Fractal grammars which recover from perturbations

Whitney Tabor (whitney.tabor@uconn.edu)
Department of Psychology, University of Connecticut
Storrs, CT 06269-1020 USA

Abstract

Neural symbolic integration may be a natural phenomenon of dynamical systems. Attractors—subsets of a state space to which a dynamical system returns when perturbed—are a broadly relevant dynamical systems phenomenon. The mathematical theory has mainly focused on autonomous dynamical systems (i.e., not driven by an environment) of the form $f : X \to X$ (where $x(t+1) = f(x(t))$ [iterated map] or $\frac{dx}{dt} = f(x)$ [differential equation]), and discovered a rich inventory of attractors, including stable fixed points, limit cycles, and chaos. Here, I focus on the iterated map case and consider certain nonautonomous dynamical systems characterized by a finite set of functions $f_1, f_2, ..., f_k : X \to X$ and a language on alphabet $\Sigma = \{1, \ldots, k\}$ of one-sided infinite strings which applies the functions in particular orders starting from a specified initial state $x(0)$ in X. I extend the definition of attractor by considering cases where the system returns to an invariant proper subset when perturbed in the environment of the language. The news of this paper is that there is a class of nonautonomous dynamical systems that have attractors for mirror recursion languages, a type of language arguably central to natural language syntax.

Keywords: dynamical systems theory, attractors, asymptotic stability, grammar, context free languages, mirror recursion languages; neural-symbolic integration

1 Introduction

This paper approaches neural symbolic integration by interpreting certain dynamical systems, which can be implemented in neural networks, as symbol processors. It uses insights from the classical theory of computation (Chomsky Hierarchy) to explore and categorize the behavior of these models, thus helping to relate them to previous, symbolically oriented work in cognitive science, especially that on natural language syntax. It begins with a brief review of methods for symbol processing using discrete-update recurrent neural networks, making a transition in the process, from a perspective in terms of neural information processing to a perspective in terms of dynamical systems theory. This leads to the question of stability—here, by "stability", I mean the ability of a network processing a complex language to get back on track if something throws it off. The paper proposes a precise definition of stability, suitable to complex language processing, and shows that at least one interesting class of dynamical systems for language processing possesses this property. The paper concludes with some remarks on implications for sentence processing research, dynamical systems research, and the project of neural-symbolic integration.

1.1 Elman Net and Subsequent Work

Elman (1991) found that a discrete update recurrent neural network (the "Simple Recurrent Network") trained on sequences of symbols encoded as indexical bit vectors learned to keep track of English-like center-embedding dependencies, suggesting that the network might be able to model

the phrase-structure foundation that many linguists posit for natural language (Chomsky, 1957; Gazdar, 1981). This work indicates a path to neural-symbolic integration, but only suggestively because the structure of the model's computation was only observed impressionistically. A series of related projects (Bodén & Wiles, 2000; Pollack, 1987; Rodriguez, 2001; Siegelmann & Sontag, 1991; Siegelmann, 1999; Tabor, 2000; Wiles & Elman, 1995) ask how networks of recurrently connected sigmoidal units can precisely process infinite state languages of various kinds. Indeed, the range of possible computations in networks is great, including all of the Chomsky Hierarchy (Siegelmann & Sontag, 1991). These projects all refer to a common principle: the network can use a fractal subset of its state space to implement a stack or tape memory. A fractal is a set that is made up of smaller replicas of itself (Mandelbrot, 1977). A key insight of all these projects is that the spatial recursive structure of a fractal can be used to keep track of the temporal recursive structure of a complex language.

Recurrent neural networks are instances of the type of feedback systems standardly studied in dynamical systems theory (the mathematical theory concerned with systems characterized in terms of how they change). Dynamical systems seem to have a precise relationship to grammars: (a) "symbolic dynamics", a method of treating a dynamical system on a connected space as discrete symbol processor has been very useful in characterizing chaos (e.g., Devaney, 1989), an important dynamical phenomenon; (b) an indexed context free language gives the topology of a key trajectory of the logistic map, a rich and much-studied dynamical system (Crutchfield & Young, 1990); (c) dynamical systems construed as symbol processors in various ways, exhibit a rich range of computational types, including finite languages, finite state languages, context free languages, queue-based languages, Turing Machines, and non-computable languages (Moore, 1998; Siegelmann, 1999; Tabor, 2009). In all the cases involving computable infinite state languages, a fractal subset of the state space supports recursive temporal computation.

