

Rigorous computation of topological entropy with respect to a finite partition

Gary Froyland ^{a,1} Oliver Junge ^b Gunter Ochs ^{c,2}

^a*Department of Mathematics and Computer Science, University of Paderborn,
33095 Paderborn, Germany, froyland@upb.de*

^b*Department of Mathematics and Computer Science, University of Paderborn,
33095 Paderborn, Germany, junge@upb.de*

^c*Institute for Dynamical Systems, University of Bremen, 28334 Bremen,
Germany, gunter@math.uni-bremen.de*

Abstract

We present a method to compute rigorous upper bounds for the topological entropy $h(T, \mathcal{A})$ of a continuous map T with respect to a fixed (coarse) partition of the phase space \mathcal{A} . Long trajectories are not used; rather a single application of T to the phase space produces a topological Markov chain which contains all orbits of T , plus some additional spurious orbits. By considering the Markov chain as a directed graph, and labelling the arcs according to the fixed partition, one constructs a sofic shift with topological entropy greater than or equal to $h(T, \mathcal{A})$. To exactly compute the entropy of the sofic shift, we produce a subshift of finite type with equal entropy via a standard technique; the exact entropy calculation for subshifts is then straightforward. We prove that the upper bounds converge monotonically to $h(T, \mathcal{A})$ as the topological Markov chains become increasingly accurate. The entire procedure is completely automatic.

Key words: topological entropy, topological Markov chain, subshift of finite type, sofic shift, right-resolving presentation

2000 AMS subject classification: 37B40 (Primary), 94A17, 37M25 (Secondary)

¹ Research supported by the Deutsche Forschungsgemeinschaft under Grant De 448/5-4.

² Research supported by the Deutsche Forschungsgemeinschaft under Grant Ar 137/12-2.

1 Introduction

The topological entropy $h(T)$ (see Walters [1] for a definition) of a continuous map $T : M \rightarrow M$ on a compact phase space M is difficult to estimate. It is even more difficult to obtain rigorous bounds. This difficulty stems from the fact that it is numerically infeasible to estimate the topological entropy directly from standard definitions using open covers or ϵ -separated/spanning sets. Most numerical approaches have focussed on one-dimensional mappings, taking advantage of the *kneading theory* of unimodal maps [2–4]. Another approach [5,6] in one dimension is to approximate a general map by a Markov map, for which the entropy is relatively easily calculated. Baldwin and Slaminka [7] improve an approach [8] where the topological entropy is estimated as the logarithmic growth rate of the one-dimensional variation of T^n with n . This approach has been extended to higher dimensions by Newhouse and Pignataro [9] where one considers the growth rate of the volume of submanifolds. Chen *et al.* [10] suggest a method of providing sharp lower bounds for the topological entropy of chaotic saddles and attractors based on successive preimages of the map. Other approaches in higher dimensions include counting the number of periodic points of a given period (this assumes that a result [11] for Axiom A systems holds in greater generality), and various methods of constructing good “grammars” (essentially approximations of Markov partitions); see [12] for a careful application of these approaches to the Hénon map.

With the exception of [10], there are no methods applicable to higher dimensional systems which provide rigorous bounds for the entropy ([10] provides a lower bound). While in principle, the approach of [9] may be applied to systems with more than one expanding direction, the computation of exponential increases in volume of images of submanifolds quickly becomes infeasible.

Our approach provides a rigorous upper bound for the entropy of a multidimensional system with respect to a fixed partition. The theoretical results are completely general with no restrictions on the map other than continuity. In contrast to [9], our method can also efficiently handle systems with more than one expanding direction.

The remainder of this section defines the objects to be estimated and their relationships to known quantities. In Section 2 we describe the main algorithm and state the corresponding theoretical results. Section 3 contains numerical examples and comparisons of our new method in a variety of situations. Implementational details are discussed in Appendix A and proofs of the main results are given in Appendix B.

1.1 Preliminary definitions

An intuitive definition of topological entropy is as follows: Partition M into a collection of sets $\mathcal{Q} = \{A_1, \dots, A_q\}$, and consider orbits $\mathcal{O}_N(x) = \{x, Tx, \dots, T^{N-1}x\}$ of length N . If $T^i x \in A_{a_i}$ for $i = 0, \dots, N-1$, then $\mathcal{O}_N(x)$ gives rise to the N -string $[a_0, a_1, \dots, a_{N-1}]$. We form the collection of all possible N -strings

$$\mathcal{W}_N(T, \mathcal{A}) = \{[a_0, a_1, \dots, a_{N-1}] : \exists x \in M \text{ s.t. } T^i x \in A_{a_i}, 0 \leq i < N\} \quad (1)$$

and consider $\log |\mathcal{W}_N(T, \mathcal{A})|/N$, which describes the growth rate of the number of distinct symbol strings generated by orbits of T with increasing length ($|\mathcal{W}_N(T, \mathcal{A})|$ denotes the cardinality of the set $\mathcal{W}_N(T, \mathcal{A})$). We define the quantity

$$h^*(T, \mathcal{A}) := \lim_{N \rightarrow \infty} \frac{\log |\mathcal{W}_N(T, \mathcal{A})|}{N} = \inf_{N \geq 1} \frac{\log |\mathcal{W}_N(T, \mathcal{A})|}{N}. \quad (2)$$

It is our aim to compute rigorous upper bounds for $h^*(T, \mathcal{A})$. The following proposition relates $h^*(T, \mathcal{A})$ to the standard topological entropy $h(T)$.

Definition 1 A partition \mathcal{A} is called generating, if $\bigvee_{i=0}^{\infty} T^{-i} \mathcal{A} = \mathfrak{B}$ (non-invertible maps) or $\bigvee_{i=-\infty}^{\infty} T^i \mathcal{A} = \mathfrak{B}$ (invertible maps) where \mathfrak{B} denotes the Borel σ -algebra.

Proposition 2 (1) If \mathcal{A} is a generating partition, then

$$h(T) \leq h^*(T, \mathcal{A}).$$

(2)

$$h(T) \leq \liminf_{\text{diam } \mathcal{A} \rightarrow 0} h^*(T, \mathcal{A}).$$

PROOF. See proofs section.

Part (ii) says that by refining arbitrary partitions \mathcal{A} , we obtain an upper bound for $h(T)$ in the limit. In practice, if a generating partition is known, we will use this. If a generating partition is not known, a geometrical study of the dynamics often provides suggestions for partitions which are near to generating (see for example, the Hénon map analysis in Section 3.3). Even in cases where \mathcal{A} is not a generating partition, our numerical studies indicate that $h^*(T, \mathcal{A})$ still provides good estimates of the topological entropy.

To summarise: we will describe a method for computing a rigorous upper bound for the topological entropy $h^*(T, \mathcal{A})$ with respect to a fixed partition \mathcal{A} . The method may be applied to any continuous multidimensional map, without any restriction on the dimension of unstable manifolds or assumptions such as hyperbolicity. In practice, we find that the obtained values are good estimates of the topological entropy $h(T)$, even when the underlying partition \mathcal{A} is not generating. Moreover, our algorithm is very efficient in terms of computing time. Alternative existing methods of entropy computation become extremely inefficient in higher dimensions, especially when the dimension of the unstable manifold exceeds one.

2 The Algorithm

We select a coarse partition \mathcal{A} and will produce upper bounds for $h^*(T, \mathcal{A})$. If there are good choices for \mathcal{A} based on the dynamics, we will use these; otherwise, we will choose a simple partition, making sure that it contains enough sets, based on bounds we now discuss.

If \mathcal{A} contains m sets, then clearly $h^*(T, \mathcal{A}) \leq \log m$; thus, an important condition for a suitable partition \mathcal{A} is that $\log m \geq h(T)$. Some rough upper bounds for $h(T)$, where T is a differentiable map on a d -dimensional manifold are

Theorem 3 (1) $h(T) \leq \max\{0, d \log \sup_{x \in M} \|D_x T\|\}$; Bowen [13],
 (2) $h(T) \leq \log \max_{x \in M} \max_{L \subset T_x M} |\det D_x T|_L|$, with M smooth and compact; S. Katok [14].

