ARTICLE Communicated by Carlos Brody

How to Compress Sequential Memory Patterns into Periodic Oscillations: General Reduction Rules

Kechen Zhang
kzhang4@jhmi.edu
Department of Biomedical Engineering, Johns Hopkins University School of Medicine, Baltimore, MD 21205, U.S.A.

A neural network with symmetric reciprocal connections always admits a Lyapunov function, whose minima correspond to the memory states stored in the network. Networks with suitable asymmetric connections can store and retrieve a sequence of memory patterns, but the dynamics of these networks cannot be characterized as readily as that of the symmetric networks due to the lack of established general methods. Here, a reduction method is developed for a class of asymmetric attractor networks that store sequences of activity patterns as associative memories, as in a Hopfield network. The method projects the original activity pattern of the network to a low-dimensional space such that sequential memory retrievals in the original network correspond to periodic oscillations in the reduced system. The reduced system is self-contained and provides quantitative information about the stability and speed of sequential memory retrievals in the original network. The time evolution of the overlaps between the network state and the stored memory patterns can also be determined from extended reduced systems. The reduction procedure can be summarized by a few reduction rules, which are applied to several network models, including coupled networks and networks with time-delayed connections, and the analytical solutions of the reduced systems are confirmed by numerical simulations of the original networks. Finally, a local learning rule that provides an approximation to the connection weights involving the pseudoinverse is also presented.

1 Introduction

Symmetric connections in a recurrent network allow random memory patterns to be stored as stable states or point attractors (Hopfield, 1982; Cohen & Grossberg, 1983). When proper asymmetric connections are added, these memory patterns are no longer stationary but can be recovered sequentially as the network state evolves spontaneously. This article focuses on the general scheme of temporal association in which both the symmetric and the asymmetric connections are given by the outer product rule like that in a Hopfield network. These associative memory networks enjoy properties

Neural Computation **26, 1542–1599** (2014)
doi:10.1162/NECO_a_00618

such as biological plausibility, robustness again noise and damage, and capability for pattern completion or recall from partial cues (see Amit, 1989, for a review).

It is not trivial for an asymmetric network to faithfully follow a long temporal sequence because interferences between consecutive memory patterns tend to make individual units switch their states at different speeds and thus destabilize the sequential retrieval. To stabilize the dynamics, Sompolinsky and Kanter (1986) and Kleinfeld (1986) introduced time delays to the asymmetric connections while keeping the symmetric connections constant. Various methods for embedding temporal sequences in Hopfield-type networks have been studied, as reviewed by Kühn and van Hemmen (1995). Unlike in a symmetric network, where the existence of a Lyapunov function guarantees the approach to a stationary state (Hopfield, 1982, 1984; Cohen & Grossberg, 1983), the dynamical stability of an asymmetric network cannot always be characterized by a Lyapunov function, which may not even exist except for special cases (Kleinfeld & Sompolinsky, 1988; Herz, Li, & van Hemmen, 1991).

The analysis in this article is based on the idea of a reduced oscillatory variable, as illustrated in Figure 1. Let $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^M$ denote M memory patterns stored in a network. Suppose the instantaneous state $\mathbf{x}(t)$ of the network goes through these memory patterns one after another as follows: $\mathbf{x}^1 \to \mathbf{x}^2 \to \cdots \to \mathbf{x}^M$. For simplicity, here we assume that each memory pattern $\mathbf{x}^m$ is a random vector with N entries that are equal to either 1 or -1 (on state or off state) with equal probability. Different memory patterns are drawn independently so that they are approximately orthogonal to one another. We define the overlap between the network state $\mathbf{x}(t)$ and the memory pattern $\mathbf{x}^m$ as the dot product $\frac{1}{N}\mathbf{x}^m \cdot \mathbf{x}(t)$, normalized by the number of units N in the network. As shown in Figure 1, we can construct a new scalar variable $x(t)$ by flipping the signs of all the memory overlaps alternately according to the order in the temporal sequence and then adding up the results. The overall effect of this procedure is to transform the vector variable $\mathbf{x}(t)$ into a new oscillatory variable defined by

$$\begin{aligned} x(t) &= -\frac{1}{N}\mathbf{x}^1 \cdot \mathbf{x}(t) + \frac{1}{N}\mathbf{x}^2 \cdot \mathbf{x}(t) - \frac{1}{N}\mathbf{x}^3 \cdot \mathbf{x}(t) + \frac{1}{N}\mathbf{x}^4 \cdot \mathbf{x}(t) \cdots \\ &= \mathbf{c} \cdot \mathbf{x}(t), \end{aligned} \tag{1.1}$$

where $\mathbf{c}$ is a fixed vector given by

$$\mathbf{c} = \frac{1}{N}\sum_{m=1}^{M} (-1)^m \mathbf{x}^m. \tag{1.2}$$

As illustrated in Figure 1, the period of the oscillation is equal to twice the memory duration T, which is the time per memory pattern in the sequence. We show later in this article that the dynamical equation of an asymmetric

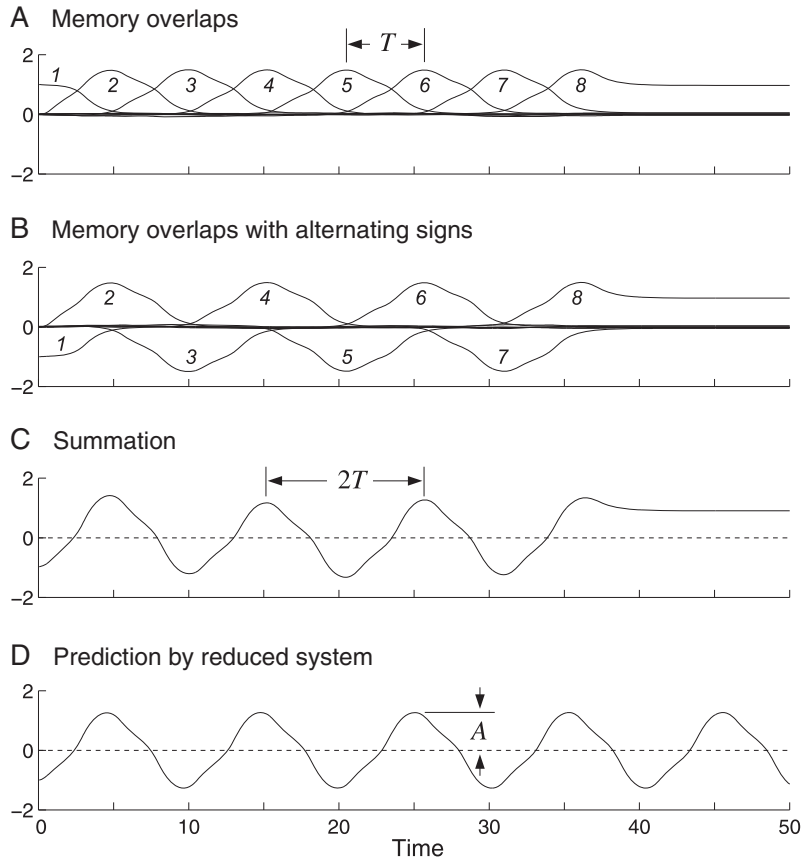

Figure 1: A simple example that shows how to compress a sequence of memory states into a periodic oscillation. (A) First compute the overlaps (dot products in this example) between the instantaneous network state and the stored memory patterns (from 1 to 8) during spontaneous sequential retrievals. (B) Flip the signs of the overlaps alternately. (C) Add up the curves in panel B to obtain a single oscillatory variable whose period is twice the memory duration T. Data were obtained numerically from a single subnet of a conjugate network (each subnet having $N = 5000$ units and $M = 8$ memory patterns with equal on and off probability, $g(x) = \tanh(2x)$, $\lambda = 0.7$, and $\tau = 1$). (D) Periodic oscillation predicted by the reduced system closely matches that in panel C.

Hopfield-type network can often be reduced to an equation that involves the reduced variable only. The reduced system is much simpler than the original network and, more important, is self-contained and can be studied on its own. Figure 1D shows the oscillatory behavior of the reduced system,

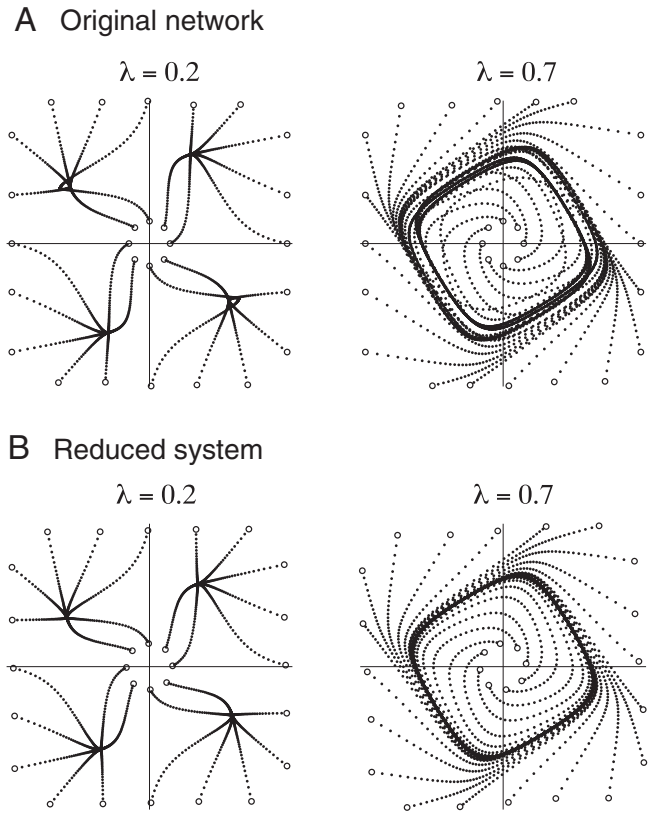

Figure 2: Similarity between the behaviors of the original network and those of its reduced system. (A) Sample trajectories traced by a pair of reduced variables, each obtained by compressing a subnet of a conjugate network (the same network as in Figure 1). The initial states (open circles) were set to be proportional to the first memory patterns in the two subnets. The network settled into one of four stationary states when the asymmetric coupling strength was low ($\lambda = 0.2$). When the coupling strength was stronger ($\lambda = 0.7$), the network retrieved a sequence of memory patterns. (B) Phase plane of the corresponding reduced system (equations 3.15 and 3.16) has four point attractors when $\lambda = 0.2$ and a limit cycle when $\lambda = 0.7$. Plot range from -2 to 2 for both axes.

whose period can be used to predict the memory duration T of the original network. Figure 1 also shows that the original system need not be periodic, even though its reduced system oscillates periodically. What the reduced system reflects is the stereotyped transient dynamics during the transition between consecutive memory patterns (see also Figure 2).

The reduction procedure for the dynamical equations of an asymmetric network can be unified by a few reduction rules. Once these rules for typical terms are derived, we can apply them directly to a dynamical equation of any given Hopfield-type network, and the outcome is a reduced dynamical equation in terms of the reduced variables only. The basic idea of the reduction method was first proposed for the special case of orthogonal memory patterns (Zhang, 1994). Here we consider the general case with arbitrary binary memory patterns that have arbitrary probability for a unit to be on. We will show that the connection weights based on pseudoinverse of the memory patterns have statistical regularities that lead to a universal reduced system regardless of the level of sparsity of the memory patterns. We will analyze several reduced systems and confirm their predictions by simulations on the original networks. Finally, the reduction method will be extended to include additional reduced equations that can fully determine the time profiles of the memory overlaps.

2 Asymmetric Networks for Storage of Sequential Memory Patterns

Before explaining the reduction rules, we first need to examine the dynamical equations of some representative neural networks that can store sequential memory patterns. We consider two types of network models modified from the Hopfield network: one based on time-delayed asymmetric connections and the other on coupled subnetworks.

We start with a Hopfield network with N units and describe the stored M memory patterns by a matrix:

$$\mathbf{X} = \left[\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^M\right], \tag{2.1}$$

with $M < N$. Here the mth memory pattern is a column vector $\mathbf{x}^m = [x_1^m, \ldots, x_N^m]^{\mathrm{T}}$, where each entry x_i^m is equal to either 1 (on state) or -1 (off state) and T indicates transpose. We assume that all these memory patterns are linearly independent, an assumption that should be valid almost certainly if all entries in matrix $\mathbf{X}$ are drawn randomly and independently of one another. The Moore-Penrose generalized inverse of $\mathbf{X}$ is given by

$$\mathbf{X}^\dagger = \begin{bmatrix} \mathbf{x}_1^\dagger \\ \vdots \\ \mathbf{x}_M^\dagger \end{bmatrix} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}, \tag{2.2}$$

where row vector $\mathbf{x}_m^\dagger$ is the mth row of matrix $\mathbf{X}^\dagger$. We have the orthogonality condition,

$$\mathbf{X}^\dagger\mathbf{X} = \mathbf{I}, \quad \text{or} \quad \mathbf{x}_m^\dagger\mathbf{x}^n = \delta_{mn}, \tag{2.3}$$

where $\mathbf{I}$ is the $M \times M$ identity matrix and δ_{mn} is the Kronecker delta. Consider the weight matrix given by

$$\mathbf{W} = \mathbf{X}\mathbf{X}^{\dagger} = \sum_{m=1}^{M} \mathbf{x}^m \mathbf{x}_m^{\dagger}, \tag{2.4}$$

and write the dynamical equation of the Hopfield network without external input as

$$\tau \frac{d\mathbf{x}}{dt} = -\mathbf{x} + \mathbf{X}\mathbf{X}^{\dagger} g(\mathbf{x}), \tag{2.5}$$

where τ is a time constant, $\mathbf{x} = [x_1, \ldots, x_N]^{\mathrm{T}}$ is the state of the network, and g is a sigmoidal gain function for the input-output relation of individual units, with $g(\mathbf{x}) = [g(x_1), \ldots, g(x_N)]^{\mathrm{T}}$. The weight matrix defined by equation 2.4 is always symmetric: $\mathbf{W}^{\mathrm{T}} = \mathbf{W}$. If follows from equations 2.4 and 2.3 that $\mathbf{W}\mathbf{X} = \mathbf{X}$, or $\mathbf{W}\mathbf{x}^m = \mathbf{x}^m$. That is, each memory pattern is an eigenvector of the weight matrix $\mathbf{W}$ with eigenvalue 1.

In the special case where the random memory patterns have equal on and off probability, we can approximate the pseudoinverse in 2.2 by

$$\mathbf{X}^{\dagger} \approx \frac{1}{N}\mathbf{X}^{\mathrm{T}}, \tag{2.6}$$

because now $\frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X} \approx \mathbf{I}$. This is the formulation used in the original Hopfield network (Hopfield, 1982). The pseudoinverse weight matrix in equation 2.4 works for any probability distribution and is therefore more general (Kohonen & Ruohonen, 1973; Kohonen, 1977; Personnaz, Guyon, & Dreyfus, 1985; Kanter & Sompolinsky, 1987).

Adding asymmetric connections to a Hopfield network yields the dynamical system

$$\tau \frac{d\mathbf{x}}{dt} = -\mathbf{x} + \mathbf{X}\mathbf{X}^{\dagger} g(\mathbf{x}) + \lambda \tilde{\mathbf{X}}\mathbf{X}^{\dagger} g(\mathbf{x}), \tag{2.7}$$

where λ is the strength of the asymmetric coupling and

$$\tilde{\mathbf{X}} = \left[\mathbf{x}^2, \mathbf{x}^3, \ldots, \mathbf{x}^M, \mathbf{0}\right] \tag{2.8}$$

is a shifted version of the memory matrix $\mathbf{X}$, with the last column padded by zeros. The goal here is for the system to spontaneously generate the temporal sequence $\mathbf{x}^1 \to \mathbf{x}^2 \to \cdots \to \mathbf{x}^M$, and eventually the system will stay at the last state $\mathbf{x}_M$, as in Figure 1. Define the asymmetric weight

matrix by

$$\tilde{\mathbf{W}} = \tilde{\mathbf{X}}\mathbf{X}^{\dagger} = \sum_{m=1}^{M-1} \mathbf{x}^{m+1}\mathbf{x}_m^{\dagger}. \tag{2.9}$$

Then it follows from equation 2.3 that $\tilde{\mathbf{W}}\mathbf{X} = \tilde{\mathbf{X}}$ or $\tilde{\mathbf{W}}\mathbf{x}^m = \mathbf{x}^{m+1}$ for $m < M$. Thus, the weight matrix $\tilde{\mathbf{W}}$ serves to drive the state of the network toward memory pattern $\mathbf{x}^{m+1}$ when the current state is $\mathbf{x}^m$. To embed a cyclic sequence (adding $\mathbf{x}^M \to \mathbf{x}^1$), replace equation 2.8 by

$$\tilde{\mathbf{X}} = \left[\mathbf{x}^2, \mathbf{x}^3, \ldots, \mathbf{x}^M, \mathbf{x}^1\right]. \tag{2.10}$$

Whether the sequence is cyclic or not is a global property that does not affect the stability and speed of local state transitions in the middle of the sequence.

The system in equation 2.7 is known to be unstable for retrieving a long sequence. A standard remedy is to add time delays to the asymmetric connections (Sompolinsky & Kanter, 1986; Kleinfeld, 1986). The dynamical equation reads

$$\tau \frac{d\mathbf{x}(t)}{dt} = -\mathbf{x}(t) + \mathbf{X}\mathbf{X}^{\dagger} g\left(\mathbf{x}(t)\right) + \lambda \int_0^{\infty} D(t')\tilde{\mathbf{X}}\mathbf{X}^{\dagger} g(\mathbf{x}(t-t'))dt', \tag{2.11}$$

where density function D describes the probability distribution of the time delays.

Another type of network considered in this article is based on coupled subnetworks or subnets (Zhang, 1994). The simplest case is the conjugate network with two subnets, described by the dynamical equations

$$\tau \frac{d\mathbf{x}}{dt} = -\mathbf{x} + \mathbf{X}\mathbf{X}^{\dagger} g(\mathbf{x}) + \lambda\tilde{\mathbf{X}}\mathbf{Y}^{\dagger} g(\mathbf{y}), \tag{2.12}$$

$$\tau \frac{d\mathbf{y}}{dt} = -\mathbf{y} + \mathbf{Y}\mathbf{Y}^{\dagger} g(\mathbf{y}) + \lambda\mathbf{Y}\mathbf{X}^{\dagger} g(\mathbf{x}). \tag{2.13}$$

Each subnet is a Hopfield network with symmetric connections. The first subnet stores the memory matrix $\mathbf{X} = \left[\mathbf{x}^1, \ldots, \mathbf{x}^M\right]$, while the second subnet stores the memory matrix $\mathbf{Y} = \left[\mathbf{y}^1, \ldots, \mathbf{y}^M\right]$. The asymmetric connections are now placed between the two subnets, which are allowed to have different sizes. Suppose the initial state of the first subnet is in memory $\mathbf{x}^1$; then the stored memory patterns should occur spontaneously and alternately in the two subnets in the following order: $\mathbf{x}^1 \to \mathbf{y}^1 \to \mathbf{x}^2 \to \mathbf{y}^2 \to \cdots \to \mathbf{x}^M \to \mathbf{y}^M$. Defining the state vector $\mathbf{z}$ of the whole network by concatenating the subnet state vectors $\mathbf{x}$ and $\mathbf{y}$, we can rewrite equations 2.12 and 2.13 as

$$\tau \frac{d\mathbf{z}}{dt} = -\mathbf{z} + \mathbf{W}g(\mathbf{z}), \tag{2.14}$$

where $\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}$, $g(\mathbf{z}) = \begin{bmatrix} g(\mathbf{x}) \\ g(\mathbf{y}) \end{bmatrix}$, and the block matrix

$$\mathbf{W} = \begin{bmatrix} \mathbf{X}\mathbf{X}^\dagger & \lambda\tilde{\mathbf{X}}\mathbf{Y}^\dagger \\ \lambda\mathbf{Y}\mathbf{X}^\dagger & \mathbf{Y}\mathbf{Y}^\dagger \end{bmatrix} \tag{2.15}$$

is the weight matrix for the entire network. Parameter λ scales the asymmetric weight components and will be referred to as the asymmetric coupling strength for various models in this article. The process of alternating memory retrievals in the conjugate network might seem to resemble the bidirectional memory model (Kosko, 1988), but they are actually very different: the bidirectional memory works for any matrix together with its transpose, whereas in equation 2.15, the two off-diagonal blocks are generally not related by a transposition.