Generally, when an iterated map dynamical system corresponds to a particular discrete-update neural network for recursive processing, the state space of the dynamical system corresponds to the recurrently-connected activation space of the network; the control parameters of the dynamical system correspond to the weights of the network; the parameterization of the dynamics associated with a particular symbol corresponds to the activation of a particular unit corresponding to the symbol, which has first-order or second-order connections to the recurrently connected units, thus specifying the update behavior of those units; the branches of the fractal are typically associated with different classes of symbols; the dynamical system may have an associated (finite) partition of the state space which specifies which maps can be applied from which states; correspondingly, the network may have one or more classifier layers outside the recurrent dynamics which map the recurrent state space to next-symbol options. (Bodén & Wiles, 2002; Rodriguez, 2001; Siegelmann, 1999; Tabor, 2000, 2003, 2011, e.g.,)

1.2 An issue: stability

The results just discussed point to the rich computational capability of neural networks and related dynamical systems. However, this computational capability is not very helpful if it is unstable—that is if disturbance of the system through noise or other forces not tied to the computation easily cause the system to stop computing successfully on future inputs. There are at least two senses of stability that one might be concerned with: (i) stability with respect to changes in the state variables—e.g., perturbation of the activation states of the neurons in a neural network, and (ii) stability with respect to changes in the parameter variables—e.g., perturbation of the weights of a neural network. When small parameter perturbations do not change the topology of the dynamics, the system is said to exhibit *structural stability*. Here we focus on state variable stability, noting that structural stability is an additional property of interest which deserves further investigation (see Conclusions). If a system is a stable language processor, then it should be able to recover from forces that temporarily knock it off the track of grammatical processing. Evidence from the sentence processing literature suggests that when people are disturbed in the course of processing a sentence (e.g., by a garden path event, or a confusion due to interference) they exhibit a disturbance which lasts beyond the word that causes the disturbance ("spill-over effects"), but typically only for a few words, with evidence for resolution of the disturbance occurring at the end of the sentence ("sentence wrap-up effects"). This suggests that human minds processing language have "asymptotic stability"—as processing progresses following a disturbance, the system returns to normal processing.

Motivated by these observations, we establish, in the next section, a definition of stability for symbol-driven dynamical automata.

2 Back in Kansas Stability

In classical formal grammar theory, languages are sets of finite-length strings. In the present work, we consider languages to be sets of one-sided infinite length strings. This simplifies the formalism. We can assign each language on the Chomsky Hierarchy a place in this framework by considering, for language L, the set of all one-sided infinite concatenations of strings from L, which we denote L^∞, thinking of this as a model of all the sentences that a speaker hears and/or produces in succession in their lifetime.

Def. An *iterated function system* (Barnsley, [1988]1993; Hutchinson, 1981) is a (finite) set of functions $IFS = \{f_1, \ldots, f_k\}$ that map from a space X to itself. We assume that X is a complete metric space with metric d.

Def. A *one-sided-infinite-string language*, L, is a set of one-sided infinite strings drawn from a finite alphabet, $\Sigma = \{1, \ldots, k\}$. We assume that for every member j of Σ, there is some string of L which contains k.

For $x \in X$, and $S = \sigma_1\sigma_2 \ldots \sigma_N$ a finite string of symbols drawn from Σ, we use the notation $IFS_S(x)$ to denote the function $f_{\sigma_1}(f_{\sigma_2}(\ldots(f_{\sigma_N}(x))\ldots))$.

Def. Consider a point $x(0)$ in X. The *labeling*, $Lab_{x(0)}$, of IFS driven by L is a function from points in X to the power set of Σ, such that $j \in Lab_{x(0)}(x)$ iff there is a finite initial substring, S, of some string in L such that $IFS_S(x(0)) = x$ and the string formed by adding j to the end of S is also an initial substring of some string in L.

