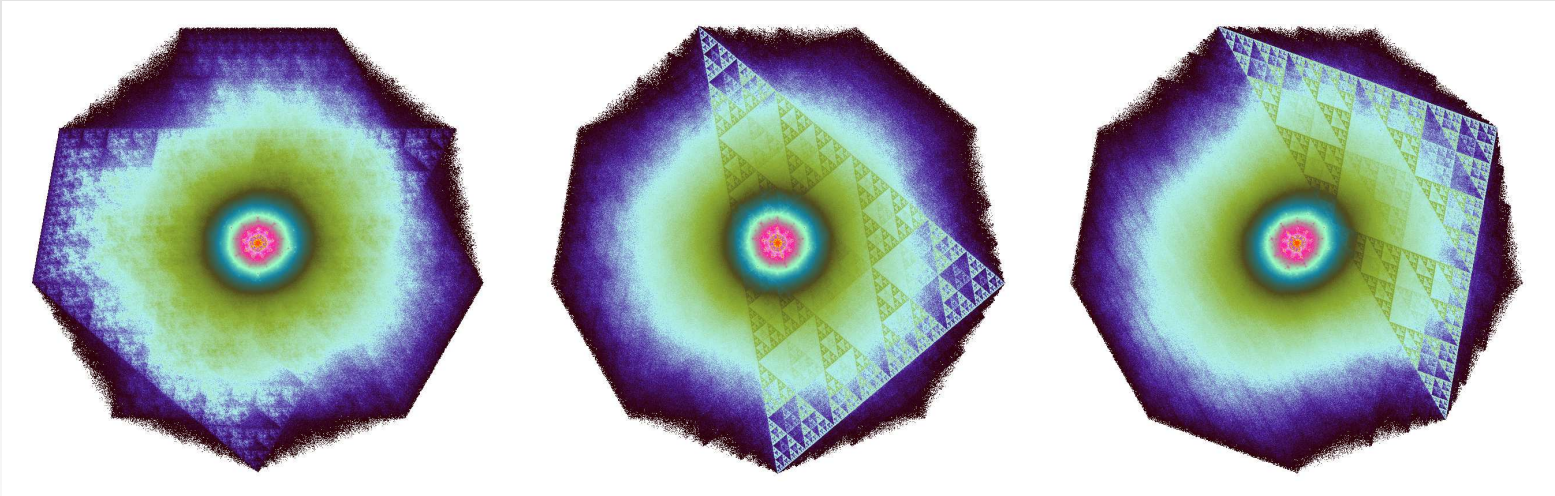
Dynamics on Asynchronous Networks

Mike Field

Chris Bick & Anushaya Mohapatra

Rice University

Research supported in part by NSF Grants DMS-0806321 & DMS-1265253



September, 2013.

Topics

Topics

- Classical dynamics, synchronous networks.

Topics

- Classical dynamics, synchronous networks.
- Contemporary problems; asynchronous networks.

Topics

- Classical dynamics, synchronous networks.
- Contemporary problems; asynchronous networks.

Caution: *Synchronous* and *Asynchronous* may not mean what you think...

Topics

- Classical dynamics, synchronous networks.
- Contemporary problems; asynchronous networks.

  Caution: *Synchronous* and *Asynchronous* may not mean what you think...

- Examples of asynchronous networks.

Topics

- Classical dynamics, synchronous networks.
- Contemporary problems; asynchronous networks.

  Caution: *Synchronous* and *Asynchronous* may not mean what you think...

- Examples of asynchronous networks.
- Dynamics on asynchronous networks and some outstanding challenges.

Classical dynamics

$$\begin{aligned} \mathbf{x}' &= f(\mathbf{x}) \\ \mathbf{x}_{n+1} &= f(\mathbf{x}_n), \quad n \geq 0. \end{aligned}$$

Typically, f is assumed to be *real analytic* or even a *polynomial*.

Examples:

Celestial mechanics (from 1687).

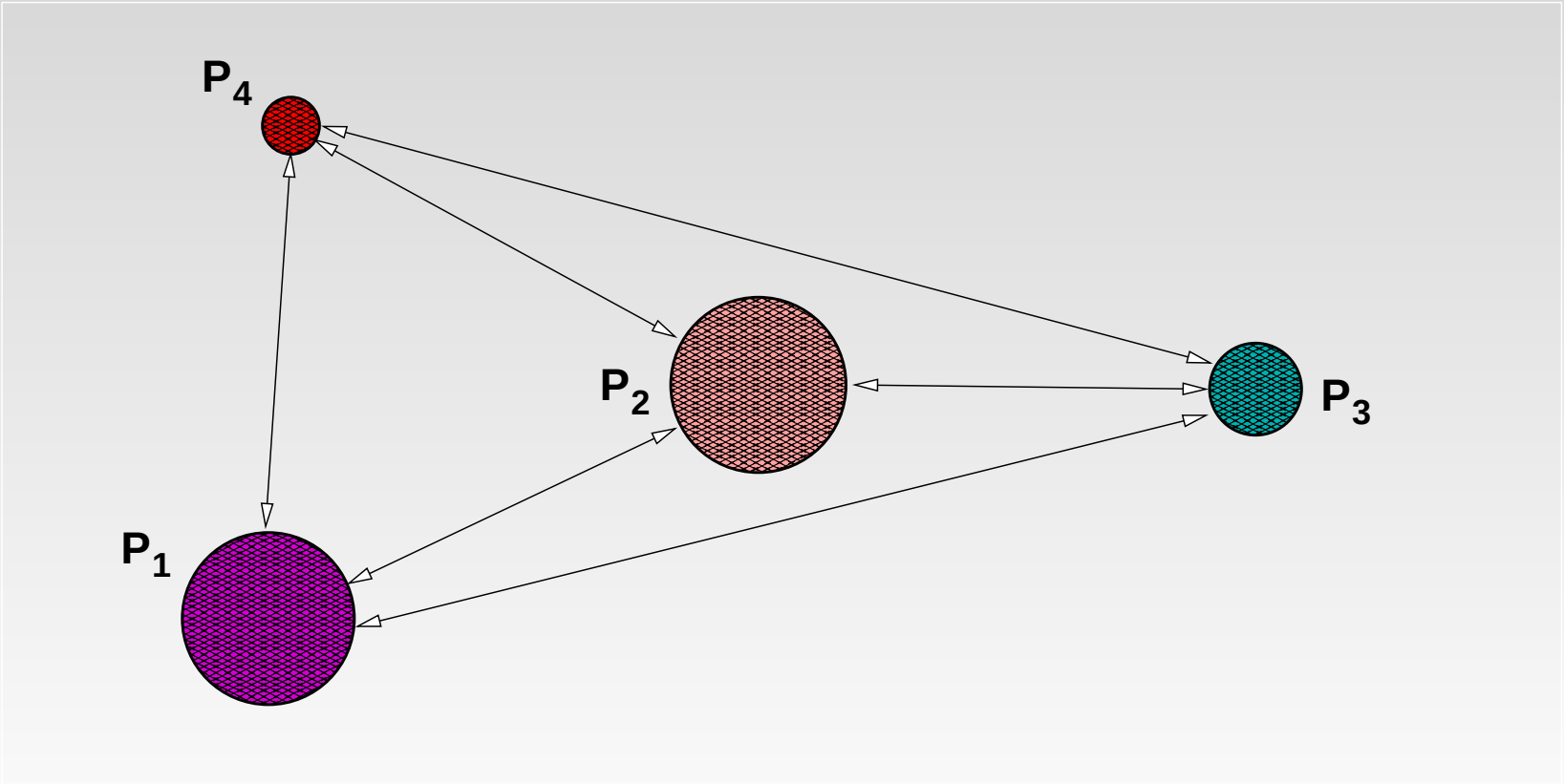
Nonlinear oscillators (1920, Van der Pol)