3 Reduction Rules

3.1 Basic Idea of the Reduction Rules. Following the example in Figure 1, we start with the state $\mathbf{x}$ of a neural network and define the reduced scalar variable x by

$$x(t) = \mathbf{c}^{\mathrm{T}}\mathbf{x}(t). \tag{3.1}$$

The fixed vector $\mathbf{c}$ will be called the compressor vector, and its general definition is

$$\mathbf{c}^{\mathrm{T}} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^\dagger = \sum_{m=1}^{M} (-1)^m \mathbf{x}_m^\dagger, \tag{3.2}$$

where vector

$$\boldsymbol{\omega}^{\mathrm{T}} = [-1, 1, -1, 1, \ldots] \tag{3.3}$$

has M entries with alternating signs and $\mathbf{X}^\dagger$ is the pseudoinverse of the memory matrix $\mathbf{X}$, as in equation 2.2. In the special case of independent memory patterns with equal on and off probability, the pseudoinverse can be approximated by equation 2.6, and the compressor vector in equation 3.2 is reduced to equation 1.2, as used in the example in Figure 1.

The idea is to left-multiply $\mathbf{c}^{\mathrm{T}}$ on both sides of the dynamical equation of a neural network and then express each term as a function of the reduced variable x. The correspondence between the original term as a function of $\mathbf{x}$ and the reduced term as a function of x will be called a reduction rule.

Table 1: Basic Reduction Rules.

Original Term	Reduced Term	Remark (equation number)
Rule 1: Network or subnet state $\mathbf{x}$	x	Definition (3.1)
Rule 2: Time derivative of state $\frac{d\mathbf{x}}{dt}$	$\frac{dx}{dt}$	Derivative rule (3.5)
Rule 3: Symmetric drive $\mathbf{X}\mathbf{X}^{\dagger}g(\mathbf{x})$	$f(x)$	Within a subnet or network (3.40)
Rule 4: Asymmetric drive $\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x})$	$-f(x)$	Within a subnet or network for shifted sequence (3.44)
Rule 5: Intersubnet drive $\mathbf{Y}\mathbf{X}^{\dagger}g(\mathbf{x})$	$f(x)$	Between subnets (3.45)
Rule 6: Intersubnet drive $\tilde{\mathbf{Y}}\mathbf{X}^{\dagger}g(\mathbf{x})$	$-f(x)$	Between subnets for shifted sequence (3.46)
Rule 7: Delayed asymmetric drive $\int_0^{\infty} D(\tau)\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x}(t-\tau))d\tau$	$-\int_0^{\infty} D(\tau)f(x(t-\tau))d\tau$	Connections with time delays (3.12)
Rule 8: Linear combination $\alpha\mathbf{u}+\beta\mathbf{v}$	$\alpha u+\beta v$	Linearity of reduction (3.6), assuming $\mathbf{u}$ and $\mathbf{v}$ reduce to u and v, respectively

Note: $f(x)$ is the odd component of the original gain function $g(x)$ (see equation 3.10).

Several reduction rules are summarized in Table 1; let us now examine them one by one.

First, it is easy to see why reduction rule 2 in Table 1 is valid. As a constant vector, $\mathbf{c}^{\mathrm{T}}$ should commute with the time derivative:

$$\mathbf{c}^{\mathrm{T}}\frac{d}{dt}=\frac{d}{dt}\mathbf{c}^{\mathrm{T}}. \tag{3.4}$$

More specifically, we have

$$\mathbf{c}^{\mathrm{T}}\frac{d\mathbf{x}}{dt}=\frac{d}{dt}\left(\mathbf{c}^{\mathrm{T}}\mathbf{x}\right)=\frac{dx}{dt}, \tag{3.5}$$

where the last step follows from equation 3.1. This proves reduction rule 2 in Table 1.

The reduction rules always reduce a vector to a scalar. The operation of the compressor vector is linear:

$$\mathbf{c}^{\mathrm{T}}\left(\alpha\mathbf{u}+\beta\mathbf{v}\right)=\alpha\left(\mathbf{c}^{\mathrm{T}}\mathbf{u}\right)+\beta\left(\mathbf{c}^{\mathrm{T}}\mathbf{u}\right), \tag{3.6}$$

where α and β are any scalars that are independent of the state $\mathbf{x}$ of the network but may depend on time, and $\mathbf{u}$ and $\mathbf{v}$ are arbitrary vectors that may depend on the state $\mathbf{x}$. This is reduction 8 in Table 1. This rule means

that a dynamical equation can be reduced by first reducing each individual term separately and then summing up the results.

The linearity of the operation of the compressor vector also implies that

$$\mathbf{c}^{\mathrm{T}}(\varphi * \mathbf{u}) = \varphi * (\mathbf{c}^{\mathrm{T}}\mathbf{u}), \tag{3.7}$$

where $*$ indicates convolution in time and φ is an arbitrary scalar function that is independent of the network state but may depend on time.

The challenge now is to move the reduced variable x inside the gain function $g(\cdot)$, which is the sole source of nonlinearity in the dynamics. The key reductions rules are

$$\mathbf{c}^{\mathrm{T}}\left(\mathbf{X}\mathbf{X}^{\dagger}g(\mathbf{x})\right) = f(x), \tag{3.8}$$

$$\mathbf{c}^{\mathrm{T}}\left(\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x})\right) = -f(x), \tag{3.9}$$

where

$$f(x) = \frac{g(x) - g(-x)}{2} \tag{3.10}$$

is the odd component of the original gain function, which is antisymmetric:

$$f(-x) = -f(x). \tag{3.11}$$

We will derive these reduction rules in section 3.3.

Reduction rule 7 for delayed connections is given by

$$\mathbf{c}^{\mathrm{T}}\left(\int_0^{\infty} D(\tau)\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g\left(\mathbf{x}(t-\tau)\right)d\tau\right) = -\int_0^{\infty} D(\tau)f\left(x(t-\tau)\right)d\tau. \tag{3.12}$$

This may be regarded as a special case of equation 3.7 if we put $\varphi = D$ and $\mathbf{u} = \tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x})$, and use $\mathbf{c}^{\mathrm{T}}\mathbf{u} = -f$ (see equation 3.9). Thus, equation 3.12 (reduction rule 7) follows the linearity of the compressor vector operation and from equation 3.9 (reduction rule 4).

3.2 Examples of Reduced Systems. By applying the reduction rules in Table 1 to each of the terms in equation 2.7 and then summing up the results, we obtain the reduced system

$$\frac{dx}{dt} = -x + f(x) - \lambda f(x). \tag{3.13}$$

This system is one-dimensional and can never oscillate periodically, which is consistent with the fact that the corresponding asymmetric network

without time delay is unstable (Hopfield, 1982; Sompolinsky & Kanter, 1986; Kleinfeld, 1986).

For the time delay network, the reduction rules in Table 1 reduce the dynamical equation 2.11 to

$$\tau \frac{dx(t)}{dt} = -x(t) + f(x(t)) - \lambda \int_0^\infty D(t') f(x(t-t'))\, dt'. \tag{3.14}$$

This equation is self-contained and can oscillate periodically for suitable parameters (see section 4).

Similarly, the conjugate network dynamical equations 2.12 and 2.13 can be reduced to

$$\tau \frac{dx}{dt} = -x + f(x) - \lambda f(y), \tag{3.15}$$

$$\tau \frac{dy}{dt} = -y + f(y) + \lambda f(x). \tag{3.16}$$

Although all the rules in Table 1 use x as the label for the reduced variable, we can readily switch the role of $\mathbf{x}$ and $\mathbf{y}$ as well as the memory matrices $\mathbf{X}$ and $\mathbf{Y}$ and apply the same rules for reduced variable y. This reduced system has a two-dimensional state space and allows limit-cycle oscillation (see Figure 2 and section 4).

3.3 Derivation of the Key Reduction Rules

3.3.1 General Considerations. In this section we derive reduction rules 3 to 6 in Table 1, which all involve putting the reduced variable inside the nonlinear gain function f. For the remaining reduction rules in Table 1, reduction rules 2 and 8 follow from the linearity of the compression operation, and rule 7 may be regarded as a linearly weighted version of rule 4, as explained at the end of section 3.1.

The derivation of the reduction rules is based on two observations. The first observation is that the state of a network during sequential memory retrievals is a linear combination of the stored memory patterns. This is because any state component that is orthogonal to the stored memory patterns always decays exponentially and thus can be ignored. More precisely, the state $\mathbf{x}$ of a network can always be written as

$$\mathbf{x}(t) = \sum_{m=1}^{M} \phi_m(t) \mathbf{x}^m + \mathbf{v}(t), \tag{3.17}$$

where vector $\mathbf{v}$ is orthogonal to all stored memory patterns $\mathbf{x}^1, \ldots, \mathbf{x}^M$. The coefficients ϕ_m are given by

$$\phi_m(t) = \mathbf{x}_m^\dagger \mathbf{x}(t), \tag{3.18}$$

which obtains by left-multiplying equation 3.17 by $\mathbf{x}_m^\dagger$ in equation 2.2 and then using the orthogonality condition 2.3 and the fact that any $\mathbf{v}$ that is orthogonal to all $\mathbf{x}^m$ is also orthogonal to all $\mathbf{x}_m^\dagger$. The latter fact means that for $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^M]$, if $\mathbf{v}^{\mathrm{T}}\mathbf{X} = \mathbf{0}^{\mathrm{T}} = [0, \ldots, 0]$ with M entries, then $\mathbf{X}^\dagger\mathbf{v} = \mathbf{0}$. This is readily verified: $\mathbf{X}^\dagger\mathbf{v} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{v} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}(\mathbf{v}^{\mathrm{T}}\mathbf{X})^{\mathrm{T}} = \mathbf{0}$. The scalar coefficients $\phi_m(t)$ are the general expressions for the memory overlaps, as illustrated in Figure 1.

All the networks considered in section 2 have dynamical equations of the form

$$\tau\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \sum_{m=1}^{M} G_m(t)\mathbf{x}^m, \tag{3.19}$$

where the scalar function G_m includes the contribution from all nonlinear terms in the original equation. To see this, we rewrite a typical nonlinear term in the dynamical equation as $\mathbf{X}\mathbf{X}^\dagger g(\mathbf{x}) = \sum_m \mathbf{x}^m[\mathbf{x}_m^\dagger g(\mathbf{x})]$. Since the expression inside the brackets is a scalar, the final result is a linear combination of the memory patterns $\mathbf{x}^m$. Similarly, for intersubnet connections, a typical term $\tilde{\mathbf{X}}\mathbf{Y}^\dagger g(\mathbf{y}) = \sum_m \mathbf{x}^{m+1}[\mathbf{y}_m^\dagger g(\mathbf{y})]$ is also a linear combination of the memory patterns $\mathbf{x}^m$.

Now, plugging equation 3.17 into equation 3.19 yields

$$\tau\sum_{m=1}^{M}\dot{\phi}_m\mathbf{x}^m + \tau\dot{\mathbf{v}} = -\sum_{m=1}^{M}\phi_m\mathbf{x}^m - \mathbf{v} + \sum_{m=1}^{M}G_m\mathbf{x}^m, \tag{3.20}$$

where G_m depends on $\mathbf{v}$ implicitly. Since $\mathbf{v}$ is orthogonal to all $\mathbf{x}^m$, left-multiplying $\mathbf{v}^{\mathrm{T}}$ on both sides of equation 3.20 leads to

$$\tau\mathbf{v}^{\mathrm{T}}\dot{\mathbf{v}} = -\mathbf{v}^{\mathrm{T}}\mathbf{v} = -v^2, \tag{3.21}$$

where $v = \|\mathbf{v}\| = \sqrt{\mathbf{v}^{\mathrm{T}}\mathbf{v}}$ is the norm or the length of vector $\mathbf{v}$. Taking the time derivative on both sides of the equation $\mathbf{v}^{\mathrm{T}}\mathbf{v} = v^2$ yields $\mathbf{v}^{\mathrm{T}}\dot{\mathbf{v}} = v\dot{v}$, which reduces equation 3.21 to

$$\tau\dot{v} = -v. \tag{3.22}$$

Therefore, the norm of vector $\mathbf{v}$ always vanishes exponentially.

From now on, we discard $\mathbf{v}$ and write the state of the system as a linear combination of the memory states:

$$\mathbf{x}(t) = \sum_{m=1}^{M}\phi_m(t)\mathbf{x}^m. \tag{3.23}$$

During sequential memory retrieval, the memory overlaps $\phi_1(t), \phi_2(t), \ldots$ should be time-shifted versions of one another, namely,

$$\phi_2(t) = \phi_1(t - T), \quad \phi_3(t) = \phi_2(t - T) = \phi_1(t - 2T), \quad \ldots \tag{3.24}$$

and so on, where T is the memory duration as illustrated in Figure 1.

The second observation is that during slow sequential retrievals, the instantaneous state of the network (or a subnet) can overlap significantly with at most two consecutive memory patterns. In other words, no more than two of the coefficients $\phi_m(t)$ are significantly different from zero at any given time. This observation is approximately true in simulations, as can be seen in the memory overlaps in Figure 1. Additional examples are in Figure 3.

This approximation is valid when the temporal sequence retrieval is relatively slow such that all individual units basically show the same stereotypical manner of changing its state, regardless of the exact temporal sequence embedded. Figure 3 compares a network with only 2 memory patterns ($M = 2$) against another network with 10 memory patterns ($M = 10$). The time profiles of the individual units have only 4 possible types when $M = 2$, and the total number of possible time profiles increases to $2^M = 1024$ when $M = 10$. However, if we compare the behaviors of individual units within the time interval between two neighboring vertical gray lines in Figure 3 (the interval is equal to the memory duration T in Figure 1), there are only four basic motifs:

1. Stay on (++).
2. Stay off (−).
3. Decrease from on to off (+−).
4. Increase from off to on (−+).

We have approximately the same basic motifs no matter whether $M = 2$ or $M = 10$. The difference between these two cases is mainly in how these motifs are combined to form a sequence. By contrast, if the retrieval is too fast, the state of an individual unit may not have enough time to fully settle into a quasi-stationary state before the drive toward the next memory pattern shows up. Now the individual units change their states in a sequence-dependent manner rather than following the same stereotypical basic motifs. For example, consider a unit that follows the sequence $1 \to -1 \to 1$ and compare it against another unit following the sequence $1 \to -1 \to -1 \to 1$. The first one stays at the -1 state for a shorter period of time and thus settles more shallowly (for the lack of time) before returning to state 1. In the second sequence, the state can go toward -1 more deeply, but it also takes longer to come back to 1. Thus, the time profiles of the two units for the same transition segment $-1 \to 1$ should be quite different. Our second observation is roughly equivalent to assuming that the temporal sequence

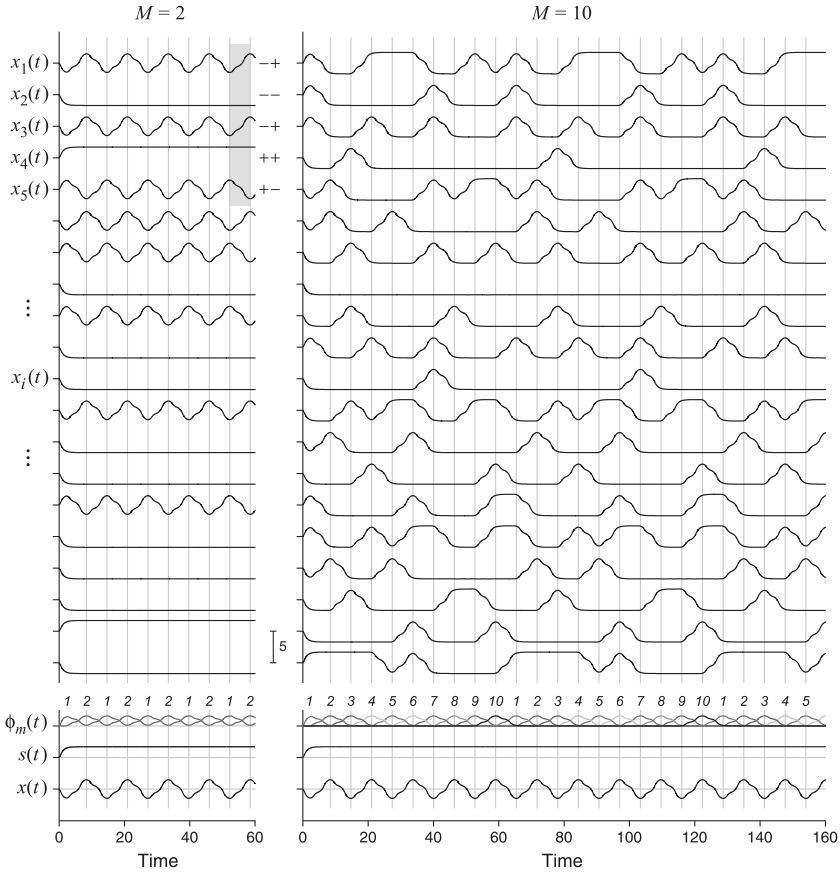

Figure 3: The states $x_i(t)$ of a sample of individual units are compared with three collective variables: the memory overlaps $\phi_m(t)$ (equation 3.18), the sum of the memory overlaps $s(t)$ (equation 5.2), and the reduced oscillatory variable $x(t)$ (equation 5.3). The data are from a subnet of a conjugate network, and each subnet stores either $M = 2$ (left column) or $M = 10$ (right column) memory patterns, arranged cyclically. The initial state of the subnet was set to zero so that one can easily tell whether a stationary state is an on or off state. Each tick mark of the vertical axes indicates the value 0 of the corresponding trace. The vertical scale is the same for all traces, and the distance between two neighboring tick marks is 5. Gray vertical lines are based on the extrema of the reduced variable $x(t)$. The shaded area in the left panel highlights several examples of the basic motifs. Parameters: probability $p = 0.3$ for on state, $N = 500$ in each subnet, $\lambda = 0.7$, $\tau = 1$, and $k = 2$ for the gain function in equation 4.7.

retrieval is slow enough so that the actual sequence has a negligible effect on the manner that an individual unit changes its state.

As a consequence of the second observation, without loss of generality we can focus on only two consecutive memory patterns, say, $\mathbf{x}^1$ and $\mathbf{x}^2$, and write the state of the network as a mixture of the two memory states:

$$\mathbf{x}(t) = \phi_1(t)\mathbf{x}^1 + \phi_2(t)\mathbf{x}^2. \tag{3.25}$$

This assumption is not identical to assuming that a total of two memory patterns are stored in the network, although in both cases, equation 3.25 holds. The statistics of the weight matrix generally depends on the number of embedded memory patterns (see below and appendix A).

Finally, consider the component form of equation 3.25 for the ith unit: $x_i(t) = \phi_1(t)x_i^1 + \phi_2(t)x_i^2$. When $x_i^1 = -1$ and $x_i^2 = 1$, we get the increase motif: $x_i(t) = -\phi_1(t) + \phi_2(t)$. When $x_i^1 = 1$ and $x_i^2 = -1$, we get the decrease motif: $x_i(t) = \phi_1(t) - \phi_2(t)$, which is related to the increase motif by a sign flip. This explains the symmetry with respect to sign flip in the time profiles of individual units seen in Figure 3.