Def. $Lab_{x(0)}$ is said to *process a symbol* j at a point x iff $j \in Lab_{x(0)}(x)$ and the system moves from x to $f_j(x)$. A labelling $Lab_{x(0)}$ under IFS is said to *process a string* S iff starting at $x(0)$, every symbol of S is processed in sequence. The *language of* $Lab_{x(0)}$ under IFS is the set of one-sided infinite strings processed by $Lab_{x(0)}$ under IFS. We say that an $IFS - L$ systems is *on* $Lab_{x(0)}$ when it visits a point x if all continuations of the current string under L are processed by $Lab_{x(0)}$.

Def. Consider $x \in X$ and $j \in \Sigma$. The *distance* $d((x, j), Lab_{x(0)})$ from the ordered pair (x, j) to the labeling, $Lab_{x(0)}$, is $inf\{d(x, y)) : y \in X$ and $j \in Lab_{x(0)}(y)\}$.

In other words, assuming a standard (infimum-based) definition of point-set distance, the distance of a point-symbol ordered pair, (x, j), to a labeling is the distance from x to the set of points at which j can come next under the labelling.

Def. A sequence of ordered point-symbol pairs $(x(n), \sigma_n)$ for $n = \{0, 1, 2, \ldots\}$ is said to *converge* to a labeling $Lab_{x(0)}$ if, for every ϵ, there exists N such that $M > N$ implies $d((x(M), \sigma_M), Lab_{x(0)}) < \epsilon$.

Def. Consider an IFS on a complete metric space X with functions $\Sigma = \{1, \ldots, k\}$ and a one-sided infinite string language L on Σ. For initial state, $x(0)$, consider the labeling $Lab_{x(0)}$ induced by L. Consider the sequence of point-symbol pairs $PSP_S = \{(x(n), \sigma_n), n = 0, 1, \ldots\}$ induced by a member, $S = \sigma_1\sigma_2 \ldots$, of L (i.e., $x((n + 1)) = f_{\sigma_n}(x(n))$ for all n). If, when the system is perturbed within a radius δ of $x(n)$ for any $(x(n), \sigma_n) \in PSP_S$ and then driven henceforth by the remaining symbols of S, it converges to $Lab_{x(0)}$, then it is said to exhibit *Back in Kansas Stability from* $x(0)$ *when driven by* S. If there is a δ such that all futures from $x(0)$ converge when perturbed once within δ, then the system is said to exhibit *Back In Kansas stability from* $x(0)$. We say, in such a case, that points of $Lab_{x(0)}$ with nonempty labeling, along with their labels, are an $attractor$ of the IFS-L.

The idea of Back in Kansas Stability is that the system is expected to recover its rhythm, not under the influence of some external signal, but rather simply through exposure to sufficient material from the familiar language. We adopt this approach because, as noted above, studies of sentence processing provide evidence that recovery from disturbance in sentence processing often takes place across multiple words and seems to involve a convergence process.

3 Lack of Back in Kansas Stability in existing fractal models

We next offer some demonstrations that existing fractal computers lack Back in Kansas Stability. As noted above, Moore (1998) describes a one-dimensional dynamical automaton that implements a stack memory for recognizing context free languages. For a stack alphabet of k symbols, Moore's system uses a Cantor-set with $k-1$ gaps between the fractal branches. Pushing and popping are accomplished by

$$\begin{array}{rcl} push_i(x) & = & \alpha x + (1-\alpha)\frac{i}{k} \\ pop_i(x) & = & push_i{}^{-1}(x) \end{array} \tag{1}$$

where $1 \leq i \leq k$ is the symbol being pushed or popped and $0 < \alpha < \frac{1}{2k+1}$ is a constant, and $x(0) = 1$. This system has the property that grammatical processing is restricted to the interval [0, 1], but an erroneous stack operation (e.g., $pop_j(push_i(x))$ where $j \neq i$) results in $|x| > 1$ (Moore's system uses this property to detect ungrammatical transitions). Because push and pop are inverses across the entire state space and grammatical processing (to empty stack) implements a corresponding pop for every push, the displacement created by any error persists no matter how many grammatical sentences follow the error (hence the system does not have Back in Kansas Stability). For example, if one implements S $\rightarrow$ ϵ, S $\rightarrow$ 1 S 2, S $\rightarrow$ 3 S 4 with a two-symbol stack alphabet, choosing $\alpha = \frac{1}{7} < \frac{1}{2 \cdot 2+1}$, then the once-erroneous sequence $1\underline{3}121212\ldots$ results in an endless cycle between -2 and 0.1429—that is, the system never returns to $Lab_{x(0)}$, and, the magnitude of the distance between the state and $Lab_{x(0)}$ endlessly visits a positive constant.