Chaotic dynamics (1963, Lorenz)

Networks

In dynamics it is often natural to group variables together leading to the concept of a network.

Example: N-body problem of celestial mechanics



Network graph for the 4-body problem

Equations

In case $N = 4$:

$$\begin{aligned} \mathbf{X}_1' &= F_1(\mathbf{X}_1; \mathbf{X}_2 - \mathbf{X}_1, \cdots, \mathbf{X}_4 - \mathbf{X}_1) \\ \cdots &= \cdots \\ \mathbf{X}_4' &= F_4(\mathbf{X}_4; \mathbf{X}_1 - \mathbf{X}_4, \cdots, \mathbf{X}_3 - \mathbf{X}_4), \end{aligned}$$

where $\mathbf{X}_i = (x_1, x_2, x_3, v_1, v_2, v_3)$ and the F_i are real analytic.

Equations

In case $N = 4$:

$$\begin{aligned} \mathbf{X}_1' &= F_1(\mathbf{X}_1; \mathbf{X}_2 - \mathbf{X}_1, \cdots, \mathbf{X}_4 - \mathbf{X}_1) \\ \cdots &= \cdots \\ \mathbf{X}_4' &= F_4(\mathbf{X}_4; \mathbf{X}_1 - \mathbf{X}_4, \cdots, \mathbf{X}_3 - \mathbf{X}_4), \end{aligned}$$

where $\mathbf{X}_i = (x_1, x_2, x_3, v_1, v_2, v_3)$ and the F_i are real analytic.

- In case $N = 1$, can reduce to a constant – zero dimensional (relative to a rotating coordinate frame).

Equations

In case $N = 4$:

$$\begin{aligned} \mathbf{X}_1' &= F_1(\mathbf{X}_1; \mathbf{X}_2 - \mathbf{X}_1, \cdots, \mathbf{X}_4 - \mathbf{X}_1) \\ \cdots &= \cdots \\ \mathbf{X}_4' &= F_4(\mathbf{X}_4; \mathbf{X}_1 - \mathbf{X}_4, \cdots, \mathbf{X}_3 - \mathbf{X}_4), \end{aligned}$$

where $\mathbf{X}_i = (x_1, x_2, x_3, v_1, v_2, v_3)$ and the F_i are real analytic.

- In case $N = 1$, can reduce to a constant – zero dimensional (relative to a rotating coordinate frame).
- Note implications of analyticity: coherence, node inter-dependence, no stops.

Phase oscillator networks

Networks of N weakly coupled nonlinear oscillators can sometimes be modelled by networks of *phase oscillators* (Kuramoto, 1984):

$$\theta_i' = \omega_i + \frac{1}{N}\sum_{j\neq i} g_{ij}(\theta_j - \theta_i),\ i = 1, \cdots, N.$$

Here $\theta_i \in \mathbb{T} = [0,1]/0{=}1$ and the g_{ij} are trigonometric polynomials.
A popular choice is to assume $g_{ij} = G$, all i, j and

$$G(\theta) = \alpha \sin(2\pi\theta) + \beta \sin(4\pi\theta)$$

Also allow $\sin(2\pi\theta + \gamma)$ etc.
(All-to-all coupled, symmetric network.)

But *why* networks?

But *why* networks?

- Understanding network dynamics in terms of network topology?

But *why* networks?

- Understanding network dynamics in terms of network topology?
- Maybe for small networks ... 4 or 5 identical nodes and perhaps in some statistical sense for large networks.

But *why* networks?

- Understanding network dynamics in terms of network topology?
- Maybe for small networks ... 4 or 5 identical nodes and perhaps in some statistical sense for large networks.
- Reductionism. Understand dynamics of *individual* nodes and then infer properties about dynamics of the complete network in terms of the node dynamics.

But *why* networks?

- Understanding network dynamics in terms of network topology?
- Maybe for small networks ... 4 or 5 identical nodes and perhaps in some statistical sense for large networks.
- Reductionism. Understand dynamics of *individual* nodes and then infer properties about dynamics of the complete network in terms of the node dynamics.
- Appropriate (and well-known) for *linear* networks. What about the *nonlinear* case?

Reductionism

N-body problem. Not helpful. Individual nodes have no dynamics!

Reductionism

N-body problem. Not helpful. Individual nodes have no dynamics!

Phase oscillator systems? Obvious problem – also occurring with N-body problem – is that nodes in the network do not *ever* evolve independently of the other nodes (analyticity again). However, there is one case when we can use reductionist logic:

Reductionism

N-body problem. Not helpful. Individual nodes have no dynamics!