3.3.2 Derivation. Left-multiplying equation 3.25 by the compressor vector in equation 3.2, we obtain an explicit expression for the reduced variable defined by equation 3.1:

$$x(t) = \mathbf{c}^{\mathrm{T}}\mathbf{x}(t) = \left(\sum_{m=1}^{M}(-1)^m\mathbf{x}_m^{\dagger}\right)\left(\phi_1(t)\mathbf{x}^1 + \phi_2(t)\mathbf{x}^2\right) = \phi_2(t) - \phi_1(t), \tag{3.26}$$

where the last step follows from the orthogonality condition 2.3. By equations 3.2 and 2.3,

$$\mathbf{c}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\dagger} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger}\mathbf{X}\mathbf{X}^{\dagger} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger} = \mathbf{c}^{\mathrm{T}}. \tag{3.27}$$

Let $\mathbf{c}^{\mathrm{T}} = [c_1, \ldots, c_N]$ be the compressor vector in component form. We have

$$\mathbf{c}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\dagger}g(\mathbf{x}) = \mathbf{c}^{\mathrm{T}}g(\mathbf{x}) \tag{3.28}$$

$$= \sum_{i=1}^{N} c_i g(\phi_1 x_i^1 + \phi_2 x_i^2) \tag{3.29}$$

$$\begin{aligned} =& C_{++}g(\phi_1 + \phi_2) + C_{--}g(-\phi_1 - \phi_2) + C_{+-}g(\phi_1 - \phi_2) \\ &+ C_{-+}g(-\phi_1 + \phi_2), \end{aligned} \tag{3.30}$$

where

$$C_{++} = \sum_{i \in I_{++}} c_i \quad \text{with} \quad I_{++} = \left\{ i\text{: } x_i^1 = +1 \text{ and } x_i^2 = +1 \right\}, \tag{3.31}$$

$$C_{--} = \sum_{i \in I_{--}} c_i \quad \text{with} \quad I_{--} = \left\{ i\text{: } x_i^1 = -1 \text{ and } x_i^2 = -1 \right\}, \tag{3.32}$$

$$C_{+-} = \sum_{i \in I_{+-}} c_i \quad \text{with} \quad I_{+-} = \left\{ i\text{: } x_i^1 = +1 \text{ and } x_i^2 = -1 \right\}, \tag{3.33}$$

$$C_{-+} = \sum_{i \in I_{-+}} c_i \quad \text{with} \quad I_{-+} = \left\{ i\text{: } x_i^1 = -1 \text{ and } x_i^2 = +1 \right\}. \tag{3.34}$$

Step 3.29 is a substitution by equation 3.25. In step 3.30, all terms in the sum in equation 3.29 are grouped according to the values of $x_i^1 = \pm 1$ and $x_i^2 = \pm 1$. The coefficient C_{++} is the sum of all the entries in $\mathbf{c}^{\mathrm{T}}$ corresponding to the units for which both x_i^1 and x_i^2 are equal to $+1$. Similarly, C_{-+} is the sum of all the entries in $\mathbf{c}^{\mathrm{T}}$ corresponding to the units for which $x_i^1 = -1$ but $x_i^2 = +1$, and so on.

The four coefficients defined by equations 3.31 to 3.34 are determined by the statistics of matrix $\mathbf{X}^{\dagger}$. Only two among these four are independent, because the following relations always hold true:

$$C_{++} = C_{--}, \tag{3.35}$$

$$C_{-+} - C_{+-} = 1. \tag{3.36}$$

To see this, first note that $\mathbf{c}^{\mathrm{T}}\mathbf{x}^1 = -1$, which follows from equations 3.2 and 2.3. On the other hand,

$$\mathbf{c}^{\mathrm{T}}\mathbf{x}^1 = \sum_{i=1}^{N} c_i x_i^1 = C_{++} - C_{--} + C_{+-} - C_{-+} = -1, \tag{3.37}$$

which follows from equations 3.31 to 3.34. Similarly,

$$\mathbf{c}^{\mathrm{T}}\mathbf{x}^2 = \sum_{i=1}^{N} c_i x_i^2 = C_{++} - C_{--} - C_{+-} + C_{-+} = 1. \tag{3.38}$$

Summation of the two equations above yields equation 3.35, while their subtraction yields equation 3.36.

Suppose the entries of the memory pattern matrix $\mathbf{X}$ are drawn independently from a random binary distribution with arbitrary probability p for

$+1$ and probability $1 - p$ for -1; then

$$C_{++} = C_{--} = 0, \quad C_{-+} = \frac{1}{2}, \quad C_{+-} = -\frac{1}{2} \tag{3.39}$$

in the limit of large network size ($N \to \infty$) as long as the total number of memory patterns is even (see appendix A for derivation). The exact value of p does not matter here. When the number of memory patterns M is odd, equations 3.39 hold true in the limit of large network size ($N \to \infty$) if one of the following two additional conditions is also satisfied: (1) the probabilities for on and off memory states are equal ($p = 1/2$), or (2) the number of memory patterns increases indefinitely ($M \to \infty$). For finite network size, equations 3.39 provide good approximations especially when M is even or when $p = 1/2$ (see appendix A). For simplicity, we always assume that M is even, while allowing p to be arbitrary.

In the following we will use equations 3.39 to derive the reduction rules. Plugging them into expression 3.30 yields

$$\mathbf{c}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\dagger}g(\mathbf{x}) = -\frac{1}{2}g(\phi_1 - \phi_2) + \frac{1}{2}g(-\phi_1 + \phi_2) = \frac{g(x) - g(-x)}{2}, \tag{3.40}$$

where the last step follows from equation 3.26. This establishes reduction rule 3 in Table 1.

To derive reduction rules involving asymmetric connections, first consider $\tilde{\mathbf{X}} = \mathbf{X}\tilde{\mathbf{I}}$, where $\tilde{\mathbf{X}}$ is derived from $\mathbf{X}$ by cyclically rotating its columns leftward by one position (see equation 2.10), and similarly, $\tilde{\mathbf{I}}$ is derived from the $M \times M$ identity matrix $\mathbf{I}$ by the same cyclic rotation. Now

$$\mathbf{X}^{\dagger}\tilde{\mathbf{X}} = \mathbf{X}^{\dagger}\mathbf{X}\tilde{\mathbf{I}} = \tilde{\mathbf{I}} \tag{3.41}$$

by the orthogonality condition 2.3. Thus,

$$\boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger}\tilde{\mathbf{X}} = \boldsymbol{\omega}^{\mathrm{T}}\tilde{\mathbf{I}} = -\boldsymbol{\omega}^{\mathrm{T}}, \tag{3.42}$$

where the last step follows from equation 3.3. Equations 3.42 and 3.2 imply that

$$\mathbf{c}^{\mathrm{T}}\tilde{\mathbf{X}}\mathbf{X}^{\dagger} = \left(\boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger}\tilde{\mathbf{X}}\right)\mathbf{X}^{\dagger} = -\boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger} = -\mathbf{c}^{\mathrm{T}}. \tag{3.43}$$

Therefore, we have

$$\mathbf{c}^{\mathrm{T}}\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x}) = -\mathbf{c}^{\mathrm{T}}g(\mathbf{x}) = -\frac{g(x) - g(-x)}{2}, \tag{3.44}$$

where the last equality follows from equations 3.28 and 3.40. This proves reduction rule 4 in Table 1.

Reduction rules 5 and 6 in Table 1 involve intersubnet connections between different subnets. The results turn out to be identical to that for connections within the same subnet. Let $\mathbf{d}^{\mathrm{T}} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{Y}^{\dagger}$ be the compressor vector for another subnet. Then $\mathbf{d}^{\mathrm{T}}\mathbf{Y}\mathbf{X}^{\dagger} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{Y}^{\dagger}\mathbf{Y}\mathbf{X}^{\dagger} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger} = \mathbf{c}^{\mathrm{T}}$ because $\mathbf{Y}^{\dagger}\mathbf{Y} = \mathbf{I}$. Thus, we have

$$\mathbf{d}^{\mathrm{T}}\mathbf{Y}\mathbf{X}^{\dagger}g(\mathbf{x}) = \mathbf{c}^{\mathrm{T}}g(\mathbf{x}) = \frac{g(x) - g(-x)}{2}, \tag{3.45}$$

where the last equality follows from equations 3.28 and 3.40. This proves reduction rule 5 in Table 1. An argument analogous to that for deriving equation 3.44 leads to

$$\mathbf{d}^{\mathrm{T}}\tilde{\mathbf{Y}}\mathbf{X}^{\dagger}g(\mathbf{x}) = -\mathbf{c}^{\mathrm{T}}g(\mathbf{x}) = -\frac{g(x) - g(-x)}{2}. \tag{3.46}$$

This is reduction rule 6 in Table 1.

4 Analysis of the Reduced Systems

In this section, we analyze several reduced systems in detail. For each system, we start with some general considerations and then focus on special cases that allow analytical solutions. The theoretical predictions by the reduced system are compared with numerical simulations of the original networks.

4.1 Example 1: Conjugate Network with Piecewise Linear Gain Function

4.1.1 General Considerations. Consider the reduced system of the conjugate network given by equations 3.15 and 3.16. The origin $(x, y) = (0, 0)$ is always an equilibrium state because, by equation 3.10, the gain function f is an odd function with $f(0) = 0$. Linearization around the origin yields

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} k-1 & -\lambda k \\ \lambda k & k-1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \tag{4.1}$$

where $k = f'(0)$ is the slope of the gain function at 0. The eigenvalues are $k - 1 \pm i\lambda k$, meaning that the origin is an unstable equilibrium state (a source) when $k > 1$. In the following, we always assume that

$$k > 1. \tag{4.2}$$

Now that the flow $(\dot{x}, \dot{y})$ diverges around the origin, if we can show that the flow converges toward the origin for large radius $r = \sqrt{x^2 + y^2}$, then a limit cycle should exist by the Poincaré-Bendixson theorem (Hirsch & Smale, 1974), assuming that the nullclines of equations 3.15 and 3.16 intersect at only the origin so that there is no other equilibrium state. It follows from equations 3.15 and 3.16 that

$$r\dot{r} = x\dot{x} + y\dot{y} = -x^2 + \big(f(x) - \lambda f(y)\big)x - y^2 + \big(f(y) + \lambda f(x)\big)y \tag{4.3}$$

$$\leq -r^2 + \big(|f(x)| + \lambda|f(y)|\big)\,|x| + \big(|f(y)| + \lambda|f(x)|\big)\,|y| \tag{4.4}$$

$$\leq -r^2 + B(1+\lambda)(|x| + |y|) \tag{4.5}$$

$$\leq -r^2 + \sqrt{2}B(1+\lambda)r, \tag{4.6}$$

where in step 4.5, we assume that the gain function is bounded, namely, $|f| \leq B$, and step 4.6 follows from the inequality $|x| + |y| \leq \sqrt{2}r$. Thus, we are guaranteed to have $\dot{r} < 0$ or converging flow when $r > \sqrt{2}B(1+\lambda)$.

4.1.2 Exact Solution for Piecewise Linear Gain Function. In the following we assume that the gain function is a clipped linear function,

$$f(x) = \begin{cases} 1 & \text{if } x > k^{-1}, \\ kx & \text{if } -k^{-1} \leq x \leq k^{-1}, \\ -1 & \text{if } x < -k^{-1}, \end{cases} \tag{4.7}$$

where k (>1) is the slope of the middle linear segment. This gain function can be written equivalently as $f(x) = (|kx+1| - |kx-1|)/2 = \min(1, \max(-1, kx))$. Here the maximum value of $|f|$ is taken as 1 without loss of generality. To see this, suppose we replace the gain function $f(u)$ by the new gain function $Bf(u/B)$, whose maximum becomes B while the middle segment slope is still k. Equations 3.15 and 3.16 with the new gain function can be changed back to their original forms if we simply replace variables x and y by the new variables x/B and y/B.

Equations 3.15 and 3.16 allow a limit cycle oscillation when

$$\lambda > \lambda_c \equiv 1 - k^{-1}, \tag{4.8}$$

where λ_c is a critical coupling strength. It follows from equations 4.2 and 4.8 that $0 < \lambda_c \leq 1$, and $\lambda_c = 1$ is achieved in the limit $k \to \infty$ or when the gain function approaches the step function $f(x) = \text{sign}(x)$. Figure 4A shows an example with $k = 2$ and $\lambda_c = 1/2$.

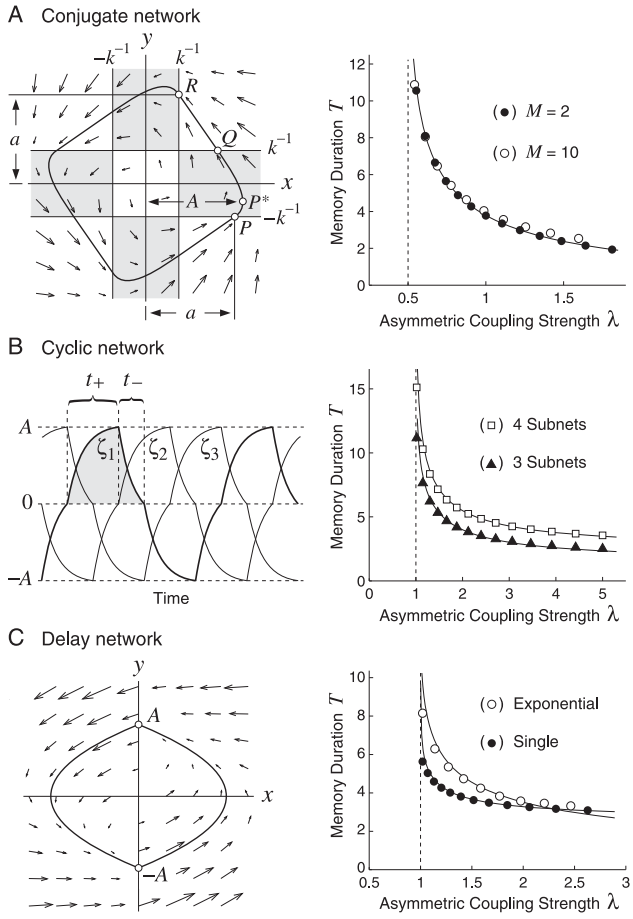

Figure 4: Analytical solutions of several reduced systems are confirmed by numerical simulations of the original networks. Right panels: The memory durations predicted by the reduced systems (curves) are compared with the results of the original networks (symbols). In all cases, $\tau = 1$. The memory patterns were drawn randomly with equal on and off probability, and the connections weights were constructed using the approximation in equation 2.6 rather than the exact pseudoinverse. (A) Conjugate network with clipped linear gain function ($k = 2$, $N = 5000$, and $M = 2$ or 10 in each subnet). Left: Limit cycle in the phase plane of the reduced system, which has nine piecewise linear regions separated by $x = \pm k^{-1}$ and $y = \pm k^{-1}$. The arrows indicate vector field $(\dot{x}, \dot{y})$ by equations 3.15 and 3.16. (B) Cyclic networks with step gain function and either three or four subnets ($N = 5000$ and $M = 5$ in each subnet). Left: stable oscillation of the reduced system with three subnets. (C) Delay networks with step gain function and either a single delay or exponentially distributed delays ($\tau_D = 2$, $N = 25{,}000$, and $M = 6$). Left: limit cycle in the equivalent phase plane of the exponential system (see equations 4.63 and 4.64).

To see why condition 4.8 holds, set $\dot{x} = 0$ in equation 3.15 and consider the nullcline equation $f(x) = x + \lambda f(y)$, with $-1 \le f(y) \le 1$. For successful transition toward the next memory, λ should be large enough such that $x + \lambda$ or $x - \lambda$ intersects $f(x)$ at a single point, which specifies the next state. If λ is too small, there are multiple intersection points that suggest additional stationary states. The critical case corresponds to the tangential situation where the lines $x \pm \lambda$ touch the gain function at $x = \pm k^{-1}$, which leads to equation 4.8. The critical strength can be obtained for an arbitrary gain function by more general arguments based on nullcline analysis (see appendix C).

To solve for the limit cycle, use the lines $x = \pm k^{-1}$ and $y = \pm k^{-1}$ to separate the phase plane into nine piecewise linear regions, as distinguished by shades in Figure 4A (left). Since the system is symmetric with respect to a 90 degree rotation, we only need to solve one-fourth of the limit cycle, say, the segment from point P to point R. The time t_{PQ} for traveling from point P to point Q (with y changing from $-k^{-1}$ to k^{-1}) can be solved from equation 3.16, which now reads $\tau\dot{y} = -y + ky + \lambda$. The result is

$$t_{PQ} = \tau \ln \gamma, \tag{4.9}$$

where

$$\gamma \equiv \left(\frac{\lambda + \lambda_c}{\lambda - \lambda_c}\right)^{1/\lambda_c - 1}. \tag{4.10}$$

Similarly, the time t_{QR} for traveling from point Q to point R (with y changing from k^{-1} to a) can be solved also from equation 3.16, which now becomes $\tau\dot{y} = -y + 1 + \lambda$, yielding

$$t_{QR} = \tau \ln \frac{\lambda + \lambda_c}{\lambda + 1 - a}. \tag{4.11}$$

Here the parameter a (>0) is defined by the intersection of the limit cycle with the lines $x = \pm k^{-1}$ or $y = \pm k^{-1}$, as shown in Figure 4A (left). Since the denominator in equation 4.11 has to be positive, we have

$$0 < a < \lambda + 1. \tag{4.12}$$

The exact value of a is determined as follows. When traveling from point P to point R, $f(x) = 1$ always holds true, and so we can multiply equation 3.16 by λ and then add to equation 3.15 to eliminate $f(y)$ and get

$$\tau \frac{d}{dt}(x + \lambda y) = -(x + \lambda y) + 1 + \lambda^2. \tag{4.13}$$

Using $x + \lambda y$ as a new variable, which changes from $a - \lambda k^{-1}$ to $k^{-1} + \lambda a$ during the time interval t_{PR}, we solve equation 4.13 and obtain

$$t_{PR} = \tau \ln \frac{\lambda^2 - \lambda_c \lambda + \lambda + 1 - a}{\lambda^2 - a\lambda + \lambda_c}. \tag{4.14}$$

Since $t_{PR} = t_{PQ} + t_{QR}$, it follows from equations 4.9, 4.11, and 4.14 that

$$\frac{\lambda^2 - \lambda_c \lambda + \lambda + 1 - a}{\lambda^2 - a\lambda + \lambda_c} = \gamma \frac{\lambda + \lambda_c}{\lambda + 1 - a}. \tag{4.15}$$

This equation is equivalent to a quadratic equation of a, which has the solution

$$a = \lambda + 1 - \eta + \sqrt{\eta^2 - \gamma\lambda^2 + \gamma\lambda_c^2}, \tag{4.16}$$

with

$$\eta \equiv \frac{\gamma - 1}{2}\lambda^2 + \frac{\gamma + 1}{2}\lambda_c \lambda, \tag{4.17}$$

where an extraneous negative solution is discarded.

The predicted memory duration T is obtained as follows. Since $T = 2t_{PR} = 2(t_{PQ} + t_{QR})$, it follows from equations 4.9 and 4.11 that

$$T = 2\tau \ln \frac{\gamma(\lambda + \lambda_c)}{\lambda + 1 - a}, \tag{4.18}$$

where λ_c, γ, and a are given by equations 4.8, 4.10, and 4.16, respectively.