Similarly, Tabor (2000) describes a fractal grammar which processes the non-finite-state context free language, S $\rightarrow$ A B C D, A $\rightarrow$ a (S), B $\rightarrow$ b (S), C $\rightarrow$ c (S), D $\rightarrow$ d (S). The model's state space is R^2 with initial state $(1/2, 1/2)$ and the IFS is given by

$$\begin{array}{rcl} f_a(\mathbf{x}) & = & \frac{\mathbf{x}}{2} + \binom{1/2}{0} \\ f_b(\mathbf{x}) & = & \mathbf{x} - \binom{1/2}{0} \\ f_c(\mathbf{x}) & = & \mathbf{x} + \binom{0}{1/2} \\ f_d(\mathbf{x}) & = & 2\left(\mathbf{x} - \binom{0}{1/2}\right) \end{array} \tag{2}$$

The points with nonempty labels in $Lab_{x(0)}$ form a "Sierpinski Gasket", a kind of two-dimensional Cantor Set. In this system, as in Moore's, all the functions are affine, and the map $f_d \circ f_c \circ f_b \circ f_a(x)$ is identity across the whole state space. Since the language always completes every $abcd$ sequence that is begun, this system, like Moore's, repeatedly revisits the magnitude of any displacement from $Lab_{x(0)}$, independently of the value of $x(0)$.

These systems have a form of stability generally recognized in dynamical systems theory—there is a bound on how far the system travels away from the invariant set of interest—but they lack the asymptotic stability that seems to characterize human behavior. The lack of asymptotic stability is related to the fact that transitions from empty stack to empty stack implement identity across the state space. But identity is only required for grammatical processing, which, in these cases, is restricted to a proper subset of the state space. It will not disturb grammatical processing to modify the map in locations away from this manifold. In the next section, we show, for an important class of languages, a simple way of modifying the off-manifold maps, so that the system gets back onto the manifold when it has been knocked off, provided it receives exposure to enough grammatical material following perturbation.

4 The Simplest Context Free Language, 1^n2^n

We begin with the simplest context free language, 1^n2^n, and then generalize to all mirror recursion languages, $L_{mirror\text{-}\kappa}$, of the form, $S \rightarrow \epsilon, S \rightarrow \sigma_{11}\ S\ \sigma_{12}, S \rightarrow \sigma_{21}\ S\ \sigma_{22}, \ldots, S \rightarrow \sigma_{\kappa 1}\ S\ \sigma_{\kappa 2}$. Mirror recursion seems to capture the gist of center-embedding structures in languages around the

world. Corballis (2007) argues that mirror recursion is the crucial recursive behavior that distinguishes human languages from animal languages. Although it is true that no human language exhibits center embedding easily beyond one center-embedded clause, the degradation, as one tests successively deeper levels of embedding appears to be graceful (Lewis, 1996), and Christiansen & Chater (1999) argue that a Simple Recurrent Network captures this quality of the human language case well. Moreover, Rodriguez (2001) and Tabor et al. (2013) provide evidence that Simple Recurrent Networks trained on center-embedded material are approximating fractal grammars. These observations motivate focusing on the mirror recursion case in the effort to understand how fractal grammars can be stable.

Let $L_{1^n2^n}$ be the one-sided-infinite-string language, $\{1^n2^n\}^\infty$. Let $X = [-2, 2]$ and let $IFS_{1\text{-}d}$ be given by

$$\begin{array}{rcrcrcl} f_1(x) & = & \frac{x}{2} + 1 & & & & \\ f_2(x) & = & 2x + 2 & & & x < & -1 \\ & & 0 & -1 \leq & x & < & 1 \\ & & -2x + 2 & 1 \leq & x & & \end{array} \tag{3}$$

Note that the 2-map is not affine but approximately quadratic. Figure 1 shows the IFS along with Lab_0 and illustrates recovery from a perturbation. In fact, this system always recovers from perturbation.

