Available online at www.sciencedirect.com

ScienceDirect

Current Opinion in Neurobiology

# Neural circuits as computational dynamical systems

David Sussillo

Many recent studies of neurons recorded from cortex reveal complex temporal dynamics. How such dynamics embody the computations that ultimately lead to behavior remains a mystery. Approaching this issue requires developing plausible hypotheses couched in terms of neural dynamics. A tool ideally suited to aid in this question is the recurrent neural network (RNN). RNNs straddle the fields of nonlinear dynamical systems and machine learning and have recently seen great advances in both theory and application. I summarize recent theoretical and technological advances and highlight an example of how RNNs helped to explain perplexing high-dimensional neurophysiological data in the prefrontal cortex.

**Addresses**
Department of Electrical Engineering and Neurosciences Program, Stanford University, Stanford, CA 94305, United States

Corresponding author: Sussillo, David (sussillo@stanford.edu)

**Current Opinion in Neurobiology** 2014, **25**:156–163

This review comes from a themed issue on **Theoretical and computational neuroscience**

Edited by **Adrienne Fairhall** and **Haim Sompolinsky**

For a complete overview see the Issue and the Editorial

Available online 5th February 2014

0959-4388/$ – see front matter, © 2014 Elsevier Ltd. All rights reserved.

**http://dx.doi.org/10.1016/j.conb.2014.01.008**

## Introduction

Systems neuroscience is heading towards the simultaneous recording or imaging of many neurons, often while an animal engages in a complicated behavior [1,2••,3,4••,5,63]. This trend has led to a wealth of data that requires new conceptual approaches, methods of analysis, and modeling techniques. For example, recorded neurons often display perplexing activity patterns [6,7,8••,9••,10] that are not easily explained in terms of tuning for sensory parameters or the variety of possible internal parameters that seem natural to an experimentalist. In particular, many studies of higher cortical areas have shown bewildering temporal dynamics at the single-cell and population levels [1,2••,3,4••,5–7,8••] (Ames *et al.*, unpublished data), often while the animal is exposed to constant or simple stimuli. As a result of these observations, many in the field are arriving at the conclusion that it is no longer enough to correlate the firing rates of individual neurons with experimental parameters. Instead, we should delve deeper and attempt to understand the dynamical mechanisms underlying the computations; or at least provide constraints on possible mechanisms [6,7,8••,9••,10,11]. We require examples of the classes of dynamics that do (or do not) allow networks of neurons to perform useful computations.

Experimental work has begun applying dynamical systems approaches [12,13], originally applied in neuroscience to single neurons, to the population responses of simultaneously and individually recorded neurons [1,2••,4••,8••,14–17]. The dynamical systems approach explicitly describes neural population responses as time-varying trajectories in a high-dimensional state space and views the dynamics as acting to shape these trajectories. In this framework individual neurons work in concert to carry out computations. Much as the population vector requires the entire neural population to readout a signal, the dynamical systems approach implies that one must understand the population in order to understand the dynamics of a single neuron.

One model class that can accommodate high-dimensional, distributed and dynamical data is the optimized recurrent neural network (RNN) (Figure 1a). RNNs are a natural model class to study mechanisms in systems neuroscience because they are both dynamical and computational. The purpose of this article is to better introduce RNNs to the field of systems neuroscience. I review recent technical progress in understanding and optimizing RNNs, focusing on a recent result from prefrontal cortex.

## Recurrent neural networks

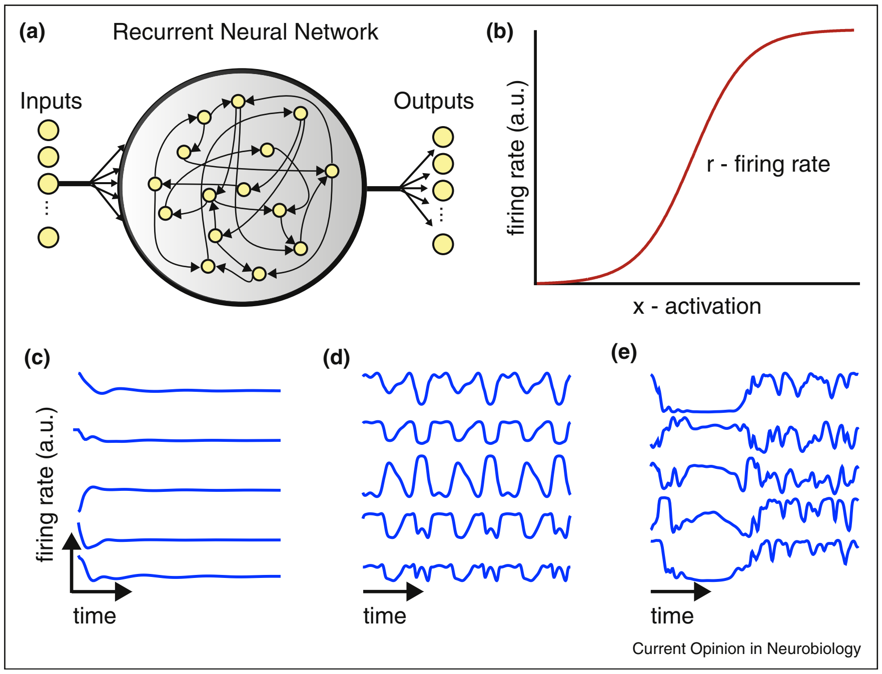
Wilson and Cowan originally developed the recurrent network in a biological context to describe the average firing rates of groups of cells [18]. A more modern and general definition is given by

$$\tau\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \mathbf{J}\mathbf{r}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{b}, \qquad (1)$$

where the $i$th component, $x_i$, of the vector $\mathbf{x}$, can be viewed as the summed and filtered synaptic currents at the soma of a biological neuron. The continuous variable, $r_i$, is the instantaneous "firing rate" and is a saturating nonlinear function of $x_i$ (Figure 1b). Thus, the RNN describes firing rates and does not explicitly model action potentials.

