Fast Multidimensional Entropy Estimation by k-d Partitioning

Dan Stowell, *Student Member, IEEE*, and Mark D. Plumbley, *Member, IEEE*

***Abstract*—We describe a nonparametric estimator for the differential entropy of a multidimensional distribution, given a limited set of data points, by a recursive rectilinear partitioning. The estimator uses an adaptive partitioning method and runs in $\Theta(N \log N)$ time, with low memory requirements. In experiments using known distributions, the estimator is several orders of magnitude faster than other estimators, with only modest increase in bias and variance.**

***Index Terms*—Entropy, estimation, multidimensional signal processing, multidimensional systems.**

I. Introduction

FOR a multivariate random variable X taking values $x \in \mathcal{X}$, $\mathcal{X} = \mathbb{R}^D$, the differential Shannon entropy is given as

$$H = -\int_{\mathcal{X}} f(x) \log f(x) dx \tag{1}$$

where $f(x)$ is the probability density function (pdf) of X [1]. Estimating this quantity from data is useful in various contexts, for example image processing [2] or genetic analysis [3]. While estimators can be constructed based on an assumed parametric form for $f(x)$, nonparametric estimators [4] can avoid errors due to model misspecification [5].

In this communication we describe a new nonparametric entropy estimator, based on a rectilinear adaptive partitioning of the data space. The partitioning procedure is similar to that used in constructing a k-d tree data structure [6], although the estimator itself does not involve the explicit construction of a k-d tree. The method produces entropy estimates with similar bias and variance to those of alternative estimators, but with improved computational efficiency of order $\Theta(N \log N)$.

In the following, we first state the standard approach to entropy estimation by adaptive partitioning (Section II), before describing our new recursive partitioning method and stopping criterion in Section III, and considering computational complexity issues in Section IV. We present empirical results on the bias, variance and efficiency of the estimator in Section V.

Manuscript received December 02, 2008; revised February 13, 2009. Current version published April 29, 2009. The work of D. Stowell was supported by the EPSRC under a Doctoral Training Account studentship; The work of M. D. Plumbley was supported by an EPSRC Leadership Fellowship (EP/G007144/1). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Lisimachos Paul Kondi.

The authors are with the Centre for Digital Music, Queen Mary University of London, London E1 4NS U.K. (e-mail: dan.stowell@elec.qmul.ac.uk).

Digital Object Identifier 10.1109/LSP.2009.2017346

II. Entropy Estimation

Consider a partition A of $\mathcal{X}$, $A = \{A_j | j = 1, \ldots, m\}$ with $A_j \cap A_k = \emptyset$ if $j \neq k$ and $\bigcup_j A_j = \mathcal{X}$. The probability mass of $f(x)$ in each cell A_j is $p_j = \int_{A_j} f(x)$. We may construct an approximation $f_A(x)$ having the same probability mass in each cell as $f(x)$, but with a uniform density in each cell:

$$f_A(x) = \frac{p_j}{\mu(A_j)}, \quad j \text{ s.t. } x \in A_j \tag{2}$$

where $\mu(A_j)$ is the D-dimensional volume of A_j.

Often we do not know the form of $f(x)$ but are given some empirical data points sampled from it. Given a set of N D-dimensional data points $\{x_i | i = 1, \ldots, N\}$, $x_i \in \mathbb{R}^D$, we estimate p_j by n_j/N where n_j is the number of data points in cell A_j. An empirical density estimate can then be made:

$$\hat{f}_A(x) = \frac{n_j}{N\mu(A_j)}, \quad j \text{ s.t. } x \in A_j. \tag{3}$$

This general form is the basis of a wide range of density estimators, depending on the choice of partitioning scheme used to specify A. A surprisingly broad class of data-adaptive partitioning schemes can be used to create a consistent estimator, meaning $\hat{f}_A(x) \to f(x)$ as $N \to \infty$ [7, Ch. 12], [8].

