Statistics and Its Interface Volume 3 (2010) 159–168

Optimal and robust design for efficient system-wide synchronization in networks of randomly-wired neuron-nodes

Michael P. McAssey*, Fushing Hsieh† and Emilio Ferrer

We describe a mode of signal transmission among neuron-nodes in a finite deterministic network. In this mode, an activated node passes a signal to its nearest neighbors and becomes deactivated, unless it concurrently receives the signal from a nearest neighbor. By means of matrix representation, we show that a connected network equipped with this mode of signal transmission converges to one of two states: 1) System-Wide Synchronization (SWS), wherein all nodes are activated; and 2) Subgroup Alternation (SGA), wherein two subsets of nodes alternate on and off. Conditions on wiring configuration required for SWS are presented. We then focus on finite random networks in which the presence of wiring between any two nodes is stochastically determined. We consider two optimal design problems: 1) How can we allocate wiring probabilities subject to a budget constraint in order to maximize the probability of achieving SWS? 2) What impact does a robustness criterion have on the optimal wiring structure? We implement the simulated annealing algorithm to find such optimal probability allocations and present our results. Under a robustness requirement, we show that robust random networks require a larger budget and significantly more triangular-loop sub-structures.

Keywords and phrases: Random network, Simulated annealing, System-wide synchronization.

1. INTRODUCTION

This study was motivated in part by a study of network coherence of emotion variables [7] and a book on brain rhythms [2]. In the emotion study, tight connections with strong wiring potentials are found in several subgroups among a collection of emotion variables. In contrast, sparse connections and weak wiring are observed between subgroups. This result immediately leads to the following question: Does this empirical network structure efficiently lead to system-wide synchronization as a phenomenon of emotion arousal? In addition, the system-wide synchronization of neuron-firing is believed to be closely related to memory reactivation in the animal brain when it is sleeping. However, many memories must be relocated from one part of the hippocampus to the other parts of the brain, and different memories are believed to be stored in different locations within the hippocampus. Thus a system of neurons responsible for one memory needs to be activated as one whole in a very efficient manner. Contemplating representations of the signal transmission in network models of these phenomena has led to the innovative approach presented here.

An artificial neural network (ANN) is an interconnected group of artificial neurons that uses a mathematical or computational model for information processing based on a connectionistic approach to computation. The study of ANNs is motivated by their comparability to biological neural networks. They are investigated in order to increase understanding about their biological counterparts, and to use the functional power of biological neural networks to guide the development of modern technology. In particular, technicians seek to adapt the brain's capacity for self-organization, learning, generalization and fault-tolerance. Introductions to ANNs and their applications may be found in [5, 6, 9, 10] and [12]. The use of ANNs to study the dynamics of neurological networks in the brain of a behaving animal is ubiquitous throughout the neuroscience literature, as in [4].

In this paper we consider a simple ANN consisting of a fixed number of neuron-like hypothetical nodes (or "neurodes"). We propose a novel approach to modeling signal transmission among these nodes as follows: a node becomes activated at a point in time when it receives a signal from any of its nearest neighbors, in a manner similar to the accumulation of action potential in a neuron. And, similar to the discharge of that accumulated action potential, every activated node transmits its signal to all its immediate neighbors at the next time point after activation and instantly becomes deactivated, until reactivated by a neighboring node. In this simulation of signal transmission in biological neural networks, nodes change activation states within a discrete time resolution.

We call one particular phase of interest pertaining to the network under consideration *system-wide synchronization* (SWS), which we specifically define as that state in which all nodes simultaneously activate all their nearest neighbors and are activated by all their nearest neighbors. We make

*All three authors are supported in part by the NSF on Grant No: BCS 0527766.
†Corresponding author.

use of this SWS phase to represent the synchronization mentioned in the above two motivating examples. In regard to this phase, we first study the kind of wiring configuration on a deterministic network that produces an efficient SWS phase. We then proceed to consider random networks. Here we consider the wiring between any two nodes to be governed by an independent Bernoulli random variable. This consideration reflects the fact that two emotion variables or two neurons may not be invariantly wired together at all times when responding to all stimuli. Furthermore, this randomness is a valid approach because the true dynamics underlying this wiring are still not yet understood, especially in neuroscience.

It is also known that not all neurons are wired together equally-well at all times, nor are the emotion variables. Thus it is important to consider the efficiency of achieving SWS upon a class of random networks subject to a budget of total probabilities for all potential wirings. Furthermore, sometimes a neuron or emotion variable in a network may malfunction. Ideally, the remaining nodes in the network should still perform and achieve the SWS phase. This leads to the necessity of robustness with respect to a malfunctioning node in the network. Hence we study the following two optimal design problems:

Q1: How can we allocate wiring probabilities subject to a budget constraint in order to maximize the probability of achieving SWS?

Q2: What impact would a robustness criterion have on this wiring structure?

To resolve the above two problems, we employ the Simulated Annealing (SA) algorithm as an optimization technique. We envision no analytic solution, but obtain numerical ones, since the number of potential wirings grows as the square of the number of nodes in the network. For example, 15 nodes will give rise to 105 potential wirings in the network. Here the computations needed for optimizing a 105-dimensional function are overwhelming. However, results we obtain in this paper as we pass from deterministic networks to random networks, and then to random networks with a robustness criterion, allow us to reduce the computational complexity to a rather manageable level. The energy functions employed in the SA algorithm are developed accordingly.

