Pattern Discovery in Time Series, Part II: Implementation, Evaluation, and Comparison

Kristina Lisa Shalizi KLINKNER@SANTAFE.EDU
Physics Department, University of San Francisco, 2130 Fulton Street, San Francisco, CA 94117

Cosma Rohilla Shalizi SHALIZI@SANTAFE.EDU
James P. Crutchfield CHAOS@SANTAFE.EDU
Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, NM 87501

Editor:

Abstract

We present a new algorithm for discovering patterns in time series or other sequential data. In the prior companion work, Part I, we reviewed the underlying theory, detailed the algorithm, and established its asymptotic reliability and various estimates of its data-size asymptotic rate of convergence. Here, in Part II, we outline the algorithm's implementation, illustrate its behavior and ability to discover even "difficult" patterns, demonstrate its superiority over alternative algorithms, and discuss its possible applications in the natural sciences and to data mining.

Draft of October 24, 2002

1. Introduction

Motivated in part by the recent explosion in the size of databases, there is an increasing awareness of the problem of *pattern discovery* (Crutchfield, 1994): Given data produced by some process, how can one extract meaningful, predictive patterns from it, without fixing, in advance, the kind of patterns one looks for? In a companion work, Part I, we explained how *computational mechanics* (Crutchfield and Young, 1989, Crutchfield, 1994, Upper, 1997, Feldman and Crutchfield, 1998, Shalizi and Crutchfield, 2001) provides an answer to this general problem, first by reviewing the underlying theory and then by introducing a new algorithm—*Causal-State Splitting Reconstruction* (CSSR)—that discovers patterns intrinsic to a process. There we demonstrated that CSSR converges to the true model in the infinite-data limit and established bounds on its convergence rate using techniques from large-deviation theory.

This sequel complements Part I's theoretical emphasis with the practical issues of implementation, evaluation on a range of prototype stochastic processes, and comparison to the closest alternative approaches that claim to do something like pattern discovery. Naturally, we expect the reader to be familiar with Part I (Shalizi et al., 2002), especially the vocabulary of computational mechanics introduced there, its notation, and the basic workings of the CSSR algorithm. Here we demonstrate, experimentally, how well the CSSR algorithm does at reconstructing the causal architecture from samples of known processes and how various parameters affect reconstruction accuracy and quality. We then compare CSSR with other procedures for inferring Markovian structure in time series. Our conclusion summarizes our results and suggests directions for future applications.

Here we first briefly review the basic concepts and notation from information theory and computational mechanics necessary for understanding, on the one hand, the CSSR algorithm and, on the other, how to evaluate its performance. Then we review CSSR itself and develop the specific evaluation measures that we use. At that point, the development turns into a sequence of pattern discovery tests applied to sources of increasing structural complexity. Finally, we compare CSSR to current popular approaches to the same problem, concluding with a few remarks on additional areas of application.

2. Information Theory

Here we give a synopsis of information theory (Cover and Thomas, 1991), since many of the performance measures for the CSSR pattern discovery are most appropriately defined as various measures of uncertainty and stored information.

For a random variable S taking values in a countable set $\mathcal{A}$, the *entropy* of S is defined as

$$H[S] \equiv -\Sigma_{s\in\mathcal{A}} \mathrm{P}(S=s)\log_2 \mathrm{P}(S=s) \ . \quad (1)$$

The entropy is interpreted as the uncertainty in S or as the mean number of yes or no questions needed to pick out the values of S on repeated trials. $H[S]$ has units of *bits* of information.

For two variables X (taking values in $\mathcal{A}$) and Y (taking values in $\mathcal{B}$) the *joint entropy* is

$$H[X,Y] \equiv -\Sigma_{(x,y)\in(\mathcal{A}\times\lfloor)} \mathrm{P}(X=x\cdot Y=y)\log_2 \mathrm{P}(X=x\cdot Y=y) \ . \quad (2)$$

One defines the conditional entropy of one random variable X on another Y from their joint entropy as follows

$$H[X|Y] \equiv H[X,Y] - H[Y] \ . \quad (3)$$

This follows directly from the definition of conditional probability, $\mathrm{P}(X=x|Y=y) \equiv \mathrm{P}(X=x\cdot Y=y)/\mathrm{P}(Y=y)$. The conditional entropy can be interpreted as the remaining uncertainty in X, once we know Y.

If we have two random variables, X and Y, governed by distributions, P and Q, respectively, that are defined over the same space $\mathcal{A}$, we may consider the *entropy of P relative to Q*:

$$\mathcal{D}(X||Y) \quad = \quad -\sum_{s\in\mathcal{A}} P(X=s)\log\frac{P(X=s)}{Q(Y=s)} \ , \quad (4)$$

with the understanding that $0\log 0/q = 0$ and $p\log p/0 = \infty$. This quantity is also known as the (Kullback-Leibler) *divergence* or the *information gain*. It vanishes when, and only when, the two distributions are equal. But it is not symmetric and does not satisfy the triangle inequality, and so $\mathcal{D}(X||Y)$ is not a distance. If Q assigns zero probability to an event that has positive probability under P, then $\mathcal{D}(X||Y) \to \infty$. In addition to various statistical uses (Kullback, 1968, Cover and Thomas, 1991), it has a useful interpretation in terms of codes. If P is the true distribution, then we need an average of $H_P[X]$ bits to describe or encode the random variable X. If, however, we mistakenly think the distribution is Q, we will need $H_P[X]+D(P||Q)$ bits. Relative entropy is thus the excess description length due to getting the distribution wrong.

The *mutual information* between X and Y is defined as

$$I[X;Y] \equiv H[X] - H[X|Y] \ . \quad (5)$$

In words, the mutual information is the average reduction of uncertainty of one variable due to knowledge of another. If knowing Y on average makes one more certain about X, then it makes sense to say that Y carries information about X. Note that $I[X;Y] \geq 0$ and that $I[X;Y]=0$ when either X and Y are independent (there is no "communication" between X and Y) or when either $H[X]=0$ or $H[Y]=0$ (there is no information to share). Note also that $I[X;Y]=I[Y;X]$.

3. Processes

The data streams we shall consider will be stationary stochastic processes. In this section we introduce this idea more formally, fix notation, and define a few classes of stochastic process to which we shall return when developing performance measures.

The main object of our attention will be a one-dimensional chain

$$\overleftrightarrow{S} \equiv \; \ldots S_{-2} S_{-1} S_0 S_1 \ldots \tag{6}$$

of random variables S_t whose values range over a finite alphabet set $\mathcal{A}$. We assume that the underlying system is described by a shift-invariant measure μ on infinite sequences $\cdots s_{-2} s_{-1} s_0 s_1 s_2 \cdots ; s_t \in \mathcal{A}$ Gray (1990). The measure μ induces a family of distributions, $\{\Pr(s_{t+1}, \ldots, s_{t+L}) : s_t \in \mathcal{A}\}$, where $\Pr(s_t)$ denotes the probability that at time t the random variable S_t takes on the particular value $s_t \in \mathcal{A}$ and $\Pr(s_{t+1}, \ldots, s_{t+L})$ denotes the joint probability over blocks of L consecutive symbols. We assume that the distribution is stationary; $\Pr(s_{t+1}, \ldots, s_{t+L}) = \Pr(s_1, \ldots, s_L)$.

These quantities are useful tools for the analysis of time series; for a detailed review see Crutchfield and Feldman (2001). We denote a block of L consecutive variables by $S^L \equiv S_1 \ldots S_L$. We shall follow the convention that a capital letter refers to a random variable, while a lowercase letter denotes a particular value of that variable. Thus, $s^L = s_1 s_2 \cdots s_L$, denotes a particular symbol block of length L. We shall use the term *process* to refer to the joint distribution $\Pr(\overleftrightarrow{S})$ over the infinite chain of variables. A process, defined in this way, is what Shannon referred to as an information source.

The *total Shannon entropy* of length-L sequences is defined

$$H(L) \; \equiv \; - \sum_{s^L \in \mathcal{A}^L} \Pr(s^L) \log_2 \Pr(s^L) \; , \tag{7}$$

where $L > 0$. The sum is understood to run over all possible blocks of L consecutive symbols. If no measurements are made, there is nothing about which to be uncertain and, thus, we define $H(0) \equiv 0$. $H(L)$ is a non-decreasing function of L, $H(L) \geq H(L-1)$, and it is concave, $H(L) - 2H(L-1) + H(L-2) \leq 0$.

The *source entropy rate* h_μ is the rate of increase with respect to L of the total Shannon entropy in the large-L limit:

$$h_\mu \equiv \lim_{L \to \infty} \frac{H(L)}{L} \; , \tag{8}$$

where μ denotes the measure over infinite sequences that induces the L-block joint distribution $\Pr(s^L)$; the units are *bits/symbol.* The limit in Eq. (8) exists for all stationary measures μ Cover and Thomas (1991). The entropy rate h_μ quantifies the irreducible randomness in sequences produced by a source: the randomness that remains after the correlations and structures in longer and longer sequence blocks are taken into account.

As is well known (see, e.g., Ref. Cover and Thomas (1991), the entropy rate may also be written as:

$$h_\mu \; = \; \lim_{L \to \infty} H[S_L | S^{L-1}] \; . \tag{9}$$

That is, h_μ is the average uncertainty of the variable S_L, given that an arbitrarily large number of preceding symbols have been seen.

4. Computational Mechanics

The fundamental representation used in computational mechanics for a stochastic process is its ϵ-machine , which is the process's minimal and unique optimal predictor. An ϵ-machine M consists of a set of *causal states* $\boldsymbol{\mathcal{S}}$ and a set of *causal-state transitions* $\boldsymbol{\mathcal{S}} = \{\mathcal{T}^{(s)}, s \in \mathcal{A}\}$ (Crutchfield and Young, 1989, Crutchfield, 1994, Shalizi and Crutchfield, 2001). The causal states are either transient or recurrent. The recurrent states all lie in a single strongly connected component. There is a unique start state $\sigma_0 \in \boldsymbol{\mathcal{S}}$.

Given a causal state and the next symbol from the process, only certain successor causal states are possible. The probability of moving from state σ_i to state σ_j on symbol s is

$$\mathcal{T}_{ij}^{(s)} \equiv \mathrm{P}(\overrightarrow{S}^1 = s, \mathcal{S}' = \sigma_j | \mathcal{S} = \sigma_i) \ . \tag{10}$$

Note that

$$\sum_{s \in \mathcal{A}} \sum_{\sigma_j \in \boldsymbol{\mathcal{S}}} T_{ij}^{(s)} = \sum_{s \in \mathcal{A}} \mathrm{P}(\overrightarrow{S}^1 = s | \mathcal{S} = \sigma_i) = 1 \ . \tag{11}$$

The probability of a sequence $s^L = s_0 s_1 \ldots s_{L-1}$ is given by starting in σ_0 and forming the telescoping product of transition probabilities along the path selected by the successive symbols in s^L. The probability of s^L appearing between any two states is given by the matrix

$$\mathcal{T}^{s^L} \equiv \mathcal{T}^{(s_{L-1})} \mathcal{T}^{(s_{L-2})} \cdots \mathcal{T}^{(s_0)} \ . \tag{12}$$

And the probability of the process generating s^L is simply that of seeing it from the start state:

$$\mathrm{P}(s^L | M) = (\mathcal{T}^{s^L})_{0j} \ , \tag{13}$$

where j denotes the causal state in which s^L ends. In this way, an ϵ-machine compactly gives the distribution over all sequences.

The amount of historical memory stored by the process is given by the *statistical complexity* of its ϵ-machine :

$$C_\mu = - \sum_{\sigma \in \boldsymbol{\mathcal{S}}} \mathrm{P}(\sigma) \log_2 \mathrm{P}(\sigma) \ , \tag{14}$$

where the causal-state probabilities are given by the principal eigenvector, normalized in probability, of the state-to-state transition matrix. The latter is simply:

$$\mathcal{S} = \sum_{s \in \mathcal{A}} \mathcal{S}^{(s)} \ . \tag{15}$$

Note that $\mathrm{P}(\sigma) = 0$, if σ is a transient state.