The within-cell uniformity of $f_A(x)$ allows us to rewrite (1) to give the following expression for its entropy:

$$H_A = \sum_{j=1}^{m} p_j \log \frac{\mu(A_j)}{p_j} \tag{4}$$

and so our partition-based estimator from data points x_j is

$$\hat{H} = \sum_{j=1}^{m} \frac{n_j}{N} \log \left(\frac{N}{n_j} \mu(A_j) \right). \tag{5}$$

To estimate the entropy from data, it thus remains for us to choose a suitable partition A for the data.

A. Partitioning Methods

A computationally simple approach to choose a partition A is to divide a dataset into quantiles along each dimension, since quantiles provide a natural way to divide a single dimension into regions of equal empirical probability. Indeed, in one dimension this approach leads to estimators such as the "m_N-spacing" estimator of [10] (see also [11]). In the multidimensional case, by dividing each dimension of $\mathbb{R}^D$ into q-quantiles, we would create a product partition having q^D cells. However, such a product partition can in fact lead to poor estimation at limited

number of data points N because $f(x)$ is not in general equal to the product of its marginal densities, and so the product partition may be a poor approximation to the structure of the ideal data partition [12].

Data-driven nonproduct partitioning methods exist. Voronoi partitioning divides the space such that each data point is the centroid of a cell, and the boundary between two adjacent cells is placed equidistant from their centroids. Delaunay triangulation partitions the space using a set of simplices defined with the data points at their corners [13, Ch. 13]. Such partitions are amenable to entropy estimation by (5), as considered by Learned-Miller [14]. However, the complexity of such diagrams has a strong interaction with dimensionality: although 2-D diagrams can be $\mathrm{O}(N \log N)$ in time and storage, at $D \geq 3$ they require $\mathrm{O}(N^{\lceil (D+1)/2 \rceil})$ time and $\mathrm{O}(N^{\lceil D/2 \rceil})$ storage [13, Ch. 13].

Partitioning by tree-like recursive splitting of a dataset is attractive for a number of reasons. It is used in nonparametric regression [7] as well as in constructing data structures for efficient spatial search [6]. The nonproduct partitions created can take various forms, but in many schemes they consist of hyperrectangular cells whose faces are axis-aligned. Such hyperrectangle-based schemes are computationally advantageous because the storage complexity of the cells does not diverge strongly, requiring only $2D$ real numbers to specify any cell. A notable example here is Darbellay and Vajda's 2-D mutual information estimator [12], which recursively splits a dataset into four subpartitions until an independence criterion is met. In Section III we will describe our new method which has commonalities with this approach, but is specialised for the fast estimation of multidimensional entropy.

B. Support Issues

If the support of the data is not known or unbounded then there will be open cells at the edges of A. These are problematic because they have effectively infinite volume and zero density, and cannot be used to calculate (5). One solution is to neglect these regions and adjust N and m to exclude the regions and their data points [14]. But for small datasets or high dimensionality, this may lead to the estimator neglecting a large proportion of the data points, leading to an estimator with high variance. It also leads to a biased estimate, tending to underestimate the support.

An alternative is to limit edge cells to finite volume by using the Maximum Likelihood Estimate of the hyperrectangular support. This reduces to the estimate that the extrema of the data sample define the support (since any broadening of the support beyond the extrema cannot increase the posterior probability of the data sample). This is of course likewise a biased estimate, but does not exclude data points from the calculation of (5), and so should provide more efficient estimation at low N. We use this approach in the following.

III. Adaptive Partitioning Method

Since the approximation $\hat{f}_A(x)$ has a uniform distribution in each cell, it is reasonable to design our adaptive partitioning scheme deliberately to produce cells with uniform empirical distribution, so that $\hat{f}_A(x)$ best approximates $f(x)$ at limited N. Partitioning by recursively splitting a dataset along quantiles produces a consistent density estimator [7, chapter 12], [8], so we design such a scheme whose stopping criterion includes a test for uniformity.

