Learning the Grammar of a Chaotic Dynamical System in a Format Suitable for Controlling Symbol Dynamics and Communications

Erik M. Bollt
Mathematics Department
572 Holloway Rd
U.S. Naval Academy
Annapolis, MD 21402-5002
bollt@euler.math.usma.edu

Milos Dolnik
Department of Chemistry and
Center for Complex Systems
Brandeis University
Waltham, MA 02254-9110
dolnik@binah.cc.brandeis.edu

Abstract

The unique possibility of encoding information into symbol dynamics corresponding to a chaotic transmitter has been demonstrated [1, 2]. To take advantage of sensitive dependence by using only small deliberate parameter perturbations requires that we respect the grammar of the symbol dynamics which represents the physical dynamical system. We present a new algorithm to learn the *observed* grammar of a symbol dynamics in terms of observed transitions between bins defined by the symbol generating partition. Controlling symbol dynamics using a map based description of the physical dynamical system requires learning the semi-conjugacy, or coding function, between the dynamics of the map on the attractor and the grammar of the corresponding symbol dynamics. In general, it is a difficult problem to learn the grammar of a subshift which is not of finite type. In this paper, we describe a pragmatic, "grid-based," transition matrix scheme to learn a finite approximation of the *observed* grammar [2, 4], in terms of all observed transitions through the chosen partition. The finite approximation of the full grammar is sufficient for course-grained control given finite control tolerances. The method poses few limitations on the dimension and the grammar of the dynamical system.

1 Introduction

Recent work has demonstrated the utilization of symbolic dynamics representation of controlled chaotic orbits for communications with chaotic signal generators [1, 2]. The evolution of trajectories of a chaotic dynamical system is equivalent to symbolic dynamics in an appropriate symbol space. The recent advent of controlling chaos using the OGY technique [3] and a variety of targeting algorithms, with physical applications, has demonstrated that chaos can be mastered, and inherent instabilities can be used as an advantage in allowing *small deliberate perturbations to cause large signal variations.* The sensitive dependence characteristic of chaos is actually advantageous to building a highly agile control system in which a small deliberate perturbation can cause a large response.

To take advantage of sensitive dependence on initial conditions, variations of control parameters must be small, within some threshold. Therefore, we are restricted to the existing dynamics presented by the map, in contrast to creating new dynamics by large or rude control perturbations. Thus, we must respect the grammar of the corresponding symbol dynamics.

2 Controlling Symbol Dynamics

The link between the symbol dynamics of the Bernoulli-shift map restricted to a subshift of symbol space and a chaotic dynamical system, in its physical coordinates, is well established [5] as a descriptive tool. The link can be thought of as a change of coordinates in which properties of the dynamics are preserved. Corresponding to each physical trajectory, there exists an infinite itinerary symbol sequence. The existence of the symbol dynamical description implies that controlling a trajectory of the physical dynamical system is equivalent to controlling a symbol sequence. Recently, S. Hayes, *et al* [1] demonstrated numerically and experimentally, that the connection between information theory and chaotic systems can be used to encode a message into a chaotic electronic circuit, described by a one-dimensional map, by controlling the symbol dynamics through *small* perturbations.

Coupling control of chaos through small perturbations with learning the grammar of the corresponding symbol dynamics means that the control perturbations are

actually a coding scheme on the original dynamics. An example application is a high-power signal generator, which operates intentionally in the chaotic regime, so that a small-scale piggy-back controller, on the microchip scale, has the ability to accurately manipulate high-powered message bearing signals [1].

Applications of sensitively controlled symbol dynamics can be found in electronic communications, as well as in chemistry and in biology. In Ref. [2], we argue that control of symbol dynamics with small energy inputs is a possible mechanism by which a biological system might hold and control information flow as the orbits of their chaotic dynamics; we numerically model encryption of information into chaotic oscillations of the Belousov-Zhabotinsky (BZ) reaction.

To control trajectories using only small perturbations, it is necessary to respect the grammatical restrictions of the corresponding symbol dynamics. Therefore, to communicate, it is first necessary to learn the grammar. The following technique is sufficient to effect course-grained and sensitively controlled communications using a wide class of chaotic dynamical systems. Our algorithm automatically learns the grammatical restrictions of a subshift, and the algorithm is applicable to a wide class of chaotic dynamical systems.