The process's entropy rate h_μ is given by the transition uncertainty averaged over it's ϵ-machine 's causal states:

$$h_\mu = - \sum \sigma \in \boldsymbol{\mathcal{S}} \mathrm{P}(\sigma) \sum_{s \in \mathcal{A}} \mathrm{P}(\sigma) \mathcal{T}_\sigma^{(s)} \log_2 \mathcal{T}_\sigma^{(s)} \ . \tag{16}$$

Note that this direct formula for h_μ is only possible due to a key property of ϵ-machines : they are deterministic in the sense of automata theory. This is also why we can use the notation $\mathcal{T}_\sigma^{(s)}$ above to unambiguously indicate the unique positive transition probability for leaving state σ on symbol s, if one exists.

A number of other ϵ-machine properties were reviewed in Part I (Shalizi et al., 2002).

5. Causal-State Splitting Reconstruction

In Part I (Shalizi et al., 2002), we introduced the Causal-State Splitting Reconstruction (CSSR aka "scissor") algorithm that estimates an ϵ-machine from samples of a process $\mathcal{P}$. It starts out assuming a simple model for $\mathcal{P}$ and elaborates model components (states and transitions) only when statistically justified by the given data. Specifically, CSSR begins assuming the process is independent, identically distributed (IID) over the alphabet $\mathcal{A}$. This is equivalent to assuming the process is structurally simple ($C_\mu(\mathcal{P}) = 0$) and is as random as possible ($h_\mu(\mathcal{P}) \leq \log_2 k$). A key and distinguishing property of CSSR is that it maintains homogeneity of the causal states and determinism of the state-to-state transitions as the model is augmented. The result is that at each stage the estimated model is an ϵ-machine .

During CSSR's operation, the number of causal states typically grows and the ϵ-machine captures increasingly more of the process's causal architecture. That is, CSSR converges "from below" to the process's true architecture in the sense that C_μ, the amount of structural information captured in the series of estimated ϵ-machines , grows from zero to the true value. That it does, in fact, converge in this limit was establised in Part I (Shalizi et al., 2002).

CSSR consists of three steps that manipulate an estimated ϵ-machine : Initialize, Homogenize, and Determinize. Our goal here, though, is to investigate the algorithm's performance on a range of different discrete stochastic processes. And so, the internal details are not of direct interest. Instead, we largely treat CSSR as a black box with various inputs and control parameters that produces approximate ϵ-machines . For a given application there are several parameters. The first characterize the available data. This is assumed to be a discrete-time series $s^N = s_0 s_1 s_2 \ldots s_{N-1}$ of N samples taken from a discrete, finite alphabet $\mathcal{A}$ of k symbols: $s_i \in \mathcal{A}$. Additional parameters control the behavior of the CSSR algorithm itself. The first is maximum length L_{max} of history sequence that will be taken into account. The second is the significance level δ of the Markovian null-hypothesis test used for morph-similarity comparison.

Given s^N, the empirical sequence distribution, denoted $\mathrm{P}_{s^N}(s^L)$, is formed by simply counting the occurrence of words s^L. Given an ϵ-machine $\widehat{\mathrm{M}} \equiv \{\widehat{\boldsymbol{\mathcal{S}}}, \widehat{\boldsymbol{\mathcal{S}}}\}$ estimated from s^N, one can form the empirical ϵ-machine sequence distribution $\mathrm{P}_{\widehat{\mathrm{M}}}(s^L)$. Stated prosaicly, then, starting with a process sample s^N and a choice of $L_{\max}$, the goal of CSSR is to estimate an ϵ-machine $\widehat{M}$ that minimizes the difference between the true and ϵ-machine 's sequence distribution and the true one. For the latter, as is common practice, one substitutes $\mathrm{P}_{s^N}(s^L)$.

6. Implementation

Give details on algorithmic implementation, not covered by Part I's formal and pseudo-code description.

For reasons of speed and memory, the conditional probability distributions were stored as a parse tree. We used the Kolmogorov-Smirnov test, modifying slightly the code in Press et al. (1992) and set the significance level $\delta = 0.001$.

Any other notable implementation details of how tests were constructed? Note those here.

CSSR is implemented in C++. The source code is at the Computational Mechanics Archive, `http://www.santafe.edu/projects/CompMech/`. It was developed and tested on UNIX; in particular, Sun Solaris 8 and Macintosh OS X.

7. Evaluation Measures

We evaluated CSSR's performance using a number of statistics: estimated entropy rate, estimated statistical complexity, empirical (in-sample) prediction error, generalization (out-of-sample) error, and the equivocation of the true causal states with respect to the inferred states. These are defined as follows.

First, one error measure for CSSR is the accuracy of the entropy-rate estimate $h_\mu(\widehat{M})$. It is worth noting that even when a reconstructed ϵ-machine is quite inaccurate in other respects, it gives good estimates of h_μ on comparatively little data and with comparatively short histories. These estimates are certainly better than those based directly on sequence-block statistics. (For more on the convergence of estimates of h_μ, see Young and Crutchfield, 1993, Crutchfield and Feldman, 2001, Schürmann and Grassberger, 1996, Shaw, 1984, Grassberger, 1988, 1989, Herzel et al., 1994, .)

Second, we estimate the statistical complexity $C_\mu(\widehat{M})$ of the inferred causal states $\widehat{\boldsymbol{\mathcal{S}}}$. As shown in Part I, this should converge to the true statistical complexity C_μ. Comparing $\widehat{\boldsymbol{\mathcal{S}}}$ to C_μ, in the cases where the latter can be calculated, indicates structural errors in the reconstruction, perhaps in the number of inferred states or transitions.

Third, we measure the empirical sequence error; that is, the error on the training data s^N. Recall that any ϵ-machine $\widehat{\mathrm{M}}$ generates a distribution $\mathrm{P}_{\widehat{\mathrm{M}}}(s^L)$ over sequences. One wants to see how far $\mathrm{P}_{\widehat{\mathrm{M}}}(s^L)$ deviates from the empirical sequence distribution $\mathrm{P}_{s^N}(s^L)$. A convenient measure of this is the *total variational distance* between the two distributions, or the *variation* for short. As above, this is

$$d_L(\mathrm{P}_{\widehat{\mathrm{M}}}(s^L), \widehat{\mathrm{P}}_N) = \sum_{s^L \in \mathcal{A}^L} |\mathrm{P}_{\widehat{\mathrm{M}}}(s^L) - \widehat{\mathrm{P}}_N| \ . \tag{17}$$

Note that variation is a nondecreasing function of L. Ideally, one would take it over infinite sequences. This not accessible numerically, so we have used the longest strings available.

Since we are reconstructing known processes, we can calculate the generalization error directly; we simply find the variation between the inferred sequence distribution $\mathrm{P}_{\widehat{\mathrm{M}}}(s^L)$ and the true distribution $\mathrm{P}(s^L)$, again computed over sequences of fixed length.

Normally, one thinks of the generalization error as the expected value of the empirical error on new data, but this is not appropriate here. To see why, suppose the algorithm is completely successful, so the distribution under the reconstructed ϵ-machine is exactly the true distribution, and the generalization error is zero. Then it would be extremely unusual for the empirical error to be zero — that would happen only if there were no sampling error.[1] Similarly, the expected empirical error on new data is the expectation value of a non-negative quantity, so it cannot be zero. The discrepancy between the generalization error and the expected empirical error grows with L (for fixed N), because there are exponentially more strings, but fewer samples per string, and so the magnitude of the sampling fluctuations grows. This generates a spurious error signal if the generalization error is defined this way. An alternative is needed.

Since the total variation is a metric, it is tempting to apply the triangle inequality and to say that

$$d(\mathrm{P}(s^L), \mathrm{P}_{\mathrm{M}}(s^L)) \leq d(\mathrm{P}(s^L), \mathrm{P}_{s^N}(s^L)) + d(\mathrm{P}_{s^N}(s^L), \mathrm{P}_{\mathrm{M}}(s^L)) \tag{18}$$

and

$$d(\mathrm{P}_{\mathrm{M}}(s^L), \mathrm{P}_{s^N}(s^L)) \leq d(\mathrm{P}_{\mathrm{M}}(s^L), \mathrm{P}(s^L)) + d(\mathrm{P}(s^L), \mathrm{P}_{s^N}(s^L)) \ , \tag{19}$$

so that

$$\begin{aligned} d(\mathrm{P}_{\mathrm{M}}(s^L), \mathrm{P}_{s^N}(s^L)) &- d(\mathrm{P}(s^L), \mathrm{P}_{s^N}(s^L)) \\ &\leq d(\mathrm{P}(s^L), \mathrm{P}_{\mathrm{M}}(s^L)) \\ &\leq d(\mathrm{P}_{\mathrm{M}}(s^L), \mathrm{P}_{s^N}(s^L)) + d(\mathrm{P}(s^L), \mathrm{P}_{s^N}(s^L)) \ . \end{aligned} \tag{20}$$

That is to say, the generalization error $d(\mathrm{P}(s^L), \mathrm{P}_{\mathrm{M}}(s^L))$ is equal to the empirical error, plus or minus the sampling fluctuation. Unfortunately, it is very hard to put reasonable bounds on the sampling fluctuation, without knowing in advance the process's structural and statistical characteristics. But the latter are what CSSR is designed to tell us.

Finally, we measure the equivocation $H[\mathcal{S}|\widehat{\mathcal{S}}]$ of the inferred states with respect to the causal states. This is how uncertain we remain about the true causal state after knowing the inferred state. Since prescient (maximally predictive) states are all refinements of the causal states (Shalizi et al., 2002), this quantity must be zero if the inferred states are prescient. In fact, we expect, and see, a tendency for the equivocation to fall. Naturally, we can only measure this for our present examples because we know what inferred states are.

Here we need to say a few words about how we will plot these numbers—i.e., versus L—and how to interrupt the resulting plots.

1. The sampling error can vanish only if all the probabilities in the true distribution are rational numbers.

8. Results on Test Cases

We tested CSSR on a wide range of processes where we can work out the true ϵ-machine by hand. Generally, CSSR converges on the true ϵ-machine "from below" as N and L increase. That is, it errs by having too few states. If L is too long, relative to N, there is a tendency for the number of states to "blow up" — spurious distinctions get made in Procedure II (Homogenize), due to undersampling, and these ramify when Procedure III (Determinize) makes the states future-resolving. If L is not too large relative to N, however, then the algorithm finds the correct structure quite rapidly, even for reasonably complex processes and usually stays with the correct structure over some range of L. This is a helpful indication that CSSR has found the right ϵ-machine . In every case, the run-time performance was much better than the worse-case analysis of Part I leads one to expect.

We emphasize that we are testing the algorithm on known processes, where we can work out the ideal results. Hence the effects of limited data size, history length, parameter choices, process characteristics, and on on can be studied systematically and stated with some confidence. We realize that this is not how most authors proceed; rather one often sees new algorithms run on complicated and poorly understood data sets and comparisons of their performance to those of rival algorithms. Since the ideal limits of predictors of such data sets are simply unknown, such testing does not indicate much about the algorithm's absolute performance. Of course, this is not to disparage building good predictors of messy real-world data. We simply feel that this methodology is not particularly useful when checking how well an algorithm works. Thus, here our tests will give evaluations that indicate absolute performance characteristics of CSSR.

8.1 "Open" Markov Models

We first consider processes where the underlying structure can be obtained directly from the observed sequence distribution, in the form of normal Markov chains. We start first with IID processes and then look at several first-order Markov chains. While these are, in a sense, very easy, some learning procedures, such as Rissanen's original "context" algorithm (Rissanen, 1983), actually estimate a number of states for them which diverges logarithmically with N. CSSR does not.

Cite (possibly) the Lempel-Ziv paper on "estimating the order of a Markov chains".

8.1.1 The IID Case: Avoiding Over-Structuring

Our first example is the simplest stochastic process possible, the fair coin; more exactly, an IID Bernoulli process with success probability $\mu = 0.5$. CSSR should do exactly nothing, since the initial ϵ-machine model is precisely the correct one. We include this case here, however, since it is *not* trivial for other reconstruction algorithms to get it right. Many methods for reconstructing dynamics from time series assume in effect that the underlying model is deterministic, or perhaps "deterministic to within measurement error". When confronted with IID noise, these methods produce more and more complicated deterministic models — in effect, trying to model noise as an infinite-dimensional dynamical system. [**Do we need citations here, or is this point made? Savit has a bad paper which does just this, quite clearly.**] Our results (Figures 1(a0–1(g)) show one advantage of a fundamentally stochastic method: to model coin-tossing, one tosses a coin. Indeed, since the generalization error is always at most equal to the empirical error, our inferred model is actually *more* faithful to the process than the sample is.