The defining feature of the RNN is the recurrent feedback from one unit in the network to another, which is communicated through the weight matrix, $\mathbf{J}$. The external inputs to the network are represented by the vector of firing rate variables, $\mathbf{u}$, and enter the system through the weights, $\mathbf{B}$. Each unit in the network receives a bias, $b_i$.

**Figure 1**



A recurrent neural network (RNN). **(a)** Highly recurrent circuitry creates a dynamic and distributed computing framework. Inputs come into the network and influence the ongoing dynamics. Outputs are read out by a weighted sum of network firing rates. The connection weights of the network are optimized using an error signal that results from comparing the network output to a desired target signal. **(b)** Each unit has a saturating nonlinearity, which makes the system computationally powerful. **(c–e)** Before or after optimization, units in RNNs can show (c) attractor dynamics, (d) oscillatory dynamics, and (e) extremely complex dynamics that are often associated with chaos. Blue lines show responses through time of 5 units in the RNN.

Finally, the time constant, $\tau$, sets the timescale of the network.

In order to read out the network activity, it is common to include a linear readout

$$\mathbf{z}(t) = \mathbf{W}\mathbf{r}(t). \tag{2}$$

The output, $\mathbf{z}$, is constructed from a weighted linear sum of the network firing rates.

While the RNN is a rather abstract model, it nevertheless shares a number of essential similarities with biological neural circuits. First, the units are nonlinear and there are many of them. Second, the units have strong feedback connections. This generates nontrivial dynamics within the circuit. Third, because the individual units are simple, they must work together in a parallel and distributed fashion to implement complex computations.

RNNs are very powerful. Given enough hidden units, a trained RNN can approximate any dynamical system [19]. RNNs can display arbitrarily complex dynamics, including attractors (Figure 1c), limit cycles (e.g. oscillations, Figure 1d) and chaos (Figure 1e) [20]. One particularly well-known example of an RNN is the Hopfield network [21], which has simple attractor dynamics.

## Optimizing RNNs

In many modeling studies, a network model is designed by hand to reproduce and thus explain a set experimental findings (e.g. see [22]). Here, I will, in contrast, focus on modeling using RNNs that have been optimized, or ‘‘trained’’. For example, assume one wanted to study integration. In the designed approach, a network would be explicitly constructed such that the network integrates an input. Specifically, the weights would be adjusted such that positive feedback internal to the network had a gain near unity. Then the network parameters would be tweaked by hand until it functioned correctly. In contrast, in the optimized approach the desired inputs and outputs are first defined. The input to the network would be the integrand and the output would be the integral. Before training, the initialized network receives the input and produces a meaningless output. This meaningless output, however, is compared to the desired integral, creating an error signal. Such an error signal solely defines what the output of the network should be, and provides information as to how the network weights should be modified. In practice, the negative gradient of the error signal is used to modify the weights of the network, {**J**, **B**, **b**, **W**}, from Eqns 1,2.

Optimizing a network tells the network *what* it should accomplish, with very few explicit instructions on *how* to do it. Therefore, if the mechanisms implemented by RNNs after optimization can be understood, then optimizing and analyzing RNNs becomes a method of *hypothesis generation* for future experiments and data analysis. In particular, the network may solve the problem in a completely unexpected way. Perhaps the optimal (or locally optimal) solution found by optimization may consider minor aspects of the problem that nevertheless influence and change the underlying solution. This approach has early precedents, for example modeling gain fields in parietal cortex [23], and also modeling pattern generation and neural dynamics in the motor system [24,25].

Unfortunately, optimizing an RNN by following the negative gradient of the error signal, a process called ''back-propagation through time'' [26], is extremely difficult. This difficulty is widely appreciated in the machine learning community and was a major impediment to further research [27,28]. Optimizing an RNN has all the problems associated with feed-forward neural networks: non-convexity, overfitting, local optima, and pathological curvature. It is widely believed that increasing the number of layers in a feed-forward network magnifies these problems. Because an RNN can be viewed as an extremely deep feed-forward network (with 100s or even 1000s of layers), these problems are significantly worse in RNNs. As a dynamical system, RNNs are plagued with complicated dynamical behavior, including chaos [20,29,30,31•]; bifurcations, which are sudden changes in dynamical behavior arising from small changes in weights or inputs [32]; and structurally unstable solutions. Because of these challenges, RNNs fell out of favor for well over a decade. Within the last several years, however, due to a deeper understanding of how to manage the dynamical phenomena found in RNNs [29,30,33], and also due to either limiting the scope of the training problem to the readout weights (i.e. training just **W**, known as 'reservoir computing') [30,34–36], or to improved optimization algorithms [37,38•,39–41], RNNs are enjoying a renaissance.

