p}_\beta(r_i, s_i) = \frac{c(r_i, s_i; n_h)}{n_h} \tag{19}$$

where $c(r_i, s_i; n_h)$ is the number of data points evoked to stimulus s_i for which r_i is one of the n_h closest points:

$$c(r_i, s_i; n_h) = |\{(r_j, s_i) : r_j \in \mathcal{S}(r_i; n_h)\}| \tag{20}$$

This gives the estimated mutual information:

$$I(R; S) \approx I(R, S; n_h) = \frac{1}{n_r} \sum_i \log_2 \frac{n_s c(r_i, s_i; n_h)}{n_h} \tag{21}$$

Remarkably, although this is a KDE estimator, it resembles a k-, or, here, n_h-, nearest-neighbors estimator. Basing KDE on the data available for spike trains appears to lead naturally to nearest neighbor estimation.

The formula for $I(R, S; n_h)$ behaves well in the extreme cases. If the responses to each stimulus are close to each other, but distant from responses to all other stimuli, then $c(r_i, s_i; n_h) = n_h$ for all stimulus-response pairs (r_i, s_i). That is, for each data point, all nearby data points are from the same stimulus. This means that the estimate will be:

$$I(R, S; n_h) = \log_2 n_s \tag{22}$$

This is the correct value, because, in this case, the response completely determines the stimulus, and so, the mutual information is exactly the entropy of the stimulus. On the other hand, if the responses to each stimulus have the same distribution, then $c(r_i, s_i; n_h)/n_h \approx 1/n_s$, so the estimated mutual information will be close to zero. This is again the correct value, because in this case, the response is independent of the stimulus.

3. Results

As a test, this method has been applied to a toy dataset modelled on the behavior of real spike trains. It is important that the method is applied to toy data that resemble the data type, electrophysiological data, on which the method is intended to perform well. As such, the toy model is selected to mimic the behavior of sets of spike trains. The formula derived above acts on the matrix of inter-data-point distances, rather than the points themselves, and so, the dataset is designed to match the distance distribution observed in real spike trains [9]. The test dataset is also designed to present a stiff challenge to any algorithm for estimating information.

The toy data are produced by varying the components of one of a set of source vectors. More precisely, to produce a test dataset, a variance, σ^2, is chosen uniformly from $[0, 1]$, and n_s sources are chosen uniformly in a n_d-dimensional box centered at the origin with unit sides parallel to the Cartesian axes. Thus, the sources are all n_d-dimensional vectors. The data points are also n_d-dimensional vectors; they are generated by drawing each component from a normal distribution about the corresponding component of the source. Thus, data points with a source $\mathbf{s} = (s_1, s_2, \ldots, s_{n_d})$ are chosen as $\mathbf{r} = (r_1, r_2, \ldots, r_{n_d})$, where the r_i are all drawn from normal distributions with variance σ^2 centered at the corresponding s_i:

$$r_i \sim \mathcal{N}(s_i, \sigma^2) \tag{23}$$

n_t data points are chosen for each source, giving $n_r = n_s n_t$ data points in all.

Each test uses 200 different datasets; random pruning is used to ensure that the values of mutual information are evenly distributed over the whole range from zero to $\log_2 n_s$; otherwise, there tends to be an excess of datasets with a low value. The true mutual information is calculated using a Monte-Carlo estimate sampled over 10,000 points. The actual probability distributions are known: the probability of finding a point $\mathbf{r}$ generated by a source, $\mathbf{s}$, depends only on the distance $d = |\mathbf{r} - \mathbf{s}|$ and is given by the χ-distribution:

$$p(d) = \frac{2^{1-n_d/2}}{\Gamma(n_d/2)} \left(\frac{d}{\sigma}\right)^{n_d-1} e^{-d^2/2\sigma^2} \tag{24}$$