Discuss the results contained in the plots.

Introduce and discuss biased coin.

Discuss the results contained in the plots.

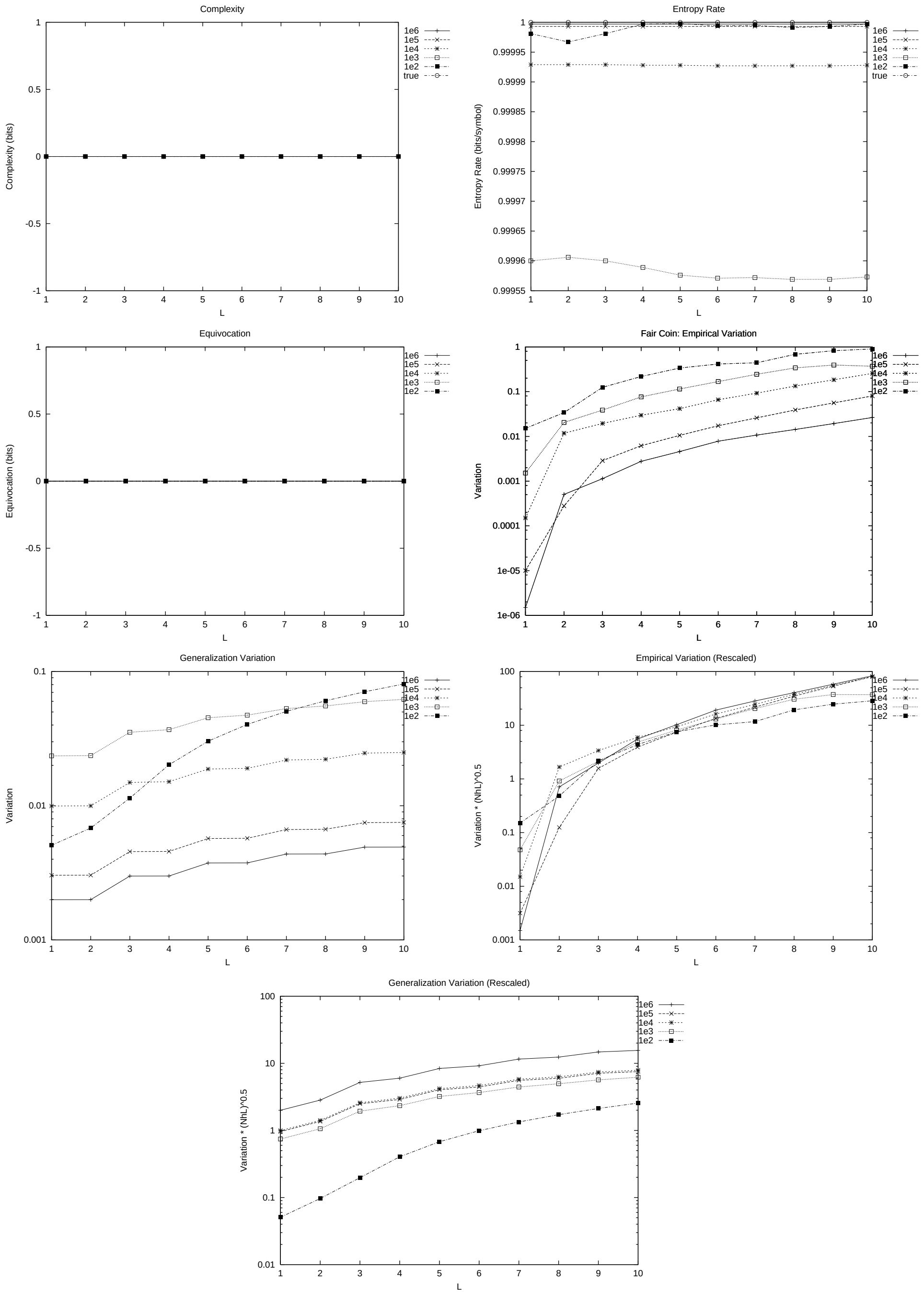

Figure 1: Fair coin reconstructed—CSSR performance measures as a function of sequence length L_{max}: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{Nh_\mu L}$; and (g) generalization error scaled by $\sqrt{Nh_\mu L}$.

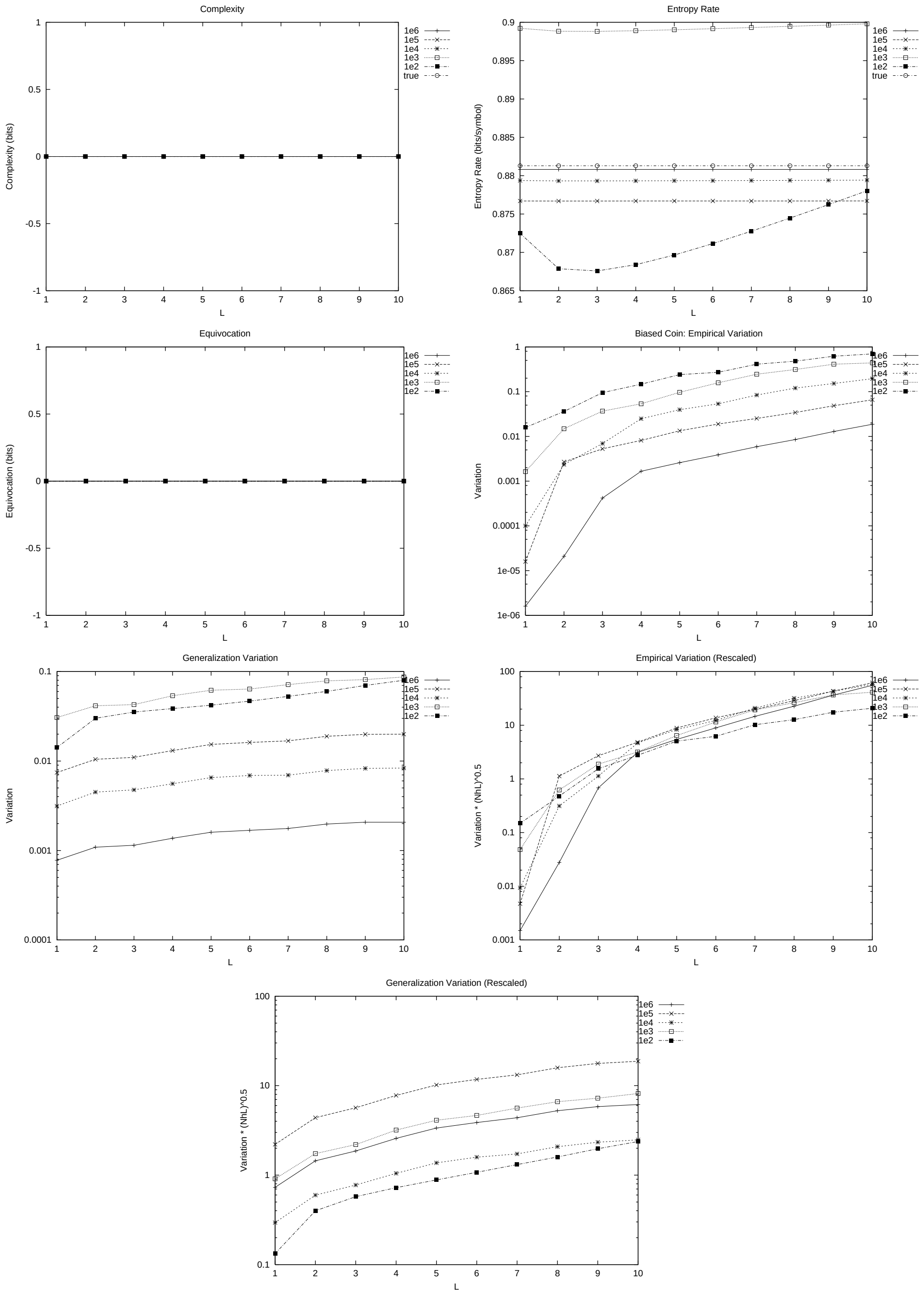

Figure 2: The biased coin—CSSR performance measures as a function of sequence length L_{max}: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{Nh_\mu L}$; and (g) generalization error scaled by $\sqrt{Nh_\mu L}$.

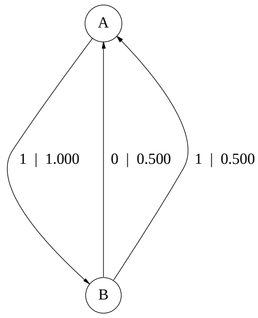

Figure 3: The Noisy Period-Two Process.

8.2 Hidden Markov Models

We now turn to the case of hidden Markov models which have a finite number of causal states. We consider two in this subsection, each of which illustrates a specific point.

8.2.1 Noisy Period-Two: The Need for Non-Suffix States

Our next example, and the first hidden Markov model, is the so-called "noisy period-two" process, chosen to illustrate the need for non-suffix states. As we see from Figure 3, there are two states, A and B, which alternate with each other (hence "period two"). State A always emits a 1, but state B emits 0s and 1s with equal probability. This corresponds to observing an ordinary period two process, which is just a Markov chain, through a noisy instrument which sometimes corrupts 0s into 1s.

Consider which histories belong in which state. Clearly, any history ending in 0 belongs to state A. Thus, any history ending in 01 belongs to B. This in turn implies that histories ending in 010 and 011 belong to A. 010 ends in 0, so it is already accounted for. But 0 is not a suffix of 011, so A contains at least *two* suffixes. It is easy to go from this to the realization that A contains an infinite number of suffixes, as does B. Algorithms that assume a finite number of one-suffix states, such as context algorithms, are unable to capture the structure of this or any other noisy-periodic process. CSSR gets the structure very rapidly, however (Figures 4(a)–4(g)).

Discuss the results contained in the plots.

Give true h_μ and C_μ.

Give the ϵ-machines for $L = 1, 2, 3, 4$.

8.2.2 Three-State Deterministic HMM: The Utility of Determinization

In all our examples so far, every state has had a different length-one morph, so we have only had to estimate the next-symbol distributions to determine the states. While many modeling methods implicitly assume this is always the case, it is easy to find examples to the contrary. It might then seem that we will have to estimate the distribution over futures of some length, and group suffixes into causal states on the basis of those distributions. Even if we knew how far forward we had to look, this would increase the amount of data required for a given level of accuracy. Determinization gives us a way out, since we still only need to estimate the next-symbol distribution.