3 Learning the Grammar of the Observed Subshift

Given a chaotic map of the form $x_{n+1} = F_\lambda(x_n)$, evolution of trajectories in the phase space can be mapped to a Bernoulli-shift map over the appropriate subshift of the full symbol space, [6, 7]. The exact form of this subshift depends on the specific choice of the partition, which resolves trajectories into an itinerary of symbols. There exists a correspondence between the dynamics in the phase space of the map, $F : \Lambda \to \Lambda$ and the Bernoulli-shift map on symbol space, $s : \Sigma \to \Sigma$ in terms of labeling the orbit of an initial condition $y_0 \in \Lambda$ through a disjoint splitting of the phase space,

$$\sigma_i = [r(y)]_i = \begin{array}{l} 0 \text{ if } y_i \in S_0 \\ 1 \text{ if } y_i \in S_1 \end{array} , \qquad (1)$$

where $\{S_0, S_1\}$ defines a Markov partition of the phase space. We define $r : \Lambda \to \Sigma$ as the itinerary function, represented as a (bi-)infinite symbol sequence $\sigma \in \Sigma$, for each $y_0 \in \Lambda$. The definition of the Markov partition requires that $S_0 \cup S_1 = \Lambda$, $S_0 \cap S_1 = \emptyset$, and that each S_i is connected. The number of partitioning sets, and hence labels in the symbol set, can be easily generalized.

The subset of the symbol sequence space $\Sigma' \subset \Sigma$ consisting of all those symbol sequences which are itineraries of initial conditions on the attractor, as defined by Eq. 1, defines a subshift; the Bernoulli shift map restricted to Σ', $s|_{\Sigma'}$ is closed and invariant. In fact, the definition of the itinerary function, r in Eq. 1 is a change of coordinates, or semi-conjugacy, when restricted to the subshift: $r : \Lambda \to \Sigma'$.

$$r \circ F|_\Lambda = s|_{\Sigma'} \circ r. \qquad (2)$$

The difficult issue in general is how to choose the generating partition to faithfully reproduce the dynamics as symbol dynamics. For one dimensional dynamics, the issue is well understood, and for two dimensionl dynamics, there has been much progress, notably using the so called "pruning-fronts" [7], however, in $n \geq 3$ dimensions, the problem is open.

Since Σ' is typically just a subshift of the fullshift Σ, there are forbidden symbol sequences; $\exists\, \sigma \in \Sigma$, such that $\sigma \notin \Sigma'$. By counting all permutations of the two symbols 0 and 1, n-at-a-time, we see that there are 2^n n-cylinders. The fact that Σ' is a subshift means that there exists $n > 0$ such that there are n-cylinders which do not appear in any $\sigma \in \Sigma'$; these are *forbidden n-bit words*. Note that n is not necessarily finite.

Each symbol labels a region of the partitioned phase space, and the appearance of the symbol in a (bi-) infinite symbol sequence represents a trajectory which passes through the partition in the specified order. Which symbols can follow, given a preceding sequence of symbols, is the key issue of grammar. *To transmit the message using only small perturbations, it is necessary to respect the grammar* corresponding to the original dynamical system. This can be a nontrivial task for an arbitrary dynamical system. A subshift of finite type is characterized by a list of finite n-bit forbidden words. In general, a subshift is not of finite type.

Under such circumstances, it may seem hopeless to learn the grammar on the computer. However, we assert the following pragmatic grid-based finite transition matrix algorithm to learn all *observed* transitions which define a finite observed grammar, approximating the full grammar. This finite approximation of the grammar is sufficient for finite precision control. The algorithm was originally developed in the setting of learning the grammar defining chaotic oscillations of the BZ reaction [2]. However, there is no foreseeable restriction to applying the algorithm to a wide class of dynamical systems. All "learning" is done automatically, so changing dynamical systems is routine; to this date, we have applied the same grammar learning computer program to several different dynamical systems.

Given an N point grid of initial conditions on the attractor $\{y_i\}_{i=1}^N \subset \Lambda$, we associate an n-bit itinerary by comparison to the partition according to Eq. 1. This establishes the *observed* coding function $r : \Lambda \to$

Σ associating the observed finite symbol sequence of itineraries, $\sigma \in \Sigma$, for each grid point on the attractor, $y_i \in \Lambda$.

