Attractor-based models for sequences and pattern generation in neural circuits

Juliana Londono Alvarez, Katie Morrison, Carina Curto

March 7, 2025

Abstract

Neural circuits in the brain perform a variety of essential functions, including input classification, pattern completion, and the generation of rhythms and oscillations that support functions such as breathing and locomotion. There is also substantial evidence that the brain encodes memories and processes information via sequences of neural activity. Traditionally, rhythmic activity and pattern generation have been modeled using coupled oscillators, whereas input classification and pattern completion have been modeled using attractor neural networks. Here, we present a theoretical model that demonstrates how attractor-based networks can also generate diverse rhythmic patterns, such as those of central pattern generators (CPGs). Additionally, we propose a mechanism for transitioning between patterns. Specifically, we construct a network that can step through a sequence of five different quadruped gaits. It is composed of two dynamically different networks: a "counter" network that can count the number of external inputs it receives, encoded via a sequence of fixed points; and a locomotion network that encodes five different quadruped gaits as limit cycles. A sequence of locomotive gaits is obtained by connecting the sequence of fixed points in the counter network with the dynamic attractors of the locomotion network. To accomplish this, we introduce a new architecture for layering networks that produces "fusion" attractors, minimizing interference between the attractors of individual layers. All of this is accomplished within a unified framework of attractor-based models using threshold-linear networks.

Keywords: attractor networks, sequences, pattern generation, locomotion

Contents

1 Introduction **1**

2 Results **4**
2.1 CTLN fixed points and attractors . . . 5
2.2 TLN counter . . . 6
2.3 Coexistence of dynamic attractors for five distinct quadruped gaits . . . 9
2.4 Sequential control of quadruped gaits . . . 14
2.5 Why does this work? The theory behind the intuition . . . 16

3 Discussion **17**

4 Materials and methods **18**

A Supplementary materials **23**
A.1 TLN counters . . . 23
A.2 Quadruped gaits . . . 26
A.3 Proofs of Section 2.5 . . . 33

1 Introduction

Attractor neural networks play an important role in computational neuroscience by providing a rich framework for modeling the behavior of neural systems [1, 2]. Originally devised as models of associative memory, these networks were designed to store static patterns representing discrete memories as attractors. Their ability to simultaneously encode multiple static patterns, represented as fixed points, made them ideal for this purpose. Perhaps the most well-known example is that of Hopfield networks.

As illustrated in Figure 1A1, the classical Hopfield paradigm models memories as stable fixed points stored in the network, each one accessible via distinct input pulses. These fixed points *coexist* in the sense that they are all accesible for the same parameters. The state space is partitioned into basins of attraction, and each input initiates a distinct trajectory in one of these basins. Importantly, all such models are nonlinear. Coexistence of attractors, even if it's only at the level of multiple stable fixed points, requires nonlinear dynamics.

While Hopfield networks are well-suited for encoding multiple static patterns, many aspects of neural activity are better modeled using *dynamic* attractors. This includes rhythms and oscillations produced by central pattern generator circuits (CPGs). CPGs are neural circuits that generate control movements such as walking, swimming, chewing, and breathing [3]. Unlike static patterns (e.g. images), these rhythmic processes typically involve neurons that fire *sequentially*, meaning that neurons take turns to fire. Because of this, dynamic attractors are more appropriate models. Furthermore, a single CPG network should potentially be able to encode multiple such patterns, as animals can have several locomotive gaits, all of which activate the same limbs [4, 5]. How can different overlapping dynamic patterns of neural activity be produced by the same network? Modeling this using attractors requires multistability of *dynamic* attractors, which is known to be a difficult problem [6, 7].

Traditional models of locomotion and other CPGs have tackled these challenges by using coupled oscillators [8, 9, 10, 11, 12, 13]. In these models, distinct activity patterns do not correspond to coexisting attractors. Instead, the parameters of the model need to be adjusted to allow different attractors to emerge. For example, synaptic strengths may need to change every time the animal transitions between different locomotive gaits; this requires precise and rapid synaptic plasticity [10, 11]. Moreover, while there is evidence for pacemaker neurons, not all neurons are intrinsic oscillators [14]. Because of these challenges, coupled oscillator models do not fit naturally within the framework of attractor neural networks, and are thus difficult to reconcile with the leading models of pattern completion and associative memory. Despite this, coupled oscillator models have remained popular. One of the reasons they are widely used is the availability of theoretical results coming from physics and mathematics [15]. Indeed, CPG models have historically stemmed from the physics and mathematics communities [8, 9, 10, 11].

The availability of theoretical tools is a compelling argument for using a given model family. Fortunately, recent theoretical advances in recurrent threshold-linear networks have provided new tools for modeling rhythmic pattern generation in an attractor framework [16, 17]. These models differ from traditional coupled oscillator models in two key ways. First, the neurons do not intrinsically oscillate; instead, patterns of activity emerge as a result of connectivity. Second, these patterns are true attractors of the system (as opposed to fine-tuned trajectories), making them more robust and stable to noise and other perturbations. Additionally, providing models within an attractor framework unifies our approach to CPGs with other classical models in neuroscience, such as associative memory models, ring attractors, etc [2, 1]. Seeking a unified framework is more than a matter of elegance: it has been suggested that cortical circuits have many features in common with CPGs, including an abundance of inhibition [18].

In this work, we provide attractor-based models for three different neural functions, summarized in Figure 1. First, we model a discrete neural integrator that can count inputs and is robust to noise, as well as a signed variant that can track position along a line or model evidence accumulation in a binary choice task [19]. These "counter" models encode a sequence of fixed point attractors, as shown in Figure 1A2. Although similar to the classical Hopfield paradigm (Figure 1A1), it differs in that the sequence is *internally* encoded, with external input pulses being identical and containing no information about which fixed point comes next in the sequence. Second, we devise a network for quadruped locomotion, where attractors correspond to five different quadruped gaits. These attractors coexist in the same network as distinct limit cycles in state space, as shown in Figure 1A3. Despite the overlapping nodes (limbs) between patterns (gaits), each one can be easily accessed through different initial conditions without changing any of the network parameters. Lastly, we combine the approaches from panels A2 and A3 to develop a network that is capable of stepping through a sequence of quadruped gaits, as illustrated in Figure 1A4. As in panel A2, the sequence of attractors is internally encoded: identical input pulses

advance the network from one attractor to the next.

Note that in the case of locomotive gaits, the attractors involve neurons firing in stereotyped sequences. We call these *sequential attractors* [20]. For example, in Figure 1C, the neurons in the attractor fire in a repeating sequence: (12)3(45)6, where (12) and (45) denote synchronous pairs of neurons. In contrast, the network activity depicted in Figure 1B converges to a static attractor where neurons 3 and 4 are coactive. When we combine the "counter" model with a network encoding dynamic attractors, as in Figure 1A4, we obtain a new type of "fusion" attractors in which a fixed point and a sequential attractor merge into a single limit cycle in a higher-dimensional state space (Figure 1D).

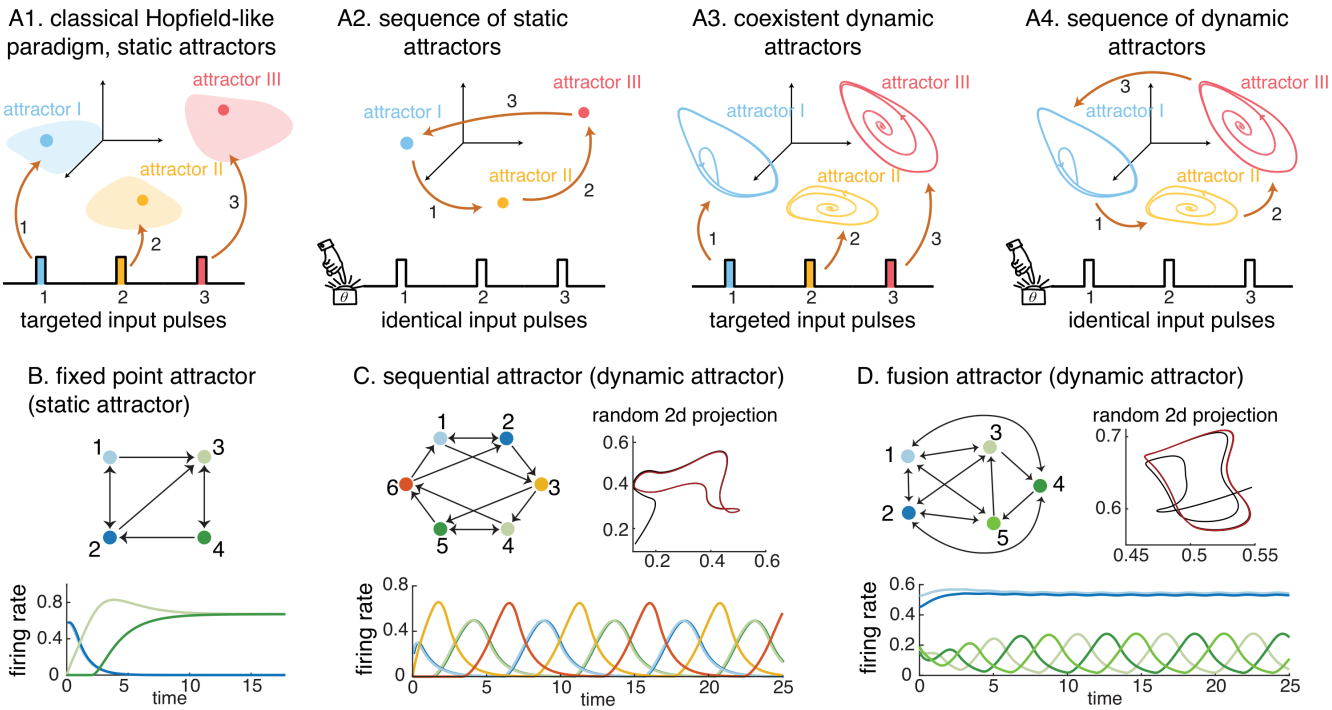


Figure 1: Summary of attractor network paradigms, and attractor types. (A1) Classical Hopfield-like paradigm: multiple stable fixed points (stable attractors) are encoded in the same network, each accessible via distinct (colored) input pulses that push the network state into the corresponding basin of attraction (colored blobs). (A2) Multiple stable fixed points are encoded in the same network, each accessible in a predetermined sequence via identical pulses. (A3) Multiple *dynamic* attractors are encoded in the same network, each accessible via distinct (colored) input pulses. (A4) Internally encoded sequence of dynamic attractors, with each step of the sequence triggered by identical inputs. (B) Fixed point attractor realized via a clique in a TLN. (C) Sequential attractor (limit cycle) realized via a cyclic union. Neurons take turns firing, with neurons 1 and 2 firing synchronously, as well as neurons 4 and 5. (D) Fusion attractor realized via a clique union. The attractor fuses a fixed point and a limit cycle. (C-D, top right) The trajectory is shown in a random 2-dimensional projection, with the last three quarters in red to highlight the limit cycle attractor.

To implement these attractor network models, we use threshold-linear networks (TLNs). TLNs are firing rate models of recurrent networks with non-oscillating units whose dynamics are abstracted from the activation functions of real neurons. We view TLNs as dynamical systems, prescribed by the coupled differential equations:

$$\frac{dx_i}{dt} = -x_i + \left[\sum_{j=1}^{n} W_{ij}x_j + b_i\right]_+ , \quad i = 1, \ldots, n, \tag{1}$$

where $x_1(t), \ldots, x_n(t)$ represent the firing rates of n recurrently-connected neurons, $W = [W_{ij}]$ is a connectivity matrix, $b_1(t), \ldots, b_n(t)$ are external inputs, and $[\cdot]_+ = \max\{0, \cdot\}$ is the familiar ReLU activation function. TLNs have a rich history in computational neuroscience [21, 22, 23, 24, 25], and are supported by a wide array of theoretical results [26, 16, 27, 28, 29], many of them recent [17, 30, 20, 31].

One such result is that threshold-linear networks with symmetric connectivity can only have stable fixed point attractors [32, 33]. Since we also want dynamic attractors, such as the sequential and fusion attractors in Figure 1C-D, we must consider non-symmetric TLNs. Many of the strongest results in this case concern a special family called combinatorial threshold-linear networks (CTLNs) [30, 31]. CTLNs are TLNs whose connectivity matrix $W = W(G, \varepsilon, \delta)$ in equation (1) is prescribed by a simple directed graph G:

$$W_{ij} = \begin{cases} 0 & \text{if } i = j, \\ -1+\varepsilon & \text{if } j \to i \text{ in } G, \\ -1-\delta & \text{if } j \not\to i \text{ in } G. \end{cases} \tag{2}$$

Parameters that satisfy $\delta > 0$ and $0 < \varepsilon < \frac{\delta}{\delta+1}$ are said to be in the *legal range*. All parameters in this work are taken from the legal range unless explicitly stated otherwise. In addition, the external input values for CTLNs are kept constant across neurons and time, $b_i(t) = \theta > 0$, in order to ensure the dynamics are internally generated. However, here we also consider input pulses where $b_i(t)$ temporarily increases for one (or all) neurons during a short time interval, as illustrated in Figure 1A by the step function shape of the pulse.

CTLNs can exhibit a rich variety of nonlinear dynamics, including multistability, limit cycles, quasi-periodicity and chaos [31]. They are thus better suited than classical Hopfield networks or symmetric TLNs for simultaneously encoding multiple dynamic patterns of activity. Indeed, Figure 1B shows some of the behaviors that can easily arise from very simple, small CTLNs and can be engineered using the theoretical results found in [34, 20]. Stable fixed point attractors can be easily obtained using *target-free cliques* (Theorem 1), as seen in Figure 1B. Similiarly, CTLNs also make it easy to encode sequential attractors (Figure 1C) and fusion attractors (Figure 1D) via *cyclic union* and *clique union* constructions [30, 34].

Here we use CTLNs as a starting point for engineering a series of attractor-based models: a discrete neural integrator with static (fixed point) attractors, a network encoding five distinct but overlapping quadruped gaits as sequential attractors, and a network that combines these mechanisms to internally encode a sequence of gaits via fusion attractors. These models illustrate the different paradigms in Figure 1A2-4. Interestingly, the fusion attractors that arise in the quadruped gaits network use different architecture from the previously known type given in Figure 1D. We thus develop new theory, given in Section 2.5, to explain how fusion attractors can be obtained by transient coupling across different layers of a network. This construction can also be done with other networks, e.g. a network controlling the swimming directions of the marine mollusk Clione [35, 36]. The resulting fusion attractors enable a network to store a prescribed sequence of dynamic attractors, where the attractors (e.g. gaits, swimming directions) are encoded independently of the sequence in which they are activated.

2 Results

Here we present models for all situations in Figure 1A2-4. First, we present two counter networks (or neural integrators), one that simply counts input pulses and another that counts signed pulses. These networks represent sequential activation of static (fixed point) attractors. Next, we present a CPG circuit for locomotion that encodes five distinct quadruped gaits. Each gait corresponds to a dynamic attractor, and all five attractors coexist in the network and are individually accessible via different initial conditions. Moreover, using targeted input pulses, the network can rapidly switch from one gait to another. Finally, we combine the counter and quadruped gait networks into a single network with multiple layers that encodes a sequence of gaits. In this model, the sequence of gaits is internally encoded via the connections between the layers, while the timing of transitions in the sequence are externally controlled via the timing of the (identical) input pulses. Each model displays rich dynamics, despite having a small number of neurons. The design principles stem directly from mathematical insights on CTLNs, which we present next.

2.1 CTLN fixed points and attractors

For CTLNs, there is a particularly strong relationship between the dynamics of the network and its defining graph G. This has given rise to extensive theory, often formulated as "graph rules," relating the fixed point structure of a CTLN to various properties of G [30, 34, 37, 20]. This theory also sheds light on the attractors of the network, as static attractors are fixed points and dynamic attractors are closely related to certain unstable fixed points [37, 34].

A fixed point x^* of equation (1) is a solution that satisfies $\left.\frac{dx_i}{dt}\right|_{x=x^*} = 0$ for each $i = 1, \ldots, n$, and its *support* is the subset of active neurons,

$$\operatorname{supp} x^* = \{i \in [n] \mid x_i^* > 0\}, \text{ where } [n] \stackrel{\text{def}}{=} \{1, \ldots, n\}.$$

Since nondegenerate TLNs can have at most one fixed point per support [31, 30], the fixed points of a network can be labeled by their supports. For a CTLN with defining graph G, and parameters ε, δ, we denote the set of all fixed point supports as:

$$\mathrm{FP}(G) = \mathrm{FP}(G, \varepsilon, \delta) \stackrel{\text{def}}{=} \{\sigma \subseteq [n] \mid \sigma \text{ is the support of a fixed point } x^*\}. \tag{3}$$

This set contains both stable and unstable fixed points. It is easy to find the exact value of the fixed point from its support: $x_\sigma^* = (I - W)_\sigma^{-1} b_\sigma$, for $\sigma \in \mathrm{FP}(G)$, where the σ subscript denotes the vector/matrix obtained by restricting to the indices in σ [34].