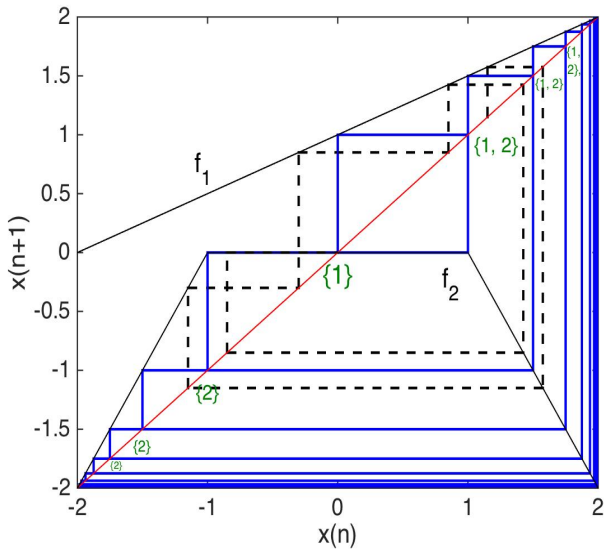

Figure 1: Cobweb diagram of IFS_{1-d} driven by $L_{1^n2^n}$ from $x(0) = 0$. The numbers in curly brackets show some of the nonempty state labels. All points in the complement of $A = \{\ldots, -\frac{7}{4}, -\frac{3}{2}, -1, 0, 1, \frac{3}{2}, \frac{7}{4}, \ldots\}$ have empty label. The dotted line shows a perturbed trajectory which recovers. After processing a single 1 (x(1) = 1), the IFS was perturbed to 1.15. Subsequently it encountered the completion of the perturbed sentence ("... 1 2 2"), followed by a single complete sentence ("1 1 2 2"), by which point it was at 0 and back on the attractor.

Thm 1. The system $IFS_{1\text{-}d}$-$L_{1^n2^n}$ processes $L_{1^n2^n}$ from $x(0) = 0$ and is Back in Kansas stable from that point.

The proof (see Appendix 1) first demonstrates that the language $L_{1^n2^n}$ is processed by $IFS_{1\text{-}d}$ from $x(0) = 0$. Then it shows that, starting from any point in the state space, if the system receives the tail of any string from $\{1^n2^n\}^\infty$, it will be back on Lab_0 within two sentences ("finite-time" convergence). This is a much stronger outcome than is required for Back in Kansas stability. The rapid convergence stems from fact that the horizontal section of the 2-map resets the system when it lands between -1 and 1. We have adopted a broad definition here, allowing also non-finite-time convergence, because simulation experiments which we will not discuss further here suggest that

similar convergence behavior occurs in systems with a single maximum, rather than a plateau, except that the convergence takes infinite time.

5 General Mirror Recursion

For 2κ symbols, let $x(0)$ be the origin in $R^{\kappa+1}$. Define $IFS_{(\kappa+1)\text{-}d}$ for $\kappa = 1, 2, \ldots, i \in \{1, \ldots, \kappa\}$ by

$$-\infty < x_1 < -1 \qquad\qquad -1 \le x_1 < 1 \qquad\qquad 1 \le x_1 < \infty$$

$$f_i(\mathbf{x}) = \begin{pmatrix} x_1/2+1 \\ x_2/2 \\ \ldots \\ x_i/2 \\ x_{i+1}/2-1 \\ x_{i+2}/2 \\ \ldots \\ x_(\kappa+1)/2 \end{pmatrix} \qquad f_i(\mathbf{x}) = \begin{pmatrix} x_1/2+1 \\ 0 \\ \ldots \\ 0 \\ -1 \\ 0 \\ \ldots \\ 0 \end{pmatrix} \qquad f_i(\mathbf{x}) = \begin{pmatrix} x_1/2+1 \\ x_2/2 \\ \ldots \\ x_i/2 \\ x_{i+1}/2-1 \\ x_{i+2}/2 \\ \ldots \\ x_(\kappa+1)/2 \end{pmatrix}$$

