Associative symbolic storage in 1D piecewise maps

Michael C. Hughes
F. W. Olin College of Engineering
(Dated: May 20, 2010)

This investigation highlights an application of one-dimensional dynamic systems theory to information processing. We describe and analyze a method invented by Andreyev et al. that allows symbolic strings (e.g. phone numbers like `867-5309`) to be encoded in a piecewise 1D map via limit cycles whose stability can be tweaked deliberately via the Linear Stability Theorem. When prompted by a stimuli that matches a previously encoded string, the corresponding cycle stabilizes and the resulting periodic attractor is interpreted as "recognition" of the stimuli. Advantages of this encoding system include the ability to recognize partial subsets of stored strings, storage capacity for multiple strings, random access to stored information, and some robustness to stimuli error.

Several neurological investigations suggest that information processing within the brain may be modeled by chaotic systems. Memory and recognition tasks in particular have been observed to exhibit dynamic properties. For example, when studying the neurological response of rabbits to various odor stimuli, Freeman observed that known smells gave rise to activity of the olfactory potential that was spatially coherent and nearly periodic [1]. Alternatively, the presentation of a novel, unknown smell results in a state of low-dimensional chaos, as if the rabbit's olfactory processing indicated an "I don't know" response (as cited in [2]). This research has spawned much academic interest in the role of chaotic systems for information processing.

Although much of this research remains neurological, some mathematical study has also been undertaken. Complex neural dynamics such as membrane potential have been successfully modeled with a one-dimensional circle map [3]. More excitingly, however, Andreyev et al. have invented a stimulus recognition process (similar to Freeman's experiments) using dynamically-constructed one-dimensional maps [2]. The result is a virtual memory system that can be initialized to remember certain symbolic strings (e.g. character sequences like `A-B-C-D-E` or numeric ones like `1-3-3-7`). Later, when prompted with known stimuli `A-B-C` the system dynamics will be pushed toward a stable periodic orbit that passes through all symbols of the initially programmed string `...-A-B-C-D-E-A-B-C-D-E-...` . Overall, Andreyev et al. demonstrate a "memory system" that can successfully store many strings simultaneously and differentiate known stimuli from novel ones via the resulting non-transient dynamics. Their procedure can successfully recognize fragments of stored strings, allow random access to stored strings, and even correct some stimuli error (e.g. `C-D-E-F` may still collapse to the `A-B-C-D-E` sequence under certain conditions.

This theoretical investigation seeks to elucidate the principles behind Andreyev et al.'s novel memory system. Although the original work covers many concepts repeated here, this work embarks on a more in-depth analysis of the transition between periodic and chaotic states and studies the recognition problem with much larger strings.

We organize this paper into four primary sections. Section (1) describes how, given a set of strings, a one-dimensional piecewise map can be constructed that encodes these strings into information regions on the unit interval such that a stable periodic orbit of the map passes through these regions. Section (2) explains how varying the slope within these regions transitions the orbit from a stable limit cycle to a chaotic regime and provides bifurcation diagrams for the transition. Section (3) outlines a recognition experiment using telephone numbers and discusses the conditions in which partial stimuli can generate recognition behavior. Finally, Section (4) concludes discussion and diagnoses the overall potential of this method for simulating memory.

(1) Construction of 1D Piecewise Maps— The task at hand is to construct a map that can can encode several symbolic strings into separate stable limit cycles. We define the term *stable limit cycle* to mean a discrete finite-period orbit which occurs as a limiting cycle for any trajectory that begins within sufficient vicinity of the cycle. We represent information here as a *symbolic string*, a finite sequence of symbols that each belong to a predefined set of symbols (an *alphabet*). For example, "MONTANA", "aabbaabb" and "01234" are all strings.

The first step of building a memory map must be to define an alphabet A containing the possible symbols of the strings in question. One application may require sequences of digits 0-9, while another might use the letters A,B,C,D,E. Any choice of a symbol set will work, so long as there are finitely many unique symbols.