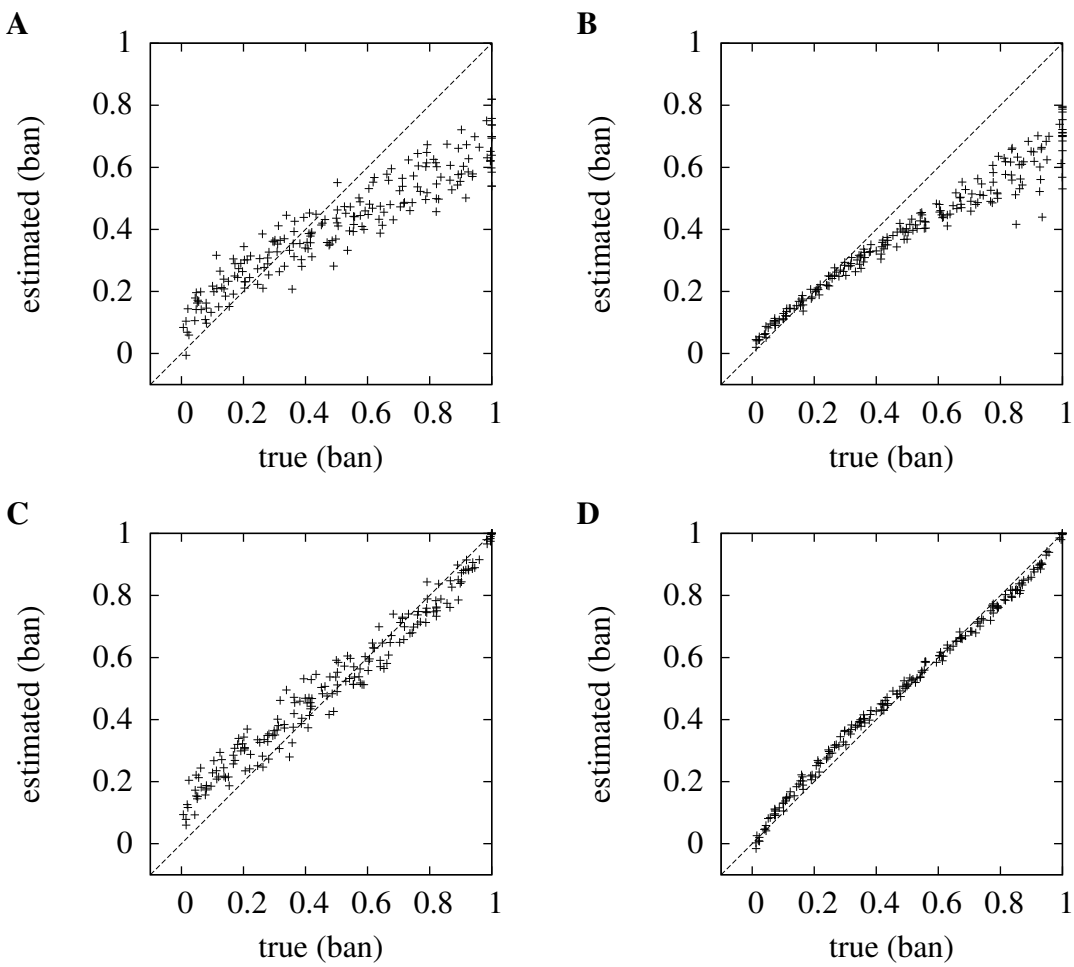
There is a bias in estimating the mutual information, in fact, bias is common to any approach to estimating mutual information [10]. The problem of reducing bias, or defining the mutual information, so that the amount of bias is low, is well studied and has produced a number of sophisticated approaches [4,10–14]. One of these, quadratic estimation, thanks to [11,13], is adapted to the current situation. Basically, it is assumed that for large numbers of data points, n_t, the estimated information, $\tilde{I}(R; S)$, is related to the true mutual information $I(R; S)$ by:

$$\tilde{I}(R; S) = I(R; S) + \frac{A}{n_t} + \frac{B}{n_t^2} + O(1/n_t^3) \tag{25}$$

This asymptotic expansion is well-motivated in the case of the plug-in approach to spike train information [10,11,15–17], and since the sources of bias are presumably similar, it is assumed the same expansion applies. In fact, this assumption is supported by plots of $I(R, S; n_h)$ against n_t. To extract $I(R; S)$, the estimate, $I(R, S; n_h)$, is calculated for λn_r with λ taking values from 0.1 to one in 0.1 increments. Least squares fit is used to estimate $I(R; S)$ from these ten values.

