Estimating Entropy Rates with Bayesian Confidence Intervals

Matthew B. Kennel
mkennel@ucsd.edu
Institute for Nonlinear Science,
University of California, San Diego,
La Jolla, CA 92093-0402, U.S.A.

Jonathon Shlens
shlens@salk.edu
Systems Neurobiology Laboratory,
Salk Institute for Biological Studies,
La Jolla, CA 92037, and,
Institute for Nonlinear Science,
University of California, San Diego,
La Jolla, CA 92093-0402, U.S.A.

Henry D. I. Abarbanel
habarbanel@ucsd.edu
Department of Physics and
Marine Physical Laboratory, Scripps Institution of Oceanography,
University of California, San Diego,
La Jolla, CA 92093-0402, U.S.A.

E. J. Chichilnisky
ej@salk.edu
Systems Neurobiology Laboratory,
Salk Institute for Biological Studies,
La Jolla, CA 92037, U.S.A.

The entropy rate quantifies the amount of uncertainty or disorder produced by any dynamical system. In a spiking neuron, this uncertainty translates into the amount of information potentially encoded and thus the subject of intense theoretical and experimental investigation. Estimating this quantity in observed, experimental data is difficult and requires a judicious selection of probabilistic models, balancing between two opposing biases. We use a model weighting principle originally developed for lossless data compression, following the minimum description length principle. This weighting yields a direct estimator of the entropy rate, which, compared to existing methods, exhibits significantly less bias and converges faster in simulation. With Monte Carlo techinques, we estimate

a Bayesian confidence interval for the entropy rate. In related work, we apply these ideas to estimate the information rates between sensory stimuli and neural responses in experimental data (Shlens, Kennel, Abarbanel, & Chichilnisky, 2004).

1 Introduction

What do neural signals from sensory systems transmit to the brain? An understanding of how neural systems make sense of the natural environment requires characterizing the language neurons use to communicate (Borst & Theunissen, 1999; Perkel & Bullock, 1968; Rieke, Warland, de Ruyter van Steveninck, & Bialek, 1997). As a step in this process, Shannon's information theory (Cover and Thomas, 1991; Shannon, 1948) provides a means of quantifying the amount of information represented without specific assumptions about what information is important to the animal or how it is represented.

Neural systems operate in a time-dependent, fluctuating sensory environment, full of spatial and temporal correlations (Field, 1987; Ruderman & Bialek, 1994; Simonicelli & Olhausen, 2001). Given these correlations, we ask how much novel information per second the neural response provides about the sensory world. This is the average mutual information rate, or the largest amount of new information per second that the brain could use to update its knowledge about the sensory world. Information rates of neural spike trains bound the performance of any candidate model of sensory representation (Bialek, Rieke, de Ruyter van Steveninck, & Warland, 1991; Borst & Theunissen, 1999; Buracas, Zador, DeWeese, & Albright, 1998; Strong, Koberle, de Ruyter van Steveninck, & Bialek, 1998; Warland, Reinagel, & Meister, 1997), without regard to how that information is encoded.

Most methods for estimating the information rate from experimental data proceed by estimating entropy rate as an intermediate step (but see Kraskov, Stogbauer, & Grassberger, 2004; Victor, 2002). The entropy rate of a dynamical or stochastic system is the rate that new uncertainty is revealed per unit of time (Lind & Marcus, 1996) and plays a central role in the theory of information transmission (Shannon, 1948). Shannon's theory showed that any transmission channel must have a capacity above this quantity in order to reproduce a signal without error. The entropy rate also gives the best possible compressed transmission rate for any lossless encoding of typical signals. In dynamical systems theory, the entropy rate (in particular, Kolmolgorov-Sinai entropy) is the principal quantification of the existence and amount of chaos (Gilmore & Lefranc, 2002; Hilborn, 2000; Ott, 2002).

Estimating the entropy rate from observed data like spike trains can be surprisingly difficult in practice. The classical definitions of entropy rate do not lead easily to a reliable and accurate estimator, given only an observed data set of finite size. Entropy rate estimators nearly always assume some underlying probabilistic model for the observed data, although this aspect is often not explicitly recognized. A reliable rate estimator requires addressing two issues:

- Accurately estimating entropies from observed data
- Selecting the appropriate model for time-correlated dynamics

Although the first issue has been vigorously investigated, the second has not. Even with sophisticated methods for entropy estimation (Costa & Hero, 2004; Miller & Madow, 1954; Nemenman, Shafee, & Bialek, 2002; Paninski, 2003; Roulston, 1999; Strong et al., 1998; Treves & Panzeri, 1995; Victor, 2000, 2002; Victor & Purpura, 1997), the final estimate of the rate (as opposed to estimating block entropies) can be dominated by a human-derived choice of some underlying modeling parameter (Schurmann & Grassberger, 1996; Strong et al., 1998). These heuristics are subjective and make constructing a confidence interval for the rate difficult.

By explicitly addressing the model selection problem, we have designed an estimator for entropy rate that requires no heuristic decisions, contains no important free parameters, and uniquely provides a Bayesian confidence interval about its estimate. We follow a statistical principle inspired by data compression (Rissanen, 1989) to weight a mixture of models appropriate for a finite time series (Kennel & Mees, 2002; London, Schreibman, Hausser, Larkum, & Segev, 2002; Willems, Shtarkov, & Tjalkens, 1995). These "context-tree" models may be used to estimate many quantities, but here we concentrate on Bayesian estimators of entropy (Nemenman et al., 2002; Wolpert & Wolf, 1995), which are combined to yield a direct estimator of entropy rate. We demonstrate through simulation that these estimates are consistent, exhibit comparatively low bias on finite data sets, outperform common alternative procedures, and provide confidence intervals on the estimated quantities. We calculate confidence intervals about the estimate using a numerical Monte Carlo method (Gamerman, 1997).

Although motivated by problems in neural coding, the algorithm presented here is a general estimator of entropy rate for any observed sequence of discrete data. In a related work, we extend these techniques to estimate the mutual information rate from experimentally observed neural spike trains (Shlens, Kennel, Abarbanel, & Chichilnisky, 2004).

This letter proceeds by reviewing the classical and Bayesian estimators of entropy as well as the common issues in extracting entropy rate from entropies. Motivation for the context tree modeling method for a time series and its justification follows. Next, we show the application of the model to entropy rate estimation and empirical results. Finally we discuss potential extensions and limitations for this method of estimating entropy rates, as well as current applications in neural data analysis.

2 Entropy and Entropy Rate

2.1 Classical and Bayesian Estimators of Entropy. We begin by outlining a Bayesian procedure for estimating the entropy and its associated confidence interval. The Shannon entropy of a discrete probability distribution $P = \{p_i\}, i \in [1, A]$,

$$H(P) = H(\{p_i\}) = -\sum_{i=1}^{A} p_i \log p_i, \tag{2.1}$$

quantifies the uncertainty represented by P, or the average number of yes-no questions needed to determine the identity i of a random draw from P (Cover & Thomas, 1991; MacKay, 2003; Shannon, 1948).[1] Although equation 2.1 provides a definition of entropy, estimating this quantity (and P) well from finite data sets of observations presents significant statistical and conceptual challenges, which are far greater when estimating the entropy rate.

Assume that one observes N independent draws of a discrete random variable from some unknown underlying distribution with alphabet A. The count vector $\mathbf{c} = [c_1, \ldots, c_A]$ accumulates the observed occurrences of each symbol. The naive entropy estimator assigns the observed frequencies for the probabilities $p_j = c_j/N$,

$$\hat{H}_{\text{naive}}(\mathbf{c}) = \sum_{j=1}^{A} -\frac{c_j}{N} \log_2 \frac{c_j}{N}. \tag{2.2}$$

This estimator yields the correct answer as $N/A \to \infty$, but in many practical cases, this estimator is significantly biased. The first-order correction (Miller & Madow, 1954; Roulston, 1999; Victor, 2000),

$$\hat{H}_{\text{MM}}(\mathbf{c}) = \sum_{j=1}^{A} -\frac{c_j}{N} \log_2 \frac{c_j}{N} + \frac{A-1}{2N} \log_2 e, \tag{2.3}$$

still retains significant bias when $N \approx A$ or $N \ll A$ (Paninski, 2003).

An alternative approach is a Bayesian estimator of entropy (Wolpert & Wolf, 1995). Consider a hypothetical probability distribution $\boldsymbol{\theta}$, which is a candidate for P. Each component θ_j is a guess for the true probability parameter p_j and, accordingly, must be $\sum_{j=1}^{A} \theta_j = 1$. Consider the probability

[1] All information-theoretic quantities in this letter use base 2 logarithms to provide units of bits.

of drawing one symbol j assuming the underlying distribution is $\boldsymbol{\theta}$. By definition, this is θ_j. The likelihood of drawing the particular set of empirically observed counts $\mathbf{c}$ is the standard multinomial,

$$P(\mathbf{c}|\boldsymbol{\theta}) = \frac{1}{Z}\prod_{j=1}^{A}(\theta_j)^{c_j}, \tag{2.4}$$

where Z is a normalization.

A Bayesian entropy estimate averages the Shannon entropy of $\boldsymbol{\theta}$, $H(\boldsymbol{\theta}) = -\sum_j \theta_j \log\theta_j$, over the relative likelihood that $\boldsymbol{\theta}$ might actually represent the truth given the observed counts $P(\boldsymbol{\theta}|\mathbf{c})$:

$$\hat{H}_{\mathrm{Bayes}}(\mathbf{c}) = \int H(\boldsymbol{\theta})P(\boldsymbol{\theta}|\mathbf{c})d\boldsymbol{\theta}. \tag{2.5}$$

By Bayes' rule, we take $P(\boldsymbol{\theta}|\mathbf{c}) \propto P(\mathbf{c}|\boldsymbol{\theta})P(\boldsymbol{\theta})$ and recast the estimation as

$$\begin{aligned}\hat{H}_{\mathrm{Bayes}}(\mathbf{c}) &= \int H(\boldsymbol{\theta})P(\mathbf{c}|\boldsymbol{\theta})P(\boldsymbol{\theta})d\boldsymbol{\theta}.\\ &= \iint H\,\delta\,[H - H(\boldsymbol{\theta})]\,P(\mathbf{c}|\boldsymbol{\theta})P(\boldsymbol{\theta})d\boldsymbol{\theta}\,dH\\ &= \int H\,P(H|\mathbf{c})\,dH.\end{aligned} \tag{2.6}$$

A disadvantage of a Bayesian approach is that one needs a somewhat arbitrary prior distribution, $P(\boldsymbol{\theta})$, on the parameters of the distribution $\boldsymbol{\theta}$. We discuss the common choice in appendix A. Nemenman et al. (2002) investigated the properties of $\hat{H}_{\mathrm{Bayes}}$ in the modest to small N/A limit. In this regime $\hat{H}_{\mathrm{Bayes}}$ is dominated by the particular prior $P(\boldsymbol{\theta})$ chosen, not the observations, meaning that the estimate does not reflect properties derived from actual data. They suggest an interesting meta-prior as a correction. In our application, this is not a concern because N/A is typically large when we estimate $\hat{H}_{\mathrm{Bayes}}$, and hence we do not apply this correction.

There is a conceptual point to consider here. Which estimate should be used if all observations occur in a single bin? An argument can be made that in this circumstance, all of these estimators ($\hat{H}_{\mathrm{Bayes}}$, $\hat{H}_{\mathrm{MM}}$) should be bypassed in favor of an estimate of zero, for example, define $\hat{H}_{\mathrm{zero}} = 0$ for deterministic data, $\hat{H}_{\mathrm{zero}} = \hat{H}_{\mathrm{Bayes}}$ or $\hat{H}_{\mathrm{MM}}$ otherwise. The idea is that if the underlying physics reliably produces the same value, then there is probably some underlying mechanistic principle preventing anything else. For instance, in neural data absent of spikes (or completely silent), a better estimate might be to use $\hat{H}_{\mathrm{zero}}$ rather than the small but positive values that $\hat{H}_{\mathrm{MM}}$ or $\hat{H}_{\mathrm{Bayes}}$ would yield. An alternative perspective is that although all observations have been the same value, the future and underlying truth is not necessarily so; thus, $\hat{H}_{\text{Bayes}}$ or $\hat{H}_{\text{MM}}$ would be appropriate. Choosing which estimator is appropriate is an externally directed prior on the expected structure of observations. In empirical testing, using $\hat{H}_{\text{zero}}$ improves bias on some physical systems we have considered. For the results we present later, the effect is very small.

2.2 Bayesian Confidence Intervals for $\hat{H}_{\text{Bayes}}$. To make inferences from real data, we need a range of results that appear compatible with the data rather than a single-point estimate of the entropy. Because all estimates are subject to statistical fluctuation, comparisons between any two quantities require comparing any difference to reasonable statistical fluctuations. We need an "error bar" on our estimate, but from experiment, we have only a single data set.

Again, we choose the Bayesian perspective and ask what the likelihood is of an estimated entropy given the observed data, $P(H|\mathbf{c})$. A variety of underlying distributions could have been sampled to generate the observed data. Each such distribution has a greater or lesser compatibility with the data (relative likelihood), and we weight their entropies accordingly. The width of the posterior distribution $P(H|\mathbf{c})$, around its mean, $\hat{H}_{\text{Bayes}}$, is a Bayesian measure of the uncertainty of the estimate. Wolpert and Wolf (1995) computed $\hat{H}_{\text{Bayes}}$ and the variance of $P(\hat{H}_{\text{Bayes}}|\mathbf{c})$ analytically using the same prior as ours.

We want to go beyond variance and find confidence intervals. We call the central portion of $P(H|\mathbf{c})$ the Bayesian confidence interval of our estimate. The Bayesian form is sometimes called a credible interval, distinguishing it from frequentist confidence intervals, which have a different definition. Specifically, we find quantiles of $P(H|\mathbf{c})$—those locations where the cumulative distribution achieves some value $0 < \alpha < 1$. With $C(H) = \int_{-\infty}^{H} P(H'|\mathbf{c})\, dH'$, a quantile H_α is defined as the value where $C(H_\alpha) = \alpha$. Then, for example, a 90% confidence interval is $[H_{0.05}, H_{0.95}]$. Roughly, we might say the observed data have been produced by a distribution whose entropy falls within the interval with 90% likelihood.[2]

Unlike variance, the quantiles for entropy are not known analytically, but they can be estimated using a Markov chain Monte Carlo (MCMC) technique (Hastings, 1970; Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953). The MCMC algorithm produces a succession of vectors $\boldsymbol{\theta}^k$, $k = 1, \ldots, N_{\text{MC}}$, which sample $P(\boldsymbol{\theta}|\mathbf{c})$. By taking the entropy function of each $\boldsymbol{\theta}^k$, we obtain N_{MC} individual samples, $H_k = H(\boldsymbol{\theta}^k)$, from the distribution $P(H|\mathbf{c})$. We calculate $\hat{H}_{\text{Bayes}}$ to be the mean of this empirical distribution approximating equation 2.6 and the Bayesian confidence interval to be the

[2] Note that $\hat{H}_{\text{Bayes}}$ is the mean, distinct from the median, $H_{0.5}$, and the mode, the maximum likelihood estimate.

Table 1: Examples of Entropy Estimates (Bits) ± Estimate of Standard Deviation.

	(5,5)	(50,50)	(1,9)	(1,99)
$\hat{H}_{\text{naive}}$	1.000 ± 0.000	1.000 ± 0.000	0.47 ± 0.27	0.081 ± 0.065
$\hat{H}_{\text{MM}}$	1.070 ± 0.000	1.007 ± 0.000	0.54 ± 0.27	0.088 ± 0.065
$\hat{H}_{\text{Bayes}}$	0.937 ± 0.082	0.993 ± 0.010	0.51 ± 0.24	0.105 ± 0.068

Note: The top row shows count vectors $\mathbf{c} =$ [# of heads, # of tails] for a binary alphabet.

quantiles $[H_{0.05}, H_{0.95}]$ of the empirical distribution. The number of samples, N_{MC}, is a user-controlled, free parameter chosen solely to balance Monte Carlo fluctuation against computation time. We typically use $N_{\text{MC}} = 199$ so that the the 5% and 95% confidence interval values can be read off directly as the 10th and 190th smallest values in a sorted list of H_k. We outline the specific prior, the analytical formula for $\hat{H}_{\text{Bayes}}$, and concrete implementation of the MCMC algorithm in appendix A.