If we have no information on how large $h(T)$ may be, we can use Theorem 3 as a guide for the number of sets \mathcal{A} should contain.

2.1 An outline of the algorithm

- (1) Choose a partition \mathcal{B} (much finer than \mathcal{A}), and define a topological Markov chain with respect to \mathcal{B} such that all words generated by T are also generated by the topological Markov chain. The entropy of the topological Markov chain (with respect to the partition \mathcal{A}) will be computed exactly and will provide an upper bound for $h^*(T, \mathcal{A})$.
- (2) The set of all words generated by orbits of the topological Markov chain forms a *sofic shift*. The entropy of this sofic shift equals the entropy of the topological Markov chain with respect to \mathcal{A} .
- (3) To compute the entropy of the sofic shift, we “present” the sofic shift in the form of a *right-resolving presentation* (a labelled graph with special properties). The entropy of the sofic shift is then easily computed from

the maximal eigenvalue of the adjacency matrix for this right-resolving presentation.

In practice, one computes the $(0,1)$ adjacency matrix for the topological Markov chain; this matrix is then transformed into another adjacency matrix representing the right-resolving presentation, and the maximal eigenvalue of this second matrix is found, providing a rigorous upper bound for $h^*(T, \mathcal{A})$. It should be emphasised that the procedure is completely general and automatic, in the sense that any partition \mathcal{B} that is a refinement of \mathcal{A} may be used, and that the construction of the topological Markov chain and resulting right-resolving presentation may be easily automated on a computer.

2.2 Constructing the topological Markov chain

We produce a refinement of \mathcal{A} , denoted $\mathcal{B} = \{B_1, \dots, B_n\}$, so that each $A \in \mathcal{A}$ is a union of elements of \mathcal{B} ; typically \mathcal{B} will be very much finer than \mathcal{A} . Define a transition matrix on \mathcal{B} via

$$\mathbf{B}_{ij} = \begin{cases} 1, & \text{if } B_i \cap T^{-1}B_j \neq \emptyset, \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

The matrix \mathbf{B} defines a topological Markov chain. Orbits of the topological Markov chain generate words from the partition \mathcal{A} in the following way. Call $[b_0, \dots, b_{N-1}]$ a \mathcal{B} -word if $\mathbf{B}_{b_i, b_{i+1}} = 1$ for $i = 0, \dots, N-2$. The \mathcal{B} -word $[b_0, \dots, b_{N-1}]$ generates a unique \mathcal{A} -word given by $[a_0, \dots, a_{N-1}]$ where $B_{b_i} \subset A_{a_i}$, $i = 1, \dots, N-1$. Define the set of all \mathcal{A} -words of length N by

$$\mathcal{W}_N(\mathcal{B}, \mathcal{A}) = \{[a_0, \dots, a_{N-1}] : \text{there is a } \mathcal{B}\text{-word } [b_0, \dots, b_{N-1}] \text{ of length } N \text{ such that } B_{b_i} \subset A_{a_i}, i = 1, \dots, N-1.\} \quad (4)$$

2.2.1 Relationship to T

Clearly orbits of the topological Markov chain are pseudo-orbits of the map T , and therefore $\mathcal{W}_N(T, \mathcal{A}) \subset \mathcal{W}_N(\mathcal{B}, \mathcal{A})$; that is, our topological Markov chain generates all words that T generates, plus some extra words. The number of extra words generated depends on how fine the partition \mathcal{B} is relative to \mathcal{A} . Our plan is therefore to fix \mathcal{A} and successively refine \mathcal{B} . At all times, we will have

$$h(\mathcal{B}, \mathcal{A}) := \lim_{N \rightarrow \infty} \frac{\log |\mathcal{W}_N(\mathcal{B}, \mathcal{A})|}{N} \geq \lim_{N \rightarrow \infty} \frac{\log |\mathcal{W}_N(T, \mathcal{A})|}{N} = h^*(T, \mathcal{A}). \quad (5)$$

Furthermore, one has

Theorem 4 *Let $\mathcal{A} = \{A_1, \dots, A_q\}$ partition M , where each A_i is compact. Then*

$$\lim_{\text{diam } \mathcal{B} \rightarrow 0} h(\mathcal{B}, \mathcal{A}) = h^*(T, \mathcal{A}). \quad (6)$$

PROOF. See Appendix B.2

2.3 The associated sofic shift

The reason for constructing such a topological Markov chain is that $h(\mathcal{B}, \mathcal{A})$ is able to be computed exactly. To do this, we construct a directed labelled graph, with nodes $1, \dots, n$ corresponding to elements of \mathcal{B} , and arcs and labels which we now define. Each element of \mathcal{A} will have a distinct label selected from the label set $L = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots\}$. The nodes i and j are connected with an arc iff $\mathbf{B}_{ij} = 1$; this arc is labelled with the label corresponding to the set $A \in \mathcal{A}$ satisfying $B_i \subset A$.

Example 5 (Simple example: Topological Markov chains and sofic shifts)

Consider the (piecewise) continuous map $T : [0, 1] \rightarrow [0, 1]$ given by

$$Tx = \begin{cases} 2x, & x < 1/2 \\ 3(x - 1/2)/2, & x \geq 1/2. \end{cases}$$

We select

$$\mathcal{A} = \{A_1, A_2\} = \{[0, 1/2), [1/2, 1)\},$$

and

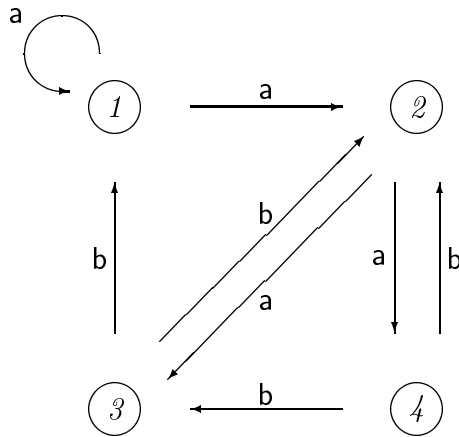
$$\mathcal{B} = \{B_1, B_2, B_3, B_4\} = \{[0, 1/4), [1/4, 1/2), [1/2, 3/4), [3/4, 1)\}.$$

By Theorem 3 (i), we see that $h(T) \leq \log 2$, so by choosing \mathcal{A} to contain two sets, we have a chance of capturing all the entropy.

It is easily verified that the matrix \mathbf{B} is given by

$$\mathbf{B} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}.$$

Let the partition sets B_1, B_2, B_3, B_4 correspond to vertices 1, 2, 3, 4 respectively. By assigning A_1 the label \mathbf{a} and A_2 the label \mathbf{b} , we arrive at the following directed labelled graph (which we denote by $G(\mathcal{B}, \mathcal{A})$).



Elements of $\mathcal{W}_N(\mathcal{B}, \mathcal{A})$ are generated by walks of length N on this graph, concatenating labels along the walk. Words generated by walks on this graph are not the result of a subshift of finite type on the symbols $\{a, b\}$ because while the words $[aa]$, $[ab]$, $[ba]$ and $[bb]$ are allowed, the word $[bbb]$ is not (if it were of finite type, one could form all possible words by concatenation of allowable 2-words).