Phase oscillator systems? Obvious problem – also occurring with N-body problem – is that nodes in the network do not *ever* evolve independently of the other nodes (analyticity again). However, there is one case when we can use reductionist logic:

Assume nodes synchronized: $\theta_i = \theta_j$, $\omega_i = \omega$, all i, j. We have a solution $\theta_i(t) = \theta_0 + t\omega$. That is, we can replace the network by a single phase oscillator (compare the 1-body problem analysis).

Properties of classical networks

Fixed connection structure – can assume connected graph (else network splits into independent connected components).

$\Longrightarrow$ nodes never evolve independently of one another.

Properties of classical networks

Fixed connection structure – can assume connected graph (else network splits into independent connected components).

$\Longrightarrow$ nodes never evolve independently of one another.

Nodes never stop and then later restart (consequence of analyticity).

Properties of classical networks

Fixed connection structure – can assume connected graph (else network splits into independent connected components).

$\Longrightarrow$ nodes never evolve independently of one another.

Nodes never stop and then later restart (consequence of analyticity).

One set of dynamical equations – no switching between equations.

Properties of classical networks

Fixed connection structure – can assume connected graph (else network splits into independent connected components).

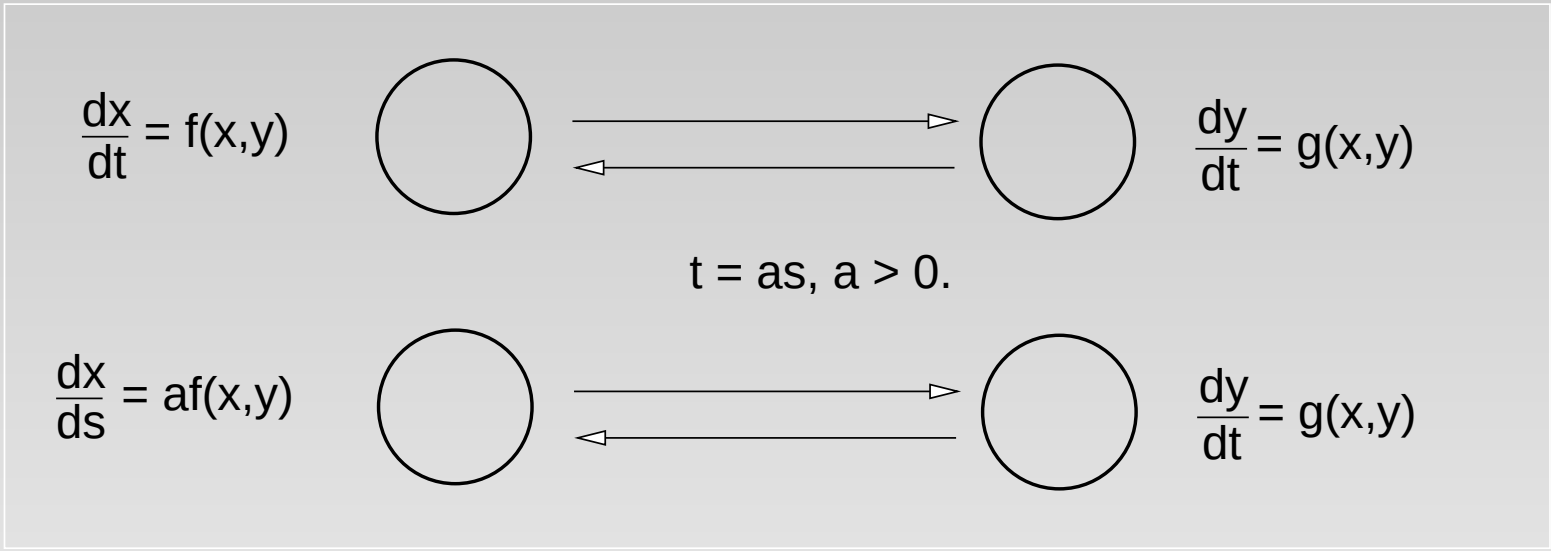
$\Longrightarrow$ nodes never evolve independently of one another.

Nodes never stop and then later restart (consequence of analyticity).

One set of dynamical equations – no switching between equations.

Global clock – all nodes run on same time (simultaneous evolution of nodes).