## Reverse engineering an RNN after optimization

Revealing the dynamical mechanisms employed by an RNN to solve a particular task involves a final step after optimization; one must reverse engineer the solution found by the RNN. Because the solution was not constructed by hand, without this step, one simply has another unintelligible network that solves the task of interest. Recently, Omri Barak and I demonstrated that RNNs could be understood by employing techniques from nonlinear dynamical systems theory [42••]. We reverse engineered a variety of RNNs that were optimized to perform simple tasks, for example, a memory device and an input-dependent pattern generator. The key step in reverse engineering involves finding the fixed points of the network (i.e. values of x where the right-hand side of Eqn 1 is zero), and performing a linearization of the network dynamics around those fixed points. The fixed points provide a ''dynamical skeleton'' for understanding the global structure of dynamics in the state-space. Linearization yields a set of linear dynamical systems, each of which approximates the nonlinear dynamics in the vicinity of a particular fixed point in state space, for example [43]. The linear dynamical systems can then be understood with standard tools such as eigenvalue analysis.

Below I describe two examples of this approach, first a simple example and then an application to neural recordings in the prefrontal cortex (PFC). In the latter example an RNN was trained to perform a task analogous to that given to a primate in a neurophysiological study of contextual decision-making. The RNN was then reverse engineered to determine the dynamical solution being used. The network firing rates were then compared with the recorded neural data to determine whether similar solutions might be at play in the PFC.

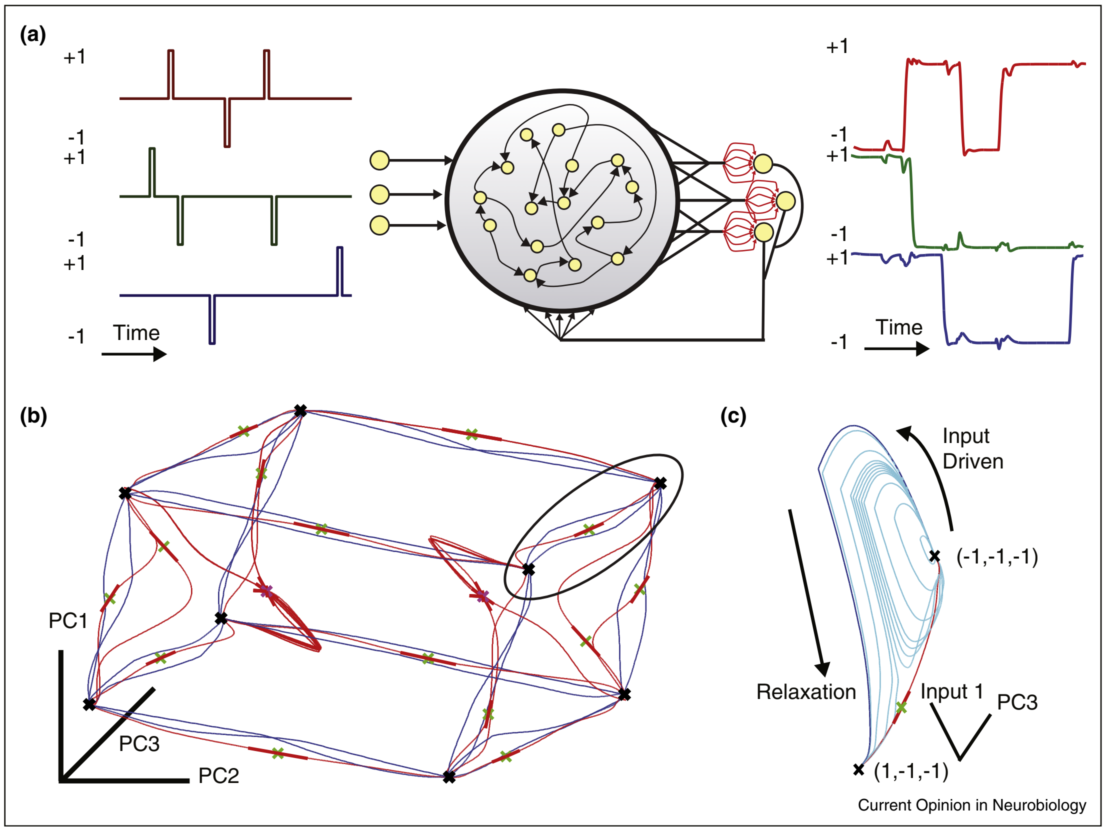
## A 3-bit memory

Understanding how memories can be represented in biological neural networks has long been studied in neuroscience. In this toy example we trained an RNN to generate the dynamics necessary to implement a 3-bit memory (Figure 2a). Three inputs enter the RNN and specify the states of the three bits individually. For example, a +1 input pulse enters the RNN through the first input line. The first output should then transition to +1 if it was at $-1$, or stay at +1 if it already held that value. Additionally, the first output should ignore the values of the second and third inputs. Outputs two and three are defined in the same way for their respective inputs. Thus, this is an example of a 3-bit memory that is resistant to cross talk.

After training [30], the RNN successfully implemented the 3-bit memory.