To illustrate this point, our next example is a three-state hidden Markov model (Figure 5). Observe that two of the states, A and B, have the same probability of emitting a 1, i.e., they have the same length-one morph. In the homogenization phase of CSSR, therefore, the histories belonging to those states are grouped together. In the determinization phase, CSSR separates them, since only

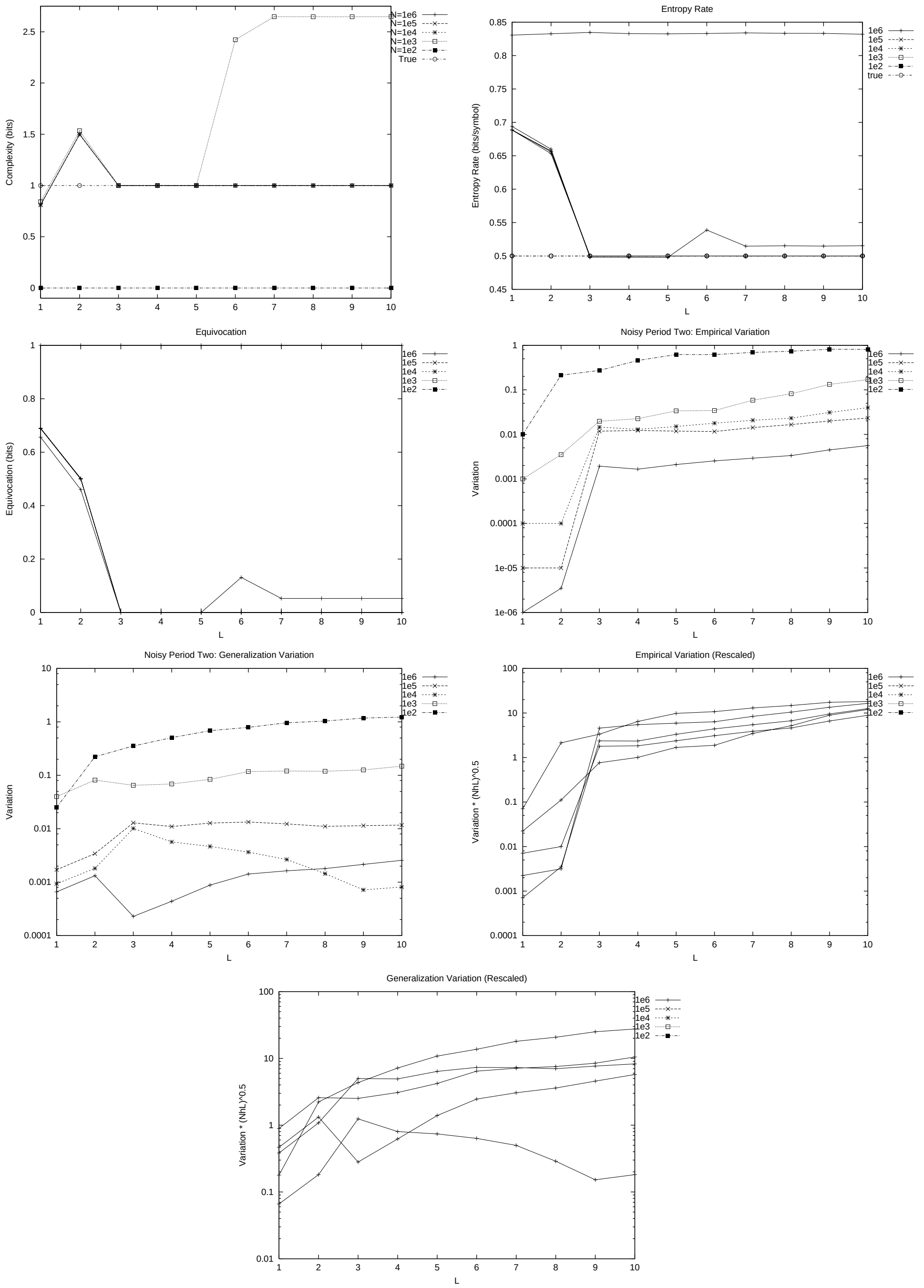

Figure 4: Noisy period-two process reconstructed—CSSR performance measures as a function of sequence length L_{max}: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{N h_\mu L}$; and (g) generalization error scaled by $\sqrt{N h_\mu L}$.

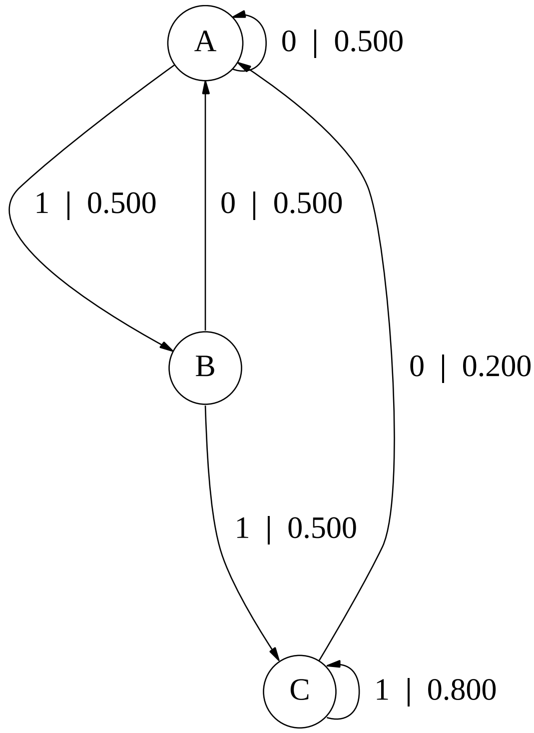

Figure 5: Three-state deterministic hidden Markov model.

some of them lead to state C. As can be seen from the results (Figures 6(a)–6(g)), CSSR gets the correct structure on the basis of a fairly short history.

Discuss the results contained in the plots.

Give true h_μ and C_μ.

Give the ϵ-machines for $L = 1, 2, 3, 4$.

8.3 Measure (Strictly) Sofic Processes

Weiss (1973) defined the class of *sofic* systems, as all those that can be described using only a finite number of states. Sofic systems are divided into two classes, depending on how many irreducible forbidden words (IFWs) they have. IFWs are the smallest words out of which all disallowed sequences are formed. Processes of *finite type* have only a finite number; *strictly sofic* systems have an infinite number of IFWs. The main and key consequence is that strictly sofic systems cannot be described by finite-order Markov chains. (There is, however, a sense in which the sequence distribution of a strictly sofic system is the limit of a sequence of increasingly larger finite-order Markov chains Weiss (1973).) *Sofic processes* are sofic systems, but with a measure put on the sequences; roughly speaking, the associated state-to-state transitions of the sofic system are given probabilities. Thus, sofic processes have a kind of long-range memory, since occurrences indefinitely far in the past can affect the distribution of future events.

The following two statements are wrong.

So far, all our examples have been of finite type. (The noisy period-two process, for instance, has only one irreducible forbidden word 00.) These are generally thought to be easier to learn than strictly sofic processes, due to their long-range memory, which finite-type processes do not have. Sofic processes are higher in the computational hierarchy and consequently require more generalization —

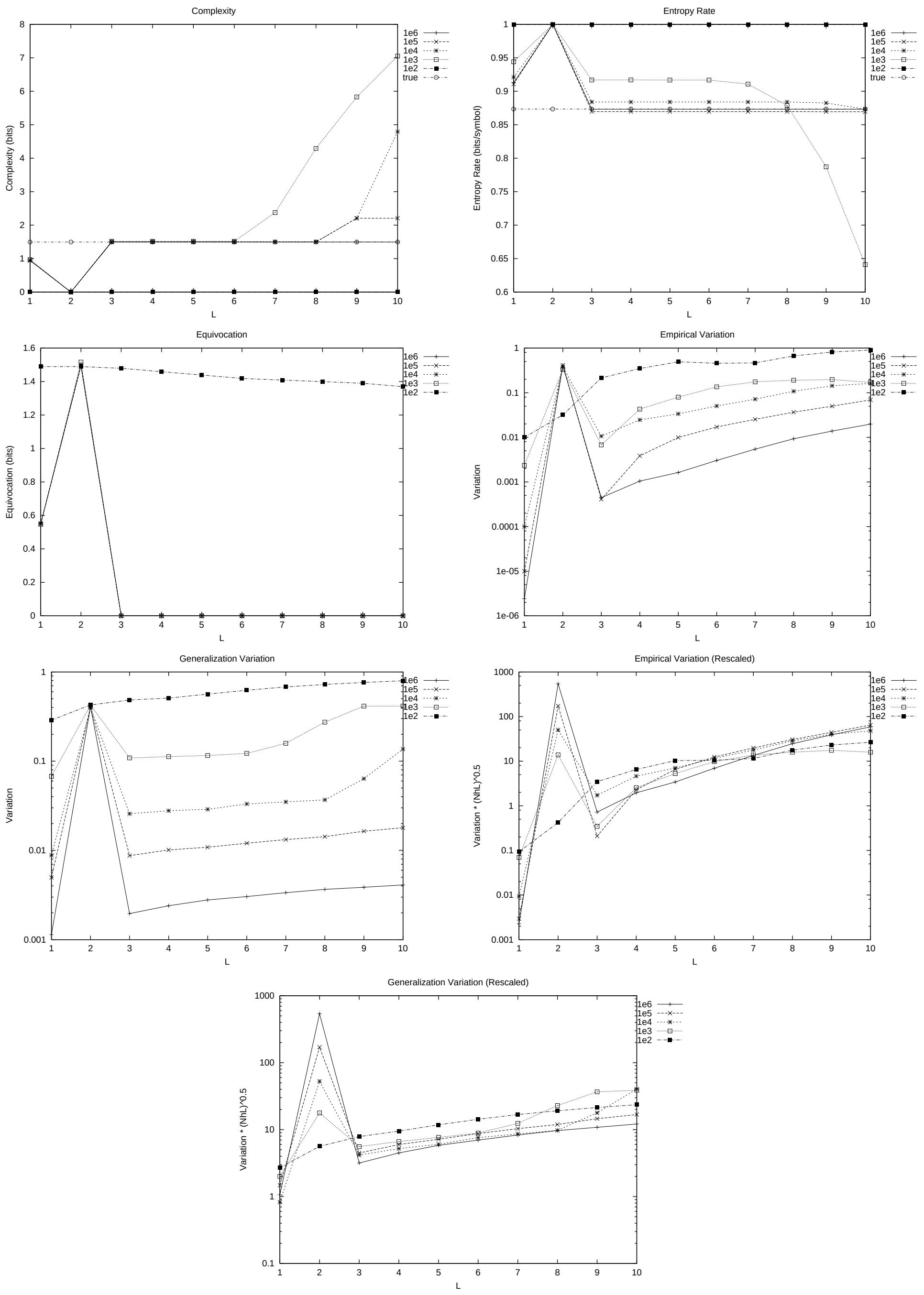

Figure 6: Three-state deterministic hidden Markov model reconstructed—CSSR performance measures as a function of sequence length $L_{\max}$: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{Nh_\mu L}$; and (g) generalization error scaled by $\sqrt{Nh_\mu L}$.

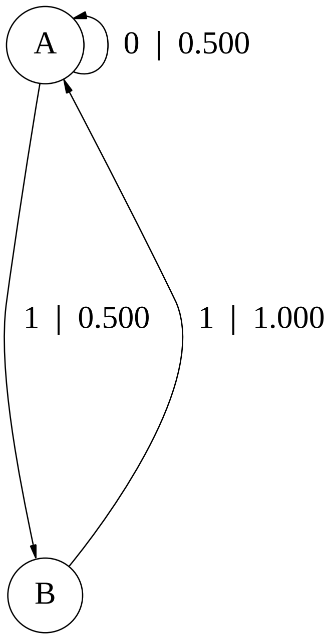

Figure 7: The even process.

in effect, the learning algorithm has to guess the rule generating the forbidden words. (This fact has been used as the basis of a definition of complexity Badii and Politi (1997), **?**.) However, nothing in our analysis of convergence depended on the process of being of finite type. We therefore expect CSSR should learn sofic as well as finite-type processes, and this is indeed the case, as the next two examples demonstrate.

8.3.1 The Even Process

The even language is defined by the following rule: 1s can appear only in blocks of even length, separated by blocks of 0s of any length. It can thus be generated by a process with two states, Figure 7. State A can emit either a 0 or a 1; it returns to itself on 0, and goes to B on 1. State B, however, must emit a 1 and go to A. State A is the only start and the only stop state. The set of irreducible forbidden words therefore consists of odd-length blocks of 1s, bound by 0s on both sides: $\{01^{2k+1}0, k = 1, 2, \ldots\}$. To make this a stochastic process, we need to attach probabilities to A's transitions; we chose to make them equally probable. There is long-range memory here, since, if the last symbol was a 1, the distribution of the next symbol depends on whether it was proceeded by an odd or even number of 1s, and this could lead to dependence on indefinitely remote events. CSSR picked out the architecture right away (Figures 8(a)–8(g)).

Discuss the results contained in the plots.