In order to predict the attractors of a TLN, it is useful to look at a special subset of minimal fixed point supports corresponding to *core motifs*:

$$\mathrm{FP}_{\text{core}}(G) \stackrel{\text{def}}{=} \{\sigma \subseteq [n] \mid \sigma \in \mathrm{FP}(G) \text{ and } \mathrm{FP}(G|_\sigma) = \{\sigma\}\}.$$

Core motifs are induced subgraphs, $G|_\sigma$, that have a unique fixed point which has full support, so that $\mathrm{FP}(G|_\sigma) = \{\sigma\}$. Note that $\mathrm{FP}_{\text{core}}(G)$ also contains both stable and unstable fixed point supports. The stable fixed points correspond to static attractors, but the unstable fixed points are also quite relevant as they provide valuable insights into the dynamic attractors [37, 34].

The two main results about CTLNs from [30] that we will use in this work are: (1) the *uniform in-degree rule*, and (2) the *cyclic union rule*. The uniform in-degree rule is used in our first set of models, the TLN counters, so we present it here. The cyclic unions result will be presented later in Section 2.3 (see Theorem 2), and is used to build the quadruped gaits network.

A directed graph G has *uniform in-degree* d if every node of G has exactly d incoming edges. For example, a clique (all-to-all connected graph) on n nodes has uniform in-degree $d = n - 1$, and a cycle has uniform in-degree $d = 1$. An induced subgraph, $G|_\sigma$, has uniform in-degree d if each node receives d incoming edges from within $G|_\sigma$. For these graphs, we always have $\sigma \in \mathrm{FP}(G|_\sigma)$. Moreover, it is easy to see whether this fixed point *survives* to be a fixed point of the larger network. The following theorem assumes that the parameters giving rise to the respective CTLN are in the legal range, but it is independent of their values within the legal range and thus they are not explicitly mentioned in the theorem:

Theorem 1 (uniform in-degree, [30])**.** *Let G be a directed graph on n nodes, and $\sigma \subseteq [n]$. Suppose $G|_\sigma$ has uniform in-degree d. For $k \notin \sigma$, let $d_k \stackrel{\text{def}}{=} |\{i \in \sigma \mid i \to k\}|$ be the number of edges k receives from σ. Then*

$$\sigma \in \mathrm{FP}(G) \quad \Leftrightarrow \quad d_k \leq d \ \text{ for all } \ k \notin \sigma.$$

Furthermore, if $|\sigma| > 1$ and $d < |\sigma|/2$, then the fixed point is unstable. If $d = |\sigma| - 1$, then the fixed point is stable.

Note that subgraphs $G|_\sigma$ that are cliques have $d = |\sigma| - 1$, and thus always correspond to stable fixed points. In fact, it has been conjectured that cliques are the only graphs that support stable fixed points in CTLNs [30]. As an example, consider the graph in Figure 1B. If we let $\sigma = \{1, 2\}$ and $\tau = \{3, 4\}$, both $G|_\sigma$ and $G|_\tau$ are cliques that have uniform in-degree 1. However, node 3 receives two edges from σ, so $\sigma \notin \mathrm{FP}(G)$. In contrast, τ does not have any external node receiving more than one edge from τ, so $\tau \in \mathrm{FP}(G)$. In other words, the fixed point corresponding to σ *dies* in the larger network G, while the fixed point for τ *survives*. Indeed, convergence to the τ fixed point is shown in Figure 1B, despite the network being initialized on σ.

2.2 TLN counter

In this section we model a discrete counter as a sequence of stable fixed points, each accessible via identical inputs, as shown in Figure 1A2. Neural integration is an important correlate of processes such as oculomotor and head direction control, short term memory keeping, decision making, and estimation of time intervals [19, 38, 39, 40]. Classic neural integration models are known to be fine-tuned, requiring exact values for the parameters in order to achieve perfect integration. By contrast, some more robust models tend to be rather insensitive to weaker inputs, requiring strong inputs to switch between adjacent states [19, 39]. The models we propose here represent a simple alternative, and are both robust and respond well to a wide variety of input strengths. Importantly, because they are encoding sequences of stable fixed points, they could be also useful to model highly stereotyped sequences, like songbird songs [41].

By Theorem 1, stable fixed points can be engineered via target-free cliques. We thus construct our network by chaining together several target-free cliques, as shown in Figure 2A. We have wrapped the chain around to ensure activity will not stall in the last clique of the chain. By Theorem 1 these 2-cliques will yield surviving fixed point supports as long as they do not have more than one outgoing edge to any other node in the graph. For instance, if we were to add the edge $1 \to 4$, then nodes $\{1, 2\}$, associated to the first clique, would no longer support a fixed point of the network, as node 4 would now be a target. We have thus built a network whose $\mathrm{FP}(G)$, by design, should contain all the cliques ($\{1, 2\}, \{3, 4\}, \{5, 6\}, \{7, 8\}, \{9, 10\}, \{11, 12\}$) as core fixed point supports. Note that each cycle ($\{1, 3, 5, 7, 9, 11\}, \{2, 4, 6, 8, 10, 12\}$) also supports a fixed point of the network by Theorem 1, since no node receives more than one edge from either cycle. Computationally, we confirm these inclusions and found, overall that $|\,\mathrm{FP}(G)| = 141$ and

$$\mathrm{FP}_{\text{core}}(G) = \{\{1, 2\}, \{3, 4\}, \{5, 6\}, \{7, 8\}, \{9, 10\}, \{11, 12\}, \{1, 3, 5, 7, 9, 11\}, \{2, 4, 6, 8, 10, 12\}\} \tag{4}$$

This indicates that we will have a stable fixed point attractor per clique, and two unstable fixed points giving rise to dynamic attractors. To test this prediction, we simulated the network using the parameters $\varepsilon = 0.25$, $\delta = 0.5$, a baseline $\theta = 1$, and an input pulse $\theta = 5$ of 3 time units duration. However, since Theorem 1 does not depend on the values of these parameters, any TLN with dynamics prescribed by equation (1), whose connectivity matrix W is built from the graph of Figure 2A according to equation (2), will have the same fixed point structure (eq. (4)). This is true for any values of ε and δ, provided they are within the legal range.

The $\mathrm{FP}_{\text{core}}(G)$ predictions about the dynamics are partially confirmed by the rate curves of Figure 2A: activity remains at the stable fixed point that the network was initialized to, until a uniform θ-pulse is sent to all neurons in the network. When the network receives the pulse, activity slides down to the next stable state in the chain. Activity is maintained in this state indefinitely until future pulses are provided to the system. Thus, the network is effectively counting the number of input pulses it has received via the position of the attractor in the linear chain of attractor states. The count is maintained via well-separated *discrete* states, providing a straightforward readout mechanism of the encoded count.

On the other hand, the supports $\{1, 3, 5, 7, 9, 11\}, \{2, 4, 6, 8, 10, 12\}$ of equation (4) (corresponding to the top and bottom cycles) predicted the existence of two dynamic attractors, which were not found

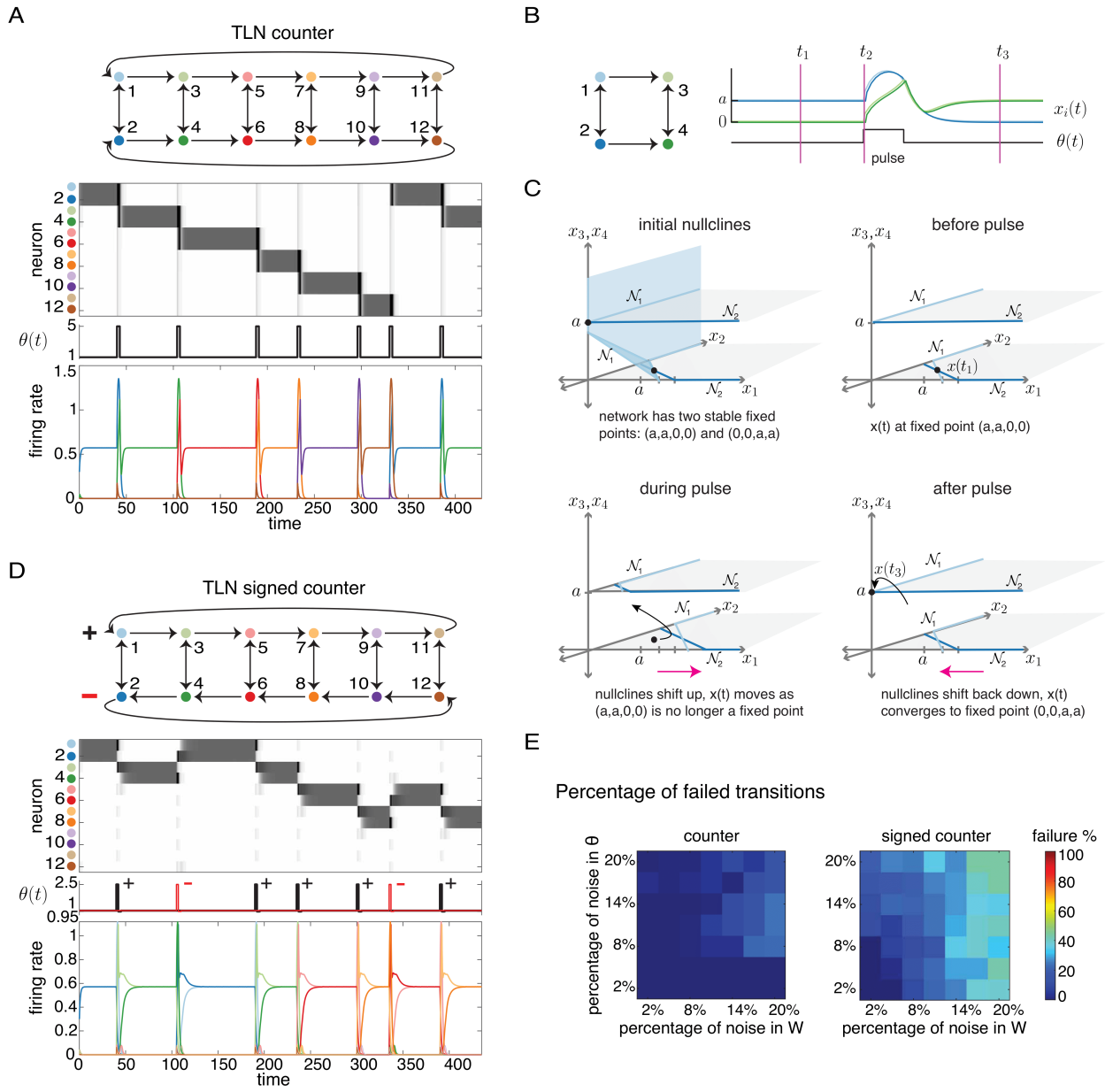


Figure 2: TLN counters. (A) Unidirectional chain of 2-cliques. Pulses are sent to all neurons in the network, sliding activity to the next clique. Pulse duration is 2 time units. (B) A small subnetwork consisting of only two 2-cliques. The state is initially in a stable fixed point supported on the $1 \leftrightarrow 2$ clique (blue) with $x(t) = (a, a, 0, 0)$ with $a = \frac{\theta_0}{2-\varepsilon}$. After the pulse, the network settles into the second stable fixed point, corresponding to the $3 \leftrightarrow 4$ clique (green), with $x(t) = (0, 0, a, a)$. (C) Illustration of the hysteresis mechanism underlying the fixed point transitions. (Top left) Nullclines for the first pair of nodes. The $\mathcal{N}_1$ nullcline is the set of states where $\frac{dx_1}{dt} = 0$. It consists of two hyperplane segments, one for $x_1 = 0$ and one for $x_1 > 0$ (shaded in blue). The $\mathcal{N}_2$ hyperplane has the same kind of structure but only the intersections with the horizontal slices $x3, x4 = 0$ and $x3, x4 = a$ are shown for ease of visualization. The system has two stable fixed points, one on each of these slices (black dots). (Top right) Before the pulse the network activity is at the stable fixed point $(a, a, 0, 0)$ supported on the $1 \leftrightarrow 2$ clique. (Bottom left) A θ-pulse shifts the nullclines in state space, causing the trajectory (black) to move. (Bottom right) When the pulse ends, the network state $x(t)$ is in a new basin of attraction and converges to the stable fixed point $(0, 0, a, a)$. (D) A TLN signed counter is constructed by flipping the direction on the bottom cycle of even numbered neurons. Black pulses shift activity right (odd neurons), red pulses shift left (even neurons). Pulse duration is 2 time units, with 3 time units of refractory period during which $\theta = 0.95$. (E) Failed transition percentage per pulse trial as a function of noise in W and θ. Heatmap shows failed transitions, with axes representing noise percentages in θ and W. For details on how it was computed see Supplementary Materials section A.1.

computationally.

Note that the value of the current fixed point depends on the value of the previous fixed point, so the network "remembers" where it came from (this is called *hysteresis*). How does this phenomenon arise from the θ pulse? In Figure 2B we present a short version of the counter, along with a detailed view of activity around the pulse. Hysteresis occurs because fixed points lie at intersections of *nullclines* $\mathcal{N}_i$ (subsets of the state space where $\frac{dx_i}{dt} = 0$), and the arrival of a pulse causes the nullclines to shift. The i-th nullcline of a TLN is the union of a pair of hyperplane segments given by

$$\mathcal{N}_i : x_i = \begin{cases} \sum_{j=1}^n W_{ij}x_j + \theta_i & \text{if } \sum_{j=1}^n W_{ij}x_j + \theta_i > 0 \\ 0 & \text{if } \sum_{j=1}^n W_{ij}x_j + \theta_i \leq 0. \end{cases}$$

Figure 2C illustrates the hysteresis mechanism for the subnetwork in panel B. In the first subpanel (top left), the nullcline $\mathcal{N}_1$ is fully shown. The other nullclines $\mathcal{N}_i$ resemble $\mathcal{N}_1$, each consisting of a slanted part when $x_i > 0$ and a segment of a coordinate hyperplane with $x_i = 0$. The network has two stable fixed points, each lying at an intersection of nullclines. We restrict attention to the horizontal slices $x_3 = x_4 = 0$ and $x_3 = x_4 = a$ where the fixed points lie. Initially, the network is settled in the $(a, a, 0, 0)$ fixed point, where $a = \frac{\theta_0}{2-\varepsilon}$ (top right subpanel). When the pulse comes, the value of $\theta(t)$ changes, causing the nullclines to slide upward (bottom left subpanel). As a result, $(a, a, 0, 0)$ is no longer a fixed point, and the trajectory moves upward ($dx_i/dt > 0$ for all i). When the pulse ends, the nullclines return to their initial positions. However, the trajectory does not return to the original $(a, a, 0, 0)$ fixed point. Instead, it settles into the second fixed point, $(0, 0, a, a)$, supported on the $3 \leftrightarrow 4$ clique.

In the full counter (panel A), there are six stable fixed points, and the selected fixed point reflects the history of the number of pulses that have transpired. A natural question is: can we adapt this construction to a signed counter, where "$-$" and "$+$" pulses shift the counter left and right?

Signed TLN counter. To create a counter that can track both "$-$" and "$+$" pulse inputs, we begin from the TLN counter architecture but reverse the direction of the bottom cycle through the even numbered nodes. For a $+$ input, a pulse is sent to the nodes in the top cycle only (odd numbered nodes); these are connected in a way that will advance the counter activity to the right. For a $-$ input, a pulse is sent to nodes in the bottom cycle only (even nodes), which causes the counter activity to shift left. This allows the network to track relative number of "$-$" v.s. "$+$" inputs. Interestingly, the signed counter is more sensitive to pulses than the original TLN counter and is more likely to advance through multiple cliques at once in an uncontrolled "roulette" style cycling (see Supplementary Materials section A.1). However, when we introduce a brief refractory period after each pulse, the signed counter performs as desired, only advancing by a single clique at a time.

The core motif analysis in this case is the same, as the cliques remain target-free and the two cycles again survive by Theorem 1. Computationally, we also found $|\mathrm{FP}(G)| = 369$ and

$$\mathrm{FP}_{\text{core}}(G) = \{\{1,2\},\{3,4\},\{5,6\},\{7,8\},\{9,10\},\{11,12\},\{1,3,5,7,9,11\},\{2,4,6,8,10,12\}\}. \quad (5)$$

Again the predicted behavior of the network is confirmed in simulations, as seen in the rate curves of Figure 2D. The dynamics of this new counter are analogous to those of the previous counter, except now the count can be decreased. When the network receives an input pulse to the plus neurons (top cycle, black pulses in Figure 2D), the activity slides to the right. If it receives an input pulse to the minus neurons (bottom cycle, red pulses in Figure 2D), the activity slides to the left. That is, the activity travels in the direction of the cycle that received the pulse. This network can not only keep a count, but also store net displacement, position on a line, or relative number of left and right cues.

The sequences of fixed point attractors in both the signed and unsigned TLN counters were obtained by chaining together core motifs (2-cliques) that were guaranteed to yield stable fixed points. Similar ideas can be used to construct a more complex counter network where the individual states representing the count are dynamic attractors [35]. In addition, even though all the networks presented in this section

keep a count modulo 6 (= number of chained motifs)—which could be useful when estimating time intervals or angles—this can be generalized to an arbitrary (but finite) number of motifs, allowing the network to keep a memory of an arbitrarily large count. Thus, the number of possible states approaches infinity as the number of links in the chain approaches infinity as well. Importantly, the networks scale linearly with the number of possible states, requiring two neurons per extra state.