$$f_{2i}(\mathbf{x}) = \begin{pmatrix} 2x_1+2 \\ 2x_2 \\ \ldots \\ 2x_i \\ 2x_{i+1}+2 \\ 2x_{i+2} \\ \ldots \\ 2x_(\kappa+1) \end{pmatrix} \qquad f_{2i}(\mathbf{x}) = \mathbf{0} \qquad f_{2i}(\mathbf{x}) = \begin{pmatrix} -2x_1+2 \\ 2x_2 \\ \ldots \\ 2x_i \\ 2x_{i+1}+2 \\ 2x_{i+2} \\ \ldots \\ 2x_(\kappa+1) \end{pmatrix}$$

Here, the -1 in the state change from the middle region for $f_i(x)$ is on dimension $i+1$.

Thm 2. Each system in the class $\{IFS_{(\kappa+1)\text{-}d}\text{-}L_{Mirror\text{-}\kappa}\}$ for $\kappa \in \{1, 2, \ldots\}$ is Back in Kansas stable from $x(0) = \mathcal{O}$, the origin in $R^{\kappa+1}$.

Appendix 2 sketches the proof of Theorem 2.

6 Conclusions

This paper has defined Back in Kansas Stability for nonautonomous dynamical systems, consisting of functions $f_1, \ldots, f_k : X \to X$ on a complete metric space, driven by a language L on $\Sigma = \{1, 2, \ldots, k\}$. The definition was motivated by the fact that people who undergo a disturbance when processing complex natural language structures seem to recover naturally during the course of processing additional words. This idea makes a prediction that many other parsing models—those that tie parsing strictly to the information content of received words—do not make: recovery from a garden path might be helped by words following the disambiguating word even if these words provide no additional structural information; some evidence in support of this idea comes from certain types of length effects in sentence processing where long disambiguating regions are easier than short ones—see Tabor & Hutchins (2004).

Another motivation came from the fact that stability is an important organizing principle of dynamical systems broadly, so when working with neural network dynamical systems, it is desirable to characterize their stability. The mirror recursion finding prompts a more specific observation: in the case of iterated maps, the well-known attractors of autonomous dynamical systems are distinguished by their cardinalities—a fixed point is a single point, a limit cycle contains a finite number of points, a chaotic attractor has uncountably many points. The case of mirror recursion with Back in Kansas Stability fills in this picture by identifying countably infinite attractive sets. It may be informative to ask what conditions support countably infinite attractors and why they are not prominent in the study of autonomous dynamical systems.

6.1 Future Work

We noted above that a dynamical system is considered structurally stable at a point in its parameter space if small changes in the parameter values do not affect the topology of the system. Structural stability seems, in one way, desirable for neural networks, when one is interested in learning: the generally widely successful approach of gradient descent learning is likely to do badly in a context where small parameter changes radically alter the system structure. Results reported in Tabor (2009) for fractal grammars suggest, possibly unfortunately, that these types of systems are not structurally stable in the vicinity of points where they model complex languages. Nevertheless, an interesting alternative perspective may be worth considering: human languages seem to change structurally in a continuous way (see discussion in Tabor et al. (2013)), at least when they change historically, and possibly also when they are learned. It may be useful to invoke, in place of topological equivalence, a notion of structural continuity—two dynamical systems are considered structurally proximal if differences in their topology take a long time to detect. If there is structural continuity, gradient based learning may still be feasible.

What light does this work shed on the challenge of neural symbolic integration? Broadly, it suggests studying neural symbolic integration by studying dynamical systems more generally. Specifically, the results on Back in Kansas Stable point to the fact that not all dynamical computers are stable in this sense; it also raises the question whether all computations can be stably implemented. Stability is very likely a property of human language systems since they have to tolerate a great deal of informational and physical buffeting. It may therefore be useful for the field of neural symbolic integration to identify, both from a dynamics point of view and a language point of view, what systems can be stable.

7 Appendix 1: Proof of Thm. 1

We first show that Lab_0 under IFS_{1-d} processes $L_{1^n2^n}$ (Part A). Then we show that the system is Back in Kansas Stable from $x(0) = 0$ (Part B).