The goal is to catalogue all of the observed n-bit words, and the transitions between these words, which are observed among the itineraries of the grid points. This can be decided by a single pass, yes/no, flagging algorithm for each n-bit word. The n-bit approximation to the grammar can be visualized for the two-symbol set, {"0," "1"} as a 2^n node directed graph (see Figure 1). Each node represents one n-bit word, and can have at

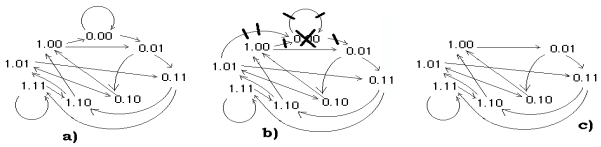

Figure 1: **a:** The 2^3 node directed graph representation generating the fullshift grammar. **b:** Restricting the grammar by forbidding "three zero's in a row." **c:** The resulting subshift generated by a smaller directed graph.

most two arrows leading into it and two arrows leading out of it, corresponding to the choice of shifting in a "0" or a "1". In the case of the fullshift grammar, every node has exactly two entering and exiting arrows; the grammar is generated by a "good DeBroine" graph. See Figure (1a). The n-bit approximation to the grammar is equivalent to a $2^n \times 2^n$ transition matrix A, where an allowed transition from node j to node i is denoted by $A_{i,j} = 1$, and the disallowed transition is denoted by $A_{i,j} = 0$. However, A is necessarily sparse, and requires at most $2n$ computer checks against the grid to decide if the transition occurs, and at most a $2 \times n$ array for storage. The maximal case represents an n-bit approximation of the fullshift grammar where the topological entropy is $h = \ln 2$, calculated as the natural logarithm of the largest eigenvalue of the corresponding transition matrix [5].

For an arbitrary grammar, there will tend to be forbidden n-bit words. Hence, the corresponding node is forbidden, which also erases the arrows both leading into and out of the node. Thus, in terms of the transition matrix representation, erasing the i^{th} node requires that the i^{th} row and column are effectively erased from A. The topological entropy of the subshift Σ_A must be less than the maximal value, $\ln 2$, corresponding to diminished information carrying capacity.

Example Grammar: Figure (1b) depicts the restrictions to the directed graph in which the 3-bit word "0.00," three zero's in a row, is forbidden. That node must be "canceled out." To maintain the closed and invariant property of the subshift, all arrows into and out of the "0.00" node must also be eliminated. Figure (1c) depicts the resulting subshift generated by the smaller directed graph.

It follows that from the node "1.00," there is no choice in what node can follow. To 3-bit accuracy,"0.01" *must* follow by requirement of the grammar. In terms of the shift map

$$s(1.00...) = 0.01.... \tag{3}$$

That is, at this node, only the single transition, shifting in a "1," can occur, and hence there is no information carrying capacity at the "1.00" node, and so the bandwidth is not full.

In contrast, in the above example grammar, the node "0.10" does not have the possibility of a transition to the forbidden "0.00" sequence, in one application of the shift map. Hence, "0.10" is an information bearing node; there are two arrows leading out of "0.10," one to "1.00," and one to "1.01". In terms of the shift map,

$$s(0.10...) = 1.00..., \text{ or } 1.01..., \tag{4}$$

is permitted by the example grammar. Knowing all of the allowed transitions, between all allowed n-bit words, as represented by a directed graph, or equivalently by a transition matrix, is the n-bit approximation of the grammar. For a finite type grammar, there exists a finite n such that all forbidden words are considered.

4 Coding and Decoding Information With Grammatical Resrictions

Given a map $x_{n+1} = F_\lambda(x_n)$ of a chaotic dynamical system, for which we have already learned an n-bit observed grammar, we describe the following scheme to pre-record all best targets to transmit either a "0" bit or a "1" from each grid point which has information carrying capacity.

4.1 Encoding

The directed graph represents the n-bit grammar of the grid points in which a digital message can be encoded according to the following scheme. Note the distinction between the n-bit register (node) and the 7-bit ASCII codes. The message, in ASCII, is coded into the dynamics as the *transitions* from information bearing nodes.

Starting from an arbitrary initial condition x_0, a simple search locates the nearest grid point y_j. As described in section 3, we associate to each grid point a node $r(y_j)$. According to the grammar, the node $r(y_j)$ is either information bearing, or noninformation bearing. An information bearing node is one which allows either of the *two* transitions, to a "0" or a "1". A noninformation bearing node, by restriction of the grammar,

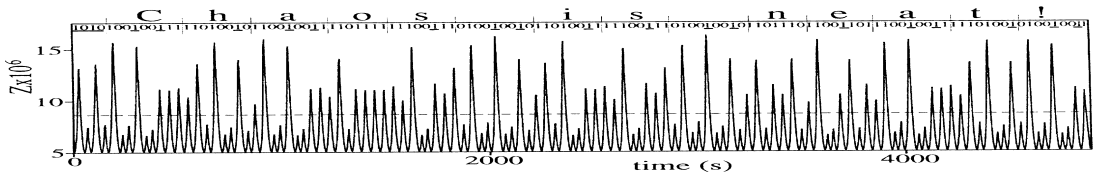

Figure 2: Encoding information in chaotic oscillations of the BZ reaction. The control technique respects the grammar limitations and uses only small parametric perturbations. The surface of section technique is by successive maxima. The decision point is denoted by the dashed line. A maximum below the decision point corresponds to a "0" bit, and maximum above the decision point signals a "1". We transmit the message "Chaos is neat!"

has only one possible outcome. From a noninformation bearing node, it is necessary to transmit the grammatically required buffer-bit. This lack of decision requires that none of the message bits can be attached to this bit.