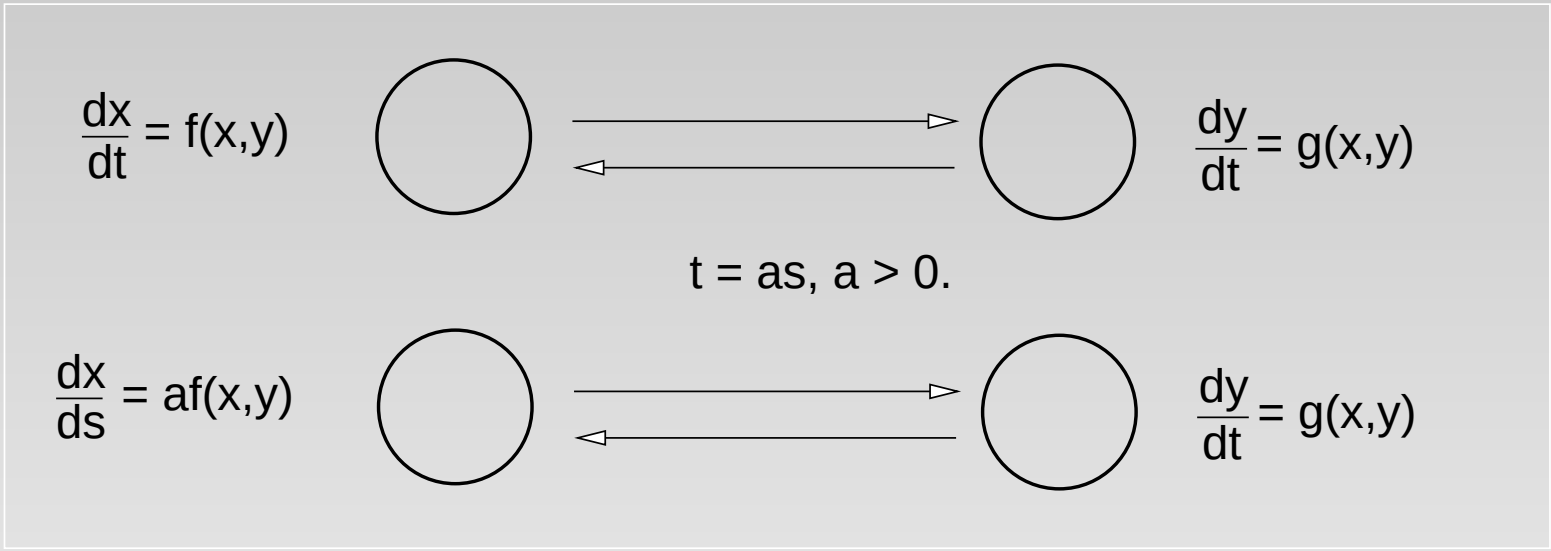
Global & Local time; Synchronous



Changing time on one node

Even for linear systems, changing time on a single node will usually qualitatively change dynamics.

Global & Local time; Synchronous



Changing time on one node

Even for linear systems, changing time on a single node will usually qualitatively change dynamics.

We call networks satisfying the properties listed previously *synchronous networks*. This should not be confused with synchronized dynamics – our terminology comes from computer science.

Asynchronous networks

In an *asynchronous network* we allow

- Variable connectivity – key property: $\Longrightarrow$ dependency relationships between nodes vary.

Asynchronous networks

In an *asynchronous network* we allow

- Variable connectivity – key property: $\Longrightarrow$ dependency relationships between nodes vary.
- Switching between dynamical equations.

Asynchronous networks

In an *asynchronous network* we allow

- Variable connectivity – key property: $\Longrightarrow$ dependency relationships between nodes vary.
- Switching between dynamical equations.
- Local clocks - no natural global clock (dynamics not synchronized to global clock).

Asynchronous networks

In an *asynchronous network* we allow

- Variable connectivity – key property: $\Longrightarrow$ dependency relationships between nodes vary.
- Switching between dynamical equations.
- Local clocks - no natural global clock (dynamics not synchronized to global clock).
- Nodes to stop and later restart.

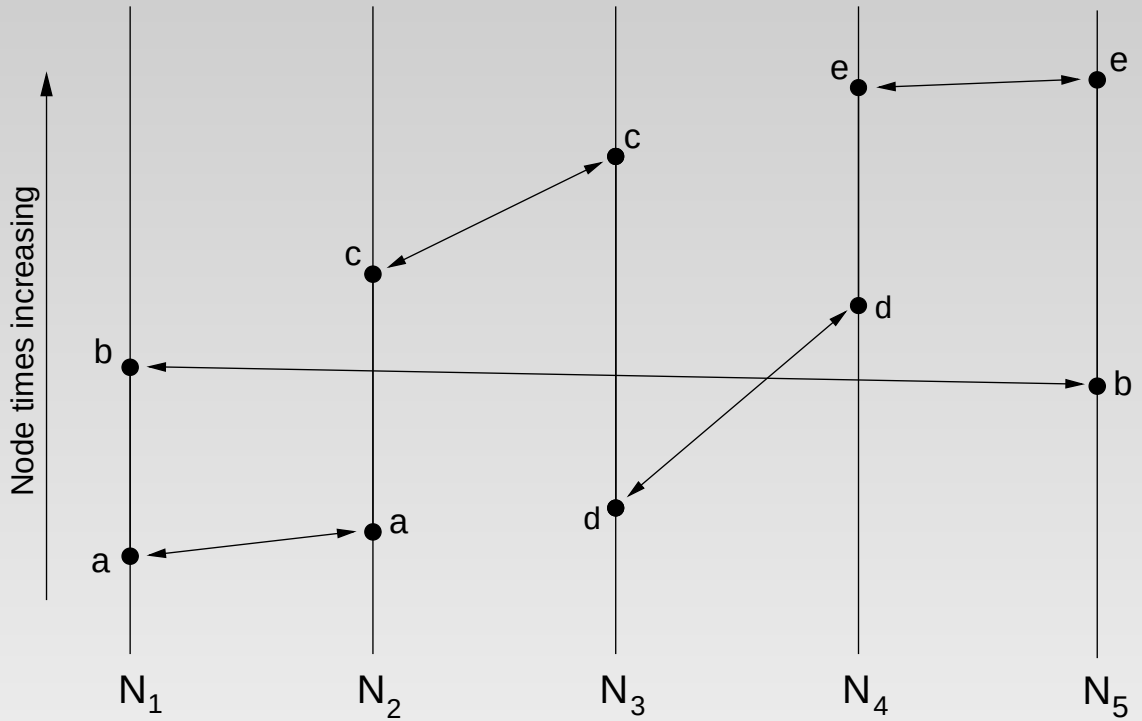
Asynchronous networks

In an *asynchronous network* we allow

- Variable connectivity – key property: $\Longrightarrow$ dependency relationships between nodes vary.
- Switching between dynamical equations.
- Local clocks - no natural global clock (dynamics not synchronized to global clock).
- Nodes to stop and later restart.

All of these characteristics are typical of networks encountered in modern technology (eg distributed networks) and science (especially biology and neuroscience). One might argue that synchronous networks are *atypical* in the 21st century.

Node clocks



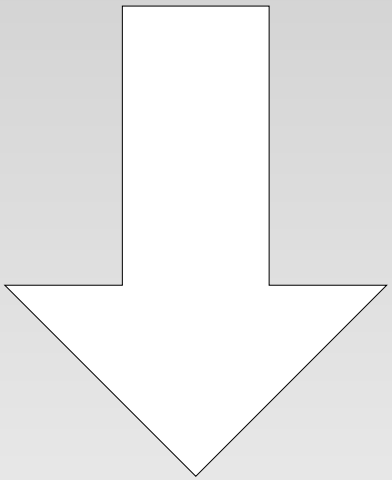
a before b,c,e .. ?d?
b before e .. ?c,d?
c ..?b,e?
d before c,e .. ?a,b?