In contrast to traditional integrators, which are highly sensitive to parameter tuning, the models we present here are robust to both parameter variations and noise. Simulations show a wide range of parameters $(\varepsilon, \delta, \theta)$ where the counters effectively track the number of input pulses received (see Supplementary Figure S1). However, performance can degrade in certain parameter regimes (e.g., rouletting behavior, stalls in a clique, or counts in multiples of two). These behaviors were studied for varying values of ε, δ, pulse intensity, and duration in Supplementary Figure S1.

The TLN counters are also robust to noise in the connectivity matrices W and in the inputs $\theta(t)$ (Figure 2E). This robustness is further explored in Supplementary Figure S2, where we also show cases where the model fails, such as when it slides too much, gets stuck, or both.

2.3 Coexistence of dynamic attractors for five distinct quadruped gaits

A classical example in neuroscience that can be modeled with coexistence of dynamic attractors is animal locomotion [43, 13, 12]. Modeling the different modes of locomotion has been challenging because of the heavy overlap in units controlling limbs between different gaits. In addition, engineering the coexistence of several dynamic attractors is difficult because basins of attraction of nonlinear systems are often too intricate to reliably access each attractor [6, 7]. Most models overcome this difficulty by requiring changes in network parameters, such as synaptic weights, in order to transition between different gaits [43, 8, 12]. Here, we present a CTLN capable of reproducing five coexistent quadruped gaits: bound, pace, trot, walk and pronk; without any change of parameters needed to access different gaits. These gaits truly coexist as distinct limit cycle attractors in the network where different gaits are accessed via different initial conditions, or by stimulation of a specific neuron involved in the gait, as shown in Figure 1A3.

Math interlude: cyclic unions as pattern generators. One important architectural family of CTLNs that gives rise to patterned activations is the *cyclic union*, defined in [30] as follows: given a set of component subgraphs $G|_{\tau_1}, \ldots, G|_{\tau_N}$, on subsets of nodes $\tau_1, \ldots, \tau_N$, the *cyclic union* of $G|_{\tau_1}, \ldots, G|_{\tau_N}$ is constructed by connecting these subgraphs in a cyclic fashion so that there are edges forward from every node in τ_i to every node in τ_{i+1} (cyclically identifying τ_N with τ_0), and there are no other edges between components (Fig. 3A). Remarkably, for cyclic unions it is possible to completely determine the fixed point supports of the full network in terms of the component fixed point supports:

Theorem 2 (cyclic unions, Theorem 13 in [30])**.** *Let G be a cyclic union of component subgraphs $G|_{\tau_1}$, …, $G|_{\tau_N}$. For any $\sigma \subseteq [n]$, let $\sigma_i \stackrel{\text{def}}{=} \sigma \cap \tau_i$. Then*

$$\sigma \in \mathrm{FP}(G) \quad \Leftrightarrow \quad \sigma_i \in \mathrm{FP}(G|_{\tau_i}) \quad \textit{for all } i \in [N].$$

Moreover,

$$\sigma \in \mathrm{FP}_{\textit{core}}(G) \quad \Leftrightarrow \quad \sigma_i \in \mathrm{FP}_{\textit{core}}(G|_{\tau_i}) \quad \textit{for all } i \in [N].$$

Note that, in particular, $\sigma_i \neq \emptyset$ for all $i \in [N]$ and so Theorem 2 guarantees that every fixed point of G hits every component. Since the core motifs of a cyclic union are found by taking a core motif from each component, we can patch together core motifs to follow cyclic activations using Theorem 2. For example, the quadruped gait bound is characterized by having both front limbs synchronized, both back limbs synchronized, and then successively cycling through these pairs, half a period off. This cycling makes it natural to group front limbs (nodes 1 and 2 in Figure 3B) in a single component, τ_1,

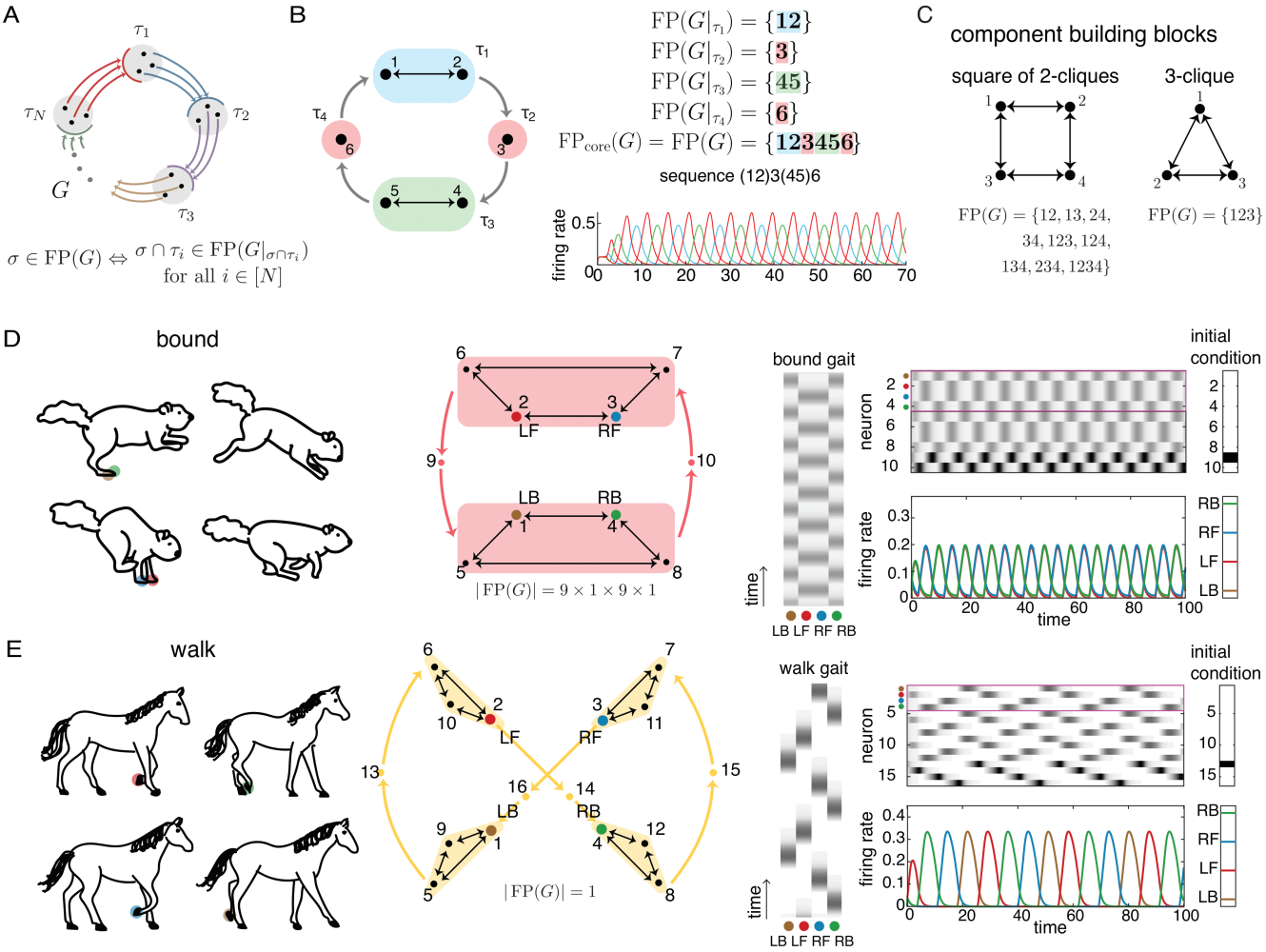


Figure 3: Design of quadruped gaits. (A) Cyclic union theorem. (B) Example of a cyclic union with two 2-cliques and two single nodes. $\mathrm{FP}(G)$ of the whole graph is made up of pieces taken from each $\mathrm{FP}(G_{\tau_i})$, colored-coded by component. Notation: $\{i_1, \ldots, i_k\}$ is simplified as $i_1 \cdots i_k$ (e.g., $12 = \{1, 2\}$). Neurons 1,2,4, and 5 represent limbs, producing a bound gait. The firing rates were obtained using parameter values $\varepsilon = 0.51, \delta = 1.76, \theta = 1$. (C) The basic building blocks for each isolated gait, and their $\mathrm{FP}(G)$ sets. (D) Souslik's bound, adapted from [42]. Colored circles represent node-limb assignment. Red shaded blocks group nodes in the same component. Red arrows represent edges from/to all nodes in a single component. Nodes 1 through 4 control the limbs (LB, LF, RF, RB). Nodes 5 to 10 are auxiliary nodes. The first four rows of the grayscale, outlined in pink and reproduced vertically, show the activity of the limbs. The network was initialized at $x_9(0) = 0.1$ and all other neurons at 0. (E) Horse's walk, adapted from [42]. Colored circles represent node-limb assignment. Yellow shaded triangles group nodes in the same component. Yellow arrows represent edges from/to all nodes in a single component. Nodes 1 through 4 control the limbs (LB, LF, RF, RB). Nodes 5 to 16 are auxiliary nodes. The first four rows of the grayscale, outlined in pink and reproduced vertically, show the activity of the limbs. The network was initialized at $x_{14}(0) = 0.1$ and all other neurons at 0.

back limbs (nodes 4 and 5) in a single component, τ_3, and to connect them through two auxiliary 1-node components, τ_2, τ_4, to ensure that the phase between groups of limbs is half a period off. Theorem 2 then says that $\mathrm{FP}(G)$ is made up of $\mathrm{FP}(G|_{\tau_i})$ pieces, $i \in [4]$. Since each τ_i is a core motif (single nodes and 2-cliques), we obtain $\mathrm{FP}(G) = \mathrm{FP}_{\text{core}}(G)$, as detailed in Figure 3B. Notice that in the resulting attractor every component is active, and the neural activity flows through the components in cyclic order. If we take nodes 1, 2, 4, and 5 to represent limb nodes, then the network in Figure 3B successfully reproduces the activations seen in the bound gait.

Single gaits. In the previous example, bound was constructed using only six nodes. In what follows, we aim to construct a single network that encodes multiple gaits. However, having so few nodes per gait becomes impractical, as adding new gaits would significantly increase the number of edges between those nodes. This would cause the six-node network in Figure 3B to collapse into a clique, ultimately yielding a stable fixed point, as per Theorem 1. To address this, we propose a slightly different construction for bound, introducing a few extra auxiliary neurons per limb to mitigate this issue.

The resulting network is shown in Figure 3D, where we replace the previous 2-clique components with a square structure composed of four 2-cliques. The $\mathrm{FP}_{\text{core}}(G)$ of this square consists of all the individual 2-cliques that form its sides, as illustrated in Figure 3C.

Theorem 2 now says that $\mathrm{FP}(G)$ of the bound gait network of Figure 3D is obtained by selecting a support $\sigma_i \in \mathrm{FP}(G|_{\tau_i})$ from each component τ_i and forming their union: $\sigma_1 \cup \{9\} \cup \sigma_2 \cup \{10\}$, where $\sigma_1 \in \{\{2,3\},\{2,6\},\{6,7\},\{3,7\}\}$ and $\sigma_2 \in \{\{1,4\},\{1,5\},\{5,8\},\{4,8\}\}$. This includes, among others, the support $\{1,2,3,4,13,14\}$ corresponding to the bound attractor seen in the rate curves of Figure 3D. In an analogous way, it is possible to similarly construct pace (left limbs are synchronized, right limbs are synchronized, half a period off) and trot (diagonal limbs are synchronized, half a period off). The core motifs are similarly derived, using the "square" building block of Figure 3C, along with Theorem 2.

A slightly different construction is needed for the gait walk, where limbs move cyclically with a quarter period difference (Figure 3E). Again, we use Theorem 2 to group each limb in a component along with its auxiliary neurons and add four auxiliary neurons to ensure that the phase between limbs is a quarter period (Figure 3E). The building blocks are now copies of the 3-clique of Figure 3C. Since the building block is a core motif, the $\mathrm{FP}_{\text{core}}(G)$ of the walk gait consists of a single element, formed by taking a core motif from each component in Figure 3E. Again, this design trick of adding extra auxiliary neurons is needed for the 5-gait network construction ahead. The resulting dynamics of this network are depicted in Figure 3E, and the network once again behaves as expected. The last gait our model includes is pronk, where all limbs are synchronized, requiring now a single component consisting of all four limbs together along with a pair of auxiliary nodes that control the frequency of the pronking movement.

All five networks described above are pictured in Figure 4A. We computationally derived the sizes of each of the $\mathrm{FP}(G)$ for each gait networks as designed in the previous paragraphs. These results, along with a summary of the previous discussion on their $\mathrm{FP}_{\text{core}}(G)$ can be found in the Supplementary Table 2. Additionally, to verify that the core motifs accurately predict the attractors, we simulated each of the gait networks as previously constructed. The results are presented in Supplementary Figure S3D, which includes the grayscale and rate curves for all isolated gaits, along with the induced subgraphs for each gait.

5-gait network. To construct the 5-gait network, we glued together the networks from Figure 4A by identifying common neurons (1 through 8) along with all induced edges between them. The resulting glued network is shown in Figure 4C, this is the 5-gait network. More precisely, this gluing identifies all copies of nodes 1 to 8 from each isolated gait to a single node in the final network, along with their induced edges. This operation does not correspond exactly to a standard graph operation; rather, it is a topological gluing, where the identification rules are determined by the vertex numbers and the edges between them.

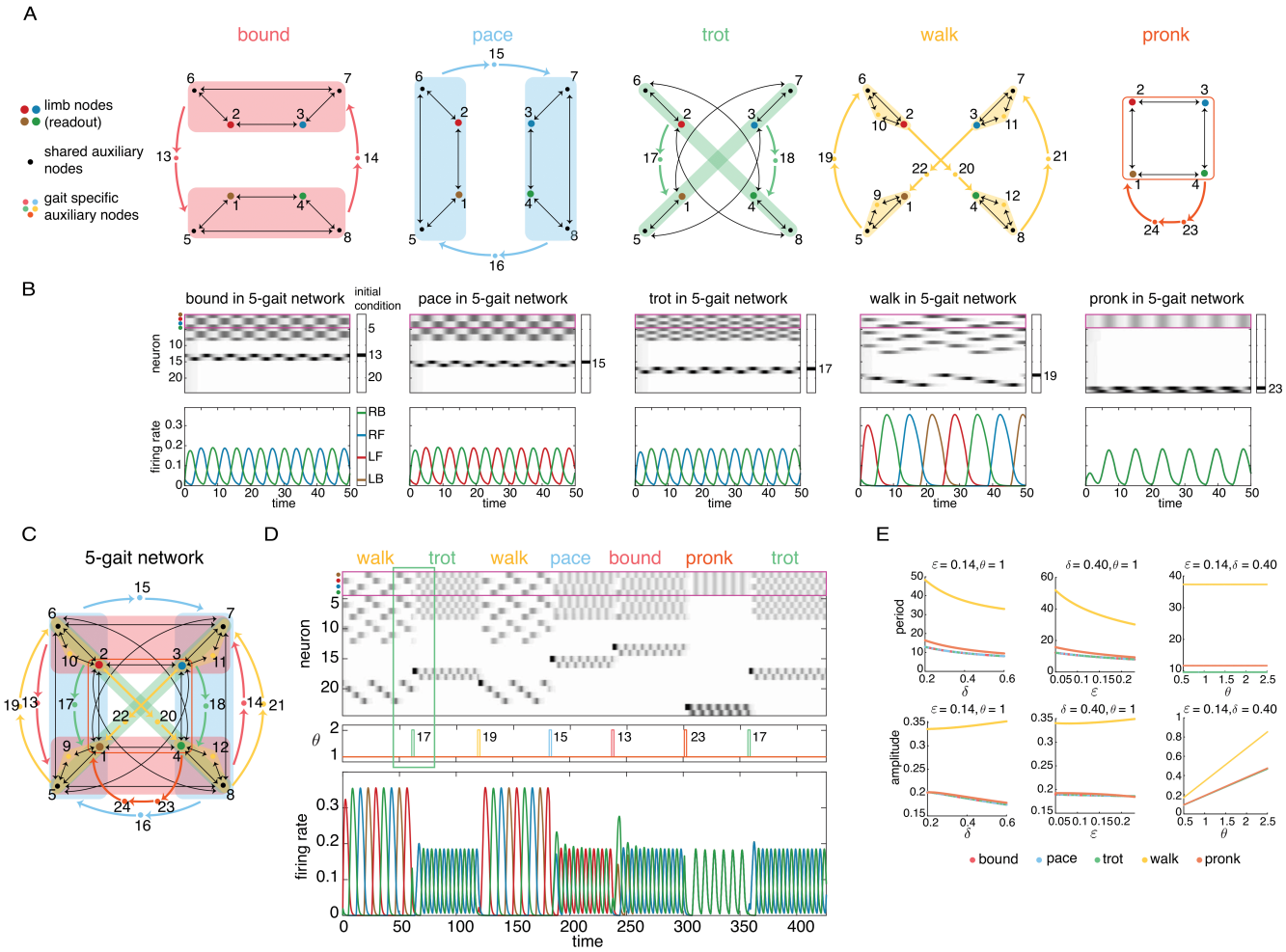


Figure 4: 5-gait network and transitions. (A) Graphs for gaits constructed individually according to Theorem 2. Color-shaded areas group nodes in the same component. Colored arrows represent edges from/to all nodes in a single component. (B) Simulation of all 5 gaits within the network from panel C below. Initial conditions were set to zero for all neurons except for neuron $x_i = 0.2$, where i corresponds to the neuron specified in the panel next to each grayscale. The pink square highlights the activity of the readout neurons (limb nodes). (C) 5-gait network after gluing all individual networks from panel A (note that the networks in panel A are not the induced subgraphs of the 5-gait network). (D) Gait transitions via θ pulses. The neuron being stimulated with the pulse in the middle plot is written and colored according to which gait it corresponds to. The duration of the pulse is 2 time units. The activity quickly transitions to the attractor representing the corresponding gait. The order in which neurons are stimulated does not matter. The green box highlights the activity around the first pulse. (E) Amplitude and period of the gaits in the 5-gait network as controlled by $\varepsilon, \delta, \theta$. Gaits are color coded as in panel D. Values were found using the last half (once the attractor has settled) of a simulation with running time $T = 500$.

