Lempel-Ziv and CTW entropy estimators for spike trains

Yun Gao
Div. of Applied Math.
Brown University
Providence, RI 02912
gao@dam.brown.edu

Ioannis Kontoyiannis
Div. of Applied Math. & Dept. of Computer Science
Brown University
Providence, RI 02912
yiannis@dam.brown.edu

Elie Bienenstock
Div. of Applied Math. & Dept. of Neuroscience
Brown University, Providence, RI 02912
elie@dam.brown.edu

Abstract

We consider two entropy estimators based on the Lempel-Ziv data compression algorithm, and we study their theoretical properties and their performance on estimating the information content of spike trains. The first estimator is a known, widely used technique, and the second one is a new estimator, which we prove can be applied to a broader range of experimental data. Their major advantages are that: 1) They make minimal assumptions on the nature of the data distribution; 2) They naturally take into account finer dependence structure arising from long-range dependence; and 3) Their parameters can be adjusted so as to balance the bias-variance trade-off. We prove that the second estimator is "universal" (i.e., it converges on all stationary and ergodic processes), and we examine the convergence rate (in term of bias and variance) for both algorithms. Furthermore, we introduce a new nonparametric variance estimation method, based on the stationary bootstrap. We report the performances of our entropy estimators on the spike trains of 29 neurons recorded simultaneously for a one-hour period from the primary motor and dorsal premotor cortices of a quietly seated monkey not engaged in a behavior, as well as on various types of simulated data. The results show that the main drawback of these methods is their slow convergence rate.

1 Introduction

Information-theoretic methods have been widely used in neuroscience, and the entropy has been adopted as the main measure for quantifying the amount of information transmitted by

a spike train. The most commonly used technique to estimate this entropy has been the so-called "plug-in" (or maximum-likelihood) estimator and its various modifications, which essentially counts the empirical frequencies of all words of a fixed length in the data, and then calculates the entropy of this empirical frequency distribution; see, e.g., [10][4] [12] [7][9]. For computational reasons, the plug-in estimator cannot go beyond word lengths of about 10 or 20, and hence it does not take into account the longer time dependence in the signal. Here we examine the performance of two entropy estimators based on the Lempel-Ziv data compression algorithm. One of them has been widely and very successfully used in various applications, and the other one is a new estimator with some more desirable theoretical properties. As we will see, both estimators naturally incorporate dependence in the process at much larger time scales, and they are consistent for a wide class of data types generated from distributions that may posses arbitrarily long memory.

The Lempel-Ziv algorithm [15][16] is a universal compression scheme that achieves the (optimal) entropy lower bound when applied to data generated by *any* stationary ergodic process. As the conditions of stationarity and ergodicity are very weak (and in some sense minimal), they appear well-suited for neural data, as we have no *a priori* bound on the length of the memory in the data, and in fact the very length of this memory is one of the main objects we intend to study.

The main gist in the workings of the Lempel-Ziv algorithm was revealed by Wyner and Ziv in [14], where they studied the connection between the entropy of a process and the longest match-lengths along a process realization. Roughly speaking, the match-lengths measure the length of the longest string starting in a fixed position in the process which re-appears somewhere else in the process (a detailed definition is given below). Intuitively, the longer the match-lengths, the more regularity there is in the data, and hence the smaller the entropy (and the more efficient the Lempel-Ziv compression). Partly motivated by this connection, a number of entropy estimators have been proposed since then and have been applied to many different kinds of data; for examples see [8][2] and the references therein.

Here we use two entropy estimators based on match-lengths, one described in [2], and a new one. We study their theoretical properties and apply them to both simulated and neuronal data. Due to space limitations we only state the main theoretical results we need without proof, and we present the experimental results in more detail. We refer the reader to [1] for a complete treatment.

Our neuronal data come from two multi-electrode arrays implanted on a monkey's primary motor cortex (MI) and dorsal premotor cortex (PMd). The arrays simultaneously recorded neural activity from 29 different neurons. A Plexon acquisition system was used to collect neural signal, and the units were spike-sorted using Plexon's Offline Sorter. The monkey was not engaged in any task when the data were collected, and the size of the data is approximately an hour. A detailed description of recording techniques is given in [3].