2.3 Comparing $\hat{H}_{\text{naive}}$, $\hat{H}_{\text{MM}}$, and $\hat{H}_{\text{Bayes}}$. We demonstrate the behavior of $\hat{H}_{\text{naive}}$, $\hat{H}_{\text{MM}}$, and $\hat{H}_{\text{Bayes}}$ on varying hypothetical count frequencies to show the properties of the estimators. Suppose we count the probability of flipping a tail in a biased coin. Table 1 shows results. If equal counts are observed, say, $(5, 5)$ and $(50, 50)$, the naive estimator gives entropy of exactly one. The classical bias-corrected version $\hat{H}_{\text{MM}}$ gives value larger than one, which is impossible for the entropy of any binary distribution. This is because the bias correction is calculated in a "frequentist" philosophy. This means that one considers the observed frequencies to reflect a distribution approximately close to truth, and one then pretends one can simulate new finite sets of observations from these probabilities. It is well known that simulating finite data from a distribution gives on average a spikier empirical distribution than truth, leading to lower entropy with a naive estimator. Thus, the naive estimator is deemed to be biased down, and the upward correction is applied to give $\hat{H}_{\text{MM}}$. The same philosophy can be applied to estimate a standard deviation (Roulston, 1999). Unfortunately, for equal counts, this estimate is identically zero, which is nonsensical as well.

$\hat{H}_{\text{Bayes}}$ does not show these pathologies. The posterior distribution of likely values has support only in the valid region, and the width of the distribution is always sensible. For nonequal counts (e.g., $(1, 9)$ and $(1, 99)$) the estimators behave more similarly. Table 1 demonstrates how $\hat{H}_{\text{Bayes}}$ may be both larger and smaller than $\hat{H}_{\text{MM}}$ in various circumstances. (See Wolpert & Wolf, 1995, for more demonstrations of properties of $\hat{H}_{\text{Bayes}}$.)

2.4 Entropy Rate. The entropy rate, as opposed to entropy, characterizes the uncertainty in a dynamical system. It is a particular asymptotic limit of the entropy of a time series. Unfortunately, as we see below, there exist major empirical quantitative problems in trying to successfully estimate the entropy rate from observed time series. Resolving these problems is the subject of our investigation.

We proceed from discrete distributions to time series of discrete symbols. This stream of integers might be a spike train, with the integer being the number of spikes in a small time interval (MacKay & McCulloch, 1952; Strong et al., 1998); a discretized interspike interval (Rapp, Vining, Cohen, Albano, & Jimenez-Montano, 1994); an arbitrary symbolic dynamical system (Lind & Marcus, 1996); or even a computer file on a hard drive (Cover & Thomas, 1991). The resulting time series of integers, $R = \{r_1, r_2, r_3, \ldots\}$, where the subscript indexes time, is termed a symbol stream with alphabet size A. The underlying process that created the time series is called a symbol source.[3] In a symbolic dynamical system, a characteristic of the underlying symbol source is the uncertainty of the next symbol (Cover & Thomas, 1991; Lind & Marcus, 1996). In other words, the uncertainty of the next symbol is an intensive property of a dynamical system. We briefly explore the significance of this property in a symbolic system.

With an independent and identically distributed (i.i.d.) symbol source, the uncertainty of the next symbol in R is simply the entropy of the discrete distribution, $H(\{r_i\})$. In contrast to the i.i.d. case, many real dynamical systems exhibit serial correlations due to internal state variables and time dependence. Consider the refractory dynamics of a spiking neuron, or inertia in a mechanical system. Knowledge of recently observed symbols or states alters the estimate of the next observation.

To capture these dynamics, consider the distribution of words, length-D blocks of successive symbols from the source, defining the block entropy

$$H_D \equiv H(\{r_{i+1}, \ldots, r_{i+D}\}),$$

which quantifies the average uncertainty in observing any pattern of D consecutive symbols. Block entropy is normally extensive in the thermodynamic sense, increasing linearly with D for sufficiently large D (e.g., the heat capacity of a solid increases with the amount of matter in question).[4] The quantity that characterizes the underlying source is, however, the new uncertainty per additional symbol. Dividing H_D by D gives an intensive quantity, which reflects a characteristic of the underlying system (e.g., the specific heat is a property of the substance). H_D/D will approach the asymptotic limit from above, as increasingly long words reveal more potential interactions.

[3] By assumption, the symbol stream and underlying source are stationary and ergodic.

[4] The statistical mechanics community is showing increasing interest in complex systems, often defined as having a substantial subextensive term (Bialek et al., 2001), often at the critical point between complete order and conventional chaos.

The entropy rate is defined by three equivalent asymptotic limits (Cover & Thomas, 1991),

$$
\begin{aligned}
h &\equiv \lim_{D\to\infty} H_D/D && (2.7)\\
&\equiv \lim_{D\to\infty} H_{D+1} - H_D && (2.8)\\
&\equiv \lim_{D\to\infty} H(r_{i+1}|r_i, r_{i-1}, \ldots, r_{i-D}), && (2.9)
\end{aligned}
$$

where h has units of bits per symbol or possibly bits per second. The limit $D \to \infty$ ensures that we account for all possible temporal correlations or historical dependence.

In the following section we highlight existing methods on entropy rate estimation that have focused on definitions 2.7 and 2.8. Subsequently, we return to definition 2.9 to derive a new estimator of entropy rate.

2.5 Estimating Entropy Rates with Block Entropy. Even with a sophisticated estimator of entropy, it is not trivial to calculate an entropy rate. Definitions 2.7 and 2.8 suggest strategies for estimating the entropy rate by first making an estimate $\hat{H}_D$ (Schurmann & Grassberger, 1996) for a range of D. The difficulty with these approaches is that two competing biases make the final estimation step of the rate a qualitative judgment with minimal rejection criteria (Treves & Panzeri, 1995), which forgoes any attempt at calculating confidence intervals on the estimated quantities. One result of this situation is that bias can significantly contaminate the estimate, and variances are underestimated (Miller & Madow, 1954; Nemenman et al., 2002; Paninski, 2003; Roulston, 1999; Treves & Panzeri, 1995).

We illustrate this situation in simulation. Consider estimating the entropy rate by calculating $\hat{H}_D$ at varying depths D. Estimating the entropy rate amounts to selecting a particular word length D^* and calculating $\hat{h}_{\text{block}} = \hat{H}_{D^*}/D^*$. Given infinite data, selecting the appropriate word length is trivial as the asymptotic behavior $\lim_{D\to\infty} H_D/D$ guarantees an accurate estimate of $\hat{h}$ for large D, where all temporal dependencies in the symbol stream have been accounted for.[5]

However, for finite data, selecting an appropriate D^*, that value where $\hat{H}_{D^*}/D^*$ is deemed to be the best estimate of the rate, is not trivial because a second but (typically) opposing bias complicates the procedure. Discrete estimators of entropy are often dominated by negative bias at large enough D due to undersampling. Thus, the observed distribution is spikier and appears to have lower entropy than truth. Estimating $\hat{h}$ by selecting a word length must contend with these two competing biases in finite data sets:

[5] The same qualitative picture arises by plotting $\hat{H}_{D+1} - \hat{H}_D$. Empirically, this estimate converges faster to the entropy rate (in D) but is prone to large errors due to statistical fluctuations (Schurmann & Grassberger, 1996).

1. **Negative bias:** At large word lengths, undersampling naive probability estimators typically produces downward bias due to Jensen's inequality (Paninski, 2003; Treves & Panzeri, 1995).
2. **Positive bias:** At small word lengths, finite D in H_D/D produces upward bias due to insufficiently modeled temporal dependencies (Cover & Thomas, 1991).

Given this situation, the common assumption is that there exists an intermediate region in D, between the realms of the two biases, to provide a plateau effect. The plateau suggests a converged region in which one can select a word length to successfully estimate $\hat{h}_{\text{block}}$. Unfortunately, as seen in Figure 1, the distinctiveness, or even existence, of a plateau depends highly on the number of data and the temporal complexity of the underlying symbol source. In experimental data from an unknown symbol source, the location of the plateau, if any, is unknown. In this approach, the rate estimate is dominated by the selection of the region of word length used, which is determined by a qualitative judgment of the slopes of Figure 1.

Another similar criterion for selecting an appropriate intermediate range of word lengths is to follow the direct method (de Ruyter van Steveninck, Lewen, Strong, Koberle, & Bialek, 1997; Reinagel & Reid, 2000; Strong et al., 1998). The convergence of block entropies to the entropy can be decomposed into extensive and subextensive terms,

$$h = \frac{H_D}{D} + \frac{f(D)}{D},$$

where $f(D)$ is defined as a monotonic function of word length that grows at less than a linear rate. For large enough D, the leading subextensive term is assumed to be constant. In practice, the data analyst plots H_D/D versus $1/D$ and assumes there is some region of the plot with a linear slope such that :

1. D is large enough so that $f(D) \approx k$.
2. D is small enough so that $\hat{H}_D$ is not contaminated by sampling bias.

Strong et al. (1998) give some weak bounds based on coincidence counting to help with the second criterion, but they are often not tight enough to be empirically effective. As before, this entropy rate estimate requires a choice based on a subjective assessment of the slope of block entropy plots. Rather than rely on human-directed estimation, we offer an approach that, in effect, automatically selects the word lengths appropriate for finite data sets. We place this problem in a larger framework, akin to probabilistic model selection.

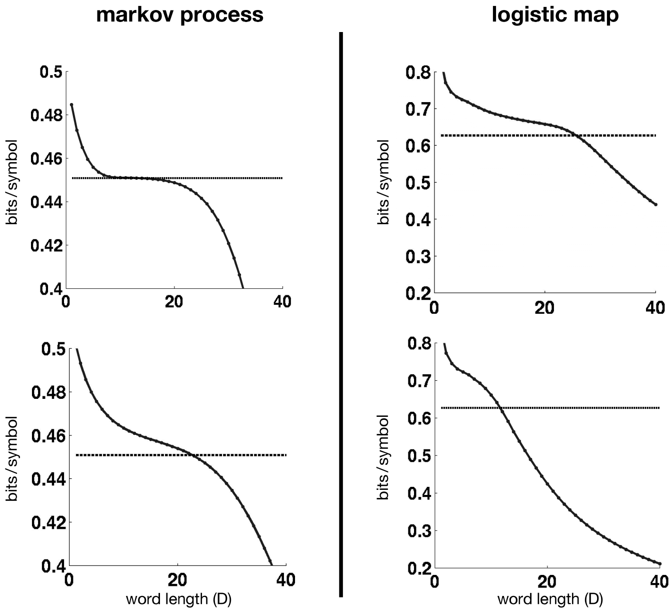

Figure 1: Estimating the entropy rate using $\hat{h}(D) = \hat{H}_D/D$ or $\hat{H}_{D+1} - \hat{H}_D$ for a simple two-symbol Markov process and a symbolized logistic map. The horizontal dashed line is the true entropy rate calculated analytically (see section 5). $\hat{H}_D$ is calculated using $\hat{H}_{\text{MM}}$ with $O(10^6)$ and $O(10^5)$ symbols in the top and bottom rows, respectively. Estimating h with $\hat{h}(D)$ becomes difficult as the plateau disappears with fewer symbols and greater temporal complexity (e.g., logistic map) in the underlying information source.

2.6 Model Selection in Entropy Rate Estimation. Selecting a plateau in the block entropies amounts to selecting a word length D^* at which we believe we have accounted for all temporal dependencies in the underlying dynamics. Implicit is a probabilistic model with A^{D^*} parameters for the distribution of words that accounts for the temporal dependencies. In other words, successive (nonoverlapping) words of length D^* are assumed to be statistically independent. Block entropies of order D^* model the spike train as independent draws from a distribution of alphabet size A^{D^*}. This relationship suggests an equivalent topology for retaining frequency counts for all words—a suffix tree, as diagrammed in Figure 2. This data structure is often used in practice for accumulating frequency distributions to estimate block entropies. Each node in the suffix tree retains the counts for a particular word, and all the nodes at a depth D retain $\hat{P}(r_{i+1}, \ldots, r_{i+D})$. Selecting a plateau prunes a suffix tree at a single depth D^*, selecting the structure necessary to model the data (Johnson, Gruner, Baggerly, & Seshagiri, 2001).

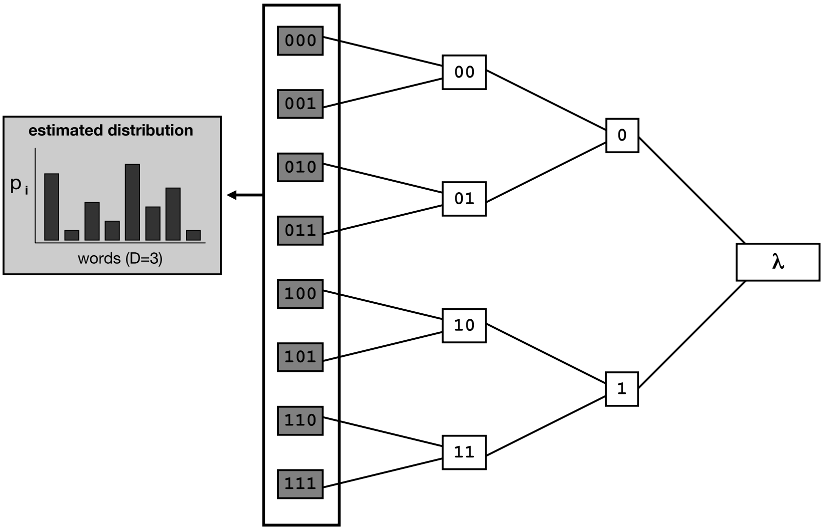

Figure 2: A suffix tree of maximum depth 3 modeling a binary symbol stream. Each node retains counts of the number of observations of the word corresponding to the node. The root node λ counts the number of symbols observed in the stream. Children of a node prepend one additional symbol earlier in time. The counts at all nodes for $D = 3$, when normalized, provide $\hat{P}(r_{i+1}, \ldots, r_{i+3})$.

Given a fixed amount of data, can we select a tree topology with a more principled, less qualitative criterion? In the following sections, we make this probabilistic model more explicit for entropy rate estimation using definition 2.9. We show techniques to select an appropriate tree topology for a finite data set.

3 Modeling Discrete Time Series

3.1 Markovian Modeling. Our approach to estimating the entropy rate directly is to focus on the conditional entropy formulation $h \equiv \lim_{D\to\infty} H(r_{i+1}|r_i, \ldots, r_{i-D})$. This is a time-series model-building approach, as we make explicit estimates for the distribution of the next symbol. A classical example in the continuous case is an autoregressive model whose future is a function of previous values of fixed order. We confine our work, though, to discrete processes.

Consider a first-order Markov chain with states $\sigma \in \Sigma$, each of which has some transition probability distribution $P(\mathrm{r}|\sigma)$ for emitting some symbol r $\in A$ and a deterministic transition to a new state σ'. This Markov chain is a stationary symbol source, and its entropy rate is

$$h = \sum_{\sigma} H\left[P(\mathrm{r}|\sigma)\right] \mu(\sigma), \tag{3.1}$$

with $H[\cdot]$ denoting the Shannon entropy (see equation 2.1) and $\mu(\sigma)$ the stationary probability of state σ.[6] This formula is exact for a Markov chain (Cover & Thomas, 1991). In our problem, we must estimate the set of states and transition probabilities from data.