Next, we choose a mechanism for encoding a string into a cycle within the unit interval. The simplest would be to represent each symbol separately. For example, the word "dab" would be broken into letters which would each be assigned a portion of the interval (0,1). We call the portion of the interval assigned to a specific symbol its *information region.* The map would then define a cycle passing through each region in the prescribed order `d-a-b`. Figure 1 shows a depth $q = 1$ encoding of the word "dab" into a one-dimensional piecewise map. Following the black line traces the stable cycle that links symbols `d-a-b`. This system appears straightforward, but has a disadvantage in the number of strings it can encode simultaneously. Within the current encoding scheme, each

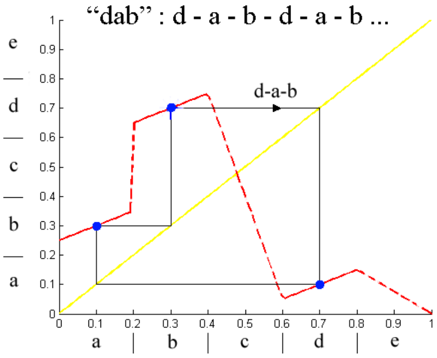


FIG. 1: Storing word “dab” in map with associative storage depth $q = 1$ and alphabet `a,b,c,d,e`. Solid red lines mark information regions (note uniform slope) while dotted red lines are non-informative. A stable cycle is shown in black.

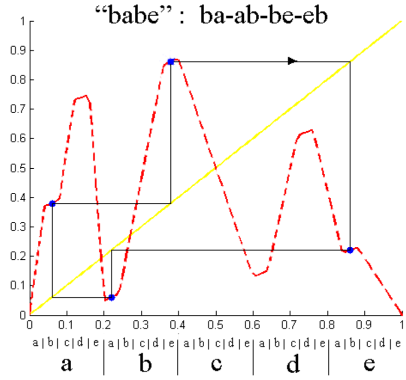


FIG. 2: Storing word “babe” in map with associative storage depth $q = 2$ and alphabet `a,b,c,d,e`. A stable cycle is shown in black.

symbol in the alphabet can only map to one other, which reduces the number of possible strings that can be stored.

An improvement would be to encode pairs of symbols, e.g. a cycle through regions `da-ab-bd`. Any depth of associative encoding q is possible. Figure 2 gives an example of $q = 2$ storage for the word “babe”. The important concept here is the recursive segmentation of the unit interval into a block for each possible combination of symbols up to the storage limit q. Armed with the known size of the alphabet N_A, we first divide the interval into N_A chunks. Next, we divide each chunk into N_A equal pieces, and so on until we have repeated this procedure q times. With this segmentation process, we have a unique information region assigned to each possible string of length q. For example, in Figure 2 each of `ba`, `ab`, `be`, and `eb` has an assigned region.

With this string-to-segment encoding established, we can finally construct a custom map to “remember” given strings. For each string, we decompose the string into size-q chunks and form a closed cycle (e.g. “babe” becomes `ba-ab-be-eb`). Then, we find each chunk’s corresponding segment as well as the next chunk’s segment. We then build the map piece-by-piece, assigning each information region to a corresponding one from the next chunk in the desired loop. After all chunks in all given strings have been assigned, we can fill in non-informative segments via interpolation.

The critical step of constructing a map for an arbitrary string $X_1 X_2 \ldots X_N$ is deciding how to map the information region of X_k onto that of X_{k+1}. Andreyev et al. propose that the midpoint of region X_k is mapped to the midpoint of X_{k+1}, and then the remaining values for segment X_k are assigned linearly given a slope parameter m around that midpoint mapping. This definition allows the stability of cycles to depend on the parameter m. This dependency will be interrogated in the next section.