8.3.2 The Misiurewicz Process

The logistic map, $x_{t+1} = \lambda x_t(x_t - 1)$, $0 \leq x \leq 1$, $0 \leq \lambda \leq 4$, is a continuous-valued, discrete-time dynamical system, one of the bedrocks of nonlinear dynamics (Devaney, 1992). Depending on the value of the control parameter λ, it can display periods of all lengths, chaos, etc. It is often convenient to study the *symbolic dynamics* of this (and other maps), where, instead of looking directly at x, we look at which cell of a partition x is in. (This *measurement partition* should not be confused with

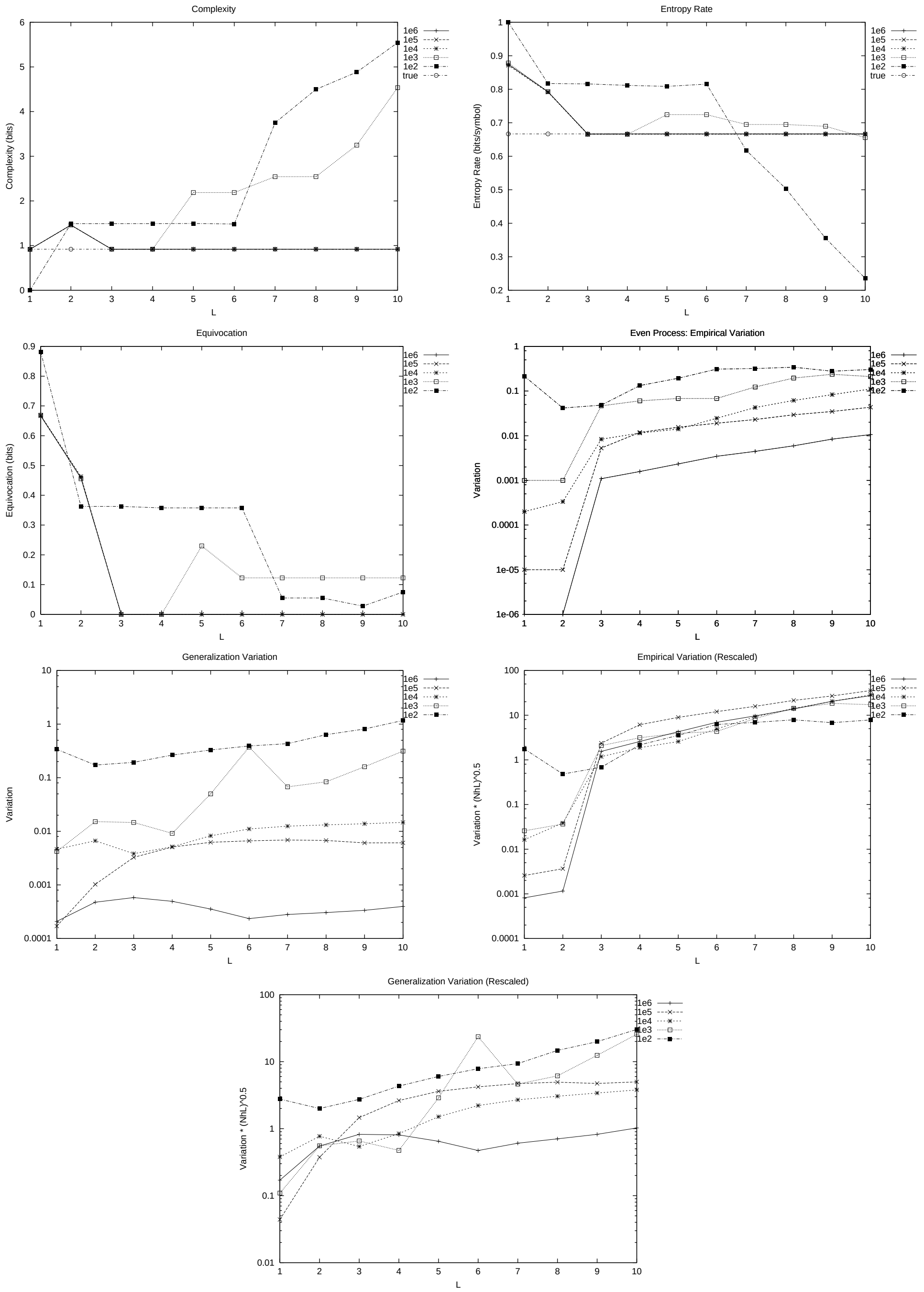

Figure 8: The even process reconstructed—CSSR performance measures as a function of sequence length L_{max}: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{N h_\mu L}$; and (g) generalization error scaled by $\sqrt{N h_\mu L}$.

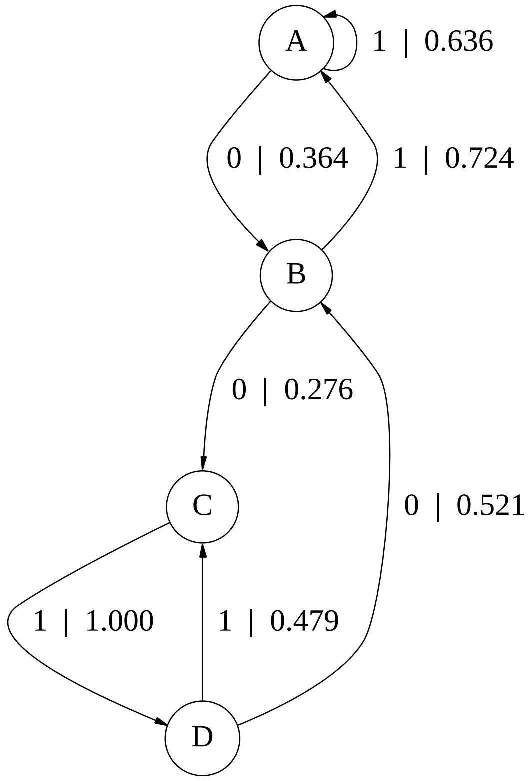

Figure 9: The Misiurewicz process.

the partition of histories which gives us the causal states.) If we chose a simple binary partition, mapping $x < 0.5$ to 0, and $x \geq 0.5$ to 1, then we have a *generating partition*, in the sense that there is a one-to-one correspondence between points x, and the sequence of symbols generated by its future iterates. We can therefore study the discrete-valued stochastic processes without losing any information. When $\lambda = 4$, we can calculate the invariant distribution $\mathrm{P}(X)$ exactly, and find that the process generated by the logistic map is exactly the fair coin process. (For studies of the machines embodied in the logistic map at different λ, see Crutchfield and Young, 1989, 1990, .).

More interesting, and more challenging for our algorithm, is the *Misiurewicz process*, which the logistic map generates when $\lambda \approx 3.9277370017867516$ (Figure 9; Crutchfield, 1992a). The map is chaotic, with rather high entropy rate. What is harder for normal methods to cope with, however, is that the number of irreducible forbidden words of length L grows exponentially with L. This would seem to tax the learning algorithm even more than the previous example, where the number of forbidden words grows only polynomially with L, but in fact we work here too (Figures 10(a)–10(g)).

Discuss the results contained in the plots.

Give true h_μ and C_μ.

Give the ϵ-machines for $L = 1, 2, 3, 4$.

8.4 A Non-Finite Process: The Simple Nondeterministic Source

While our algorithm assumes the process has only finitely many causal states, this is not always so. Context-free languages, or even Markov chains on the integers, have countable infinities of states. Indeed, there are processes which, while they can be represented as finite hidden Markov models,

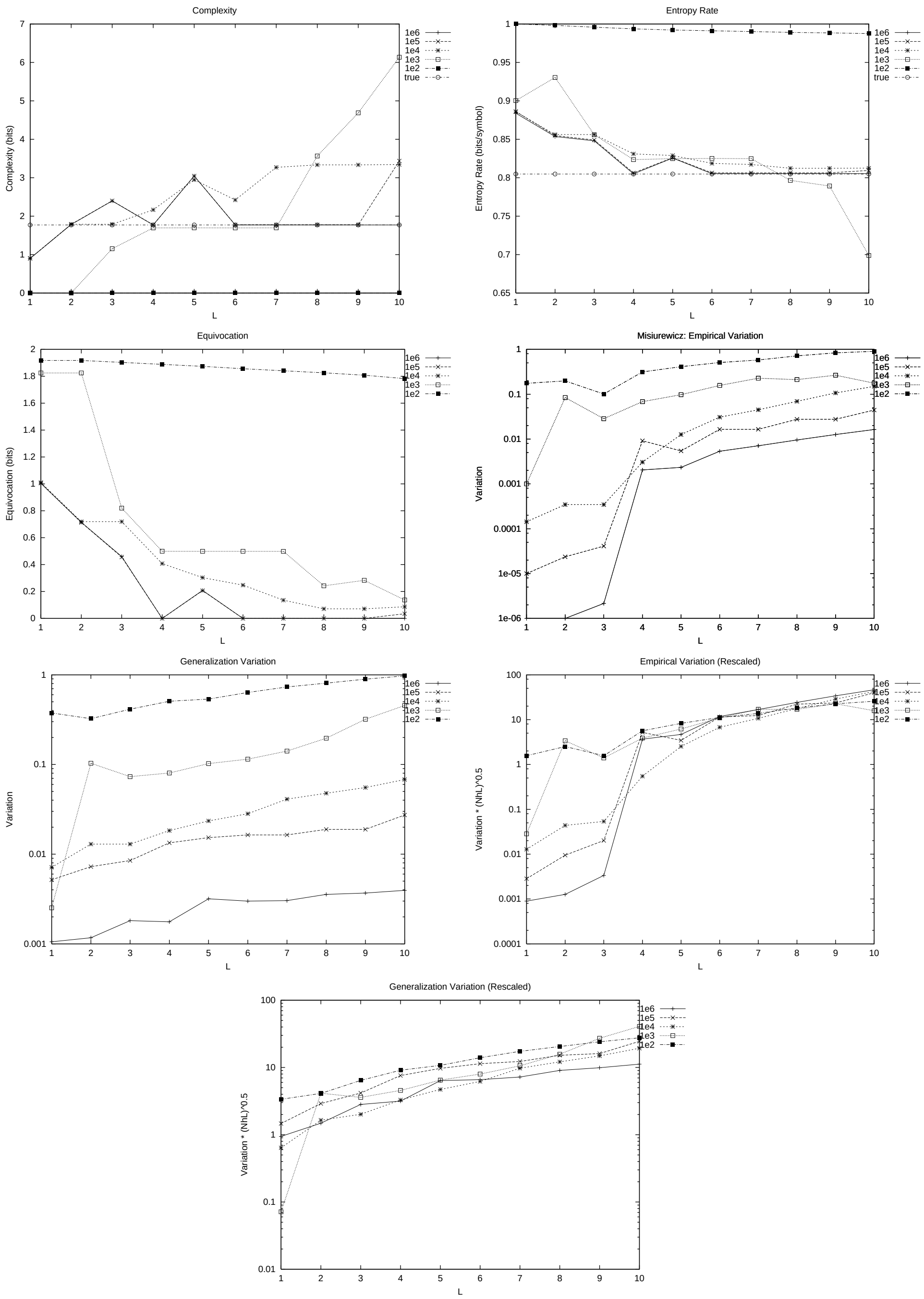

Figure 10: The Misiurewicz process reconstructed—CSSR performance measures as a function of sequence length $L_{\max}$: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{Nh_\mu L}$; and (g) generalization error scaled by $\sqrt{Nh_\mu L}$.

have an infinite number of causal states (Upper, 1997). Now, any finite run of such processes, or even finite set of runs, could be described with only a finite number states that looked causal. We would like a pattern-discovery algorithm to produce increasingly large finite approximations to the infinite state machine, as we give it more data.[2] This is precisely what our algorithm does for a case where we can find the infinite causal state machine analytically.

We have already seen one example of the symbolic dynamics of the logistic map in the Misiurewicz process in Sec. 8.3.2. There we remarked that the best measurement partition on the continuous variable x is the one which divides x at 0.5, because it leads to a topological conjugacy between values of x and infinite sequences of measurement symbols. We also remarked that the invariant distribution of x can be calculated exactly when $\lambda = 4$. Suppose we keep that value of the control parameter, but use a different, suboptimal measurement partition? Specifically we will use a non-generating partition, which maps x to 0 if $x < \hat{x}$, and to 1 if $x \geq \hat{x}$, where $\hat{x}$ is the larger pre-image of 1/2. The resulting discrete stochastic process can be represented by a hidden Markov model with two states, as in Figure 11(a). Notice that the states are *not* deterministic; hence this machine is called the "simple nondeterministic source," or SNS.

