Chaos, Complexity, and Inference (36-462)

Lecture 5

Cosma Shalizi

29 January 2008

Symbolic dynamics: Reducing continuous time series to sequences of discrete symbols

Stochastic processes: How to get random sequences from deterministic dynamics

Reading I should have assigned: [1]

Symbolic Dynamics

Start with our favorite dynamical system, with a continuous state S_t and a map Φ

$$S_{t+1} = \Phi(S_t)$$

Partition $\mathcal{B}$: divide the state space up into non-overlapping **cells**, $B_0, B_1, \ldots B_{k-1}$
$b(S_t) =$ label (**symbol**) for the cell S_t is in
$= X_t$ (say)
symbol sequence X

$$\begin{aligned} X_1^\infty &= b(S_1), b(S_2), b(S_3), \ldots \\ &= b(S_1), b(\Phi(S_1)), b(\Phi^{(2)}(S_1)), \ldots \end{aligned}$$

i.e., given initial condition S_1 and partition $\mathcal{B}$, symbol sequence X_1^∞ is fixed

The Shift Map

Seen symbols, what about the dynamics?
Shift map Σ

$$\Sigma(X_1^\infty) = X_2^\infty$$

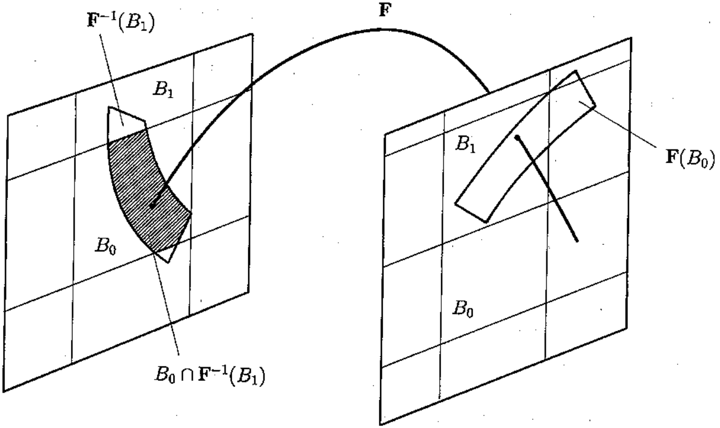
Σ shifts the symbol sequence one place over
$\Sigma^{(k)}$ shifts the symbol sequence k places

Why do this?

1. Model of finite-resolution measurements
2. “Continuous math is hard; let’s go discretize”
   - Discrete-math mathematical tools
   - Probability tools
3. Sometimes involves no real loss

Refinement of partitions

Subdivide cells according to which symbol they *will* give us



From [2, p. 71]

In math, work out $\Phi^{-1}B_i$ for each cell B_i

Now the new partition $\mathcal{B}^2$ is all the sets $B_i \cap \Phi^{-1}B_j$

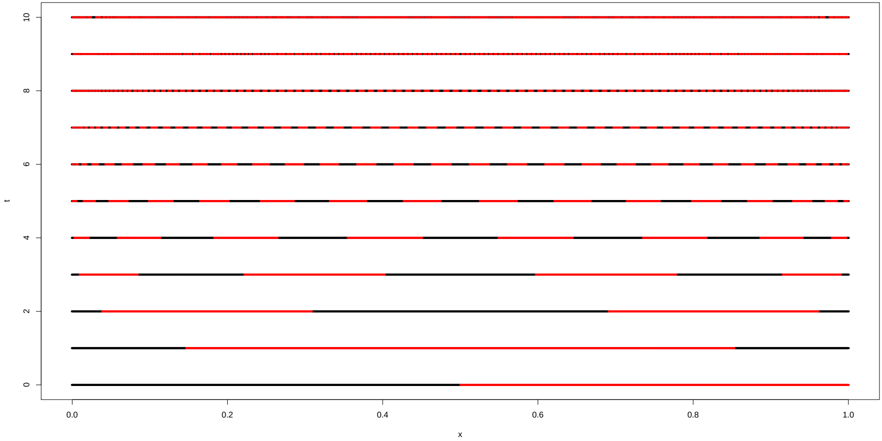
Refinement: knowing the cell in $\mathcal{B}^2$ tells you the cell in $\mathcal{B}$, but not the other way

Generating Partitions

A partition is **generating** if the cells of $\mathcal{B}$, $\mathcal{B}^2$, $\mathcal{B}^3, \ldots$ keep getting smaller forever
or: the *infinite* symbol sequence X_1^∞ corresponds to a *unique* initial condition S_1
then we are back in change-of-coordinates land:

$$\begin{array}{ccc} S_t & \xrightarrow{\Phi} & S_{t+1} \\ b \downarrow & & b \downarrow \\ X_t^\infty & \xrightarrow{\Sigma} & X_{t+1}^\infty \end{array}$$

Example: a generating partition for the logistic map is $B_0 = [0, 0.5)$, $B_1 = [0.5, 1]$.
Write the symbols as L, R so they don't get mixed up with other things



shows $\mathcal{B}, \Phi^{-1}\mathcal{B}, \Phi^{-2}\mathcal{B}, \ldots$

Where did all the details go?