We reverse engineered the RNN by finding all the fixed points and linear systems around those fixed points. There were $2^3 = 8$ attractors (stable fixed points) in the network (Figure 2b, black 'x') that implemented the 3-bit memory. Additionally, there were many saddle points (green 'x'). A saddle point is a fixed point with both stable and unstable dimensions. In this case, the majority of saddle points had a single unstable dimension with the remaining dimensions being stable. With such a configuration, saddle points can ''direct traffic'' by first attracting

**Figure 2**



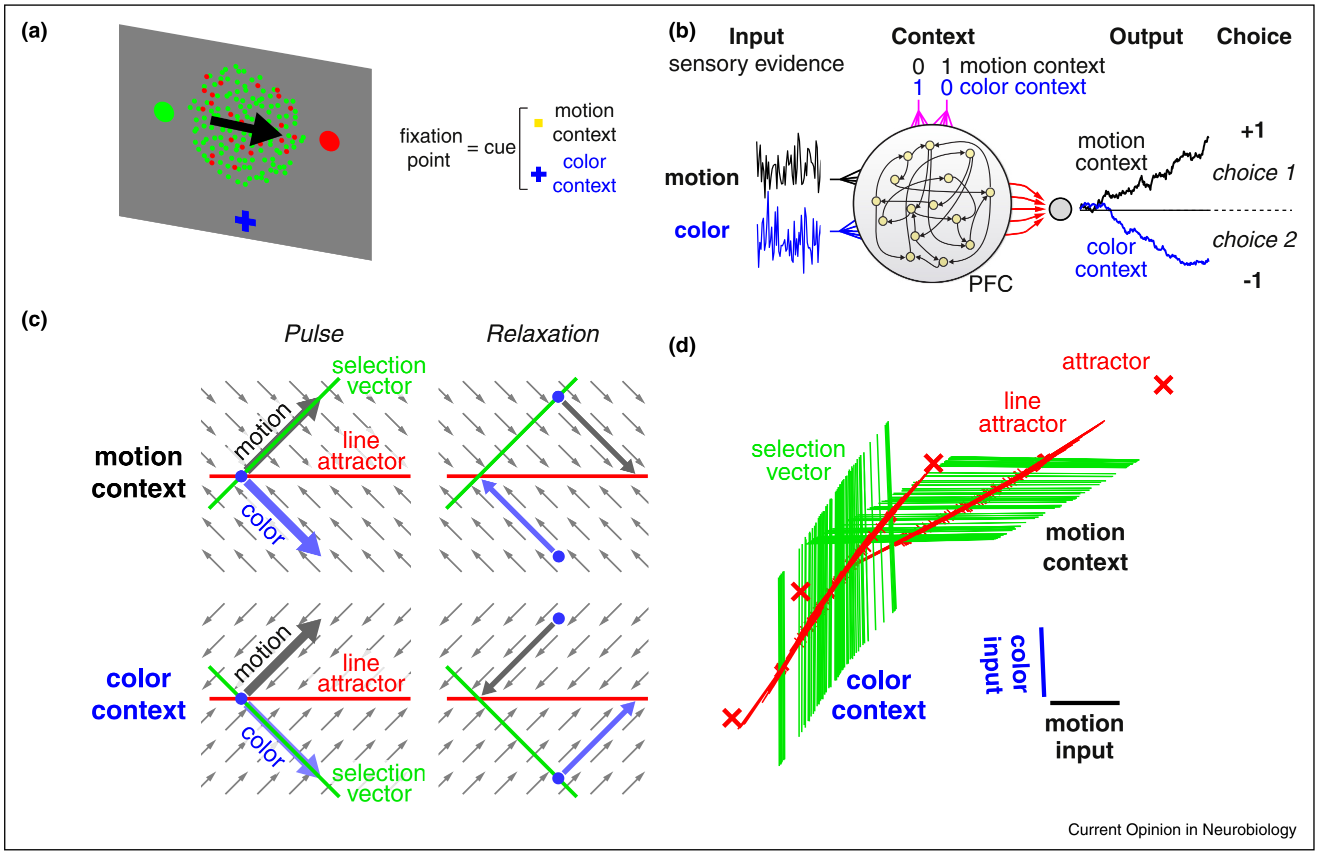
Toy example of a 3-bit memory. **(a)** The inputs (left) and outputs (right) of the 3-bit memory task input/output pairs are dark red/red, dark green/green, and dark blue/blue, respectively). Randomly timed input pulses are injected into the system. A given output should transition to +1(−1) or stay at +1(−1) when the related input pulse is +1(−1). Additionally, an output should ignore any input unrelated to it. The RNN is shown with three inputs and three outputs, used to implement the 3-bit memory task. The output weights are shown in red, and are the only units trained in this example [35]. The output is fed back to the hidden units, thus powerfully affecting the dynamics. **(b)** The 3D state space representation of the 3-bit memory task. A 3-bit memory has 8 possible memory states. These states are represented as stable fixed points (black 'x'). The system trajectories (blue) show transitions from one state to another state, which result from the appropriate input coming into the system (e.g. the output (−1, −1, −1) transitions to (+1, −1, −1) when a +1 pulse is injected into the first input line). Saddle points (green 'x') with only one unstable dimension (thick red lines) mediate transitions between stable states. The thin red lines show trajectories started near the saddle points, which move towards nearby stable fixed points. **(c)** A simulated pulse experiment showing that the saddle points in-between the attractors mediate the transition from one memory to another. Input pulses of varying sizes (blue for normal input pulse, cyan for smaller, perturbed input pulses) are injected into the network while the network is at the attractor associated with memory (−1, −1, −1), see black oval in (b). Initially, the relaxation dynamics result in trajectories that return to the original attractor, but eventually the input pulses are large enough to pass beyond the saddle point so that the resulting trajectories continue on to the fixed point that represents (+1, −1, −1) (reproduced with permission from [42••]).