At each step, we split a set of data points by their sample median along one axis, producing two subpartitions of approximately equal probability. This has a close analogy in the approach used to create a k-d tree data structure [6], hence we will refer to it as k-d *partitioning*. Such rectilinear partitioning is computationally efficient to implement: not only because the splitting procedure needs only consider one dimension at a time, but because unlike in the Voronoi or Delaunay schemes any given cell is a hyperrectangle, completely specified by only $2D$ real numbers.

It remains to select a test of uniformity. Various tests exist [15], but in the present work we seek a computationally efficient estimator, so we require a test which is computationally light enough to be performed many times during estimation (once at each branch of the recursion). Since our partitioning scheme requires measurement of the sample median, we might attempt to use the distribution of the sample median in a uniform distribution to design a statistical test for uniformity.

The distribution of the sample median tends to a normal distribution [16] which can be standardised as

$$Z_j = \sqrt{n_j} \frac{2 \cdot \mathrm{med}_d(A_j) - \min_d(A_j) - \max_d(A_j)}{\max_d(A_j) - \min_d(A_j)} \tag{6}$$

where $\mathrm{med}_d(A_j)$, $\min_d(A_j)$, $\max_d(A_j)$ respectively denote the median, minimum and maximum of the hyperrectangular cell A_j along dimension d. An improbable value for Z_j (we use the 95% confidence threshold for a standard normal distribution, $|Z_j| > 1.96$) indicates significant deviation from uniformity, and that the cell should be divided further.

This test is weak, having a high probability of Type II error if the distribution is nonuniform along a dimension other than d, and so can lead to early termination of branching. We therefore combine it with an additional heuristic criterion that requires partitioning to proceed to at least a minimum branching level L_N, so that the cell boundaries must reflect at least some of the structure of the distribution. We use the partitioning level at which there are $\sqrt{N}$ data points in each partition,

$$L_N = \left\lceil \frac{1}{2} \log_2 N \right\rceil . \tag{7}$$

This is analogous to the common choice of $m = \sqrt{N}$ in the m-spacings entropy estimator, which in that case is chosen as a good compromise between bias and variance [14]. Our combined stopping criterion is therefore $L \geq L_N$ and $|Z_j| > 1.96$.

The recursive estimation procedure is summarised as pseudocode in Fig. 1.

To produce a reasonable estimate, we expect to require a minimum amount of data values. We require the estimator to be able to partition at least once along each dimension—in order that no dimension is neglected in the volume estimation—so the estimator must have the potential to branch to D levels. The number of levels in the full binary tree approximates $\log_2 N$, which gives us a lower limit of $N \geq 2^D$. This limit will become important at high dimensionality.

```
KDPEE({x_i}, D, N)
    L_N ← result of equation (7)
    A_0 ← range({x_i})
    return KDPEE_RECURSE(A_0, 1)

KDPEE_RECURSE(A, level)
    d ← level  mod D
    n ← count(x_i ∈ A)
    med ← median along dth dimension of x_i ∈ A
    Z ← result of equation (6)
    if level ≥ L_N and |Z| ≥ 1.96
      then
          return (n/N) log((N/n) μ(A))
      else
          U ← A ∩ (dimension_d < med)
          V ← A ∩ (dimension_d ≥ med)
          return KDPEE_RECURSE(U, level+1)
                 +KDPEE_RECURSE(V, level+1)
```

Fig. 1. The k-d partitioning entropy estimation algorithm for a set of N D-dimensional data points $\{x_i\}$. Note that the dimensions are given an arbitrary order, $0 \ldots (D-1)$. A_0 is the initial partition with a single cell containing all the x_i.

IV. Complexity

The complexity of all-nearest-neighbour-based estimators such as that of Kybic [17] is dominated by their All Nearest Neighbour (ANN) search algorithm. The naïve ANN search takes $\Theta(N^2D)$ time, but improved methods exist [18]. For example, using a *cover tree* data structure, ANN can be performed in $\mathrm{O}(c^{12}N \log N)$ time, where c is a data-dependent "expansion constant" which may grow rapidly with increasing dimensionality [19]. Time complexity of $\mathrm{O}(N \log N)$ is possible in a parallel-computation framework [20].