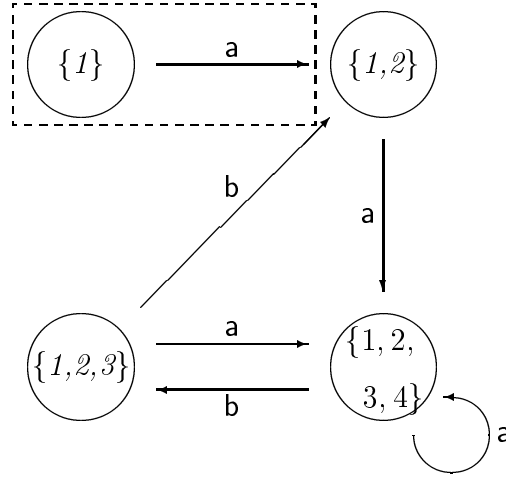
2.4 A (reduced) right-resolving presentation of the sofic shift

Let $\mathcal{W}(G(\mathcal{B}, \mathcal{A}))$ denote the set of all bi-infinite words that are generated by walks on the labelled directed graph we have just constructed. The problem with the graph $G(\mathcal{B}, \mathcal{A})$ in Example 5 is that there can be more than one arc with the same label emanating from the same node. This makes entropy computation difficult, and so we seek a new graph (or presentation) $G'(\mathcal{B}, \mathcal{A})$ for which (i) $\mathcal{W}(G'(\mathcal{B}, \mathcal{A})) = \mathcal{W}(G(\mathcal{A}, \mathcal{B}))$ and (ii) that from each node of $G'(\mathcal{B}, \mathcal{A})$, all outgoing arcs have different labels. We now describe an algorithm for finding the “essential subgraph” \mathcal{R} of such a graph G' . We will call \mathcal{R} a *reduced right-resolving presentation* of G ; see Appendix B.3 for details and proofs of necessary results. Nodes of the reduced right-resolving presentation will be called *hypernodes* and arcs will be called *hyperarcs*. Hypernodes will be subsets of $\{1, 2, \dots, n\}$.

Example 6 (Simple example: Reduced right-resolving presentations)

The figure below shows the result of Algorithm 2.4 applied to the graph in Example 5 (using the hypernode $\{1\}$ as the initial hypernode).

- (1) Begin with an empty graph \mathcal{R} and add a single hypernode consisting of a single (randomly selected) node of $G(\mathcal{B}, \mathcal{A})$.
- (2) If possible, choose a hypernode in the graph \mathcal{R} with no outgoing hyperarcs and call this hypernode H . Otherwise, go to the pruning step (vi).
- (3) Denote by H' the set of all nodes in $G(\mathcal{B}, \mathcal{A})$ reached by all outgoing arcs in $G(\mathcal{B}, \mathcal{A})$ starting at nodes in the hypernode H and labelled (in $G(\mathcal{B}, \mathcal{A})$) with a . Add the hypernode H' and a hyperarc labelled a starting at H and terminating at H' .
- (4) Repeat (iii) for all labels in the alphabet.
- (5) Return to (ii).
- (6) Remove all hypernodes in \mathcal{R} without incoming hyperarcs (along with their outgoing hyperarcs).
- (7) If some hypernodes were removed, return to (vi), otherwise stop.



We briefly outline the construction of this graph:

- The node 1 in $G(\mathcal{B}, \mathcal{A})$ has outgoing arcs labelled with a terminating at nodes 1 and 2 in $G(\mathcal{B}, \mathcal{A})$, therefore we create a hypernode $\{1, 2\}$ and a hyperarc labelled a starting at hypernode $\{1\}$ and terminating at hypernode $\{1, 2\}$.
- Now consider the hypernode $\{1, 2\}$. Nodes 1 and 2 have outgoing arcs labelled a terminating at nodes 1, 2, 3, and 4, so we create the hypernode $\{1, 2, 3, 4\}$ and add a hyperarc labelled a starting at $\{1, 2\}$ and terminating at $\{1, 2, 3, 4\}$.
- Collectively, nodes 1, 2, 3, and 4 have outgoing arcs labelled both a and b emanating from them. Take label a . The terminal nodes of these arcs are 1, 2, 3, and 4, so we add a hyperarc labelled a looping from the hypernode $\{1, 2, 3, 4\}$ back to itself. Now take label b . The terminal nodes of these arcs are 1, 2, and 3, so we add a new hypernode $\{1, 2, 3\}$ and a hyperarc from $\{1, 2, 3, 4\}$ to $\{1, 2, 3\}$ labelled b .
- Nodes 1, 2, and 3 again collectively have outgoing arcs labelled with both a and b . First take a . The terminal nodes of these arcs are 1, 2, 3, and 4, so we add a hyperarc labelled a from $\{1, 2, 3\}$ to $\{1, 2, 3, 4\}$. Now consider the

arcs labelled \mathbf{b} . The terminal nodes are 1 and 2, so we add a hyperarc from $\{1, 2, 3\}$ to $\{1, 2\}$ labelled \mathbf{b} .

- All hypernodes now have outgoing arcs, so we begin pruning.
- Only hypernode $\{1\}$ has no incoming hyperarc, so we delete this single hypernode and its outgoing arc (this removal is indicated by the dashed box in Example 6).
- All remaining hypernodes have incoming hyperarcs, so we stop.

Note that the hyperarcs leaving each hypernode are all labelled differently in Example 6, in contrast to the arcs and nodes of the graph of Example 5. The reader may like to verify that the final pruned graph of Example 6 is independent of the hypernode used to initiate Algorithm 2.4.

2.5 Entropy of the sofic shift

One forms an adjacency matrix for the graph produced by Algorithm 2.4 as follows. Suppose that there are m hypernodes; then define an $m \times m$ matrix by

$$\mathbf{R}_{ij} = l, \text{ if hypernode } i \text{ has } l \text{ outgoing hyperarcs} \quad (7)$$

terminating at hypernode j .

We are now in a position to compute the entropy of our topological Markov chain with respect to the partition \mathcal{A} .

Proposition 7 *Suppose that T is transitive and let λ denote the maximal eigenvalue of \mathbf{R} . Then $h^*(T, \mathcal{A}) \leq h(\mathcal{B}, \mathcal{A}) = \log \lambda$.*

PROOF. See Appendix B.3.

Example 8 (Simple example: Entropy of sofic shifts) *The adjacency matrix for the reduced right resolving presentation of Example 6 is*

$$\mathbf{R} = \begin{matrix} & \{1, 2\} & \{1, 2, 3\} & \{1, 2, 3, 4\} \\ \begin{matrix} \{1, 2\} \\ \{1, 2, 3\} \\ \{1, 2, 3, 4\} \end{matrix} & \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \end{matrix} \quad (8)$$

The largest eigenvalue of \mathbf{R} is approximately $\lambda = 1.8393$, and therefore $h(\mathcal{B}, \mathcal{A}) = \log 1.8393 \geq h^(T, \mathcal{A})$.*

If we had set $\mathcal{B} = \mathcal{A}$, we would have obtained $h(\mathcal{B}, \mathcal{A}) = \log 2$. As noted at the end of Example 5, by setting $\mathcal{B} = \{B_1, B_2, B_3, B_4\}$, we eliminate the word [bbb] (a word which would be allowed if $\mathcal{B} = \mathcal{A}$). Thus the refinement of \mathcal{B} from two sets to four sets eliminates this word and reduces the entropy to $\log 1.8993$. By refining the partition \mathcal{B} further, more “illegal” words will be eliminated, and the bound $h(\mathcal{B}, \mathcal{A})$ will decrease monotonically. In the limit of the diameters of the partition sets in \mathcal{B} going to zero, $h(\mathcal{B}, \mathcal{A}) \downarrow h^*(T, \mathcal{A})$; this is the content of Theorem 4.

Remark 9 For non-transitive T , $h(T)$ is equal to the maximum of the entropy of T restricted to each transitive region. It is possible that the topological Markov chain governed by \mathbf{B} is still transitive, and in this case, the algorithms may be applied exactly as before to obtain upper bounds for $h^*(T, \mathcal{A})$ and $h(T)$. If the topological Markov chain is not transitive (not irreducible), then the algorithms must be applied separately to each irreducible component of \mathbf{B} .