The shape of the limit cycle segment from point P to point Q is readily solved from equations 3.15 and 3.16 as a curve $(X(t), Y(t))$ parameterized by time, starting from the initial condition $(X(0), Y(0)) = P = (a, -k^{-1})$. Alternatively, the trajectory can also be solved from

$$\frac{dx}{dy} = \frac{\dot{x}}{\dot{y}} = \frac{-x + f(x) - \lambda f(y)}{-y + f(y) + \lambda f(x)}. \tag{4.19}$$

As illustrated in Figure 4A (left), the maximum amplitude A ($\geq a$) of variable x should be reached at a point $P^* = (A, P_y^*)$, which is characterized by $\dot{x} = 0$. Figure 1D illustrates the meaning of the maximum amplitude A in time domain. The maximum amplitude of variable y is also equal to A due to the symmetry of the system. It follows from equation 3.15 that $\dot{x} = -x + 1 - \lambda f(y) \leq -x + 1 + \lambda$ because $|f(y)| \leq 1$. Now, setting $\dot{x} = 0$ yields $0 \leq -A + 1 + \lambda$. Thus,

$$a \leq A \leq 1 + \lambda. \tag{4.20}$$

To determine the exact value of A, we take the solution curve $(X(t), Y(t))$ mentioned above and use the condition $\dot{X}(t_{PP^*}) = 0$ to solve for the time t_{PP^*} for traveling from point P to point P^*. Then we obtain $A = X(t_{PP^*})$, which reads

$$A = \frac{\lambda^2}{\lambda_c} + 1 - \frac{1}{\lambda_c}\left(\lambda^2 - \lambda_c\lambda\right)^{1-\lambda_c}\left(\lambda^2 - \lambda_c\lambda + \lambda + 1 - a\right)^{\lambda_c}, \tag{4.21}$$

together with $P_y^* = Y(t_{PP^*}) = (\lambda_c - 1)(A - 1)/\lambda$. The segment of the limit cycle between point $Q = (Q_x, k^{-1})$ and point $R = (k^{-1}, a)$ turns out to be a straight line, with $Q_x = \lambda^2 + \lambda_c\lambda - \lambda + 1 - (\lambda^2 - \lambda_c\lambda + \lambda + 1 - a)/\gamma$.

4.1.3 Limit Cases. Now consider a few limit cases of the solutions obtained above:

1. As $k \to \infty$ (step gain function limit), we have $a \to 2$, $A \to 2$, and $T \to 2\tau \ln((\lambda + 1)/(\lambda - 1))$, which is consistent with equation 4.43 to be discussed later. The shape of the limit cycle approaches the square described by $|x| + |y| = 2$ (see Figure 5B, left).
2. As $\lambda \to \lambda_c$ (weak coupling limit), we have $a \to 1 + \lambda_c$, $A \to 1 + \lambda_c$, and $T \to \infty$, with an asymptotic formula $T \sim (2\tau/\lambda_c) \ln\left(2\lambda_c^{\lambda_c+1}/(\lambda - \lambda_c)\right)$, which follows from equation 4.18 using the asymptote: $(\lambda + 1 - a) \sim (\lambda - \lambda_c)/\lambda_c$. Therefore, there is no limit on how slow the cycle limit can be. The shape of the limit cycle approaches a square made of line segments described by $|x + \lambda_c y| = \lambda_c^2 + 1$ and $|\lambda_c x - y| = \lambda_c^2 + 1$ (see Figure 5B, middle).
3. As $\lambda \to \infty$ (strong coupling limit), we have

$$a_\infty \equiv \lim_{\lambda\to\infty} a = \lambda_c + \sqrt{(2\lambda_c^2 + 1)/3}, \tag{4.22}$$

$$A_\infty \equiv \lim_{\lambda\to\infty} A = (\lambda_c + 1)/2 + \sqrt{(2\lambda_c^2 + 1)/3}, \tag{4.23}$$

and $\lim_{\lambda\to\infty} T \to 0$, with an asymptotic formula $T \sim 2\tau(a_\infty - \lambda_c + 1)/\lambda$. Thus, there is no top speed limit on the limit cycle oscillation. The segment of the limit cycle between point P and point Q (see Figure 4A, left) approaches the parabola given by $x + y^2/(2(1 - \lambda_c)) = A_\infty$, while the segment between Q and R approaches the line $x + y = a_\infty - \lambda_c + 1$ (see Figure 5B, right).

An implicit assumption underlying most analysis in this section is that the limit cycle should stay outside the center square enclosed by $\pm k^{-1}$ in the phase plane (see Figures 4A and 5A). Setting $a_\infty = k^{-1}$, with a_∞ given by equation 4.22, we find $\lambda_c = 1/5$ or $k = 5/4$. When $k \geq 5/4$, parameter a as given by equation 4.16 decreases monotonically as λ increases, and so if the intrusion into the center square does not occur in the limit $\lambda \to \infty$, it cannot occur for any $\lambda > \lambda_c$. The critical case where the limit cycle touches a corner

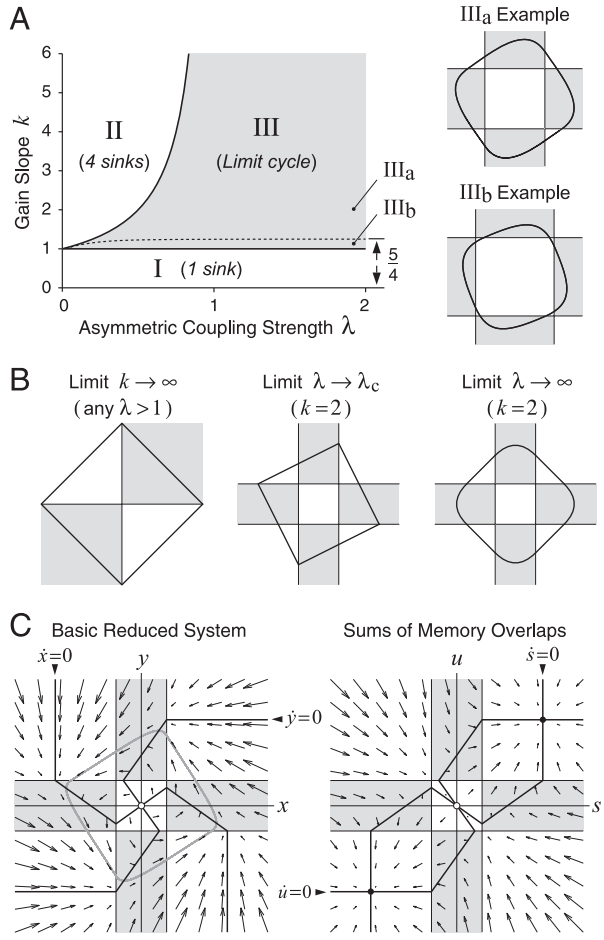

Figure 5: Properties of the reduced system for the conjugate network with clipped linear gain function. (A) Phase diagram for parameters k and λ. Regions I and II allow only point attractors (sinks). A limit cycle exists in region III (shaded area bounded by $k > 1$ and $\lambda > \lambda_c = 1 - k^{-1}$). In subregion III$_a$, the limit cycle avoids the center square enclosed by the straight lines $x = \pm k^{-1}$ and $y = \pm k^{-1}$, whereas in subregion III$_b$, the limit cycle enters the center square, as illustrated by the two examples (top: $k = 1.5$ and $\lambda = 1$; bottom: $k = 1.15$ and $\lambda = 0.5$; plot range: $[-1.5, 1.5]$ for both axes). (B) Shape of the limit cycle in three limit cases. The phase plane is divided into piecewise linear regions (distinguished by shades) by the straight lines $x = \pm k^{-1}$ and $y = \pm k^{-1}$. Plot range: $[-2, 2]$ for both axes. (C) Nullclines of the basic reduced system (left, equations 3.15 and 3.16) and the accompanying system for the sums of memory overlaps (right, equations 5.22 and 5.23) intersect at stable (filled circles) or unstable (open circles) equilibrium states. The limit cycle is shown in gray (left). Parameters: $k = 2$, $\lambda = 0.7$. Plot range: $[-2.5, 2.5]$ for both axes.

of the center square means that $a_\infty = k^{-1}$, which yields $k = 5/4$, as stated above. Therefore, if we assume $k \geq 5/4$, it is sufficient to guarantee that the limit cycle stays clear of the center square so that our analysis in this section is valid. For $1 < k < 5/4$, the limit cycle may or may not actually enter the home square. If it does, the system is still piecewise linear and solvable, but further consideration will be omitted.

4.1.4 Phase Diagram. The phase diagram for parameters k and λ is shown in Figure 5A. The parameter space is divided by the two curves $k = 1$ and $\lambda = \lambda_c = 1 - k^{-1}$ into the following three regions:

I. $k < 1$. The system described by equations 3.15 and 3.16 has a single equilibrium point, a sink at the origin $(0, 0)$.

II. $k > 1$ and $\lambda < \lambda_c$. There are nine equilibrium points, including a source at the origin, four sinks, and four saddle points. The locations of these equilibrium points are symmetric with respect to a 90 degree rotation about the origin. In the quadrant with $x > 0$ and $y > 0$, the saddle is located at $(\lambda/\lambda_c - \lambda, 1 + \lambda^2/\lambda_c)$, while the sink is located at $(1 - \lambda, 1 + \lambda)$.

III. $k > 1$ and $\lambda > \lambda_c$ (shaded region). The system has a limit cycle and a source at the origin. The shaded region is divided into two subregions (III_a and III_b) by the dashed curve, which was obtained for each given k by numerically solving for λ from the equation $a = k^{-1}$, with a given by equation 4.16. This dashed curve approaches the asymptote $k = 5/4$ for large λ . In the subregion below the dashed curve (III_b), the limit cycle intrudes into the center square enclosed by $\pm k^{-1}$ (see an example in Figure 5A), whereas in the subregion above the dashed curve (III_a), the limit cycle stays outside the center square so that the formulas derived in this section are valid.

4.2 Example 2: Cyclic Network with Step Gain Function

4.2.1 General Cyclic Networks. A cyclic network consists of n Hopfield networks linked into a ring by unidirectional connections. Each subnet drives memory transitions in the next subnet so that memory retrievals occur cyclically. The conjugate network may be regarded as the special case with $n = 2$.

Consider a cyclic network with $n = 3$ subnets that obey the dynamical equations:

$$\tau \dot{\mathbf{x}} = -\mathbf{x} + \mathbf{X}\mathbf{X}^\dagger g(\mathbf{x}) + \lambda \tilde{\mathbf{X}}\mathbf{Z}^\dagger g(\mathbf{z}), \tag{4.24}$$

$$\tau \dot{\mathbf{y}} = -\mathbf{y} + \mathbf{Y}\mathbf{Y}^\dagger g(\mathbf{y}) + \lambda \mathbf{Y}\mathbf{X}^\dagger g(\mathbf{x}), \tag{4.25}$$

$$\tau \dot{\mathbf{z}} = -\mathbf{z} + \mathbf{Z}\mathbf{Z}^\dagger g(\mathbf{z}) + \lambda \mathbf{Z}\mathbf{Y}^\dagger g(\mathbf{y}). \tag{4.26}$$

The stored memory patterns $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^M]$, $\mathbf{Y} = [\mathbf{y}^1, \ldots, \mathbf{y}^M]$, and $\mathbf{Z} = [\mathbf{z}^1, \ldots, \mathbf{z}^M]$ are retrieved in the following order:

$$\mathbf{x}^1 \to \mathbf{y}^1 \to \mathbf{z}^1 \to \mathbf{x}^2 \to \mathbf{y}^2 \to \mathbf{z}^2 \to \cdots \to \mathbf{x}^M \to \mathbf{y}^M \to \mathbf{z}^M. \tag{4.27}$$

The only place where a shifted memory matrix $\tilde{\mathbf{X}}$ occurs is the last term in equation 4.24. This is because in the sequence above, only transition $\mathbf{z}^m \to \mathbf{x}^{m+1}$ advances the memory index from m to $m+1$, whereas all other transitions, such as $\mathbf{x}^m \to \mathbf{y}^m$, leave the memory index m unchanged.

Left-multiplying the three equations 4.24 to 4.26 by $\boldsymbol{\omega}^T\mathbf{X}^\dagger$, $\boldsymbol{\omega}^T\mathbf{Y}^\dagger$, and $\boldsymbol{\omega}^T\mathbf{Z}^\dagger$, respectively, we get the reduced system:

$$\tau\dot{x} = -x + f(x) - \lambda f(z), \tag{4.28}$$

$$\tau\dot{y} = -y + f(y) + \lambda f(x), \tag{4.29}$$

$$\tau\dot{z} = -z + f(z) + \lambda f(y). \tag{4.30}$$

More generally, a cyclic network with n subnets obeys the dynamical equations:

$$\tau_1\dot{\boldsymbol{\zeta}}_1 = -\boldsymbol{\zeta}_1 + \mathbf{Z}_1\mathbf{Z}_1^\dagger g_1(\boldsymbol{\zeta}_1) + \lambda_1\tilde{\mathbf{Z}}_1\mathbf{Z}_n^\dagger g_n(\boldsymbol{\zeta}_n), \tag{4.31}$$

$$\tau_k\dot{\boldsymbol{\zeta}}_k = -\boldsymbol{\zeta}_k + \mathbf{Z}_k\mathbf{Z}_k^\dagger g_k(\boldsymbol{\zeta}_k) + \lambda_k\mathbf{Z}_k\mathbf{Z}_{k-1}^\dagger g_{k-1}(\boldsymbol{\zeta}_{k-1}), \quad k = 2, \ldots, n \tag{4.32}$$

where time constant τ_l, gain function g_l, and coupling strength λ_l are allowed to vary from subnet to subnet ($l = 1, \ldots, n$). This system subsumes the the conjugate network (equations 2.12 and 2.13) and the cyclic network with three subnets (equations 4.24–4.26) as special cases. The reduced system is given by

$$\tau_1\dot{\zeta}_1 = -\zeta_1 + f_1(\zeta_1) - \lambda_1 f_n(\zeta_n), \tag{4.33}$$

$$\tau_k\dot{\zeta}_k = -\zeta_k + f_k(\zeta_k) + \lambda_k f_{k-1}(\zeta_{k-1}), \qquad k = 2, \ldots, n, \tag{4.34}$$

where ζ_l is the variable reduced from the state $\boldsymbol{\zeta}_l$ of the lth subnet in equations 4.31 and 4.32 and $f_l(x) = (g_l(x) - g_l(-x))/2$ is the odd component of gain function g_l ($l = 1, \ldots, n$).

4.2.2 Exact Solution for Step Gain Function. In the following, we consider a cyclic network with n subnets, assuming a single step gain function $f(x) = \text{sign}(x)$ and a single coupling strength $\lambda > \lambda_c = 1$ for all the units. The system described by equations 4.33 and 4.34 is piecewise linear and can be solved analytically. As illustrated in Figure 4B, the system has a periodic

solution, and the predicted memory duration T is equal to one-half of the period. It can be computed as $T = t_+ + t_-$, where t_+ is the duration of the rising phase and t_- is the the duration of the falling phase with $\zeta_1 > 0$ (see Figure 4B). During the rising phase, ζ_1 increases from 0 to A, with A being the amplitude of oscillation. Here ζ_1 satisfies equation 4.33, which now becomes $\tau\dot{\zeta}_1 = -\zeta_1 + 1 + \lambda$. Solving this linear equation yields

$$t_+ = \tau \ln \frac{\lambda + 1}{\lambda + 1 - A}. \tag{4.35}$$

Similarly, during the falling phase, ζ_1 decreases from A to 0, and equation 4.33 becomes $\tau\dot{\zeta}_1 = -\zeta_1 + 1 - \lambda$, with the solution

$$t_- = \tau \ln \frac{\lambda - 1 + A}{\lambda - 1}. \tag{4.36}$$

Thus,

$$T = t_+ + t_- = \tau \ln \frac{(\lambda + 1)(\lambda - 1 + A)}{(\lambda - 1)(\lambda + 1 - A)}. \tag{4.37}$$

Next we will determine the amplitude A. We always have

$$0 < A < 1 + \lambda \tag{4.38}$$

because the amplitude of an oscillation is positive by definition and the denominator in the logarithm in equation 4.35 must be positive. As illustrated in Figure 4B, we observe that

$$t_+ = (n - 1)t_- \tag{4.39}$$

because during the rising phase of variable ζ_1 with duration t_+, the other variables $\zeta_2, \ldots, \zeta_n$ go through consecutive phases that are equivalent to either the falling phase of ζ_1 or its mirror image, each of duration t_-. Substitution of equations 4.35 and 4.36 into equation 4.39 yields an equation that amplitude A must satisfy:

$$(\lambda + 1 - A)\,(\lambda - 1 + A)^{n-1} = (\lambda + 1)(\lambda - 1)^{n-1}. \tag{4.40}$$

Using this equation, we can rewrite equation 4.37 in the following equivalent form:

$$T = n\tau \ln \frac{\lambda - 1 + A}{\lambda - 1}. \tag{4.41}$$

Equation 4.40 is a polynomial equation of degree n in A. It always has a trivial solution $A = 0$, which should be discarded. Eliminating the factor A from equation 4.40 reduces it to a polynomial equation of degree $(n-1)$:

$$A^{n-1} + \sum_{k=0}^{n-2} \binom{n-1}{k} \left(2\lambda - n\frac{\lambda+1}{k+1}\right) (\lambda-1)^{n-2-k} A^k = 0. \tag{4.42}$$

When $n = 2$ (conjugate network), equation 4.42 is a linear equation with the solution $A = 2$. Now equation 4.37 becomes

$$T = 2\tau \ln \frac{\lambda+1}{\lambda-1} = 4\tau \tanh^{-1}\frac{1}{\lambda}. \tag{4.43}$$

This result is consistent with equation 4.18 in the limit $k \to \infty$. The shape of the limit cycle is a square defined by $|x| + |y| = 2$ (see Figure 5B, left), regardless of the value of λ ($>\lambda_c = 1$).

When $n = 3$ (cyclic network with three subnets), equation 4.42 is a quadratic equation with the solution

$$A = \frac{1}{2}\left(3 - \lambda + \sqrt{5\lambda^2 + 2\lambda - 3}\right). \tag{4.44}$$

A negative solution is discarded.

When $n = 4$ (cyclic network with four subnets), equation 4.40 is a cubic equation, which is solvable by radicals (Press, Teukolsky, Vetterling, & Flannery, 2007). The solution can be written as

$$A = \frac{1}{3}\left(4 - 2\lambda + 2\frac{2\lambda^2 + \lambda - 1}{C} + C\right), \tag{4.45}$$

where

$$C = \left(19\lambda^3 - 6\lambda^2 - 15\lambda + 10 + 3^{3/2}(\lambda^2 - 1)\sqrt{11\lambda^2 - 12\lambda + 4}\right)^{1/3}. \tag{4.46}$$

Here we have discarded a pair of complex conjugate roots as extraneous solutions. It is noted that sign before the factor $3^{3/2}$ in equation 4.46 can be flipped from $+$ to $-$ without changing the value of A in equation 4.45.

Figure 4B (right) shows the examples for $n = 3$ and 4. Here the memory duration T is obtained by substituting equation 4.44 or 4.45 into equation 4.41.

4.2.3 Asymptotic Properties. In the limit of weak coupling, namely, $\lambda \to \lambda_c = 1$, we always have $A \to 2$ because equation 4.40 reduces to

$(2 - A)A^{n-1} = 0$, whose only nontrivial solution is $A = 2$. Now T in equation 4.41 increases without bound as $\lambda \to 1$, with an asymptotic formula $T \sim n\tau \ln(2/(\lambda - 1))$.