(2) Determining Stable and Chaotic Regimes — A well-known result of elementary system dynamics is the Linear Stability Theorem, which mandates that a fixed point x will be an attractor if the map function $f(x)$ satisfies $|f'(x)| < 1$. Otherwise, the point x will not attract. This stability criteria also holds for points within an orbit. That is, if all points along an orbit have the same slope m, that orbit will converge to a stable cycle if $|m| < 1$.

We can explore this transition from stability to chaos within our own map construction process. Let us choose as our alphabet the digits `0-9` and encode a simple numerical string `‘‘02468’’` at $q = 2$ storage depth. Now, we allow this piecewise map to run for 1000 iterations for many different values of m. We can then create a bifurcation diagram of the slope m against the observed non-transient values of the map function $f(x)$, as shown in Figure 3. Tracing this graph as m increases, we see that for $m < 1$ a stable limit cycle is firmly established that oscillates between fixed points `02-24-46-68-80`. Immediately above $m = 1$, however, we see some deviation from these points, although the limiting values are still very near the stable regime. Around $m = 1.07$ we see the system drift in and out of disorder, although a bulk of limiting values are observed near the original fixed points. Finally, for $m \geq 1.25$ chaotic behavior dominates. This plot validates the observation that altering the slope m for a particular string can control whether the corresponding encoded orbit becomes a stable limit cycle or not.

Behind this result lies the key to employing a piecewise map as a memory system that can recognize multiple strings using one piecewise map. Essentially, when encoding a string, if the slope m at each information region is set such that $|m| < 1$, then a stable orbit will arise and the system can “remember” that entire cycle if prompted near any one of its points. However, if multiple stable cycles exist (all with slope $|m| < 1$), the system may be liable to converge on the wrong string depending on the initial conditions and particular strings. The solution to this problem of unwanted convergence, as Andreyev et al. propose, is to initially store all information regions with some slope $m_u > 1$. Then, when given a

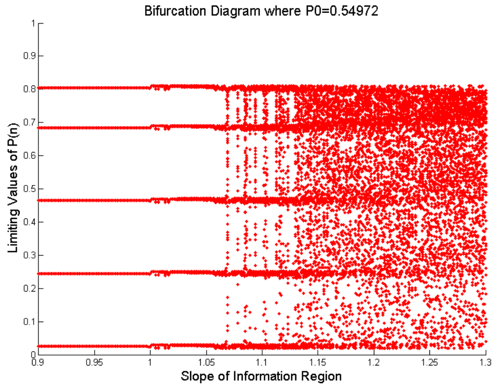


FIG. 3: Bifurcation diagram for map constructed for string `02468` with depth $q = 2$ across a range of information region slope m. Note that for $m < 1$ the system converges to stable limit cycle `02-24-46-68-80`. Just above $m = 1$ the expected cycle is no longer the only observed limiting behavior. As m increases the limiting behavior diverges further and further from the cycle until it is roughly chaotic for $m \geq 1.25$.

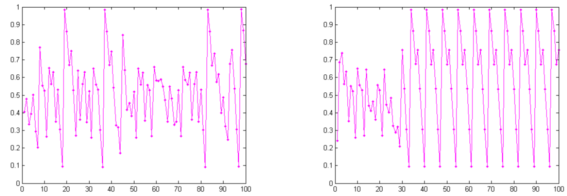
stimuli string, decompose it into its information regions and temporarily *switch* each one's slope to some value $0 < m_s < 1$. The resulting map should converge onto a stable cycle, provided the stimuli matched one of the original strings.

A closer examination indicates that it is not even necessary for every region in a cycle to have slope $0 < m_s < 1$ in order for that cycle to be an attractor. Instead, as explained by Andreyev et al., the stability of an orbit through the points $x_1, x_2, ...x_k$ is governed by that orbit's eigenvalue λ [2]. When $\lambda < 1$, the cycle is an attractor, otherwise the orbit is not stable. For a one-dimensional map $x_{n+1} = f(x_n)$, we have a closed form expression for the eigenvalue of an orbit given the derivative at each point:

$$\lambda = f'(x_1) \cdot f'(x_2) \cdot \ldots \cdot f'(x_k) \tag{1}$$

This fact indicates that so long as sufficiently many points of the cycle have slopes satisfying $|m| < 1$, the product $f'(x_1) \cdot f'(x_2) \cdot \ldots \cdot f'(x_k)$ may have value less than one, meaning the entire cycle will be an attractor. For a recognition system, this means that a full recognition of a stored string can be achieved with only a partially remembered stimuli, e.g. a prompt of `0-1-2` may be sufficient to generate the stored string `0-1-2-3-4` . This result proves to be a primary advantage of associative storage in one-dimensional maps. Determining the exact size of a stimulus string necessary to provoke a stable recognized cycle will be a discussed in the next section.

One importation caution must be stated when determining the stable limit cycles of a given piecewise map. Each map is intentionally built to contain several predefined stable limit cycles. However, these cycles are



(a)Stimuli = `5-3` (b)Stimuli = `5-3-0`

FIG. 4: Attempts to retrieve Jenny's phone number given stimuli `5-3` (a) and `5-3-0` (b). (a) leads to chaotic behavior over time ($\lambda = 1.898$) while (b) was prompted with sufficiently many symbols to converge onto the stored stable cycle `8-6-7-5-3-0-9` stored at depth $q = 2$ with $\lambda = 0.633$. This appearance of a periodic limit cycle signals a "recognition" event for the memory system.

not necessarily the only possible attractors within the piecewise map. Other unintentional *parasitic* attractors can arise within non-informative regions depending on the specific characteristics of a map. Without careful attention, these parasitic cycles can disrupt the recognition process. However, the piecewise nature of the map makes them easy to remove without disturbing the overall map dynamics. Parasitic attractors can be identified by setting all information regions to have non-attractive slope $m > 1$ and observing a bifurcation diagram for fixed points or attractive cycles. Once identified, parasitic attractors can be removed by either setting the local slope greater than one or changing the value of the map at that point.

(3) Application: Telephone Directory Recognition — This section describes how the principles outlined above can be used to construct a basic telephone number recognition system. The motivation here is that telephone numbers are often difficult to remember in full, but sometimes a few digits come to mind. We can build a memory system that can produce an entire stored number given a partially correct prompt.

First, given a set of telephone number strings and a storage depth q, we can build a piecewise map that encodes each string into a limit cycle with unstable slope $m_u > 1$. Then, when prompted by some stimulus string s, we can set all size q substrings of s to a stable slope $m_s < 1$. If sufficiently many stimulus symbols are present, the resulting map will have a stable limit cycle.

Consider a situation in which we have just met a new colleague named Jenny. We store her phone number – `867-5309` – along with the other numbers we happen to know: `656-3552`, `012-3457`, and `979-6114`. We can construct a piecewise map with depth $q = 2$ that accommodates all of these phone numbers. Remember that we cannot store multiple phone numbers that repeat an adjacent pair of digits e.g. `...550 ...` and `...559 ...`, as the mapping for the chunk `55` cannot simultaneously yield `50` and `59`. This restricts our acceptable inputs to non-similar strings, but still allows the storage of many strings simultaneously.

We initially set all information regions with slope $m_u = 1.5$, which places them into the chaotic regime. The resulting piecewise map can then be altered by stimuli such that some information regions gain new slope $m_s = 0.25$. For example, given the stimuli `530`, we alter the information region slopes corresponding to the chunks `53` and `30`. We then observe that the total cycle of 7 symbols contains 2 with slope $m_s = 0.25$ and five with $m_u = 1.5$ for a total eigenvalue $\lambda = (.25)^2 \cdot (1.5)^5 = .474$. This value satisfies $|\lambda| < 1$, so the cycle attracts and the memory system "recognizes" the stimulus `530` as part of Jenny's number.

Two types of iterative responses to stimuli for this system are shown in Figure 4. Case (a) shows that a short stimulus `5 3` does not produce a stable orbit after many iterations (although several repeated paths can be seen). In contrast, case (b) indicates that stimulus `530` arrives at a stable limit cycle representing Jenny's number after 30 iterations, as expected.