trajectories towards them (due to the many stable dimensions, which are attracting) and then directing the trajectories by repelling them in one direction or the other along the unstable dimension. In the case of the 3-bit memory, the saddle nodes were responsible for implementing the input-dependent transitions between the stable attractors (Figure 3c). They functioned by aligning their unstable dimension with the axis between two stable attractors. Thus, for a particular input to cause a transition, the input pulse had to push the state trajectory beyond the boundary made by the associated saddle point, after which the relaxation dynamics funneled the state trajectory to the new attractor.

## Context dependent decision making in prefrontal cortex

Animals are not limited to simple stimulus and response reflexes. They can rapidly and flexibly accommodate to context: as the context changes, the same stimuli can elicit dramatically different behaviors [44]. To study this type of contextually dependent decision making [8••], monkeys were trained to flexibly select and accumulate evidence from noisy visual stimuli in order to make a discrimination [45–49]. On the basis of a contextual cue, the monkeys either differentiated the direction of motion or color of a random-dot display (Figure 3a). While the monkeys engaged in the task, neural responses in

**Figure 3**



RNN modeling of contextual decision making in prefrontal cortex. **(a)** Monkeys differentiated either the motion or color of a random dot stimulus that contained both moving and colored dots on all trials (black arrow represents rightward motion). A contextual cue (blue 'plus' for color or yellow 'square' for motion) indicated which of motion or color was relevant. The monkey indicated its choice with a saccade to one of two targets (large green or red circles). **(b)** An analogous RNN model to the monkey task. The input to the network was two streams of white noise ''sensory evidence'' and a binary contextual signal that indicated which of the two streams the network should select/integrate. The mean of the white noise distribution was offset from zero and its sign indicated whether the sensory input was evidence for either choice 1 or choice 2. The output of the network was +1 or −1, indicating choice 1 or choice 2, respectively. All synaptic weights were trained in this model [38•]. **(c)** A cartoon explaining how motion input is integrated in the motion context and color input is integrated in the color context. The motion and color line attractors (red) remain parallel across contexts. To implement selectivity, the selection vector (green) changes with context and aligns to the direction of the relevant stimulus (thick gray arrow in motion context, thick blue arrow in color context). Network dynamics are always orthogonal to the selection vector (thin gray arrows), resulting in integration (a forward step) after a pulse of the relevant stimulus. Following a pulse of irrelevant stimulus, the state returns to the same place on the line attractor. **(d)** Visualization of the 'dynamical skeleton' in the space spanned by the motion and color inputs. Shown are the line attractors (small red 'x' and red lines) and selection vectors (green lines) of the local linear systems. The selection vectors are aligned to the relevant input axis and are orthogonal to the irrelevant input axis. Each line attractor is bounded on both sides by an attracting fixed point (big red 'x'), which represents the decision (+1 or −1) in that context (reproduced with permission from [8••]).

prefrontal cortex (PFC) were recorded. These neurons showed mixed selectivity to both motion and color sensory evidence, regardless of which stimulus was relevant. Further, in MT, an area known to represent visual motion evidence, responses were not significantly different between the motion and color contexts.

These findings are seemingly at odds with the attention literature [44,50–53] because top-down attention is correlated with firing rate modulation in early sensory cortices. One hypothesis inspired by this literature is that contextual decision-making could be solved using the mechanisms of top-down attention, for example, by amplifying the representation of the relevant stimulus dimension in the related cortical areas [50]. By influencing the relative strength of the relevant stimulus in comparison to the irrelevant stimulus, the irrelevant stimulus would be effectively ''gated out''. However, given the findings of the study [8••], this intuitive hypothesis of ''gating the source'' does not appear to be the case for this task.

To discover how a single circuit could selectively integrate one stimulus while ignoring another, despite the presence of both, the RNN approach was applied. Specifically, two white noise inputs were injected into an RNN and the RNN was optimized to emit a +1 (or a $-1$) if the relevant white noise input was drawn from a distribution with a mean greater than zero (less than zero) (Figure 3b). The white noise signals were intended to be analogous to color and motion evidence represented in early cortical visual areas upstream of PFC. The output of the RNN was intended to be analogous to the decision leading to the saccade of the monkey. To indicate which input was relevant, a static contextual input was also injected into the RNN. After optimizing the network, the RNN was analyzed in exactly the same way as the PFC data. The RNN population responses showed strong similarity to PFC population responses [8••].

The RNN was reverse engineered to discover the fixed points, linear dynamical systems, and state-space representations of the task-related signals [8••,42••]. The network's representation of the signals of color evidence, motion evidence, context, and the RNN's choice were represented in highly separable axes in state space. The attracting fixed points for a given context (e.g. when color is relevant) were arranged along a line, which is called a line attractor [54–56]. Line attractors are a dynamical mechanism for the accumulation of evidence towards a choice. In this case, approximate line attractors implemented the mechanism for the integration of the relevant noisy input stream.