Partially ordered time structure

Examples: computation

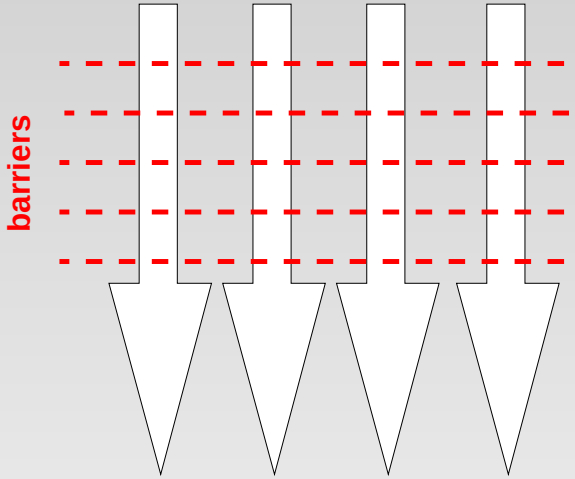
Single processor computation

[Synchronous]



Threaded or parallel computation

[Globally Asynchronous + Locally Synchronous: GALS]

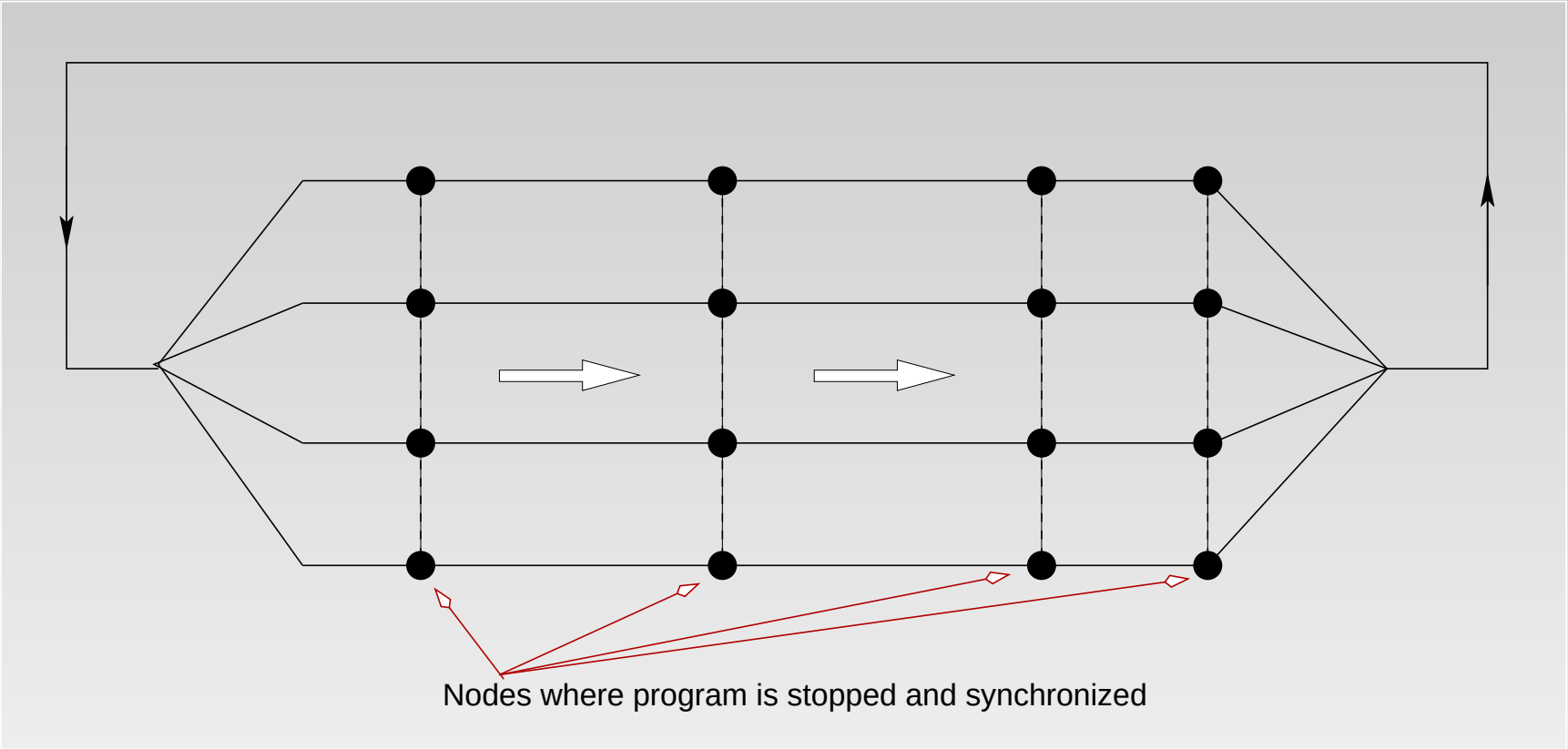


Threads need to be synchronized at each barrier. There may also be locks if, for example, other variables need to be written.

[Non deterministic process]