Learned-Miller's estimator based on Voronoi-region partitions [14] is, like ours, a multidimensional partitioning estimator. As discussed in Section II-A the complexity of Voronoi or Delaunay partitioning schemes is $\mathrm{O}(N^{\lceil (D+1)/2 \rceil})$ in time and $\mathrm{O}(N^{\lceil D/2 \rceil})$ in storage, meaning that for example a 3-D Voronoi diagram is $\mathrm{O}(N^2)$ in time and storage.

Kernel density estimation (KDE) can also be a basis for entropy estimation [4]. Methods have been proposed to improve on the naïve KDE complexity of $\mathrm{O}(N^2D)$, although their actual time complexity is not yet clear [21].

For our algorithm, the time complexity is dominated by the median partitioning, which we perform in $\Theta(N)$ time using Hoare's method [22]. At each partitioning level we have m_L cells each containing approximately N/m_L points, meaning that the total complexity of the m_L median-finding operations remains at $\Theta(N)$ for each level. For any given dataset, the stopping criterion (6) may result in termination as soon as we reach level L_N or may force us to continue further, even to the full extent of partitioning. Therefore the number of levels processed lies in the range $(1/2)\log_2 N$ to $\log_2 N$. This gives an overall time complexity of $\Theta(N \log N)$ at any dimensionality. For $D > 2$ and a single processor this is therefore an improvement over the other methods.

The memory requirements of our algorithm are also low. In-place partitioning of the data can be used, and no additional data structures are required, so space complexity is $\Theta(N)$. This is the same order as the cover-tree-based ANN estimator, and better than the Voronoi-based estimator.

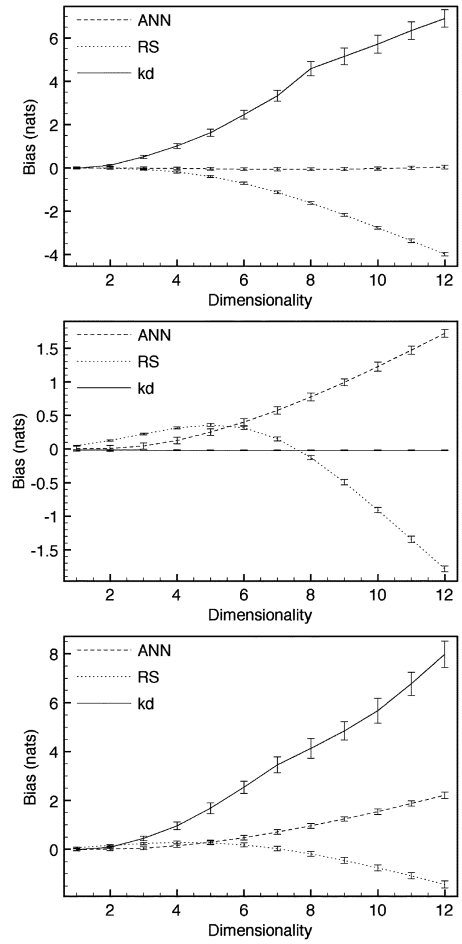

Fig. 2. Bias of some entropy estimators at increasing dimensionality. Error bars show the 95% confidence interval exaggerated by a factor of 10 for visibility. Distributions tested are gaussian (top), uniform (middle), exponential (bottom). $N = 5000$, 100 runs. **ANN** = all-nearest-neighbours estimator. **RS** = resubstitution estimator. **kd** = k-d partitioning estimator.

V. Experiments

We tested our k-d estimation algorithm against samples from some common distribution types, with $N = 5000$ and D from 1 to 12. In each case we ran 100 simulations and calculated the mean deviation from the theoretical entropy value of the distribution, as well as the variance of the entropy estimates. These will be expressed as deviations from the true entropy, which in all cases was 2 nats $(2/\ln 2 \text{ bits})$ or greater.