As mentioned before, the inclusion of several auxiliary nodes was necessary to merge the five networks of Figure 4A into a single one. Indeed, nodes 1 to 4 are shared by all gaits and represent the limbs: 1. LB - left back, 2. LF - left front, 3. RF - right front, 4. RB - right back. Nodes 5 to 8 are auxiliary (not representing limbs) and are shared by bound, pace and trot. The rest of the nodes are gait-specific, meaning that they are uniquely associated to a single gait and therefore only active when such gait is active, and are off otherwise. These nodes play the important roles of (1) keeping the phase between foot steps in each gait and (2) activating a specific gait by stimulating one of them. These auxiliary nodes are: nodes 9 to 12, specific to walk; and nodes 13 to 23 which are specific to a unique gait, as color coded in Figure 4A.

Although the complex structure of the glued network makes it hard to analytically derive the $\mathrm{FP}(G)$, we computationally found its core motifs (Supplementary Materials Table 2). Interestingly, none of these core motifs precisely correspond to the gait attractors, and yet all gait attractors are in fact preserved in the glued network, as seen in Figure 4B. The corresponding grayscale graphs and rate curves for each gait within the glued network are plotted below the respective graph in Figure 4A. Furthermore, the dynamics of each gait, when embedded in the glued network, are remarkably similar to those of the isolated networks. This is further illustrated in Supplementary Figure S3D, which shows the grayscale plots and rate curves for all isolated gaits, along with the induced subgraphs for each gait.

To see that each gait is accessible via different initial conditions, observe in Figure 4B that when all neurons, but neuron 13, are initialized to zero, the system evolves into the bound gait. This is because neuron 13 corresponds *uniquely* to bound. Similarly, when the network is initialized on neuron 15, the dynamics go into pace. When it is initialized on neuron 17, the dynamics go into trot. And so on. Recall that walk has two different sets of gait-specific neurons (9 - 12 and 19 - 22), and both kinds will send the network into the walk gait. Hence, our 5-gait network successfully encodes these coexistent dynamic attractors that have several overlapping active nodes (nodes 1 to 8), with no interference between attractors. Moreover, the attractors are not overly sensitive to the value of initial conditions, as long as the highest firing neuron in the initial condition is one that is uniquely associated with the desired gait.

Moreover, since all gaits coexist as distinct limit cycle attractors in the network, each accessible via different initial conditions, it is also possible to smoothly transition between gaits by means of a targeted external pulse θ. Specifically, it suffices to stimulate an appropriate gait-specific auxiliary neuron with a pulse. The network quickly settles into the dynamic attractor corresponding to the gait of the auxiliary neuron stimulated, as seen in Figure 4D. There, pulses are sequentially sent to neurons 17, 19, 15, 13, 23 and 17, producing the expected transitions between gaits. Any sequence of pulses sent to gait-specific auxiliary neurons will produce the respective transitions, regardless of the desired ordering of the gaits.

Finally, we evaluated the network's performance under several parameter values, and show how the amplitude and period of the firing rate curve for each gait can be modulated by these parameters, as shown in Figure 4E. Increasing ε and δ resulted in a decrease in the period, while the period remained flat when varying θ, meaning the timing was unchanged but the intensity of firing increased. Notably, the walk gait showed an increase in amplitude, while the other gaits exhibited reduced amplitude as δ increased. For additional examples of how varying ε, δ, and θ affect the attractors (e.g., preserved symmetry, loss of symmetry, or switching to a different attractor), see Supplementary Figure S4. Details on how the period and amplitude of oscillations in firing rates change with these parameters, and how these values were obtained, are provided in Supplementary Figure S5.

Additionally, we evaluated the network's performance under noise added to both the connectivity matrix W and the input signal θ. Bound, pace, and trot gaits remained stable with up to 2% noise but began to break down at 4%. In contrast, walk and pronk gaits were robust to noise levels as high as 10% and beyond. Supplementary Figure S6 further explores how gaits behave under varying levels of noise in both W and θ. While all gaits survived with up to 2% noise, others withstood higher levels but exhibited distorted behavior; the attractors remained, but their dynamics were less stable. This figure also examines transitions between gaits under noisy W and θ, showing that while transitions were

disrupted at higher noise levels, they remained relatively stable at 1% noise.

2.4 Sequential control of quadruped gaits

In this section, we design a network that encodes a sequence of gaits *internally*, in contrast to the setup in Figure 4D, where each gait was selected by targeted input pulses, meaning any sequence was determined *externally*. To achieve internal encoding, we use fusion attractors, which combine the activity of the counter network from Figure 2A with the 5-gait network from Figure 4C. The objective is to create a network that, given identical inputs (with externally controlled input timing), outputs a predetermined sequence of gaits (or any set of dynamic attractors), as illustrated in Figure 1A4.

Fusion attractors are formed by combining stable fixed points from the counter network with dynamic attractors representing gaits from the 5-gait network, as illustrated in Figure 5A. This combination of stable fixed point with gait attractors results in a total of $5 \cdot m$ fusion attractors, coming from 5 possible gaits and m possible fixed points from the counter.

Figure 5A shows an example where the counter consists of four 2-cliques (stable fixed points). Out of the possible $5 \cdot 4 = 20$ fusion attractors listed in Figure 5B, only four are activated (bold). These are determined by the thick brown arrows—every gait-clique pairing that receives brown arrows from the same clique will be activated as a fusion attractor. For instance, the first fusion attractor triggered in Figure 5A is clique 2 + walk, since both receive edges from the same (previous) clique. Note that the cliques are always stepped through in order, but their pairing can be with any gait, dictated by the connectivity between the previous clique and the gait network. As a result, gaits can be executed in any pre-specified order. Also note that the external pulse is what drives the sequence forward.

In practice, directly connecting the counter network to the 5-gait network caused interference, where the fixed-point attractors of the counter disrupted the dynamic attractors of the quadruped gaits network. To resolve this, we introduced an intermediate relay layer (see Figure 5C) that is inactive except during the pulse transitions. The attractors of this 3-layer network include $7 \cdot 5 = 35$ fusion attractors, each consisting of one of the 7 stable fixed points from the cliques in layer L1 together with one of the 5 gaits from layer L3. While all 35 fusions are technically attractors of the network, in practice, only 6 of them are typically accessed; specifically, the sequence of 6 that are encoded in the network via the connections between L1, L2, and L3 (see thick brown arrows in Figure 5C).

From the prescription of brown arrows in Figure 5C, the network encodes the sequence: clique 2 + pace, clique 3 + walk, clique 4 + bound, clique 5 + pronk, clique 6 + trot, clique 7 + walk, accessed in that order by identical pulses. While gaits can be repeated in the sequence, each fusion attractor is different from previous repetitions of that gait. For instance, clique 3 + walk and clique 7 + walk both output the same gait (walk), but they correspond to different fusion attractors. This difference in attractors is key to remembering location in the sequence, as L1 keeps track of which step in the sequence is active.

Fusion attractors that are active are easily identifiable in the grayscale of Figure 5D. Recall the fusion attractor pairings are determined by the thick brown arrows—everything receiving an brown arrow from the same clique will be fused together. For example, pronk fuses with clique 5 as both pronk (via node 19) and clique 5 receive inputs from clique 4.

Also, note that the input pulses in Figure 5D, which target layers L1 and L2, are identical and uniform within each layer. Importantly, layer 2 receives no external θ input outside of pulse times; specifically $\theta_{L1} = \theta_{L2} = 1$ outside of pulse times, while $\theta_{L3} = 0$. The network's output corresponds to the activity of the readout neurons, or limb neurons, which directly encode the gaits.

Finally, as seen in Figure 5E, this network is not technically a CTLN, as the off-diagonal values are not exclusively $-1 + \varepsilon$ or $-1 - \delta$. Non-edges within a layer and non-edges from one layer to the next are valued at $-1 - \delta$, while all other non-edges are set to 0. Consequently, the matrix is sparse, unlike CTLNs, with many zeros in addition to strong and weak inhibition.

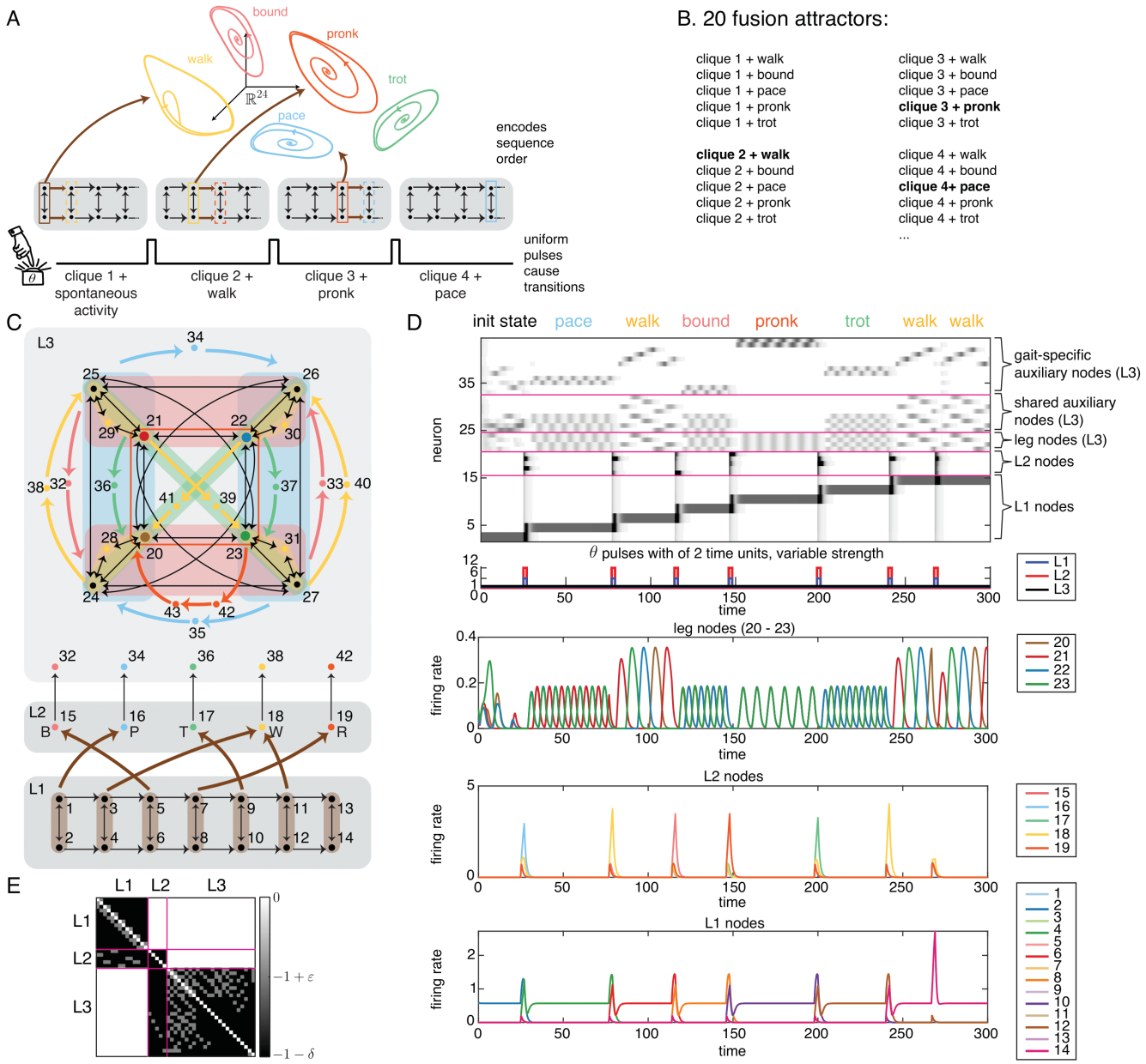


Figure 5: Sequential control of quadruped gaits. (A) Network design. Multiple attractors coexist in $\mathbb{R}^{24}$ (state space of the quadruped gaits network), accessed via uniform global pulses (θ-pulses) sent to the TLN counter network (gray). Each pulse triggers the activity to advance following the brown arrows. The sequence order is encoded by the brown connections, separately from the attractors. The active clique is highlighted with a solid box, and the clique to be activated is highlighted with a dashed box, color coded by the gait it will be coactivated with. (B) List of fusion attractors that are possible from the gait attractors and counter network pictured in A. Out of the $5 \cdot 4$ possible attractors, 3 of them are selected by the prescription of the brown arrows of A. Selected attractors in bold. (C) Layered network for sequential control of quadruped gaits. L1 (unsigned counter) encodes the sequence steps. L2 (independent set) acts as relay layer, each node connects to an attractor-specific node in L3 (5-gait network). Attractor-specific nodes corresponding to different gaits are redrawn at the bottom for clarity. Every node within the shaded block sends edges to where the thick arrow points. (D) Grayscale, pulses and rate curves. On top of the grayscale is the order in which gaits are accessed, matching those specified by the brown arrows in C. Only neurons in L1 and L2 receive pulses. L2 does not receive any external input outside of the pulse times. There are four different θ values pictured. Neurons in L1 have a baseline $\theta = 1$ and pulse $\theta = 6$. Neurons in L2 have a baseline $\theta = 0$ and pulse $\theta = 12$. Neurons in L3 have a constant $\theta = 1$. Note that transitions between cliques in L1 are communicated effectively as pulses to L3 via L2. The pairs of attractors that are fused are the ones that get co-activated by the prior clique after a pulse. For example, starting in clique $\{1, 2\}$, when the pulse comes, both clique $\{3, 4\}$ and the pace gait get stimulated, producing $\{3, 4\}$-P as the first fusion attractor. (E) W matrix for the network in C.

2.5 Why does this work? The theory behind the intuition

The mechanism behind the network in the previous section is actually provably simple. It involves two distinct networks that are transiently coupled through a relay layer, as seen in Figure 6. To formalize this, we need to establish the existence of fusion attractors in networks with this structure. Specifically, these are networks with one layer encoding only stable fixed points, an intermediate relay layer, and another layer encoding dynamic attractors. In fact, the structure of the network in the previous section can be further relaxed.

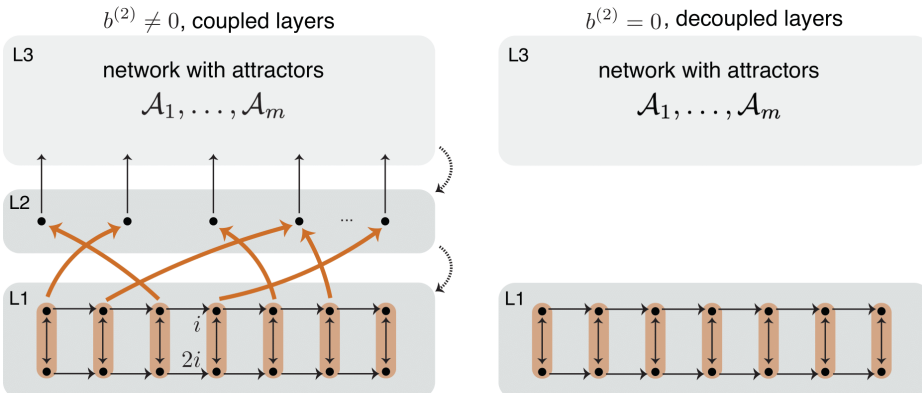


Figure 6: Fusion attractors obtained via decoupled layers. (A) General structure of network for Theorem 3. L1 is a network with only stable fixed point attractors. L2 an arbitrary network (independent set in this case). L3 is an arbitrary network with given attractors. The only connections allowed by the theorem are between consecutive layers (black, orange and dashed edges). (B) Resulting network after external input is turned off for layer 2 ($b^{(2)} = 0$).

Theorem 3. *Consider a TLN* (W, b) *where*

$$W = \left[\begin{array}{c|c|c} W^{(1)} & * & 0 \\ \hline * & W^{(2)} & * \\ \hline 0 & * & W^{(3)} \end{array}\right], \tag{6}$$

and

$$b = \left(\begin{array}{c} b^{(1)} \\ \hline b^{(2)} \\ \hline b^{(3)} \end{array}\right), \tag{7}$$

and suppose that $(W^{(1)}, b^{(1)})$ *is a network with* k *stable fixed points and no other attractors;* $(W^{(2)}, b^{(2)})$ *is an independent set on* ℓ *nodes; and* $(W^{(3)}, b^{(3)})$ *is an arbitrary recurrent network with attractors* $\mathcal{A}_1, \ldots, \mathcal{A}_m$.

Then, if $b^{(2)} \leq 0$*, the attractors of the network* (W, b) *include the* $m \cdot k$ *fusion attractors consisting of one stable fixed point attractor coming from* $(W^{(1)}, b^{(1)})$ *fused with one* $\mathcal{A}_j$ *attractor coming from* $(W^{(3)}, b^{(3)})$.