It is straightforward to work out the causal states on the basis of the HMM presentation (Crutchfield, 1992b, Upper, 1997); but there are a countable infinity of states (Figure 11(b))! Every 0 encountered takes us back to a resetting state, corresponding in part to the A of Figure 11(a). We leave A on a 1, and each succeeding 1 takes us further down the chain. The probability of emitting a 0 in the k^{th} state in the chain is $k/2(k+1)$. The probability of being in the k^{th} state drops geometrically with k, and the series for the statistical complexity converges (to ≈ 2.71147 bits).

Clearly, reconstruction with any finite L will fail to capture the complete structure of the SNS. What is gratifying is that, as we raise L, we get more and more of the states in the chain, and in general at finite L the inferred machine is visibly a truncation of the full, infinite machine (see Figures 13(a)–13(f) for ϵ-machines reconstructed at values of L from 2 to 9). The behavior of $\widehat{C_\mu}$ (Figure 12(a)) and the equivocation (Figure 12(c)) reflect this: as we raise L, we split the last state in the reconstructed chain, raising $\widehat{C_\mu}$ and lowering the equivocation, though we can never reduce the latter to zero.

Discuss the results contained in the plots.

General observation: It seems that in some of the statistics (various errors) there is a nice scaling once the basic achitecture is inferred. Can we develop a theory for this?

9. Selecting Parameters for the Learning Process

There are two parameters for the learning algorithm that must be chosen: the maximum length L_{max} of history to be considered and the significance level δ of the distribution identity test.

When it comes to chosing the history length, there is a tension between chosing L_{max} long enough to capture the process's structure and chosing L_{max} short enough that histories are not undersampled. As noted in the CSSR convergence proof in Part I, L_{max} must be long enough that every state contains a suffix of length L_{max} or less. For example, in the case of a period-P process, this implies that $L_{max} \geq P$. It would seem that one should simply use as large an L_{max} as possible. However, as mentioned when discussing the empirical error rate, the longer L_{max} is, the less representative our sample of histories. An easy way to see this is as follows. The entropy

2. Actually, what we really want it to do is to eventually guess that it could get away with a much more compact description if it "went up" to a higher-level representations — e.g., modeling with a push-down stack instead of a finite-state process. Such hierarchical ϵ-machine reconstruction is beyond the scope of this method and this paper, but see Crutchfield (1994).

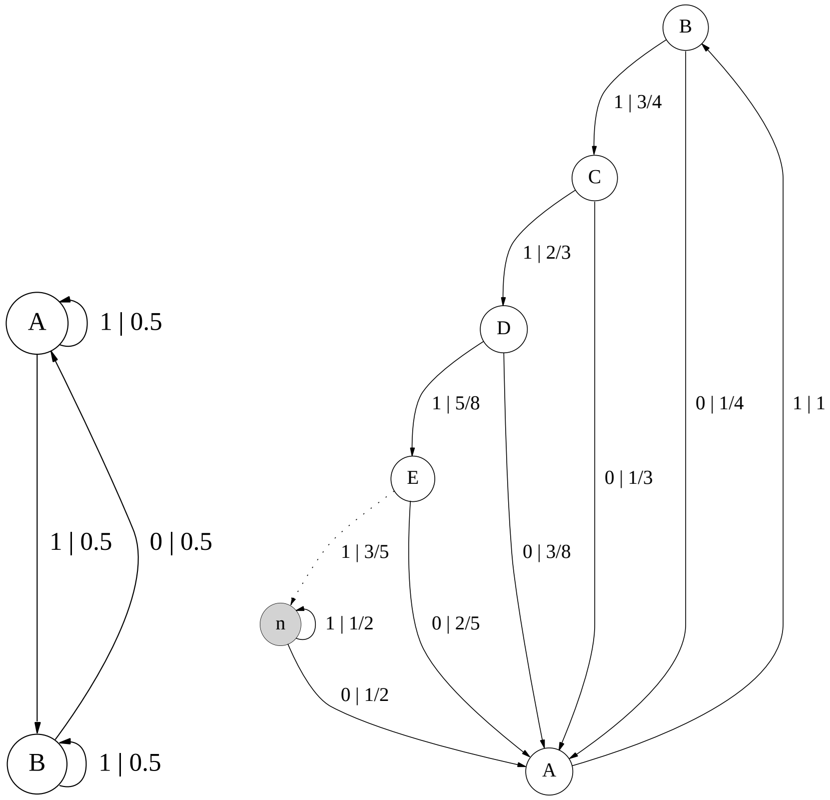

Figure 11: The simple nondeterministic source: (a) its hidden-Markov-model presentation and (b) its ϵ-machine . Note that the ϵ-machine has a countably-infinite series of causal states leading to the asymptotic state labeled "n" (in grey). Ater Crutchfield (1994).

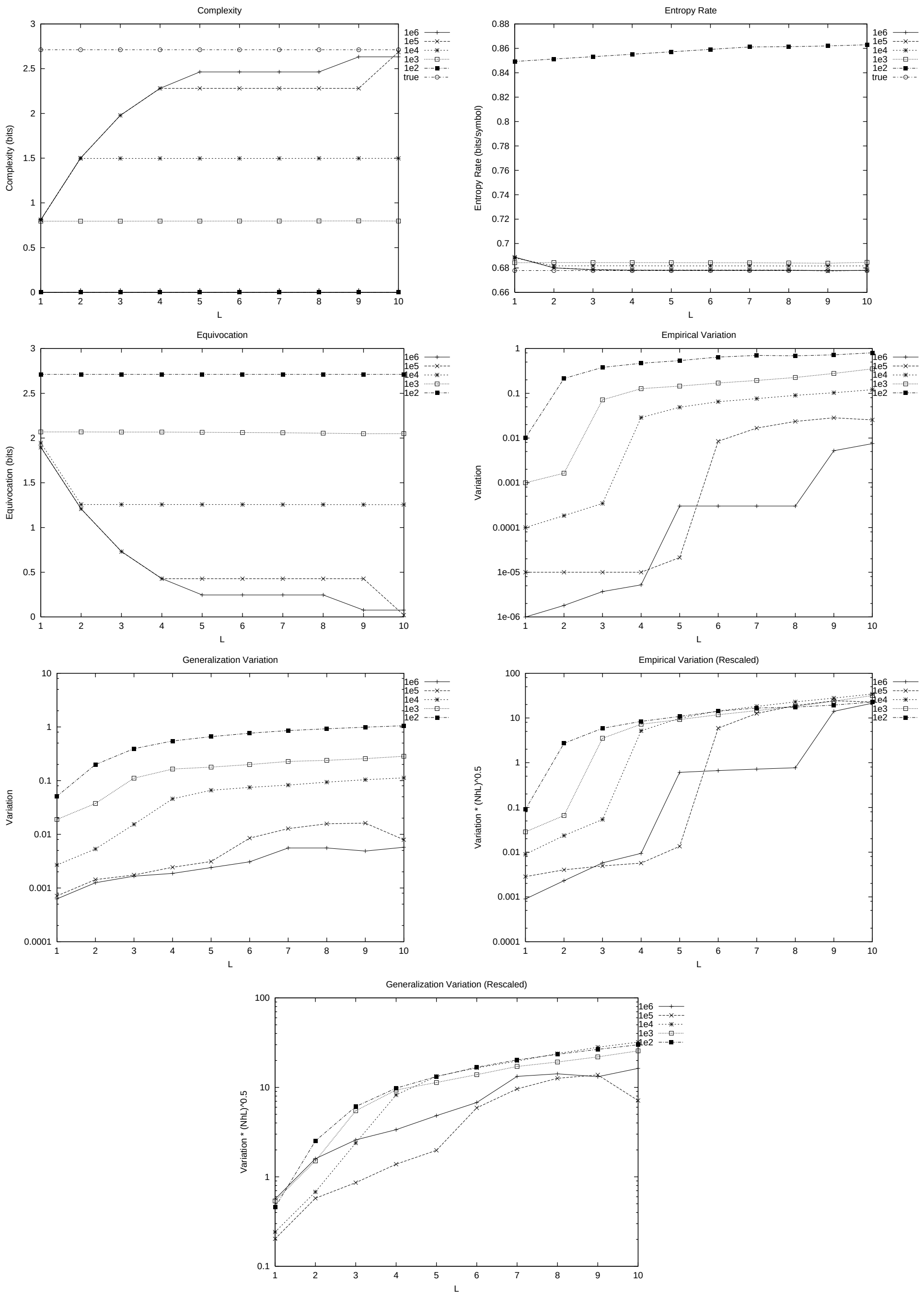

Figure 12: The simple nondeterministic source reconstructed—CSSR performance measures as a function of sequence length L_{max}: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{N h_\mu L}$; and (g) generalization error scaled by $\sqrt{N h_\mu L}$.

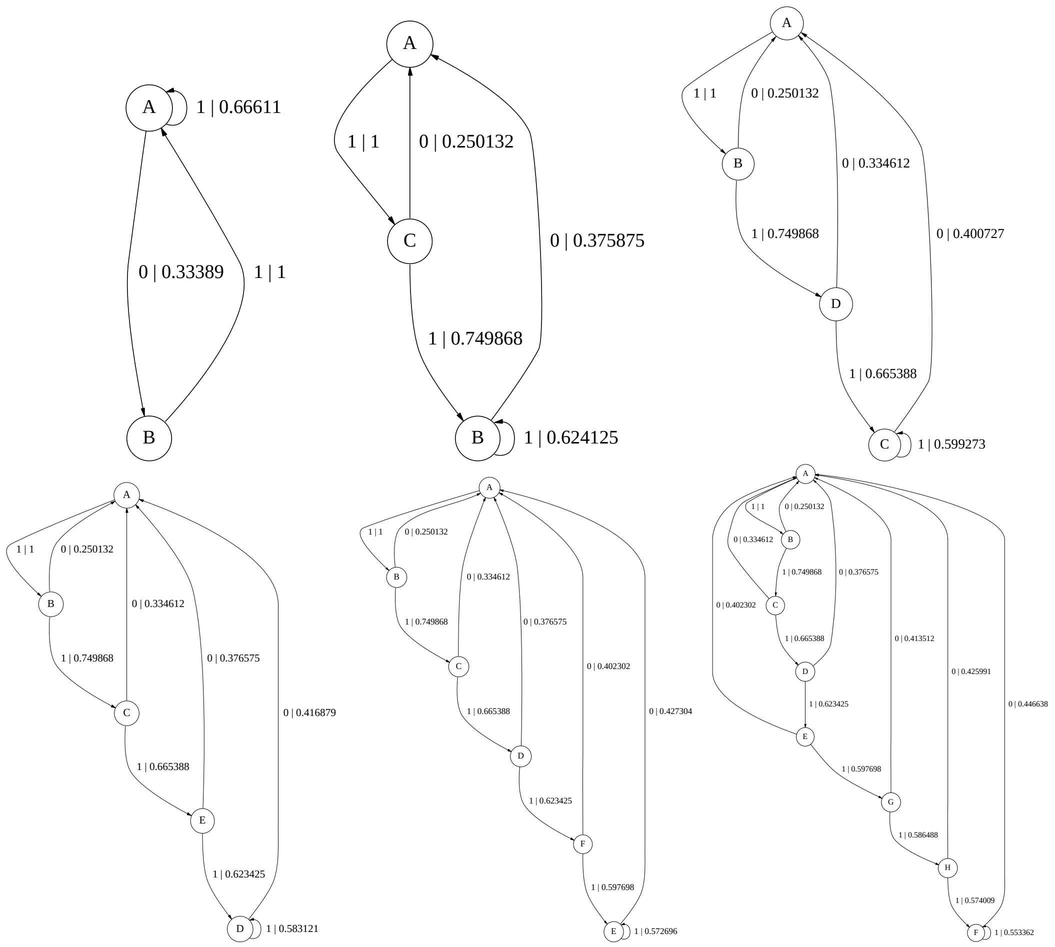

Figure 13: Inferred ϵ-machine for the simple nondeterministic source ($N = 10^6$) with (a) $L = 1$; (b) $L = 2$; (c) $L = 3$; (d) $L = 4$; (e) $L = 5$; and (f) $L = 9$.

ergodic theorem[3] tells us that

$$\frac{1}{L} \log \mathrm{P}(\overrightarrow{S}^L = s^L) \rightarrow_{L \rightarrow infty} h_\mu \ , \tag{21}$$

for almost all s^L; that is, for those sequences in the process's *typical set*. For large L, then, there are about 2^{Lh_μ} strings with substantial probability. This is an exponentially small fraction of all possible strings $\mathcal{A}^L$, but still an exponentially growing *number*. To get an adequate sample of each string therefore demands an exponentially growing number of samples.