Each grid point y_j has one fixed n-bit code $r(y_j)$, and therefore there is a closest y_i such that $r(F_\lambda(y_i))$ has a "0" as the least significant digit, if shifting in a "0" from node $r(y_j)$ is permitted by the *observed grammar.* Likewise, there is a closest y_k such that $r(F_\lambda(y_k))$ has a "1" as the least significant digit, if permitted by the observed grammar. Thus, it is possible to pre-process the grid.

Before any control is done experimentally, we precalculate and record the closest point to grid point y_j which shifts in a "0," and the closest point which shifts in a "1". All best targets of all grid points can be stored in a $2 \times N$ array, for an N point grid. A huge advantage of pre-recording is speed and simplicity for "on-the-fly" control of an experimental orbit of x_0; this only requires association of x_t, at time t, to its nearest grid point y_j, and then targeting y_j's pre-recorded target which transmits the desired bit, at minimal energy cost. All pertinent data can be stored in a $3 \times N$ array, where the entries of the array can be arranged as follows:

1. The first column, $y_{j,1}$, records the code/itinerary of the grid point, $r(y_j)$. Forbidden n-bit words will never appear on this list of *observed* itineraries.

2. The second column, $y_{j,2}$, contains the grid number of the closest grid point y_i' to the uncontrolled image $F_\lambda(y_j)$, which shifts in a "0". y_i' represents the target whose code $r(y_i')$ has a "0" as the least significant digit and it is the grid point which shifts in the "0", while requiring the smallest parameter perturbation. If shifting a "0" into the n-bit window is not allowed by the grammar, we record a -1 in this position, and it is necessary to transmit a non-message bearing buffer bit "1".

3. The third column, $y_{j,3}$, contains the grid number of y_k', the closest grid point to the natural image $F_\lambda(y_j)$ which shifts in a "1", or else, a -1 if required by the grammar.

Whenever both $y_{j,2} > 0$ and $y_{j,3} > 0$, indicating $r(y_j)$ is a message bearing node, the desired bit is transmitted, and on the next iterate, the next bit of the message is considered. If $y_{j,2} < 0$ or $y_{j,3} < 0$, then a non-message bearing "buffer-bit" is sent, and on the next iterate, the same message bit is again attempted for transmission.

Note that actually targeting a pre-recorded target y_l' which *by construction* is close to the uncontrolled image $F_\lambda(y_i)$ can be achieved by small parameter variations $\delta\lambda$. According to the standard topology of symbol spaces Σ, (the norm on symbol space $\|\cdot\|_\Sigma = \sum_{i=-\infty}^{\infty} \frac{\sigma_i}{2^{|i|}}$ weights symbols heavily that are nearest the decimal), and semi-conjugacy between the map and its symbol dynamics, small perturbations to δy, the mapping iteration cause small variations in the n-bit symbol representation $\delta\sigma$. I.e., there is an n such that the least significant bit of an n-bit sliding block window, over σ, is controlled. Since the variation δy is expected to be small, the required small variation to the parameter $\delta\lambda$ can generally be calculated by linearization of the map with respect to the control parameter on the surface of section [2, 8, 9].

We mention two improvements to the above. First, instead of a uniformly spaced sampling grid, it is experimentally more feasible to set one initial condition, which we iterate, and observe as a single long, and ergodically wandering orbit. This sequence $y_{i\,i=0}^{N}$ can then be converted to a sequence of n-bit symbol sequences, and retained as a generalized grid, which is distributed according to the invariant measure; more work is spent learning the grammar, where it is more likely to be needed. Second, the grammar can be further pre-processed. Note that all the points with like n-bit codes are close, due to equivalence of the topologies. Therefore, a memory savings can be achieved by simply storing the 2^n centers (means of like coded iterates y_i) for each code, instead of all $3 \times N$ observations.

4.2 Decoding

We read the itinerary directly from the controlled time-series x_t, by comparison of x_t to the partition at each

time t. This information is easily translated into the sequence of n-bit nodes visited by considering a sliding n-bit block through the long itinerary sequence corresponding to x_t.