A Markov model–based entropy rate estimator applies a deterministic projection from the semi-infinite, conditioned past into a finite set of states Σ. We can equate the set of finite states $\Sigma \equiv R_D$, where $R_D \equiv \{r_i, \ldots, r_{i-D}\}$ is the finite conditioning depth D. If a finite history of recent symbols uniquely determines the next symbol, then this is a Markov model, and we take equation 3.1 as the entropy rate h for the system. To estimate the entropy, we need to first select the right set of states Σ and then estimate the transition and stationary distributions. We may then calculate the entropy rate directly from this model estimated from the observation.[7]

This is a model of finite-time conditional dependence, which is distinct from the timescale of absolute correlation. To understand the difference, imagine a first-order Markov chain, which has a large probability at staying in the same state. Equivalently, consider a classical autoregressive gaussian process with a single lag. The autocorrelation in the resulting time series of these processes could be quite high at significant time delays, but the conditional correlation—how much history is necessary to make probabilistic predictions—remains just one time step.

3.2 Context Trees. The Markov formulation does not by itself make the problem of the selection of word length go away. In a naive view, there are still up to A^D conditioning states with D-order conditional dependence. One could imagine using a statistical model selection criterion to select some intermediate optimal D^*, balancing statistics and model generality. We adopt a more flexible model, however. Instead of just a single depth D^*, giving all states with histories of length D^*, we adopt a variable-depth tree. At any time, the history of recent symbols can be projected down to the longest matching suffix in the tree, called a terminal node. Our model is an elaboration of a suffix tree, called a context tree, where each node also

[6] Technically, this assumes the Markov chain is sufficiently mixing, with one strongly connected component. *Mixing* means that the influence of dynamics long in the past has dissipated. *Strongly connected* means that eventually all states, even in a multistable system, are visited.

[7] *Directly* means that that no block entropies $\hat{H}_D$ need be calculated as an intermediate step. Rather, we calculate $\hat{h}$ in one step using equation 3.1.

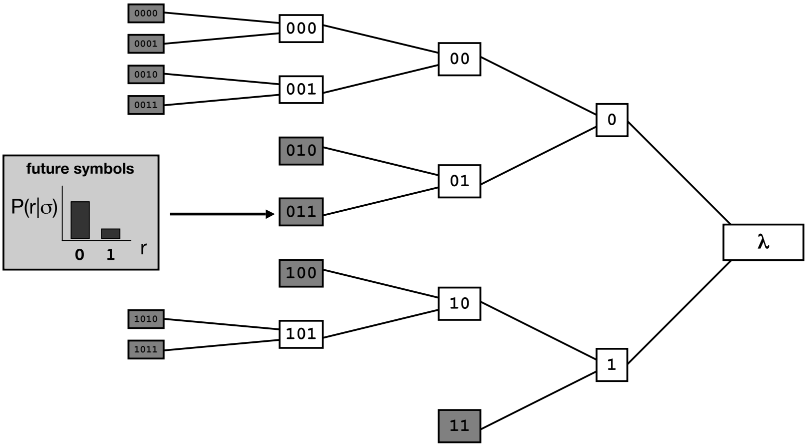

Figure 3: A context tree for a binary symbol stream. Each terminal node retains a distribution of future symbols that occurred after the word corresponding to the node. This distribution is an explicit representation of emitting the additional symbol, which, along with a buffer of recently emitted symbols, determines the transition to a new state.

stores probability distributions for emitting one future symbol.[8] Figure 3 diagrams how these terminal nodes are akin to the Markov conditioning states. The root node λ corresponds to the empty string, and children of any node correspond to strings with one additional symbol prepended as an earlier symbol in time. The stochastic process underlying a context tree is called a tree machine, and its entropy rate can also be given by equation 3.1 (see Kennel and Mees, 2002).

In summary, the conditional definition of entropy rate 2.9 implies a Markov model, where the conditioning word length is the order of the Markov process. The generalization of a Markov model to a context tree provides a more flexible framework, where the conditioning word length need not be a single value (e.g., D^*) but is variable, specific to each conditioning state. The complexity of the model is specified not by the word length but rather by the tree topology. Compared to a fixed-order Markov model, a context tree can better adapt to the dependencies of an underlying source while maintaining statistical precision. There now exist two remaining questions to address.

- How do we select the appropriate tree topology (addressed in sections 3.3 through 3.6)?
- How do we estimate the quantities in equation 3.1 (addressed in section 4)?

[8] Kennel and Mees (2002) detail specific computational issues and optimizations in implementing a context tree.

3.3 Modeling with the Minimum Description Length Principle. Selecting an appropriate tree topology is a statistical inference problem (Duda, Hart, & Stork, 2000). If we allow greater depths and longer conditioning sequences, we can represent more complex dynamics, but in return we must estimate more free parameters. This model complexity trade-off is the same phenomenon previously seen as opposing biases in the conventional estimation via block entropies.

We resolve this problem by following the minimum description length (MDL) principle. We detour briefly to discuss how MDL addresses model selection through data compression ideas. We return to symbol streams to calculate a modified log likelihood, or code length, for any observed data. We use these code lengths to select not just a single tree topology but a weighting over all possible tree topologies, which balances these model complexity issues according to Bayesian theory. We may weight any statistic computable over nodes, or trees, derived from the underlying data. In section 4, we apply this weighting to estimate the entropy rate.

We briefly review Rissanen's MDL theory (Rissanen, 1989) as a model selection principle (also see Balasubramanian, 1997). Model selection is treated as a data compression problem and can be viewed as a particular form of Bayesian estimation. Given a transmitter (encoder), a lossless communication channel, and a receiver (decoder) that know only the overall model class (but not any specific model), what is the smallest number of bits that we need to losslessly code and decode the observed data? The MDL principle asserts that the model that requires the smallest number of bits, termed the *description length*, is the most desirable one. The description length for expressing any data set is the sum of the *code lengths* to encode the model $L(\text{model})$ and the residuals $L(\text{data}|\text{model})$. Larger numbers of model parameters or decreased predictive performance increase the respective code lengths. Models that minimize the sum $L(\text{data}|\text{model}) + L(\text{model})$ appropriately balance the predictive performance $L(\text{data}|\text{model})$ with the complexity $L(\text{model})$.

Statistical estimation in MDL selects a single best model and examines functionals on this model. Instead of estimating by selecting a model, we may also estimate by weighting (Solomonoff, 1964). Rather than selecting that one model with the lowest code length (the MDL principle), we may average over models, weighting good models more than bad ones. This is very similar to Bayesian averaging, weighting by the "posterior distribution." Given code lengths L estimated from our models and data, we weight any quantity Q by $P(\text{model}|\text{data}) \propto 2^{-L(\text{data}|\text{model})}$ to get an estimator for Q,

$$\hat{Q}_w(\text{data}) = \frac{\int Q(\text{model}) 2^{-L(\text{data}|\text{model})}}{\int 2^{-L(\text{data}|\text{model})}}. \tag{3.2}$$

This can often provide superior performance compared to choosing a single model, especially when no single model is clearly better than the others.

When one model dominates, the weighted estimator is very quickly also dominated by the single best model and gives nearly identical results to the MDL principle.

3.4 Calculating Code Lengths of Symbol Streams. The first step in applying the MDL formalism to judge the appropriate model complexity (i.e., tree topology) is to calculate the code length or the number of bits necessary to transmit losslessly a symbol stream. We assume, temporarily, that we have a sequence of symbols $\{r_1, \ldots, r_N\}$ that have no additional temporal structure. That is, these symbols are drawn independently, from a fixed probability distribution $\boldsymbol{\theta}$ over A bins. Section 2.5 described conventional estimators of entropy for this problem. Now, our task is to estimate a fair minimal code length for these data: How many bits would it take to transmit these N symbols down a channel and reconstruct them exactly at a hypothetical receiver? If $\boldsymbol{\theta}$ were known, then it would take $NH(\boldsymbol{\theta})$ bits (also the negative log likelihood). However, $\boldsymbol{\theta}$ is unknown, and the extra parametric complexity must be accounted for properly. Generally, it is not permissible to substitute an arbitrary entropy estimator $\hat{H}$ and call $N\hat{H}$ a code length.

Rissanen (1989) proposed stochastic complexity $L = -\log \int P(\mathbf{c}|\theta) P(\theta) d\theta$ as an extended log likelihood (or code length) to properly account for this model complexity. The difference of two such stochastic complexities, Rissanen's test statistic for choosing the better model, is precisely the logarithm of the Bayes factor used in Bayesian hypothesis testing and model selection (Lehmann & Casella, 1998). Frequently these integrals cannot be done analytically without making large N approximations. For small N, these approximations may result in pathological behavior such as nonmonotonicity or negative code lengths. In our application, such problems would be fatal, as smooth behavior is required down to even $N = 1$. A simple and effective prescription to obtain a fair code length and avoid such pathologies is to adapt predictive techniques used in sequential data compression (Cover & Thomas, 1991). At time step $k + 1$, make a probability estimate $\hat{\boldsymbol{\theta}}(s_{k+1})$ conditioned only on previously observed data, $s_1, \ldots, s_k$. The total code length of an entire symbol stream, is $\sum_j -\log_2 \hat{\boldsymbol{\theta}}(s_j)$. The complexity cost for coding the parameters is included implicitly because the early data are coded with poorer estimates than the later data.

One caveat to sequential prediction is that we must provide a small but finite probability for potential symbols not yet observed. Thus, we cannot predict probability zero for any symbol value, because if it should occur, it would give an infinite code length. We use the Krischevsky-Trofimov (KT) estimator (Krichevsky & Trofimov, 1981), which adds to all possible symbols a small "ballast" of weight $\beta > 0$. Given count vector $\mathbf{c}$, this estimator predicts a probability for symbol $j \in [1, A]$,

$$\hat{\theta}_{\mathrm{KT},j}(\mathbf{c}) = \frac{c_j + \beta}{\sum_k (c_k + \beta)}. \tag{3.3}$$

The free parameter $\beta > 0$ prevents estimating $\hat{\theta} = 0$, which would give an infinite code length should that symbol actually occur.[9]

We apply the KT estimator sequentially, using the counts of symbols, $\mathbf{c}^i$, which were seen through time i, that is, $\mathbf{c}^i_j = \sum_{l=1}^{i} \delta(r_l = j)$. We may sequentially code the stream with a net predictive code length,

$$L_{\mathrm{KT}}(\mathbf{c}, \beta) = \sum_{i=1}^{N} -\log_2 \hat{\boldsymbol{\theta}}_{\mathrm{KT}}(r_i|\mathbf{c}^{i-1}), \tag{3.4}$$

having seen all symbols $\mathbf{c} = \mathbf{c}^N$. It is critical that the estimator for symbol r_i uses $\mathbf{c}^{i-1}$ and not $\mathbf{c}^i$, so that a hypothetical receiver could causally reconstruct the identical $\hat{\theta}$ before seeing r_i and, given the compressed bits, decode the actual r_i. For this particular estimator 3.3, we may conveniently compute the code length equation 3.14 using only the final counts and not the sequence:

$$L_{\mathrm{KT}}(\mathbf{c}, \beta) = \log_2 \left[\frac{\Gamma(N + A\beta)}{\Gamma(A\beta)}\right] - \sum_{j=1}^{A} \log_2 \left[\frac{\Gamma(c_j + \beta)}{\Gamma(\beta)}\right]. \tag{3.5}$$

It is a particularly convenient that for the KT estimator, the sequential predictive formula can be collapsed so that the actual sequence order is unimportant. It turns out that the stochastic complexity integral of the multinomial 2.4 with the Dirichlet prior can be evaluated exactly (Wolpert & Wolf, 1995), and, moreover, the answer is identical to equation 3.5. This three-way coincidence is specific to KT and not generically true for other estimators. In general, we favor sequential predictive computation as the least risky option over analytical approximation to stochastic complexity or Bayesian integrals.

For the KT estimator, selecting a priori $\beta = 1/A$ tends to perform well empirically (Nemenman et al., 2002; Schurmann & Grassberger, 1996). The distribution of Shannon entropy of the Dirichlet prior happens to be widest with $\beta = 1/A$ as well. In the final section, we outline an optimization approach for β to eliminate this free parameter if desired.

3.5 Context Tree Weighting. Of course, real symbolic sequences are not independent and identically distributed. Our larger model class is that of a

[9] We intentionally use the same β as in equation A.1, because the KT estimate is also the Bayesian estimate of $\boldsymbol{\theta}$ with the same Dirichlet prior, $\hat{\boldsymbol{\theta}}_{\mathrm{KT}} = \int \boldsymbol{\theta} P(\mathbf{c}|\boldsymbol{\theta}) P_{\mathrm{Dir}}(\boldsymbol{\theta}) d\boldsymbol{\theta}$. Note that $\hat{H}_{\mathrm{Bayes}}$, is not the same as the estimator plugging in $\hat{\boldsymbol{\theta}}_{\mathrm{KT}}$ into the Shannon entropy formula. As $\hat{\boldsymbol{\theta}}_{\mathrm{KT}}$ always smooths the empirical distribution toward being more uniform, plugging it in would result in a larger value for entropy than the naive estimator, just like $\hat{H}_{\mathrm{MM}}$. The Bayesian estimate $\hat{H}_{\mathrm{Bayes}}$ may be larger than, smaller than, or equal to $\hat{H}_{\mathrm{naive}}$ or $\hat{H}_{\mathrm{MM}}$, depending on the observations.

context tree, as previously discussed. The issue now is which nodes of the tree (i.e., its topology) one should choose as the best, balancing statistics versus modeling ability. We choose the optimal topology using equation 3.5 as a subcomponent in the calculation.

Consider for a moment the simplest model selection problem: compare coding using no history (i.e., the root node only) versus conditioning on one past binary symbol (i.e., descending one level of the tree). Which is a better model: $P(r_i)$, or $P(r_i|r_{i-1})$? We calculate the code length $L(\mathrm{n})$ to code the conditional observations for each node. The MDL model selection criterion is to compare $L(\text{parent})$ to $\sum_c L(c)$, the sum of the children's code length. If the first is smaller, then $P(r_i)$ is the better; otherwise, the more complex model is better. Imagine that the process actually were independent. The distribution of $\mathbf{c}(\text{parent})$ would look similar to $\mathbf{c}(c)$. Usually $L(\text{parent})$ would be smaller than $\sum_c L(c)$ because only one parameter (implicitly) needs to be encoded instead of two. If, however, the distributions at the children were significantly distinguishable from the parent, then the smaller code length would go to the more complex model and the children preferred. Instead of a hard choice, we may weight our relative trust in the two models as proportional to 2^{-L} (Solomonoff, 1964), and form a weighted code length at the node that captures this weighting.

This process is continued recursively: at each node we consider the weighting between the model that stops "here" versus the composite of all models that descend from this subtree. This is the key insight of context tree weighting (CTW) (Willems et al., 1995). Considering binary trees to maximum depth D, CTW is like Bayesian weighting over 2^{2^D} possible topologies, but is efficiently implementable in time linear in the size of the input data. The original invention of CTW was as a universal sequential source coding (data compression) algorithm, whose performance may be easily bounded by analytical proof. Here, we are interested in statistical estimation, and explicit sequentiality is not required. We now outline the steps in a batch estimation of the weighted context tree and its code length:

1. Form a context tree for the entire data set, retaining at each node n the conditional future counts, $\mathbf{c}(\mathrm{n})$.
2. Compute and store, for every n, its local code length estimate, $L_e = L_{\mathrm{KT}}(\mathbf{c}(\mathrm{n}), \beta)$, with equation 3.5.
3. Compute recursively the weighted code length at each node. With the coding probability distributions defined $P_e = 2^{-L_e}$, $P_w = 2^{-L_w}$ and child nodes of n denoted as c, the fundamental formula of context tree weighting is (Willems et al., 1995)

$$P_w = \frac{1}{2}P_e + \frac{1}{2}\prod_{\mathrm{c}} P_w(\mathrm{c}). \tag{3.6}$$

Defining, for clarity, L_c to be the sum of L_w over extant children c, $L_c = \sum_{\mathrm{c}} L_w(\mathrm{c})$, we have

$$\begin{aligned} L_w(\mathrm{n}) &= -\log_2 \frac{2^{-L_e} + 2^{-L_c}}{2} \\ &= 1 + \min(L_e, L_c) - \log_2(1 + 2^{-|L_e - L_c|}). \end{aligned} \tag{3.7}$$

In practice, this involves a depth-first descent of the context tree, computing L_w for all children before doing so for the present node. If there are no children, or the number of observations at n is but one (i.e., $\sum_a \mathbf{c}_a(\mathrm{n}) = 1$), then instead define $L_w = L_e$ and stop recursion. (With only one observation, any child would have an identical code length.)