For comparison, we also tested two other common types of estimator: a KDE-based resubstitution estimator, and an ANN estimator. We used publicly-available implementations due to Ihler,[1] which use a k-d tree to speed up the KDE and ANN algorithms. All three implementations are Matlab code using C/C++ for the main calculations. We did not test the Voronoi-region-based estimator because it becomes impractical beyond around 4 dimensions (Learned-Miller, pers. comm.).

Fig. 2 plots the bias for up to 12 dimensions, for each of the three different estimators. In general, the estimators all provide bias performance at a similar order of magnitude and with a

[1]http://www.ics.uci.edu/~ihler/code/

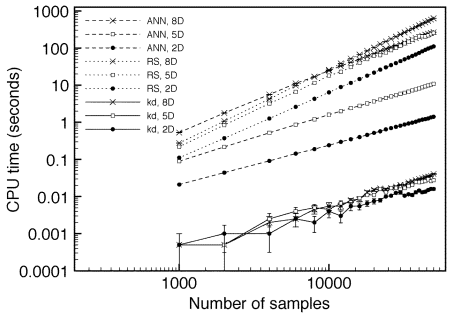

Fig. 3. CPU time for the estimators in Fig. 2, using Gaussian distributions and $D \in 2, 5, 8$. Tests performed in Matlab 7.4 (Mac OSX, 2 GHz Intel Core 2 Duo processor). Data points are averaged over 10 runs each (20 runs each for our estimator). 95% confidence intervals are shown (some are not visible).

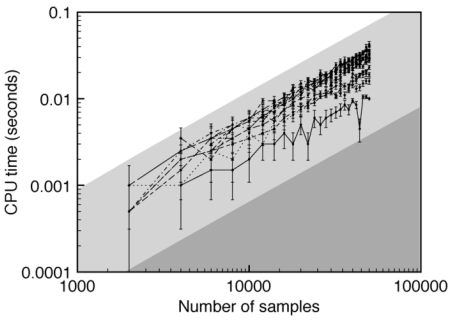

Fig. 4. CPU time for our estimator, calculated as in Fig. 3 but for all D ranging from 1 to 12. The shaded areas indicate slopes of $kN \log N$.

similar deterioration at higher dimensionality, although our estimator exhibits roughly twice as much bias as the others. The narrow confidence intervals on the graphs (exaggerated for visibility in Fig. 2) reflect the low variance of the estimators.

The upward bias of our estimator for nonuniform distributions at higher dimensions is likely to be due to underestimation of the support, neglecting regions of low probability (see Section II-B). This would lead to some overestimation of the evenness of the distribution and therefore of the entropy. Since the estimator is consistent, this bias should decrease with increasing N.

Fig. 3 plots the CPU time taken by the same three estimators, at various data sizes and $D \in 2, 5, 8$. In all tested cases our estimator is faster, by between one and three orders of magnitude. More importantly, the times taken by the resubstitution and ANN estimators diverge much more strongly than those for our estimator, at increasing D and/or N. As we expect from Section IV, CPU time for our estimator is broadly compatible with $\Theta(N \log N)$ (Fig. 4).

VI. Conclusion

We have described a nonparametric entropy estimator using k-d partitioning which has a very simple and efficient implementation on digital systems, running in $\Theta(N \log N)$ time for any dimensionality of data. In experiments with known distributions, our estimator exhibits bias and variance comparable with other estimators.

The estimator is available for Matlab or GNU Octave.[2]

Acknowledgment

The authors would like to thank S. Abdallah, J. Reiss, E. Learned-Miller and A. Ihler for useful discussions, and A. Ihler for making his entropy estimation library available.