Threaded & parallel computation

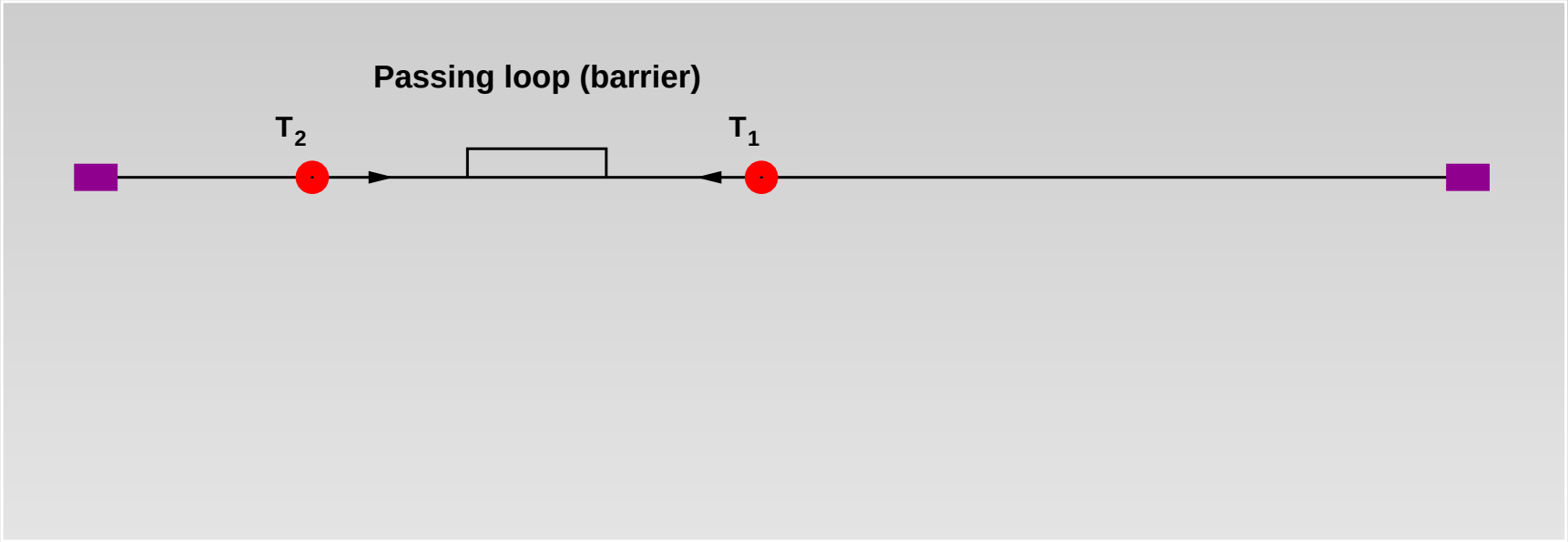
Computation ctd.



Connection structures – 4 threads

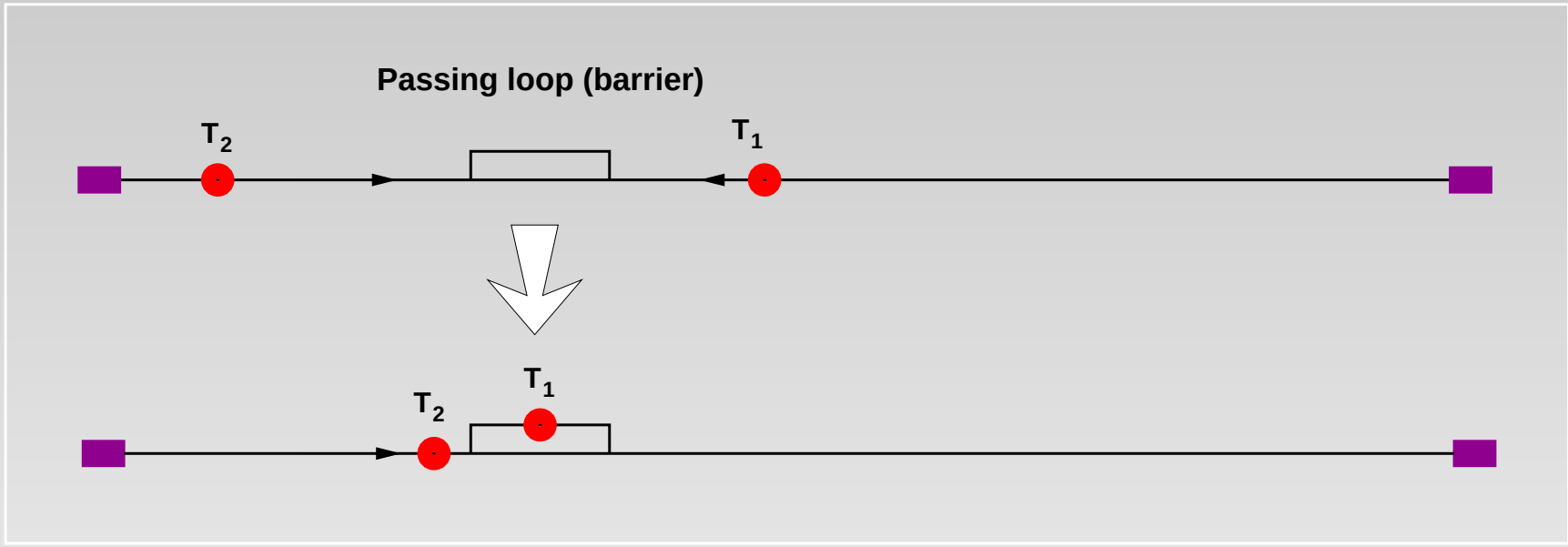
Note: Deadlocks (stop); race conditions (errors)

Constrained transport: passing loop



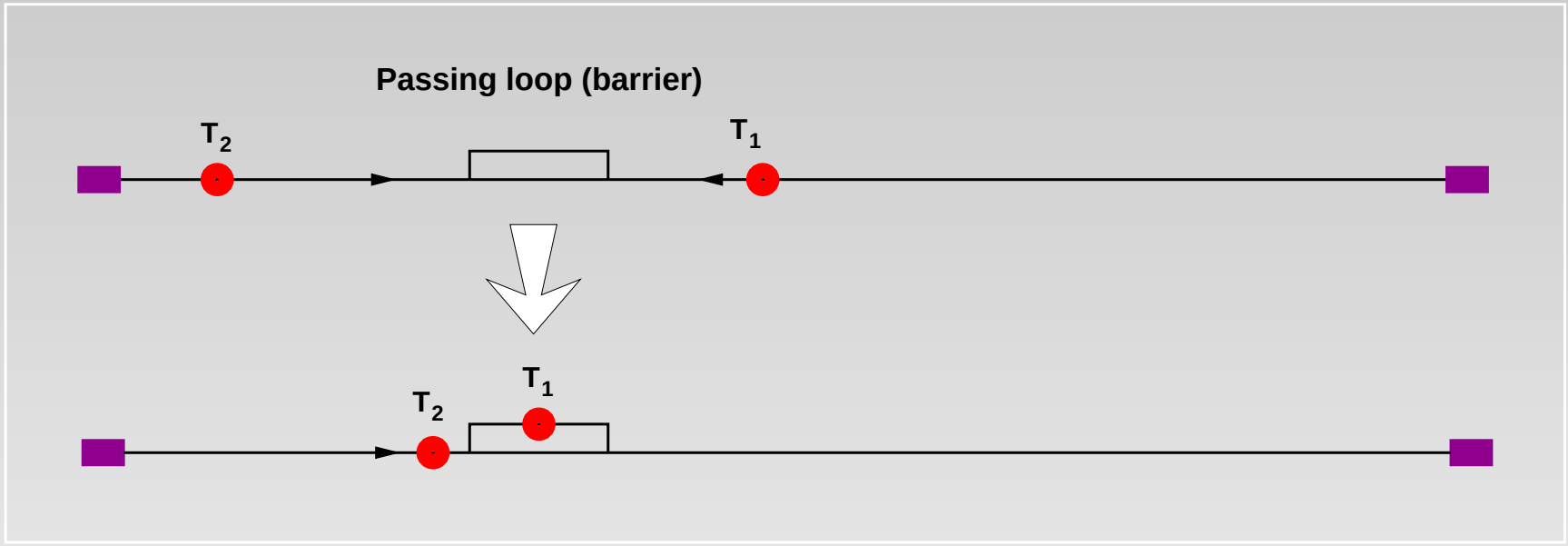
Single track line with a passing loop; two trains