2 The Methods

2.1 The Lempel-Ziv Estimators

Let $(\ldots, X_{-1}, X_0, X_1, X_2, \ldots)$ be a random process with values in a finite set A. In the case of neuronal data, the process $\{X_i\}$ takes on the values 1 and 0, signifying whether or not a spike occurred in a given neuron's spike train within a given 1ms window. For every position i in the data, and any "window length" n, we consider the length of the longest

segment in the data starting at i which also appears in the window $(X_{i-n}, \ldots, X_{i-2}, X_{i-1})$ of length n preceding position i. Specifically, we define L_i^n as $1+$ that longest match length:

$$L_i^n = 1 + \max\{\ell : (X_i, \ldots, X_{i+\ell}) = (X_j, \ldots, X_{j+\ell}) \text{ for some } i - n \leq j \leq i - 1\}.$$

In [14] it was shown that, if the entropy of the process is H, then for any fixed position i, the match lengths grow logarithmically with the window size n, and in fact $L_i^n / \log n \to 1/H$ as $n \to \infty$. This result suggests that the quantity $\log n / L_i^n$ can be used as an entropy estimator, and, clearly, in order to make more efficient use of the data and reduce the variance, it would be more reasonable to look at the average value of various match-lengths L_i^n taken at different positions i; see the discussion in [2]. To that effect, the following estimator is considered in [2]:

$$\hat{H}_{n,k} = \left[\frac{1}{k} \sum_{i=1}^{k} \frac{L_i^n}{\log n} \right]^{-1}, \tag{1}$$

where it is shown that, under appropriate conditions and when the number of matches k equals the window length n, the estimator $\hat{H}_{n,k}$ is consistent, i.e., it converges to the entropy of the underlying process as $n \to \infty$. To be specific, it is assumed that the process is stationary, ergodic, it takes on only finitely many values, and most importantly that it satisfies the *Doeblin Condition* (DC). This condition says that there is a finite number of steps, say r, in the process, such that, after r time steps, no matter what has occurred before, anything can happen with positive probability. Formally, it is assumed that there exists an integer $r \geq 1$ and a real number $\beta > 0$ such that, $\Pr(X_r = a \,|\, X_0, X_{-1}, \ldots) > \beta$, for all a and with probability one in the conditioning [i.e., for almost all semi-infinite realizations of the past $(\, X_0, X_{-1}, \ldots)$]. Although condition (DC) is not very restrictive, and one can easily argue that it is probably satisfied for neural data, we will see that estimator $\hat{H}_{n,k}$ can be modified so as to make (DC) irrelevant. To that end, we introduce:

$$\tilde{H}_{n,k} = \frac{1}{k} \sum_{i=1}^{k} \frac{\log n}{L_i^n}. \tag{2}$$

Below we list three basic properties of $\hat{H}_{n,k}$ and $\tilde{H}_{n,k}$.

Theorem 1 *Let $(\ldots, X_{-1}, X_0, X_1, X_2, \ldots)$ be an arbitrary random process with values in a finite set A.*

(i) For any values of k and n, with probability one we have $\tilde{H}_{n,k} \geq \hat{H}_{n,k}$.

(ii) If the process is stationary, ergodic, and it satisfies Doeblin's condition (DC), then with probability one we have:

$$\hat{H}_{n,k} \to H, \quad \text{as } k, n \to \infty.$$

(iii) If the process is stationary and ergodic (even if (DC) does not hold), then with probability one we have:

$$\tilde{H}_{n,k} \to H, \quad \text{as } k, n \to \infty.$$

We only give a brief outline of the proof; see [1] for full details. But first some remarks are in order. Note that in parts (ii) and (iii) we did not specify in what way the parameters k and n go to infinity. From the proof it is immediate that we have convergence in the

following cases: 1. If k and n both go to infinity at roughly the same rate so that $k/n \to 1$; 2. If $n \to \infty$ and k varies arbitrarily but stays bounded; 3. If $n \to \infty$ and $k = k_n$ varies with n in such a way that it increases to infinity as $n \to \infty$; and 4. If the two limits as n and k tend to infinity are taken separately, i.e., first $k \to \infty$ and then $n \to \infty$, or vice versa. More general cases are considered in [1].