This paper is organized as follows. We first describe a mode of signal transmission in a connected deterministic network in which the nodes change activation states as described above, and show that one of two distinct phases will occur: system-wide synchronization (SWS), or subgroup alternation (SGA). We then obtain both geometric and algebraic criteria by which we may determine which phase a network will generate. Next, we consider random networks under a constraint on the sum of the edge-probabilities, and search for an optimal allocation of the edge-probabilities such that SWS networks will occur with high probability, using the simulated annealing algorithm. Then we impose a further goal of finding an optimal allocation that will frequently generate SWS networks that are also robust against loss of a node, and present our findings. We conclude with remarks on potential applications of our findings.

2. DETERMINISTIC NETWORKS AND SIGNAL TRANSMISSION

Consider a network $\mathcal{N}$ consisting of n nodes, arbitrarily labeled $1, 2, \ldots, n$, and let the set of nodes be denoted $\{\mathcal{N}\}$. An edge (or wiring) between nodes i and j may be denoted $e_{i,j}$, and the set of all edges in network $\mathcal{N}$ is thus denoted $\mathscr{E}\{\mathcal{N}\}$. Note that the cardinality of $\mathscr{E}\{\mathcal{N}\}$ is at most $n(n-1)/2$. In a neural network, nodes represent neurons and edges represent the connections between them. We will regard a network as a system through which a signal travels from node to node by means of available edges. The definitions of all terms and notation used in the foregoing discussion, when not provided, may be found in the literature, e.g., in [1, 11] or [13].

Initially, we will regard every node in a network $\mathcal{N}$ to be in an "off" state, that is, *deactivated*. Then at some moment, say, at step 0, node i switches to an "on" state, i.e., is *activated*. It may be that a signal was applied to node i from some external source, or some internal process activated it. This signal will then be transmitted throughout the network in discrete time steps, activating other nodes as it reaches them. However, once the signal is transmitted from any node to its immediate neighbors, that node deactivates until the signal returns at a later step. The only exception occurs if the node receives the signal back from a neighbor during the same step in which it also transmits the signal. Hence, at step 1 each neighbor of node i will be activated, but node i will again become deactivated. At step 2, the neighbors of the neighbors of node i (which include node i itself) will be activated, while the neighbors of node i are deactivated. We will assume that the signal then continues to be transmitted indefinitely in this manner at steps $3, 4, \ldots$. This mode of signal transmission, in which nodes switch to an "off" state until turned "on" by neighboring nodes, is intended to represent the behavior of neurons in the brain.

Note that, if $\mathcal{N}$ is a connected network, then its diameter with respect to node i (i.e., the shortest path between node i and the node most distant from it) cannot exceed $n-1$ for each $i = 1, \ldots, n$, so that the overall diameter $D(\mathcal{N})$ of the network may not exceed $n-1$ (a connected network consisting of a string of n consecutive nodes, with $n-1$ edges linking them, has the maximum possible diameter). Clearly then, a signal which originates at any node in a connected network $\mathcal{N}$ will have been propagated at least once to every other node within $n-1$ steps. Our interest is in network configurations which result in the simultaneous and sustained activation of all nodes after finitely many steps.

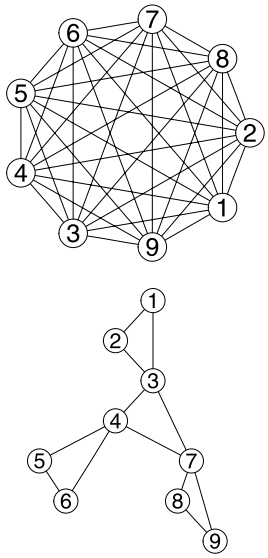

Figure 1. Networks $\mathcal{N}_A$ and $\mathcal{N}_B$.

For example, suppose a network consists of nine nodes, as in Figure 1. Note that network $\mathcal{N}_A$ is fully connected, while network $\mathcal{N}_B$ is connected, but not fully. Each network can be represented by a symmetric 9×9 connectivity matrix, with zeros on the main diagonal, and with m_{ij} indicating the presence of an edge between nodes i and j, as shown below:

$$M(\mathcal{N}_A) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$M(\mathcal{N}_B) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Signal transmission within a network $\mathcal{N}$ is modeled by a variation of matrix multiplication. Define the *state* of node i at step k, denoted $s_k(i)$, such that $s_k(i) = 1$ if node i is activated at step k, and $s_k(i) = 0$ if node i is deactivated at step k. Let the *network state vector* $\mathbf{v} = (v_1, \ldots, v_n)$ at step k be a vector of length n such that, at step k, $v_i = s_k(i)$ for $i = 1, \ldots, n$. Thus at step 0 the network state vector consists of a 1 in the position corresponding to one of the nodes, and a 0 in every other position. At subsequent steps, the elements of the vector will change to reflect the changes in the state of the network. In this setting, the product $[M\mathbf{v}]$ of the matrix $M = M(\mathcal{N})$ with $\mathbf{v}$ at step k produces the network state vector $\mathbf{v}$ at step $k+1$, where $[M\mathbf{v}]$ is found by computing the usual product of a matrix with a vector, but with the restriction that all non-zero entries in the result are set equal to one. Hence if $\mathbf{v}$ represents the state of the network at step 0, then $[M^k\mathbf{v}]$ represents the state of the network at step k.