The proof of this theorem is provided in Supplementary Materials Section A.3. This theorem specifically shows that the role of L2 is to transiently couple layers L1 and L3 during the pulse. Importantly, the theorem allows us to select among the attractors by initializing the system in the desired attractor in L1 and L3. Since the attractors are robust, the influence of L2 remains transient and minimal. Similar results have been established for other architectures involving CTLN layers, which can be found in [35].

Note that for the specific case of L1 being the counter network in Figure 6 we have a stable fixed point for each clique, by Theorem 1. Note that because the cycles are absent in this modified counter network, there are no additional core fixed points. This leads us to conjecture that the only attractors in

the L1 network of Figure 6 are the stable fixed points associated with the cliques, and by Theorem 3, we have:

Conjecture 4. *The network in Figure 6 has exactly $m \cdot k$ fusion attractors given by clique i + $\mathcal{A}_j$ for $i \in [k], j \in [m]$.*

This suggests that designing networks that step through attractors using this method should be relatively straightforward. Another example using a different set of dynamic attractors corresponding to the swimming directions of a marine mollusk can be found in [35].

3 Discussion

Our aim has been to unify models of CPGs, traditionally modeled using coupled oscillators, with models of associative memory networks, traditionally modeled using recurrent networks. This led us to chose the TLN framework, as it can produce a wide variety of dynamic behaviors, including coexistence of multiple static and dynamic patterns. Indeed, as we demonstrated in this work the dynamic variety of TLNs makes them well-suited for modeling a range of neural functions. While the model is not intended as a direct biological analog, it suggests mechanisms that could be tested in both neural and robotic systems.

To begin, we designed a network that encodes a count using multiple stable fixed points. Although this is similar to Hopfield networks, our network is not symmetric and no energy function exists. As a result, it can exhibit hysteresis, a property that enables the network dynamics to "remember" prior states. This hysteresis mechanism is precisely what allows the network to increase or decrease the count by moving through a series of attractor states in response to external inputs. This model provides a robust and simple alternative to traditional neural integrators and evidence accumulators [19].

The same TLN framework also allowed us to encode multiple dynamic attractors in a controlled and deliberate way. Rather than just encoding arbitrary dynamic patterns, we could design specific patterns of activation to encode. Using this method, we successfully encoded five distinct quadruped gaits—bound, pace, trot, walk, and pronk—within the network. Remarkably, the four nodes controlling the limbs, which are common to all gaits, fire in distinct patterns depending on the active gait. Additionally, rapid transitions between gaits are easily achieved by applying external targeted pulses.

Finally, by combining these two approaches, we applied the concept of fusion attractors (Fig. 1D) to create a network that can step through a sequence of quadruped gaits. In this case, the fixed point counter steps through a sequence of static attractors using external pulses, which in turn activates one of the dynamic patterns of the quadruped gait network. By integrating these two mechanisms, we translated a sequence of pulses into a sequence of gait activations.

This final construction preserves the key properties of the individual components. From the counter, we have an external controller that induces transitions between identical attractors on different subsets of neurons. From the quadruped gaits, we have several patterns available to select from. From the conjunction, we get the freedom to choose the sequence order. All of this is integrated by the fusion attractors, where the fixed-point component remains unchanged, but the gait paired with each fixed point depends on the network's connectivity. In other words, while the counter determines which fixed point is active, the network structure dictates which gait is associated with it. In this way, we encode the sequence of gaits within the network while maintaining external control over the transitions, much like the choreography of dance moves. Three additional benefits arise from the design of this model:

1. **Sequential information is internally encoded**. Since external pulses are sent to all neurons in layers L1 and L2, rather than to specific neurons controlling each gait (as in Section 2.3), the sequence is fully encoded within the network, and the pulses themselves carry no information about what the next step is.

2. **Sequential information is independently encoded from motor commands**. The order of the sequence is encoded in the connections between layers L2 and L3, while the dynamic attractors are encoded in L3. This disassociates the sequence order from the elements themselves, allowing the network to reuse elements by adding just a few extra neurons and connections. This efficient reuse of attractors is a remarkable feature, as dynamic attractors are usually very sensitive to interference [6, 7].

3. **Timing of transitions is externally encoded**. Timing of the sequence is controlled by external pulses, and therefore a sequence's execution can be sped up or slowed down, or rhythm altered altogether, without interfering with attractors or sequence order. Moreover, there is no need to change synaptic time constants or physiological variables to alter the duration of each sequence step.

4 Materials and methods

Network simulations (deterministic). Each network was simulated using the following equations:

$$\frac{dx_i}{dt} = -x_i + \left[\sum_{j=1}^{n} W_{ij}x_j + b_i(t)\right]_+, \quad i = 1, \ldots, n, \tag{8}$$

Here, W is obtained from the graph displayed in each figure, according to the prescription given by equation (2). The particular values of ε and δ vary per network, and can be found in Table 1.

The value of $b_i(t)$ also varies per network. The pulse values and the neurons pulsed were different for each network (as detailed below), but all of them had a baseline $b_i(t) = b = 1$. Pulses could be either targeted to specific neurons (attractor-specific) or sent to all the neurons in the network (global). When pulses were targeted, the color of the pulse in the corresponding figure indicates which neurons received it. Values of the targeted pulses, along with its baseline, for each network are detailed below. The pulsed neurons and pulse times can be inferred from the pulse plot in each of the simulations, between the gray scales and the rate curves.

Table 1: Values of the parameters used in each network simulation.

Network G	ε	δ	θ	ON neurons at $x(0)$	init
TLN counter (Fig 2A)	0.25	0.5	$1, 5$	1,2	0.3
signed TLN counter (Fig 2D)	0.25	0.5	$0.95, 1, 2.5$	1,2	0.3
bound (Fig 3D)	0.25	0.5	1	9	0.1
walk (Fig 3E)	0.25	0.5	1	13	0.1
all gaits in supernetwork (Fig 4B)	0.25	0.5	1	13,15,17,19,23	0.2
gait transitions (Fig 4D)	0.25	0.5	$1, 2$	19	0.3
sequential control of gaits (Fig 5D)	0.25	0.5	$1, 4.6$	1,15	0.2

Each solution $x_i(t)$, for $i = 1, \ldots, n$, was generated by solving the system of ODEs in equation (8), corresponding to each network as prescribed by the values in Table 1. The system was solved in MATLAB using a standard ODE solver (ode45). The initial conditions for all networks are: all neurons off ($x_i(0) = 0$), except the ones specified in the "ON neurons" column of Table 1. Those in the ON neurons column were initialized to the value indicated in the "init" column, i.e. $x_j(0) =$ init for j an ON neuron.

The solution $x_i(t)$ is displayed in two different forms in each figure. On top there is a gray scale, where darker gray corresponds to higher firing rates. Below these are the pulses received by the network. And on the bottom are the rate curves of each $x_i(t)$.

Acknowledgments

We would like to thank the members of our laboratory for their constructive feedback on the manuscript. Special thanks to Nicole Sanderson for her detailed comments. This work was supported by the grants NIH R01 EB022862, NSF DMS-1951165 and DMS-1951599. Part of this work was supported by the grant NSF DMS-1929284 while the authors were in residence at the Institute for Computational and Experimental Research in Mathematics (ICERM) in Providence, RI, during the Math + Neuroscience program in Fall 2023.

References

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

[2] M. Khona and I. R. Fiete. Attractor and integrator networks in the brain. *Nature Reviews Neuroscience*, 23(12):744–766, December 2022.

[3] E. Marder and D. Bucher. Central pattern generators and the control of rhythmic movements. *Curr. Bio.*, 11(23):R986–996, 2001.

[4] S. Grillner and P. Wallén. Cellular bases of a vertebrate locomotor system – steering, intersegmental and segmental co-ordination and sensory control. *Brain Res. Rev.*, 40:92–106, 2002.

[5] M. Arriaga and E. B. Han. Dedicated hippocampal inhibitory networks for locomotion and immobility. *J. Neurosci.*, 37:9222–9238, 2017.

[6] A. N. Pisarchik and U. Feudel. Control of multistability. *Physics Reports*, 540(4):167–218, 2014.

[7] U. Feudel, A. N. Pisarchik, and K. Showalter. Multistability and tipping: From mathematics and physics to climate and brain–minireview and preface to the focus issue. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(3):033501, 2018.

[8] J.A. White, C.C. Chow, J. Ritt, et al. Synchronization and oscillatory dynamics in heterogeneous, mutually inhibited neurons. *J. Comput. Neurosci.*, 5(1):5–16, 1998.

[9] M.A. Whittington, R.D. Traub, N. Kopell, et al. Inhibition-based rhythms: experimental and mathematical observations on network dynamics. *Int. J. Psychophysiol.*, 38(3):315–336, 2000.

[10] M. Golubitsky, I. Stewart, P-L. Buono, et al. Symmetry in locomotor central pattern generators and animal gaits. *Nature*, 401:693–695, 1999.

[11] J. J. Collins and I. N. Stewart. Coupled nonlinear oscillators and the symmetries of animal gaits. *Journal of Nonlinear Science*, 3(1):349–392, December 1993.

[12] S. Dutta, A. Parihar, A. Khanna, et al. Programmable coupled oscillators for synchronized locomotion. *Nature Communications*, 10(1):3299, July 2019.

[13] A. J. Ijspeert. Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks*, 21(4):642–653, 2008.

[14] Y. Penn, M. Segal, and E. Moses. Network synchronization in hippocampal neurons. *Proceedings of the National Academy of Sciences*, 113(12):3341–3346, 2016.

[15] P. Ashwin, S. Coombes, and R. Nicks. Mathematical frameworks for oscillatory network dynamics in neuroscience. *J Math Neurosci*, 6(1):2, December 2016.

[16] R. Hahnloser, H. S. Seung, and J.-J. Slotine. Permitted and forbidden sets in symmetric threshold-linear networks. *Neural Comput*, 15:621–638, March 2003.

[17] C. Curto and K. Morrison. Pattern completion in symmetric threshold-linear networks. *Neural Computation*, 28:2825–2852, December 2016.

[18] R. Yuste, J.N. MacLean, J. Smith, et al. The cortex as a central pattern generator. *Nat. Rev. Neurosci.*, 6:477–483, 2005.

[19] Mark S Goldman, A. Compte, and X. J. Wang. *Neural Integrator Models*, pages 165–178. Elsevier Ltd, 2010.

[20] C. Parmelee, J. Londono Alvarez, C. Curto, et al. Sequential attractors in combinatorial threshold-linear networks. *SIAM Journal on Applied Dynamical Systems*, 21(2):1597–1630, 2022.

[21] H. S. Seung and R. Yuste. *Appendix E: Neural Networks*, pages 1581–1600. McGraw-Hill Education/Medical, 5th edition, 2012.

[22] M. V. Tsodyks, W. E. Skaggs, T. J. Sejnowski, et al. Paradoxical effects of external modulation of inhibitory interneurons. *J Neurosci*, 17(11):4382–4388, June 1997.

[23] R. H. Hahnloser, R. Sarpeshkar, M.A. Mahowald, et al. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405:947–951, 2000.

[24] J. K. Lappalainen, F. D. Tschopp, S. Prakhya, et al. Connectome-constrained deep mechanistic networks predict neural responses across the fly visual system at single-neuron resolution. *bioRxiv*, 2023.

[25] T. Biswas and J. E. Fitzgerald. Geometric framework to predict structure from function in neural networks. *Phys. Rev. Res.*, 4:023255, June 2022.

[26] X. Xie, R. H. Hahnloser, and H.S. Seung. Selectively grouping neurons in recurrent networks of lateral inhibition. *Neural Comput.*, 14:2627–2646, 2002.

[27] C. Curto, A. Degeratu, and V. Itskov. Encoding binary neural codes in networks of threshold-linear neurons. *Neural Comput.*, 25:2858–2903, 2013.

[28] K. Morrison and C. Curto. *Predicting neural network dynamics via graphical analysis*. Book chapter in Algebraic and Combinatorial Computational Biology, edited by R. Robeva and M. Macaulay. Elsevier, 2018.

[29] C. Curto, J. Geneson, and K. Morrison. Stable fixed points of combinatorial threshold-linear networks. *Advances in Applied Mathematics*, 154:102652, 2024.

[30] C. Curto, J. Geneson, and K. Morrison. Fixed points of competitive threshold-linear networks. *Neural Comput.*, 31(1):94–155, 01 2019.

[31] K. Morrison, A. Degeratu, V. Itskov, et al. Diversity of emergent dynamics in competitive threshold-linear networks. *SIAM Journal on Applied Dynamical Systems*, 23(1):855–884, 2024.

[32] J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci.*, 79(8):2554–2558, 1982.

[33] R. H. Hahnloser, H.S. Seung, and J.J. Slotine. Permitted and forbidden sets in symmetric threshold-linear networks. *Neural Comput.*, 15(3):621–638, 2003.

[34] C. Curto and K. Morrison. Graph rules for recurrent neural network dynamics. *Notices of the American Mathematical Society*, 70(04):536–551, 2023.

[35] J. Londono Alvarez. *Attractor-based models for sequences and pattern generation in neural circuits*. PhD thesis, The Pennsylvania State University, 2024. Available at `https://arxiv.org/abs/2410.11012`.

[36] P. Varona, M. I. Rabinovich, A. I. Selverston, et al. Winnerless competition between sensory neurons generates chaos: A possible mechanism for molluscan hunting behavior. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 12(3):672–677, 2002.

[37] C. Parmelee, S. Moore, K. Morrison, and C. Curto. Core motifs predict dynamic attractors in combinatorial threshold-linear networks. *Plos One*, 17(3):1–21, 03 2022.

[38] U. Büttner and J. A. Büttner-Ennever. Present concepts of oculomotor organization. In J. A. B"uttner-Ennever, editor, *Neuroanatomy of the Oculomotor System*, volume 151 of *Progress in Brain Research*, pages 1–42. Elsevier, 2006.

[39] H. S. Seung, D. D. Lee, B. Y. Reis, et al. Stability of the memory of eye position in a recurrent network of conductance-based model neurons. *Neuron*, 26(1):259–271, 2000.

[40] M. E. Mazurek, J. D. Roitman, J. Ditterich, et al. A role for neural integrators in perceptual decision making. *Cerebral Cortex*, 13(11):1257–1269, November 2003.

[41] M. A. Long, D. Z. Jin, and M. S. Fee. Support for a synaptic chain model of neuronal sequence generation. *Nature*, 468(7322):394–399, November 2010.

[42] P. P. Gambaryan. *How mammals run: anatomical adaptations*. Wiley, New York, 1974.

[43] M. Golubitsky, I. Stewart, P. L. Buono, et al. A modular network for legged locomotion. *Physica D: Nonlinear Phenomena*, 115(1):56–72, 1998.

[44] D. Purves, G. J. Augustine, D. Fitzpatrick, W. C. Hall, A.-S. LaMantia, and L. E. White. *Neuroscience*. Sinauer Associates, Sunderland, MA, 2 edition, 2001.

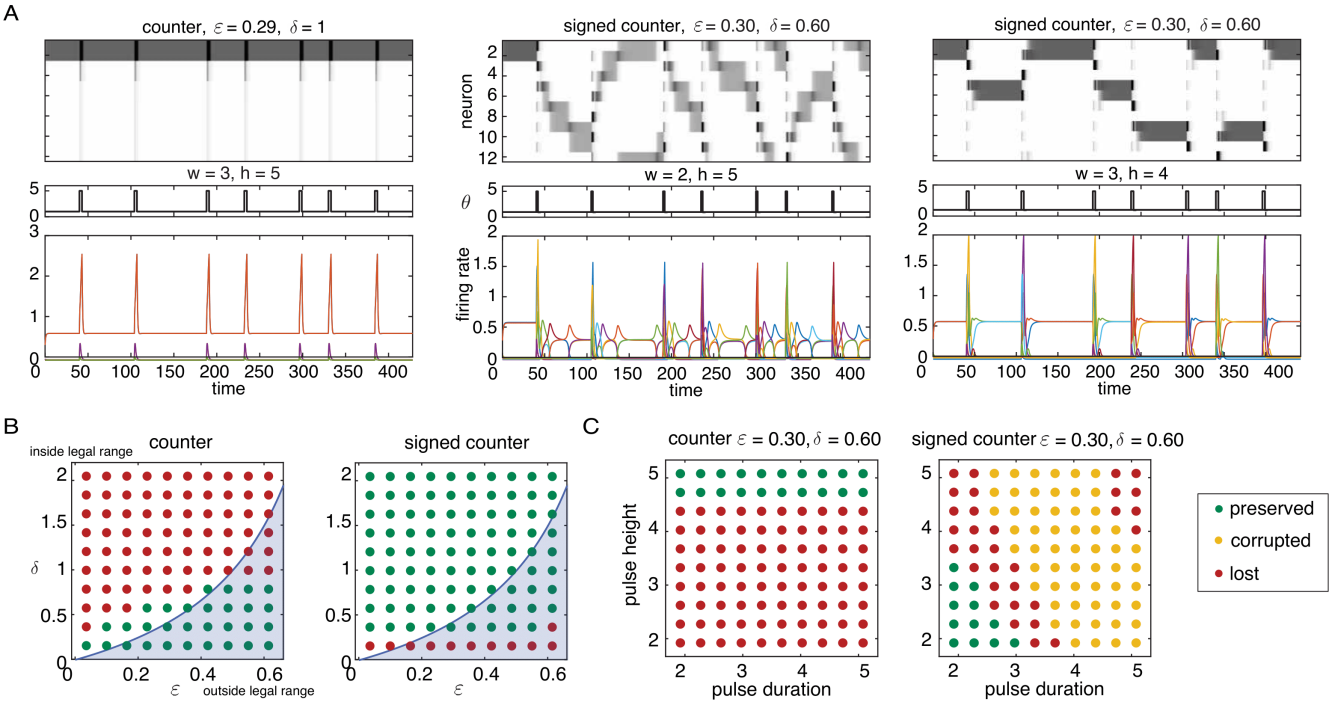
[45] T. L. Williams. A new model for force generation by skeletal muscle, incorporating work-dependent deactivation. *J. Exp. Biol.*, 213(4):643–650, February 2010.