Proof Outline. Part (i) follows by Jensen's inequality applied to the convex function $1/x$, and with respect to the uniform distribution $(1/k, 1/k, \ldots, 1/k)$. Part (ii) was proved in [2] for the case $k = n$. The additional cases mentioned above follow easily from the same proof. Part (iii) is proved in an analogous way in [1], using Maker's theorem and Kac's lemma. □

2.2 Bias and Variance

In practical applications with a finite amount of data, we need to choose the values of the parameters k and n so that $k + n$ is approximately equal to our total data length. Here we are faced with the following trade-off: Using a long window size n we are more likely to capture the longer-term dependence in the data, but as shown in [5][11] the match lengths L_i^n starting at different positions i have great fluctuation. So a large window size n and a small number of matching positions k will yield estimates with high variance. On the other hand, if we take n small and average a large number k of estimates we reduce the variance but we increase the bias, since the expected value of $L_i^n / \log n$ converges to $1/H$ very slowly [13].

Therefore we need to choose n and k such the above bias/variance trade-off is balanced. Looking at the earlier theoretical results of [5][11][13] in more detail, we argue in [1] that under appropriate conditions, the bias varies approximately as $O(1/\log n)$, whereas the variance is approximately $O(1/k)$. This indicates that we should probably choose values of n and k such that $k \approx O(\log n)^2$.

Although the above theoretical estimates yield useful guidelines for choosing n and k, we also need a method for evaluating the relative estimation error on particular data sets. To that effect we consider the following procedure, which adapts the stationary bootstrap of [6] to our problem. Let $L = (L_1^n, L_2^n, \ldots, L_k^n)$ be the sequence of match-lengths computed directly from the data. For $m = 1, 2 \ldots, B$, let L^{*m} be the m-th resampled pseudo-time series with the same length as L, which yields the estimates $\hat{H}^*(m)$ and $\tilde{H}^*(m)$ corresponding to (1) and (2) respectively. The bootstrap estimate of the variance of $\hat{H}$ is

$$\hat{\sigma}^2 = \sum_{b=1}^{B} [\hat{H}^*(b) - \hat{H}^*(\cdot)]^2/(B-1),$$

where $\hat{H}^*(\cdot) = \sum_{b=1}^{B} \hat{H}^*(b)/B$; similarly for $\tilde{H}$. Finally, the resampling of L is done as follows: Each time we randomly draw a subblock of L with random length, geometrically distributed with mean $1/p$, and then we concatenate the blocks together until reaching length k. Finally, the choice of p is done by studying the autocorrelogram of L (which is typically decreasing with the lag) and choose a cutoff threshold. We then take the corresponding lag to be the average block size, and choose p as the reciprocal of that lag.

3 Entropy Estimates of Spike Trains

Our spike train data are binned with 1 ms bin size, and have total length $T = 3,606,073$. For the estimation we chose the window size n to be 99.4% of T, so that $n = 3,584,463$ and and the number of matches $k = T - n = 21,610$. Table 1 show the estimates produced by $\hat{H}_{n,k}$ and $\tilde{H}_{n,k}$ and compares them with other entropy estimation methods.