To illustrate, for both networks $\mathcal{N}_A$ and $\mathcal{N}_B$ we display the evolution of the network state vector at steps 0 through 5 if the signal originates at node 1:

Network state vector for $\mathcal{N}_A$, steps 0 through 5

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \cdots$$

Network state vector for $\mathcal{N}_B$, steps 0 through 5

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \cdots$$

Notice that, by step 2, the network state vector for $\mathcal{N}_A$ consists of a 1 in each position, indicating that all nine nodes are simultaneously activated, and remains in this state thereafter. We describe this phenomenon as system-wide synchronization (SWS). Hence network $\mathcal{N}_A$ has achieved SWS after two steps. The same result occurs with the network state vector for $\mathcal{N}_B$, but not until step 3. For either network, if we instead begin the signal at a different node, the network still achieves SWS, but not necessarily after the same number of steps, because the signal will eventually reach node 1. In general, let us say that a connected network is SWS if and only if there exists some step k at which all of its nodes are concurrently activated. Let us define the *order* of a SWS network $\mathcal{N}$ *with respect to* node i, denoted $O_i(\mathcal{N})$, as the minimum number of steps from activation of node i at step 0 until SWS occurs. Then the *order* of a SWS

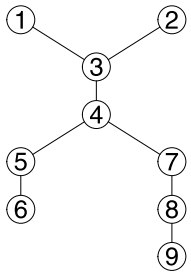

Figure 2. Network $\mathcal{N}_C$.

network $\mathcal{N}$ having n nodes, denoted $O(\mathcal{N})$, may be defined as the maximum over $i = 1, \ldots, n$ of $O_i(\mathcal{N})$.

One may then inquire whether or not all connected networks are SWS. Consider network $\mathcal{N}_C$ in Figure 2, along with the corresponding evolution of the network state vector when node 1 is activated at step 0. We observe that SWS does not occur in this case:

Network state vector for $\mathcal{N}_C$

$$\begin{bmatrix} 1\\0\\0\\0\\0\\0\\0\\0\\0 \end{bmatrix} \rightarrow \begin{bmatrix} 0\\0\\1\\0\\0\\0\\0\\0\\0 \end{bmatrix} \rightarrow \begin{bmatrix} 1\\1\\0\\1\\0\\0\\0\\0\\0 \end{bmatrix} \rightarrow \begin{bmatrix} 0\\0\\1\\0\\1\\0\\1\\0\\0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1\\1\\0\\1\\0\\1\\0\\1\\0 \end{bmatrix} \rightarrow \begin{bmatrix} 0\\0\\1\\0\\1\\0\\1\\0\\1 \end{bmatrix} \rightarrow \cdots \rightarrow \begin{bmatrix} 1\\1\\0\\1\\0\\1\\0\\1\\0 \end{bmatrix} \rightarrow \begin{bmatrix} 0\\0\\1\\0\\1\\0\\1\\0\\1 \end{bmatrix} \rightarrow \cdots$$

This time, the network state vector eventually begins to alternate indefinitely between two complementary states, so that SWS never occurs. Instead, the network is partitioned into two subgroups which are activated at alternating time steps. Hence we may classify a network as a SWS network if SWS occurs after finitely many steps, and as a Subgroup Alternating (SGA) network if the above phenomenon occurs.

3. SYSTEM-WIDE SYNCHRONIZATION OF DETERMINISTIC NETWORKS

We now consider whether there is some geometric feature of a network that determines whether or not SWS occurs. To address this question, let us consider the following two lemmas, whose proofs are supplied in the appendix. Here, a node is periodic with period 2 if the node is activated at even-numbered time steps and deactivated at odd-numbered time steps, or vice versa, while a node is periodic with period 1 if it is activated at every time step. Also, $D_i(\mathcal{N})$ denotes the diameter of network $\mathcal{N}$ with respect to node i.

Lemma 1. *Every activated node in a connected network is periodic with period $p \leq 2$.*

Lemma 2. *If an activated node i in a connected network $\mathcal{N}$ becomes periodic with period $p = 1$ at some step k, then SWS will occur within $D_i(\mathcal{N})$ additional steps.*

Together, these lemmas infer that the key to the SWS of a connected network is that one of the nodes must become periodic with period 1 at some step in the signal transmission process. Note that this occurred in the evolution of the network state vector for $\mathcal{N}_B$ at step 1, since nodes 2 and 3 remained activated in going from the first to the second step. The geometric feature which allowed this to happen is the presence in the structure of the network of a loop consisting of an odd number of edges. We will refer to such a loop, in which we have a closed path consisting of an odd number of edges, as an *odd-length loop*.

Note that in network $\mathcal{N}_B$ there are four such loops of length 3, while in network $\mathcal{N}_A$ there are numerous such loops of lengths 3, 5, 7 or 9. But there are no odd-length loops in network $\mathcal{N}_C$. This leads to the following theorem and corollary, whose proofs are in the appendix:

Theorem 1. *System-wide synchronization occurs in a connected network if and only if its structure includes a loop consisting of an odd number of edges.*

Now, suppose we have a SWS network. We can then determine an upper bound on the order of the network, based on its diameter:

Corollary 1. *A SWS network $\mathcal{N}$ having n nodes has order $O(\mathcal{N}) \leq 2D(\mathcal{N})$, regardless of the node at which the signal originates.*

Hence the most efficient SWS network, in terms of having the smallest order, is one with the smallest diameter possible. For example, if $n - 1$ nodes are each connected by a single edge to one hub node, the network will have a diameter of 2. Thus if this same network also includes a loop of length 3, it will be a SWS network of order at most 4. Network $\mathcal{N}_A$ has a diameter of 1, since each node is a neighbor to every other node. Thus $\mathcal{N}_A$ is a SWS network of order 2.

4. SUBGROUP ALTERNATION IN DETERMINISTIC NETWORKS

Corollary 1 together with Lemmas 1 and 2 imply that if a connected network $\mathcal{N}$ is not SWS, then after at most $2D(\mathcal{N})$

steps every node will be periodic with period $p = 2$. Moreover, the network will itself have a period of 2 no later than step $2D(\mathcal{N})$, with one subset of nodes activated simultaneously at only the odd-numbered steps, and the remaining subset of nodes activated concurrently at only the even-numbered steps, as noted in the example of network $\mathcal{N}_C$. Of course, which subset corresponds to the odd-numbered steps depends on the choice of initial node. Hence if a network is not SWS, it will begin subgroup alternation (SGA) after finitely many steps.