4. At the root node λ, $L_w(\lambda)$ is the CTW code length for the sequence.

For statistical modeling applications, representing local code lengths L_e, L_w with standard finite-precision floating point is sufficient, contrary to comments in London et al. (2002) indicating a need to use arbitrary precision arithmetic to store coding distributions. Note that even for $A > 2$, the factors of $1/2$ in equation 3.6 remain as is, reflecting the assumed prior distribution on context trees: that a tree terminating at a given node has an equal prior probability as all subtrees that descend from that node. As discussed briefly in Willems et al. (1995), one could also choose to weight the current node and descendants by factors γ and $1 - \gamma$ for $0 < \gamma < 1$.

3.6 Weighting General Statistics. The model selection principle behind CTW may be adapted for more general statistical estimation tasks than source coding. In particular, if we are able to evaluate some statistic $Q(\mathrm{n})$ as a local function of the particular context and the counts observed there, we can find the weighted estimator Q_w that weights the local Q values with the same weighting as used in CTW. To compute Q_w:

1. Produce the context tree from the observed sequence and compute the per node code lengths as described in section 3.5.
2. For every node, compute the local weighting factor $W(\mathrm{n})$. $W(\mathrm{n})$ is the relative weight of the current node versus the subtree of possible child nodes:

$$W(\mathrm{n}) = \frac{2^{-L_e}}{2^{-L_e} + 2^{-L_c}}. \tag{3.8}$$

If there are no children or the current node has only one observation, $W(\mathrm{n}) = 1$.

3. Evaluate the local statistic $Q(\mathrm{n})$ for each node from the counts. Compute the weighted statistic Q_w,

$$Q_w(\mathrm{n}) = W(\mathrm{n})Q(\mathrm{n}) + (1 - W(\mathrm{n}))\left(\sum_{\mathrm{c}} Q_w(\mathrm{c})\right). \tag{3.9}$$

   Like L_w this requires a recursive depth first search.

4. The final weighted statistic for the whole tree is $Q_w(\lambda)$, the value at the root node.

The weighting can also be understood conceptually as an average over all possible nodes in all possible trees:

$$Q_w = \sum_{\mathrm{n}} \mathcal{W}(\mathrm{n})Q(\mathrm{n}), \tag{3.10}$$

with $\mathcal{W}(\mathrm{n})$ the net product of all $(1 - W(\mathrm{n}))$ and $W(\mathrm{n})$ weighting factors descending from the root node to n. The operation of standard CTW is the special case, $Q = \hat{\boldsymbol{\theta}}_{\mathrm{KT}}\delta(\mathrm{n} = \text{history})$, of this general statistical weighting.

4 Estimating the Entropy Rate with Context Tree Weighting

We now present an algorithm for estimating the entropy rate following the Markov formulation (see equation 3.1) and using the weighting over all tree topologies. We diagram and outline all of these results in parallel in appendix D. As an aside, we could estimate the entropy rate using the code length directly $\hat{h}_{\mathrm{CTW}} = L_w(\lambda)/N$; however, this estimator is always biased from above, because it obeys the redundancy inherent in any compression scheme (Kennel & Mees, 2002; London et al., 2002). Instead, we emphasize that we use the code lengths (and compression technique) only to define the appropriate weighting over Markov models. This is similar in spirit to string-matching entropy estimators (Amigo, Szczepanski, Wajnryb, & Sanchez-Vives, 2004; Kontoyiannis, Algoet, Suhov, & Wyner, 1998; Lempel & Ziv, 1976; Wyner, Ziv, & Wyner, 1998), which use the same key internal quantities as the Lempel-Ziv class of compression algorithms (Ziv & Lempel, 1977), but without the additional overhead necessary to produce a literal compressed representation.

4.1 Estimating $\hat{h}$ with the Weighted Context Tree. We first fill an unbounded context tree with all observed symbols R, and find the code lengths as in section 3.5. Recall that at each node, we accumulate the counts of all future symbols $\mathbf{c} = [c_1, \ldots, c_A]$ (where we have suppressed n for notational convenience), which occurred after the string corresponding to the node.

Every node corresponds to a potential Markov conditioning state σ, with

$$\hat{P}(\mathrm{r}|\mathrm{n}) = \frac{c_\mathrm{r}}{\sum_{i=1}^{A} c_i} \tag{4.1}$$

$$\hat{\mu}(\mathrm{n}) = \frac{N(\mathrm{n})}{N} = \frac{\sum_{i=1}^{A} c_i}{N} \tag{4.2}$$

as the estimated transition and stationary probabilities, and N is the total number of symbols. The estimated transition probability $\hat{P}(\mathrm{r}|\mathrm{n})$ has an alphabet of size A and an occupancy ratio $N(\mathrm{n})/A \gg 1$ for heavily weighted nodes. This is in stark contrast to the explosion of words A^D in block entropies. Treating each node as a Markov state via equation 3.1, we estimate the entropy rate by using the local function

$$Q(\mathrm{n}) = \hat{H}[\mathbf{c}|\mathrm{n}]\hat{\mu}(\mathrm{n}), \tag{4.3}$$

where $\hat{H}$ is any estimator of the entropy from observed counts. Following section 2, we select $Q(\mathrm{n}) = \hat{H}_{\mathrm{Bayes}}(\mathbf{c}|\mathrm{n})\hat{\mu}(\mathrm{n})$ and substitute it in into the general weighting procedure of section 3.6, yielding our direct rate estimate $\hat{h}_{\mathrm{Bayes}}$ as the value of Q_w at the root. We could also use $\hat{H}_{\mathrm{MM}}$ instead of $\hat{H}_{\mathrm{Bayes}}$ to get a rate estimate $\hat{h}_{\mathrm{MM}}$, which also performs well. This does not lead as easily to a confidence interval, however, as we now discuss for $\hat{h}_{\mathrm{Bayes}}$.

4.2 Bayesian Confidence Intervals for $\hat{h}_{\mathrm{Bayes}}$. The previous section shows how to get a point estimate of the entropy rate. Now we want to find the distribution of entropy rates that seem likely from the data. For clarity, this algorithm is also summarized in appendix D.

We simulate tree topologies according to their posterior likelihood, and at each of their nodes, simulate one possible $\hat{H}_{\mathrm{Bayes}}$ as in section 2.2. Symbolically we perform a Monte Carlo simulation of the integrand of the abstractly represented

$$\hat{h}_{\mathrm{Bayes}} = \int h(\Theta, \mu) P(\Theta|T, \mathbf{C}) P(T)\, d\Theta\, dT, \tag{4.4}$$

where T represents the topology of a particular tree, Θ the union of parameter vectors at all terminal nodes of that tree, $\mathbf{C}$ the observed future counts at those nodes, and $h(\Theta, \mu)$ the Shannon entropy rate operator on Θ and the occupation probability for those nodes. The result will be samples from the integrand of equation 4.4. Its expectation value is an estimate of the entropy rate $\hat{h}_{\mathrm{Bayes}}$ and the error bars—estimated width of reasonable rates given sampling variation—from the quantiles of the empirical distribution.

We randomly draw tree topologies with relative probability equal to the implicit CTW weighting over tree topologies. Consider recursive descent down from the root node λ. At each node n, draw a random variate $\xi \in [0, 1)$. If ξ is less than the local weighting factor $W(\mathrm{n})$, then stop here: this is a terminal node for the model. Otherwise, recurse down all extant branches until all paths have terminated. This yields a topology (set of terminal nodes) of a good model, drawn with probability proportional to its weighting as implied by the CTW formulas.

At each terminal node of this subtree, draw a single sample of $P(H|\mathbf{c})$, using the MCMC procedure and the counts for that particular node. Call it $H^*(\mathrm{n})$. Combine these draws to calculate one estimated sample of the entropy rate:

$$h_i^* = \sum_{\mathrm{n}} H^*(\mathrm{n})\hat{\mu}(\mathrm{n}). \tag{4.5}$$

The sum here is over only those terminal nodes selected stochastically for this single estimate. We repeat this procedure N_{MC} times, every time drawing a new topology and set of $H^*(\mathrm{n})$ values, and using equation 4.5 to calculate h_i^* for each such draw. This gives samples drawn from the underlying $P(h|\text{data})$. Their mean is the final rate estimate,

$$\hat{h}_{\text{Bayes}} = \frac{1}{N_{\mathrm{MC}}} \sum_{i=1}^{N_{\mathrm{MC}}} h_i^*. \tag{4.6}$$

The distribution of the samples provides a confidence interval for the estimate accounting for natural statistical variation. For instance, to estimate a 90% confidence interval, sort 199 samples of h_i^*; the 10th and 190th elements in the sorted set approximate the true confidence interval. For computational simplicity, we use $\hat{\mu}(\mathrm{n}) = N_{\mathrm{n}}/N$, although a more correct μ is fully determined by the node probabilities. Appendix B discusses this issue in detail.

4.3 Asymptotic Consistency of $\hat{h}_{\text{Bayes}}$. As a data compression method, the arbitrary depth CTW method has been proven to achieve entropy for stationary and ergodic sources, meaning that $\lim_{N\to\infty} \hat{h}_{\mathrm{CTW}} \to h$ for all but infinitesimally improbable strings of length N (Willems, 1998). In statistical terms, this means that $\hat{h}_{\mathrm{CTW}}$ is an asymptotically consistent estimator of the entropy rate. We here give an argument suggesting the same is true of $\hat{h}_{\text{Bayes}}$. Both $\hat{h}_{\mathrm{CTW}}$ and $\hat{h}_{\text{Bayes}}$ can be expressed as a global sum of the form 3.10, so we may write

$$\left|\hat{h}_{\mathrm{CTW}} - \hat{h}_{\text{Bayes}}\right| = \sum_{\mathrm{n}} \mathcal{W}_{\mathrm{n}}\, \mu(\mathrm{n}) \left| \frac{L_{\mathrm{KT}}(\mathrm{n})}{N_{\mathrm{n}}} - \hat{H}_{\text{Bayes}}(\mathrm{n}) \right|.$$

Using the results in Wolpert and Wolf (1995) and equation. 3.5, we computed the large N asymptotic expansions of the difference of the local entropy estimators:

$$\lim_{N\to\infty}(L_{\mathrm{KT}}/N_{\mathrm{n}} - \hat{H}_{\mathrm{Bayes}}) = C_1\frac{\log N_{\mathrm{n}}}{N_{\mathrm{n}}} + C_2(\boldsymbol{\theta}, A, \beta)\frac{1}{N_{\mathrm{n}}} + \dots$$

with $C_1 = \beta(A-2) + \frac{1}{2}$ and $C_2(\boldsymbol{\theta}, A, \beta)$ a complicated formula depending only on the local $\boldsymbol{\theta}_{\mathrm{n}}$ and parameters. As $\mu(\mathrm{n}) = N_{\mathrm{n}}/N \leq 1$, this gives

$$\begin{aligned}\left|\hat{h}_{\mathrm{CTW}} - \hat{h}_{\mathrm{Bayes}}\right| &\leq \sum_{\mathrm{n}} \mathcal{W}_{\mathrm{n}}\left(C_1\frac{\log N}{N} + \frac{|C_2|}{N} + \dots\right)\\ &\leq C_1\frac{\log N}{N} + O\left(\frac{1}{N}\right), \qquad (4.7)\end{aligned}$$

since $\sum_{\mathrm{n}} \mathcal{W}_{\mathrm{n}} = 1$. As $\hat{h}_{\mathrm{CTW}}$ is asymptotically consistent and $\hat{h}_{\mathrm{Bayes}} \to \hat{h}_{\mathrm{CTW}}$, then $\hat{h}_{\mathrm{Bayes}}$ is asymptotically consistent. In practice, it appears that $\hat{h}_{\mathrm{CTW}}$ has more bias. Rissanen's theory (Rissanen, 1989) shows that because $\hat{h}_{\mathrm{CTW}}$ is an actual compressed code length per symbol, and unlike $\hat{h}_{\mathrm{Bayes}}$, it cannot converge to truth any faster than $\hat{h}_{\mathrm{CTW}} \to h + O(\frac{\log N}{N})$.

5 Results

We compare the performance of our estimator $\hat{h}_{\mathrm{Bayes}}$ to several other estimators of entropy rate: the direct method $\hat{h}_{\mathrm{direct}}$ (Strong et al., 1998), two string-matching based estimators $\hat{h}_{\mathrm{LZ}}$, $\hat{h}_{\mathrm{SM}}$, and a strict context tree weighting-based estimator $\hat{h}_{\mathrm{CTW}}$ (Kennel & Mees, 2002; London et al., 2002). $\hat{h}_{\mathrm{CTW}}$ and $\hat{h}_{\mathrm{direct}}$ have been discussed previously. $\hat{h}_{\mathrm{LZ}}$ is the Lempel-Ziv complexity measure based on a parsing of the input string with respect to an adaptive dictionary (Amigo et al., 2004; Lempel & Ziv, 1976). $\hat{h}_{\mathrm{SM}}$ averages the length of longest matching strings in a previous fixed-size buffer (Kontoyiannis et al., 1998). See appendix C for a complete description of these algorithms.

5.1 Testing Convergence and Bias. The first example is a source that is a simple three-state hidden Markov model. Its transition matrix is

$$M = \begin{bmatrix} 0 & 1/3 & 2/3 \\ 1/5 & 4/5 & 0 \\ 1/10 & 0 & 9/10 \end{bmatrix}, \qquad (5.1)$$

emitting a 0 if the first nonzero transition on each line is taken and a 1 if the second is taken. The entropy rate can be calculated analytically employing

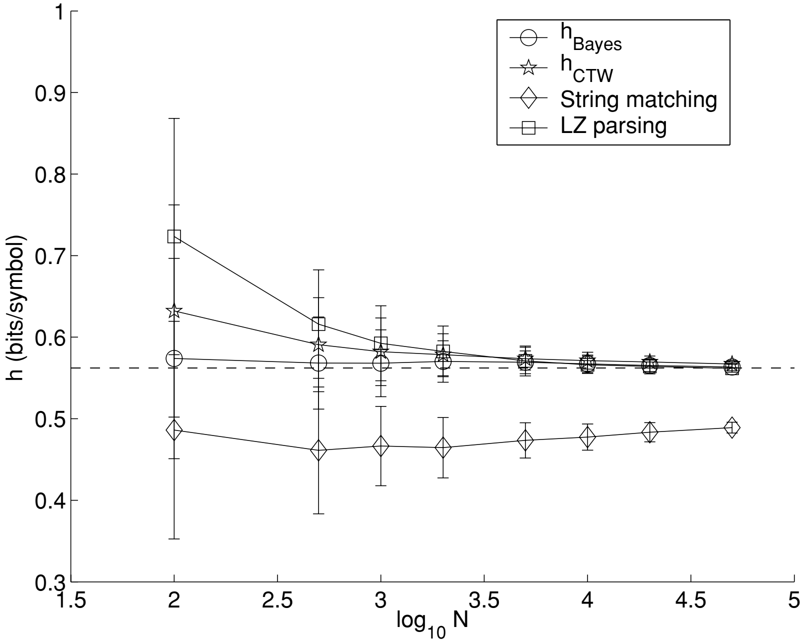

Figure 4: Convergence of several entropy rate estimates, $\hat{h}_{\text{Bayes}}$, $\hat{h}_{\text{LZ}}$, $\hat{h}_{\text{CTW}}$, $\hat{h}_{\text{SM}}$, on a simple, binary Markov process. The horizontal dashed line is the true entropy rate. $\hat{h}_{\text{Bayes}}$ converges quickest with the smallest overall bias and competitive variance. Data shown are ensembles over draws from the source, with error bars giving sample standard deviations.