neuron	plug-in word=20ms	Ref. [10]	$\hat{H}$	$\tilde{H}$
1	1.9330	1.6389	0.7290	1.5241
2	4.9075	4.7458	2.5020	3.7588
3	4.5934	4.0868	1.4481	2.8372
4	2.8103	2.5679	1.0962	1.6583
5	2.9040	2.3871	1.5636	2.6849
6	2.6641	2.3677	1.1090	2.8625
7	3.3830	2.6269	1.3786	3.5143
8	1.7815	1.8208	0.7406	2.3843
9	1.9040	1.7852	0.6644	1.7293
10	2.3764	2.5448	1.2055	3.2562
11	14.0634	13.6261	5.7085	8.7684
12	0.3142	0.3262	0.2796	1.0902
13	7.2784	7.4749	3.8182	5.5020
14	0.4695	0.5022	0.1965	0.8127
15	3.7516	4.2728	2.7643	4.8172
16	0.1045	0.0778	0.0896	0.4336
17	4.4711	3.4756	1.2266	3.3472
18	5.7890	6.2609	1.8508	3.3505
19	4.9886	4.7457	1.6807	3.3362
20	2.4904	2.8044	0.9557	1.7657
21	0.1933	0.1428	0.3988	1.0176
22	0.2350	0.2334	0.4006	1.7245
23	1.8215	1.4728	0.9895	2.3280
24	4.0583	3.8855	2.2699	3.1679
25	2.5494	1.8335	1.6671	4.2782
26	3.7618	2.3589	2.1744	3.2721
27	0.9157	0.6415	0.4143	1.2433
28	2.0656	1.9417	1.2165	2.4484
29	2.9396	2.6239	1.5072	2.7408

Table 1: Entropy estimates by several methods in bits per 50ms

The main limitation of the plug-in estimate is that it can only use words of length up to 20ms, and for word lengths around 20ms the undersampling problem makes the estimate unstable. Moreover, this method completely misses the effects of longer term dependence. The method in [10] also begins with the plug-in for relatively short word lengths, and then extrapolates in an *ad hoc* fashion to "infinite" word lengths (which would correspond to the true entropy rate of the underlying process). It is worth noting that, although the plug-in estimator always has negative bias, when used with a finite word length ℓ it only gives an estimate of the order-ℓ entropy, which is typically larger than the entropy rate itself.

Match-length estimators, on the other hand can deal with a much longer window size. As we see from Table 1, $\hat{H}$ is much lower than both versions of plug-in for 26 out of 29 cells, which indicates that the $\hat{H}$ estimate is more accurate.

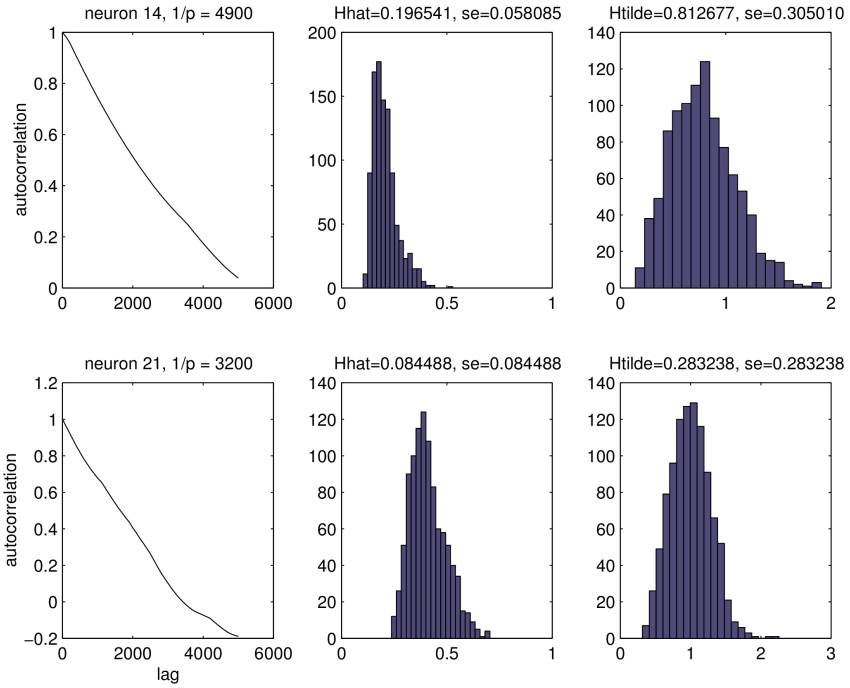

Figure 1: Bootstrap estimate of the standard error of $\hat{H}$ and $\tilde{H}$ for two cells. The first subplot shows how the average block size p is chosen (cutoff threshold is 0.05), and the second and third plots show the histogram of the bootstrap replications.