3 Examples and Results

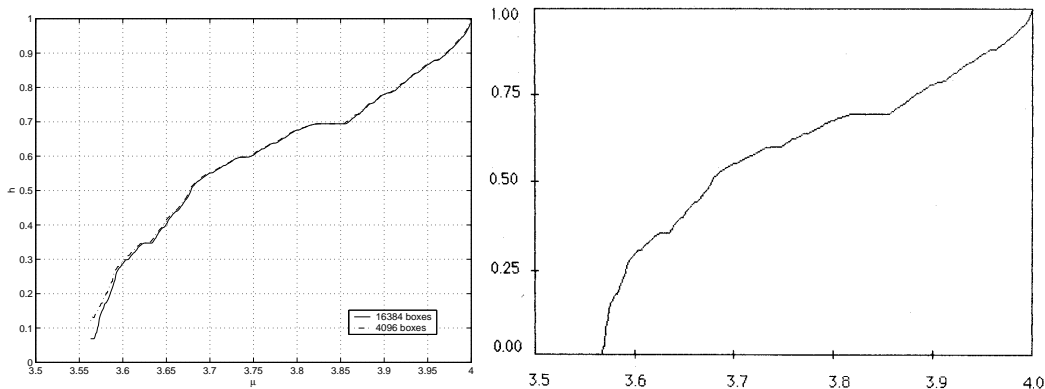
The examples we present range from well-studied systems to those whose entropy is at present not able to be efficiently computed. For some examples, we state that our computations are rigorous. In making this claim, we are working under the assumption that our computer performs exact calculations. In principle, it is possible to allow for roundoff errors by making the boundaries of the sets in \mathcal{B} fuzzy³, however we have not done this here, and do not believe our results would substantially differ by allowing for roundoff.

Each of the calculations presented here takes on the order of seconds to minutes of computer time (including the construction of the Markov chain \mathbf{B}).

3.1 A family of Logistic mappings

To compare our method with previous work, we consider estimating the topological entropy $h^*(T, \mathcal{A})$ of the family of Logistic maps $T_\mu(x) = \mu x(1 - x)$, for $3.5 \leq \mu \leq 4$. For all parameter values μ , we have used $\mathcal{A} = \{[0, 1/2), [1/2, 1]\}$ and \mathcal{B} is either an equipartition of $[0, 1]$ into 2^{12} or 2^{14} subintervals. Shown in Figure 1 are graphs of $h(\mathcal{B}, \mathcal{A})$ versus μ calculated using our method (left frame) and from [3] (right frame).

³ If numerical images land within some tolerance region of a boundary, we accept membership in sets on *both* sides of the boundary. By doing this we allow *more*



(a) Estimate of the entropy (to base 2) of the Logistic mappings $x \mapsto \mu x(1-x)$ versus μ : Method of present paper

(b) Entropy (to base 2) of the Logistic mappings $x \mapsto \mu x(1-x)$ versus μ : Figure reproduced from [3]

Fig. 1. Comparison with previous results.

The agreement is very good, demonstrating that

- (1) this simple choice of \mathcal{A} yields good results even when it is not generating. That is, $h^*(T, \mathcal{A}) \approx h(T)$ here.
- (2) The *bound* $h^*(\mathcal{B}, \mathcal{A})$ for $h^*(T, \mathcal{A})$ is very tight and can be considered as an estimate.

3.2 A hyperbolic linear automorphism of the 2-torus

We now demonstrate the efficacy of our method in a two-dimensional example where the exact value of the entropy is known. Let $T : \mathbb{T}^2 \rightarrow \mathbb{T}^2$ be given by $T(x, y) = (x + y, x) \pmod{1}$. It is known that $h(T) = \log((\sqrt{5} + 1)/2) \approx \log 1.6180$. We set $\mathcal{A} = \{[0, 1) \times [0, 1/2), [0, 1) \times [1/2, 1)\}$. The refined partitions \mathcal{B} will be produced by repeated bisections of the torus; that is, elements of \mathcal{B} will be of the form $[(p-1)/2^k, p/2^k) \times [(q-1)/2^k, q/2^k)$, $p, q = 1, \dots, 2^k$, $k \geq 1$. Our results are summarised in Table 1. The trend here seems to be that the number of hypernodes in \mathcal{R} is a little less than twice that of the cardinality of \mathcal{B} , while the maximal hypernode size is not very large in comparison to the cardinality of \mathcal{B} ; that is, lots of hypernodes, but each hypernode is of relatively small size. Our next example exhibits completely contrasting behaviour. We admit that we do not fully understand how the properties of T or \mathcal{A} influence the properties of the induced hypergraph, such as its size and connectivity; it would be interesting to derive some general principles to help further decrease

pseudo-orbits of T and so preserve our upper bound for $h(T)$.

Table 1
Upper bounds for $h^*(T, \mathcal{A})$ for the linear toral automorphism

k	cardinality of \mathcal{B}	Number of hypernodes in \mathcal{R}	maximal hypernode size	Entropy estimate $h(\mathcal{B}, \mathcal{A})$
2	16	32	5	$\log 1.8393$
3	64	144	10	$\log 1.7494$
4	256	448	21	$\log 1.6916$
5	1024	1792	42	$\log 1.6583$
6	4096	7472	83	$\log 1.6393$

computing time and memory requirements.

3.3 Hénon

We use the standard Hénon map, $T(x, y) = (1 - 1.4x^2 + 0.3y, x)$, and consider the action of T restricted to the box $[-1.5, 1.5] \times [-1.5, 1.5]$. Recent numerical results [12] suggest that $h(T) \approx 0.4651$. To begin with, we use the simple partition $\mathcal{A} = \{[-1.5, 1.5] \times [0, 1.5], [-1.5, 1.5] \times [-1.5, 0]\}$. The elements of our refined partitions \mathcal{B} will be of the form $[-1.5 + 3(p-1)/2^k, -1.5 + 3p/2^k) \times [-1.5 + 3(q-1)/2^k, -1.5 + 3q/2^k)$, $p, q = 1, \dots, 2^k$, $k \geq 1$.

Rather than partitioning *all* of $[-1.5, 1.5] \times [-1.5, 1.5]$ at every level of refinement of \mathcal{B} , we “trim” the topological Markov chain (and the current partition) by finding the *strongly connected component* of the associated graph [15]. It can be shown [16] that the trimmed partition at each level of refinement contains the *chain recurrent set* of T (see [17] for a definition). This trimmed partition produces the maximal irreducible component of the topological Markov chain, and it may be shown using arguments similar to those in the proof of Proposition 21 that this irreducible component generates the same entropy as topological Markov chains produced without any trimming.

Numerical routines [18] to produce topological Markov chains by rigorously calculating all possible intersections in (3) are coded in the GAIO⁴ package. The use of these routines allow us to state that we have rigorous upper bounds for $h^*(T, \mathcal{A})$. Some refined partitions are shown in Figure 2 and numerical results are shown in Table 2. Note that the final value of 0.4628 is *below* the accepted value of the entropy. If the accepted value of $h(T)$ is correct, this proves that \mathcal{A} is not generating for T .