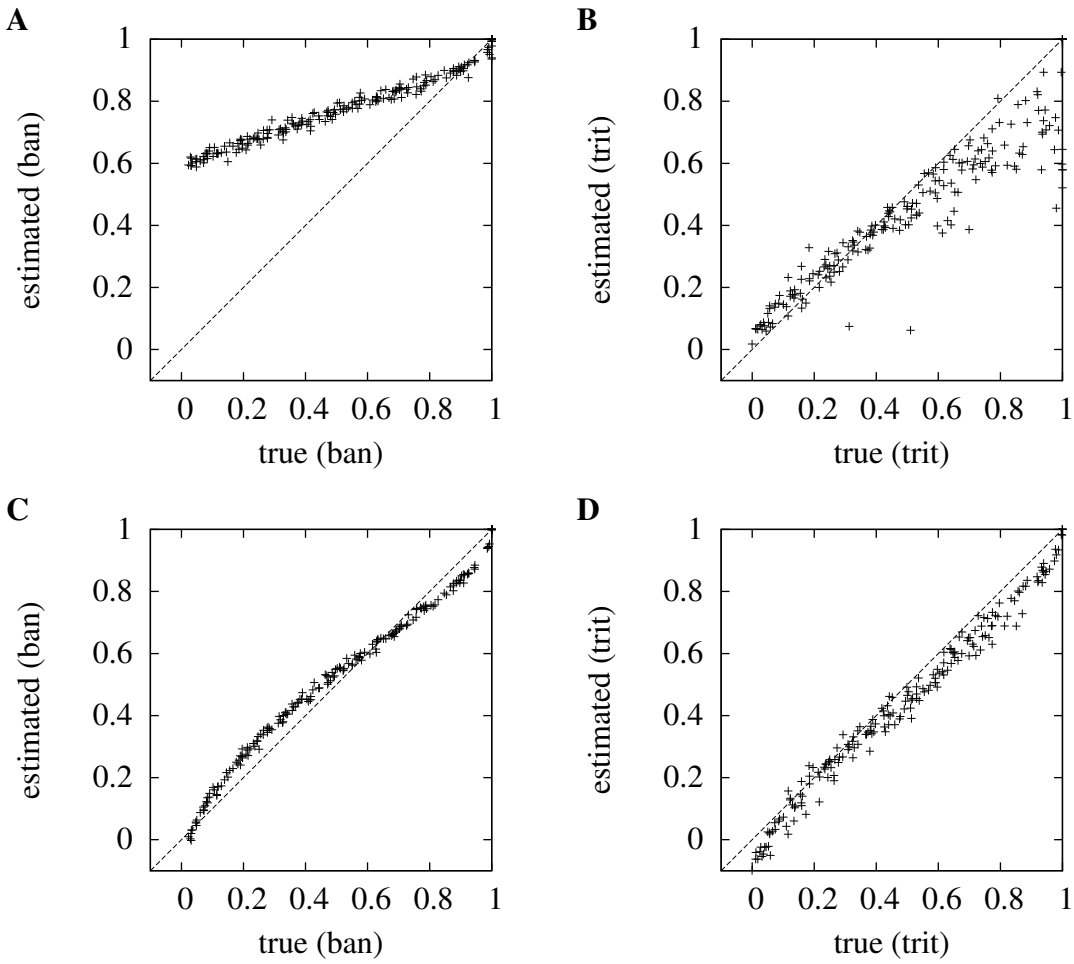
The new method works well on these toy data. It is compared to a histogram approach, where the n_d-dimensional space is discretized into bins and counting is used to estimate the probability of each bin. This is an analog of the plug-in method, and the same quadratic estimation technique is used to reduce bias.

Figure 1. Comparing kernel density estimation (KDE) to the histogram method for ten sources, $n_s = 10$, and three dimensions, $n_d = 3$. In each case, the true information is plotted against the estimated information; the line, $y = x$, which represents perfect estimation, is plotted for clarity. For convenience, the mutual information has been normalized, so in each case, the value plotted is the estimate of $I(R; S)/\log_2 n_s$, with a maximum value of one; in the cases plotted here, that means the information is measured in ban. (**A**) and (**B**) show the distribution for the histogram method for $n_t = 10$ and $n_t = 200$; (**C**) and (**D**) show the kernel method.