Part A

By definition, every string of L is processed by Lab_0 under IFS_{1-d}. Regarding strings of Lab_0 under IFS_{1-d}, note that 1's only ever occur if $x \geq 0$. If the system starts at 0 and experiences a sequence of the form 1^j2^i where $0 < i < j$, then $x < 0$. Therefore, substrings of the form 21^j2^i1, $0 \leq i < j$ are not processed by Lab_0. If the system starts at 0 and experiences a sequence of the form 1^j2^j where $0 \leq j$, then the system is at $x = 0$, where a 2 never occurs (because, under L, balanced 1s and 2s are always followed by a 1 and 0 is never reached except via balanced 1s and 2s). Therefore substrings of the form 21^j2^i1 where $i > j > 0$ never occur. Therefore the strings processed by Lab_0 under IFS_{1-d} are all and only the strings of $L_{1^n2^n}$.

Part B.

We wish to show that under perturbation up to radius r followed by grammatical continuation to infinity, IFS returns to $Lab_{x(0)}$ in the limit. In fact, in this case, under *any* perturbation, followed by at most one complete string of the form IFS reinhabits $Lab_{x(0)}$.

First, it is useful to define some terminology. We refer to the string in which the perturbation occurs as the *perturbation string*. We refer to the interval $h = [-1, 1]$, where the function b has a fixed value, as the *flat interval*. Note that when h is in the flat interval, $2^k(h) = 0$ for any $k \geq 0$. Note also that if the system is in a state $h_p < 0$ and it processes a string of the form 1^n2^n, then it will arrive at 0 when it processes the last 2 (this follows from the facts (i) that, at every symbol, $1^n2^n(h_p)$ is sandwiched between $1^n2^n(h_0)$ and 0 and (ii) $1^n2^n(h_0) = 0$.) Whenever the state of the system is thus sandwiched, we say that the system is *ahead* with respect to grammatical processing.

Turning now to the proof of B, there are two cases:

(i) The perturbation occurs at some time before the last 1 in 1^j2^j for some $j \geq 0$.

Within this case, it is useful to consider two subcases:

(i-1) The perturbation decreases h. In this case, during processing of the perturbation string, the system is ahead, so it arrives at 0 by the last 2 and is therefore on $Lab_{x(0)}$.

(i-2) The perturbation increases h. In this case, the system is in [-2, -1] at the end of the string. Therefore, the system is ahead during the processing of the subsequent string. Consequently, it reaches $Lab_{x(0)}$ by the end of the post-perturbation string.

(ii) The perturbation occurs after the last 1 and before the last 2 in 1^j2^j for some $j \geq 0$.

Again, we consider two cases:

(ii-1) The perturbation decreases h. In this case, the system is still in [-2, -1] at the end of the string and the future states follow the pattern of (i-2) above.

(ii-2) The perturbation increases h. If, after the increase, $h < -1$, then the remaining 2's bring the system to the flat interval and the case is like (i-1) above. If, after the increase, $-1 \leq h \leq 1$, then the remaining 2's keep the system at 0 and the system is on the attractor at the start of the next string. If, after the increase, $h > 1$, then the system is in [-2, -1] at the end of the string and the future follows (i-2) above.

Thus, in all cases, the system returns to $Lab_{x(0)}$ at least by the end of the post-perturbation string. ■

8 Appendix 2: Sketch of Proof of Thm. 2

The proof of this theorem builds on the proof of Theorem 1. The system behaves on the first dimension as if the string were 1^n2^n with all the push symbols invoking 1 and all the pop symbols invoking 2, following the same dynamics as in the 1-dimensional case above. Whenever the system is in the flat region of the pop maps and in the second half of a sentence, dimension 1 becomes 0 and stays there until the end of the sentence. Then, the start of the new sentence resets all the dimensions to the appropriate values so the effect of perturbation on any dimension is removed. By Thm. 1, this will always happen within two sentences of a perturbation.

Furthermore, when the system, initialized to $x(0) = \mathbf{0}$, is unperturbed, $f_{2i}(\mathbf{x}) \circ f_i(\mathbf{x})$ implements identity for visited points, $\mathbf{x}$ and every stack state corresponds to a distinct point in X because the push maps on X form a just-touching fractal (Barnsley, [1988]1993) so the unperturbed system in $k+1$ dimensions processes $L_{Mirror-k}$. ■

Acknowledgments

Thanks to Harry Dankowicz and Garrett Smith for comments on an earlier version of this work. We gratefully acknowledge support from NSF PAC grant 1059662 and NSF INSPIRE 1246920.