⁴ Available at <http://www.upb.de/math/~agdellnitz/gaio>

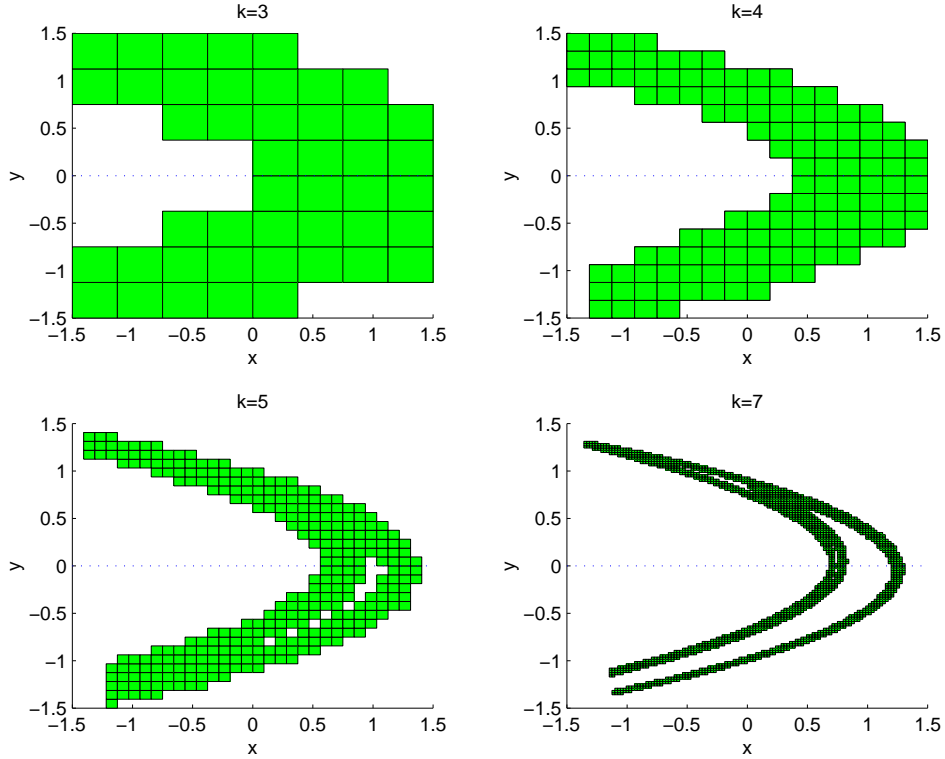


Fig. 2. Example trimmed partitions of $[-1.5, 1.5] \times [-1.5, 1.5]$ for the Hénon map. The horizontal dotted line denotes the boundary of the coarse partition \mathcal{A} .

Table 2

Upper bounds for $h^*(T, \mathcal{A})$ for the Hénon map.

k	cardinality of \mathcal{B}	Number of hypernodes in \mathcal{R}	maximal hypernode size	Entropy estimate $h(\mathcal{B}, \mathcal{A})$
4	122	31	92	0.6244
5	295	29	171	0.5858
6	694	57	338	0.5518
7	1713	163	682	0.5225
8	4197	234	1360	0.5024
9	9766	270	2621	0.4780
10	21639	892	5162	0.4669
11	49296	1086	10286	0.4628

Following the suggestion of [12] that a partition which is near to generating may be constructed by joining up primary and close-to-primary homoclinic tangencies, we recomputed the entropy estimates using an approximation of the partition in [12]. This approximate partition is given by the two-set par-

tition created when $M = [-1.5, 1.5] \times [-1.5, 1.5]$ is bisected by the polygonal arc with vertices $(-1.5, -0.01)$, $(0.703, -0.01)$, $(0.8, 0.025)$, $(1.231, -0.085)$, $(1.272, -0.07)$, and $(1.5, -0.07)$. The results of Table 3 suggest that this new

Table 3

Upper bounds for $h^*(T, \mathcal{A})$ for the Hénon map.

k	cardinality of \mathcal{B}	Number of hypernodes in \mathcal{R}	Entropy estimate $h(\mathcal{B}, \mathcal{A})$
4	122	31	0.6244
5	295	28	0.5858
6	694	81	0.5466
7	1713	180	0.5200
8	4197	244	0.5028
9	9766	409	0.4825
10	21639	980	0.4720
11	49396	1162	0.4687

partition does indeed capture more entropy than the simple-minded horizontal bisection, and our final value now indeed represents an upper bound.

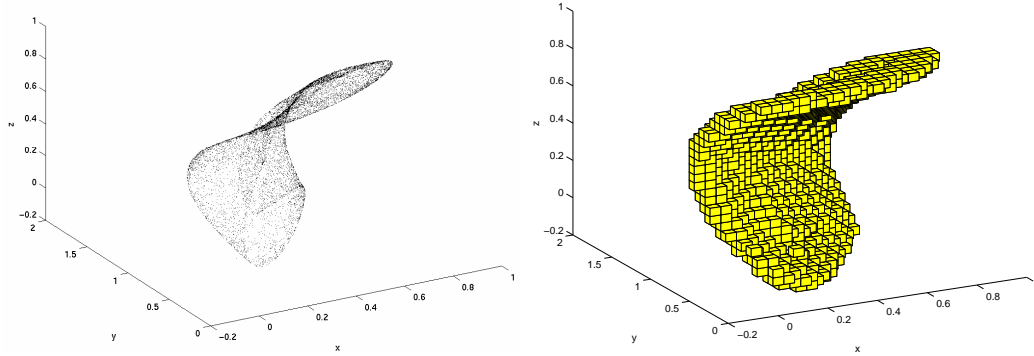
3.4 3D Logistic

Our final example studies a three-dimensional chaotic system (9) which appears numerically to possess an approximately two-dimensional attractor. Define

$$T(x, y, z) = (y - \mu x, \lambda y(1 - x), x - \gamma z), \quad (9)$$

where $\mu = 1.2$, $\lambda = 2.35$, and $\gamma = 0.1$. A plot of a long trajectory of T is shown in Figure 3 (a), and a rigorous outer box covering of the chain recurrent set is displayed in Figure 3 (b).

To select a suitable number of partition sets for the coarse partition \mathcal{A} , we use the bound of Theorem 3 (ii), which suggests that 8 sets are sufficient to capture all of the entropy. We bi-partitioned a cube containing a long trajectory along each coordinate axis at $x = 0.3$, $y = 1.0$, $z = 0.3$, to create an 8-set partition. Results of our calculations are summarised in Table 4.



(a) Plot of a trajectory of T of length 10000

(b) Rigorous outer box covering of the chain recurrent set.

Fig. 3. Invariant attracting sets for the 3D Logistic map

Table 4

Upper bounds for $h^*(T, \mathcal{A})$ for the 3D Logistic map. The * in the final line of the first column indicates that the partition sets of \mathcal{B} have diameter $1/32$ in two coordinate directions and $1/64$ in the third coordinate direction.

k	cardinality of \mathcal{B}	Number of hypernodes in \mathcal{R}	maximal hypernode size	Entropy bound $h(\mathcal{B}, \mathcal{A})$
3	289	49	276	1.197
4	1352	285	1190	1.040
5	6037	1288	4625	0.892
5*	10428	2496	7977	0.837

To obtain a rough “ballpark” comparison for our entropy bounds, we estimated the Lyapunov exponents for the 3D Logistic map and found that $\lambda_1 \approx 0.18$, $\lambda_2 \approx 0.13$, and $\lambda_3 \approx -2.31$. Assuming that the Pesin equality holds for this system, one has that the measure-theoretic entropy of T (using the physical measure μ exhibited by most orbits) is $h_\mu(T) \approx 0.18 + 0.13 = 0.31$. Since $h(T) = \sup\{h_\mu(T) : \mu \text{ is a } T\text{-invariant probability measure}\}$, the value 0.31 represents a rough lower bound for $h(T)$. Our best bound of 0.837 for $h(\mathcal{B}, \mathcal{A})$ is consistent with this result. It is reasonable that 0.837 is significantly higher than 0.31 as our boxes in \mathcal{B} are still relatively large, allowing many spurious orbits to remain.

While the application of our method to systems with higher dimensional unstable manifolds poses no problems, the method of [9] would require significantly more computational effort since it necessitates the computation of exponential growth rates of volumes of higher-dimensional manifolds.

A Implementational details

In this section, we give more details on the numerical realisation of the algorithms presented. The two main computational steps are:

- (1) *Compute the topological Markov-chain on the fine partition \mathcal{B} .* For reasons of efficiency, in practice one does not compute the transition matrix (3) on a full partition of M , but instead uses subdivision techniques in order to compute finer and finer coverings of the chain recurrent set of T in M [19,15,16]. In each step of this technique the matrix (3) is computed on the current collection of sets. For maps these computations can be made both rigorous and efficient [18].
- (2) *Compute the (reduced) right-resolving presentation of the sofic shift.* This involves the construction of the (“hyper”-)graph \mathcal{R} where the nodes of \mathcal{R} consist of subsets of the set of nodes of $G(\mathcal{B}, \mathcal{A})$; see Section 2.4. In step (iii) of Algorithm 2.4 a candidate for a new hypernode together with a corresponding hyperarc is constructed and is eventually added to \mathcal{R} if it is not already present.