(10), $h \approx 0.5623$. Figure 4 shows impressive performance of $\hat{h}_{\text{Bayes}}$, over the straight code length estimator $\hat{h}_{\text{CTW}}$ and string matching methods.

The next example is from more complex logistic map dynamics. The continuous space dynamical system is $x_{n+1} = 1 - ax_n^2$. For $a = 1.7$, the system is in a generic chaotic regime giving dynamics in $x \in [-1, 1]$. The cutoff $x = 0$ yields

$$r_n = \begin{cases} 1, & x_n \geq 0 \\ 0, & x_n < 0, \end{cases}$$

a generating partition, and thus a binary symbolization, preserving all the dynamical information from the continuous time series in the symbol stream with a known entropy rate (Kennel & Buhl, 2003). Although the continuous equation of motion is simple, the symbolic dynamics are not, with significant serial dependence among the binary symbols. In fact, it is not a Markov chain

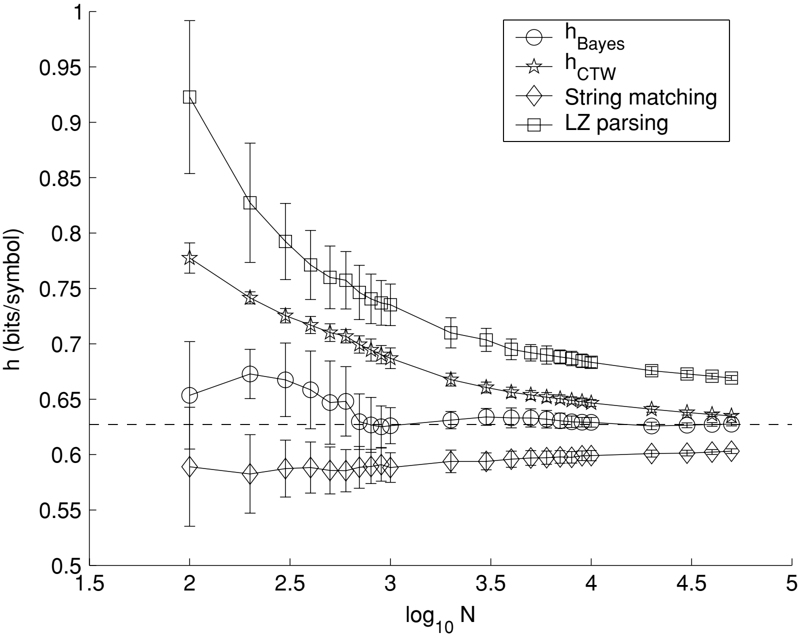

Figure 5: Convergence of several entropy estimates, $\hat{h}_{\mathrm{Bayes}}$, $\hat{h}_{\mathrm{LZ}}$, $\hat{h}_{\mathrm{CTW}}$, $\hat{h}_{\mathrm{SM}}$, on a symbolized logistic map. The horizontal dashed line is the true entropy rate. Again, $\hat{h}_{\mathrm{Bayes}}$ converges quickest with the smallest overall bias and competitive variance. Data shown are ensembles over draws from the source, with error bars giving sample standard deviations.

of any finite order and shows deep dependence, making this a challenging problem. Figure 5 shows results. Once again $\hat{h}_{\mathrm{Bayes}}$ provides a consistently more accurate estimate. One thing to notice is that whereas in the previous Markov chain example, $\hat{h}_{\mathrm{LZ}}$ was rather good and $\hat{h}_{\mathrm{SM}}$ rather poor, here their performances are reversed. In our experience, this is frequently the case. In various examples when we already knew the exact value, one of $\hat{h}_{\mathrm{LZ}}$ and $\hat{h}_{\mathrm{SM}}$ came close to the truth, and the other was quite erroneous—-but we could never predict ahead of time which one would be better, making them difficult to use for analyzing experimental data where truth is unknown. Finally, Figure 6 shows a comparison of $\hat{h}_{\mathrm{Bayes}}$ to the direct method $\hat{h}_{\mathrm{direct}}$ for the two sources. The plateau disappears for small N or more complex dynamics making estimation of $\hat{h}_{\mathrm{direct}}$ unreliable.

5.2 Testing Error Bars. We now examine the quality of the confidence intervals in the estimation of $\hat{h}_{\mathrm{Bayes}}$, that is, the quantiles of the h^* distribution, equation 4.5. The test procedure is as follows: we produce N_S time

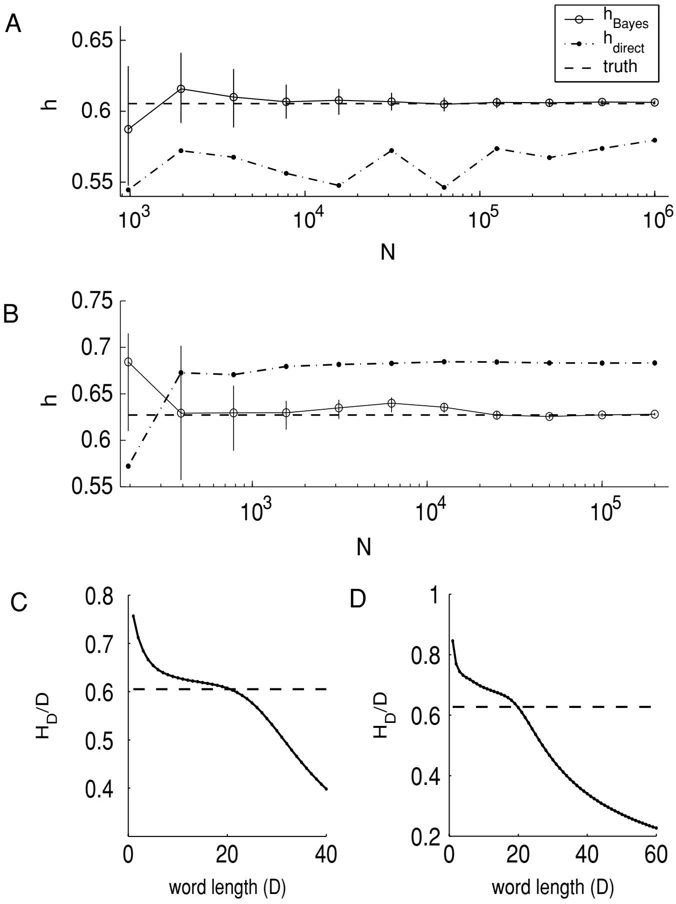

Figure 6: Comparison of $\hat{h}_{\text{Bayes}}$ (circle, solid line) to $\hat{h}_{\text{direct}}$ (dash-dot line) for (A) a simple Markov process and (B) a symbolized logistic map. The horizontal dashed line is the true entropy rate. We implemented a simple extrapolation algorithm for calculating $\hat{h}_{\text{direct}}$; however, a human-directed approach could yield a more accurate estimate. (C) Block entropy for Markov process, and (D) for logistic map N = O(10^6). The plateau effect in the block entropies $\hat{H}_D$ disappears for the more complex logistic dynamics or decreased N (number of symbols), making the estimate of $\hat{h}_{\text{direct}}$ less reliable.

series from the source and find $\hat{h}_{\text{Bayes}}(i)$ for each of them, $i = 1, \ldots, N_S$. We find the central 90% quantile of this distribution and take it as the real "error bar"—the dispersion of the statistic under repeated samplings from the source. For each time series, there is an individual confidence interval estimated by the central 90% quantile of the distribution of h^*. Ideally, the

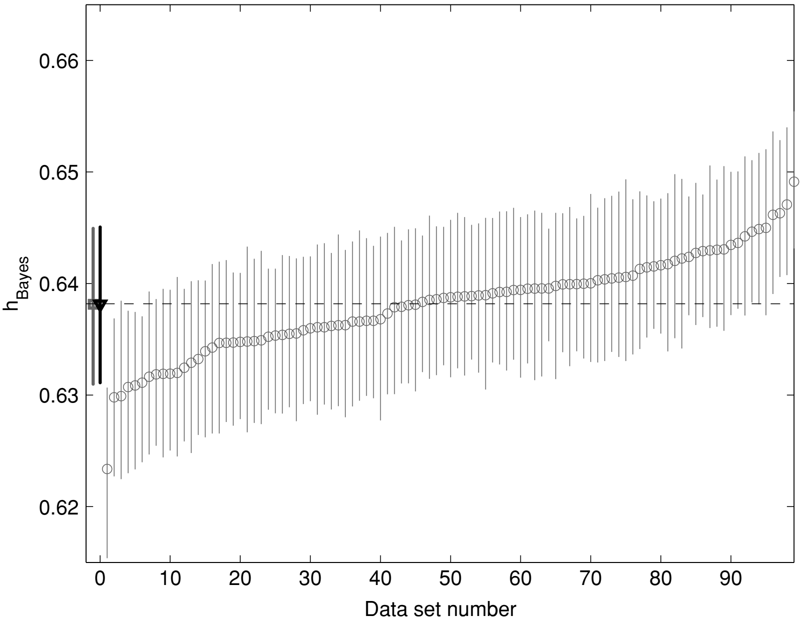

Figure 7: Testing the Bayesian confidence interval on 99 data sets of length $N = 5000$ from a pseudo-logistic source (gray circles). Average of $\hat{h}_{\text{Bayes}}$ and average of individually estimated confidence intervals (left-most triangle, black). Average of $\hat{h}_{\text{Bayes}}$ and 90% percentile on actual ensemble (left-most square, gray). The horizontal dashed line is the true entropy rate. The confidence interval is well calibrated: the ensemble average of estimators is very close to the exact value, and the average size of their confidence intervals (each estimated from a single data set) is also nearly equal to the variation in $\hat{h}_{\text{Bayes}}$ under new time series from the source.

ensemble-average size of the individual confidence intervals should be the same size as the size of the variation of the statistic under actual new samples from the source.

Our two sources with known entropy rates are the logistic map (as previously discussed) and a pseudo-logistic map, that is, a Markov chain that was estimated from a sample of logistic map symbols from a single tree model found from the method in Kennel and Mees (2002). Comparing these similar sources is instructive. Figure 7 shows results on the logistic-like Markov chain. Here, at $N = 5000$, as for most N, the estimator performs exceptionally well with almost no bias in ensemble, and estimating sampling variation accurately. Across the wide range of N, truth lies within the 90% intervals for about 90% of the samples from the source, implying the

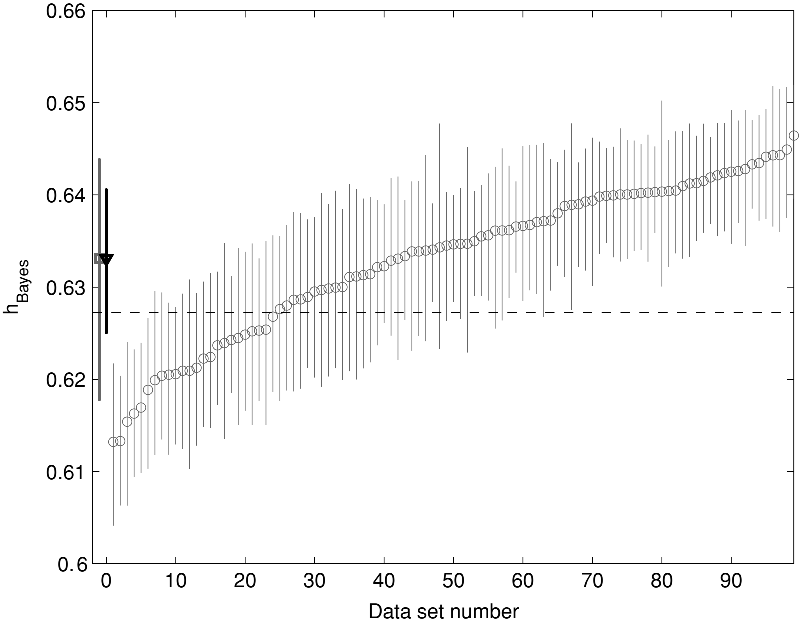

Figure 8: Testing the Bayesian confidence interval on 99 data sets of length $N = 5000$ from a true, symbolized logistic source. This figure follows Figure 7. Here, the confidence interval is not well calibrated, and the actual variation under sampling from the source is larger than the size estimated by Bayesian posterior. Note that these data are for a value of N that shows a peculiarly large relative discrepancy in Figure 10.

error bars are calibrated well. The general weighted tree model appears to estimate this Markov chain quite well.

Figure 8 shows the equivalent result for the challenging case of small N in the real logistic map. Here, the ensemble average is slightly higher than truth, but the variation in actual samples from the ensemble is larger than the size of the variation estimated by the Bayesian posterior. In other words, for a small number of samples from this more complex source, the error bar is not well calibrated and engulfs truth for roughly 60% of the samples from the source. Our estimator performs well in an absolute sense despite this mismatch. This is also evident in Figures 9 and 10, which show the similar summaries of true variation and estimated variation across a range of N. Paninski (2003) examined variance estimators for block entropy but found they could severely underestimate the true variance in the undersampled limit. By contrast, we get empirically more accurate error bars for rate, staying in a good regime by dint of estimating entropy rate directly with the model selection criteria.

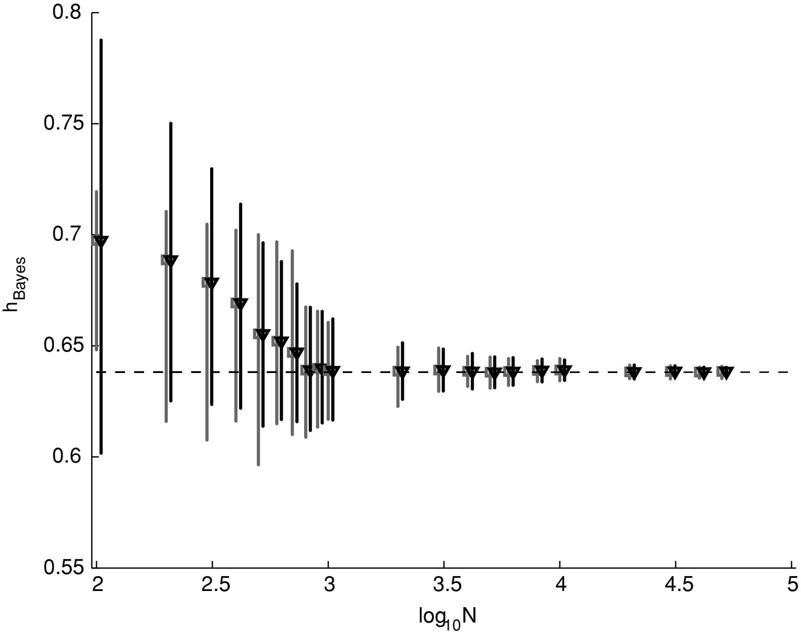

Figure 9: Convergence of the Bayesian confidence interval on the pseudo-logistic source: 90% percentile of $\hat{h}_{\mathrm{Bayes}}$ distribution under sampling from source (gray squares) and average from posterior estimation (black triangles). The horizontal dashed line is the true entropy rate. For all N, the average confidence interval engulfs the exact value, and with increasing N, the estimated error bar size matches the actual size accurately.