What factors affect whether an abbreviated stimulus will generate a "recognition" event? The length of the stimulus k, the total length of the stored cycle n, and the slope values of the switched m_s and unswitched m_u regions yield an expression for a condition of stability based on the the eigenvalue of the orbit.

$$|\lambda| = |m_s|^k \cdot |m_u|^{n-k} < 1 \tag{2}$$

Algebra reveals the criteria for sufficient stimuli size k

$$k > \frac{n}{1 - \frac{\log(m_s)}{\log(m_u)}} \tag{3}$$

Equation 3 above provides a baseline for determining the length of stimulus required to provoke a stable limit cycle. This theoretical limit was investigated in an experiment in which four 10 digit phone numbers were encoded into a single piecewise memory map with depth $q = 2$ and initial information region slope $m_u = 1.5$. Stimuli of increasing length from $k = 1$ to $k = 7$ were then provided. For each length, an altered map was constructed for 1000 values of the switching slope m_s across the range (0,1). Each map was then given a random initial condition and iterated across 1000 generations. The final 100 generations were tested for the existence of the stable cycle, and the critical switching slope s_s which formed boundary between stability $m_s < s_s$ and chaos $m_s > s_s$ was extracted. Figure 5 plots the trend in observed s_s value as a function of the provided stimulus length k. We can see this experimental curve tracks the theoretical limit very well in shape, with an apparently constant horizontal offset. We attribute this offset to the fact that k must be an integer, which requires rounding up from the theoretical curve.

Overall, Figure 5 provides a graphical reference for choosing a sufficient stimulus length k such that the likelihood of observed stable periodic attractor (a "recognition event") remains high. The proper choice of k, m_s, and m_u is in summary a crucial prerequisite for any test of recognition, as insufficient values will lead to chaotic behavior (an "unrecognized" response) no matter the input.

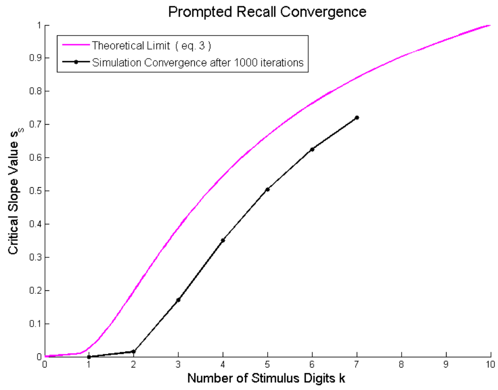


FIG. 5: Plot showing theoretical and observed minimum stimulus length k required to produce stable limit cycles as function of switching slope m_s. The map in question encoded length 10 strings at depth $q = 2$ with initial information region slop $m_u = 1.5$. Theoretical curve comes from equation 3

(4) Conclusion and Further Directions This work has described the process and principles behind Andreyev et al.'s method for encoding strings within piecewise one-dimensional maps for stimulated recognition. This theoretical investigation provides a basis for understanding real-world experiments indicating that recognition events can trigger quasi-stable periodic orbits in neurological tissue. Further research could explore ways to avoid parasitic limit cycles during construction or allow encoding of similar strings (e.g. `867-5309` and `861-5300`) at low information depth.

[1] C. A. Skarda and W. J. Freeman, "How brains make chaos in order to make sense of the word." Behavioral Brain Sci., **10**, pp. 161-165, (1987).

[2] Y. V. Andreyev, A. S. Dmitriev, and S. O. Starkov. "Information Processing in 1-D Systems with Chaos." IEEE Trans. on Circuits and Systems., **44**, 1, (1997).

[3] M. Zeller, M. Bauer, and W. Martienssen. "Neural dynamics modelled by one-dimensional circle maps." Chaos, Solitons & Fractals. **5**, 6, pp. 885-893, (1995).