This model differed from other applications of line attractors in that the full solution to the problem of contextual integration was embedded in a nonlinear system. The nonlinearity allowed the implementation of two approximate line attractors, one for integration of motion evidence and the other for color evidence. In both contexts, and along every point of the line attractor, the RNN implemented the selectivity by using a non-normal linear dynamical system [57–59]. The defining feature of non-normal linear systems is the separation of their left and right eigenvectors. In a normal linear system (where the right and left eigenvectors are the same), one can reason about the effects of an input by examining the projection of the input vector onto eigenvectors of interest. In a non-normal system, the effect of an input relates to both the left and right eigenvectors and can be much more difficult to understand.

Relevant to this study, the line attractor (right zero eigenvector) and selection vector (left zero eigenvector) differed in orientation in state space. Specifically, the RNN implemented selectivity by aligning the selection vector to have a significant projection onto the direction of the relevant input and aligning it perpendicular to the direction of the irrelevant input (Figure 3c). So the selection vector changed orientation with context, while the line attractor did not. Changing the selection vector in a context dependent manner is a sensible solution to the contextual integration problem because it is the projection of the input onto the selection vector that defines the integrand. Further, because the network dynamics are always orthogonal to the selection vector (Figure 3c), aligning the selection vector to the relevant input allows the relevant input to move the system farther along the line attractor, thus accumulating evidence for a choice. Aligning the selection vector orthogonal to the irrelevant input prevents the irrelevant input from contributing to the accumulated evidence. The full nonlinear, global arrangement of the 'dynamical skeleton' is shown in Figure 3d.

The solution found by optimizing the RNN is not at all obvious *a priori*. Indeed, to the best of my knowledge a general solution to the problem of context-dependent selection has not been proposed. Many studies have shown that prefrontal cortex appears to represent large numbers of complex feature attributes (e.g. [6,9••,10]). It seems clear that the ability to learn new tasks requires that such representations be combined in a dynamic and flexible fashion. The mechanism of orienting the selection vectors easily generalizes to multiple inputs and may tell us something of the operation of prefrontal cortex more generally. Finally, a direct prediction of this model is that a pulse stimulation protocol aligned to either the motion or color axes should result in differential network decay dynamics across contexts.

## Conclusions

The study of neural dynamics at the circuit and systems level is an area of extremely active research. RNNs are a near ideal modeling framework for studying neural circuit dynamics because they share fundamental features with biological tissue, for example, feedback, nonlinearity, and parallel and distributed computing. Many challenges remain, however. Currently, RNNs do not take anatomy into account, other than the existence of feedback. Anatomical features such as columnar structure, structure of local cortical circuits, cell type, and projection patterns are completely absent. As this information becomes increasingly known, the RNN synaptic weights can be initialized and constrained with this information and the same framework will still apply. A core theoretical issue that must be addressed, despite the mathematical difficulty, is that very little is understood about non-autonomous dynamical systems (i.e. those with inputs) [12,60]. Finally, many researchers are actively studying how generic dynamical systems such as RNNs can be embedded in spiking neural networks [61,62•].

By training RNNs on what to compute, but not how to compute it, researchers can generate novel ideas and testable hypotheses regarding biological circuit mechanism. Further, RNNs provide a rigorous test bed in which to test ideas related to neural computation at the network level. The combined approaches of animal behavior and neurophysiology, alongside RNN modeling, may prove a powerful combination for handling the onslaught of high-dimensional neural data that is to come.

## Acknowledgements

I am grateful to Larry Abbott, Mark Churchland, Valerio Mante and Krishna Shenoy for helpful discussions and feedback. This work was supported in part by DARPA Reorganization and Plasticity to Accelerate Injury Recovery (REPAIR) award (N66001-10-C-2010) and an NIH Director's Pioneer Award (1DP1OD006409).

## References and recommended reading

Papers of particular interest, published within the period of review, have been highlighted as:

- • of special interest
- •• of outstanding interest

1. Ahrens MB, Li JM, Orger MB, Robson DN, Schier AF, Engert F, Portugues R: **Brain-wide neuronal dynamics during motor adaptation in zebrafish**. *Nature* 2012, **485**:471-477.

2. •• Churchland MM, Cunningham JP, Kaufman MT, Foster JD, Nuyujukian P, Ryu SI, Shenoy KV: **Neural population dynamics during reaching**. *Nature* 2012, **487**:51-56.
This manuscript gives a compelling example of population dynamics in motor cortex.

3. Crowe DA, Averbeck BB, Chafee MV: **Rapid sequences of population activity patterns dynamically encode task-critical spatial information in parietal cortex**. *J Neurosci* 2010, **30**:11640-11653.

4. •• Harvey CD, Coen P, Tank DW: **Choice-specific sequences in parietal cortex during a virtual-navigation decision task**. *Nature* 2012, **484**:62-68.
The authors find strong population dynamics in posterior parietal cortex of mice running in a T-maze in which a high-dimensional representation is used to encode spatial location along the track.

5. Ponce-Alvarez A, Nácher V, Luna R, Riehle A, Romo R: **Dynamics of cortical neuronal ensembles transit from decision making to storage for later report**. *J Neurosci* 2012, **32**:11956-11969.