As mentioned, there exist efficient algorithms for step (i) – at least in the case where the dynamical system is given by a discrete map. In step (ii) one is faced with the question of how to efficiently store and compute the hypergraph \mathcal{R} . Since the transitions between hypernodes are directly given by the transitions between the nodes in $G(\mathcal{B}, \mathcal{A})$, it is immediate to compute a candidate for a new hypernode from a given one. After the creation of this candidate, one has to decide whether this hypernode is already contained in \mathcal{R} or not. Depending on the data structure used to store \mathcal{R} and on the dynamics of the underlying system this may be computationally expensive. In the following we are going to describe a data structure which enables this decision to be accomplished efficiently. An upper bound for the complexity of the corresponding search within \mathcal{R} is given by $\mathcal{O}(n \log(n))$, where n is the number of sets in \mathcal{B} .

A.1 The data structure

Let $\mathcal{B} = \{B_1, \dots, B_n\}$ be the fine partition. Every hypernode $R \in \mathcal{R}$ is then uniquely determined by a subset of the set $\{1, \dots, n\}$. A common way to store such a set is by a bit-vector of length n , where the i -th bit is set to 1 if and only if $i \in R$. The disadvantage of this approach is that every candidate for a new hypernode has to be compared bitwise with every existing one. We are not going to use this approach but will instead make use of the following idea: If two hypernodes $S, R \in \mathcal{R}$ differ only by one element, it is sufficient to store the common part once and additionally the difference between S and R . In

comparing a candidate with S and R it suffices to compare the common part once and additionally the differences.

Let us be more precise. We are going to construct a tree \mathcal{T} in order to store the elements of \mathcal{R} . To every node $N \in \mathcal{T}$ except the root $N_0 \in \mathcal{T}$ we assign a pair $(i(N), s(N))$, where $i(N) \in \{1, \dots, n\}$ denotes a node within the graph $G(\mathcal{B}, \mathcal{A})$ of the sofic shift and $s(N) \in \{0, 1, \dots, |\mathcal{R}|\}$ is either 0 or denotes⁵ a node in \mathcal{R} . Every node of \mathcal{T} may have at most n children. Now the hypernode $R = \{r_1, \dots, r_\ell\} \in \mathcal{R}$ (where we sort the elements of R such that $r_1 < r_2 < \dots < r_\ell$) is represented as a path $(N_0, N_1, \dots, N_\ell)$ in \mathcal{T} with

$$i(N_j) = r_j, \quad j = 1, \dots, \ell \quad \text{and} \quad s(N_\ell) > 0.$$

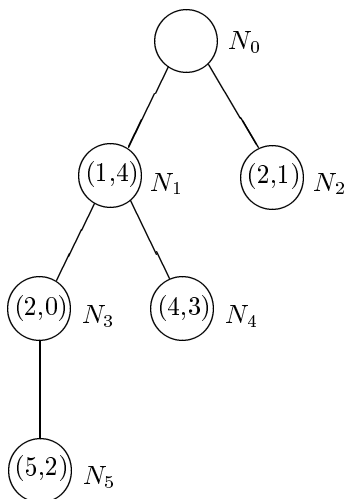


Fig. A.1. Example of a tree for the storage of the reduced right resolving presentation \mathcal{R} .

As an example consider a hypergraph \mathcal{R} with nodes

$$\{\{2\}, \{1, 2, 5\}, \{1, 4\}, \{1\}\}.$$

The corresponding tree is then given as depicted in Figure A.1. The corresponding paths in \mathcal{T} are

$$\begin{aligned} \{2\} &: (N_0, N_2), \\ \{1, 2, 5\} &: (N_0, N_1, N_3, N_5), \\ \{1, 4\} &: (N_0, N_1, N_4), \\ \{1\} &: (N_0, N_1). \end{aligned}$$

⁵ the value of $s(N)$ is necessary only to label nodes in the transition matrix \mathbf{R} ; for the purposes of the tree storage, we need only check if $s(N)$ is nonzero or not.

A.2 Storage of the tree

The maximal number of children of a node depends on the number of sets in the partition \mathcal{B} . Since the actual number of children of a node varies from node to node and may differ significantly (essentially depending on the dynamics of the underlying system), it is not advisable to pre-allocate memory for every possible child of a node. Rather we store the actual children of a node in a set, such that the complexity of the task of looking for a particular child is of order $\mathcal{O}(\log n)$.

A.3 Handling of a candidate for a new hypernode

Let us return to the original task now, namely the problem of having to decide whether a newly created candidate R for a new hypernode of \mathcal{R} is already contained in \mathcal{R} or has to be inserted. Let again $R = \{r_1, \dots, r_\ell\}$ with $r_1 < \dots < r_\ell$. We start with r_1 and look for a child N_1 of the root of \mathcal{T} such that

$$i(N_1) = r_1.$$

If it exists, we repeat the process with r_2 and the children of N_1 . If not, we insert the new hypernode R into \mathcal{T} by constructing the corresponding path in \mathcal{T} as described above.

A.4 Construction of the hypergraph \mathcal{R}

So far we have only discussed how to efficiently store the elements of \mathcal{R} . In fact, while we are building the tree \mathcal{T} we are constructing the transition matrix \mathbf{R} corresponding to \mathcal{R} on the fly. Every time we create a candidate for new hypernode we either update an existing entry of this matrix (in the case when the candidate is already contained in \mathcal{T}) or create a new one (in the case when the candidate is inserted into \mathcal{T}).

B Proofs

B.1 Proof of Proposition 2

Lemma 10 *Let ν be any T -invariant probability measure and $h_\nu(T, \mathcal{A})$ denote the standard metric entropy of T with respect to the partition \mathcal{A} . Then*

$$h_\nu(T, \mathcal{A}) \leq h^*(T, \mathcal{A}).$$

PROOF. Let \mathcal{A} be a partition, and define $H_\nu(\mathcal{A}) = \sum_{A \in \mathcal{A}} \nu(A) \log(\nu(A))$. Further let $|\mathcal{A}|$ denote the cardinality of \mathcal{A} . It is clear that $H_\nu(\bigvee_{i=0}^{N-1} T^{-i} \mathcal{A}) \leq \log |\bigvee_{i=0}^{N-1} T^{-i} \mathcal{A}|$ (Corollary 4.2 [1]). Thus

$$h_\nu(T, \mathcal{A}) \leq h^*(T, \mathcal{A}) := \lim_{N \rightarrow \infty} (1/N) \log \left| \bigvee_{i=0}^{N-1} T^{-i} \mathcal{A} \right|$$

for all invariant ν .

PROOF. [of Proposition 2]

- (1) Using Lemma 10 and the facts that (i) if \mathcal{A} is generating, then $h_\nu(T) = h_\nu(T, \mathcal{A})$ (Theorems 4.17 and 4.18 [1]), and (ii) the variational principle (Theorem 8.6 [1]) that states that $h(T) = \sup_{\nu \text{ is } T\text{-invariant}} h_\nu(T)$, the result follows.
- (2) By Theorem 8.3 [1], $\lim_{\text{diam } \mathcal{A} \rightarrow 0} h_\nu(T, \mathcal{A}) = h_\nu(T)$. Combining this with the variational principle, and Lemma 10, we have the result.