Put the other way around, the maximum history length must grow no more than logarithmically with the number of data samples available. Let us write $L_{max}(N)$ for the maximum $L_{\max}$ we can use when we have N data points. If the observed process satisfies the weak Bernoulli property, which random functions of irreducible Markov chains do, Marton and Shields (1994) showed that $L_{\max}(N) \leq \log N/(h_\mu + \varepsilon)$, for some positive ε, is a sufficient condition for sequence probability estimates to converge. One might wonder if cleverness could improve this; but they also showed that, if $L_{\max}(N) \geq \log N/h_\mu$, probability estimates over L_{max}-words do not converge. Applying this bound requires knowledge of h_μ, but note that, if we bound it by $\log k$, we get a more conservative result, so we will certainly be safe if we do so. Moreover, experience shows that the ϵ-machine we estimate typically yields good values of h_μ even for very short values of L and even when the ϵ-machine is otherwise a poor estimate of the process's structure.[4]

Some of the above claims seem wrong for periodic processes. Punt on this or remark?

As to the significance level δ, we have seen in the proof of convergence in Part I that asymptotically δ does not matter — histories with distinct morphs will be split, histories with the same morph will be joined — but, in the meanwhile, it does effect the kind of error made. The significance level is the probability that two samples, drawn from the same distribution, would give at least as large a value of the test statistic. If this value is small, therefore, we will split only on large values of the test statistic. Therefore only glaring (as it were) distinctions between empirical distributions will lead to splits. To become sensitive to very small distinctions between states, we need to turn up the significance level, at which point we become more likely to split due to fluctuations. By controlling the significance level, we indicate how much we are willing to risk seeing "structure" (more causal states or transitions) that is not there. That is, we create states merely due to sampling error, rather than real differences in conditional probabilities. We found $\delta = 0.001$ to work well, but we were interested in fairly large L.

I don't understand this last comment.

10. Comparison with Other Methods

We discuss how our algorithm differs from two rival types of procedure: conventional hidden Markov model algorithms and variable-length Markov models, originating with Rissanen's "context" algorithm. A formal comparison between CSSR and prior subtree-merging ϵ-machine reconstruction algorithms was given in Part I (Shalizi et al., 2002). All of these procedures, along with the present algorithm, attempt to find a set of Markovian states that are capable of generating the data. It is important therefore to be clear on their distinctions and relative merits.

The most important distinction between causal-state algorithms and other algorithms is that the former attempt to infer the causal states and the latter do not. That is, causal-state algorithms have a well defined structural goal, with well understood properties — particularly optimal prediction,

3. This is also know as the Shannon-MacMillan-Breiman theorem or as the asymptotic equipartition property, (Cover and Thomas, 1991) AEP, depending on the field.
4. There is a conjecture (D. Eric Smith, personal communication, 2001) that, if we write $h_\mu(L)$ for the estimate of h_μ using length L strings, the smallest L sufficient to resolve all the causal states is the smallest L' such that $h_\mu(L') = h_\mu(L'+1)$ and for all larger $L \geq L'$. That is, the shortest adequate L is the one at which the predictability gain (Crutchfield and Feldman, 2001) becomes zero. While plausible, this is unproven.

minimality, and the like, as noted in Part I. In short, the ϵ-machine *is* how one should represent patterns in time series; by tying the algorithm to this well developed theory of patterns, we gain an idea of what the results *should* be like (Shalizi and Crutchfield, 2001) in ideal settings and so one has a clear picture of the *goals* of any modeling or learning process.

10.1 Conventional HMM Algorithms

Standard algorithms for fitting HMMs to data — e.g., the many variations on the expectation-maximization algorithm — all assume a fixed number of states. Many, in addition, make assumptions about the process's architecture — what *kinds* of states there are, which states emit what symbols, and what states are connected to what other states. It will be evident by this point that CSSR does not need to make such assumptions. The causal architecture is inferred from observed behavior, rather than being guessed at by (human) trial and error. The EM algorithm is a good device when the correct structure is known; but it does parameter estimation, *not* pattern discovery.

As an added bonus, the states inferred by our algorithm are always effective and future-resolving. After seeing only a finite number of symbols, we can determine with certainty what state the process is in (effectiveness) and thereafter never again become uncertain about the state (future-resolving). Conventional HMM algorithms do not have this property, which can considerably complicate calibration and leads to higher than necessary prediction errors. (That is, the estimated h_μ is larger than necessary.)

There are some published HMM algorithms that attempt to infer architecture along with transition probabilities, which either start with a large number of states and merge them (Stolcke and Omohundro, 1993) or with a small number of states and increase them by duplication (Fujiwara et al., 1994). In no case known to us does the architecture change in response to anything more than the distribution for the next symbol, i.e., in our terms, the length-1 morph. And in no case is there any structural justification for the kinds of states inferred, such as computational mechanics provides.

Unsworth et al. (2001) propose using a first-order Markov model to produce surrogate data for testing for deterministic nonlinearity in time series and suggest using a KS test to gauge the appropriate level of discretization. The overlap between their proposal and CSSR is essentially confined to an overlap of terminology.

10.2 Variable-Length Markov Models

While the CSSR algorithm could be described as one which finds a "variable-length Markov model" in that it estimates the Markovian order of the hidden process, that phrase is the conventional name for the "context" algorithm of Rissanen and its descendants (Rissanen, 1983, Bühlmann and Wyner, 1999, Willems et al., 1995, Tino and Dorffner, 2001). There are points of similarity between those algorithms and CSSR. Both work by examining increasingly long histories, splitting off states as thresholds of error are exceeded and both infer architecture as well as parameters. However, CSSR has several important advantages over the context approach. First, we use the known properties of the states we are looking for to guide search. Second, rather than creating states when some (arbitrary) threshold for, say, divergence is reached, we use well understood statistical tests for identity of distribution. Third, context algorithms suffer from the following defect. The states they infer are what one might call *suffixal* — all states consist of all and only the histories ending in a particular suffix. For many, many processes, the causal states are *not* suffixal or, at least, not all of them are. In these cases, an infinite number of "contexts" are needed to represent a single causal state.

We briefly alluded to this problem when discussing the noisy period-two process. Recall that it has two causal states, labeled A and B in Figure 3. Any history terminated by a 0, followed by an even number of 1s, belongs to state A; and any history terminated by a 0, followed by an odd number of 1s, belongs to B. Clearly, therefore, both A and B correspond to an *infinite* number of suffix states, since there is no suffix common to all and only the histories belong in either state. Reflection shows that adding noise to a single transition in any periodic process will have the same effect. If a context (suffix-tree) algorithm is used to try to learn such processes, the number of states it will estimate will diverge as a function of the history length. As we have pointed out above and previously Shalizi and Crutchfield (2001), this leads to blatantly incorrect structural inferences. For example, though the fair-coin process has a single causal state, context algorithms infer an infinite number of suffix-states. One practical result is that data-compression methods based on this class of algorithm "compress" random files into those that are substantially larger than the original.

As a check on these conclusions, we implemented a context algorithm of the Rissanen type. For fairness, it used the same statistical test as CSSR to decide when to add new contexts. We then ran it on both the noisy period-two process (Figures 14(a)–14(g)) and the three-state deterministic HMM (Figures 15(a)–15(g)). In both cases, the context algorithm produces drastically more states than CSSR. In fact, the number of context-states grows exponentially with L_{max}. Moreover, it suffers badly from over-fitting. [**Note to readers: the generalization error plots for TMD are wrong beyond $L = 6$; there were so many states the program which did the calculation crashed! I'll fix this by the next iteration of the paper. CRS**] This is particularly notable in the case of the three-state deterministic HMM, since it has a perfectly good context-tree representation. The context algorithm *finds* this, but rejects it, since it attempts to do better by over-refinement. This is not a problem if one wants to do lossless data compression of relatively compressible (low entropy-rate) data source, for which, indeed, context trees can work very well. This failure, however, almost entirely precludes the use of context methods in prediction.

Discuss the plots. Make direct plot-by-plot comparisons to CSSR plots on NPT example.

Discuss the plots. Make direct plot-by-plot comparisons to CSSR plots on the TMD example.

11. Conclusion

11.1 Failure modes

It seems only fair, having just critiqued context methods, to turn the same critical light on CSSR. How can CSSR not work?

Consider two-biased coins that are randomly switch. The ϵ-machine for this process consists of an infinite number of transient causal states and two recurrent causal states. The finite-data approximations produced by CSSR, though optimal in the sense used here, do not look "like" the two biased coins ϵ-machine , since the approximate ϵ-machines mix the transient (or "synchronization") causal states and recurrent causal states.

How to get around this? Answer: Extend CSSR to account for and directly reconstruct a minimal "nondeterministic" ϵ-machine using the approach outlined in (Crutchfield, 1994, Upper, 1997). To regain the benefits of determinism, one can systematically convert between deterministic and nondeterministic ϵ-machine (Upper, 1997). The resulting algorithms is considerably more sophisticated and so will be the subject of future presentations elsewhere.

11.2 Applications

Speech, where the use and benefits of HMMs are well established. What would using deterministic models give in this area?

Potentially, our algorithm could be applied in any domain where HMMs have proved their value (e.g., bioinformatics (Baldi and Brunak, 1998)), or where there are poorly-understood processes generating sequential data, in which one wishes to find non-obvious patterns.

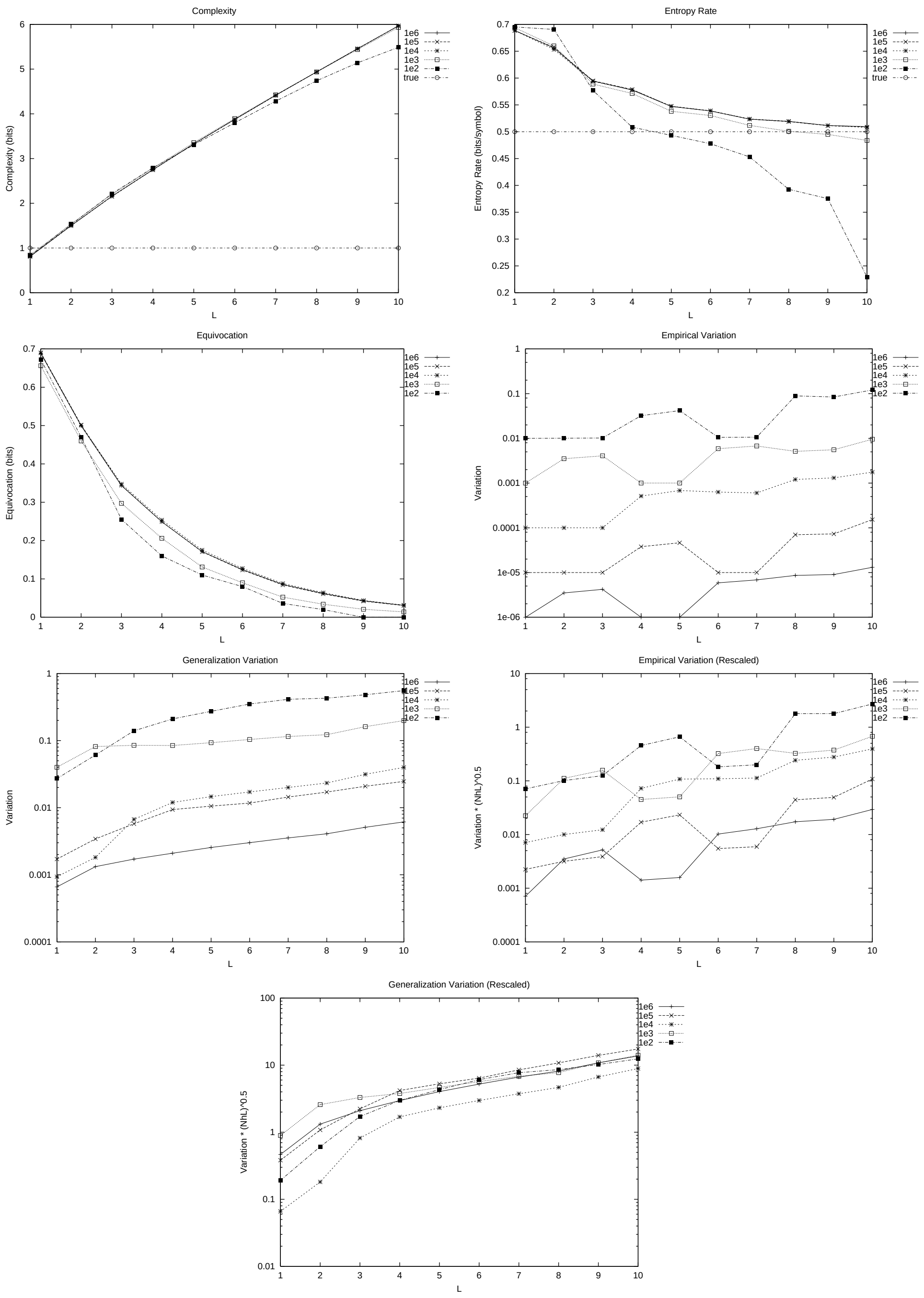

Figure 14: Context-tree reconstruction of noisy period-two process—context-tree performance measures as a function of sequence length L_{max}: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; (f) empirical error scaled by $\sqrt{N h_\mu L}$; and (g) generalization error scaled by $\sqrt{N h_\mu L}$.