6. Jun JK, Miller P, Hernández A, Zainos A, Lemus L, Brody CD, Romo R: **Heterogeneous population coding of a short-term memory and decision task**. *J Neurosci* 2010, **30**:916-929.

7. Churchland MM, Shenoy KV: **Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex**. *J Neurophysiol* 2007, **97**:4235-4257.

8. •• Mante V, Sussillo D, Shenoy KV, Newsome WT: **Context-dependent computation by recurrent dynamics in prefrontal cortex**. *Nature* 2013, **503**:78-84.
The authors provide a detailed example of how the mechanisms of a dynamical computation can be discovered in an RNN, providing a powerful hypothesis for the underlying mechanism in a complex cortical circuit.

9. •• Rigotti M, Barak O, Warden MR, Wang X-J, Dew ND, Miller EK, Fusi S: **The importance of mixed selectivity in complex cognitive tasks**. *Nature* 2013 http://dx.doi.org/10.1038/nature12160.
This manuscript argues persuasively that mixed selectivity, a signature of high-dimensional neural representations, is a fundamental component of the computational power of prefrontal cortex.

10. Roy JE, Riesenhuber M, Poggio T, Miller EK: **Prefrontal cortex activity during flexible categorization**. *J Neurosci* 2010, **30**:8519-8528.

11. Barak O, Sussillo D, Romo R, Tsodyks M, Abbott LF: **From fixed points to chaos: three models of delayed discrimination**. *Prog Neurobiol* 2013, **103**:214-222.

12. Rabinovich M, Varona P, Selverston A, Abarbanel H: **Dynamical principles in neuroscience**. *Rev Mod Phys* 2006, **78**:1213-1265.

13. Shenoy KV, Sahani M, Churchland MM: **Cortical control of arm movements: a dynamical systems perspective**. *Annu Rev Neurosci* 2013 http://dx.doi.org/10.1146/annurev-neuro-062111-150509.

14. Broome BM, Jayaraman V, Laurent G: **Encoding and decoding of overlapping odor sequences**. *Neuron* 2006, **51**:467-482.

15. Machens CK, Romo R, Brody CD: **Flexible control of mutual inhibition: a neural model of two-interval discrimination**. *Science* 2005, **307**:1121-1124.

16. Stopfer M, Jayaraman V, Laurent G: **Intensity versus identity coding in an olfactory system**. *Neuron* 2003, **39**:991-1004.

17. Jones LM, Fontanini A, Sadacca BF, Miller P, Katz DB: **Natural stimuli evoke dynamic sequences of states in sensory cortical ensembles**. *Proc Natl Acad Sci U S A* 2007, **104**:18772-18777.

18. Wilson H, Cowan J: **Excitatory and inhibitory interactions in localized populations of model neurons**. *Biophys J* 1972, **12**:1-24.

19. Doya K: *[Proceedings] 1992 IEEE International Symposium on Circuits and Systems*. IEEE; 1992:2777-2780.

20. Sompolinsky H, Crisanti A, Sommers H: **Chaos in random neural networks**. *Phys Rev Lett* 1988, **61**:259-262.

21. Hopfield JJ: **Neural networks and physical systems with emergent collective computational abilities [Internet]**. *Proc Natl Acad U S A* 1982, **79**:2554-2558.

22. Wang X-J: **Neural dynamics and circuit mechanisms of decision-making**. *Curr Opin Neurobiol* 2012, **22**:1039-1046.

23. Zipser D, Andersen RA: **A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons**. *Nature* 1988, **331**:679-684.

24. Fetz EE: **Cortical mechanisms controlling limb movement**. *Curr Opin Neurobiol* 1993, **3**:932-939.

25. Yu BM, Afshar A, Santhanam G, Ryu SI, Shenoy KV, Sahani M: **Extracting dynamical structure embedded in neural activity**. *Neural Inform Process Syst (NIPS)* 2006, **18**:1545-1552.

26. Werbos PJ: **Backpropagation through time: what it does and how to do it**. *Proc IEEE* 1990, **78**:1550-1560.

27. Bengio Y, Simard P, Frasconi P: **Learning long-term dependencies with gradient descent is difficult**. *IEEE Trans Neural Netw* 1994, **5**:157-166.

28. Hochreiter S, Schmidhuber J: **Long short-term memory**. *Neural Comput* 1997, **9**:1735-1780.

29. Rajan K, Abbott L, Sompolinsky H: **Stimulus-dependent suppression of chaos in recurrent neural networks**. *Phys Rev E* 2010, **82**:82.

30. Sussillo D, Abbott LF: **Generating coherent patterns of activity from chaotic neural networks**. *Neuron* 2009, **63**:544-557.

31. • Laje R, Buonomano DV: **Robust timing and motor patterns by taming chaos in recurrent neural networks**. *Nat Neurosci* 2013 http://dx.doi.org/10.1038/nn.3405.
The authors demonstrate that extremely complex dynamics could in principle be used to encode time in biological neural networks.

32. Doya K: **Bifurcations in the learning of recurrent neural networks**. In *ISCAS'92. Proceedings, 1992 IEEE International Symposium on Circuits and Systems, vol 6*. 1992:2777-2780.

33. Jaeger H: **The "echo State": approach to analysing and training recurrent neural networks**. *publica.fraunhofer.de* 2001.

34. Maass W, Natschläger T, Markram H: **Real-time computing without stable states: a new framework for neural computation based on perturbations**. *Neural Comput* 2002, **14**:2531-2560.