We may then identify a feature of the matrix $M(\mathcal{N}) = M$ corresponding to a connected network $\mathcal{N}$ that establishes the network as SGA. This feature is the ability to partition M into two sub-matrices M_1 and M_2 such that M_1 consists of $m \geq 1$ columns from M and M_2 consists of the remaining $n - m \geq 1$ columns, and such that each sub-matrix consists of one or more rows of zeros, but in complementary rows.

As an example, consider the matrix corresponding to SGA network $\mathcal{N}_C$:

$$M(\mathcal{N}_C) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

We will take columns 1, 2, 4, 6 and 8 for M_1 and the remaining columns for M_2:

$$M_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Note that M_1 has rows of zeros in rows 1, 2, 4, 6 and 8, which correspond to one of the two subsets of nodes which are activated concurrently once the network achieves SGA. Meanwhile M_2 has rows of zeros only in the other four rows, i.e., in rows 3, 5, 7 and 9, corresponding to the other subset of nodes.

Given this condition, we may simply relabel the nodes of $\mathcal{N}_C$ in such a way that $M(\mathcal{N}_C)$ takes on a block off-diagonal form. This is done by consecutively labeling the nodes corresponding to one of the two subsets, and then labeling the remaining nodes corresponding to the other subset. For instance, if in $\mathcal{N}_C$ we relabel node 3 as node 6, node 4 as node 3, node 5 as node 7, node 6 as node 4, node 7 as node 8, and node 8 as node 5, the matrix for network $\mathcal{N}_C$ becomes

$$M(\mathcal{N}_C) = \left[\begin{array}{ccccc|cccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

The lower-left block is matrix M_1 with its zero-rows removed, while the upper-right block is matrix M_2 with its zero-rows removed. If we call the lower-left block A, then the upper-right block is its transpose A'. In general, the matrix of every SGA network can be represented in this manner. This may be stated in a theorem:

Theorem 2. *A connected network $\mathcal{N}$ consisting of n nodes achieves subgroup alternation if and only if there is some $m \in \{1, \ldots, n-1\}$ and some permutation of the node labels $\{1, \ldots, n\}$ such that its matrix $M = M(\mathcal{N})$ can be partitioned into a block off-diagonal matrix of the form*

$$M = \begin{bmatrix} 0 & A' \\ A & 0 \end{bmatrix},$$

where A is an $(n-m) \times m$ matrix.

The proof is supplied in the appendix.

Now suppose M takes this block off-diagonal form, and suppose λ is an eigenvalue of M with corresponding eigenvector $\mathbf{v} = (v_1, \ldots, v_m, v_{m+1}, \ldots, v_n)'$. Since M is symmetric, λ is real. Then the block off-diagonal form of M and the relation $M\mathbf{v} = \lambda\mathbf{v}$ implies $A'(v_{m+1}, \ldots, v_n)' = \lambda(v_1, \ldots, v_m)'$ and $A(v_1, \ldots, v_m) = \lambda(v_{m+1}, \ldots, v_n)'$. If $\lambda \neq 0$, then $-\lambda$ must also be an eigenvector of M, with corresponding eigenvector $\mathbf{v} = (-v_1, \ldots, -v_m, v_{m+1}, \ldots, v_n)'$, since the relations

$$A'(v_{m+1}, \ldots, v_n)' = -\lambda(-v_1, \ldots, -v_m)' = \lambda(v_1, \ldots, v_m)'$$

and

$$A(-v_1, \ldots, -v_m) = -A(v_1, \ldots, v_m)' = -\lambda(v_{m+1}, \ldots, v_n)'$$

also hold. Hence, if n is even, the set of eigenvalues of M must be representable as $\{\pm\lambda_1, \ldots, \pm\lambda_{n/2}\}$. If n is odd, the requirement that nonzero eigenvalues appear in positive/negative pairs requires that at least one eigenvalue must be zero, so that the sum of the eigenvalues equals the trace of M, which is zero. Thus, if n is odd, the set of eigenvalues of M must be representable as $\{0, \pm\lambda_1, \ldots, \pm\lambda_{(n-1)/2}\}$. In general, the matrix corresponding to a SGA network must have symmetry of its eigenvalues about zero. Therefore, if a network $\mathcal{N}$ is SWS, the eigenvalues corresponding to its

matrix will *not* be symmetric about zero, i.e., there will be at least one nonzero eigenvalue whose additive inverse is not an eigenvalue.

Hence we have obtained both a geometric criterion (odd-length loops) and an algebraic criterion (asymmetry of eigenvalues) which may be used to determine whether or not a deterministic network is SWS.

5. OPTIMIZATION OF RANDOM NETWORKS UNDER BUDGET CONSTRAINTS

Armed with these criteria, we turn our attention to a random network consisting of n nodes with $N = n(n-1)/2$ distinct potential edges. In a sequence of independent trials, each edge may or may not occur, so that any particular realization of the network may or may not be SWS (or even connected). Moreover, even when a SWS network is realized, it may not be relatively efficient, i.e., the average number of steps required to achieve SWS over the n nodes may be relatively large. Suppose that the probability that edge $e_{i,j}$ occurs between nodes i and j in any trial is some fixed number $p_{i,j} \in [0,1]$. We may then form a vector $\mathbf{p}$ of length N consisting of these probabilities, such that

$$\mathbf{p} = (p_{1,2}, p_{1,3}, \ldots, p_{1,n}, p_{2,3}, \ldots, p_{2,n}, \ldots, p_{n-2,n-1}, p_{n-2,n}, p_{n-1,n}) .$$

Suppose further that there exists some fixed budget constraint $B \leq N$ such that

$$B = \sum_{1 \leq i < j \leq n} p_{i,j} .$$

We then ask the question: Are there optimal allocations of probabilities among the N components of $\mathbf{p}$ that conform to the budget constraint and that maximize the probability that a relatively efficient SWS network will be realized in any trial?