In the limit of strong coupling, namely, $\lambda \to \infty$, we have a heuristic asymptotic formula,

$$A = \alpha\lambda + \beta + \varepsilon, \tag{4.47}$$

where $\lim_{\lambda\to\infty} \varepsilon = 0$ and coefficients α and β are independent of λ and can be determined by

$$\alpha = \lim_{\lambda\to\infty} (A/\lambda), \qquad \beta = \lim_{\lambda\to\infty} (A - \alpha\lambda). \tag{4.48}$$

When $n = 3$, for instance, we obtain from equation 4.44 that $\alpha = (\sqrt{5} - 1)/2$ and $\beta = 3/2 + \sqrt{5}/10$; when $n = 2$, we have $A = 2$, and hence $\alpha = 0$ and $\beta = 2$. In general, as $\lambda \to \infty$, it follows from equation 4.47 and inequality 4.38 that $0 \le \alpha \le 1$. Writing equation 4.40 in an equivalent form,

$$\left(1 - \frac{A}{\lambda + 1}\right)\left(1 + \frac{A}{\lambda - 1}\right)^{n-1} = 1, \tag{4.49}$$

we see that A cannot grow faster than λ because otherwise the left-hand side of this equation would grow indefinitely rather than equaling 1. This observation justifies an asymptotic formula that is linear in λ (see equation 4.47). Substituting equation 4.47 into equation 4.41 yields

$$\lim_{\lambda\to\infty} T = n\tau \ln(1 + \alpha), \tag{4.50}$$

which is the limit of the fastest possible sequential retrieval. This limit is zero when $n = 2$ and greater than zero when $n \ge 3$. This means that there is a top speed limit for cyclic networks with three or more subnets.

Coefficients α and β can be solved directly without using equations 4.48, which require an explicit formula of A. Coefficient α satisfies the equation

$$(1 - \alpha)(1 + \alpha)^{n-1} = 1, \qquad n \ge 2, \tag{4.51}$$

which is derived by substituting equation 4.47 into equation 4.49 and then taking the limit $\lambda \to \infty$. The solution of equation 4.51 is verified to be consistent with the limit in equation 4.48 when A is given explicitly by equation 4.44 for $n = 3$ or equation 4.45 for $n = 4$. As n increases, α approaches 1 because otherwise, the left-hand side of equation 4.51 would exceed 1 due to the explosion of the term $(1 + \alpha)^{n-1}$. Similarly, we find

$$\beta = \frac{n\alpha + (2 - n)\alpha^2}{n\alpha + 2 - n}, \qquad n \ge 3 \tag{4.52}$$

from equations 4.47 and 4.49 by Taylor expansion with small terms in the order of $1/(\lambda - 1)$ and then taking the limit $\lambda \to \infty$ after some algebra.

4.3 Example 3: Delay Network with Step Gain Function and a Single Delay. We start with the reduced equation 3.14 and assume all the asymmetric connections have the same time delay τ_D. Now the delay distribution is a Dirac delta function $D(t) = \delta(t - \tau_D)$, and equation 3.14 reduces to

$$\tau \dot{x}(t) = -x(t) + f(x(t)) - \lambda f(x(t - \tau_D)). \tag{4.53}$$

We assume the gain function is the step function: $f(x) = \text{sign}(x)$. When $\lambda > \lambda_c = 1$, this system oscillates periodically between $-A$ and A, where $A > 0$ is the amplitude of oscillation. Since equation 4.53 is symmetric with respect to sign flipping ($x \to -x$), we only need to consider half of the period with x going from 0 to A and then back to 0. This half-period will be divided into a rising phase ($\dot{x} > 0$) and a falling phase ($\dot{x} < 0$).

Consider the rising phase ($\dot{x} > 0$ with x increasing from 0 to A). At the beginning of this phase, $x = 0$ (just turning from negative to positive). This means that its past value $x(t - \tau_D) < 0$. Thus, equation 4.53 becomes

$$\tau \dot{x}(t) = -x(t) + 1 + \lambda. \tag{4.54}$$

After a duration of τ_D, the value of $x(t - \tau_D)$ turns positive, echoing the beginning of the rising phase. At this moment, the last term of equation 4.53 flips its sign, and the falling phase starts. Thus, the duration of the rising phase is

$$t_+ = \tau_D. \tag{4.55}$$

Solving equation 4.54 with the conditions $x(0) = 0$ and $x(\tau_D) = A$ yields the amplitude:

$$A = (\lambda + 1)(1 - e^{-\tau_D/\tau}). \tag{4.56}$$

Similarly, during the falling phase ($\dot{x} < 0$ with x decreasing from A to 0), equation 4.53 becomes $\tau \dot{x}(t) = -x(t) + 1 - \lambda$. Solving this equation with the conditions $x(0) = A$ and $x(t_-) = 0$ yields the duration of the falling phase:

$$t_- = \tau \ln \frac{\lambda - 1 + A}{\lambda - 1}. \tag{4.57}$$

The predicted memory duration is

$$T = t_+ + t_- = \tau \ln \frac{2\lambda e^{\tau_D/\tau} - \lambda - 1}{\lambda - 1}, \tag{4.58}$$

which follows from equations 4.55 to 4.57 and the identity $t_+ = \tau_D = \tau \ln e^{\tau_D/\tau}$. Figure 4C (right) plots an example of this formula.

Finally, when $\lambda \to 1$, we have $T \to \infty$, with an asymptotic formula $T \sim \tau \ln\big((2e^{\tau_D/\tau} - 2)/(\lambda - 1)\big)$. When $\lambda \to \infty$, we have $T \to \tau \ln(2e^{\tau_D/\tau} - 1) > \tau_D$, which is the top speed limit.

4.4 Example 4: Delay Network with Exponential Distribution of Delays

4.4.1 Equivalence to a Coupled Network. Consider the dynamical equation 3.14 with an exponential delay distribution:

$$D(t) = e^{-t/\tau_D}/\tau_D, \tag{4.59}$$

where τ_D is the average delay. Define a new auxiliary variable,

$$y(t) \equiv \int_0^\infty D(t') f(x(t - t'))\, dt' = \frac{e^{-t/\tau_D}}{\tau_D} \int_{-\infty}^t e^{\zeta/\tau_D} f(x(\zeta))\, d\zeta, \tag{4.60}$$

where the last step is a substation by equation 4.59 using a new dummy variable $\zeta = t - t'$. It follows from equation 4.60 that

$$\frac{dy(t)}{dt} = -\frac{e^{-t/\tau_D}}{\tau_D^2} \int_{-\infty}^t e^{\zeta/\tau_D} f(x(\zeta))\, d\zeta + \frac{e^{-t/\tau_D}}{\tau_D} e^{t/\tau_D} f(x(t)) \tag{4.61}$$

$$= -y(t)/\tau_D + f(x(t))/\tau_D. \tag{4.62}$$

Thus, the dynamical equation 3.14 is equivalent to

$$\tau \dot{x} = -x + f(x) - \lambda y, \tag{4.63}$$

$$\tau_D \dot{y} = -y + f(x), \tag{4.64}$$

where y is defined by equation 4.60 and equation 4.64 comes from equation 4.62. This system may be regarded as a special case ($n = 2$) of the general cyclic network described by equations 4.33 and 4.34, which allow different subnets to have different time constants and coupling strengths.

4.4.2 Exact Solution for Step Gain Function. In the following, we assume a step gain function $f(x) = \text{sign}(x)$ in equations 4.63 and 4.64. This system has a two-dimensional phase plane that contains a solvable limit cycle when $\lambda > \lambda_c = 1$ (see Figure 4C). Since the system is invariant with respect to the transformation $(x, y) \to (-x, -y)$, which is a 180-degree rotation, we need

to consider only half of the phase plane, say, the half with $x > 0$. Now equations 4.63 and 4.64 become

$$\tau\dot{x} = -x + 1 - \lambda y, \tag{4.65}$$

$$\tau_D\dot{y} = -y + 1. \tag{4.66}$$

The memory duration T is equal to half of the period of the limit cycle. From time 0 to time T, the state (x, y) moves from $(0, -A)$ to $(0, A)$ along the limit cycle, where A is the oscillation amplitude of variable y (see Figure 4C, left). Solving equation 4.66 under the conditions $y(0) = -A$ and $y(T) = A$ yields

$$T = \tau_D \ln \frac{1+A}{1-A} = 2\tau_D \tanh^{-1} A \tag{4.67}$$

or, equivalently,

$$A = \tanh(T/2\tau_D). \tag{4.68}$$

We need another equation in order to determine both T and A.

When $\tau_D \neq \tau$, multiplying equation 4.66 by $\alpha = \lambda\tau/(\tau_D - \tau)$ and then adding to equation 4.65, we find $\tau\dot{u} = -u + 1 + \alpha$, where $u = x + \beta y$ with $\beta = \lambda\tau_D/(\tau_D - \tau)$. Solving this equation under the conditions $u(0) = -\beta A$ and $u(T) = \beta A$ yields

$$e^{T/\tau} = \frac{1 + \alpha + \beta A}{1 + \alpha - \beta A}. \tag{4.69}$$

Substituting equation 4.68 into equation 4.69 and expressing the tanh function in terms of e^{T/τ_D}, we obtain, after some algebra, an equation that T must satisfy,

$$e^{T/\tau} - e^{T/\tau_D} + he^{T/\tau}e^{T/\tau_D} - h = 0, \tag{4.70}$$

where

$$h = \frac{(\lambda - 1)(\tau - \tau_D)}{(\lambda - 1)\tau + (\lambda + 1)\tau_D}. \tag{4.71}$$

It is verified that $|h| < 1$ for $\lambda > 1$. Equation 4.69 can be written equivalently as

$$Z - Z^\kappa + hZ^{1+\kappa} - h = 0, \tag{4.72}$$

where $Z = e^{T/\tau}$ and $\kappa = \tau/\tau_D$. If we define $Z = e^{-T/\tau}$, equation 4.72 remains valid. Another form of equation 4.72 is $(1 - hZ)(1 + hZ^\kappa) = 1 - h^2$.

The value of T is determined by equation 4.70 or, equivalently, by equation 4.72. There is always a trivial solution $T = 0$, which should be ignored. To verify that equation 4.70 also has another solution $T > 0$, define function

$$\varphi(T) = e^{T/\tau} - e^{T/\tau_D} + he^{T/\tau}e^{T/\tau_D} - h \tag{4.73}$$

and consider its derivative at $T = 0$:

$$\varphi'(0) = 1/\tau - 1/\tau_D + h(1/\tau + 1/\tau_D) = -2h/(\tau(\lambda - 1)), \tag{4.74}$$

where the last step follows from equation 4.71. Note that $\varphi'(0)$ and h always have opposite signs since we assume $\lambda > \lambda_c = 1$. We start from the trivial root $\varphi(0) = 0$ and consider the value of $\varphi(T)$ as T increases. When T is small, $\varphi(T)$ has the same sign as $\varphi'(0)$, or the opposite sign of h. For large T, $\varphi(T)$ is dominated by the term $he^{T/\tau}e^{T/\tau_D}$, which has the same sign as h. Therefore, $\varphi(T)$ for small enough T and $\varphi(T)$ for large enough T have opposite signs. The continuity of φ implies the existence of another root $\varphi(T) = 0$ for some $T > 0$.

When $\tau_D = \tau$, solving linear equations 4.65 and 4.66 with the conditions $(x(0), y(0)) = (0, -A)$ and $(x(T), y(T)) = (0, A)$ yields an equation for T:

$$e^{2T/\tau} - be^{T/\tau}T/\tau - 1 = 0, \tag{4.75}$$

where

$$b = 2\lambda/(\lambda - 1). \tag{4.76}$$

Equation 4.75 can also be derived from equation 4.70 in the limit $\tau_D \to \tau$ by dividing it by h and then observing that

$$\lim_{\tau_D \to \tau} \frac{e^{T/\tau} - e^{T/\tau_D}}{h} = -be^{T/\tau}T/\tau. \tag{4.77}$$

Equation 4.75 has a trivial solution $T = 0$ and another solution $T > 0$. To see this, let $\psi(T)$ denote the left-hand side of equation 4.75. Since $\psi(0) = 0$ and $\psi'(0) = -2/((\lambda - 1)\tau) < 0$, we have $\psi(T) < 0$ for small enough T. For large enough $T > 0$ we have $\psi(T) > 0$ because the term $e^{2T/\tau}$ dominates. Thus, there must exist another root $\psi(T) = 0$ for some $T > 0$.

We can solve for T at least numerically from equation 4.70 or equation 4.75. Figure 4C shows an example for $\tau = 1$ and $\tau_D = 2$.

5 Time Evolution of the Memory Overlaps

The oscillatory reduced variable as illustrated in Figure 1 essentially reflects the difference between consecutive memory overlaps, but it provides only partial information about the time courses of the memory overlaps themselves. To determine the memory overlaps explicitly, we consider the sum of memory overlaps and show that this sum satisfies an additional reduced dynamical equation, assuming once again that the network state has significant simultaneous overlaps with at most two memory patterns. The extended reduced equations allow us to completely determine the time courses of the memory overlaps in addition to the time courses of the original reduced variables.

5.1 Reduction Rules for the Sum of Memory Overlaps. Based on the compressor vector in equation 3.2, we define a modified compressor vector for the sum of memory overlaps as follows:

$$\mathbf{s}^{\mathrm{T}} = \mathbf{1}^{\mathrm{T}}\mathbf{X}^{\dagger} = \sum_{m=1}^{M} \mathbf{x}_m^{\dagger}, \tag{5.1}$$

where $\mathbf{1}^{\mathrm{T}} = [1, \ldots, 1]$ with M entries. Applying this vector to the network state $\mathbf{x}(t) = \sum_{m=1}^{M} \phi_m(t)\mathbf{x}^m$ as given by equation 3.23, we obtain a new scalar variable:

$$s(t) = \mathbf{s}^{\mathrm{T}}\mathbf{x}(t) = \sum_{m=1}^{M} \phi_m(t). \tag{5.2}$$

This is the sum of all the memory overlaps. For comparison, the original reduced variable in equation 3.1 reads

$$x(t) = \mathbf{c}^{\mathrm{T}}\mathbf{x}(t) = \sum_{m=1}^{M} (-1)^m \phi_m(t). \tag{5.3}$$

The derivation of the reduction rules in section 3.3 is based on the assumption of simultaneous overlaps with no more than two memory patterns. Under this assumption, we can write, without loss of generality, the reduced variable as

$$x(t) = \phi_2(t) - \phi_1(t), \tag{5.4}$$

as given by equation 3.26. Under the same condition, we can write

$$s(t) = \phi_1(t) + \phi_2(t). \tag{5.5}$$

Since we already know how to solve for $x(t)$, all we need now is another independent equation for $s(t)$. Once $x(t)$ and $s(t)$ are determined, so is

$$\phi_1(t) = \frac{s(t) - x(t)}{2}. \tag{5.6}$$

All other memory overlaps $\phi_m(t)$ are simply time-shifted versions of $\phi_1(t)$ (see equation 3.24).

Now we will employ the same argument as in section 3.3 to derive a new set of reduction rules for vector $\mathbf{s}^{\mathrm{T}} = [s_1, \ldots, s_N]$ given by equation 5.1. Define

$$S_{++} = \sum_{i \in I_{++}} s_i, \quad S_{--} = \sum_{i \in I_{--}} s_i, \quad S_{+-} = \sum_{i \in I_{+-}} s_i, \quad S_{-+} = \sum_{i \in I_{-+}} s_i, \tag{5.7}$$

where I_{++}, I_{--}, I_{+-}, and I_{-+} are the same index sets as defined in equations 3.31 to 3.34. Only two among these four coefficients are independent because by similar argument that leads to equations 3.35 and 3.36, we now find

$$S_{++} - S_{--} = 1, \qquad S_{-+} = S_{+-}. \tag{5.8}$$

The actual values of these coefficients depend on the statistics of the memory matrix and can be determined in the large network limit, as shown in appendix B.

Table 2 contains a new set of reduction rules obtained by left-multiplying $\mathbf{s}^{\mathrm{T}}$ with various terms in the original dynamical equations. To derive the new reduction rule 3, note that

$$\mathbf{s}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\dagger} = \mathbf{1}^{\mathrm{T}}\mathbf{X}^{\dagger}\mathbf{X}\mathbf{X}^{\dagger} = \mathbf{1}^{\mathrm{T}}\mathbf{X}^{\dagger} = \mathbf{s}^{\mathrm{T}} \tag{5.9}$$

by equations 5.1 and 2.3. Therefore,

$$\mathbf{s}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\dagger} g(\mathbf{x}) = \mathbf{s}^{\mathrm{T}} g(\mathbf{x}) \tag{5.10}$$

$$= S_{++} g(s) + S_{--} g(-s) + S_{+-} g(x) + S_{-+} g(-x) \tag{5.11}$$

$$= f(s) + (2S_{++} - 1) g_{\mathrm{e}}(s) + 2S_{-+} g_{\mathrm{e}}(x) \tag{5.12}$$

$$\equiv F(s, x), \tag{5.13}$$

where s and x are given by equations 5.5 and 5.4,

$$f(z) = \frac{g(z) - g(-z)}{2} \tag{5.14}$$

Table 2: Reduction Rules for the Sum of Memory Overlaps.

Original Term	Reduced Term	Remark (equation number)
Rule 1: Network or subnet state $\mathbf{x}$	s	Definition (5.2)
Rule 2: Time derivative of state $\frac{d\mathbf{x}}{dt}$	$\frac{ds}{dt}$	Derivative rule
Rule 3: Symmetric drive $\mathbf{X}\mathbf{X}^{\dagger}g(\mathbf{x})$	$F(s, x)$	Within a subnet or network (5.12)
Rule 4: Asymmetric drive $\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x})$	$F(s, x)$	Within a subnet or network for shifted sequence (5.17)
Rule 5: Intersubnet drive $\mathbf{Y}\mathbf{X}^{\dagger}g(\mathbf{x})$	$F(s, x)$	Between subnets
Rule 6: Intersubnet drive $\tilde{\mathbf{Y}}\mathbf{X}^{\dagger}g(\mathbf{x})$	$F(s, x)$	Between subnets for shifted sequence
Rule 7: Delayed asymmetric drive $\int_0^{\infty} D(\tau)\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x}(t-\tau))d\tau$	$\int_0^{\infty} D(\tau)F(s(t-\tau), x(t-\tau))d\tau$	Connections with time delays
Rule 8: Linear combination $\alpha\mathbf{u} + \beta\mathbf{v}$	$\alpha u + \beta v$	Linearity of reduction, assuming $\mathbf{u}$ and $\mathbf{v}$ reduce to u and v, respectively

Note: Function $F(s, x)$ is given by equation 5.12.

is the odd component of the original gain function as used before, and

$$g_e(z) = \frac{g(z) + g(-z)}{2} \tag{5.15}$$

is the even component. Expression 5.11 is analogous to expression 3.30. To go from expression 5.11 to expression 5.12, use equations 5.8 to eliminate S_{--} and S_{+-} and then make the substitutions $g(\pm z) = g_e(z) \pm f(z)$.

To derive reduction rule 4 in Table 2, consider

$$\mathbf{s}^{\mathrm{T}}\tilde{\mathbf{X}}\mathbf{X}^{\dagger} = \mathbf{1}^{\mathrm{T}}\mathbf{X}^{\dagger}\tilde{\mathbf{X}}\mathbf{X}^{\dagger} = \mathbf{1}^{\mathrm{T}}\tilde{\mathbf{I}}\mathbf{X}^{\dagger} = \mathbf{s}^{\mathrm{T}}, \tag{5.16}$$

where the first step is a substitution by equation 5.1, the second step follows from equation 3.41, and the last step follows from the identity $\mathbf{1}^{\mathrm{T}}\tilde{\mathbf{I}} = \mathbf{1}^{\mathrm{T}}$ and equation 5.1. Thus

$$\mathbf{s}^{\mathrm{T}}\tilde{\mathbf{X}}\mathbf{X}^{\dagger}g(\mathbf{x}) = \mathbf{s}^{\mathrm{T}}g(\mathbf{x}) = F(s, x), \tag{5.17}$$

where the last step follows from equations 5.10 to 5.13. This gives reduction rule 4 in Table 2.