We must strip off the "buffer bits" which were added during encryption. Given knowledge of permissible transitions through the 2^n node directed graph, the sequence of nodes actually visited can be translated into the encoded message by reading the transitions *only from the information bearing nodes*. In a similar way as during encryption, the decoder reads the choice of shifting in a "0" or a "1" in terms of the actual transitions away from information bearing nodes. However, the transitions away from nonbearing nodes must be ignored, effectively stripping the buffer bits that the transmitter was required to send. The method has the advantage that interpreting the grammar is done automatically on the computer. This saves a large part of the difficulty and limitation found in [1] of interpreting the grammar of the dynamical system; it is never necessary to try to "manually" deduce or learn any fundamental word restrictions (such as forbidding "000") and then to manually insert and then strip buffer bits to respect the grammar, on a case by case basis, to prevent the forbidden words. The complete list of n-bit nodes and their transitions is sufficient for encoding *and* decoding.

5 Targeting

Complete n-bit control over the symbol dynamics of a dynamical system corresponds to complete course-grained control of *all* possible orbits. Targeting is normally a difficult problem, but it is straight-forward in this setting. For example, targeting a period one orbit, can be achieved in n iterates, by "transmitting" the repeating sequence "11111..." if permitted by the grammar. See Figure 3. The n iterate delay is the time

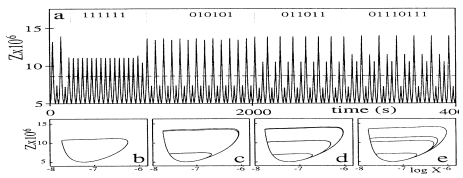

Figure 3: Targeting in the chaotic BZ reaction. a) Stabilization of the periodic orbits by encoding periodic sequences of bits. Phase plane portrait of stabilized b) period-one oscillations - encoded sequence "11111..." c) period-two oscillations - encoded sequence "010101..." d) period-three oscillations - encoded sequence "011011011011..." e) period-four oscillations - encoded sequence "011101110111...".

it takes for the information pertaining to the arbitrary initial condition to clear from the bit register.

6 Conclusions

The numerous potential applications of controlling symbol dynamics include the engineering of new nonlinear electronic communications devices, and modeling phenomenon in biology, chemistry, and cognitive science. Essential questions in chaos theory pertaining to information theoretic properties are greatly simplified in the presence of complete control of symbolic dynamics. For example, targeting becomes a simple matter of "transmitting" the code corresponding to a desired target.

We have presented an algorithm to solve the critical problem of learning the grammar of an arbitrary chaotic dynamical system. The proposed format of pre-recording low-energy information bearing targets on the attractor allows for fast and easy access during experimental control.

References

[1] S. Hayes, C. Grebogi, E. Ott,"Communication with Chaos," Phys. Rev. Lett. **70**, 3031 (1993). S. Hayes, C. Grebogi, E. Ott, A. Mark,"Experimental Control of Chaos for Communication," Phys. Rev. Lett. **73**, 1781 (1994).

[2] E. Bollt, M. Dolnik,"Encoding Information in Chemical Chaos by Controlling Symbolic Dynamics," Phys. Rev. E, June 1997.

[3] E. Ott, C. Grebogi, J.A. Yorke, 1990, "Controlling Chaos," Phys. Rev. Lett. 64, 1196-1199.

[4] E. Bollt, "Communicating with a High Power Digital Transmitter Using Only Small Power Variations through Control of Chaos," *Proc. 3rd Annual US Army Research Lab Technical Symposium*, 1996.

[5] C. Robinson, *Dynamical Systems: Stability, Symbol Dynamics, and Chaos,* (CRC Press, Ann Arbor 1995).

[6] J.P.Crutchfield, N.H. Packard, " Symbolic Dynamics of One-Dimensional Maps: Entropies, Finite Precision, and Noise," Int. J. Th. Phys. Vol. 21, Nos 6/7, 1982.

[7] P. Cvitanovic, G. Gunaratne, I. Procaccia, "Topological and metric properties of Henon-type strange attractors," Phys. Rev. A Vol. 38, No. 3 (1988).

[8] E. Bollt, "Controlling the Chaotic Logistic Map," PRIMUS Vol VII, **1**, 1, 1997.

[9] T. Shinbrot, W. Ditto, C. Grebogi, E. Ott, J.A. Yorke, 1992, "Using Chaos to Direct Trajectories to Target in Systems Describable by a One-Dimensional Map," Phys. Rev. A. 45 4165-8.