Certainly, if B is large enough (at least n), we may assign a probability of 1 to each of n edges chosen such that, when present, the resulting network is SWS and as efficient as possible. Thus we are interested in situations where B is relatively small, so that we cannot assign high probabilities very liberally. In nature, systems are configured so as to allocate limited resources in an optimal manner. By requiring a low budget, we seek to model this tendency.

One exhaustive method would require an examination of each of the 2^N possible networks to identify which of these are SWS and have a desired level of efficiency. We would then need to search for those vectors in $[0,1]^N$ that conform to the budget constraint while maximizing the probability of producing one of these identified networks in any trial. This is clearly impractical.

As an alternative, we first substitute $[0,1]^N$ with the lattice $\{0.0, 0.1, \ldots, 0.9, 1.0\}^N \subset [0,1]^N$, and require B to be a positive multiple of 0.1. If we find optimal vectors in the lattice, it may be assumed that the optimal vectors in $[0,1]^N$ lie nearby. Thus we now consider the space consisting of only those vectors $\mathbf{p}$ whose components $p_{i,j}$ lie in the set $\{0.0, 0.1, 0.2, \ldots, 0.9, 1.0\}$ and sum to B. Nevertheless, an exhaustive exploration of this search space remains intractable. Hence we turn to the simulated annealing (SA) search algorithm, as developed independently by Kirkpatrick et al. [8], and by Černý [3].

In the SA algorithm, we start at some initial vector in the search space. Then we select a neighboring vector, which we define to be a vector whose components match those of the initial vector in all but two of the N positions, and which differ from the initial vector in those two positions by ± 0.1. For instance, if $\mathbf{p} = (0.2, 0.1, 0.3, 0.4, 0.0, 0.8)$, the vector $(0.2, 0.1, 0.2, 0.4, 0.1, 0.8)$ would be a neighbor. If the neighboring vector is more likely to produce an efficient SWS network, we move to that vector. Otherwise, we may still move to it with a certain probability which gradually diminishes from one to zero as the algorithm progresses. Then, at the next iteration, we select a neighbor and repeat.

To implement the SA algorithm, we require a positive real-valued *energy function* $E(\mathbf{p})$ that estimates the "energy" of each point $\mathbf{p}$ in the search space. Here the energy of $\mathbf{p}$ represents its likelihood to generate networks which are *not* SWS, and networks which are SWS but relatively inefficient. Hence the energy function is constructed such that it decreases toward zero as we encounter optimal solutions.

Our energy function consists of I iterations. At each iteration the function uses the probabilities $p_{i,j}$ to create a realization of the random network. The function then determines whether the realized network is SWS. If the network is SWS, the energy function determines the geometric mean order of the network starting at each of the n nodes, as a measure of its efficiency. Upon completing I iterations, the energy function computes the proportion φ of the I realized networks which were *not* SWS, and, among those realized networks which were SWS, the average ψ of the geometric mean order. Note that we intend to minimize both φ and ψ. We then form a weighted average $W = a\varphi + b\psi$, with a and b chosen such that φ dominates until it becomes very small, at which point ψ begins to have greater influence. To control the rate at which the energy function decreases as W decreases, we input W into the sine function, ensuring that $0 \leq W \leq \pi/2$. In our implementation, we use

$$E(\mathbf{p}) = \sin(0.5\varphi + 0.001\psi).$$

As anyone would surmise, this energy function is highly variable, no matter what value is chosen for I (we used $I = 1,000$), unless $\mathbf{p}$ consists almost exclusively of ones and zeros. To reduce the variability, we take the average function value over 10 applications to any vector $\mathbf{p}$. Nevertheless, the variability remains. While this does not prove to be a

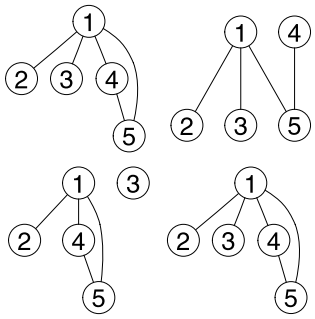

Figure 3. Four realizations of an optimized random five-node network.

critical problem when n is small (say, 4 or 5), it has a deleterious effect on the success of the algorithm when $n = 10$ or $n = 15$.

Throughout the SA algorithm, we maintain a record of the best vector we have encountered, i.e., the vector whose computed energy is lowest. Once the algorithm terminates, the best vector is declared to be optimal in the sense of having the lowest energy in the search space, and thus the greatest likelihood of producing efficient SWS networks among those having the same budget constraint. In practice, the vector identified by the algorithm may not be optimal, but it is usually quite good. Moreover, it is not unique, since permuting the labels of the nodes would change the labels of the potential edges among them, producing a different vector of probabilities.

The SA algorithm also requires a temperature function which decreases monotonically from one to zero as the algorithm progresses from its first iteration to its last. We use the function $T(z) = 1 - e^{5(z-1)}$, where z is the proportion of iterations completed. The algorithm relies on the temperature at any iteration to determine the probability of moving from a state whose energy is lower to a neighboring state whose energy is higher, as mentioned above. This probability decreases as the temperature decreases, so that the algorithm gradually narrows its focus to one convex region of the search space.