35. Jaeger H, Haas H: **Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication**. *Science* 2004, **304**:78-80.

36. Sussillo D, Abbott LF: **Transferring learning from external to internal weights in echo-state networks with sparse connectivity**. *PLoS ONE* 2012, **7**:e37372.

37. Martens J: **Deep learning via Hessian-free optimization**. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*. 2010.

38. Martens J, Sutskever I: **Learning recurrent neural networks with**
• **Hessian-free optimization**. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*. 2011.

An important paper in the machine learning community that shows that RNNs can be optimized to solve the so-called ''pathological temporal problems'', thus rejuvenating the study of RNNs in the machine learning field.

39. Martens J, Sutskever I: **Training deep and recurrent networks with Hessian-free optimization**. *Neural Networks: Tricks of the Trade*. Berlin/Heidelberg: Springer; 2012, 479-535.

40. Sutskever I: *Training recurrent neural networks*. . (PhD thesis) 2013.

41. Bengio Y, Boulanger-Lewandowski N, Pascanu R: **Advances in optimizing recurrent networks**. *arXiv* 2012, **cs.LG**

42. Sussillo D, Barak O: **Opening the black box: low-dimensional**
•• **dynamics in high-dimensional recurrent neural networks**. *Neural Comput* 2013, **25**:626-649.

This paper provides the critical link between viewing RNNs as neural networks and also as dynamical systems. Often RNNs are considered 'black-box' approaches, implying that their mechanism cannot be understood. However, the paper shows that in simple cases an RNN can be 'reverse engineered' to reveal its underlying dynamical mechanism.

43. Rabinovich MI, Huerta R, Varona P, Afraimovich VS: **Transient cognitive dynamics, metastability, and decision making**. *PLoS Comput Biol* 2008, **4**:e1000072.

44. Miller EK, Cohen JD: **An integrative theory of prefrontal cortex function**. *Annu Rev Neurosci* 2001, **24**:167-202.

45. Shadlen MN, Newsome WT: **Motion perception: seeing and deciding**. *Proc Natl Acad Sci U S A* 1996, **93**:628-633.

46. Shadlen MN, Newsome WT: **Neural basis of a perceptual decision in the parietal cortex (area LIP) of the rhesus monkey**. *J Neurophysiol* 2001, **86**:1916-1936.

47. Gold JI, Shadlen MN: **The neural basis of decision making**. *Annu Rev Neurosci* 2007, **30**:535-574.

48. Roitman JD, Shadlen MN: **Response of neurons in the lateral intraparietal area during a combined visual discrimination reaction time task**. *J Neurosci* 2002, **22**:9475-9489.

49. Kiani R, Hanks TD, Shadlen MN: **Bounded integration in parietal cortex underlies decisions even when viewing duration is dictated by the environment**. *J Neurosci* 2008, **28**:3017-3029.

50. Desimone R, Duncan J: **Neural mechanisms of selective visual attention**. *Annu Rev Neurosci* 1995, **18**:193-222.

51. Moran J, Desimone R: **Selective attention gates visual processing in the extrastriate cortex**. *Science* 1985, **229**:782-784.

52. Reynolds JH, Chelazzi L: **Attentional modulation of visual processing**. *Annu Rev Neurosci* 2004, **27**:611-647.

53. Noudoost B, Chang MH, Steinmetz NA, Moore T: **Top-down control of visual attention**. *Curr Opin Neurobiol* 2010, **20**:183-190.

54. Seung HS: **How the brain keeps the eyes still**. *Proc Natl Acad Sci USA* 1996, **93**:13339-13344.

55. Lim S, Goldman MS: **Noise tolerance of attractor and feedforward memory models**. *Neural Comput* 2012, **24**:332-390.

56. Song P, Wang X-J: **Angular path integration by moving ''hill of activity'': a spiking neuron model without recurrent excitation of the head-direction system**. *J Neurosci* 2005, **25**:1002-1014.

57. Ganguli S, Huh D, Sompolinsky H: **Memory traces in dynamical systems**. *Proc Natl Acad Sci USA* 2008, **105**:18970-18975.

58. Goldman MS: **Memory without feedback in a neural network**. *Neuron* 2009, **61**:621-634.

59. Murphy BK, Miller KD: **Balanced amplification: a new mechanism of selective amplification of neural activity patterns**. *Neuron* 2009, **61**:635-648.

60. Manjunath G, Jaeger H: **Echo state property linked to an input: exploring a fundamental characteristic of recurrent neural networks**. *Neural Comput* 2013, **25**:671-696.

61. Eliasmith C: **A unified approach to building and controlling spiking attractor networks**. *Neural Comput* 2005, **17**:1276-1314.

62. Eliasmith C, Stewart TC, Choo X, Bekolay T, DeWolf T, Tang Y,
• Tang C, Rasmussen D: **A large-scale model of the functioning brain**. *Science* 2012, **338**:1202-1205.

The authors used optimization techniques as well as specific hypotheses about network functionality to build a very large spiking neural network that can successfully achieve many behaviors.

63. Ames KC, Ryu SI, Shenoy KV: **Neural dynamics of reaching following incorrect or absent motor preparation**. *Neuron* 2014, **81**:438-451 http://dx.doi.org/10.1016/j.neuron.2013.11.003.