The main drawback of the match-length estimators is their slow rate of convergence. The bias is relatively and very hard to evaluate analytically. The variance on the other hand can be estimated using the bootstrap technique described above. See Figure 1. Note that the histogram of the bootstrap replications looks not very far from Gaussian, which indicates that the standard error estimate has approximately converged to its limiting Gaussian distribution.

4 Results for Simulated Data

To get a better idea of how quickly $\hat{H}$ and $\tilde{H}$ converge, we first apply them to simulated data generated from an $i.i.d.$ process (also sometimes referred to as a homogeneous Poisson model) and to a homogeneous Markov chain. In Table 2 we show the resulting entropy estimates on five realizations of an $i.i.d.$ process with rate 50 Hz, which is close to the typical firing rate of a neuron In Table 3 we show corresponding results on five realizations of homogeneous Markov chain with transition matrix P =[0.1 0.9 ; 0.9 0.1]. In both cases we chose the window size n=3,590,000, and the number of matches k=10,000, as in the neuronal data. The resulting bias is about -11% for $\hat{H}$, and +13% for $\tilde{H}$, so the large bias is indeed the major problem of these match-length estimators.

We also applied $\hat{H}$ and $\tilde{H}$ to data from a slightly more realistic model for neuronal data,

namely, an independent process with varying rate (also referred to as an "inhomogeneous Poisson process") with a rate function drawn from a Gaussian process with varying kernel width, which is supposed to simulate the slow varying rates of neurons. The data length and the parameter choices are the same as above. Table 4 shows entropy estimates for two different kernel widths. Each time, a rate function is first drawn at random, and an inhomogeneous Poisson process is generated with that rate function. The results show that the estimates are varying wildly.

No.	plug-in word=20ms	Ref. [10]	$\hat{H}$	$\tilde{H}$
1	7.0943	6.9841	5.8309	7.3311
2	7.0298	7.1576	6.3735	8.0865
3	7.0804	7.1496	6.6101	8.4944
4	7.0713	7.0087	6.0549	7.6590
5	7.0621	7.0914	6.4599	8.2073
bootstrapped $\hat{\sigma}$			0.2751	0.4749

Table 2: Entropy estimates on simulated data from a homogeneous Poisson process, in bits per 50ms. The true entropy is H_{true}=7.072. The bootstrap estimated standard errors are for the first realization.

No.	plug-in word=20ms	Ref. [10]	$\hat{H}$	$\tilde{H}$
1	24.7004	23.1110	22.1769	24.3011
2	24.7196	23.2915	21.7436	24.0792
3	24.7087	23.1009	21.9016	24.5349
4	24.6937	23.1689	21.8957	24.3082
5	24.7182	23.2156	21.7458	24.0298
bootstrapped $\hat{\sigma}$			0.4397	0.5221

Table 3: Entropy estimates on simulated data from a Markov Chain, in bits per 50ms. The true entropy is H_{true}=23.4498. The bootstrap estimated standard errors are for the first realization.

5 Concluding Remarks

We examined the performance of two entropy estimators inspired by the Lempel-Ziv compression algorithm, and based on match-lengths. These estimators are consistent under very weak conditions, but their convergence rate is slow. In cases when the memory of the process generating the data is short, the plug-in estimator is an adequate method, and it actually outperforms the match-length estimators. But in cases when the memory length is not known or is to be tested (as in the case of neural data), the match-length estimators are more appropriate and their estimates are at least known to be consistent in the large data limit, whereas those of the plug-in method is not.