In Figure 1, the new method is compared to the histogram method when $n_s = 10$ and $n_d = 3$, and for both low and high numbers of trials, $n_t = 10$ and $n_t = 200$. For the histogram method, the optimum discretization width is used. This optimal width is large, $h = 5$ in each case; this roughly corresponds to a different bin for each octant of the three-dimensional space containing the data. In the new method, the bandwidth is not optimized on a case by case basis; instead, the kernel bandwidth, n_h, is chosen as being equal to the number of trials, n_t. It can be seen that the new method is better at estimating the information: for $n_t = 10$, it has an average absolute error of 0.189 bits, compared to 0.481 bits for the histogram method; for $n_t = 200$, the average absolute error is 0.083 bits, compared to 0.442 bits for the histogram approach.

Figure 2. Comparing the KDE to the histogram method for high and low numbers of sources and dimensions. The true information is plotted against the estimated information; in (**A**) and (**C**), $n_s = 10$ and $n_d = 10$; in (**B**) and (**D**), $n_s = 3$ and $n_d = 3$. The top row, (**A**) and (**B**), is for the histogram method, the bottom row, (**C**) and (**D**), is for the kernel method. As before, the normalized information, $I(S;R)/\log_2 n_s$, is plotted. So, for $n_s = 10$, the information is in ban, for $n_s = 3$, in trit, and in each case, the maximum mutual information is one. $n_r = 200$ for all graphs.



In Figure 2, the histogram and kernel methods are compared for $n_s = 10$ and $n_d = 10$ and for $n_s = 3$ and $n_d = 3$; the number of trials is $n_t = 200$ in each case. The kernel method outperforms the histogram

method. When $n_s = 10$ and $n_d = 10$, the average absolute error for the kernel method is 0.139 bits, compared to 0.876 bits for the histogram method; for $n_s = 3$ and $n_d = 3$, its average absolute error is 0.076 bits compared to 0.141 bits for the histogram. Furthermore, the errors for the kernel method are less clearly modulated by the actual information, which makes the method less prone to producing misleading results.

4. Discussion

Although the actual method presented here is very different, it was inspired in part by the transmitted information method for calculating mutual information using metric-based clustering described in [18] and by the novel approach introduced in [19], where a kernel-like approach to mutual information is developed. Another significant motivation was the interesting technique given in [20], where the information is estimated by measuring how large a sphere could be placed around each data point without it touching another data point. In [20], the actual volume of the sphere is required, or, rather, the rate the volume changes with diameter. This is calculated by foliating the space of spike trains into subspaces with a fixed spike number and interpreting the spike times as coordinates. This is avoided here by using the Monte-Carlo estimate of volumes. Finally, the copula construction is related to the approach described here. In fact, the construction here can be thought of as a reverse copula construction [21].

An important part of the derivation of the kernel method is the change of measure to one based on the distribution. Since the kernel size is defined using a volume based on this measure, the radius of the kernel adapts to the density of data points. This is similar to the adaptive partitioning described, for example, in [22]. Like the plug-in method of computing mutual information for spike trains, adaptive partitioning is a discretization approach. However, rather than breaking the space into regions of fixed width, the discrete regions are chosen dynamically, using estimates of the cumulative distribution, similar to what is proposed here.

One striking aspect of KDE seen here is that it reduces to a kth nearest-neighbor (kNN) estimator. The kNN approach to estimating the mutual information of variables lying in metric spaces has been studied directly in [23]. Rather than using a KDE of the probability distribution, a Kozachenko-Leonenko estimator [24] is used. To estimate $I(X;Y)$, where X and Y are both continuous random variables taking values in $\mathcal{X}$ and $\mathcal{Y}$, Kozachenko-Leonenko estimates are calculated for $H(X)$, $H(Y)$ and $H(X,Y)$; by using different values of k in each space, the terms that would otherwise depend on the dimension of $\mathcal{X}$ and $\mathcal{Y}$ cancel.