A Supplementary materials

A.1 TLN counters

Robustness of TLN counters. In general, CTLNs are expected to perform robustly, since much of their dynamic information is influenced by $\mathrm{FP}_{\text{core}}(G)$ and we know this set is preserved across the legal range $0 < \varepsilon < \frac{\delta}{\delta+1}$ for model parameters ε, δ for certain networks.

To asses the performance of both counters across parameter space, we ran the simulations of Figure 2 using various pairs of (ε, δ) parameters, and various combinations of pulses strengths and durations. We found three different possible outcomes, exemplified in Figure S1A. We classified these according to whether the function is preserved (keeps the correct count), corrupted (counts in multiples of two) or lost (does not keep a consistent count) as $\varepsilon, \delta, \theta$ vary.



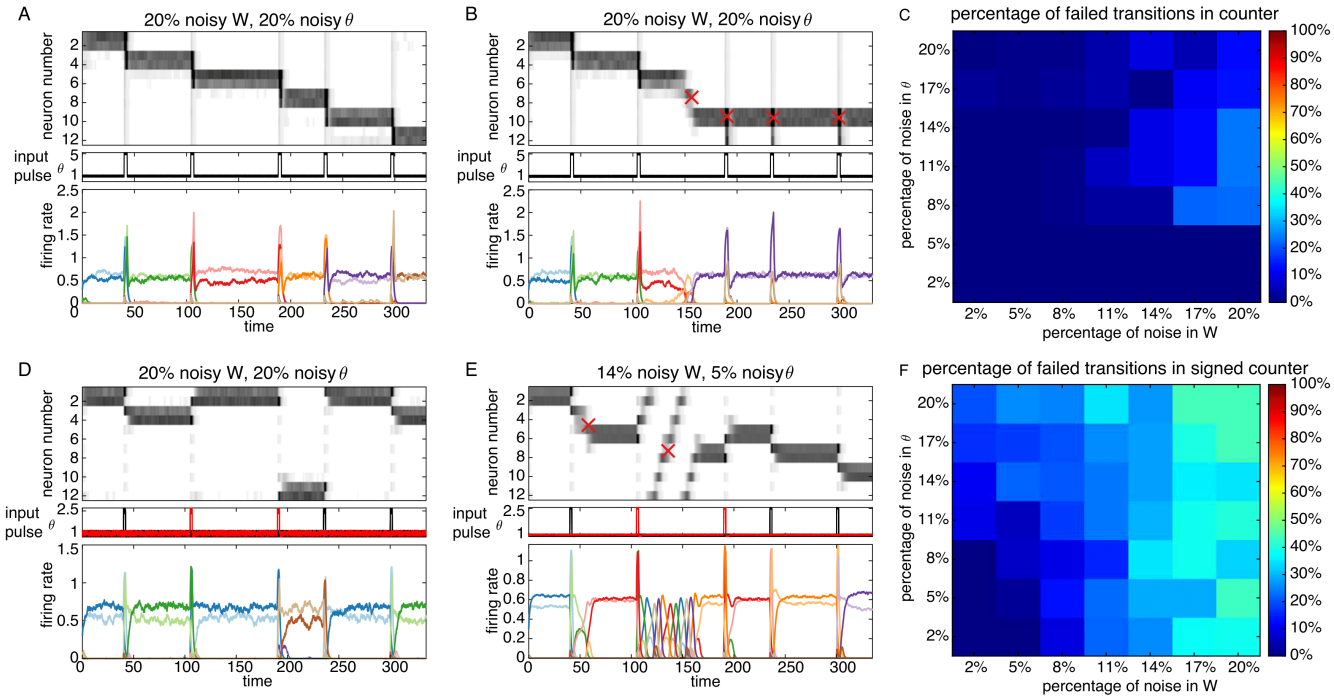
Supplementary Figure S1: Good parameter grid for TLN counters. TLN counters' behavior is classified according to whether the function is preserved (keeps the correct count, green dots), corrupted (counts in multiples of two, yellow dots) or lost (does not keep a consistent count, red dots) when $\varepsilon, \delta, \theta$ vary. (A) Examples of what can go wrong. In the first plot the pulses do not move the counter from the current fixed point (function is lost), in the second plot the signed counters enter a "roulette" behavior (function is lost), and in the third plot the counter consistently slides two positions (function is corrupted). (B) Various counter behaviors for fixed values of baseline $\theta = 1$, with pulses $\theta = 5$ for counter, $\theta = 2.5$ and $\theta = 0.95$ for signed counter. ε, δ vary according to the axes. One dot correspond to one simulation of 7 pulses, with a running time of $T = 428$ time units for the counter and $T = 435$ time units for the signed counter. Shaded in blue are parameter pairs outside the legal range $0 < \varepsilon < \frac{\delta}{\delta+1}$. Green dot indicate no failures for the entire duration of the simulation. Any failure among the 7 pulses classifies as lost or corrupted function. (C) Various counter behaviors for a fixed values of $\varepsilon = 0.30, \delta = 0.60$ and baseline $\theta = 1$. Pulse height and duration vary according to the axes.

A brief refractory period after the pulse is needed for the signed counter to stop activity from sliding forward more than one clique, as otherwise the counter could enter the rouletting behavior of Figure S1, second panel.

The results for all pairs (ε, δ) of parameters and various combinations of pulses strengths and durations are recorded in the bottom row of Figure S1, where green indicates preserved, yellow indicates corrupted and red indicates lost performance. One dot corresponds to one simulation of 7 pulses, with a running time of $T = 428$ time units for the counter and $T = 435$ time units for the signed counter. We

found that there is a good range of parameters where the counters behave as expected, successfully keeping the count of the number of input pulses received.

This robustness across CTLN parameter space is not surprising given that $\mathrm{FP}_{\text{core}}(G)$ was preserved across the space. But will the performance of the counters survive added noise in the input and connections? To verify robustness to noise, we computed the proportion of failed transitions among $m = 100$ pulses in both fixed point counters. This was done by introducing varying percentages of noise into the θ input, varied every $dt = 0.1$ of a time unit, and into the connectivity matrix W, as specified in the axes of Figure S2. A failed transition is defined as any behavior that does not advance the counter a single clique forward/backwards, depending on the sign of the pulse. Figure S2 contains examples of all possible transition failures, as well as examples of successful transitions in noisy conditions, as detailed below. All simulations used to quantify the failures were done with $\varepsilon = 0.25, \delta = 0.5$.



Supplementary Figure S2: Noisy TLN counters. (A) Example of a 20% noisy counter that performs well. (B) Example of a noisy counter with four failures, marked with red crosses: first it slides too many cliques forward, and then it gets stuck in the same clique indefinitely. (C) Percentage of failed transitions for the counter in 100 trials, for several percentages of noise in W and θ. (D) Example of a 20% noisy signed counter that performs well. (E) Example of a noisy signed counter with two failures, marked with red crosses: first it slides too many cliques forward, and then it rolls around the counter until it settles in some arbitrary clique. (F) Percentage of failed transitions for the signed counter in 100 trials, for several percentages of noise in W and θ.

Panel A of Figure S2 is an example of a very noisy counter that still preforms well. Each pulse advances the counter a single clique forward, as expected. Panel B of Figure S2 shows a noisy counter with two types of failures: first the counter advances too many steps, and then it gets stuck in one of the cliques (altogether these would count as four failures in our analysis, even though there are probably just two defective cliques). Panel D of Figure S2 is an example of a very noisy signed counter that performs perfectly well. Panel E of Figure S2 is an example of things that can go wrong in the signed counter. The first failure advances the counter one more than expected, the second failure makes the counter go into a kind of roulette behavior until it stops at a random clique. We have also observed the signed counter getting stuck at some point, as in Figure S2B.

The percentages of noise in Figure S2 were calculated as follows: The noise in the external input θ is i.i.d random noise from the interval $(-1, 1)$, and it was introduced to vary every 0.1 fraction of a unit

time. The noise in the connectivity matrix W is obtained by perturbing the adjacency matrix A with i.i.d random noise from the interval $(0, 1)$. More precisely, if denote by $\widetilde{A}, \widetilde{W}, \widetilde{\theta}$ the noisy versions of A, W, θ; where A is the transpose of the standard graph-theoretic adjacency matrix of the graph G defining the network, then the noisy versions are:

$$\begin{aligned}
\widetilde{A} &= A + pR \\
\widetilde{W} &= (-1+\varepsilon)\widetilde{A} + (-1+\delta)(\mathbf{1} - I - \widetilde{A}) \\
\tilde{\theta} &= \theta + qS
\end{aligned}$$

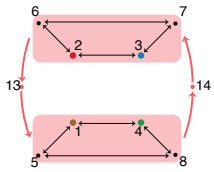
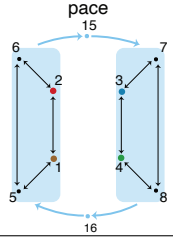
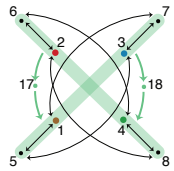
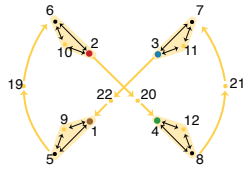
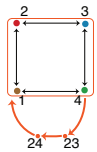
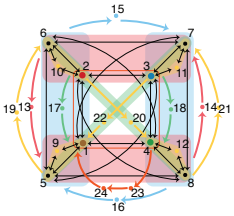
where p, q are the percentages of noise introduced, and S, R are random matrices the same size as W, θ with entries between $(0, 1)$ and $(-1, 1)$, respectively, and $\mathbf{1}$ is a matrix of 1's the same size as A. This is equivalent to perturbing W by an amount proportional to length of the interval $[-1-\delta, -1+\varepsilon]$, that is:

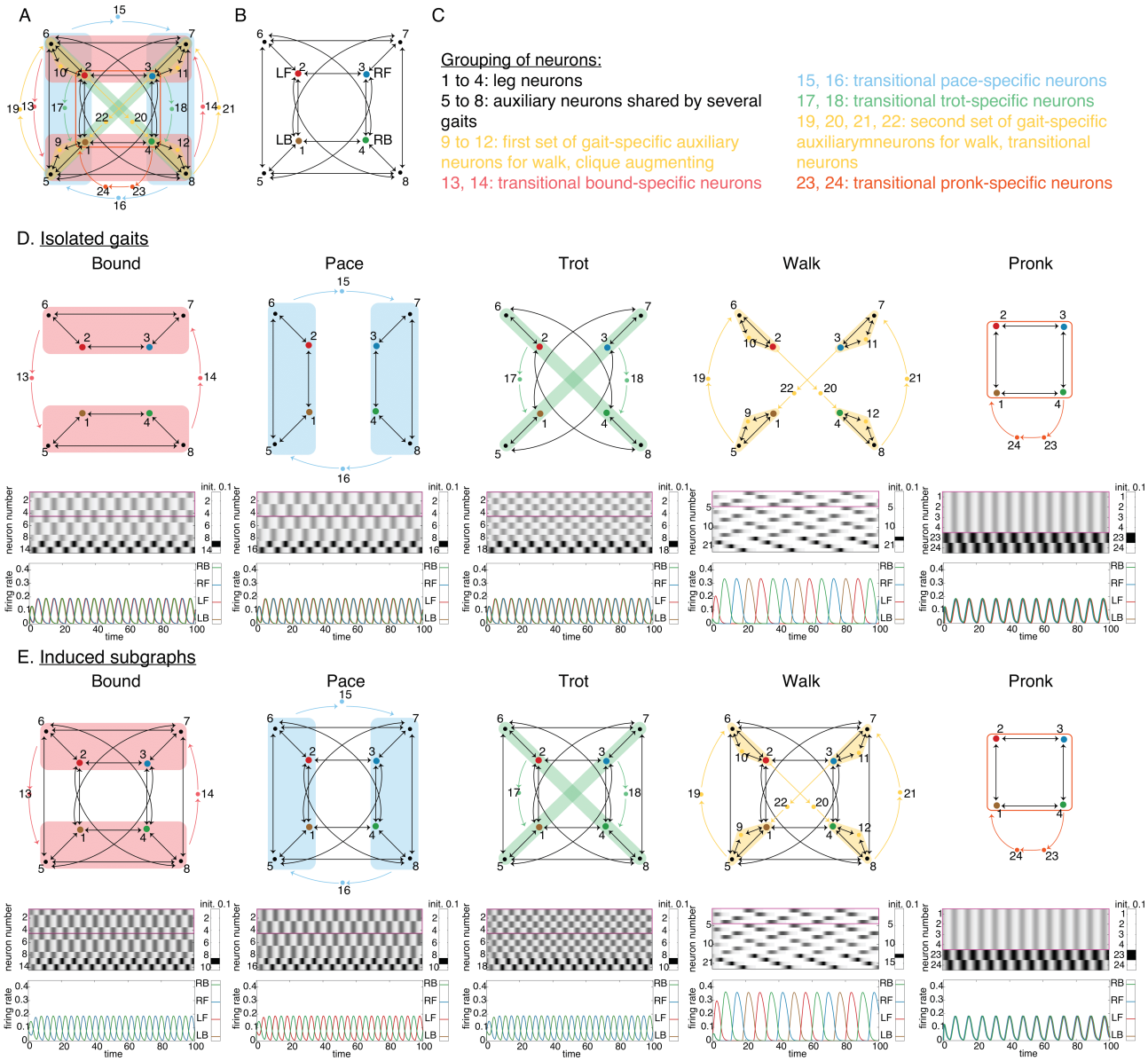
$$\begin{aligned}
\widetilde{W} &= (-1+\varepsilon)\widetilde{A} + (-1+\delta)(\mathbf{1} - I - \widetilde{A}) \\
&= W + p[(-1+\varepsilon) - (-1-\delta)]R
\end{aligned}$$

We choose to perturb the adjacency matrix instead for computational ease. For each percentage of noise pair (p, q), we ran 20 simulations, each one consisting of 5 (signed) pulses in the TLN (signed) counter, as exemplified in Figure S2. The results of counting over all these pulses are summarized in panels C and F of Figure S2. Each pixel represents the fraction of failed transitions for each pair (p, q). We found that discrete counters perform perfectly well up to 5% noise, establishing that perfect binary synapses are not necessary. When the noise limits are pushed beyond, counters lose stability and start to slide too many attractors, or to get stuck in one of the cliques. The (unsigned) counter, not surprisingly, proved to be a lot more robust, being barely affected by the amount of noise in W. Although not as robust, the signed counter accurately kept the count 50% of times under 20% noise.

A.2 Quadruped gaits

Table 2: Gait networks' $\mathrm{FP}_{\text{core}}(G)$, for individual gaits and for entire network containing all gaits from Section 2.3.

Network G	$\lvert\mathrm{FP}(G)\rvert$	$\lvert\mathrm{FP}_{\text{core}}(G)\rvert$	$\mathrm{FP}_{\text{core}}(G)$
bound 	81	16	$\sigma_1 \cup \{13\} \cup \sigma_2 \cup \{14\}$, where $\sigma_1 \in \{\{2,3\},\{2,6\},\{6,7\},\{3,7\}\}$ and $\sigma_2 \in \{\{1,4\},\{1,5\},\{5,8\},\{4,8\}\}$
pace 	81	16	$\sigma_1 \cup \{15\} \cup \sigma_2 \cup \{16\}$, where $\sigma_1 \in \{\{1,2\},\{2,6\},\{5,6\},\{5,1\}\}$ and $\sigma_2 \in \{\{3,4\},\{3,7\},\{7,8\},\{4,8\}\}$
trot 	81	16	$\sigma_1 \cup \{17\} \cup \sigma_2 \cup \{18\}$, where $\sigma_1 \in \{\{2,6\},\{2,8\},\{8,4\},\{4,6\}\}$ and $\sigma_2 \in \{\{1,5\},\{5,3\},\{3,7\},\{1,7\}\}$
walk 	1	1	$\{1,5,9\} \cup \{19\} \cup \{2,6,10\} \cup \{20\} \cup$ $\{4,8,12\} \cup \{21\} \cup \{3,7,11\} \cup \{22\}$
pronk 	9	4	$\sigma_1 \cup \{23\} \cup \{24\}$, where $\sigma_1 \in \{\{1,2\},\{2,3\},\{3,4\},\{4,1\}\}$
	875	8	$\{1,2,23,24\},\{1,4,23,24\},\{2,3,23,24\},\{3,4,23,24\},$ $\{1,6,7,8,13,16,18\},\{2,5,7,8,14,16,17\},$ $\{3,5,6,8,14,15,18\},\{4,5,6,7,13,15,17\}$



Supplementary Figure S3: Simulation of five gaits isolated (as opposed to embedded within full network), and as *induced* subgraphs of 5-gait network. Notice that the dynamics of isolated gaits and induced gaits are the same. All simulations shown here were done with $\varepsilon = 0.25, \delta = 0.5$. (A) 5-gait network. (B) Induced subgraph of the 5-gait network nodes 1 - 8. These nodes are shared by two or more gaits. (C) Grouping of neurons by type. Types are assigned on the basis of function, and color coded by gait. (D) Graphs, grayscale and rate curves of all isolated gaits. These are not induced subgraphs. (E) Graphs, grayscale and rate curves of induced subgraphs for all gaits. The attractors corresponding to each gait are still perfectly preserved.