Controlling for the discrepancy seen in Figure 10 is necessary in experimental data in which truth is unknown. What we think to be happening here is that for some samples from the source, the tree weighting detects statistically significant deep contexts, but for other data sets (more similar to the Markov approximation), the typical depth is comparatively lower. As a result, a finite bias remains, and the variation in the entropy rate on sampling from the true source is larger than expected under the weighted mixture of Markov models. We feel that no model-based statistical estimation procedure can ever be immune to this effect. As a philosophical principle, it is impossible for an estimator to guess outside its model assumptions about what might happen in other, unseen data. Whatever it can use to guess—a model class, estimated parameters, and one observed data set—is already accounted for in its Bayesian posterior distribution. This mismatch can be greater with small amounts of data and complex information sources. We thus ask how we can discern from single trials whether we are within this regime. We suspect the result shown here may represent a rather difficult

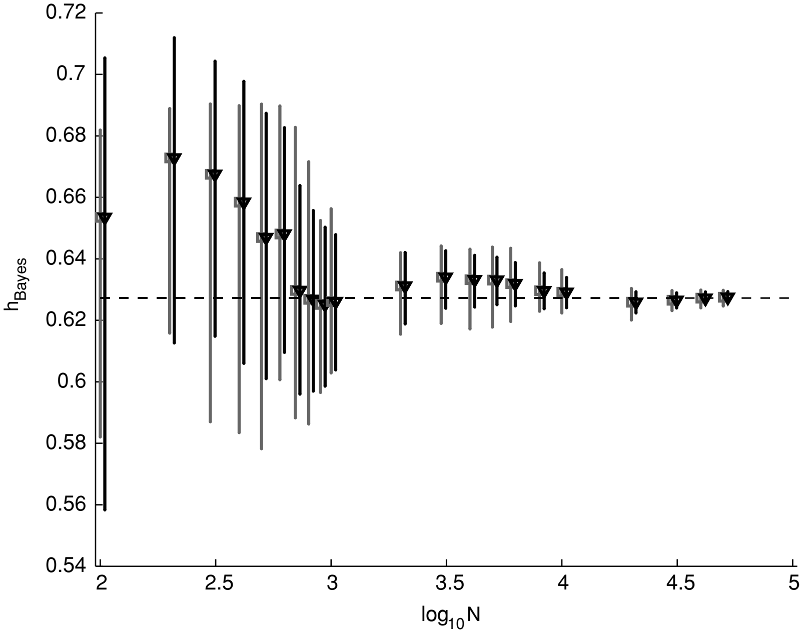

Figure 10: Convergence of the Bayesian confidence interval on the true symbolized logistic map. Symbols are the same as in Figure 9. The estimated confidence interval size is smaller than the true size until N is large.

case: we believe that symbolic dynamics of a zero-noise deterministic chaotic system are harder to estimate (on account of more deep, significant contexts) than what would typically occur in neural data. Furthermore, we highlighted a specific value of N where the mismatch is particularly large.

5.3 Diagnostics. If structural mismatch is sometimes unavoidable, are there signs when it is lurking? If it is, then the statistical user might imagine that there remains some positive bias and that truth might be in the unlikely tails of the posterior (outside the Bayesian confidence interval) more often than nominal. Imagine the situation when the underlying source has a complex structure but the length of data is limited. For arbitrarily short data sets, the method will not choose models with a large number of parameters (i.e., a large word length) because that usually leads to overfitting. As more data are seen, more complex models will be justifiable, which typically exhibit a lower entropy rate. If adding still more data does not seem to increase the effective complexity further, then one could think that the estimated tree at that point is a good, convergent model. If we were estimating only a single tree topology, it would be easy to judge convergence of complexity and topology: the number of terminal nodes times $A-1$.

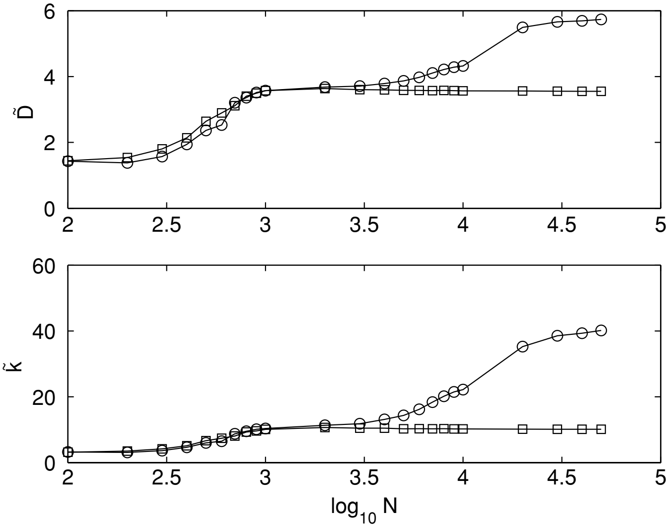

Figure 11: Convergence of diagnostics over number of symbols (N) for the pseudo-logistic source (squares) and the true symbolized logistic map (circles). (Top) $\langle\tilde{D}\rangle$, expectation of average effective depth over samples from the source. (Bottom) $\langle\tilde{k}\rangle$, expectation of estimated number of parameters (following the Rissanen ansatz) over samples from the source.

For the weighted trees, we offer two statistics that quantify the effective depth and the complexity. We define the effective depth, $\tilde{D}$, using the general formula 3.9 with $Q(\mathrm{n}) = D(\mathrm{n})\hat{\mu}(\mathrm{n})$, where $D(\mathrm{n})$ is the depth of node n. $\tilde{D}$ is the effective, average conditioning depth or word length used for prediction.

Rissanen's theory (Rissanen, 1989) provides an alternate, more interesting complexity measure. Asymptotically in N, the description length scales as

$$L = -\log P(\hat{\Theta}) + \frac{k}{2}\log N + O(k), \tag{5.2}$$

where P is the likelihood at the maximum likelihood solution and k is the effective number of free parameters. We identify L with the overall net CTW code length $L_w(\lambda)$, and $-\log P(\hat{\Theta}) \approx Nh$. Since h is unknown, we substitute one of our low-bias estimators (e.g., $\hat{h}_{\mathrm{Bayes}}$) and invert to solve for $\tilde{k}$:

$$\tilde{k} = 2(L_w(\lambda) - N\hat{h})/\log N. \tag{5.3}$$

Figure 11 shows ensemble averages of $\tilde{D}$ and $\tilde{k}$ for the two sources. For smaller N, they track very well, but for middle ranges of N, the statistics for the pseudo-logistic source have converged, but the ones for true logistic data continue to increase, especially $\tilde{k}$, as expected. For the logistic-like Markov source, $\tilde{k}$ asymptotes at about 10, precisely the number of terminal nodes in the Markov model used as a source. Given but a single data set, one would not observe as smooth a rise, of course, but even still, the lack of convergence in $\tilde{D}$ or $\tilde{k}$ with N is suggestive of structural mismatch. The prescription from the diagnostics is to collect more data, and if that is not possible, consider the possibility that the estimated confidence intervals might not engulf the truth because of modeling-induced bias.

6 Discussion

6.1 Extensions. There remain several opportunities for improvement and future investigations. Of course, estimating mutual information rates of driven input-output systems is a key problem in neuroscience. We address this in a subsequent work by conditioning on the stimulus similar to Strong et al. (1998), although conditioning on the stimulus can be done any way that is feasible. If the stimulus history were on a small, discrete set, then it could be included in the context tree history as well as past observations. Another large issue revolves around selecting the appropriate embedding or representation for a dynamical system (Kennel & Buhl, 2003)—in particular, for a spike train (MacKay & McCulloch, 1952; Rapp et al., 1994; Victor, 2002). There is no reason to suppose a priori that a fixed-width spike count discretization, though commonly used, is the best, as compared to, for example, some kind of discretized interspike interval representation.

After selecting the appropriate embedding, the issue of the proper choice for alphabet size A and the Dirichlet parameter β remains. In this work, we assumed that A represented the nominal alphabet size—that is, the total cardinality of all possible symbols of the space in question. In simple cases, this is known by construction. For example, the total cardinality is known a priori in a spike train discretized so that no more than A spikes can ever occur in a bin. In this case, outside physical knowledge constrains the alphabet to size A. There are cases, however, where the true A is excessively large or potentially unknown. The cardinality of observed symbols m might be substantially less than A (Nemenman et al., 2002; Paninski, 2003). A concrete example would be with a large multichannel recording of k distinct neurons. With only a spiking or nonspiking representation, the A defined in a general outer product sense would be 2^k, but far fewer combinations might ever be observed in practice. We believe that more sophisticated representations and approaches are likely to be necessary in this case, with the code-length-based ideas proving useful for weighting among representations as well as histories.

There are some very simple two-part coding approaches that may suffice if the nominal A is not excessively large. Conceptually, the transmission sequence consists of sending the alphabet identity information at each node before the symbols are encoded in the reduced alphabet. Assuming uniform prior on m, it takes $\log_2 A$ bits to specify m and then $\log_2 \binom{A}{m}$ to specify which of them occur, so that the code length may be expressed as $\log_2 A + \log_2 \binom{A}{m} + L_{\mathrm{KT}}(m)$ with $L_{\mathrm{KT}}(m)$ the KT code length over the m actually observed symbols ($L_{\mathrm{KT}}(1) = 0$). In the context tree, note that the A in this expression is actually the value of m for each node's parent, because the identity of used symbols at a child is a subset of the parent. For the root, of course, it is the true nominal A.

The text compression community has dealt with the unknown A issue frequently and has devised numerous solutions. A typical approach is to mix regular conditional estimators (over the nonzero alphabet) with some extra "escape" probability to allow for the situation that a new symbol may have occurred. The escape and ordinary probabilities are defined conditionally for sequential estimation in a variety of ways. Shtarkov, Tjalkens, and Willems (1995) demonstrated a zero-memory estimator whose leading asymptotic term is proportional to m and not A. Nemenman (2002) and Orlitsky, Santhanam, and Zhang (2004) described estimators for the potentially unbounded alphabet case. All of these considerations would go into the zero-memory local code length estimator, modifying equations 3.3, and 3.5. These options may require actual sequential computation for the local code length and may not have a clean batch expression like equation 3.5.

For moderate A, another option is optimization of net code length over a varying β. There needs to be a penalty for varying β, $L_\beta = -\log_2 P(\beta)$ over some assumed prior on β (e.g., $P(\beta) = A^2 \beta e^{-A\beta}$, an arbitrary choice that has a maximum at $\beta = 1/A$). The total code length, $L_T(\beta) = L_{CTW}(\beta) + L_\beta$ is computed over β, and following the MDL criterion, the minimizing β^* is located and $\hat{h}_{\mathrm{Bayes}}$ evaluated there. The corresponding Bayesian weighting, an alternative to minimizing, is (Solomonoff, 1964)

$$\hat{h} = \frac{\int \hat{h}(\beta) 2^{-L_T(\beta)} d\beta}{\int 2^{-L_T(\beta)} d\beta}. \tag{6.1}$$

This differs conceptually from the use of alphabet-adaptive zero-memory estimators, because here there is but a single β chosen for all nodes, whereas the other estimators would adapt to the actual alphabet observed at each node of the tree, which will induce some additional overhead. Of course, local optimization over β is possible on each node, though here, unlike the global optimization case, careful attention to discretization issues for the free parameter would be necessary (Rissanen, 1989). Figure 12 shows the effect of varying β on the discretized logistic map ($A = 2$). $\hat{h}_{\mathrm{CTW}} = L_{CTW}/N$ has a modest minimum around $\beta \approx 0.5$. The effect on $\hat{h}_{\mathrm{Bayes}}$ is even smaller still for a wide plateau; thus, for this data set with a small alphabet, this extension has little benefit beyond the heuristic $\beta = 1/A$. See Nemenman et al. (2002) and Nemenman, Bialek, and de Ruyter van Steveninck (2004)

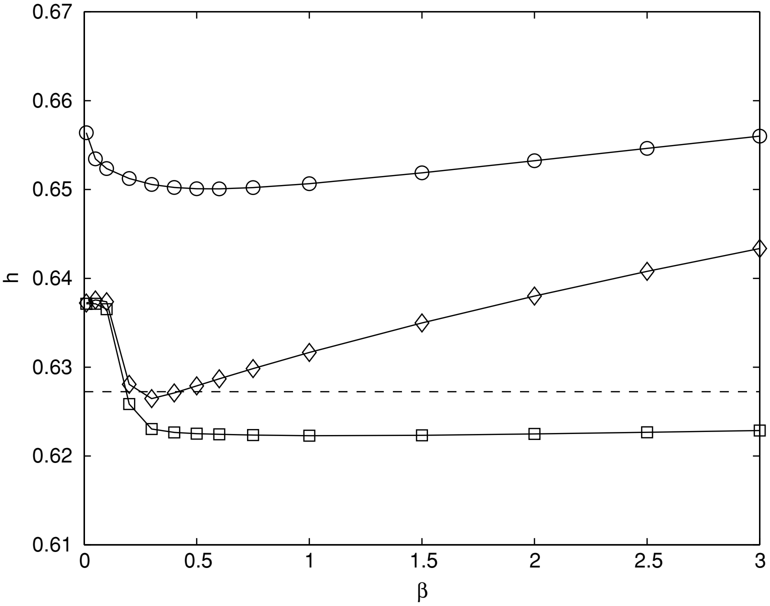

Figure 12: Truth (dashed line), $\hat{h}_{\text{CTW}}$ (circles, upper curve), $\hat{h}_{\text{Bayes}}$ (diamonds, medium curve), $\hat{h}_{\text{Bayes}}$ utilizing $\hat{H}_{\text{zero}}$ (squares, lower curve) over a range of β on 5000 symbols from the logistic map as previously described. The fact that the diamond curve comes closest to truth around $\beta = 1/2$ versus squares is coincidence; for other samples from the source and for other data set sizes, the situation is often reversed. The fact that $\hat{h}_{\text{Bayes}}$ depends less on β when utilizing $\hat{H}_{\text{zero}}$ is a general phenomenon.

for a discussion in substantially larger A and an interesting estimator that also averages over β in a different way from equation 6.1.

6.2 Limitations. It is important to recognize beforehand when the method may give suboptimal results. Since we make explicit time-series models of the observed data, we might encounter difficulties when the model scheme cannot efficiently predict the data using the short-term context tree history scheme. The method is based on universal compression, so that eventually, with sufficient data, it will find conditional dependencies, but the number of data necessary may be excessively large. The CTW weighting prefers smaller and shallower trees (fewer parameters) over deep ones. If the important dependencies are very deep, they may not contribute effectively to a proper estimate. The result would be a positive bias because the estimated tree is too shallow.

Consider a spike train discretized with an extremely small bin size (e.g., 1 ns), such that the number of symbols between individual spikes is large. The CTW tree would put most model weighting near the root, in effect viewing the spike train as a nearly Poisson process. This would occur because the context tree would not discern enough statistically significant dependence to overcome the complexity penalty. One might diagnose this if, at coarser bin sizes, the estimated entropy rate diverged from a Poisson assumption.

Another situation may be multiscale dynamics, where a slow variable modulates the dynamics of a fast variable. Without additional hints, the tree would be sensitive only to the fast dynamics. An approach would be to define some discretized slow variable appropriately averaged over the original data and allow it to occur at some level or levels in the tree as context. If it induced a lower net code length, it would be deemed preferable by MDL considerations.

More exotic dynamical systems with high complexity may present difficulties to the present method as well as all other general-purpose entropy rate estimation methods. For example symbol sources with a substantial power law subextensive term $f(D)$ would exhibit this pathology (Bialek, Nemenman, & Tishby, 2001). In these cases, deep contexts may have substantial conditional influence on the future, and only a small subset of the underlying state space may have been observed. This behavior may be nearly indistinguishable in experimental practice from ordinary nonstationarity in a time series. Such dynamics can have either zero or positive Shannon or Kolmogorov-Sinai entropy rate, but the present estimator would likely overestimate the rate for both circumstances.

A canonical example of this case is the symbolic dynamics of the logistic map at the parameter boundary between periodicity and chaos, where the true entropy rate is zero, but it may take a large data set to notice this. Perturbing the output of that process by flipping a few bits (in our case, with probability $p = 0.02$) gives a new process with the entropy of that Poisson process, but far higher complexity than a point Poisson process. Figure 13 shows results of our estimator and the direct method (Strong et al., 1998) using a block entropy estimator of Grassberger (2003). For ordinary extensive chaos with $h > 0$, one typically expects to find a region of linear behavior in $\hat{H}_D$ versus $1/D$ and extrapolate to $1/D \rightarrow 0$, but this does not happen at the boundary of chaos. On both of these data sets, there are two apparently good scaling regions, leading to different entropy estimates, with the Ma bound not providing an objective way to choose one from the other (though $N/2^{H(D)}$ does stay reasonably large as $D \rightarrow \infty$ for the no-noise logistic map and does not for the noisified set). Our estimator, $\hat{h}_{\text{Bayes}}$, automatically and correctly diagnoses the zero entropy case from the positive entropy case. For the latter, there is a noticeable positive bias, as is expected on this difficult process, which has a very long effective memory.