Most of these maps get pretty complicated pretty quickly
e.g. try writing out Φ^{20} for the logistic map
but Σ^{20} is as trivial as Σ
Trick: the complexity has moved out of the dynamical map to the state space — now the space of symbol sequences — and, possibly, probability distributions on the sequence space
This lets us use different mathematical tools

When are there generating partitions?

For one-dimensional maps, make a generating partition by putting boundaries at the “critical” points, i.e. maxima, minima, vertical asymptotes

For higher-dimensional maps, there are fewer general rules; don’t always exist

Estimating generating partitions

"Symbolic false nearest neighbors" [3]: if you have a generating partition, then close symbol sequences should only come from close points in the state-space

1. Reconstruct your state space
2. Start with an initial partition
3. Calculate distances among symbols sequences and distances among state points
4. Find "false symbolic neighbors"
5. Tweak partition boundaries to reduce the number of false neighbors
6. Iterate to convergence

Another approach ("symbolic shadowing"): similar symbol sequences should imply close *trajectories* in state space [4]

Discrete Stochastic Processes

... from fully deterministic continuous dynamics

If S_1 has a distribution, then so do $X_1 = b(S_1)$, $X_2 = b(\Phi(S_1))$, $\ldots$, $X_t = b(\Phi^{(t-1)}(S_1))$, $\ldots$

In general the X_t will be dependent on each other

$\Rightarrow$ symbol sequences are stochastic processes

Studying these processes can tell us about the dynamical system

Symbolic dynamics tells us about how stochastic processes arise

Again with the Logistic Map

$r = 1$

$B_0 = [0, \frac{1}{2})$, $B_1 = [\frac{1}{2}, 1]$

so X_t are binary variables (values L, R)

S_1 in invariant distribution

Claim: X_1^∞ is a sequence of IID, with $P(X_t = L) = 0.5$

Translation: the logistic map gives us perfect coin-tossing

The usual argument for this

1. smoothly change coordinates to go from the logistic map, with state S_t, to the tent map, with state R_t
2. this changes the invariant distribution to be the uniform distribution
3. leaves the generating partition alone
4. write R_t as a binary number
5. $b(R_t) = R$ if and only if the first digit of R_t is "1"
6. $b(R_{t+1}) = R$ if the first digit of R_t was "1" and its second digit was "0", or if the first digit was "0" and the second digit was "1"; etc. for other two-digit combinations
7. $b(R_{t+1})$ is independent of $b(R_t)$
8. in fact $b(R_{t_1}), b(R_{t_2}), b(R_{t_3}), \ldots$ are all independent

But "why think when you can do the experiment?"

EXERCISE 1: Write a program to simulate the symbolic dynamics of the logistic map with $r = 1$. Tabulate the frequencies of sub-sequences of length $2n$. Test whether X_t^{t+n-1} is independent of X_{t-n}^{t-1}.

Independent symbols is an extreme case
More general, dependence across symbols
Two aspects:

- *absolute* restrictions on what sequences can appear (today)
- *relative frequency* dependence (next lecture)

Forbidden Sequences

Take $r = 0.966$; this is moderately chaotic (Lyapunov exponent ≈ 0.42)

You can verify either by calculation or simulation that “LLL” never appears

nor “LLRR”

nor infinitely many others

These are all **forbidden**

Those which do appear are **allowed**

Also say allowed and forbidden **words** (because they’re made from letters)

Topological Entropy Rate

Every allowed word of length n implies a word of length $n-1$
$\Rightarrow$ At least as many longer words as shorter words
$W_n =$ number of allowed words of length n

$$h_0 \equiv \lim_{n\to\infty} \frac{1}{n} \log W_n$$

Measures the exponential growth in the number of allowed patterns as their length grows
$h_0 \geq 0$
Think of e^{h_0} as saying, roughly, how many choices there are for ways of continuing the typical sequence
$h_0 > 0$ is necessary for sensitive dependence on initial conditions
fixed points or periodicity $\Rightarrow h_0 = 0$
With $r = 1$, $W_n = 2^n$ so $h_0 = \log 2$, thus two possible continuations

Careful: $h_0 = 0$ can mean only one possible sequence, or just sub-exponential growth

EXERCISE 2: Write a program to calculate the topological entropy rate for the logistic map at any r
EXERCISE 3: How would you put a standard error on your estimate of h_0?

Languages

A **word** is a finite sequence of symbols

A **(formal) language** is a set of allowed words

A **(formal) grammar** is a collection of rules which give you all and only the allowed words

blame the linguists for mixed metaphor

See [2] for more on how this applies to dynamical systems

See [5] for statistical aspects

See [6, 7] for good introductions to formal languages

Regular expressions

Simplest sorts of formal grammars; used in Unix, Perl, etc.