This approach can be modified to estimate $I(R;S)$, where S is a discrete random variable. Using the approach described in [23] to estimate $H(R)$ and $H(R|S)$ gives:

$$I_e(R;S) \approx \digamma(n_k) + \digamma(n_t n_s) - \digamma(n_t) - \frac{1}{n_r}\sum_i \digamma[C(r_i, s_i; n_k)] \tag{26}$$

where $\digamma(x)$ is the digamma function, n_k is an integer parameter and $C(r_i, s_i; n_k)$ is similar to $c(r_i, s_i; n_h)$ above. Whereas $c_k(r_i, s_i; n_h)$ counts the number of responses to s_i for which r_i is one of the n_h closest data points, $C(r_i, s_i; n_k)$ is computed by first finding the distance, d, from r_i to the n_kth nearest spike-train response to stimulus s_i; then, $C(r_i, s_i; n_k)$ counts the number of spike trains, from any stimulus, that is at most a distance of d from r_i. $I_e(R;S)$ is the mutual information with base e, so

$I(R;S) = I_e(R;S)/\ln 2$. During the derivation of this formula, expressions involving the dimension of $\mathcal{S}$ appear, but ultimately, they all cancel, leaving an estimate which can be applied in the case of interest here, where $\mathcal{S}$ has no dimension. Since the digamma function can be approximated as:

$$\digamma(x) \approx \ln x - \frac{1}{2x} \tag{27}$$

for large x, this kNN approach and the kernel method produce very similar estimates. The similarity between the two formulas, despite the different routes taken to them, lends credibility to both estimators.

Other versions of the kernel method can be envisaged. A kernel with a different shape could be used or the kernel could be defined by the radius rather than by the volume of the support. The volume of the support and, therefore, the normalization would then vary from data point to data point. This volume could be estimated by counting, as it was here. However, as mentioned above, the volume-based bandwidth has the advantage that it gives a kernel that is adaptive: the radius varies as the density of data points changes. Another intriguing possibility is to investigate if it would be possible to follow [20] and [23] more closely than has been done here and use a Monte-Carlo volume estimate to derive a Kozachenko and Leonenko estimator. Finally, KDE applied to two continuous random variables could be used to derive an estimate for the mutual information between two sets of spike trains or between a set of spike trains and a non-discrete stimulus, such as position in a maze.

There is no general, principled approach to choosing bandwidths for KDE methods. There are heuristic methods, such as cross-validation [25,26], but these include implicit assumptions about how the distribution of the data is itself drawn from a family of distributions, assumptions that may not apply to a particular experimental situation. The KDE approach developed here includes a term analogous to bandwidth, and although a simple choice of this bandwidth is suggested and gives accurate estimates, the problem of optimal bandwidth selection will require further study.

Applying the KDE approach to spike trains means it is necessary to specify a spike train metric [18,27,28]. Although the metric is only used to arrange points in the order of proximity, the dependence on a metric does mean that the estimated mutual information will only include mutual information encoded in features of the spike train that affect the metric. As described in [20], in the context of another metric-dependent estimator of mutual information, this means the mutual information may underestimate the true mutual information, but it does allow the coding structure of spike trains to be probed by manipulating the spike train metrics.

It is becoming increasingly possible to measure large number spike trains from large numbers of spike trains simultaneously. There are metrics for measuring distances between sets of multi-neuron responses [29–31], and so, the approach described here can also be applied to multi-neuronal data.