The new reduction rules 1, 2, and 8 in Table 2 are analogous to their corresponding original reduction rules in Table 1. However, the new reduction rules 3 to 7 in Table 2 are quite different from their counterparts in Table 1 because the reduced term $F(s, x)$ depends on the new summation variable s,

as well as the original reduced variable x. Notice also that in Table 1, the sign of the reduced term $\pm f(x)$ can be either positive or negative, whereas in Table 2, the sign of the reduced term $F(s, x)$ is always positive.

5.2 Extended Reduced System for the Sums of Memory Overlaps. Applying the reduction rules in Table 2 to equations 2.12 and 2.13 of the conjugate network yields the reduced system:

$$\tau \dot{s} = -s + F(s, x) + \lambda F(u, y), \tag{5.18}$$

$$\tau \dot{u} = -u + F(u, y) + \lambda F(s, x), \tag{5.19}$$

where s and u are the sums of memory overlaps in the two subnets, while x and y are the original reduced variables governed by equations 3.15 and 3.16. Applying the reduction rules to equation 2.11 of the delay network yields

$$\tau \dot{s}(t) = -s(t) + F(s(t), x(t)) + \lambda \int_0^\infty D(t') F(s(t - t'), x(t - t'))\, dt', \tag{5.20}$$

where s is the sum of memory overlaps and x is the original reduced variable governed by equation 3.14.

These equations imply that in general, the time evolution equation of the sum of memory overlaps is not self-contained because of its dependence on the original reduced variables, which obey their own dynamical equations. By contrast, the dynamics of the original reduced variables is self-sufficient and independent of the dynamics of the sum of memory overlaps. To completely determine the time evolution of the memory overlaps, we need to combine the new equations for the sums of memory overlaps with the original reduced equations.

Nevertheless, the memory overlap dynamics can become self-contained in several special situations. When the original gain function g is an odd function, its even component g_e vanishes, so that expression 5.12 reduces to

$$F(s, x) = f(s), \tag{5.21}$$

which no longer depends on x. Another special case is $p = 1/2$, which leads to $S_{++} = 1/2$ and $S_{-+} = 0$ by equations B.5 and B.6, so that expression 5.12 also reduces to equation 5.21. Finally, when $M = 2$, we have $S_{-+} = 0$ by equation B.6, so that expression 5.12 becomes $F(s, x) = f(s) + (2S_{++} - 1)g_e(s)$, which is independent of x regardless of the value of S_{++}.

In the analysis below, we assume that equation 5.21 holds. Now equations 5.18 and 5.19 read

$$\tau \dot{s} = -s + f(s) + \lambda f(u), \tag{5.22}$$

$$\tau \dot{u} = -u + f(u) + \lambda f(s). \tag{5.23}$$

They are almost identical to equations 3.15 and 3.16 of the original reduced system, except that the sign of the last term in equation 5.22 is positive rather than negative. These equations apply to the example in Figure 3 because the clipped linear gain function in equation 4.7 is an odd function.

5.3 An Explicit Solution of Memory Overlaps. In this section we solve for the memory overlaps from equations 5.22 and 5.23. This system has simple dynamics that allows only point attractors, in contrast to the corresponding original reduced system, which allows a limit cycle.

Due to the symmetry of their nullclines, equations 5.22 and 5.23 permit only an equilibrium state that lies on the diagonal line $s = u$ in the phase plane. Figure 5C shows an example with a clipped linear gain function f given by equation 4.7.

The nullcline for $\dot{u} = 0$ (see Figure 5C, right) looks identical to the nullcline for $\dot{y} = 0$ (see Figure 5C, left) because equations 3.16 and 5.23 are equivalent. This nullcline is piecewise linear, intersecting lines $x = \pm k^{-1}$ at points $(\pm k^{-1}, \pm(1 + \lambda))$ and lines $y = \pm k^{-1}$ at points $(\mp(\lambda_c - \lambda_c^2)/\lambda, \pm k^{-1})$. The nullclines for $\dot{s} = 0$ and $\dot{u} = 0$ are symmetric with respect to the diagonal line $u = s$.

Thus, the equilibrium state s^* of equation 5.22 satisfies the algebraic equation:

$$0 = -s^* + (1 + \lambda) f(s^*). \tag{5.24}$$

After $s(t)$ reaches its the equilibrium state, we have

$$\phi_1(t) + \phi_2(t) = s^* \tag{5.25}$$

by equation 5.5. Combining this result with equation 5.4, we get the explicit solutions:

$$\phi_1(t) = \frac{s^* - x(t)}{2}, \qquad \phi_2(t) = \frac{s^* + x(t)}{2}. \tag{5.26}$$

Therefore, the time profile of the memory overlap $\phi_m(t)$ is a scaled and shifted version of the time profile of the original reduced variable $x(t)$.

This explains the similarity between the reduced variable and the memory overlaps in Figures 1 and 3.

Since the overlaps with different memory patterns are time-shifted versions of one another (see equation 3.24), substituting equations 5.26 into $\phi_2(t-T) = \phi_1(t)$ yields

$$x(t-T) = -x(t). \tag{5.27}$$

This result is consistent with the symmetry of the reduced system with respect to sign flipping $x \to -x$ (see section 6.2). It is also consistent with the fact that the reduced variable has period $2T$, that is,

$$x(t-2T) = -x(t-T) = x(t). \tag{5.28}$$

These properties are obvious in Figures 1 and 3.

Under the assumption of significant overlap with at most two memory patterns, equation 5.25 implies that the sum of all memory overlaps should reach the same constant,

$$s(t) = \sum_{m=1}^{M} \phi_m(t) = s^*, \tag{5.29}$$

after the system settles. This prediction is consistent with the numerical simulation in Figure 3, where the empirical plateau value of $s(t)$ is $1.699998(\pm 9)$ for $M = 2$, and $1.703(\pm 1)$ for $M = 10$. For comparison, the predicted theoretical value is $s^* = 1 + \lambda = 1.7$, where

$$s^* = 1 + \lambda \tag{5.30}$$

is a solution to equation 5.24 when f is the clipped linear gain function. Now equation 5.24 also admits a trivial solution $s^* = 0$ and an extraneous negative solution $s^* = -1 - \lambda$. The stationary state $(0, 0)$ is a saddle and therefore unstable, while the stationary states $(1 + \lambda, 1 + \lambda)$ and $(-1 - \lambda, -1 - \lambda)$ are both stable sinks (Figure 5C, right).

The time delay equation 5.20 also has a stationary solution $s(t) = s^*$ that satisfies equation 5.24, assuming that condition 5.21 holds. This is easily verified by plugging $s(t) = s^*$ into equation 5.20 and noting that $\int_0^\infty D(t')dt' = 1$.

Finally, when the gain function g is not an odd function, equation 5.21 no longer holds. Then we have to solve for the sum $s(t)$ from equations 5.18 and 5.19, which now depend on variables x and y. This means that the sum $s(t)$ will be oscillatory instead of approaching a constant whenever variables x and y are oscillatory. This prediction is also confirmed by numerical simulation (data not shown).

6 Summary and Extensions

6.1 How the Reduced System Is Related to the Original Network. The reduced variable is defined in such a manner that if the original network can retrieve a sequence of memory patterns, the reduced variable must exhibit a periodic oscillation (see Figure 1). A logically equivalent statement is that whenever a reduced system does not oscillate periodically, one can predict that the corresponding original network cannot retrieve a sequence of memory patterns.

All the reduced systems considered in this article have parameter values that prohibit periodic oscillations. An extreme case is the simple network described by equation 2.7. Its reduced equation 3.13 is one-dimensional and therefore cannot oscillate at all, regardless of the choice of parameters. All other reduced systems studied in this article approach some fixed point attractor when the asymmetric coupling strength λ is below a certain critical value λ_c. Numerical simulations confirmed that when $\lambda < \lambda_c$, the original networks indeed settled into a stationary state rather than followed a sequence of memory patterns (see Figures 2, 4, and 6).

When a reduced system oscillates periodically, what does it tell us about the original network? First, based on how a reduced variable is constructed, we expect that the original network should be able to retrieve a sequence of memory patterns robustly, and moreover, the duration of each memory should be equal to half of the period of the oscillation in the reduced system (see Figure 1). This prediction has been confirmed quantitatively in several models (see Figure 4, right panels, and Figure 6). In all of these cases, sequential memory retrievals occurred when the asymmetric coupling strength was strong enough ($\lambda > \lambda_c$) and the retrievals tended to be faster for greater λ, as predicted by the reduced systems.

The reduced system also tells us about the shape of the time profile of the overlap between the network state and a stored memory pattern. When the gain function in the original network is an odd function, as is the case in Figure 3, the reduced variable has the same time profile as the oscillations of the individual units, up to a sign flip, if there are only two memory patterns (see Figure 3, left). When more memory patterns are stored, different individual units may follow different individual sequences, but the stereotypical manner of how an individual unit changes its state shows the same basic time profile (see Figure 3, right). When the gain function is not an odd function, the overlap time profile can be determined if we extend the reduced system by including additional equations for the sum of memory overlaps (see section 5).

The reduced system method yields good approximations to the behaviors of the original networks with diverse architectures and a wide range of parameter values (see Figures 4 and 6). Since the reduction rules have been derived under the assumption of a large network, the correspondence between a reduced system and the original network is statistically in nature.

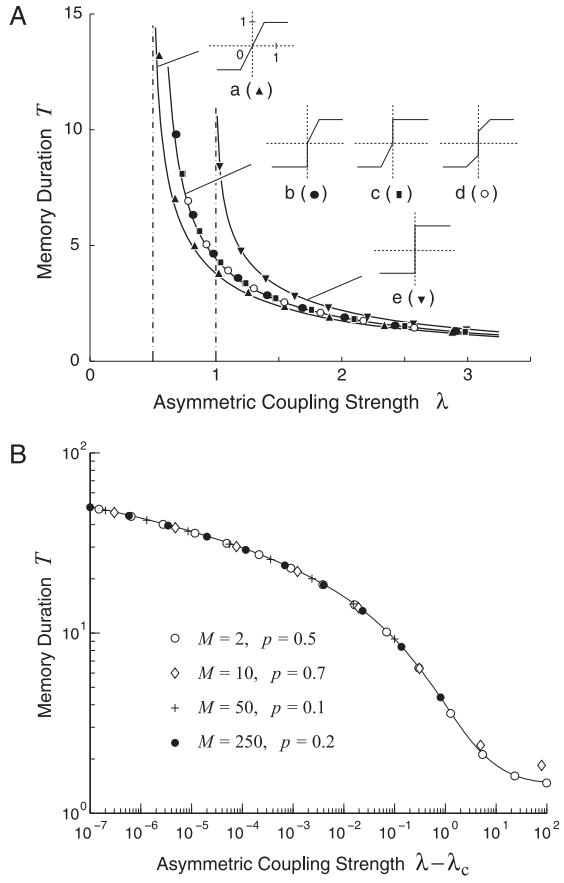

Figure 6: Different original networks that can be compressed into the same reduced system have similar global properties. (A) Using different gain functions with an identical odd component in an original network leads to the same reduced system. This equivalence is confirmed by numerical simulations of a conjugate network with various gain functions (insets). Gain functions a and e are odd functions considered in section 4.1, whereas gain functions b, c, and d have an identical odd component, which is the same as d itself. The memory duration data from the last three cases fall on the same theoretical curve obtained numerically from the reduced system. Each subnet of the conjugate network has $N = 5000$ units, and $M = 8$ memory patterns with equal on and off probability. The vertical dashed lines indicate the critical coupling strength ($\lambda_c = 1/2$ or 1). (B) The memory durations of various cyclic network with three subnets all follow the same theoretical curve as predicted by equations 4.37 and 4.44, regardless of the total number of memory patterns (M), and the probability (p) for the units to be on in these memory patterns. Each subnet has $N = 1000$ units, and the gain function is the step function. The critical coupling strength $\lambda_c = 1$.

As expected, numerical simulations confirm that a larger network tends to follow more closely the predictions by its reduced system.

All the reduced systems analyzed in this article (see section 4) predict a minimum required coupling strength λ_c for sequential memory retrievals, which is consistent with the behaviors of the original networks. When λ decreases toward λ_c, all the reduced systems predict that the memory duration should increase indefinitely without bound. In simulations, the original networks initially followed the predictions and, of course, failed eventually when λ was too close to λ_c. The empirical values of λ_c were much closer to the theoretical predictions when the pseudoinverse (see equation 2.2) rather than the transpose approximation (see equation 2.6) was used in the weight matrix. In the latter case, the empirical values of λ_c tended to be slightly lower than the predictions.

The main limitation of the reduction method is its inability to predict a maximum coupling strength for sequential memory transitions. All of the reduced systems analyzed in section 4 keep on oscillating as the coupling strength increases indefinitely ($\lambda \to \infty$). In numerical simulations, if λ is too large, the original network may get stuck in a fixed point or show other oscillatory behaviors rather than follow the desired memory sequence. For instance, in Figure 4A, with $M = 10$ memory patterns, the conjugate network could not be driven beyond $\lambda \sim 1.8$. The same was true in Figure 6B for the network with $M = 250$. The delay networks in Figure 4C were even more prone to failure for large λ. The problem with large λ is expected in the sense that the derivation of the reduced system is based on the assumption of relatively slow state transition, which translates into a relatively small λ. All of these discrepancies depend on the size of the network and the number of memory patterns embedded, factors that have not been taken into account by the reduced system in the present form.

6.2 Symmetry and Equivalence. The reduction rules in Table 1 imply that the reduced system involves only the odd component f of the original gain function g, but not g itself. Since f is always an odd function

$$f(-x) = -f(x), \tag{6.1}$$

the reduced system is invariant under sign flipping of all its variables (e.g., $x \to -x$). As a consequence, the oscillation of a reduced variable should have positive and negative segments that are mirror images of each other (see Figures 1 and 3). Equations 5.27 and 5.28 reflect this symmetry with respect to sign flip. When the original gain function itself is not an odd function, namely, $g(-x) \neq -g(x)$, flipping the states of all the units in the original network can affect the outputs of these units but not the macroscopic behavior captured by the reduced system. In fact, the raw state of an individual unit (before applying the gain function) is also symmetric with

respect to sign flip even if the original gain function g is not an odd function, as explained following equation 3.25. Numerical simulation confirmed that the memory duration T indeed depended on only the odd component of g (see Figure 6A).

From a given neural network, we may derive more than one reduced system, depending on how we define the reduced variables. For example, if we choose to define the sign vector $\boldsymbol{\omega}^{\mathrm{T}}$ as $[1, -1, \ldots]$ instead of $[-1, 1, \ldots]$ in equation 3.3, then the compressor vector $\mathbf{c}^{\mathrm{T}} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger}$ in 3.2 would change its sign, and so would the reduced variable $x = \mathbf{c}^{\mathrm{T}}\mathbf{x}$. The final effect is equivalent to a sign flip $x \to -x$ in the final reduced system. For another example, in a conjugate network we may call the reduced variable from the first subnet as y, and that from the second subnet as x, or vice versa. Since these different reduced systems correspond to the same original network, they are equivalent and should have an identical period of oscillation and condition of stability, and so on. The difference is essentially a change of the coordinate system.

To elaborate on the idea of equivalence, let us directly change the coordinate system of the conjugate equations 3.15 and 3.16, replacing (x, y) by $(\pm x, \pm y)$ or $(\pm y, \pm x)$, with all possible combinations of signs. Transformations that map (x, y) to $(-y, x)$, $(-x, -y)$ or $(y, -x)$ are rotations in multiple of 90 degrees, and they all leave equations 3.15 and 3.16 unchanged. Reflections map (x, y) to (y, x), $(-y, -x)$, $(x, -y)$, or $(-x, y)$, and they all transform equations 3.15 and 3.16 into

$$\tau\dot{x} = -x + f(x) + \lambda f(y), \tag{6.2}$$

$$\tau\dot{y} = -y + f(y) - \lambda f(x). \tag{6.3}$$

This system is equivalent to the original reduced system, but with switched x and y variables.

Similarly, for the cyclic network with three subnets, we may replace (x, y, z) by any permutation of the three variables with arbitrary sign flipping. For example, $(x, y, z) \to (x, -y, z)$ transforms equations 4.28 to 4.30 into an equivalent but symmetric form:

$$\tau\dot{x} = -x + f(x) - \lambda f(y), \tag{6.4}$$

$$\tau\dot{y} = -y + f(y) - \lambda f(z), \tag{6.5}$$

$$\tau\dot{z} = -z + f(z) - \lambda f(x). \tag{6.6}$$

This symmetry may help analyze the system. For instance, in the special case $\lambda = 1$, adding up the three equations yields $\tau d(x + y + z)/dt = -(x + y + z)$, which implies $x + y + z \to 0$ as $t \to \infty$ for all initial state. Thus, any stable states of the reduced system, including a limit cycle, can exist only on

the plane given by $x + y + z = 0$. The reduced system for the cyclic network with three subnets has eight equivalent forms. In this article, we need to consider only a single form of reduced system for all practical calculations.

Finally, note that the extended reduced system for the sum of memory overlaps depends in general on both the odd component and the even component of the original gain function (see section 5), in contrast to the basic reduced system, which depends on only the odd component of the original gain function. Therefore, the effect of sign flipping becomes more complicated in the extended reduced system.

6.3 Complex Sequences. The reduced system captures the stereotyped manner of sequential memory transition between two consecutive memory patterns. The reduced system may stay the same regardless of the global structure of the stored temporal sequences, which might be linear, cyclic, or part of a more complex arrangement with converging branches. For concrete examples, let

$$\mathbf{X} = \left[\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4, \mathbf{x}^5, \mathbf{x}^6, \mathbf{x}^7, \mathbf{x}^8, \mathbf{x}^9, \mathbf{x}^{10}\right] \tag{6.7}$$

be a matrix of linearly independent memory patterns, and let its shifted version be given by

$$\tilde{\mathbf{X}} = \left[\mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4, \mathbf{x}^1, \mathbf{x}^6, \mathbf{x}^7, \mathbf{x}^8, \mathbf{x}^9, \mathbf{x}^{10}, \mathbf{x}^7\right]. \tag{6.8}$$

The weight matrix $\mathbf{W} = \tilde{\mathbf{X}}\mathbf{X}^{\dagger}$ specifies two separate groups of sequences as follows:

$$\begin{array}{llllllll} \mathbf{x^1} & \rightarrow & \mathbf{x}^2 & & \mathbf{x^5} & \rightarrow & \mathbf{x}^6 \rightarrow \mathbf{x}^7 & \rightarrow \mathbf{x}^8 \\ \uparrow & & \downarrow & & & & \uparrow \quad \downarrow & \\ \mathbf{x}^4 & \leftarrow & \mathbf{x}^3 & & & & \mathbf{x}^{10} \leftarrow \mathbf{x}^9 & \end{array}. \tag{6.9}$$

The transition between any two neighboring memory patterns has the same dynamics. A reduced variable can describe this stereotyped dynamics while ignoring everything else.

Although the networks considered in this article cannot directly embed a sequence with repeated memory patterns, such as

$$\mathbf{a} \rightarrow \mathbf{b} \rightarrow \mathbf{a} \rightarrow \mathbf{a} \rightarrow \mathbf{c} \rightarrow \cdots, \tag{6.10}$$

one can get around this problem by adding hidden units to the network storing an augmented sequence as follows:

$$\begin{bmatrix} \mathbf{a} \\ \mathbf{u}^1 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{b} \\ \mathbf{u}^2 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{a} \\ \mathbf{u}^3 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{a} \\ \mathbf{u}^4 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{c} \\ \mathbf{u}^5 \end{bmatrix} \rightarrow \cdots \tag{6.11}$$

The augmented memory patterns shown above are all distinct and linearly independent as long as patterns $\mathbf{u}^1, \mathbf{u}^2, \ldots$ for the hidden units are linearly independent (Kühn & van Hemmen, 1995).