On the positive side, recent theoretical results have shown that a particular context tree compression scheme has an asymptotic rate of convergence

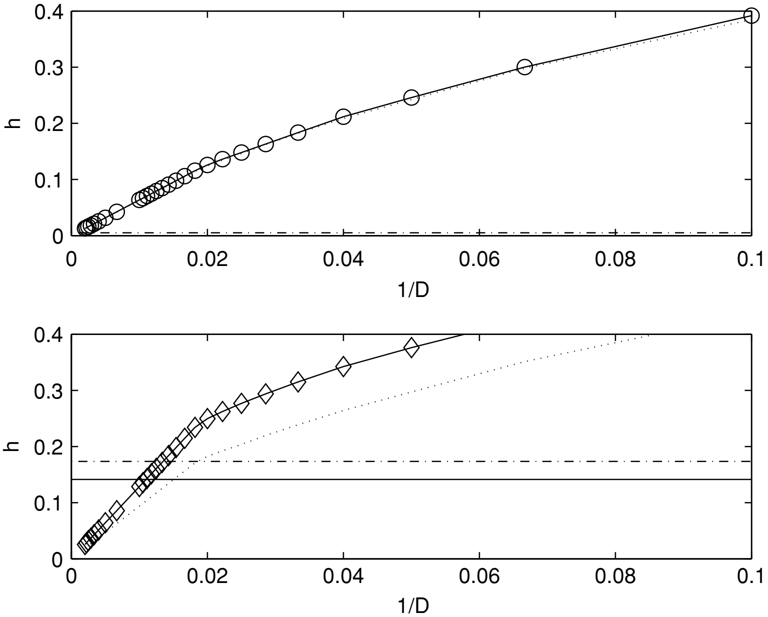

Figure 13: (Upper) Logistic map at accumulation point, $N = 100,000$. True $h = 0$. $\hat{H}_D/D$ versus $1/D$ (circles), Ma lower bound (dotted line), $\hat{h}_{\text{Bayes}}$ (dash-dotted line). The direct method shows two apparently good scaling regions—one for larger D, which extrapolates to $h \approx 0.05$, and the other to zero. (Lower) Logistic map at accumulation point, perturbed by Poisson process with $p = 0.02$. True $h \approx 0.1414$ (solid line). $\hat{H}_D/D$ (diamond), $\hat{h}_{\text{Bayes}}$ (dash-dotted line). Again, there are apparently two good scaling regions: the upper leads to close to the true entropy and the lower, extrapolating to zero. The Ma bounds (dotted lines) do not help constrain the proper scaling region for the direct method. $\hat{h}_{\text{Bayes}}$ gets in the right vicinity, though in the lower plot, there is a positive bias as expected.

in compression rate, which is as good as any other method for almost all strings (Martin, Seroussi, & Weinberger, 2004). While the actual rate of convergence depends on the character of the source, this result does suggest that one cannot do any better than context trees for compression as a class. J. Ziv (personal communication, 2004) conjectured the same would be found for most practical context tree methods such as CTW. It is not clear whether this result also applies to entropy rate estimation, but it gives confidence that the modeling approach is likely to be generally successful. We note that modeling, as in estimating probabilities of finite strings, is a distinct task from entropy estimation, and as entropy is a nonlinear function of the underlying distribution, different models might be good for dynamics estimation versus entropy estimation. Accurately modeling probabilities of words of variable length is a harder task. The present method does in fact attempt to make models, but note that minimizing the average compression rate (the implied loss function for choosing the tree weightings) is not as difficult as bounding prediction error on specific strings, and as the target of average compression rate in universal compression is bounded by entropy rate, the conceptual difference may not be so large here. Recall that in the classical case, the entropy estimate depends only on the histogram of occupancies (Nemenman et al., 2004; Paninski, 2003), and the specific labels attached to these may be shuffled without changing the entropy estimate. In our method, once a tree is estimated, the specific labels attached to the contexts designating their history may also be shuffled without changing the entropy rate estimate.

6.3 Conclusions. We have discussed how to calculate the entropy rate in any observed, symbolic dynamical system. We highlighted difficulties with estimating entropy rates using traditional estimation techniques and presented a new approach to estimating entropy rates, following the rationale of selecting a word length D more rigorously. Specifically, we examined how to generate a more appropriate model for a symbol stream by using a Markovian assumption and following the minimum description length principle to select the appropriate model structure and complexity. This probabilistic model provides a framework for sampling the Bayesian posterior distribution, $P(\hat{h}_{\text{Bayes}} \mid \text{data})$, estimating the range of possible models and their entropy rates that could reasonably give rise to our data. We thus can provide Bayesian confidence intervals on the entropy rate of a discrete (symbolic) time series.[10] In simulation, our estimator outperforms other forms of entropy rate estimation, such as algorithms based on string matching and strict CTW compression, showing much lower bias and competitive variance. For well-converged situations, the Bayesian confidence interval appears nearly exact, matching the true posterior distribution of the potential entropy rates.

In the neural coding literature, substantial effort has focused on estimating block entropies well in the undersampled regime (Nemenman et al., 2004, 2002; Paninski, 2003; Strong et al., 1998; Treves & Panzeri, 1995; Victor, 2002; Victor & Purpura, 1997); however, little work has approached rate estimation from the perspective of model selection. Although several papers have applied ideas from compression to estimate the entropy rate (Amigo et al., 2004; Kontoyiannis et al., 1998; Lempel & Ziv, 1976), these string-matching estimators have followed the spirit of coincidence detection (Ma, 1981) and equipartition principles from coding theory, not explicit model

[10] Fortran 95 source code as well as a MATLAB interface for both Bayesian entropy rate and information rate estimation is available online at http://www.snl.salk.edu/~shlens/info-theory.html.

selection. One previous work (London et al., 2002) did employ the CTW's compression ratio itself, $\hat{h}_{\text{CTW}}$, as an estimator for analyzing neural data.

In related work, we extend these methods to estimate the information rate, with a Bayesian confidence interval, in real neural data. By extracting a probabilistic model for the spike train conditioned on stimuli, we can sample the posterior distribution of average and specific mutual information rates associated with a neuron (Butts, 2003; DeWeese & Meister, 1999). The reliability of our estimator, coupled with Bayesian confidence intervals, increases our confidence in the validity of estimates of information rates in neural data and offers a method to compare these quantities across temporal resolutions, stimulus distributions, and correlations in neural responses (Brenner, Strong, Koberle, Bialek, & de Ruyter van Steveninck, 2000; Lewen, Bialek, & de Ruyter van Steveninck, 2001; Liu, Tzonev, Rebrik, & Miller, 2001; Reinagel & Reid, 2000; Rieke, Bodnar, & Bialek, 1995; Schneidman, Bialek, & Berry, 2003).

Appendix A: Markov Chain Monte Carlo for Bayesian Entropy Estimation

In this appendix we outline a numerical method to obtain samples from the integrand of equation 2.6. We select for $P(\boldsymbol{\theta})$ the Dirichlet prior,

$$P_{\text{Dir}}(\boldsymbol{\theta}) = \frac{1}{Z'}\delta\left(\sum_j \theta_j - 1\right)\prod_j (\theta_j)^{\beta-1}, \tag{A.1}$$

which smoothly parameterizes between the prior uniform on $\boldsymbol{\theta}$ ($\beta = 1$) and the maximum-likelihood situation ($\beta = 0$). Except for the limiting cases of β, this prior does not have an intuitive motivation, but it does allow analytical computation of some integrals, explaining its historical use in classical Bayesian statistics as a "conjugate prior" for the multinomial distribution (see equation 2.4). Wolpert and Wolf (1995) analytically computed the expectation in equation 2.6,

$$\hat{H}_{\text{Bayes}} = \sum_i \frac{c_i + \beta}{N + A\beta}\left[\psi(N + A\beta + 1) - \psi(c_i + \beta + 1)\right], \tag{A.2}$$

with $\psi(z) = d/dz \log_2 \Gamma(z)$. We are interested in the distribution of the integrand, the posterior distribution of entropy, having observed a set of counts.

Markov chain Monte Carlo is a numerical algorithm to draw samples from a probability distribution, particularly useful for estimating

$$\langle F \rangle = \int F(\boldsymbol{\theta})P(\boldsymbol{\theta})d\boldsymbol{\theta} \tag{A.3}$$

for some function F and probability distribution P. We refer the reader to a standard reference for MCMC (Gamerman, 1997) and focus on our specific implementation. In our problem,

$$P(\boldsymbol{\theta}) = P(\mathbf{c}|\boldsymbol{\theta}) P_{\mathrm{Dir}}(\boldsymbol{\theta})$$
$$F(\boldsymbol{\theta}) = H(\boldsymbol{\theta}) = -\sum_j \theta_j \log \theta_j,$$

where we have avoided complicated normalizations. A key advantage is that MCMC works even if P is not normalized. In many Bayesian problems, a numerator of P is easy to compute, but the normalization may be analytically intractable.

The inputs are $\mathbf{c}$ and β, and the output is a succession of H_k, which samples from the posterior distribution $P(H \mid \mathbf{c})$. Define $N = \sum c_j$.

1. Set $\boldsymbol{\theta}^0 = \mathbf{c}/N$. Set $i = 1$. Set $k = 1$.
2. Randomly draw a candidate $\boldsymbol{\theta}'$. For $j = 1, \ldots, A - 1$ and $\sigma_j = \frac{1}{N}\left(\min(c_j, N - c_j) + \frac{1}{2}\right)^{1/2}$, set

   $$\boldsymbol{\theta}'_j = \boldsymbol{\theta}^{i-1}_j + \sigma_j \mathcal{N}(0, 1),$$

   with $\mathcal{N}(0, 1)$ a gaussian random variate of zero mean and unit variance. Set $\theta'_A = 1 - \sum_{j=1}^{A-1} \theta'_j$.
3. If any element of $\boldsymbol{\theta}'$ is < 0 or >1, then reject it a priori.
4. Otherwise, draw a uniform random variate $\xi \in [0, 1)$ and test the likelihood ratio $R = P(\boldsymbol{\theta}')/P(\boldsymbol{\theta}^{i-1})$. If $\xi \leq \min(R, 1)$, then accept $\boldsymbol{\theta}'$; otherwise, reject it.
5. If $\boldsymbol{\theta}'$ is accepted, set $\boldsymbol{\theta}^i = \boldsymbol{\theta}'$; otherwise $\boldsymbol{\theta}^i = \boldsymbol{\theta}^{i-1}$.
6. If ($i \bmod 100) = 0$, then record $H_k = H(\boldsymbol{\theta}^i)$ and increment k.
7. Increment i by 1. Go to step 2 unless N_{MC} values of H_k have been recorded.

A key requirement for proper MCMC estimation is that the Markov chain of successive $\boldsymbol{\theta}^i$ be sufficiently well mixing such that a reasonable-length finite sample can serve to well approximate the true integral. The parameter value of 100 (found to be good for $A = 2$ or $A = 3$) in the penultimate step is arbitrary and should be increased if the mixing is insufficient. The central freedom is defining the distribution of the candidate $\boldsymbol{\theta}^*$ perturbations in step 2. For our problem, we have found empirically that this specific implementation produces good results for evaluating integrals or estimating quantiles in the Bayesian posterior distribution.

Appendix B: Computing μ from Node Probabilities

Equation 4.5 is a rough, first-order approximation of a more accurate estimate. Suppose we have estimated a tree topology with given terminal nodes, and at each node, we have the emission probabilities $\boldsymbol{\theta}$. Knowing the active contexts and symbol emission probabilities provides the transition probabilities as well. Theoretically, the stationary density μ is completely determined from those two quantities by the Chapman-Kolmogorov equation of detailed balance (Gamerman, 1997). If the set of terminal nodes and allowable transitions happen to define a first-order Markov chain, then the stationary density can be computed as an eigenvector problem (Cover & Thomas, 1991). This is not guaranteed. An arbitrary tree is not necessarily a first-order Markov chain, and computing its stationary μ can be more difficult.

One solution is to employ the algorithm of Kennel & Mees (2002) to temporarily expand the tree topology until it is a first-order chain. At that point, the stationary μ is computable by the eigenvector method.

Another solution is to iterate the detailed balance equations for the tree machine. Starting with $\mu(\mathrm{n}) = N(\mathrm{n})/N$, the tree can evolve density to density until the μ converges numerically to a stationary distribution. When a transition is not strictly first order, the relative probabilities of appropriate past states are estimated using the previous iterate's μ.

Both of these solutions make the computation of h_i^* significantly more intricate. We have implemented these more accurate methods for finding μ, but in our empirical investigation, the effect on the estimate, in the ensemble average, appears to be minimal. Thus, for computational simplicity, we usually use $\mu(\mathrm{n}) = N(\mathrm{n})/N$.

Appendix C: String Matching Entropy Estimation

This estimator is used in section 5. With time index i and integer n, define Λ_i^n as the length of the shortest substring starting at s_i that does not appear anywhere as a contiguous substring of the previous n symbols $r_{i-n}, \ldots, r_{i-1}$. Consider sequentially parsing a string from first to last. At some cursor location i, compute Λ_i^i, which is the length of the "phrase", advance the cursor by that amount, and repeat until the entire string is parsed into M phrases. An entropy rate estimate for stationary and ergodic sequences is (Lempel & Ziv, 1976)

$$\hat{h}_{\mathrm{LZ}} = \frac{M}{N} \log_2 N,$$

measuring entropy in bits. Amigo et al. (2004) examined the performance of this estimator and found it compared favorably to the method of Strong et al.

(1998) for small data sets and was substantially superior to the better-known data compression method in Ziv & Lempel (1978).

The string matching estimator of Kontoyiannis et al. (1998) is similar to the Lempel-Ziv estimator, but averages string match length at every location instead of skipping by the length of the found phrase:

$$\hat{h}_{\mathrm{SM}} = \left(\frac{1}{n} \sum_{i=1}^{n} \Lambda_i^n \right)^{-1} \log_2 n.$$

To implement C with N observed symbols, we first remove a small number Δ of symbols off the end and then split the remaining into two halves. String matching begins with the first element of the second half, $(N - \Delta)/2 + 1$, and examines the previous $n = (N - \Delta)/2$ characters. The length Δ excess padding is necessary to allow string matches from the end locations of the match buffer. Δ is presumed to be a few times longer than the expected match length, $\langle \Lambda \rangle \approx \log n / h$.

Appendix D: Overview of Method

Figure 14 provides a block diagram of the entire procedure for estimating the entropy rate of any discrete (symbolic) time series. The upper gray box details how to estimate an appropriate weighting of probabilistic models, while the lower gray box outlines how to estimate the range of likely entropy rates from this model. The curved boxes represent calculations, and squared boxes represent procedural aspects.

1. β is an arbitrary parameter for the Dirichlet prior. Selecting $\beta = 1/A$, where A is the alphabet size, works well in practice, but see section 3.4 for a complete discussion.

2. Although section 3.2 discusses how to build a context tree, many optimizations exist to ease their computational tractability (Kennel & Mees, 2002).

3. Recursively descend the tree, and calculate equation 3.5 at each node.

4. Beginning at tree leaves, recursively ascend the tree and sum the code lengths of all children c at each node $L_{\mathrm{c}} = \sum_{\mathrm{c}} L_w(\mathrm{c})$.

5. Recursively descend the tree to calculate equation 3.8 at each node.

6. Draw a single sample ξ from a uniform probability distribution [0, 1].

7. Descend down the tree from the current node to the next level, and select a single child to examine. All remaining nodes are placed on the stack for future processing.