For example, consider a random network consisting of five nodes and thus 10 potential edges. Suppose $B = 4$, which is low enough to ensure that a SWS network cannot be guaranteed in any realization. If we start with an initial vector consisting of a probability of 0.4 for each potential edge (with corresponding energy 0.3536), the SA algorithm returns as the optimal result $(0.8, 0.8, 0.8, 0.8, 0.0, 0.0, 0.0, 0.0, 0.0, 0.8)$, whose energy is found to be 0.3301. Four realizations of the random network with this optimal allocation of probabilities to the 10 potential edges are displayed in Figure 3. We note that only three of the networks are connected, and of these only two are SWS (having a loop of length 3). The two SWS networks are identical, each of diameter 2. Given the low budget, this is the best allocation of probabilities we can find.

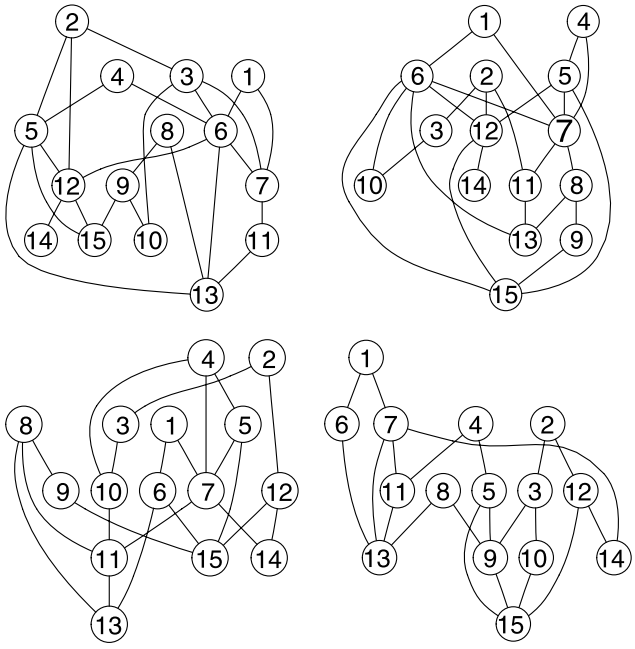

Figure 4. Four realizations of an optimized random fifteen-node network.

As a second example, consider a random network consisting of 15 nodes and 105 potential edges, with a relatively high budget of $B = 21$. If we start with an initial vector consisting of a probability of 0.2 for each potential edge (with corresponding energy 0.2510), the SA algorithm returns an optimal result whose energy is computed to be 0.0077, which is remarkably low. Four realizations of this random network are displayed in Figure 4. Note that all four networks are SWS, as loops of length 3 may be easily identified among each set of edges, and odd-length loops of higher dimension are also evident.

Hence the SA algorithm proves quite effective in finding an allocation of probabilities among potential edges that maximizes the likelihood that any realization of a random network will be SWS.

6. ROBUST NETWORKS

A desirable feature of a SWS network is that it be *robust*. One way of defining robustness is in terms of the loss of wirings between nodes. An alternative definition of robustness involves the preservation of a SWS network if one node "malfunctions." We focus on networks which are robust under the latter definition, i.e., robust against the loss of a node. Observe that network $\mathcal{N}_A$ is robust against loss of a node, whereas if nodes 3, 4 or 7 were to be removed from network $\mathcal{N}_B$, the remaining sub-network would not even be connected.

We may modify the energy function $E(\mathbf{p})$ implemented in the SA algorithm so that we also check each SWS network

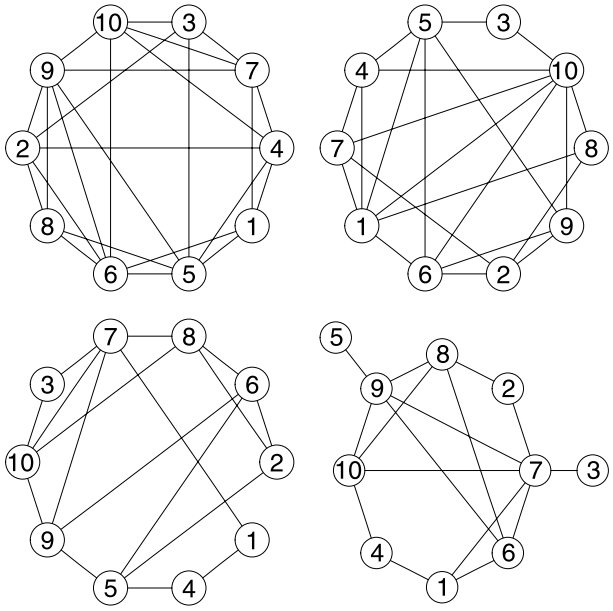

Figure 5. Four realizations of a robust optimized random ten-node network.

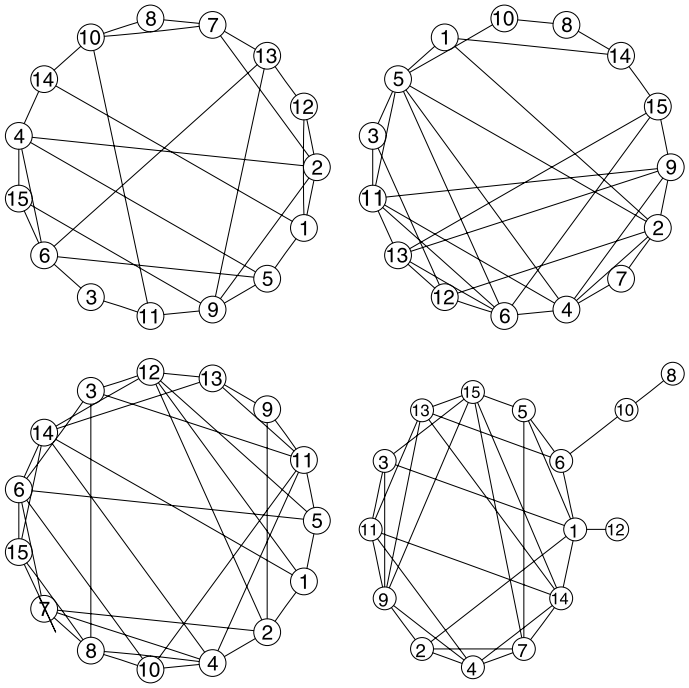

Figure 6. Four realizations of a robust optimized random fifteen-node network.