Good parameters. A common drawback of many models is that they may require very fine-tuned values of their parameters to perform as expected. To see if this issue arises for the quadruped gait network, we assess the robustness of the network in Figure 4C. Individual gaits were designed as a cyclic union (Thm. 2) of building blocks (Fig. 3) whose $\mathrm{FP}(G)$ is parameter independent [30], meaning that its $\mathrm{FP}(G)$ set is preserved across the legal range of parameters $0 < \varepsilon < \frac{\delta}{\delta+1}$. Also, since cyclic unions of parameter independent components are also parameter independent, all isolated gaits are parameter independent as well. Indeed, it is a simple consequence of Theorem 2:

Corollary 1. Let G be a cyclic union of component subgraphs $G|_{\tau_1}, \ldots, G|_{\tau_N}$. Then $\mathrm{FP}(G)$ is parameter independent if and only if $\mathrm{FP}(G|_{\tau_i})$ is parameter independent for all $i \in [N]$.

Proof. Suppose $\sigma \in \mathrm{FP}(G, \varepsilon, \delta, \theta)$ for all $\varepsilon, \delta, \theta$ in the legal range. By Theorem 2, $\sigma \cap \tau_i \in \mathrm{FP}(G|_{\tau_i})$ for all $\varepsilon, \delta, \theta$ in the legal range and all $i \in [N]$, meaning that $\mathrm{FP}(G|_{\tau_i})$ is also parameter independent. The other implication is analogous. □

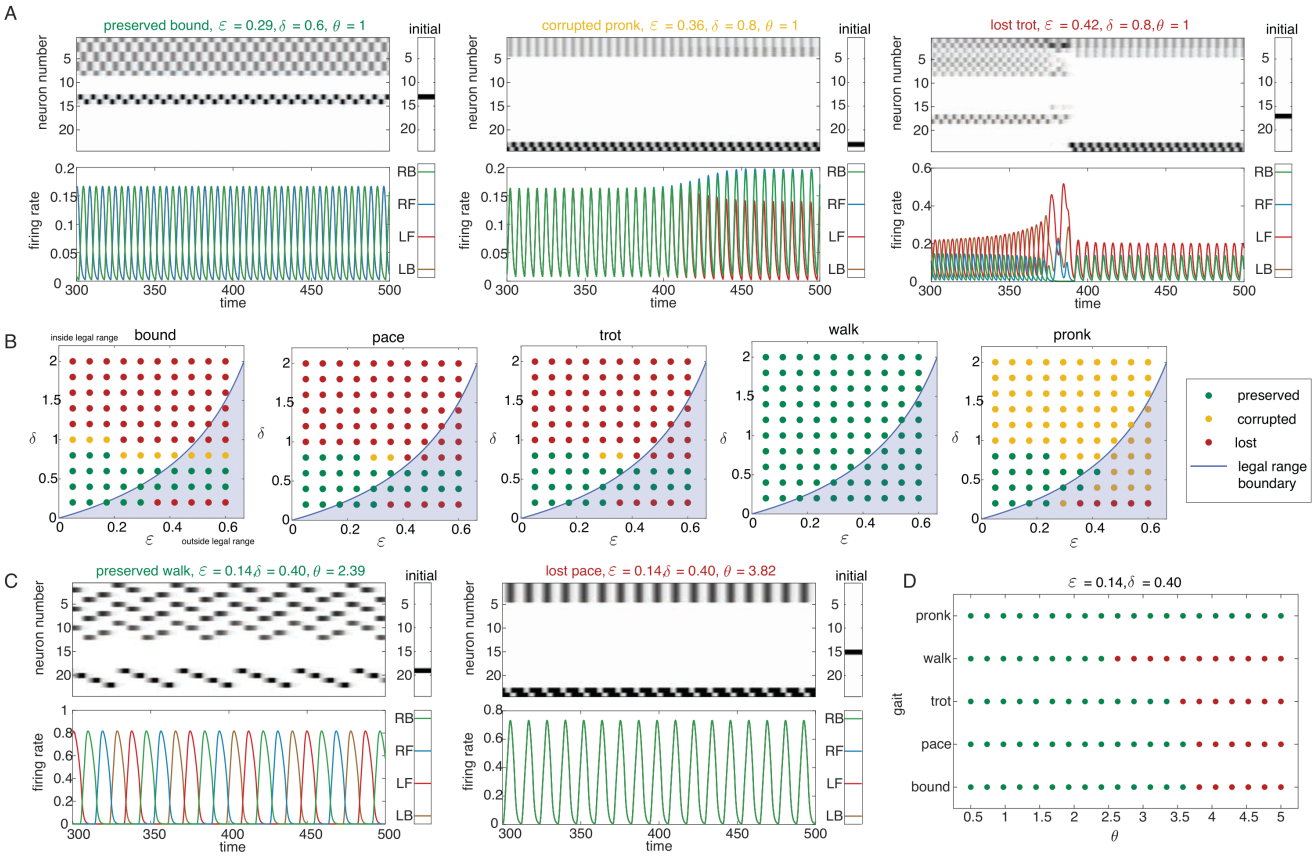
This implies that the $\mathrm{FP}(G)$ set does not vary with parameter changes within the legal range, and so we can anticipate the gait attractors to be present in each individual gait network for a wide range of parameter values. Indeed each gait, *isolated*, turns out to be very robust. The same unfortunately cannot be said about the glued five-gait network, since it is not truly a cyclic union, but rather a gluing of cyclic unions, for which we do not have theoretical results yet. This begs the question: how sensitive is the network to the choice of $\varepsilon, \delta, \theta$ parameters? Are the five attractors still present, and easily accessible, if we vary the parameter values?

To answer this question, we tried to reproduce the simulations of Figure 4B using different $\varepsilon, \delta, \theta$ parameters. First, we observed how the attractor corresponding to each gait behaved under several values of ε, δ in the five-gait network. We found that there were three different possibilities: either the attractor is *preserved*, meaning that it is identical in qualitative behavior to the attractors in Figure 4B (up to a change in amplitude or period); *corrupted*, indicating that the auxiliary neurons corresponding uniquely to that gait are still active and firing cyclically, but the leg nodes have lost their symmetry; or *lost*, when the auxiliary neurons corresponding uniquely to that gait are not firing cyclically (this usually translated into a gait becoming a different attractor, which could either be another gait, a stable fixed point or a chaotic-looking attractor).

An example of each of these outcomes is exemplified in Figure S4A, where only the last 200 time units of the simulation are shown, for clarity. In the left example, the network was initialized to the bound gait, and the gait is perfectly preserved. In the middle plot, the network was initialized to the pronk gait, and even though the auxiliary neurons corresponding to pronk are still active, the leg neurons lost their symmetry towards the end. In the right plot, the network was initialized to the trot gait, but near the end of the simulation the network settles into a corrupted form of pronk, as indicated by the auxiliary neurons of pronk being active. The results of all simulations are summarized in Figure S4B. Green denotes preserved attractor, yellow denotes corrupted attractor and red denotes lost attractor. The intersection of the green dotted regions is the range in which all gaits are preserved: $0.05 < \varepsilon < 0.23$, $0.2 < \delta < 0.6$, for at least the first 500 units of time. These are the "good parameters".

We then did the same analysis but varying the θ parameter only. Figure S4C shows examples of what can happen in this case. In the first example, the network was initialized to the walk gait, and the gait is perfectly preserved. In the second example, on the other hand, the network was initialized to the pace gait, but immediately went into pronk. Figure S4D shows the same summary of results, but carried out for constant values $\varepsilon = 0.14$, $\delta = 0.40$, and varying θ. These values were arbitrarily chosen from the $0.05 < \varepsilon < 0.23$, $0.2 < \delta < 0.6$ good parameters from above.

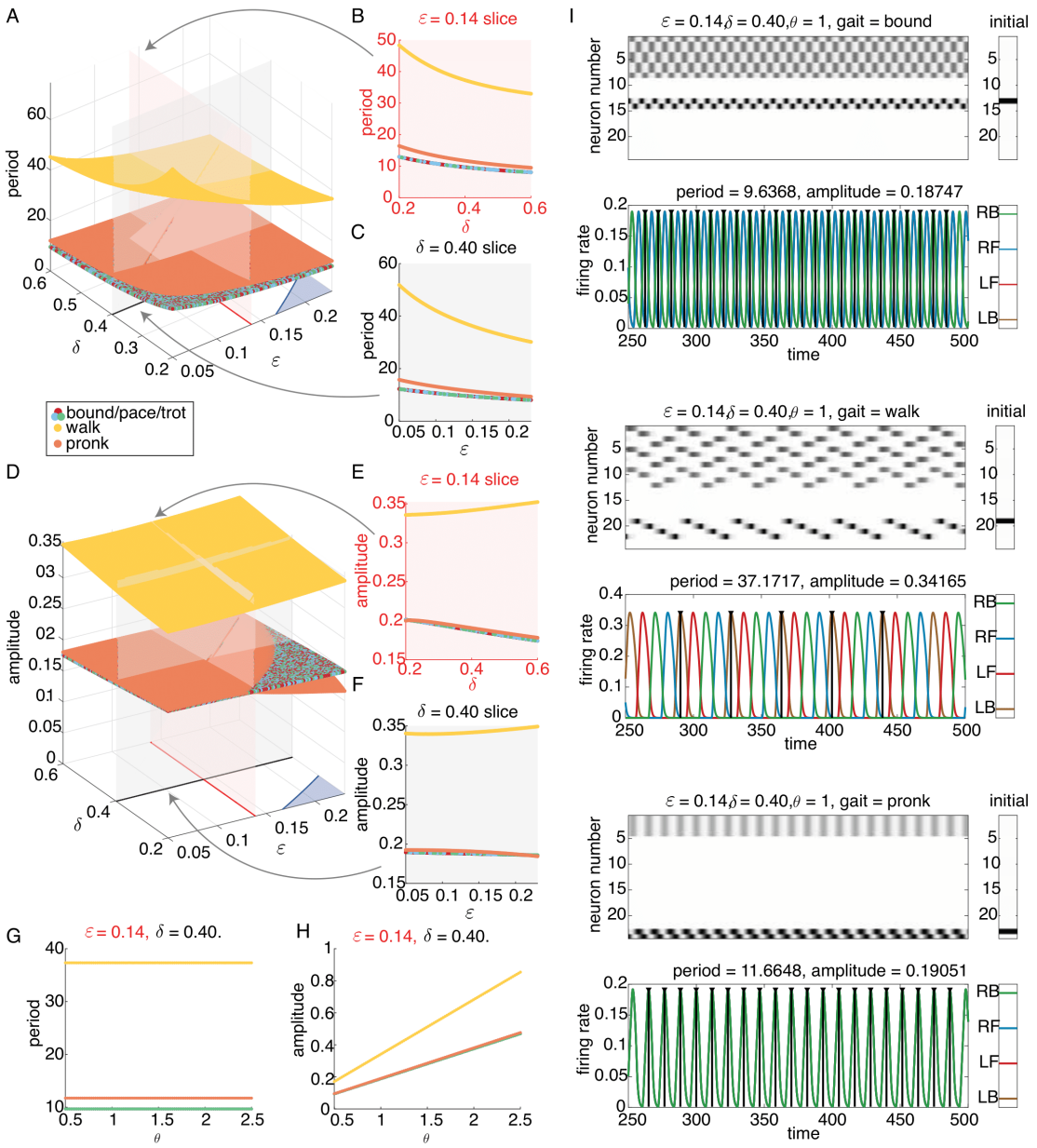
In conclusion, we have computationally found that all the attractors corresponding to the five gaits in our network are preserved for all $0.05 < \varepsilon < 0.23$, $0.2 < \delta < 0.6$ and $0.5 < \theta < 2.5$, for at least 500 units of time. Having a range of parameters available raises the question: what role do the parameters



Supplementary Figure S4: Gaits survival under several values of $\varepsilon, \delta, \theta$. All plots were generated with a running time of 500 time units. (A) Three examples of what can happen to an attractor under three different parameter values. Left: the network was initialized on the bound gait, and the gait is perfectly preserved. Middle: the network was initialized on the pronk gait, and near the end of the simulation, the auxiliary neurons corresponding to pronk are still active, but the limb neurons lost their symmetry. Right: the network was initialized on the trot gait, but near the end of the simulation the network settles into a corrupted form of pronk, as indicated by the auxiliary neurons of pronk being active. (B) Gaits are classified according to whether the attractors are either preserved (green dots), corrupted (yellow dots) or lost to a different attractor (red dots) when ε, δ vary. All the dots were generated using $\theta = 1$ fixed. (C) Two examples of what can happen to an attractor under two different θ parameter values. Left: the network was initialized on the walk gait, and the gait is perfectly preserved. Right: the network was initialized on the pace gait, but immediately went into pronk. (D) Gaits are classified according to whether the attractors are either preserved (green dots), corrupted (yellow dots) or lost to a different attractor (red dots) when θ varies. All the dots were generated using $\varepsilon = 0.14$ and $\delta = 0.40$.

play in modulating the characteristics of the gaits such as period and amplitude? These are important questions because period and amplitude in firing rates affect muscle tension [44, 45].

Parameter modulation. Next, we compute the period and amplitude of each gait within the five-gait network, using the good parameters from Figure S4. To do so, we ran a single simulation for $T = 500$ time units and then we used the MATLAB function findpeaks to find the location and value of the peaks of a single rate curve. Examples of this can be seen in Figure S5I. The black bars in the rate curves show the amplitude, and the black triangles mark the time of the peaks, both calculated using only the first leg ($x_1(t)$ - LB). We found these peaks only during the last half of the simulation to avoid taking into account the time before settling into the attractor. To find the periods we subtracted all the consecutive peak locations. We had to get rid of the first and last period and amplitude data point because they became corrupted by chopping the curve. Then we averaged the periods and peak values to obtain a single period and a single amplitude per simulation. This is the value shown in Figures S5A-H.



Supplementary Figure S5: Amplitude and period of the gaits in the 5-gait network (Figure 4C), as controlled by $\varepsilon, \delta, \theta$. Gaits are color coded. Values were found using the last half (once the attractor has settled) of a simulation with running time $T = 500$. (A) Values of period for a $(0.05, 0.23) \times (0.2, 0.6)$ grid of (ε, δ) values and all gaits. The legal range boundary $\varepsilon = \frac{\delta}{\delta+1}$ is shown in blue in the (ε, δ)-plane. The values above the blue shaded area are outside the legal range, but the gait attractors are preserved nonetheless. The data points for bound, pace and trot are superimposed. (B) Cross section of plot in panel A corresponding to $\varepsilon = 0.14$. (C) Cross section of plot in panel A corresponding to $\delta = 0.40$. (D) Values of amplitude for a $(0.05, 0.23) \times (0.2, 0.6)$ grid of (ε, δ) values and all gaits. The legal range boundary $\varepsilon = \frac{\delta}{\delta+1}$ is shown in blue in the (ε, δ)-plane. The values above the blue shaded area are outside the legal range, but the gait attractors are preserved nonetheless. The data points for bound, pace and trot are superimposed. (E) Cross section of plot in panel D corresponding to $\varepsilon = 0.14$. (F) Cross section of plot in panel D corresponding to $\delta = 0.40$. (G) Values of period for the cross sections in panels B and C ($\varepsilon = 0.14, \delta = 0.40$) and varying values of θ. (H) Values of amplitude for the cross sections in panels E and F ($\varepsilon = 0.14, \delta = 0.40$) and varying values of θ. (I) Three examples of how period and amplitude were computed. The black bars show the amplitude and the black triangles mark the time of the peaks, both calculated using only the first limb (LB).

There, each point corresponds to a single simulation. The color of the point corresponds to a given gait, as specified in the label of Figure S5A. Note that, since bound, pace and trot have virtually the

same structure, the data point for these are overlapped in Figures S5A-H. We also show cross sections for Figures S5A,D. The blue shaded corner of the 3-dimensional plots corresponds to values outside the legal range. In Figures S5A-F we are varying only ε and δ, with a fixed value of $\theta = 1$. In Figures S5G-H, we use the fixed values of ε, δ in the slices above to vary the value of θ.

Notice that for fixed values of $\varepsilon, \delta, \theta$, period decreases with ε and δ; whereas amplitude increases for walk, and decreases for all other gaits. Period is seen to be invariant to θ changes, and amplitude increases with increasing values of θ. It is known that an increased firing rate produces an increase in muscle tension by summation of more frequent successive motor contractions [44, 45]. This implies that in our model, period would correspond to how fast/slow muscles contract and relax (bigger period means muscles contract and relax faster). The period therefore is controlling how fast the movement is executed, that is, the speed of the gait. Amplitude of firing rate corresponds to tension in muscle, because the more action potentials, the more muscle units are activated.