Remark 11 *Before we continue, we point out a few subtle differences between $h^*(T, \mathcal{A})$ and $h(T)$.*

- (1) **$h^*(T, \mathcal{A})$ too low:** *If the partition \mathcal{A} is not complex enough to fully capture the total entropy of T (for example, if \mathcal{A} is not generating), then one may have $h^*(T, \mathcal{A}) < h(T)$. For example, set $T : [0, 1] \ni Tx = 4x \pmod{1}$, and $\mathcal{A} = \{[0, 1/2), [1/2, 1]\}$. It is easy to see that $h^*(T, \mathcal{A}) = \log 2 < \log 4 = h(T)$.*
- (2) **$h^*(T, \mathcal{A})$ too high:** *Sometimes the use of a partition generates an artificial amount of entropy and $h^*(T, \mathcal{A}) > h(T)$. For example, define $T : \{x \in \mathbb{R}^2 : \|x\|_2 \leq 1\} \ni$ (using polar coordinates) by $T(\theta, r) = (2\theta \pmod{1}, r/2)$, and set $\mathcal{A} = \{0 \leq \theta < 1/2\}, \{1/2 \leq \theta < 1\}$. Now $h^*(T, \mathcal{A}) = \log 2$, however, $h(T) = 0$ as T is a contraction with respect to the Euclidean norm.*

While we must be careful that situation (i) does not occur in practice (see the discussion at the beginning of Section 2 for basic precautions), we believe that situation (ii) is rare.

B.2 Convergence of $h(\mathcal{B}, \mathcal{A}) \rightarrow h^*(T, \mathcal{A})$

PROOF. [of Theorem 4]

- (1) For $N \in \mathbb{N}$ and $\varepsilon \geq 0$ we define the following shift invariant subsets of $\{1, \dots, q\}^{\mathbb{Z}}$:

$$W(\infty, \varepsilon) := \{a = (a_i) : \exists (x_i)_{i=0}^{\infty}, x_i \in A_{a_i}, d(Tx_i, x_{i+1}) \leq \varepsilon\}$$

and $W(N, \varepsilon) :=$

$$\{a = (a_i) : \forall i \geq 0, \exists x_i, \dots, x_{i+N-1} \text{ with } x_i \in A_{a_i} \text{ and } d(Tx_i, x_{i+1}) \leq \varepsilon\}$$

Clearly $W(N_2, \varepsilon_1) \subset W(N_1, \varepsilon_2)$ if $N_1 \leq N_2 \leq \infty$ and $0 \leq \varepsilon_1 \leq \varepsilon_2$. If $N < \infty$ and $\varepsilon \geq 0$ is arbitrary, then $W(N, \varepsilon)$ is a subshift of finite type and thus compact.

- (2) We will show $W(\infty, \varepsilon) = \bigcap_{N \in \mathbb{N}} W(N, \varepsilon)$ for every $\varepsilon \geq 0$, which implies in particular that $W(\infty, \varepsilon)$ is compact.

Let $a \in \bigcap_{N \in \mathbb{N}} W(N, \varepsilon)$. Then for every N there are $x_{i,N}$ for $0 \leq i \leq N-1$ such that $d(Tx_{i,N}, x_{i+1,N}) \leq \varepsilon$. By compactness there exists a subsequence of $(x_{0,N})_{N \in \mathbb{N}}$ converging to some $x_0 \in A_{a_0}$. By switching to a further subsequence we can achieve that the corresponding subsequence of $(x_{1,N})$ converges to some $x_1 \in A_{a_1}$. By continuity we have $d(Tx_0, x_1) \leq \varepsilon$. This implies $a \in W(\infty, \varepsilon)$. The reverse inclusion is obvious.

- (3) In this part we show $W(N, 0) = \bigcap_{\varepsilon > 0} W(N, \varepsilon)$, if $N < \infty$.

Consider a fixed word $a = [a_0, \dots, a_{N-1}]$. For $\varepsilon \geq 0$ the set

$$C_\varepsilon := \{(x_0, \dots, x_{N-1}) \in M^N : \forall i \ x_i \in A_{a_i} \text{ and } d(Tx_i, x_{i+1}) \leq \varepsilon\}$$

is compact. Clearly $C_0 \subset C_\varepsilon$ for every $\varepsilon > 0$, i.e. $C_0 \subset \bigcap_{\varepsilon > 0} C_\varepsilon$. On the other hand, if $x = (x_0, \dots, x_{N-1}) \in C_\varepsilon$ for every $\varepsilon > 0$, then by continuity $d(Tx_i, x_{i+1}) = 0$ for every i , i.e. $x \in C_0$. By compactness this implies $C_0 \neq \emptyset$ if and only if $C_\varepsilon \neq \emptyset$ for all $\varepsilon > 0$. Since $W(N, \varepsilon)$ for $N < \infty$ is uniquely determined by all words of length N , the claim follows.

- (4) To conclude the proof we observe

$$W(\infty, 0) = \bigcap_{N \in \mathbb{N}} \bigcap_{\varepsilon > 0} W(N, \varepsilon) = \bigcap_{\varepsilon > 0} \bigcap_{N \in \mathbb{N}} W(N, \varepsilon) = \bigcap_{\varepsilon > 0} W(\infty, \varepsilon).$$

Let $a(N, \varepsilon)$ for $N \in \mathbb{N}$, $\varepsilon \geq 0$ denote the number of distinct symbol sequences of length N in $W(\infty, \varepsilon)$. Then

$$\begin{aligned} \lim_{\varepsilon \rightarrow 0} \lim_{N \rightarrow \infty} \frac{\log a(N, \varepsilon)}{N} &= \lim_{\varepsilon \rightarrow 0} \inf_{N \in \mathbb{N}} \frac{\log a(N, \varepsilon)}{N} \\ &= \inf_{\varepsilon > 0} \inf_{N \in \mathbb{N}} \frac{\log a(N, \varepsilon)}{N} \\ &= \inf_{N \in \mathbb{N}} \inf_{\varepsilon > 0} \frac{\log a(N, \varepsilon)}{N} \\ &= \inf_{N \in \mathbb{N}} \frac{\log a(N, 0)}{N} \\ &= h^*(T, \mathcal{A}) \end{aligned}$$

The assertion of the theorem follows because $a(N, \epsilon) \geq |\mathcal{W}_N(\mathcal{B}, \mathcal{A})|$ if $\text{diam } \mathcal{B} \leq \epsilon$.

Remark 12 *In the case of non-compact A_i one may apply Theorem 4 to $\mathcal{A} = \{\bar{A}_1, \dots, \bar{A}_q\}$.*

B.3 Construction of the reduced right-resolving presentation

The set $\mathcal{W}(G(\mathcal{B}, \mathcal{A}))$ is by definition a sofic shift; that is, it is a set of bi-infinite words formed by the concatenation of labels read on walks around a directed labelled graph. In the sequel we select the necessary results from [20].

Definition 13 *We call a shift \mathcal{W} -irreducible, if given words $u, v \in \mathcal{W}(G(\mathcal{A}, \mathcal{B}))$, one can always find a word $w \in \mathcal{W}(G(\mathcal{A}, \mathcal{B}))$ such that $uwv \in \mathcal{W}(G(\mathcal{A}, \mathcal{B}))$.*

Definition 14 *A graph G will be called irreducible if there is a path from any node i to any other node j . A non-negative matrix B will be called irreducible if given states i, j there is an $N \geq 0$ such that $(B^N)_{ij} > 0$.*

In the case of subshifts of finite type, and in particular, our topological Markov chain governed by \mathbf{B} , \mathcal{W} -irreducibility of the topological Markov chain is equivalent to irreducibility of the $(0,1)$ transition matrix.

Lemma 15 *Suppose that the $(0,1)$ transition matrix \mathbf{B} for the topological Markov chain is irreducible. Then the associated sofic shift is \mathcal{W} -irreducible.*

PROOF. The word u must terminate at some node of the topological Markov chain, call it n_1 ; likewise, n_2 is a starting node of the word v . The word w is simply the word read by taking a path from n_1 to n_2 , a path which exists by irreducibility of the topological Markov chain.