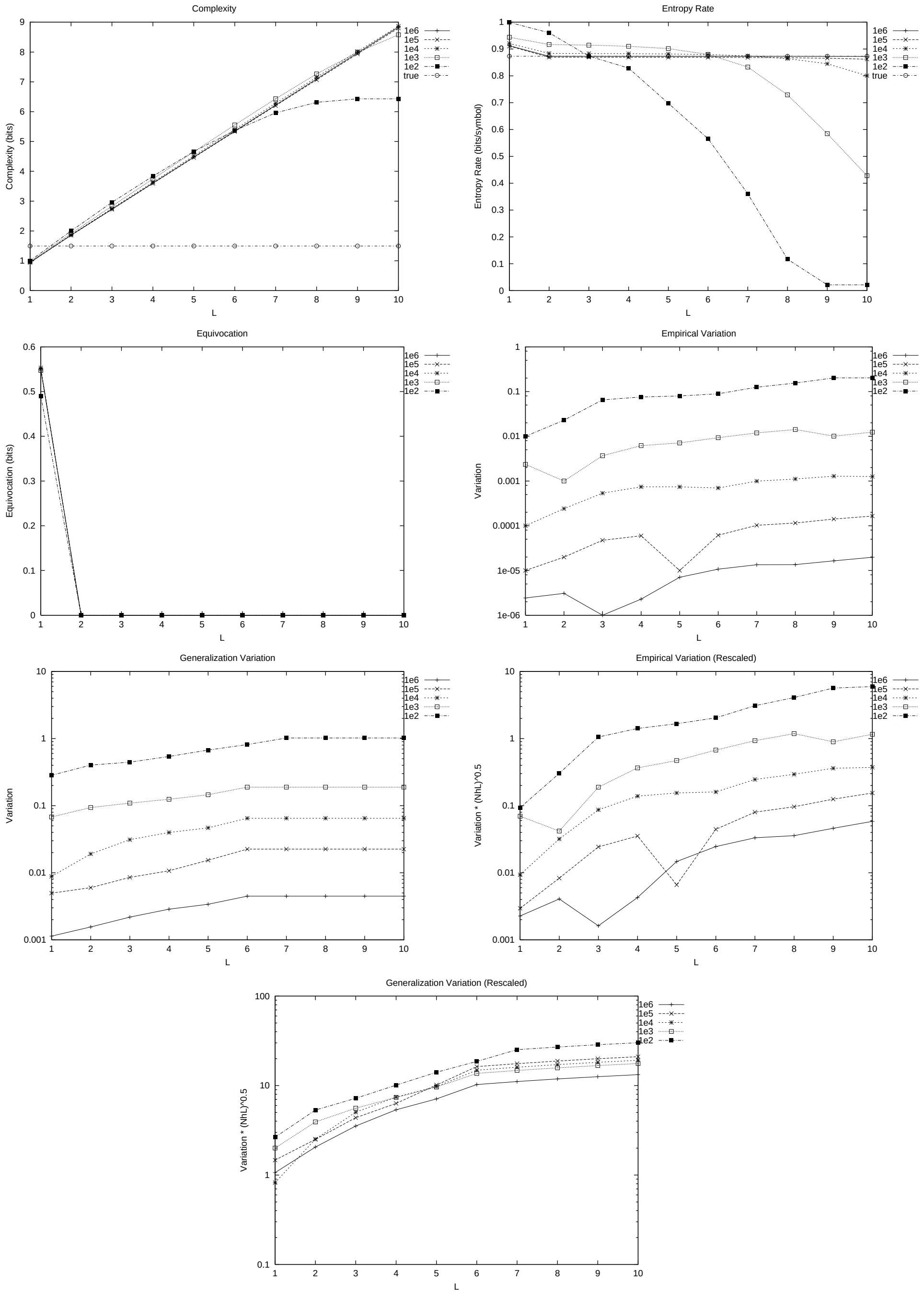

Figure 15: Context-tree reconstruction of three-state deterministic hidden Markov model—Context-tree performance measures as a function of sequence length $L_{\max}$: (a) true and estimated statistical complexity $C_\mu(\widehat{\mathrm{M}})$; (b) true and estimated entropy rate $h_\mu(\widehat{\mathrm{M}})$; (c) equivocation; (d) empirical error; (e) generalization error; **[The lines become flat beyond $L = 6$ because there were so many states, the program calculating generalization error crashed. This will be fixed before the next iteration of the MS.]** (f) empirical error scaled by $\sqrt{N h_\mu L}$; and (g) generalization error scaled by $\sqrt{N h_\mu L}$.

Synchrony in neural systems [Causl Sync paper].
Emergence of cooperation in MAS [MG paper].

11.3 Summary

In Parts I and II of this work, we presented a new algorithm for pattern discovery in time series. In short, given samples of a conditionally stationary source, the CSSR algorithm reliably infers the correct causal model — the ϵ-machine — of the source. Here in Part II we demonstrated empirically that CSSR works reliably on both finite-type and strictly sofic processes and reasonably on infinite-state processes. Moreover, CSSR performs significantly better than conventional HMM algorithmsa and context-tree methods. Finally, we indicated how to extend the approach to even more complex processes that these classes.

Acknowledgments

This work was supported by SFI's Dynamics of Learning project, under DARPA cooperative agreement F30602-00-2-0583, and SFI's Network Dynamics Program, supported by Intel Corporation. KLS received additional support from the NSF-SFI Research Experience for Undergraduates program for her summer 2000 internship. We thank K. Kedi for assistance in programming and comments on the manuscript.

References

Remo Badii and Antonio Politi. *Complexity: Hierarchical Structures and Scaling in Physics*, volume 6 of *Cambridge Nonlinear Science Series*. Cambridge University Press, Cambridge, 1997.

Pierre Baldi and Søren Brunak. *Bioinformatics: The Machine Learning Approach*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Massachusetts, 1998.

Peter Bühlmann and Abraham J. Wyner. Variable length Markov chains. *Annals of Statistics*, 27: 480–513, 1999. URL `http://www.stat.berkeley.edu/tech-reports/479.abstract1`.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

James P. Crutchfield. Semantics and thermodynamics. In Martin Casdagli and Stephen Eubank, editors, *Nonlinear Modeling and Forecasting*, volume 12 of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 317–359, Reading, Massachusetts, 1992a. Addison-Wesley.

James P. Crutchfield. Unreconstructible at any radius. *Physics Letters A*, 171:52–60, 1992b.

James P. Crutchfield. The calculi of emergence: Computation, dynamics, and induction. *Physica D*, 75:11–54, 1994.

James P. Crutchfield and David P. Feldman. Regularities unseen, randomness observed: Levels of entropy convergence. *CHAOS*, in press, 2001. E-print, arxiv.org, cond-mat/0102181.

James P. Crutchfield and Karl Young. Inferring statistical complexity. *Physical Review Letters*, 63: 105–108, 1989.

James P. Crutchfield and Karl Young. Computation at the onset of chaos. In Wojciech H. Zurek, editor, *Complexity, Entropy, and the Physics of Information*, volume 8 of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 223–269, Reading, Massachusetts, 1990. Addison-Wesley.

Robert L. Devaney. *A First Course in Chaotic Dynamical Systems: Theory and Experiment*. Addison-Wesley, Reading, Mass., 1992.

David P. Feldman and James P. Crutchfield. Discovering noncritical organization: Statistical mechanical, information theoretic, and computational views of patterns in simple one-dimensional spin systems. *Journal of Statistical Physics*, submitted, 1998. URL `http://www.santafe.edu/projects/CompMech/papers/DNCO.html`. Santa Fe Institute Working Paper 98-04-026.

Y. Fujiwara, M. Asogawa, and A. Konagaya. Stochastic motif extraction using hidden Markov models. In *ISMB 94: Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, pages 138–146, Menlo Park, California, 1994. AAAI Press.

P. Grassberger. Finite sample corrections to entropy and dimension estimates. *Physics Letters A*, 128:369–73, 1988.

P. Grassberger. Estimating the information content of symbol sequences and efficient codes. *IEEE Transactions on Information Theory*, 35:669–674, 1989.

R. M. Gray. *Entropy and Information Theory*. Springer-Verlag, New York, 1990.

H. Herzel, A.O. Schmitt, and W. Ebeling. Finite sample effects in sequence analysis. *Pergamon*, 1994.

Solomon Kullback. *Information Theory and Statistics*. Dover Books, New York, 2nd edition, 1968. First edition New York: Wiley, 1959.

Katalin Marton and Paul C. Shields. Entropy and the consistent estimation of joint distributions. *Annals of Probability*, 23:960–977, 1994. See also an important Correction, *Annals of Probability*, **24** (1996): 541–545.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, 2nd edition, 1992. URL `http://lib-www.lanl.gov/numerical/`.

Jorma Rissanen. A universal data compression system. *IEEE Transactions in Information Theory*, IT-29:656–664, 1983.

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

Cosma Rohilla Shalizi and James P. Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of Statistical Physics*, 104:819–881, 2001.

Cosma Rohilla Shalizi, Kristina Lisa Shalizi, and James P. Crutchfield. Pattern discovery in time series, Part I: Theory, algorithm, analysis, and convergence. *Journal of Machine Learning Research*, submitted, 2002.

R. Shaw. *The Dripping Faucet as a Model Chaotic System*. Aerial Press, Santa Cruz, California, 1984.

A. Stolcke and S. Omohundro. Hidden Markov model induction by Bayesian model merging. In Stephen José Hanson, J. D. Gocwn, and C. Lee Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 11–18. Morgan Kaufmann, San Mateo, California, 1993.

Peter Tino and Georg Dorffner. Predicting the future of discrete sequences from fractal representations of the past. *Machine Learning*, 45:187–217, 2001.

C. P. Unsworth, M. R. Cowper, S. McLaughlin, and B. Mulgrew. A new method to detect nonlinearity in a time-series: Synthesizing surrogate data using a Kolmogorov-Smirnoff tested, hidden Markov model. *Physica D*, 155:51–68, 2001.

Daniel R. Upper. *Theory and Algorithms for Hidden Markov Models and Generalized Hidden Markov Models*. PhD thesis, University of California, Berkeley, 1997. URL `http://www.santafe.edu/projects/CompMech/`.

Benjamin Weiss. Subshifts of finite type and sofic systems. *Monatshefte für Mathematik*, 77:462–474, 1973.

Frans Willems, Yuri Shtarkov, and Tjalling Tjalkens. The context-tree weighting method: Basic properties. *IEEE Transactions on Information Theory*, IT-41:653–664, 1995.

Karl Young and James P. Crutchfield. Fluctuation spectroscopy. *Chaos, Solitons, and Fractals*, 4: 5–39, 1993.