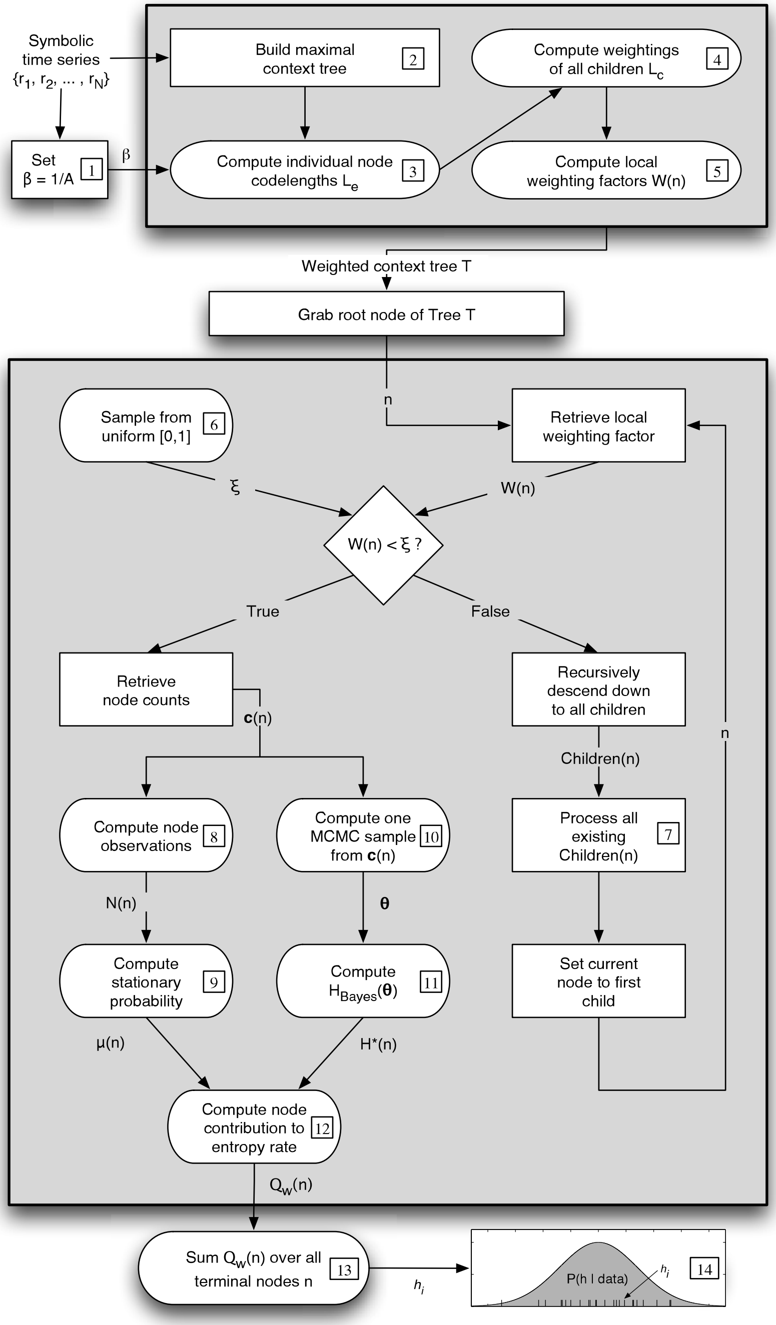
Symbolic time series
{r₁, r₂, ... , r_N}
Set
β = 1/A
1
β
Build maximal context tree
2
Compute individual node codelengths L_e
3
Compute weightings of all children L_c
4
Compute local weighting factors W(n)
5
Weighted context tree T
Grab root node of Tree T
n
Sample from uniform [0,1]
6
Retrieve local weighting factor
ξ
W(n)
W(n) < ξ ?
True
False
Retrieve node counts
c(n)
Recursively descend down to all children
Children(n)
n
Compute node observations
8
Compute one MCMC sample from c(n)
10
Process all existing Children(n)
7
N(n)
θ
Compute stationary probability
9
Compute H_Bayes(θ)
11
Set current node to first child
μ(n)
H*(n)
Compute node contribution to entropy rate
12
Q_w(n)
Sum Q_w(n) over all terminal nodes n
13
h_i
P(h I data)
h_i
14

Figure 14: Schematic block diagram for the entire procedure for estimating the range of entropy rates associated in a discrete time series.

8. The number of node observations is the sum of the counts of future symbols $N(\mathrm{n}) = \sum_{i=1}^{A} c_i(\mathrm{n})$.

9. The stationary probability distribution $\hat{\mu}(\mathrm{n})$ follows equation 4.2 (but see appendix B).

10. See section 4.2 as well as appendix A on how to generate a single Markov chain Monte Carlo sample from $P(\boldsymbol{\theta}|\mathbf{c})$.

11. Use equations 2.4 and 2.5 to estimate the corresponding entropy.

12. Following equation 4.5, the contribution of this node to the entropy rate is $Q_w(\mathrm{n}) = H^*(\mathrm{n})\hat{\mu}(\mathrm{n})$.

13. Sum up Q_w from all nodes to estimate the entropy rate from the weighted probabilistic model. The weighting $W(\mathrm{n})$ is implicit in the probabilistic arrival at node n.

14. The final output of this procedure is a single sample from the continuous distribution $P(h|\mathrm{data})$ denoted by a small line in the graph. We repeat the estimation procedure N_{MC} times to generate multiple samples from the underlying distribution.

Acknowledgments

This work was supported by NSF IGERT and La Jolla Interfaces in the Sciences (J.S.); a Sloan Research Fellowship and NIH grant EY13150 (E.J.C.). We thank Pam Reinagel for valuable discussion and support; and colleagues at the Institute for Nonlinear Science (UCSD) and the Systems Neurobiology Laboratory (Salk Institute) for tremendous feedback and experimental assistance.

References

Amigo, J., Szczepanski, J., Wajnryb, E., & Sanchez-Vives, M. (2004). Estimating the entropy rate of spike trains via Lempel-Ziv complexity. *Neural Computation, 16,* 717–736.

Balasubramanian, V. (1997). Statistical inference, Occam's razor, and statistical mechanics on the space of probability distributions. *Neural Computation, 9,* 349–368.

Bialek, W., Nemenman, I., & Tishby, N. (2001). Predictability, complexity, and learning. *Neural Computation, 13,* 2409–2463.

Bialek, W., Rieke, F., de Ruyter van Steveninck, R., & Warland, D. (1991). Reading a neural code. *Science, 252,* 1854–1857.

Borst, A., & Theunissen, F. (1999). Information theory and neural coding. *Nature Neuroscience, 2,* 947–957.

Brenner, N., Strong, S., Koberle, R., Bialek, W., & de Ruyter van Steveninck, R. (2000). Synergy in a neural code. *Neural Computation, 12,* 1531–1552.

Buracas, G., Zador, A., DeWeese, M., & Albright, T. (1998). Efficient discrimination of temporal patterns by motion-sensitive neurons in primate visual cortex. *Neuron, 20,* 959–969.

Butts, D. (2003). How much information is associated with a particular stimulus? *Network, 14,* 177–187.

Costa, J., & Hero, A. (2004). Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Transactions on Signal Processing, 25,* 2210–2221.

Cover, T., & Thomas, J. (1991). *Elements of information theory*. New York: Wiley.

de Ruyter van Steveninck, R., Lewen, G., Strong, S., Koberle, R., & Bialek, W. (1997). Reproducibility and variability in neural spike trains. *Science, 275,* 1805–1808.

DeWeese, M., & Meister, M. (1999). How to measure the information gained from one symbol. *Network, 10,* 325–340.

Duda, R., Hart, P., & Stork, D. (2000). *Pattern classification* (2nd ed.). New York: Wiley.

Field, D. (1987). Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America, A4,* 2379–2394.

Gamerman, D. (1997). *Markov chain Monte Carlo: Stochastic simulation of Bayesian inference*. New York: CRC Press.

Gilmore, R., & Lefranc, M. (2002). *The topology of chaos: Alice in stretch and squeeze land*. New York: Wiley.

Grassberger, P. (2003). Entropy estimates from insufficient samplings. *e-print,* physics/0307138.

Hastings, W. (1970). Monte Carlo sampling using Markov chains and their applications. *Biometrica, 57,* 97–109.

Hilborn, R. (2000). *Chaos and nonlinear dynamics: An introduction for scientists and engineers*. New York: Oxford University Press.

Johnson, D., Gruner, C., Baggerly, K., & Seshagiri, C. (2001). Information-theoretic analysis of neural coding. *Journal of Computational Neuroscience, 10,* 47–69.

Kennel, M., & Buhl, M. (2003). Estimating good discrete partitions from observed data: Symbolic false nearest neighbors. *Phys. Rev. Lett., 91,* 084102.

Kennel, M., & Mees, A. (2002). Context-tree modeling of observed symbolic dynamics. *Physical Review E, 66,* 056209.

Kontoyiannis, I., Algoet, P., Suhov, Y., & Wyner, A. (1998). Nonparametric entropy estimation for stationary processes and random fields with applications to English text. *IEEE Transactions in Information Theory, 44,* 1319–1327.

Kraskov, A., Stogbauer, H., & Grassberger, P. (2004). Estimating mutual information. *Physical Review E, 69,* 006138.

Krichevsky, R. E., & Trofimov, V. K. (1981). The performance of universal coding. *IEEE Transactions in Information Theory, 28,* 199–207.

Lehmann, E., & Casella, G. (1998). *Theory of point estimation*. New York: Springer.

Lempel, A., & Ziv, J. (1976). On the complexity of finite sequences. *IEEE Transactions in Information Theory, 22,* 75–78.

Lewen, G., Bialek, W., & de Ruyter van Steveninck, R. (2001). Neural coding of naturalistic motion stimuli. *Network: Computation in Neural Systems, 12,* 317–329.

Lind, D., & Marcus, B. (1996). *Symbolic dynamics and coding*. Cambridge: Cambridge University Press.

Liu, R., Tzonev, S., Rebrik, S., & Miller, K. (2001). Variability and information in a neural code of the cat lateral geniculate nucleus. *Journal of Neurophysiology, 86,* 2789–2806.

London, M., Schreibman, A., Hausser, M., Larkum, M., & Segev, I. (2002). The information efficacy of a synapse. *Nature Neuroscience, 5,* 332–340.

Ma, S. (1981). Calculation of entropy from data of motion. *Journal of Statistical Physics, 26,* 221–240.

MacKay, D. (2003). *Information theory, inference and learning algorithms*. Cambridge: Cambridge University Press.

MacKay, D., & McCulloch, W. (1952). The limiting information capacity of a neuronal link. *Bulletin of Mathematical Biophysics, 14,* 127–135.

Martin, A., Seroussi, G., & Weinberger, M. (2004). Linear time universal coding and time reversal of tree sources via FSM closure. *IEEE Transactions on Information Theory, 50,* 1442–1468.

Metropolis, N., Rosenbluth, M., Rosenbluth, A., Teller, M., & Teller, A. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics, 21,* 1087–1091.

Miller, G., & Madow, W. (1954). On the maximum likelihood estimate of the Shannon-Wiener measure of information. *Air Force Cambridge Research Center Technical Report, 75,* 54.

Nemenman, I. (2002). Inference of entropies of discrete random variables with unknown cardinalities. physics/02070009. Available online: www.arxiv.org.

Nemenman, I., Bialek, W., & de Ruyter van Steveninck, R. (2004). Entropy and information in neural spike trains: Progress on the sampling problem. *Physical Review E*, 69.

Nemenman, I., Shafee, F., & Bialek, W. (2002). Entropy and inference, revisited. *Advances in neural information processsing systems, 14,* In T. G. Dietterich, S. Becker, & Z. Ghahraman: (Eds.), Cambridge, MA: MIT Press.

Orlitsky, A., Santhanam, N., & Zhang, J. (2004). Universal compression of memoryless sources over unknown alphabets. *IEEE Trans. Inf. Theo., 50,* 1469–1481.

Ott, E. (2002). *Chaos in dynamical systems*. Cambridge: Cambridge University Press.

Paninski, L. (2003). Estmation of entropy and mutual information. *Neural Computation, 15,* 1191–1254.

Perkel, D., & Bullock, T. (1968). Neural coding. *Neurosci Res. Prog. Sum., 3,* 405–527.

Rapp, P., Vining, E., Cohen, N., Albano, A., & Jimenez-Montano, M. (1994). The algorithmic complexity of neural spike trains increases during focal seizures. *Journal of Neuroscience, 14,* 4731–4739.

Reinagel, P., & Reid, R. (2000). Temporal coding of visual information in the thalamus. *Journal of Neuroscience, 20,* 5392–5400.

Rieke, F., Bodnar, D., & Bialek, W. (1995). Naturalistic stimuli increase the rate and efficiency of information transmission by primary auditory afferents. *Proceedings of the Royal Society of London B: Biological Sciences, 262,* 259–265.

Rieke, F., Warland, D., de Ruyter van Steveninck, R., & Bialek, W. (1997). *Spikes: Exploring the neural code*. Cambridge, MA: MIT Press.

Rissanen, J. (1989). *Stochastic complexity in statistical inquiry*. Singapore: World Scientific.

Roulston, M. (1999). Estimating the errors on measured entropy and mutual information. *Physica D*, *125*, 285–294.

Ruderman, D., & Bialek, W. (1994). Statistics of natural images: Scaling in the woods. *Physical Review Letters*, *73*, 814–817.

Schneidman, E., Bialek, W., & Berry, M. (2003). Synergy, redundancy, and independence in population codes. *Journal of Neuroscience*, *23*, 11539–11553.

Schurmann, T., & Grassberger, P. (1996). Entropy estimation of symbol sequences. *Chaos*, *6*, 414–427.

Shannon, C. (1948). A mathematical theory of communication. *Bell Systems Technology Journal*, *27*, 379–423.

Shlens, J., Kennel, M., Abarbanel, H., & Chichilnisky, E. (In press). *Estimating information rates with Bayesian confidence intervals in neural spike trains.* University of California, San Diego, and Salk Institute.

Shtarkov, Y., Tjalkens, T., & Willems, F. (1995). Multialphabet weighting: Universal coding of context tree sources. *Problems of Info. Trans.*, *33*, 17–28.

Simonicelli, E., & Olhausen, B. (2001). Natural image statistics and neural representation. *Annual Review of Neuroscience*, *24*, 1193–1215.

Solomonoff, R. (1964). A formal theory of inductive inference. Part I. *Information and Control*, *7*, 1–22.

Strong, S., Koberle, R., de Ruyter van Steveninck, R., & Bialek, W. (1998). Entropy and information in neural spike trains. *Physical Review Letters*, *80*, 197–200.

Treves, A., & Panzeri, S. (1995). The upward bias in measures of information derived from limited data samples. *Neural Computation*, *7*, 399–407.

Victor, J. (2000). Asymptotic bias in information estimates and the exponential (Bell) polynomials. *Neural Computation*, *12*, 2797–2804.

Victor, J. (2002). Binless strategies for estimation of information from neural data. *Physical Review E*, *66*, 051903.

Victor, J., & Purpura, K. (1997). Metric-space analysis of spike trains: Theory, algorithms, and application. *Network*, *8*, 127–164.

Warland, D., Reinagel, P., & Meister, M. (1997). Decoding visual information from a population of retinal ganglion cells. *Journal of Neurophysiology*, *78*, 2336–2350.

Willems, F. (1998). Context tree weighting: Extensions. *IEEE Transactions on Information Theory*, *44*, 792–798.

Willems, F., Shtarkov, Y., & Tjalkens, T. (1995). The context tree weighting method: Basic properties. *IEEE Transactions in Information Theory*, *IT-41*, 653–664.

Wolpert, D., & Wolf, D. (1995). Estimating functions of probability distributions from a finite set of samples. *Physical Review E*, *52*, 6841–6854.

Wyner, A. D., Ziv, J., & Wyner, A. J. (1998). On the role of pattern matching in information theory. *IEEE Transactions on Information Theory*, *44*, 2045–2056.

Ziv, J., & Lempel, A. (1977). A universal algorithm for sequential data compression. *IEEE Transactions in Information Theory*, *23*, 337–343.

Ziv, J., & Lempel, A. (1978). Compression of individual sequences via variable-rate coding. *IEEE Transactions in Information Theory*, *24*, 530–536.

Received June 23, 2004; accepted January 5, 2005.