References

[1] C. Arndt, *Information Measures*. Hong Kong: Springer, 2001.
[2] C.-I. Chang, Y. Du, J. Wang, S.-M. Guo, and P. D. Thouin, "Survey and comparative analysis of entropy and relative entropy thresholding techniques," *Proc. Inst. Elect. Eng., Vis., Image Signal Process.*, vol. 153, no. 6, pp. 837–850, Dec. 2006.
[3] D. C. Martins, U. M. Braga-Neto, R. F. Hashimoto, M. L. Bittner, and E. R. Dougherty, "Intrinsically multivariate predictive genes," *J. Sel. Topics Signal Process.*, vol. 2, no. 3, pp. 424–439, Jun. 2008.
[4] J. Beirlant, E. J. Dudewicz, L. Gyorfi, and E. C. van der Meulen, "Nonparametric entropy estimation: An overview," *Int. J. Math. Statist. Sci.*, vol. 6, pp. 17–39, 1997.
[5] J. Victor, "Binless strategies for estimation of information from neural data," *Phys. Rev. E*, vol. 66, no. 5, p. 51903, 2002.
[6] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.
[7] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. London, U.K.: Chapman & Hall/CRC, 1984.
[8] L. C. Zhao, P. R. Krishnaiah, and X. R. Chen, "Almost sure L_r-norm convergence for data-based histogram density estimates," *Theory Probabil. Applicat.*, vol. 35, no. 2, pp. 396–403, 1990.
[9] G. Lugosi and A. Nobel, "Consistency of data-driven histogram methods for density estimation and classification," *Ann. Statist.*, vol. 24, no. 2, pp. 687–706, 1996.
[10] O. Vasicek, "A test for normality based on sample entropy," *J. Royal Statist. Soc. B (Methodological)*, vol. 38, no. 1, pp. 54–59, 1976.
[11] E. G. Learned-Miller and J. W. Fisher, III, "ICA using spacings estimates of entropy," *J. Mach. Learn. Res.*, vol. 4, pp. 1271–1295, 2003.
[12] G. A. Darbellay and I. Vajda, "Estimation of the information by an adaptive partitioning of the observation space," *IEEE Trans. Inf. Theory*, vol. 45, no. 4, pp. 1315–1321, 1999.
[13] H. Edelsbrunner, *Algorithms in Computational Geometry*, ser. EATCS Monographs on Theoretical Computer Science. Berlin, Germany: Springer, 1987, vol. 10.
[14] E. G. Learned-Miller, "A new class of entropy estimators for multi-dimensional densities," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'03)*, Apr. 2003, vol. 3, no. III, pp. 297–300.
[15] C. P. Quesenberry and F. L. Miller, "Power studies of some tests for uniformity," *J. Statist. Comput. Simul.*, vol. 5, no. 3, pp. 169–191, 1977.
[16] J. T. Chu, "On the distribution of the sample median," *Ann. Math. Statist.*, vol. 26, pp. 112–116, 1955.
[17] J. Kybic, "Incremental updating of nearest neighbor-based high-dimensional entropy estimation," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'06)*, 2006, vol. 3, pp. 804–807.
[18] G. Navarro and R. Baeza-Yates, "Searching in metric spaces," *ACM Comput. Surv.*, vol. 33, pp. 273–321, 2001.
[19] A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," in *Proc. 23rd Int. Conf. Machine Learning*, New York, 2006, pp. 97–104, ACM Press.
[20] P. B. Callahan, "Optimal parallel all-nearest-neighbors using the well-separated pair decomposition," in *Proc. 34th IEEE Symp. Foundations of Computer Science*, 1993, pp. 332–340.
[21] D. Lang, M. Klaas, and N. de Freitas, "Insights on Fast Kernel Density Estimation Algorithms," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2004, Tech. Rep..
[22] C. A. R. Hoare, "Algorithm 63 (partition) and algorithm 65 (find)," *Commun. ACM*, vol. 4, pp. 321–322, 1961.

[2]http://www.elec.qmul.ac.uk/digitalmusic/downloads