6.4 How to Learn the Pseudoinverse of Memory Patterns. Consider a Hopfield network with N units and M memory patterns: $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^M]$, where each column $\mathbf{x}^m = [x_1^m, \ldots, x_N^m]^{\mathrm{T}}$ is a memory pattern. The weight matrix

$$\mathbf{W} = \frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{T}} \quad \text{or} \quad w_{ij} = \frac{1}{N}\sum_{m=1}^{M} x_i^m x_j^m \tag{6.12}$$

can be learned by a simple Hebb rule of the form $\Delta w_{ij} \propto x_i^m x_j^m$. This learning rule is local in the sense that w_{ij} depends on only the activities of units i and j, but not any other units.

More generally, a weight matrix of the form $\mathbf{W} = \mathbf{X}\mathbf{X}^{\dagger}$ cannot be learned locally because w_{ij} depends on the activities of not only units i and j, but also all other units throughout the network. However, as shown in appendix A, the pseudoinverse $\mathbf{X}^{\dagger}$ has an approximation given by equation A.9 for large N. We can use this approximation and write the weight matrix as

$$\mathbf{W} = \mathbf{X}\mathbf{X}^{\dagger} \approx a\mathbf{X}\mathbf{X}^{\mathrm{T}} - b\,(\mathbf{X}\mathbf{1})\,(\mathbf{X}\mathbf{1})^{\mathrm{T}} \tag{6.13}$$

or, equivalently,

$$w_{ij} = a\sum_{m=1}^{M} x_i^m x_j^m - b\left(\sum_{m=1}^{M} x_i^m\right)\left(\sum_{m=1}^{M} x_j^m\right), \tag{6.14}$$

where $\mathbf{1} = [1, \ldots, 1]^{\mathrm{T}}$ with M entries, $a = 1/(N\sigma^2)$ and $b = a\mu^2/(\sigma^2 + \mu^2 M)$. Here μ is the mean and σ^2 is the variance of the memory states x_i^m. In the special case where the memory states have equal on and off probability ($p = 1/2$), we have $\mu = 0$ and $\sigma^2 = 1$ by equations A.10 and A.11, so that $a = 1/N$ and $b = 0$, reducing equation 6.13 to equation 6.12.

The weight w_{ij} in equation 6.14 is local since it depends on only the activities of units i and j. Equation 6.14 may be interpreted loosely as a modified Hebb rule, which learns all the memory patterns as a batch and subtracts out a background of total activity (or average activity) across all the memory patterns. The formula actually resembles the model in Sejnowski (1977). Similarly, for weight matrix of the form $\mathbf{W} = \tilde{\mathbf{X}}\mathbf{X}^{\dagger}$, we need to replace

x_i^m by x_i^{m+1} in equation 6.14. For weight between different subnets, such as $\mathbf{W} = \mathbf{Y}\mathbf{X}^\dagger$, we just need to replace x_i^m by y_i^m. In all these cases, the weight matrix remains local.

Numerical simulations confirmed that weight matrices based on the pseudoinverse approximation as considered above could indeed store and retrieve a temporal sequence in a cyclic network, in a manner as predicted by the reduced system, although the errors tended to be larger than that when exact pseudoinverses were used.

7 Discussion

The dynamics reduction method developed here provides a concise characterization of the transient nonlinear dynamics of a class of asymmetric networks during sequential memory retrievals. On the one hand, the dynamics of the reduced system is completely self-contained and formally independent of the instantaneous state of the original network. Thus, a reduced system can be analyzed on its own without referring to the original network. On the other hand, the two systems are linked by a compression procedure such that a periodic oscillation in the reduced system corresponds to spontaneous sequential memory retrievals in the original network (see Figures 1 and 2). This correspondence allows the reduced system to predict the stability and the speed of sequential retrievals in the original network. The reduction method can be summarized by a few general reduction rules (see Table 1), which may apply to arbitrary Hopfield-type networks that may contain sparse memory patterns, asymmetric connections, time delays, and coupled subnets.

Numerical simulations have confirmed the correspondence between the behaviors of an original network and the predictions of its reduced system. For example, the reduced system can correctly predict the existence of a critical coupling strength for sequential memory retrievals, the quantitative relationship between the memory duration and the coupling strength (see Figures 4 and 6B), and the odd symmetry of the effective gain function (see Figure 6A).

The reduction method has been generalized to derive an extended set of reduction rules for the sum of memory overlaps without sign flipping (see Table 2). The extended reduced system can be used along with the basic reduced system to fully determine the time profiles of the overlaps between the network state and the stored memory patterns in a sequence (see section 5).

Simulations have also revealed limitations of the reduced system method. Memory sequence retrieval in an original network may fail when the asymmetric coupling strength λ is either too large or too small. The reduced system can correctly predict the minimal coupling strength (λ_c) required for sequential retrieval, with extremely high accuracy when exact

pseudoinverse is used in the weights of the original network. However, as λ increases, the cyclic network with three or more subnets often tends to be more stable than the conjugate network, which in turn tends to be more stable than the time delay networks. Since the reduced systems can oscillate perfectly in all these cases as $\lambda \to \infty$, they do not provide complete information about the stability of the original networks for large λ. Thus, the main shortcoming of the reduced system is that it cannot predict the failure of sequential retrievals when λ is too large (see section 6.1). Because the key reduction rules are based on asymptotic properties of a sufficiently large network with fixed number of memory patterns (see appendix A), all finer details of the behaviors of the original network, including any effects related to the actual network size and the memory load, are ignored in the reduced system.

Although the method developed here is based on associate memories in Hopfield-type networks, it might generalize to other situations as well. For example, it should be straightforward to extend the method from continuous dynamics as used in this article to stochastic dynamics. In general, given any regular sequential state transitions in a network, one may always construct an oscillatory variable like that in Figure 1. The real problem is whether a self-contained system can be derived, at least approximately. Another possible extension of the method is in the area of temporal sequence recognition. For example, in a subnet of a cyclic network with sparse memory patterns, one may set the asymmetric connections to be too weak to drive spontaneous retrieval by themselves unless assisted by an input pattern that matches the next memory pattern in the stored sequence. In the reduced system, this process should correspond to jumping between point attractors under external driving force.

How the brain handles temporal sequences is largely unknown, and recent theoretical and experimental results point in several interesting directions (see, e.g., Diesmann, Gewaltig, & Aertsen, 1999; Hopfield & Brody, 2001; Hahnloser, Kozhevnikov, & Fee, 2002; Maass, Natschläger, & Markram, 2002; Ikegaya et al., 2004; Mauk & Buonomano, 2004; Jin, Fujii, & Graybiel, 2009). The hippocampal system is another widely used model system in the study of attractor dynamics, oscillations, and temporal sequences (McNaughton et al., 2006; Buzsáki, 2006; Burgess, 2007; Hasselmo, 2012; Knierim & Zhang, 2012). The networks studied in this article, especially the coupled networks, take very little to implement in biologically plausible hardwares. The connection weight matrix based on pseudoinverse of memory patterns can potentially be learned by local learning rules (see section 6.4). To control the speed of sequential retrieval, adjustment of the coupling strength λ could be implemented as multiplicative modulation of the gain function $g(x)$ because in the original network, what matters is the product $\lambda g(x)$. For various simplified network models, the reduced system method developed here may provide a few convenient tools for further exploration and analysis.

Appendix A: Asymptotic Properties of the Compressor Vector

In this appendix, we show that the compressor vector based on the pseudoinverse of binary memory patterns has some statistical regularities, which are used for deriving the reduction rules in section 3.3. We will derive equations A.21 and A.22 below for C_{++} and C_{-+} only, because C_{--} and C_{+-} can be obtained from equations 3.35 and 3.36.

A.1 Approximation to the Pseudoinverse of Memory Matrix. Let $\mathbf{X} = \left[\mathbf{x}^1, \ldots, \mathbf{x}^M\right]$ be M random memory patterns, where each memory pattern is a column vector $\mathbf{x}^m = [x_1^m, \ldots, x_N^m]^{\mathrm{T}}$ with N entries. We assume that different entries of matrix $\mathbf{X}$ are drawn independently from the same probability distribution. We start with the general case and do not require the memory patterns to be binary for the time being. The memory state x_i^m for any given m and i has the same statistics. Let

$$\mu = \left\langle x_i^m \right\rangle \tag{A.1}$$

be the mean and

$$\sigma^2 = \left\langle \left(x_i^m - \mu\right)^2 \right\rangle = \left\langle \left(x_i^m\right)^2 \right\rangle - \mu^2 \tag{A.2}$$

be the variance. Now the second moment is

$$\left\langle \left(x_i^m\right)^2 \right\rangle = \mu^2 + \sigma^2. \tag{A.3}$$

Consider matrix $\frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X}$. Its entry in the mth row and the nth column is

$$\left\{\frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X}\right\}_{m,n} = \frac{1}{N}\sum_{i=1}^{N} x_i^m x_i^n \to \begin{cases} \mu^2 + \sigma^2 & \text{if } m = n \\ \mu^2 & \text{if } m \neq n \end{cases} \tag{A.4}$$

in the limit of large network ($N \to \infty$). When $m = n$, the result is the same as the second moment in equation A.3; when $m \neq n$, we have two independent random variables x_i^m and x_i^n, so that the average of their product is equal to the product of their averages. In other words,

$$\lim_{N\to\infty} \frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X} = \sigma^2\mathbf{I} + \mu^2\mathbf{1}\mathbf{1}^{\mathrm{T}}, \tag{A.5}$$

where $\mathbf{I}$ is the $M \times M$ identity matrix and $\mathbf{1} = [1, \ldots, 1]^{\mathrm{T}}$ with M entries.

For finite N, equation A.5 holds approximately, and we write

$$\frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X} = \sigma^2\mathbf{I} + \mu^2\mathbf{1}\mathbf{1}^{\mathrm{T}} + O\left(N^{-1/2}\right), \tag{A.6}$$

where the term $O(N^{-1/2})$ means that each entry of the matrix should include an additional error term, which approaches 0 at the rate of $N^{-1/2}$ as $N \to \infty$. Using the identity, $(c\mathbf{I} + \mathbf{a}\mathbf{b}^{\mathrm{T}})^{-1} = \mathbf{I}/c - \mathbf{a}\mathbf{b}^{\mathrm{T}}/(c^2 + c\mathbf{b}^{\mathrm{T}}\mathbf{a})$ for arbitrary scalar c and column vectors $\mathbf{a}$ and $\mathbf{b}$, we obtain the inverse,

$$\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1} = \frac{1}{N\sigma^2}\left(\mathbf{I} - q\mathbf{1}\mathbf{1}^{\mathrm{T}}\right) + O\left(N^{-3/2}\right), \tag{A.7}$$

where

$$q = \frac{\mu^2}{\sigma^2 + \mu^2 M}. \tag{A.8}$$

This leads to an approximation to the pseudoinverse:

$$\mathbf{X}^{\dagger} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}} = \frac{1}{N\sigma^2}\left(\mathbf{X}^{\mathrm{T}} - q\mathbf{1}(\mathbf{X}\mathbf{1})^{\mathrm{T}}\right) + O\left(N^{-3/2}\right). \tag{A.9}$$

In the special case of binary memory patterns, suppose the entries of matrix $\mathbf{X}$ are drawn independently, taking the value $+1$ with probability p and the value -1 with probability $1 - p$. Now the mean and variance of each memory state x_i^m are given by

$$\mu = \left\langle x_i^m \right\rangle = 2p - 1, \tag{A.10}$$

$$\sigma^2 = \left\langle \left(x_i^m\right)^2 \right\rangle - \mu^2 = 4p(1-p). \tag{A.11}$$

The approximate formula A.9 is readily applicable to the present case. In particular, when $p = 1/2$, we find $\mu = 0$, $\sigma^2 = 1$ and $q = 0$, so that equation A.9 reduces to equation 2.6.

Figure 7A shows that equation A.9 provides a reasonably good approximation to the exact pseudoinverse of binary memory matrix. The entries in the approximate pseudoinverse matrix $\mathbf{X}^{\dagger} \approx \frac{1}{N\sigma^2}\left(\mathbf{X}^{\mathrm{T}} - q\mathbf{1}(\mathbf{X}\mathbf{1})^{\mathrm{T}}\right)$ by equation A.9 may have at most $M + 3$ distinct values, whereas the values of the entries of the exact pseudoinverse $\mathbf{X}^{\dagger} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}$ are scattered more smoothly and, for large N, show multiple peaks at the corresponding discrete values specified by the approximation (see Figure 7A).

A.2 Statistics of the Compressor Vector. By substituting equation A.9 into equation 3.2, the compressor vector now reads

$$\mathbf{c}^{\mathrm{T}} = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{X}^{\dagger} = \frac{1}{N\sigma^2}\left((\mathbf{X}\boldsymbol{\omega})^{\mathrm{T}} - q\left(\boldsymbol{\omega}^{\mathrm{T}}\mathbf{1}\right)(\mathbf{X}\mathbf{1})^{\mathrm{T}}\right) + O\left(N^{-3/2}\right), \tag{A.12}$$

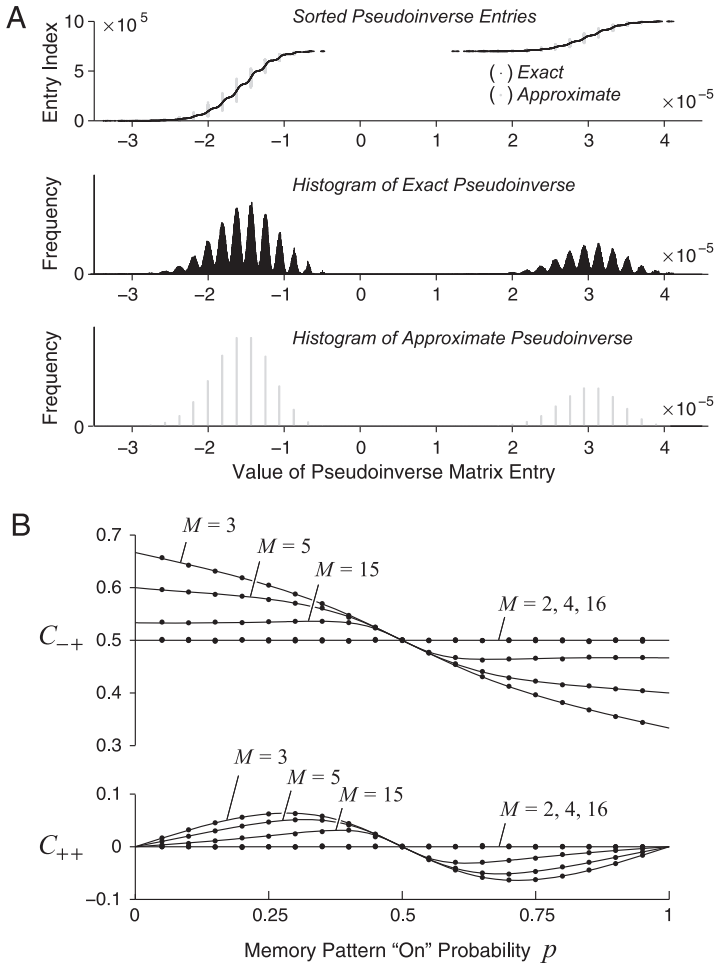

Figure 7: The pseudoinverse of the memory pattern matrix of a large network has statistical regularities that make it possible to derive the reduction rules. (A) Pseudoinverse approximation by equation A.9 is compared with the exact pseudoinverse, showing that the approximation holds up to the level of individual matrix entries. Top: All the entries of the exact pseudoinverse matrix are sorted and displayed (black dots) and then compared against the corresponding entries of the approximation (gray dots). Any two dots with the same vertical coordinate ("Entry Index") come from the same location in the two matrices. Middle and bottom: Histogram of all the entries in the exact or the approximate pseudoinverse matrix. The approximate case shows discrete clusters. Frequency in arbitrary units. Parameters: $M = 20$, $N = 50{,}000$, and $p = 0.3$. (B) Statistical properties of the compressor vector based on the pseudoinverse, as predicted by equations A.21 and A.22 (curves), are confirmed by simulations ($N = 1000$ and each data point is the average of 1000 random repeats).

where the term $O(N^{-3/2})$ means that each entry of the vector includes an additional error term that vanishes at the rate of $N^{-3/2}$ as $N \to \infty$. Here $\boldsymbol{\omega}^{\mathrm{T}}\mathbf{1}$ is the sum of all entries of the vector $\boldsymbol{\omega}^{\mathrm{T}} = [-1, 1, \ldots]$ in equation 3.3, so that

$$\boldsymbol{\omega}^{\mathrm{T}}\mathbf{1} = \begin{cases} 0 & \text{if } M \text{ is even,} \\ -1 & \text{if } M \text{ is odd.} \end{cases} \tag{A.13}$$

When M is an even number, equation A.12 reduces to

$$\mathbf{c}^{\mathrm{T}} = \frac{(\mathbf{X}\boldsymbol{\omega})^{\mathrm{T}}}{N\sigma^2} + O(N^{-3/2}), \tag{A.14}$$

where

$$\mathbf{X}\boldsymbol{\omega} = \sum_{m=1}^{M} (-1)^m \mathbf{x}^m = -\mathbf{x}^1 + \mathbf{x}^2 - \mathbf{x}^3 + \cdots + \mathbf{x}^M. \tag{A.15}$$

Now coefficient C_{++} defined by equation 3.31 becomes

$$C_{++} = \sum_{i \in I_{++}} c_i = \frac{O(N^{1/2})}{N\sigma^2} + |I_{++}| O(N^{-3/2}) \to 0 \tag{A.16}$$

as $N \to \infty$, where $|I_{++}|$ is the total number of elements in the set I_{++}. To see why equations A.16 should hold, first note that the contribution by $-\mathbf{x}^1 + \mathbf{x}^2$ from equation A.15 to the sum in equation A.16 is zero, because for each index $i \in I_{++}$, we have $x_i^1 = x_i^2 = 1$ by definition, so that $-x_i^1 + x_i^2 = -1 + 1 = 0$. The term $O(N^{1/2})$ is the total contribution by $-\mathbf{x}^3 + \cdots + \mathbf{x}^M$, because these terms can be arranged in pairs with opposite signs, so that the result should cancel out on average, leaving a residual noise term that should increase as $N^{1/2}$. The term $|I_{++}| O(N^{-3/2})$ comes from the error term in equation A.14, and it should vanish as $N \to \infty$ because $|I_{++}|$ increases linearly with N. By similar analysis,

$$C_{-+} = \sum_{i \in I_{-+}} c_i = \frac{2|I_{-+}| + O(N^{1/2})}{N\sigma^2} + |I_{-+}| O(N^{-3/2}) \to \frac{1}{2} \tag{A.17}$$

as $N \to \infty$. The term $2|I_{-+}|$ is the contribution by $-\mathbf{x}^1 + \mathbf{x}^2$ from equation A.15 because for each index $i \in I_{-+}$, we have $-x_i^1 + x_i^2 = -(-1) + 1 = 2$, and the total number of indices in I_{-+} is $|I_{-+}|$. The term $O(N^{1/2})$ is the total contribution by $-\mathbf{x}^3 + \cdots + \mathbf{x}^M$, whereas the term $|I_{-+}| O(N^{-3/2})$ comes from the error term in equation A.14. The last step in equation A.17 follows from the relation $|I_{-+}|/N \to (1-p)p$ and equation A.11.