Constrained transport: passing loop



Single track line with a passing loop; two trains

Constrained transport: passing loop



Single track line with a passing loop; two trains

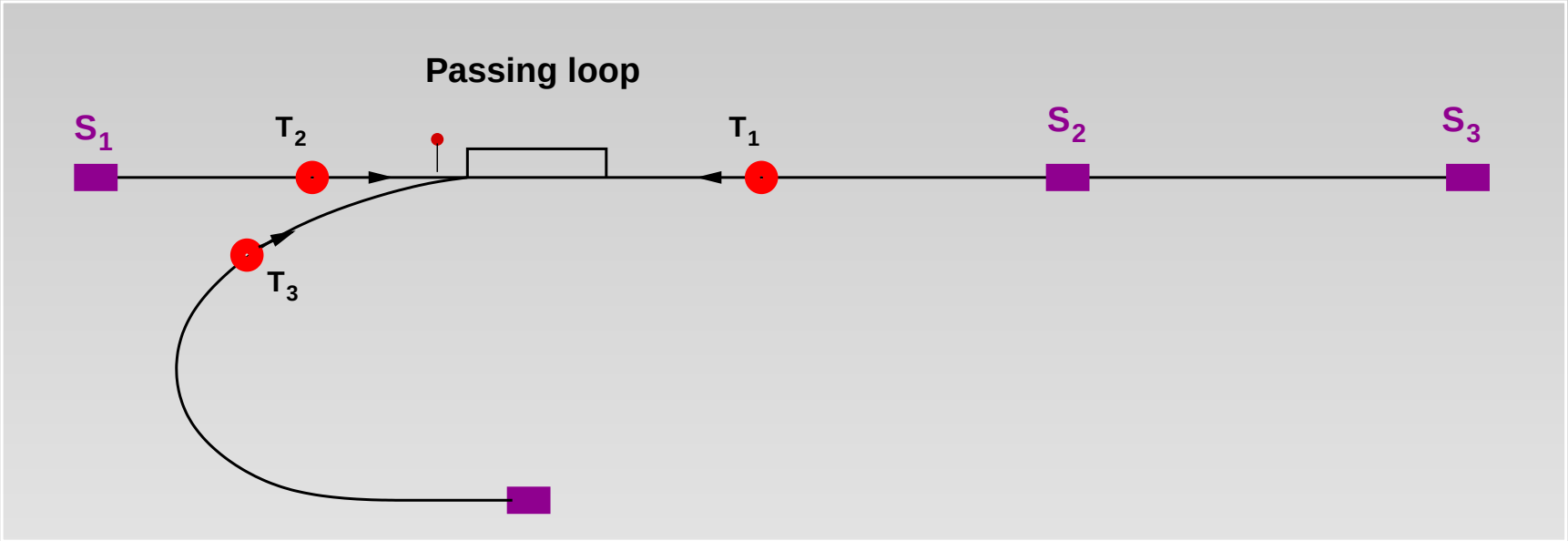
Issues:

Deadlocks (or livelocks: convergence to blocking attractor).

Logic

Note: *Order* of entry into passing loop irrelevant.