But while we have showed that there is a fairly wide range of parameters under which our network behaves as expected, and where we can modulate the properties of the firing rates, there is still the assumption that all synaptic connections are perfect and identical: either $-1 + \varepsilon$ or $-1 - \delta$. Is our network robust to noise in the synapses? What about the inputs? Are transitions still possible in the presence of noisy connections?

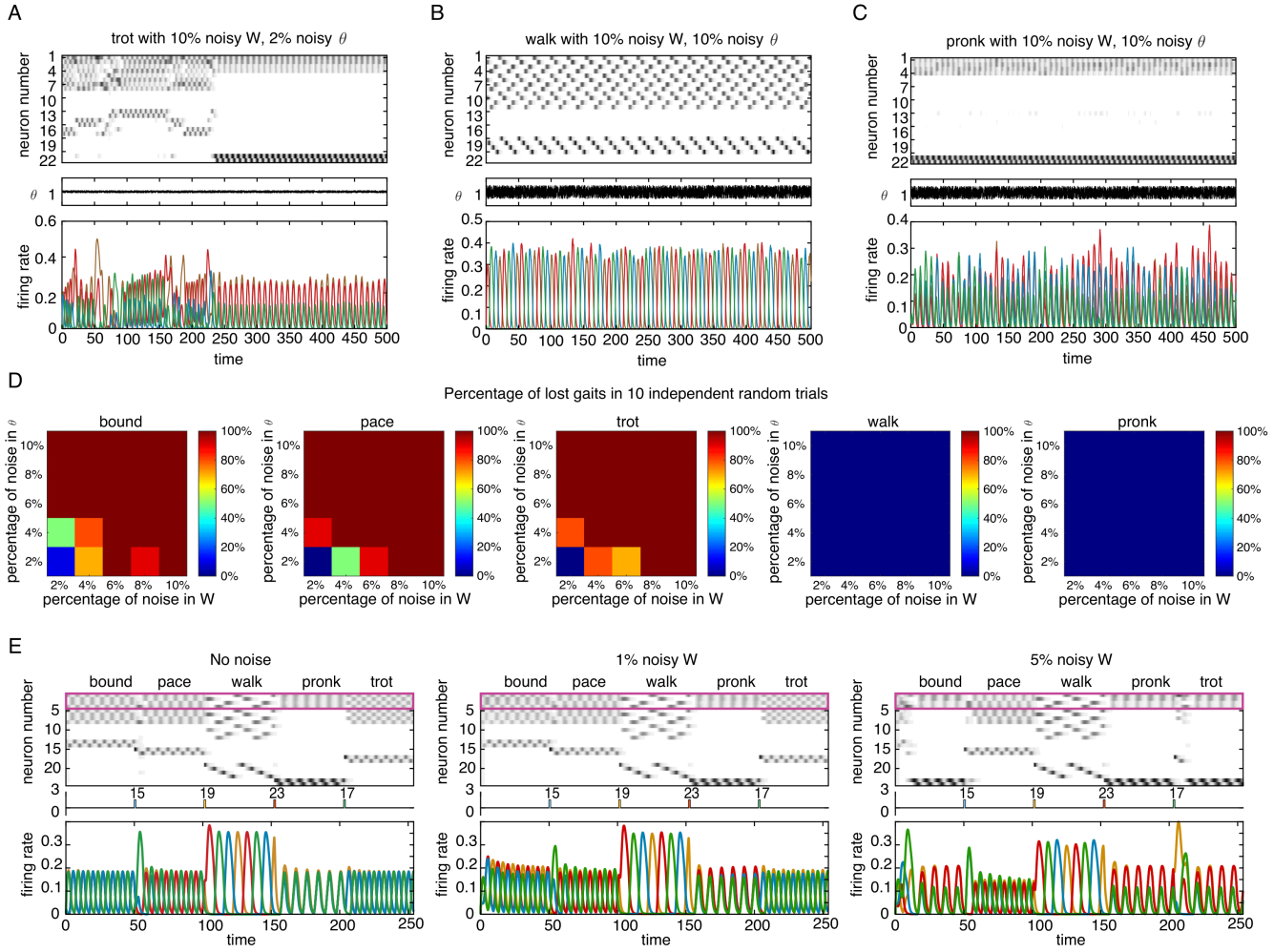
Robustness to noise To explore these questions, we added uniform noise to W and θ and computationally tested whether the attractor was lost. A gait is said to be lost when the attractor corresponding to said gait degenerates into a different attractor, which is clear from which auxiliary neurons are active. We did not measure if the attractor was corrupt or not, because any amount of noise will break the symmetry of the legs.

This was done in the same way as we did for the TLN counters. We ran a single noisy simulation and observed if the attractor was lost or not. Examples of the behaviors we observed can be found in Figure S6A-C. In A, the network was initialized to trot, but it chaotically jumps between different attractors, until it settles into pronk. In B, the network was initialized to walk, and it remains there, as well as it can. This means, the legs lost their symmetry, but the high-firing neurons are the same as in walk, in the same order. This classifies as not lost in our analysis. In C, we observe the same situation but with pronk. Here, the asymmetry of the legs is more notable. These are wobbly gaits.

Figure S6D summarizes our findings. We simulated each gait 10 times, and then calculated the percentage of lost gaits in those 10 trials, for varying percentages of noise in W and θ. The color of the pixel in Figure S6D represents the percentage of lost gaits out of these 10 trials. A trial corresponds to a single noisy pair (W, θ), which was used to test all gaits once. All simulations used to quantify the failures were done with $\varepsilon = 0.25, \delta = 0.5$ and θ-noise with $dt = 0.1$.

In around 90% of trials, gaits were not lost at 2% noisy W and θ, indicating that that the perfect binary synapses of CTLNs are not necessary for this network to produce the desired gaits. Larger levels of noise, however, degrade the network's performance significantly. Interestingly, at bigger levels of noise, when gaits were lost, the network most commonly settled into the pronk gait. Pronk also happens to be one of the two most robust gaits (see pronk in Figure S6D). This supports our suspicion that pronk's basin of attraction is bigger. This is perhaps not surprising given how different the construction of pronk is (as well as walk) from the three less robust gaits: bound, pace, and trot. This raises the question of what would happen to the robustness of the network if there were no pronk or walk, so that the network is perfectly symmetric.

Finally, we also tested the ability of the network to transition under 1% and 5% noise in W, as shown in Figure S6E. Transitions are robust at 1% added noise, but past 5% gaits are commonly lost to pronk and thus transitions are not successful.



Supplementary Figure S6: Percentage of lost gaits in noisy 5-gait networks. All simulations used to quantify the failures were done with $\varepsilon = 0.25, \delta = 0.5$ and noise time-scale for θ $dt = 0.1$. We did not measure if the attractor was corrupt or not, because any amount of noise will break the symmetry of the limbs. (A) Example of lost trot. The network chaotically jumps between different attractors, until it settles in pronk. (B) Example of a very noisy network, where walk is not lost, but it is corrupt. (C) Example of a very noisy network, where pronk is not lost, but it is corrupt. (D) Percentage of lost gaits in 10 trials, for several percentages of noise in W and θ. A trial corresponds to a single matrix, which was used to test all gaits once. (E) Transitions for two values of noise in W. Transitions are robust at 1% added noise, but past 5% gaits are commonly lost to pronk and thus transitions are not successful.

A.3 Proofs of Section 2.5

Definition 5. We say that (W, b) is a *competitive* TLN on n neurons if $W_{ij} \leq 0$ and $W_{ii} = 0$ for all $i, j \in [n]$.

Note that this definition is different from the one in [30], which also requires $b_i \geq 0$ for all $i \in [n]$.

Definition 6. We say that a TLN (W, b) is *non-degenerate* if

- $b_i > 0$ for at least one $i \in [n]$
- $\det(I - W_\sigma) \neq 0$ for each $\sigma \subseteq [n]$, and
- for each $\sigma \subseteq [n]$ such that $b_i > 0$ for all $i \in \sigma$, the corresponding Cramer's determinant is nonzero: $\det((I - W_\sigma)_i; b_\sigma) \neq 0$.

Lemma 7. *Let (W, b) be a competitive, nondegenerate TLN, and suppose $b_k \leq 0$ for some $k \in [n]$. Then $x_k(t) \to 0$ as $t \to \infty$. In particular,*

$$\mathrm{FP}(W, b) = \mathrm{FP}(W_{[n]\setminus\{k\}}, b_{[n]\setminus\{k\}}),$$

where $(W_{[n]\setminus\{k\}}, b_{[n]\setminus\{k\}})$ is the restricted subnetwork obtained by removing node k from (W, b).

Proof. Suppose $b_k \leq 0$ for neuron k. Then we have that $\left[\sum_{j=1}^n W_{kj} x_j + b_k\right]_+ = 0$, since $W_{kj} \leq 0$ for all $k, j \in [n]$. Thus $\frac{dx_k}{dt} = -x_k$, and so $x_k \to 0$ as $t \to \infty$. Hence at any fixed point, we must have $x_k^* = 0$. Thus, if $\sigma \in \mathrm{FP}(W, b)$, then $k \notin \sigma$.

Denote $\widehat{W} = W_{[n]\setminus\{k\}}$, $\widehat{b} = b|_{[n]\setminus\{k\}}$. We now show that $\mathrm{FP}(W, b) \subseteq \mathrm{FP}(\widehat{W}, \widehat{b})$, and vice versa. Let $\sigma \in \mathrm{FP}(W, b)$, and let x^* be the associated fixed point. As shown above, $x_k^* = 0$. Now let $\widehat{x}^* := x^*|_{[n]\setminus\{k\}}$. It is easy to check that $\widehat{x}^*$ is a fixed point of $(\widehat{W}, \widehat{b})$. Indeed, for all $i \neq k$ we have that

$$\begin{aligned}
\left.\frac{d\widehat{x}_i}{dt}\right|_{\widehat{x}=\widehat{x}^*} &= -\widehat{x}_i^* + \left[\sum_{j\in[n]\setminus\{k\}} \widehat{W}_{ij}\widehat{x}_j^* + \widehat{b}_i\right]_+ = -x_i^* + \left[\sum_{j\in[n]\setminus\{k\}} W_{ij}x_j^* + b_i\right]_+ \\
&= -x_i^* + \left[\sum_{j\in[n]} W_{ij}x_j^* + b_i\right]_+ = \left.\frac{dx_i}{dt}\right|_{x=x^*} = 0.
\end{aligned}$$

To see that $\mathrm{FP}(\widehat{W}, \widehat{b}) \subseteq \mathrm{FP}(W, b)$, suppose $\sigma \in \mathrm{FP}(\widehat{W}, \widehat{b})$ and let $\widehat{x}^*$ be the associated fixed point. Define x^* such that $x_i^* = \widehat{x}_i^*$ for all $i \neq k$, and $x_k^* = 0$. Clearly, $\mathrm{supp}(x^*) = \sigma$ as well. We now check that x^* is a fixed point of (W, b). Indeed, for $i \neq k$ we have:

$$\begin{aligned}
\left.\frac{dx_i}{dt}\right|_{x=x^*} &= -x_i^* + \left[\sum_{j\in[n]} W_{ij}x_j^* + b_i\right]_+ = -x_i^* + \left[\sum_{j\in[n]\setminus\{k\}} W_{ij}x_j^* + b_i\right]_+ \\
&= -\widehat{x}_i^* + \left[\sum_{j\in[n]\setminus\{k\}} \widehat{W}_{ij}\widehat{x}_j^* + \widehat{b}_i\right]_+ = \left.\frac{d\widehat{x}_i}{dt}\right|_{\widehat{x}=\widehat{x}^*} = 0.
\end{aligned}$$

Moreover, since $dx_k/dt = -x_k$ in (W, b), evaluating at x^* we obtain:

$$\left.\frac{dx_k}{dt}\right|_{x=x^*} = -x_k^* = 0.$$

Thus, x^* is a fixed point of (W, b), and hence $\sigma \in \mathrm{FP}(W, b)$. □

Theorem 8 (Restatement of Theorem 3)**.** *Consider a TLN* (W, b) *where*

$$W = \left[\begin{array}{c|c|c} W^{(1)} & * & 0 \\ \hline * & W^{(2)} & * \\ \hline 0 & * & W^{(3)} \end{array}\right], \tag{9}$$

and

$$b = \left(\begin{array}{c} b^{(1)} \\ \hline b^{(2)} \\ \hline b^{(3)} \end{array}\right), \tag{10}$$

and suppose that $(W^{(1)}, b^{(1)})$ *is a network with* k *stable fixed points and no other attractors;* $(W^{(2)}, b^{(2)})$ *is an independent set on* ℓ *nodes; and* $(W^{(3)}, b^{(3)})$ *is an arbitrary recurrent network with attractors* $\mathcal{A}_1, \ldots, \mathcal{A}_m$*.*

Then, if $b^{(2)} \leq 0$*, the attractors of the network* (W, b) *include the* $m \cdot k$ *fusion attractors consisting of one stable fixed point attractor coming from* $(W^{(1)}, b^{(1)})$ *fused with one* $\mathcal{A}_j$ *attractor coming from* $(W^{(3)}, b^{(3)})$*.*

Proof. Define the (i, j)-th fusion attractor as

$$\mathcal{A}^{(i,j)} = \{\vec{x}^*_{(i)}\} \times \{0\} \times \mathcal{A}_j = \left\{(\vec{x}^*_{(i)}, \vec{0}, \vec{z}) \mid \vec{z} \in \mathcal{A}_j\right\} \subseteq \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3},$$

where $\vec{x}^*_{(i)}$ denotes the i-th stable fixed point of $(W^{(1)}, b^{(1)})$.

We will show that $\mathcal{A}^{(i,j)}$ is an attractor of (W, b), i.e., it is a minimal attracting forward-invariant subset of $\mathbb{R}^{n_1+n_2+n_3}$.

Let $F_t(x(t_0), y(t_0), z(t_0))$ denote the trajectory at time t given an initial condition at t_0, so that

$$F_t(x(t_0), y(t_0), z(t_0)) = (x(t), y(t), z(t)).$$

Forward invariance: We must prove that for any initial condition $(x(t_0), y(t_0), z(t_0)) \in \mathcal{A}^{(i,j)}$, the trajectory remains in $\mathcal{A}^{(i,j)}$ for all $t \geq t_0$.

First, note that because $b^{(2)} \leq 0$, and W is competitive, everything inside the nonlinearity is 0 for layer 2 and the dynamics reduce to exponential decay. Since $y(t_0) = 0$ already, it follows that $F_t(y(t_0)) = 0$ for all $t \geq t_0$.

Next, note that layer 1 only receives inputs from layer 2 and itself. Since $F_t(y(t_0)) = 0$, there are no external inputs to layer 1 that can kick the activity out of the initial fixed point of $(W^{(1)}, b^{(1)})$, and thus $F_t(x(t_0)) = x^*_{(i)}$ for all $t \geq t_0$.

Finally, using the same argument as above (no input from external layers) and noting that $z(t_0) \in \mathcal{A}_j$ (which is an attractor of $(W^{(3)}, b^{(3)})$), we get that $F_t(z(t_0)) \in \mathcal{A}_j$ for all $t \geq t_0$.

Thus, we have shown that $\mathcal{A}^{(i,j)}$ is forward-invariant.

Attracting neighborhood: Since $x^*_{(i)}$ and $\mathcal{A}_j$ are both attractors, they each have attracting neighborhoods $B(x^*_{(i)})$ and $B(\mathcal{A}_j)$. Moreover, since $b^{(2)} \leq 0$, we have $F_t(y(t_0)) \to 0$ as $t \to \infty$ for all initial conditions, by Lemma 7.

We claim that $B(x^*_{(i)}) \times B_\varepsilon(0) \times B(\mathcal{A}_j)$ is an attracting neighborhood of $\mathcal{A}^{(i,j)}$ for some suitably chosen ε. Indeed, it is possible to choose ε small enough that the sum of $x^*_{(i)}$ with the ε input from layer 2 still falls within the attracting neighborhood $B(x^*_{(i)})$. That is, a choice of ε is such that the perturbation to the dynamics of layer 1 is small enough for the trajectory to converge to the stable fixed point $x^*_{(i)}$. A similar argument applies to the third layer, as the only external inputs it recieves come from layer 2, which can be bounded arbitrary close to 0. Taken together, we see that for $t > M$, $F_t(x(t_0), y(t_0), z(t_0)) \in \mathcal{A}^{(i,j)}$.

Minimality: Suppose there exists a smaller subset $\mathcal{A}' \subset \mathcal{A}^{(i,j)}$ that is also an attractor. By the structure of $\mathcal{A}^{(i,j)}$, $\mathcal{A}'$ must have the form $\{x^*_{(i)}\} \times \{0\} \times A'_j$ for some proper subset $A'_j \subset \mathcal{A}_j$. Since the dynamics of

layer 3 only involve $W^{(3)}$ inputs in these sets (since layer 2 nodes are 0), the vector field in A'_j matches that of $\mathcal{A}_j$, implying that A'_j is an attractor of $(W^{(3)}, b^{(3)})$, smaller than $\mathcal{A}_j$, contradicting the minimality of $\mathcal{A}_j$ as an attractor of $(W^{(3)}, b^{(3)})$.

$\square$