When M is an odd number, equation A.12 becomes

$$\mathbf{c}^{\mathrm{T}} = \frac{1}{N\sigma^2}\left((\mathbf{X}\boldsymbol{\omega})^{\mathrm{T}} + q\,(\mathbf{X}\mathbf{1})^{\mathrm{T}}\right), \tag{A.18}$$

where $\mathbf{X}\mathbf{1} = \mathbf{x}^1 + \mathbf{x}^2 + \cdots + \mathbf{x}^M$ and $\mathbf{X}\boldsymbol{\omega} = -\mathbf{x}^1 + \mathbf{x}^2 - \cdots - \mathbf{x}^M$. The last term in $\mathbf{X}\boldsymbol{\omega}$ has a negative sign because M is odd, whereas the remaining $M-1$ terms can be arranged in pairs with opposite signs. Thus, we have

$$C_{++} = \frac{|I_{++}|}{N\sigma^2}\Big(\overbrace{-1+1}^{\text{by } -\mathbf{x}^1+\mathbf{x}^2} + \overbrace{(-\mu)}^{\text{by } -\mathbf{x}^M} + q\Big(\overbrace{1+1}^{\text{by } \mathbf{x}^1+\mathbf{x}^2} + \overbrace{(M-2)\mu}^{\text{by } \mathbf{x}^3+\cdots+\mathbf{x}^M}\Big)\Big) + O\big(N^{-1/2}\big), \tag{A.19}$$

where the contributions by various terms are labeled above the over-braces. Besides the contribution from $\mathbf{x}^1$ and $\mathbf{x}^2$, as considered before for even M, we also need to consider the effect of the unpaired term $-\mathbf{x}^M$. Since x_i^M has the mean value μ, the total contribution by $-\mathbf{x}^M$ should be $|I_{++}|(-\mu)$. By the same argument, the contribution by $\mathbf{x}^3 + \cdots + \mathbf{x}^M$ is $|I_{++}|(M-2)\mu$. The term $O\big(N^{-1/2}\big)$ represents the total effects all the remaining noisy terms. Similarly,

$$C_{-+} = \frac{|I_{-+}|}{N\sigma^2}\Big(\overbrace{-(-1)+1}^{\text{by } -\mathbf{x}^1+\mathbf{x}^2} + \overbrace{(-\mu)}^{\text{by } -\mathbf{x}^M} + q\Big(\overbrace{-1+1}^{\text{by } \mathbf{x}^1+\mathbf{x}^2} + \overbrace{(M-2)\mu}^{\text{by } \mathbf{x}^3+\cdots+\mathbf{x}^M}\Big)\Big) + O\big(N^{-1/2}\big), \tag{A.20}$$

where $|I_{-+}|$ is the total number of indices in the set I_{-+}.

We finally obtain from equations A.16, A.17, A.19, and A.20 that

$$C_{++} = \begin{cases} 0 & \text{if } M \text{ is even,} \\ \dfrac{p(1-p)(1-2p)}{M(2p-1)^2 + 4p(1-p)} & \text{if } M \text{ is odd,} \end{cases} \tag{A.21}$$

$$C_{-+} = \begin{cases} 1/2 & \text{if } M \text{ is even,} \\ \dfrac{M(1-2p)^2 + 2p^2(1-2p) + 1}{2M(2p-1)^2 + 8p(1-p)} & \text{if } M \text{ is odd,} \end{cases} \tag{A.22}$$

in the large network limit $N \to \infty$, using equations A.8, A.10, and A.11, together with the limits $|I_{++}|/N \to p^2$ and $|I_{-+}|/N \to (1-p)p$. When $p = 1/2$, we have $C_{++} = 0$ and $C_{-+} = 1/2$ for any M. For odd M, the curves of C_{++} and C_{-+} are antisymmetric about the point $p = 1/2$, at which point all

curves share the same slope $-1/2$. For odd M, we always have $C_{++} \to 0$ and $C_{-+} \to 1/2$ as $M \to \infty$, regardless of the value of p. The results from equations A.21 and A.22 agree nicely with direct numerical simulations with large N (see Figure 7B).

Finally, in the special case with only two memory patterns ($M = 2$), it is verified that the following equations hold exactly for any given network size N:

$$C_{++} = 0, \quad C_{-+} = \frac{|I_{-+}|}{|I_{-+}| + |I_{+-}|}. \tag{A.23}$$

Since both $|I_{-+}|/N$ and $|I_{+-}|/N$ approach $p(1-p)$ as $N \to \infty$, we have $C_{-+} \to 1/2$, confirming that equations A.23 are consistent with equations A.21 and A.22.

Appendix B: Statistical Properties of the Extended Compressor Vector for Memory Overlap Summation

In this section we evaluate the coefficients S_{++} and S_{-+} defined by equations 5.7. The memory matrix $\mathbf{X}$ is the same as that considered in appendix A. We start with the approximate pseudoinverse in equation A.9,

$$\mathbf{X}^{\dagger} = \frac{1}{N\sigma^2}\left(\mathbf{X}^{\mathrm{T}} - q\mathbf{1}(\mathbf{X}\mathbf{1})^{\mathrm{T}}\right) + O\left(N^{-3/2}\right), \tag{B.1}$$

where q is given by equation A.8 and $\mathbf{1} = [1, \ldots, 1]^{\mathrm{T}}$ with M entries. Now the extended compressor vector defined by equation 5.1 becomes

$$\mathbf{s}^{\mathrm{T}} = \mathbf{1}^{\mathrm{T}}\mathbf{X}^{\dagger} = \frac{1 - qM}{N\sigma^2}(\mathbf{X}\mathbf{1})^{\mathrm{T}} + O\left(N^{-3/2}\right), \tag{B.2}$$

where $\mathbf{X}\mathbf{1} = \mathbf{x}^1 + \cdots + \mathbf{x}^M$. By similar argument as in section A.2 of appendix A, we get

$$S_{++} = \sum_{i \in I_{++}} s_i = \frac{1 - qM}{N\sigma^2}|I_{++}|\Big(\overbrace{1+1}^{\text{by } \mathbf{x}^1 + \mathbf{x}^2} + \overbrace{(M-2)\mu}^{\text{by } \mathbf{x}^3 + \cdots + \mathbf{x}^M} \Big) + O\left(N^{-1/2}\right). \tag{B.3}$$

Unlike the results in appendix A, here the formula is the same regardless of whether number M is even or odd. Similarly,

$$S_{-+} = \sum_{i \in I_{-+}} s_i = \frac{1 - qM}{N\sigma^2} |I_{-+}| \Big(\overbrace{-1+1}^{\text{by } \mathbf{x}^1 + \mathbf{x}^2} + \overbrace{(M-2)\mu}^{\text{by } \mathbf{x}^3 + \cdots + \mathbf{x}^M} \Big) + O\big(N^{-1/2}\big). \tag{B.4}$$

In the large network limit $N \to \infty$, we obtain

$$S_{++} = \frac{p^2(2 + (2p-1)(M-2))}{M(2p-1)^2 + 4p(1-p)}, \tag{B.5}$$

$$S_{-+} = \frac{p(1-p)(2p-1)(M-2)}{M(2p-1)^2 + 4p(1-p)}, \tag{B.6}$$

using equations A.8, A.10, and A.11, together with the limits $|I_{++}|/N \to p^2$ and $|I_{-+}|/N \to (1-p)p$. These two formulas have been confirmed by numerical simulations (data not shown). In the special case $p = 1/2$, we have $S_{++} = 1/2$ and $S_{-+} = 0$. Given coefficients S_{++} and S_{-+} in equations B.5 and B.6, the remaining two coefficients, namely, S_{--} and S_{+-}, are readily obtained from equations 5.8.

Appendix C: Critical Coupling Strength of the Conjugate Network

In this section we determine the critical asymmetric coupling strength λ_c of the reduced system for a conjugate network (equations 3.15 and 3.16), with an arbitrary gain function.

C.1 Exact Solution. Setting $\dot{x} = 0$ in equation 3.15 yields the nullcline equation,

$$-x + f(x) - \lambda f(y) = 0, \tag{C.1}$$

whereas setting $\dot{y} = 0$ in equation 3.16 yields

$$-y + f(y) + \lambda f(x) = 0. \tag{C.2}$$

Taking the derivative of these two equations with respect to x, we find

$$-1 + f'(x) - \lambda f'(y) \frac{dy}{dx} = 0, \tag{C.3}$$

$$\big(-1 + f'(y)\big) \frac{dy}{dx} + \lambda f'(x) = 0, \tag{C.4}$$

where f' is the derivative of f. The intersections of the two nullclines are the stationary states of the system.

Figure 8 shows an example with $f(u) = \tanh(2u)$. When $\lambda < \lambda_c$, the two nullclines intersect at nine points, including four sinks, four saddles, and one source at the origin (see Figures 8A and 8B). Each nullcline, say, the one for $\dot{x} = 0$, can have disjoint branches when λ is too small (see Figure 8A). For a given x, equation C.1 yields a solution $y = f^{-1}((-x + f(x))/\lambda)$ only if $|-x + f(x)|/\lambda = |f(y)| \leq 1$, because it is impossible to have $|f(y)| = |\tanh(2y)| > 1$. Therefore, the asymptotes for the disjoint branches of $\dot{x} = 0$ should satisfy the algebraic equation: $|-x + f(x)| = \lambda$.

When $\lambda > \lambda_c$, there is only one intersection point at the origin, and the system has a limit cycle (see Figure 8D and section 4.1). At the critical strength $\lambda = \lambda_c$, the two nullclines touch at four points (see Figure 8C).

In order to compute λ_c, note that when the two nullclines touch at a point, the slope dy/dx must be the same at that point. Canceling dy/dx from equations C.3 and C.4 yields

$$(1 + \lambda^2) f'(x) f'(y) - f'(x) - f'(y) + 1 = 0. \tag{C.5}$$

Thus, solving the three simultaneous algebraic equations C.1, C.2, and C.5 should yield the coordinates (x_c, y_c) of a contact point, as well as the critical coupling strength λ_c.

For the example in Figure 8, by numerically solving equations C.1, C.2, and C.5, we found $\lambda_c = 0.2713$, $x_c = -0.4387$, and $y_c = 1.173$. The solutions for x_c and y_c are not unique because of rotation symmetry.

C.2 Approximate Solution. The approximation is based on the observation that $dy/dx \approx 0$ at a contact point for the nullclines $\dot{x} = 0$ (see Figure 8C). Setting $dy/dx = 0$ at $x = x_c$ in equation C.3, we get

$$f'(x_c) \approx 1. \tag{C.6}$$

Solving x_c from this algebraic equation gives one coordinate of the touch point. Cancelling λ from equations C.1 and C.2 yields

$$\big(f(x) - x\big) f(x) + \big(f(y) - y\big) f(y) = 0. \tag{C.7}$$

Since the touch point approximately coincides with the point with the largest (or least) value of y on nullcline $\dot{x} = 0$ (Figure 8C), we set $f(y_c) = f_{\max}$ in equation C.7, with $f_{\max}$ being maximum possible value of the gain function f, and then obtain

$$y_c \approx f_{\max} + \big(f(x_c) - x_c\big) f(x_c)/f_{\max}. \tag{C.8}$$

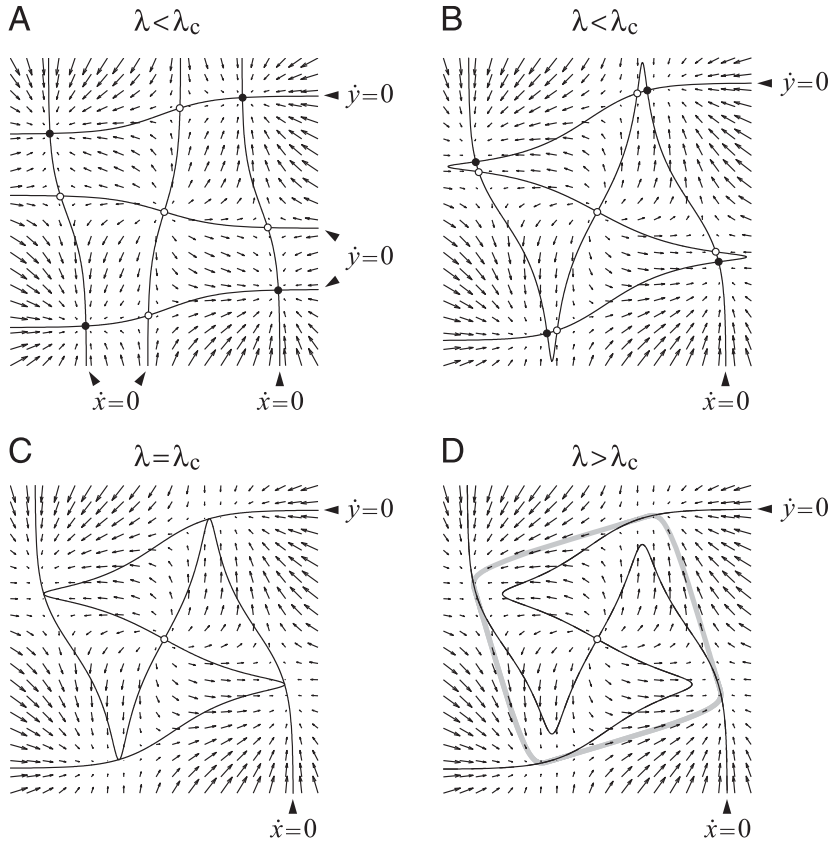

Figure 8: Analysis of the critical asymmetric coupling strength for the reduced system of the conjugate network. The origin $(0, 0)$ is always an unstable stationary point (source). The arrows indicate the vector field $(\dot{x}, \dot{y})$. Plot range is from -1.5 to 1.5 for both axes. Gain function $f(u) = \tanh(2u)$. (A) When the asymmetric coupling is too weak ($\lambda = 0.15 < \lambda_c$), the nullcline for $\dot{x} = 0$ consists of three disjoint branches, and so does the nullcline for $\dot{y} = 0$. These two sets of nullclines intersect at eight stationary points besides the origin, including four stable sinks (filled circles) and four unstable saddle points (open circles). (B) When the asymmetric coupling is stronger but still below the critical value ($\lambda = 0.268 < \lambda_c$), the two nullclines, each now becoming a single continuous curve, intersect at eight stationary points besides the origin, including four stable sinks (filled circles) and four unstable saddle points (open circles). (C) At the critical coupling strength ($\lambda = \lambda_c = 0.2713$), the four pairs of sinks and saddles in panel B now merge into four stationary points at which the two nullclines are tangent to each other. (D) When the asymmetric coupling exceeds the critical value ($\lambda = 0.28 > \lambda_c$), the nullclines intersect only at the origin, while a limit cycle (gray curve) emerges.

Finally, the critical coupling strength can be solved from equations C.1 and C.6:

$$\lambda_c \approx \big(f(x_c) - x_c\big)/f_{\max}. \tag{C.9}$$

For example, for $f(u) = \tanh(ku)$ with $k = 2$, we have $f_{\max} = 1$. Equation C.6 yields a solution $x_c \approx -k^{-1}\operatorname{arcsech} k^{-1/2} \approx -0.4407$, which leads to $y_c \approx 1.188$ and $\lambda_c \approx 0.2711$ by equations C.8 and C.9. These results are fairly close to the exact numerical solutions given at the end of last subsection.

Acknowledgments

This work was supported partially by grants AFOSR FA9550-12-1-0018 and NIH R01MH079511.

References

Amit, D. J. (1989). *Modeling brain function: The world of attractor neural networks*. Cambridge: Cambridge University Press.

Burgess, N. (2007). Computational models of the spatial and mnemonic functions of the hippocampus. In P. Andersen, R. Morris, D. Amaral, T. Bliss, & J. O'Keefe (Eds.), *The hippocampus book* (pp. 715–749). Oxford: Oxford University Press.

Buzsáki, G. (2006). *Rhythms of the brain*. Oxford: Oxford University Press.

Cohen, M. A., & Grossberg, S. (1983). Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man and Cybernetics*, *13*, 815–826.

Diesmann, M., Gewaltig, M.-O., & Aertsen, A. (1999). Stable propagation of synchronous spiking in cortical neural networks. *Nature*, *402*, 529–533.

Hahnloser, R. H. R., Kozhevnikov, A. A., & Fee, M. S. (2002). An ultra-sparse code underlies the generation of neural sequences in a songbird. *Nature*, *419*, 65–70.

Hasselmo, M. E. (2012). *How we remember: Brain mechanisms of episodic memory*. Cambridge, MA: MIT Press.

Herz, A. V. M., Li, Z., & van Hemmen, J. L. (1991). Statistical mechanics of temporal association in neural networks with transmission delays. *Physical Review Letters*, *66*, 1370–1373.

Hirsch, M. W., & Smale, S. (1974). *Differential equations, dynamical systems, and linear algebra*. New York: Academic Press.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences USA*, *79*, 2554–2558.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences USA*, *81*, 3088–3092.

Hopfield, J. J., & Brody, C. D. (2001). What is a moment? Transient synchrony as a collective mechanism for spatiotemporal integration. *Proceedings of the National Academy of Sciences USA*, *98*, 1282–1287.

Ikegaya, Y., Aaron, G., Cossart, R., Aronov, D., Lampl, I., Ferster, D., & Yuste, R. (2004). Synfire chains and cortical songs: Temporal modules of cortical activity. *Science*, *304*, 559–564.

Jin, D. Z., Fujii, N., & Graybiel, A. M. (2009). Neural representation of time in cortico-basal ganglia circuits. *Proceedings of the National Academy of Sciences USA*, *106*, 19156–19161.

Kanter, I., & Sompolinsky, H. (1987). Associative recall of memory without errors. *Physical Review A*, *35*, 380–392.

Kleinfeld, D. (1986). Sequential state generation by model neural networks. *Proceedings of the National Academy of Sciences USA*, *83*, 9469–9473.

Kleinfeld, D., & Sompolinsky, H. (1988). Associative neural network model for the generation of temporal patterns: Theory and application to central pattern generators. *Biophysical Journal*, *54*, 1039–1051.

Knierim, J. J., & Zhang, K. (2012). Attractor dynamics of spatially correlated neural activity in the limbic system. *Annual Reviews of Neuroscience*, *35*, 267–285.

Kohonen, T. (1977). *Associative memory: A system-theoretical approach*. Berlin: Springer-Verlag.

Kohonen, T., & Ruohonen, M. (1973). Representation of associated data by matrix operators. *IEEE Transactions on Computers*, *22*, 701–702.

Kosko, B. (1988). Bidirectional associative memories. *IEEE Transactions on Systems, Man and Cybernetics*, *18*, 49–60.

Kühn, R., & van Hemmen, J. L. (1995). Temporal association. In E. Domany, J. L. van Hemmen, & K. Schulten (Eds.), *Models of neural networks I* (pp. 221–288). Berlin: Springer.

Maass, W., Natschläger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, *14*, 2531–2560.

Mauk, M. D., & Buonomano, D. V. (2004). The neural basis of temporal processing. *Annual Reviews of Neuroscience*, *27*, 307–340.

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., & Moser, M. B. (2006). Path integration and the neural basis of "cognitive map." *Nature Reviews Neuroscience*, *7*, 663–678.

Personnaz, L., Guyon, I., & Dreyfus, G. (1985). Information storage and retrieval in spin-glass-like neural networks. *Journal de Physique Lettres (Paris)*, *46*, 359–365.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes: The art of scientific computing* (3rd ed.). Cambridge: Cambridge University Press.

Sejnowski, T. J. (1977). Storing covariance with nonlinearly interacting neurons. *Journal of Mathematical Biology*, *4*, 303–321.

Sompolinsky, H., & Kanter, I. (1986). Temporal association in asymmetric neural network. *Physical Review Letters*, *57*, 2861–2864.

Zhang, K. (1994). *Temporal association by Hebbian connections: The method of characteristic system* (Department of Cognitive Science Technical Report 9402). La Jolla, CA: University of California, San Diego.

Received October 25, 2010; accepted February 17, 2014.