References

Barnsley, M. ([1988]1993). *Fractals everywhere, 2nd ed.* Boston: Academic Press.

Bodén, M., & Wiles, J. (2000). Context-free and context sensitive dynamics in recurrent neural networks. *Connection Science*, *12*(3), 197–210.

Bodén, M., & Wiles, J. (2002). On learning context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, *13*(2), 491–493.

Chomsky, N. (1957). *Syntactic structures*. The Hague: Mouton and Co.

Christiansen, M. H., & Chater, N. (1999). Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science*, *23*, 157–205.

Corballis, M. C. (2007). Recursion, language, and starlings. *Cognitive Science*, *31*, 697–704.

Crutchfield, J. P., & Young, K. (1990). Computation at the onset of chaos. In W. H. Zurek (Ed.), *Complexity, entropy, and the physics of information* (pp. 223–70). Redwood City, California: Addison-Wesley.

Devaney, R. L. (1989). *An introduction to chaotic dynamical systems, 2nd ed.* Redwood City, CA: Addison-Wesley.

Elman, J. L. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, *7*, 195–225.

Gazdar, G. (1981). On syntactic categories. *Philosophical Transactions (Series B) of the Royal Society*, *295*, 267-83.

Hutchinson, J. E. (1981). Fractals and self similarity. *Indiana University Mathematics Journal*, *30*(5), 713-747.

Lewis, R. (1996). Interference in short-term memory: The magical number two (or three) in sentence processing. *Journal of Psycholinguistic Research*, *25*(1).

Mandelbrot, B. (1977). *Fractals, form, chance, and dimension*. San Francisco: Freeman.

Moore, C. (1998). Dynamical recognizers: Real-time language recognition by analog computers. *Theoretical Computer Science*, *201*, 99–136.

Pollack, J. (1987). *On connectionist models of natural language processing.* (Unpublished doctoral dissertation, University of Illinois.)

Rodriguez, P. (2001). Simple recurrent networks learn context-free and context-sensitive languages by counting. *Neural Computation*, *13*(9), 2093-2118.

Siegelmann, H. T. (1999). *Neural networks and analog computation: Beyond the turing limit*. Boston: Birkhäuser.

Siegelmann, H. T., & Sontag, E. D. (1991). Turing computability with neural nets. *Applied Mathematics Letters*, *4*(6), 77-80.

Tabor, W. (2000). Fractal encoding of context-free grammars in connectionist networks. *Expert Systems: The International Journal of Knowledge Engineering and Neural Networks*, *17*(1), 41-56.

Tabor, W. (2002). The value of symbolic computation. *Ecological Psychology*, *14*(1/2), 21–52.

Tabor, W. (2003). Learning exponential state growth languages by hill climbing. *IEEE Transactions on Neural Networks*, *14*(2), 444-446.

Tabor, W. (2009). A dynamical systems perspective on the relationship between symbolic and non-symbolic computation. *Cognitive Neurodynamics*, *3*(4), 415-427.

Tabor, W. (2011). Recursion and recursion-like structure in ensembles of neural elements. In H. Sayama, A. Minai, D. Braha, & Y. Bar-Yam (Eds.), *Unifying themes in complex systems. proceedings of the viii international conference on complex systems* (p. 1494-1508). Cambridge, MA: New England Complex Systems Institute. (http//necsi.edu/events/iccs2011/proceedings.html)

Tabor, W., Cho, P. W., & Szkudlarek, E. (2013). Fractal analysis illuminates the form of connectionist structural gradualness. *Topics in Cognitive Science*, *5*, 634–667.

Tabor, W., & Hutchins, S. (2004). Evidence for self-organized sentence processing: Digging in effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *30*(2), 431-450.

Wiles, J., & Elman, J. (1995). Landscapes in recurrent networks. In J. D. Moore & J. F. Lehman (Eds.), *Proceedings of the 17th annual cognitive science conference.* Lawrence Erlbaum Associates.