Random matrices and attractors – technical exposé
Morgan ~ 9/21
DRAFT – a thoughtdump!

I'm interested in understanding matrices with multiple eigenvectors of maximal eigenvalue 1 in order to identify the potential of recurrent neural network attractors. To do so, I'd investigate the attractors emerging from large random matrix processes and their relative properties. This investigation might identify a way that neural networks could implement finite state machines and other virtual neural networks.

Some background on stochastic matrices. Any real stochastic matrices (Markov chain processes) with (contingently) a spectral radius of 1 having a *single* eigenvector of eigenvalue 1 (ie and where all other eigenvectors have eigenvalues less than 1 in absolute value), then repeated application of the matrix operator is guaranteed to converge to that eigenvector as the unique stationary lowest-energy distribution state from any initial state. When there are more than one eigenvector sharing eigenvalue 1, each of these stationary distributions are (equally??) likely, stationary, stable, and lowest-energy, and the one converged to depends on the initial distribution; [[recursive macro on convexity of stationaries?]]. (However, a stochastic matrix with any amount of randomness will almost surely not have its eigenvalue 1 have multiplicity greater than 1.)

In linear neural networks, training* adjusts the feedforward matrix layer towards being a matrix of orthonormal eigenvectors*. Specifically, the network tends to focus its training improvements on the single strongest mode until it's learned fully and then proceeding to optimize for the next mode, up until the matrix is saturated into a full-rank suite of eigenvectors. (This mode-by-mode behavior concords with the Low-Rank Approximation via Eckart–Young–Mirsky theorem.) Therefore, the number of stationary optimal distributions tends to increase.

Attractors, as they are called in dynamical systems, are observed in biological neural network (such as in grid and place cells*) and in idealized/theoretical recurrent models such as the Hopfield network. From work in dynamical systems, it's well known that attractor states are located at eigenvector states. This is no coincidence: it is a consequence of Markovity / ergodicity of such a system. However, it is also well known that in grid and place cell ensembles, states can be held stationary in one of *many* attractors. For example, head direction cells implement ring attractors, a collection of attractors that are individually state-stationary but permit states to move to adjacent states if the creature moves (and integrates) its head direction angle. The presence of many attractors accords with the expectation that these an adult brain's neuron ensembles for fundamental motion and sense of location and direction are more or less fully-trained.

The open questions lie in understanding multiple attractors. What are their shapes? (This is somewhat solved. I'm still digesting the results.) What are their interplay? (This is likely solved/resolved, and I've yet to locate the results.) What does it take to move a state from one attractor to the next? (A relatively easy problem amounting to orthogonal vector arithmetic for some kinds of matrices. Some results: such transitions must be symmetric ,by maybe 'detailed reversibility' $M_{ij} p_i = M_{ji} p_j$?; MCMC methods or temperature-based simulated annealing procedure would be available for experiments; ...). How do attractor-bumping external inputs interact with the attractor eigenstates especially as they train -- need they require an external utility such as a handling network? And, what I'm most curious about, how do the answers to these questions shed light on the plausibility of neurons implementing a virtual state-machine automaton*? If these attractor machines are implemented by parallel distributed neural networks and cursorily resemble the activity of neurons themselves* (and would therefore be recursive), do they implement *distributed* state machines, and what are the properties of such distributed automata?

Other questions of related but tangential curiosity and would benefit first from a careful literature

review: Do distributed attractor systems have a fundamental distinction between states and actions between the states? Neuron trained knowledge held in synapses is somewhat inextricable from activity invoking those synapses. Do tensor-product units or high-order networks have a role in random matrix attractors for neural networks, such as in the LSTM?

I propose:

a) examining the eigenspectra of different kinds of random matrices as models of linear neural network layers, such as those defined by stochastic matrices, matrices those with restricted values such as ternary (a la Hopfield networks), binary (for which there seems to be substantial work done especially in spectral graph theory), nearly-orthogonal, network layers along training course, etc. I've started running simulations of the eigenspectra of such matrices in <https://github.com/taoketao/random_matrices_and_eigenvalues>. In accordance with the literature, the distributions of eigenvalues of many matrices follow the Marchenko-Pastur distribution, which is characterized by the motif y=sqrt(1/x-1). I hope to develop some insight into quasi-eigenvalue 'stationary distributions' of proper neural networks layers, i.e., linear matrix operators adorned with nonlinear activation functions; cf. “Correlation Between Eigenvalue Spectra and Dynamics of Neural Networks” by Qingguo Zhou, Tao Jin, Hong Zhao, 2009.

b) simultaneously, continue & finish reviewing and digesting the results and answers to the questions I've asked that already exist in the literature. There's plenty to learn from specific papers and from the general study of stochastic processes, dynamical systems, thermodynamics, spin glasses, the expository and empirical work and mathematical treatments of Hopfield networks / Boltzmann machines, linear algebra, theory of computation, the works of Saxe, Ganguli, Lake, Smolensky, McClelland and Rumelhart, high-dimensional applied statistics and optimization (i.e. 'machine learning'), deep learning frontiers, and data from neuroscience and behavioral cognitive science and spatial cognition.

c) subsequently, gain familiarity with and intuition for multiple-attractor systems.

d) subsequently, make concrete hypotheses and questions about attractors as automata.

e) simultaneously, make concrete mathematical analyses or hypotheses+questions about the recursive implementation of neurons and consequences of a positive result.

* 'training': rigor and specification pending. See Saxe, McClelland, and Ganguli, 2013 and 2019. And, generally speaking, while these results have been suggested to be similar for nonlinear layers (pers. comm. PDP lab group ~2017 by jlmcc, Stephen Hansen, Andrew Lampinen), this kind of analysis has not to my knowledge been widely applied to recurrent network layers. Furthermore, the connection between gradual acquisition of orthonormal eigenvectors which correspond to attractors is not *tied* to the question of point, the behavior of multiple viable attractors.
* 'orthonormal eigenvectors': if the orthogonal eigenvectors are not normalized, proper normalization can be achieved by, for example, competitive learning paradigms embraced in theoretical neuroscience, rendering this a minor concern if any. Source: Andrew Lampinen, pers. comm. Rigor pending.
* 'grid and place cells': while it is relatively common knowledge that these work by means of attractors, the statement could use direction to evidence and support
* concerning neural networks as finite-state automata: read, for example, “Turing computability with neural nets, Siegelmann and Sontag, 1991” and its references.
* 'resemble the activity of neurons themselves': A conjecture! But intuitively plausible: neurons, which hold (fleeting) activation states accessible to many other external processes -- as do attractors -- *and* are modified by inputs with specific linear-hyperplane thresholds akin to the (linear-hyperplane?) edge of two attractor basins, do parallel distributed computing. Based on these characteristics, it would seem

that a recurrent layer of neurons could itself implement a virtual recurrent layer of neurons – albeit, about 0.138 times as large as the base. The question lies in the consistency (~systematicity) of activations between neurons and state transitions, the vertical interactions between recursive levels of hypothetical virtual neural network layers.