$\mathcal{N}$ among the I realized networks for robustness. We achieve this by successively removing the ith row and column from $M(\mathcal{N})$, $i = 1, \ldots, n$, and determining whether the resulting submatrix corresponds to a SWS network. This obviously slows down the algorithm considerably. Let χ denote the proportion of SWS networks which are *not* robust among I realized networks. This time we form the weighted average $W = a\phi + b\psi + c\chi$, with the constants a, b and c chosen to balance the influences of system-wide synchronization, robustness, and efficiency as the algorithm progresses. In our implementation, we use

$$E(\mathbf{p}) = \sin(0.5\phi + 0.001\psi + 0.1\chi).$$

To make the problem more realistic, and hence more interesting, we further restrict the probabilities $p_{i,j}$ comprising the probability vector $\mathbf{p}$ to the set $\{0.0, 0.1, \ldots, 0.9\}$, so that no edge in the network can ever be guaranteed. This reduces the search space slightly.

We implement the SA algorithm with these modifications, using a network of 10 nodes and a budget of $B = 22.5$, beginning with an equidistribution of $p_{i,j} = 0.5$ among the 45 potential edges, yielding an energy of 0.100. The optimal probability vector returned by the procedure has a computed energy of 0.046, which is quite good given the additional restrictions. Then we repeat using a 15-node network with a budget of 31.5, starting with an equidistribution of $p_{i,j} = 0.3$ among the 105 potential edges, and an energy of 0.292. The SA algorithm returns an optimal result whose energy is 0.180. Four realizations of random networks based on this optimal result are displayed in Figures 5 and 6.

In Figure 5, the lower-right network is SWS, but not robust, since the loss of nodes 7 or 9 would render the remaining network unconnected. But the other three networks are both SWS and robust. Coincidentally, the lower-right network in Figure 6 is also SWS, but not robust, while the other three are both. We observe that in the robust networks, odd-length loops (particularly, triangles) occur at multiple distinct locations in the network. Hence optimal robust configurations of random networks are those which connect smaller subnetworks which are themselves SWS networks. This of course requires a sufficient budget to permit such a configuration.

7. CONCLUSION

We have defined system-wide synchronization in networks based on the premise that, once a signal leaves a node, the node becomes inactive, and remains so until the signal returns to it, as is known of neurons. Working with this definition, we discover that a system which can be modeled by such a network either achieves a state of system-wide synchronization after an initial start-up period, or wavers indefinitely between two complementary states. This outcome ultimately depends only on the geometry of the network. Hence the functionality of such a system would require attention to whether or not its structure contains loops with an odd number of edges.

In a system where the connectivity among nodes is random, perhaps dependent on the correlations between them,

we may further determine an optimal allocation of probabilities among the potential edges, subject to certain budget constraints, such that the functionality of the system is most likely. In application to such a system which is not functioning optimally, one might determine how to reallocate resources in order to improve performance. Moreover, to promote robustness of such a system, one should ensure that its subsystems are independently designed to function optimally.

The results obtained in this paper may be summarized as follows:

S0 [Deterministic network:] A SWS phase can only be achieved in a connected network containing at least one substructure consisting of a closed loop comprised of an odd-number of edges.

S1 [Random network:] Any wiring probability allocation scheme achieves a high potential of producing an efficient SWS phase when it frequently gives rise to several heavily-connected hubs.

S2 [Random network with robustness:] The robustness constraint requires a higher budget, and at the same time replaces the potential hubs with many well-scattered potential triangular substructures.

Result [S0] regarding deterministic networks affords a huge reduction in computations for Result [S1], which in turn partly anticipates Result [S2]. However the appearance of abundant triangular substructures in a robust random network is somewhat surprising.

At the end we postulate immediate implications of these results on the study of emotion and memory reactivation as follows:

I1: An emotion arousal can be effectively triggered when subgroups of emotion variables, such as behavioral, experiential and physiological variables, are well-wired, even though the subgroups themselves are sparsely connected with redundant wiring.

I2: Even in the absence of an inhibiting mechanism, the feed-forward and feed-back mechanisms of signal transmission are sufficient to efficiently and robustly generate SWS phases among a designated group of neurons with strong local connections and sparse global wiring.