Definition 16 *A labelled graph G' is a right-resolving presentation of a labelled graph G if (i) $\mathcal{W}(G') = \mathcal{W}(G)$ and (ii) for each node of G' , all outgoing arcs are labelled differently.*

Theorem 17 (Thm 3.3.2 and 3.3.11 [20]) (1) *Every sofic shift has a right-resolving presentation.*

(2) *A sofic shift is \mathcal{W} -irreducible iff it has a \mathcal{W} -irreducible right-resolving presentation.*

Theorem 18 (Thm 4.3.3 [20]) *The entropy of a \mathcal{W} -irreducible sofic shift is given by the maximal eigenvalue of the adjacency matrix \mathbf{R} of an irreducible right-resolving presentation.*

A graph of a \mathcal{W} -irreducible right-resolving presentation may not be irreducible. The following two results say that each irreducible component of such a graph carries all of the entropy.

Lemma 19 *If the graph of a \mathcal{W} -irreducible right-resolving presentation $G'(\mathcal{A}, \mathcal{B})$ has more than one irreducible component, then all irreducible components generate the same words.*

PROOF. Suppose that one has two irreducible components C_1 and C_2 , such that one may not move from C_1 to C_2 (reverse movement may or may not be allowed). Let v be a word generated in C_2 that cannot be generated in C_1 , and let u be an arbitrary word generated in C_1 . Given these words $u, v \in \mathcal{W}(G'(\mathcal{A}, \mathcal{B}))$, we should be able to find a word $w \in \mathcal{W}(G'(\mathcal{A}, \mathcal{B}))$ such that $uwv \in \mathcal{W}(G'(\mathcal{A}, \mathcal{B}))$. Clearly the word uwv may not be formed entirely in C_1 , and one can not move into C_2 ; thus \mathcal{W} -irreducibility is contradicted.

Corollary 20 *The entropy generated by any irreducible component of the graph of a \mathcal{W} -irreducible right-resolving presentation equals the entropy generated by the full graph.*

PROOF. Follows directly from Lemma 19.

The main construction embedded in Algorithm 2.4 is described in the proof of Theorem 17 (i). However, the formal construction of a full right-resolving presentation, as described in [20] requires an initial hypernode collection consisting of *all* subsets of the nodes $\{1, 2, \dots, n\}$. That is, one must begin with 2^n hypernodes; something which is computationally infeasible. We now show that one does not need to construct a full right-resolving presentation to find a suitable adjacency matrix; rather a subgraph containing any irreducible component of the full graph is sufficient.

Proposition 21 *Algorithm 2.4 applied to a \mathcal{W} -irreducible sofic shift produces a directed labelled graph such that the logarithm of the maximal eigenvalue l of the adjacency matrix \mathbf{R} for this graph is equal to $h(\mathcal{B}, \mathcal{A})$.*

PROOF. We consider a \mathcal{W} -irreducible right-resolving presentation G_{full} , with existence guaranteed by Theorem 17 (i). Lemma 19 and Corollary 20 tell us that we may focus on any one of the irreducible components of G_{full} . We seek to find a subgraph $G_{\text{Alg1}} \subset G_{\text{full}}$ containing one of the (possibly several) irreducible components of G_{full} by choosing a (random) starting hypernode and traversing all possible outgoing hyperarcs as in Algorithm 2.4 (steps (i) to (v)) and then deleting all non-recurrent hypernodes (steps (vi) and (vii)). Clearly,

our algorithm finds a subgraph G_{Alg1} containing an irreducible component G_{irred} . This irreducible component G_{irred} governs a \mathcal{W} -irreducible shift by arguments as in the proof of Lemma 15. The fact that G_{irred} carries all of the entropy generated by G_{full} follows from Corollary 20. We now need only show that this entropy is equal to $\log \lambda$, where λ is the maximal eigenvalue of the matrix \mathbf{R} . In analogy to the construction of \mathbf{R} , create the matrix $\mathbf{R}_{\text{irred}}$ using only nodes in G_{irred} (the matrix $\mathbf{R}_{\text{irred}}$ is irreducible by definition). Applying Corollary 20 and Theorem 18, we have that $h(\mathcal{B}, \mathcal{A}) = \log \lambda_{\text{irred}}$, where λ_{irred} is the maximal eigenvalue of $\mathbf{R}_{\text{irred}}$. Since our subgraph G_{Alg1} consists of the irreducible component G_{irred} , plus, possibly additional nodes whose connecting arcs eventually lead into G_{irred} , but do not have incoming arcs originating in G_{irred} , one can show that $l = l_{\text{irred}}$, and the result is proven.

PROOF. [of Proposition 7] Transitivity of T implies irreducibility of \mathbf{B} . The result then follows by Lemma 15 and Proposition 21.

References

- [1] P. Walters, An introduction to ergodic theory, Springer-Verlag, New York, 1982.
- [2] P. Collet, J. Crutchfield, J. Eckmann, Computing the topological entropy of maps, *Comm. Math. Phys.* 88 (1983) 257–262.
- [3] L. Block, J. Keesling, S. Li, K. Peterson, An improved algorithm for computing topological entropy, *J. Stat. Phys.* 55 (5/6) (1989) 929–939.
- [4] L. Block, J. Keesling, Computing the topological entropy of maps of the interval with three monotone pieces, *J. Stat. Phys.* 66 (3-4) (1992) 755–774.
- [5] A. Boyarsky, P. Gora, Computing the topological entropy of general one-dimensional maps, *Trans. Amer. Math. Soc.* 323 (1) (1991) 39–49.
- [6] N. Balmforth, E. Spiegel, C. Tresser, Topological entropy of one-dimensional maps: Approximations and bounds, *Phys. Rev. Lett.* 72 (1) (1994) 80–83.
- [7] S. L. Baldwin, E. E. Slaminka, Calculating topological entropy, *J. Stat. Phys.* 89 (5–6) (1997) 1017–1033.
- [8] M. Misiurewicz, W. Szlenk, Entropy of piecewise monotone mappings, *Studia Math.* (1980) 45–63.
- [9] S. Newhouse, T. Pignataro, On the estimation of topological entropy, *J. Stat. Phys.* 72 (1993) 1331–1351.
- [10] Q. Chen, E. Ott, L. P. Hurd, Calculating topological entropies of chaotic dynamical systems, *Phys. Lett. A* 156 (48).

- [11] R. Bowen, Periodic points and measures for axiom A diffeomorphisms, *Trans. Amer. Math. Soc.* 154 (1971) 377–397.
- [12] G. D’Alessandro, P. Grassberger, S. Isola, A. Politi, On the topology of the Hénon map, *J. Phys. A.* 23 (22) (1990) 5285–5294.
- [13] R. Bowen, Entropy for group endomorphisms and homogeneous spaces, *Trans. Amer. Math. Soc.* 153 (1971) 401–414.
- [14] S. R. Katok, The estimation from above for the topological entropy of a diffeomorphism, in: Z. Nitecki, C. Robinson (Eds.), *Global Theory of Dynamical Systems. Proceedings, Northwestern 1979*, Vol. 819 of *Lecture Notes in Mathematics*, Springer-Verlag, Berlin, 1980, pp. 258–264.
- [15] M. Dellnitz, O. Junge, M. Rumpf, R. Strzodka, The computation of an unstable invariant set inside a cylinder containing a knotted flow, in: B. Fiedler, K. Gröger, J. Sprekels (Eds.), *Equadiff 99*, World Scientific, 2000.
- [16] G. Osipenko, Construction of attractors and filtrations, in: K. Mischaikow, M. Mrozek, P. Zgliczynski (Eds.), *Conley Index Theory*, Banach Center Publications 47, 1999, pp. 173–191.
- [17] C. Robinson, *Dynamical Systems: Stability, Symbolic Dynamics, and Chaos*, CRC, Boca Raton, 1995.
- [18] O. Junge, Rigorous discretization of subdivision techniques, in: B. Fiedler, K. Gröger, J. Sprekels (Eds.), *Equadiff 99*, World Scientific, 2000.
- [19] M. Dellnitz, A. Hohmann, A subdivision algorithm for the computation of unstable manifolds and global attractors, *Num. Math.* 75 (1997) 293–317.
- [20] D. Lind, B. Marcus, *An Introduction to Symbolic Dynamics and Coding*, Cambridge University Press, Cambridge, 1995.