	5 sec				200 sec			
No.	plug-in word=20ms	Ref. [10]	$\hat{H}$	$\tilde{H}$	plug-in word=20ms	Ref. [10]	$\hat{H}$	$\tilde{H}$
1	7.0362	7.2129	7.2501	9.4366	7.0530	2.1632	5.0155	6.5355
2	7.0342	6.9179	6.1985	7.6106	7.0414	8.4571	7.4371	9.0451
3	7.0446	6.9657	5.1667	6.3577	7.0310	4.6627	8.6755	11.3509
4	7.0681	6.9708	4.8771	6.7595	7.0589	9.0950	5.5526	7.0706
5	7.0530	6.2969	7.7206	10.1253	7.0284	6.8757	6.9882	9.0128
6	7.0464	6.8660	6.4834	8.6458	7.0257	6.666	4.7727	6.3624
7	7.0517	7.2284	5.6214	7.7639	7.0552	2.9313	9.1142	11.5276
8	7.0827	6.7124	7.1077	9.4711	7.0636	7.2763	6.0086	7.5258
9	7.0726	7.3780	7.0379	9.3916	7.0817	5.6254	6.9552	8.8151
10	7.0690	7.0543	5.3524	7.0504	7.0953	6.1526	7.1462	8.9858

Table 4: Entropy of simulated data from inhomogeneous Poisson process in bits per 50ms. The rate functions are drawn from Gaussian process with different kernel width. Bootstrap estimated standard errors are for the first realization.

Acknowledgment

Y. Gao was supported by Burroughs Wellcome fund. I. Kontoyiannis was supported in part by NSF grant #0073378-CCR, and by USDA-IFAFS grant #00-52100-9615. E. Bienenstock was supported by NSF-ITR Grant #0113679 and NINDS Contract N01-NS-9-2322. We thank Nicho Hatsopoulos for providing us with this neural dataset.

References

[1] Y. Gao, I. Kontoyiannis and E. Bienenstock. Estimating the entropy rate of spike trains using Lempel-Ziv estimators. *Preprint*, 2003

[2] I. Kontoyiannis, P.H. Algoet, Yu.M. Suhov, and A.J. Wyner. Nonparametric entropy estimation for stationary processes and random fields, with applications to English text. *IEEE Trans. Inform. Theory*, 44:1319–1327, 1998

[3] E. Maynard, N. Hatsopoulos, C. Ojakangas, B. Acuna, J. Sanes, R. Normann, and J. Donoghue. Neuronal interaction improve cortical population coding of movement direction. *J. of Neuroscience*, 19(18):8083–8093, 1999

[4] L. Paninski. Estimation of entropy and mutual information. *Neural Comp.*, 15: 1191–1253, 2003.

[5] B. Pittel. Asymptotic growth of a class of random trees. *Ann. Probab.*, 13:414–427,1985

[6] D. Politis and J. Romano. The stationary bootstrap. *JASA*, 89:1303–1313, 1994

[7] P. Reinagel. Infomation theory in the brain. *Current Biology*, 10(15):R542-R544, 2000

[8] T. Schürmann and P.Grassberger. Entropy estimation of symbol sequences. *Chaos*, 6:414–427, 1996

[9] C.F. Stevens and A.Zador. Information through a spiking neuron *NIPS*, 8, 1996

[10] S.P. Strong, R. Koberle, R. de Ruyter van Steveninck, and W. Bialek. Entropy and information in neural spike trains. *Physical Review Letters*, 80:197–200, 1998

[11] W. Szpankowski. Asymptotic properties of data compression and suffix trees. *IEEE Trans. Inform. Theory*, 39:1647–1659, 1993

[12] D.K. Warland, P. Reinagel and M. Meister. Decoding visual infomation from a population of retinal ganglion cells *J. of Neurophysiology*, 78(5):2336-2350, 1997

[13] A.D. Wyner and A.J. Wyner. Improved redundancy of a version of the Lempel-Ziv algorithm. *IEEE Trans. Inform. Theory*, 35:723–731, 1995.

[14] A.D. Wyner and J. Ziv. Some asymptotic properties of entropy of a stationary ergodic data source with applications to data compression. *IEEE Trans. Inform. Theory*, 35:1250–1258, 1989

[15] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Trans. Inform. Theory*, 23:337–343, 1977

[16] J. Ziv and A. Lempel. Compression of individual sequences via variable rate coding. *IEEE Trans. Inform. Theory*, 24:530–536, 1978