APPENDIX A. PROOFS

Proof of Lemma 1. Assume node i is activated at step k. At step $k+1$, each neighbor of i is activated, while i may either be deactivated or reactivated by a neighbor. At step $k+2$, each neighbor of node i reactivates node i. This cycle then repeats indefinitely, so that node i is never deactivated for more than one consecutive step. Hence node i is periodic with period at most two. □

Proof of Lemma 2. Assume activated node i is periodic with period $p = 1$ at step k. At step $k+1$, every neighbor of node i is activated, and node i remains activated since node i has period 1. Now the neighbors of node i have period 1 since their neighbor, node i, is activated at every step. At step $k+2$, every neighbor of the neighbors of node i is activated, node i remains activated, and every neighbor of node i remains activated. Now the neighbors of the neighbors of node i have period 1. By step $k + D_i(\mathcal{N})$, this effect will have been transferred to the nodes which are at the greatest distance from node i, so that every node in $\mathcal{N}$ is activated. Hence all nodes of $\mathcal{N}$ are simultaneously activated within $D_i(\mathcal{N})$ additional steps. □

Proof of Theorem 1. It is evident from the proof of Lemma 2 that an activated node i becomes periodic with period 1 if and only if the signal reaches both i and some neighbor of i in the same step. This requires the presence of a closed path in the network structure comprised of an odd number of nodes, i.e., an odd-length loop. Then, and only then, the signal will follow two branches which reach two neighboring nodes at the same step. By Lemma 2, such a network is SWS. □

Proof of Corollary. Assume $\mathcal{N}$ is a SWS network with n nodes. If node i is activated at step 0, the signal will propagate to every node of $\mathcal{N}$ within $D_i(\mathcal{N}) \leq D(\mathcal{N})$ steps. Since $\mathcal{N}$ is SWS, $\mathcal{N}$ must have at least one odd-length loop in its structure. Thus the signal must enter the loop and reach two neighboring nodes h and j at the same step while propagating throughout the network, so that h and j each become periodic with period 1. By Lemma 2, $\mathcal{N}$ will then achieve simultaneous activation of all nodes within $D_h(\mathcal{N}) \leq D(\mathcal{N})$ steps. Therefore SWS occurs within $D_i(\mathcal{N}) + D_h(\mathcal{N}) \leq 2D(\mathcal{N})$ steps. Hence $O_i(\mathcal{N}) \leq 2D(\mathcal{N})$. Since i is arbitrary, $O(\mathcal{N}) \leq 2D(\mathcal{N})$. □

Proof of Theorem 2. Assume $\mathcal{N}$ is a connected network which is SGA, with corresponding $n \times n$ matrix M. Then after finitely many steps every node of $\mathcal{N}$ is periodic with period 2. This implies that the network state vector $\mathbf{v} = (v_1, \ldots, v_n)$ eventually begins to alternate indefinitely between two states, call them α and β, in which each v_i alternates between one and zero for $i = 1, \ldots, n$ (at each step, $\mathbf{v}$ consists of at least one zero component and at least one nonzero component). Since each successive state results from the product $[M\mathbf{v}]$, the inner product of the ith row of M with $\mathbf{v}$ must be zero whenever $v_i = 1$ and be nonzero otherwise. Hence the ith row of M must have zeros in those columns corresponding to the positions of the nonzero components in $\mathbf{v}$ in one of the two alternating states, and a one in at least one of the remaining columns. We may thus let M_1 be the matrix consisting of the $m \geq 1$ columns of M which have zeros in the rows corresponding to all the positions of the nonzero components in $\mathbf{v}$ when it is in state α, and let M_2 be the matrix consisting of the remaining $n - m \geq 1$ columns of M which have zeros in the rows corresponding to all the positions of the nonzero components

in $\mathbf{v}$ when it is in state β. Then M_1 has one or more rows consisting only of zeros, while each corresponding row of M_2 contains at least one nonzero entry. Likewise, M_2 has one or more rows consisting only of zeros, while each corresponding row of M_1 contains at least one nonzero entry. By relabeling the nodes so that those which are activated in state α are consecutively numbered $1, \ldots, m$, and those which are activated in state β are consecutively numbered $m+1, \ldots n$, the matrix M will thus take the form

$$M = \begin{bmatrix} 0 & A' \\ A & 0 \end{bmatrix},$$

where A is an $(n-m) \times m$ matrix.

Moreover, if M may be partitioned in this way, then after finitely many steps every node of $\mathcal{N}$ must be periodic with period 2. Hence $\mathcal{N}$ is SGA. □

Received 1 October 2009

REFERENCES

[1] Boccara, N. (2004). *Modeling complex systems*, Springer-Verlag.

[2] Buzsáki, G. (2006). *Rhythms of the Brain*, Oxford University Press.

[3] Černý, V. (1985). A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45: 41–51.

[4] Fujisawa, S., Amarasingham, A., Harrison, M. and Buzsáki, G. (2008). Behavior-dependent short-term assembly dynamics in the medial prefontal cortex.

[5] Gurney, K. (1997). *An Introduction to Neural Networks*, London: Routledge.

[6] Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*, Prentice Hall.

[7] Hsieh, F., Ferrer, E., Chen, S., Mauss, I.B. and Gross, J.J. (2008). A small-world network approach for evaluating coherence in multivariate systems: an application to psycho-physiological emotion data. Under revision for *Psychometrika*.

[8] Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983). Optimization by Simulated Annealing. *Science*, New Series **220** (4598): 671–680.

[9] Kriesel, D. (2007). *A Brief Introduction to Neural Networks*, available online at http://www.dkriesel.com.

[10] Lawrence, J. (1994). *Introduction to Neural Networks*, California Scientific Software Press.

[11] Newman, M.E.J., Barabási, A.-L. and Watts, D.J. (2006). *The Structure and Dynamics of Networks*, Princeton University Press.

[12] Smith, M. (1993). *Neural Networks for Statistical Modeling*, Van Nostrand Reinhold.

[13] Watts, D.J. (1999). *Small Worlds: The Dynamics of Networks between Order and Randomness*. Princeton University Press.

Michael P. McAssey
Department of Statistics
University of California, Davis
USA
E-mail address: mmcassey@wald.ucdavis.edu

Fushing Hsieh
Department of Statistics
University of California, Davis
USA
E-mail address: fushing@wald.ucdavis.edu

Emilio Ferrer
Department of Psychology
University of California, Davis
USA
E-mail address: eferrer@ucdavis.edu