Basic operations:

- **literals** e.g. L, R, etc., depending on alphabet; also "none of the above", abbreviated ε
- **alternation** "this or that"; make an arbitrary choice from two sets
  e.g. "$L|R$" means "either L or R"
- **concatenation** string together
  $L(L|R)$ means "L, followed by either L or R", means "LL or LR"
- **"star", repetition** Repeat something zero or more times
  "L^*" matches "$\varepsilon, L, LL, LLL, \ldots$"
  "$(L(L|R))^*$" matches
  $\varepsilon, LL, LR, LLLL, LLLR, LRLL, LRLR, \ldots$

$(LR)^*$ a period-two sequence
forbidden words include: RR, LL

$(L(L|R))^*$ "odd-place symbols must be 'L', even-place can be L or R"
or: "every other symbol must be 'L' "
forbidden words include: RR
periodicity here is hidden

$(LRR(RR)^*)^* = (L(RR)^+)^*$ blocks of Rs, of even length, separated by isolated Ls
forbidden words include: LL, $LRRRL$

$(L^*(RR)^*)^*$ even-length blocks of Rs, separated by blocks of Ls of arbitrary length
forbidden words include: $LRRRL$

Not describable by any regular expression: every "(" must be followed eventually by a matching ")"

Expressive Machinery

Basic theorem [8]: every regular expression can be implemented by a machine ("automaton") with a *finite* memory; finite automata can only implement regular expressions

"implement": check if words match the expression, or generate words which match; equivalent

think about generating, it feels more like dynamics

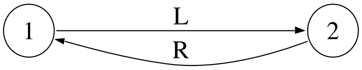
Machines as Directed Graphs

write machines as circle-and-arrow diagrams, directed graphs

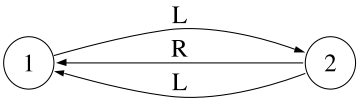
- circles: "states"; fixes possible future sequences
  for symbolic dynamics, each state in the diagram is a *set* of states in the original, continuous state-space
- arrows: go from circle to circle, labeled with symbols from the alphabet

paths generate words: write down the labels hit following that path
concatenation $\approx$ following arrows
alternation $\approx$ more than one out-going arrow from a circle
star $\approx$ loops
should distinguish allowed "start" and "stop" states

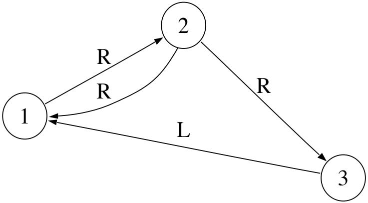
Some Machines and Expressions



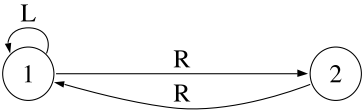
$(LR)^*$



$(L(L|R))^*$



$(L(RR)^+)^*$



$(L^*(RR)^*)^*$

Sofic Systems

Sofic: only finitely many circles (also called **finitary**)

Finite type: to determine what symbol is possible next, need only look back k symbols, for some *fixed* k

Strictly sofic: sofic, but not of finite type

$(LR)^*$ is of finite type

$(L(L|R))^*$, $(L(RR)^*)^*$ and $(L^*(RR)^*)^*$ are strictly sofic

These are the skeletons of stochastic processes

Finite type $\approx$ finite-order Markov chains

Strictly sofic $\approx$ hidden Markov, long-range correlations

next time: some statistics!

[1] C. S. Daw, C. E. A. Finney, and E. R. Tracy. A review of symbolic analysis of experimental data. *Review of Scientific Instruments*, 74:916–930, 2003. URL `http://www-chaos.engr.utk.edu/abs/abs-rsi2002.html`.

[2] Remo Badii and Antonio Politi. *Complexity: Hierarchical Structures and Scaling in Physics*. Cambridge University Press, Cambridge, England, 1997.

[3] Matthew B. Kennel and Michael Buhl. Estimating good discrete partitions from observed data: symbolic false nearest neighbors. *Physical Review Letters*, 91:084102, 2003. URL `http://arxiv.org/abs/nlin.CD/0304054`.

[4] Yoshito Hirata, Kevin Judd, and Devin Kilminster. Estimating a generating partition from observed time series: Symbolic shadowing. *Physical Review E*, 70:016215, 2004.

[5] Eugene Charniak. *Statistical Language Learning*. MIT Press, Cambridge, Massachusetts, 1993.

[6] Marvin Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

[7] Harry R. Lewis and Christos H. Papadimitriou. *Elements of the Theory of Computation*. Prentice-Hall, Upper Saddle River, New Jersey, second edition, 1998.

[8] S. C. Kleene. Representation of events in nerve nets and finite automata. In Claude E. Shannon and John McCarthy, editors, *Automata Studies*, volume 34 of *Annals of Mathematics Studies*, pages 3–41, Princeton, New Jersey, 1956. Princeton University Press.