Acknowledgments

R. Joshua Tobin is grateful to the Irish Research Council in Science, Engineering and Technology for financial support. Conor J. Houghton is grateful to the James S. McDonnell Foundation for financial support through a Scholar Award in Human Cognition.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. De Ruyter van Steveninck, R.R.; Lewen, G.D.; Strong, S.P.; Koberle, R.; Bialek, W. Reproducibility and variability in neural spike trains. *Science* **1997**, *275*, 1805–1808.
2. Strong, S.; Koberle, R.; de Ruyter van Steveninck, R.R.; Bialek, W. Entropy and information in neural spike trains. *Phys. Rev. Lett.* **1998**, *80*, 197–200.
3. Nemenman, I.; Bialek, W.; de Ruyter van Steveninck, R. Entropy and information in neural spike trains: Progress on the sampling problem. *Phys. Rev. E* **2004**, *69*, 056111.
4. Nemenman, I.; Lewen, G.; Bialek, W.; de Ruyter van Steveninck, R.R. Neural coding of natural stimuli: Information at sub-millisecond resolution. *BMC Neurosci.* **2007**, *8*, S7.
5. Rosenblatt, M. Remarks on some nonparametric estimates of a density function. *Ann. Math. Stat.* **1956**, *27*, 832–837.
6. Parzen, E. On estimation of a probability density function and mode. *Ann. Math. Stat.* **1962**, *33*, 1065–1076.
7. Silverman, B. *Density Estimation*; Chapman and Hall: London, UK, 1986.
8. Jones, M.C. Simple boundary correction for kernel density estimation. *Stat. Comput.* **1993**, *3*, 135–146.
9. Gillespie, J.; Houghton, C. A metric space approach to the information capacity of spike trains. *J. Comput. Neurosci.* **2011**, *30*, 201–209.
10. Paninski, L. Estimation of entropy and mutual information. *Neural Comput.* **2003**, *15*, 1191–1253.
11. Treves, A.; Panzeri, S. The upward bias in measures of information derived from limited data samples. *Neural Comput.* **1995**, *7*, 399–407.
12. Panzeri, S.; Treves, A. Analytical estimates of limited sampling biases in different information measures. *Network* **1996**, *7*, 87–107.
13. Panzeri, S.; Senatore, R.; Montemurro, M.A.; Petersen, R.S. Correcting for the sampling bias problem in spike train information measures. *J. Neurophysiol.* **2007**, *98*, 1064–1072.
14. Montemurro, M.; Senatore, R.; Panzeri, S. Tight data-robust bounds to mutual information combining shuffling and model selection techniques. *Neural Comput.* **2007**, *19*, 2913–2957.
15. Miller, G. Note on the Bias of Information Estimates. In *Information Theory in Psychology II-B*; Quastler, H., Ed.; Free Press: Glencoe, IL, USA, 1955; pp. 95–100.
16. Carlton, A. On the bias of information estimates. *Psychol. Bull.* **1969**, *71*, 108–109.
17. Victor, J.D. Asymptotic bias in information estimates and the exponential (Bell) polynomials. *Neural Comput.* **2000**, *12*, 2797–2804.
18. Victor, J.D.; Purpura, K.P. Nature and precision of temporal coding in visual cortex: A metric-space analysis. *J. Neurophysiol.* **1996**, *76*, 1310–1326.
19. Brasselet, R.; Johansson, R.S.; Arleo, A. Quantifying neurotransmission reliability through metrics-based information analysis. *Neural Comput.* **2011**, *23*, 852–881.
20. Victor, J.D. Binless strategies for estimation of information from neural data. *Phys. Rev. E* **2002**, *66*, 051903.
21. Calsaverini, R.S.; Vicente, R. An information-theoretic approach to statistical dependence: Copula information. *Europhys. Lett.* **2009**, *88*, 68003.

22. Darbellay, G.A.; Vajda, I. Estimation of the information by an adaptive partitioning of the observation space. *IEEE Trans. Inf. Theory* **1999**, *45*, 1315–1321.
23. Kraskov, A.; Stögbauer, H.; Grassberger, P. Estimating mutual information *Phys. Rev. E* **2004**, *69*, 066138.
24. Kozachenko, L.; Leonenko, N. On statistical estimation of entropy of a random vector. *Probl. Inf. Transmi.* **1987**, *23*, 9–16.
25. Rudemo, M. Empirical choice of histograms and kernel density estimators. *Scand. J. Stat.* **1982**, *9*, 65–78.
26. Hall, P. Large sample optimality of least squares cross-validation in density estimation. *Ann. Stat.* **1983**, *11*, 1156–1174.
27. Van Rossum, M. A novel spike distance. *Neural Comput.* **2001**, *13*, 751–763.
28. Houghton, C.; Victor, J.D. Spike Rates and Spike Metrics. In *Visual Population Codes: Toward a Common Multivariate Framework for Cell Recording and Functional Imaging*; Kriegeskorte, N., Kreiman, G., Eds.; MIT Press: Cambridge, MA, USA, 2012; Chapter 8.
29. Aronov, D.; Reich, D.S.; Mechler, F.; Victor, J.D. Neural coding of spatial phase in v1 of the macaque monkey. *J. Neurophysiol.* **2003**, *89*, 3304–3327.
30. Houghton, C.; Sen, K. A new multi-neuron spike-train metric. *Neural Comput.* **2008**, *20*, 1495–1511.
31. Kreuz, T.; Chicharro, D.; Houghton, C.; Andrzeja, R.G.; Mormann, F. Monitoring spike train synchrony. *J. Neurophysiol.* **2013**, *109*, 1457–1472.