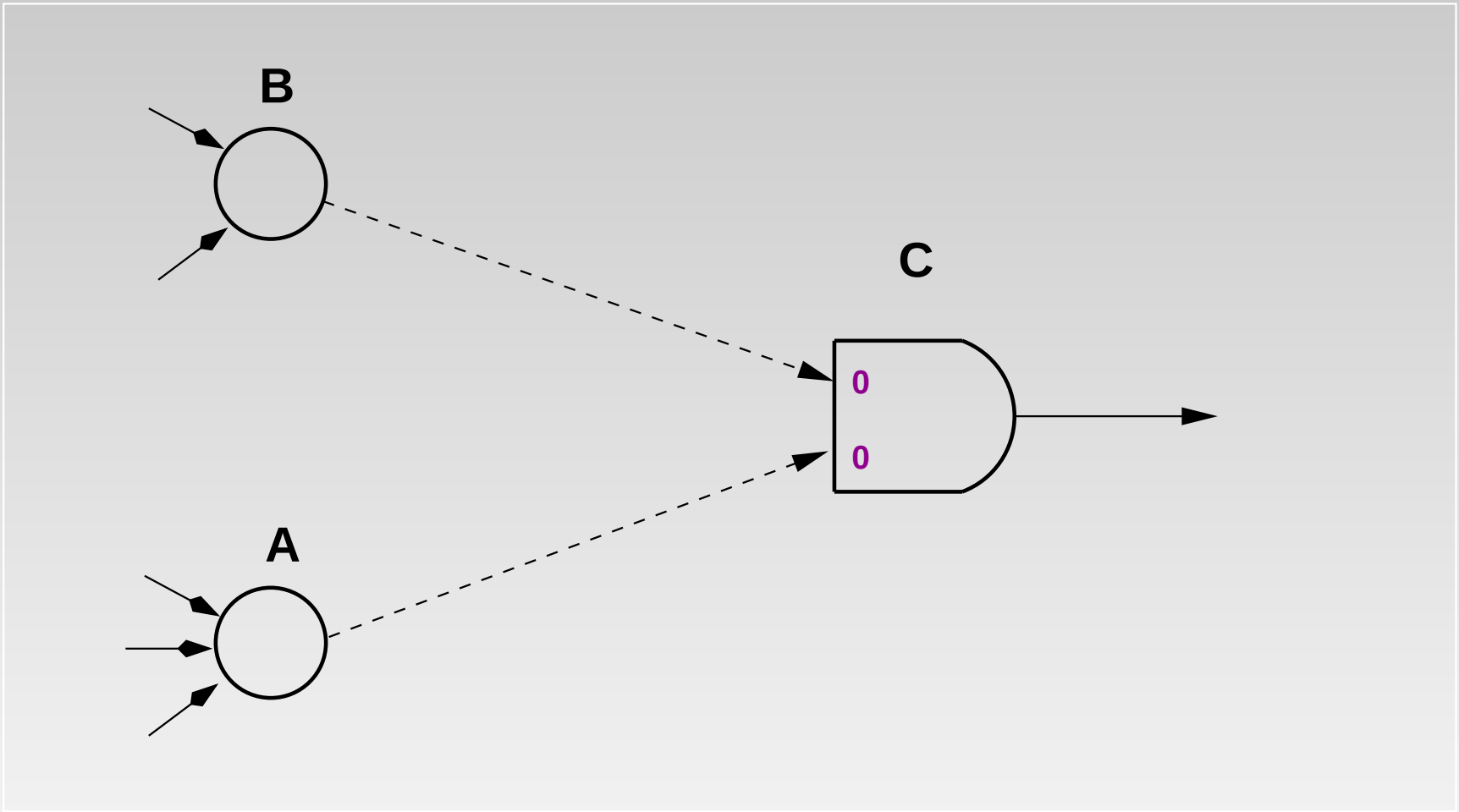
Passing loop, variation



Single track line with a passing loop and branch; three trains

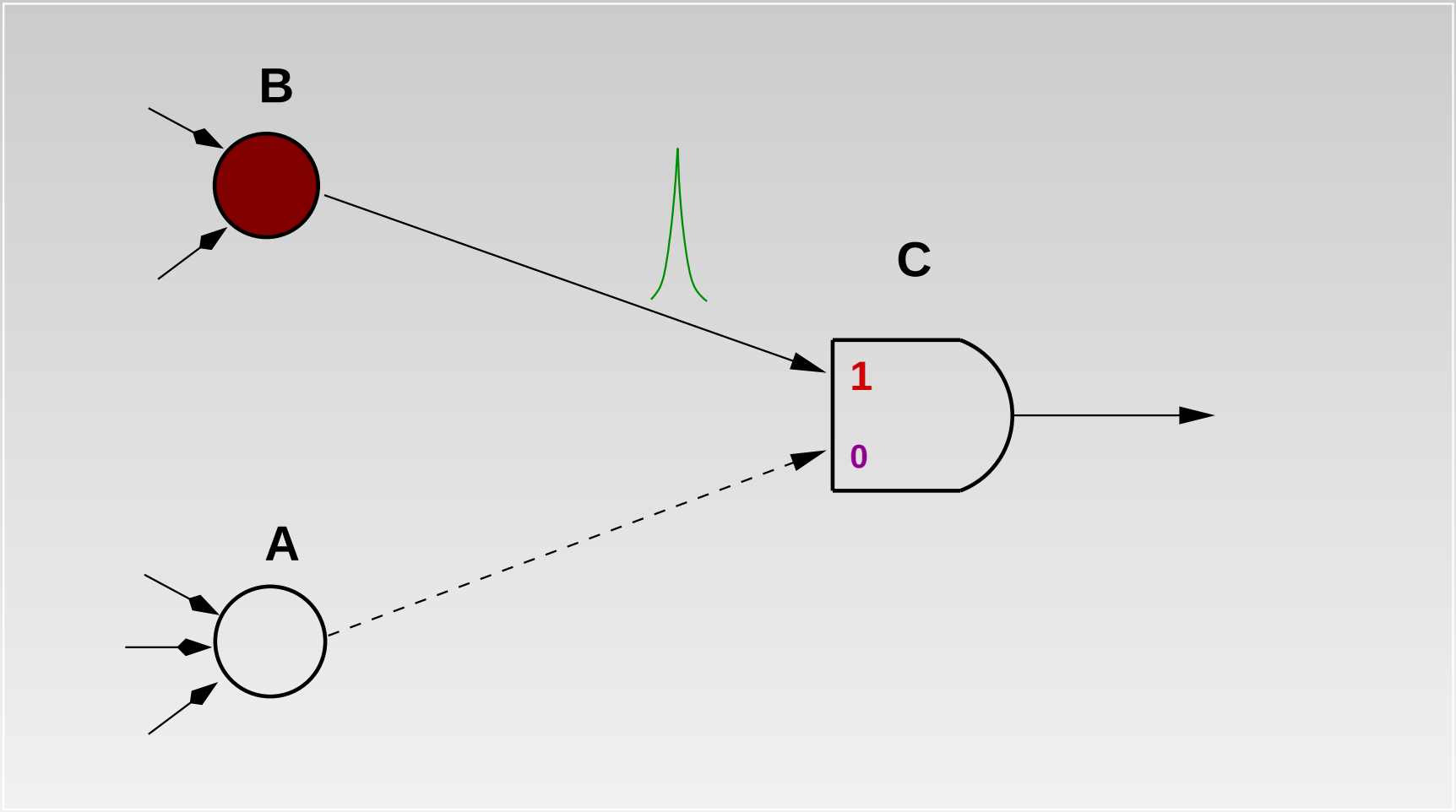
T_1 terminates at S_1; T_2 at S_2; T_3 at S_3.
Unlike in the previous case, *order of entry of trains into loop is critical* – asynchronous logic is now fragile: an error results in a deadlock or race (if T_2, T_3 attempt to enter loop at same time). Simple model – but very widely applicable.

Spiking neuron models; switching



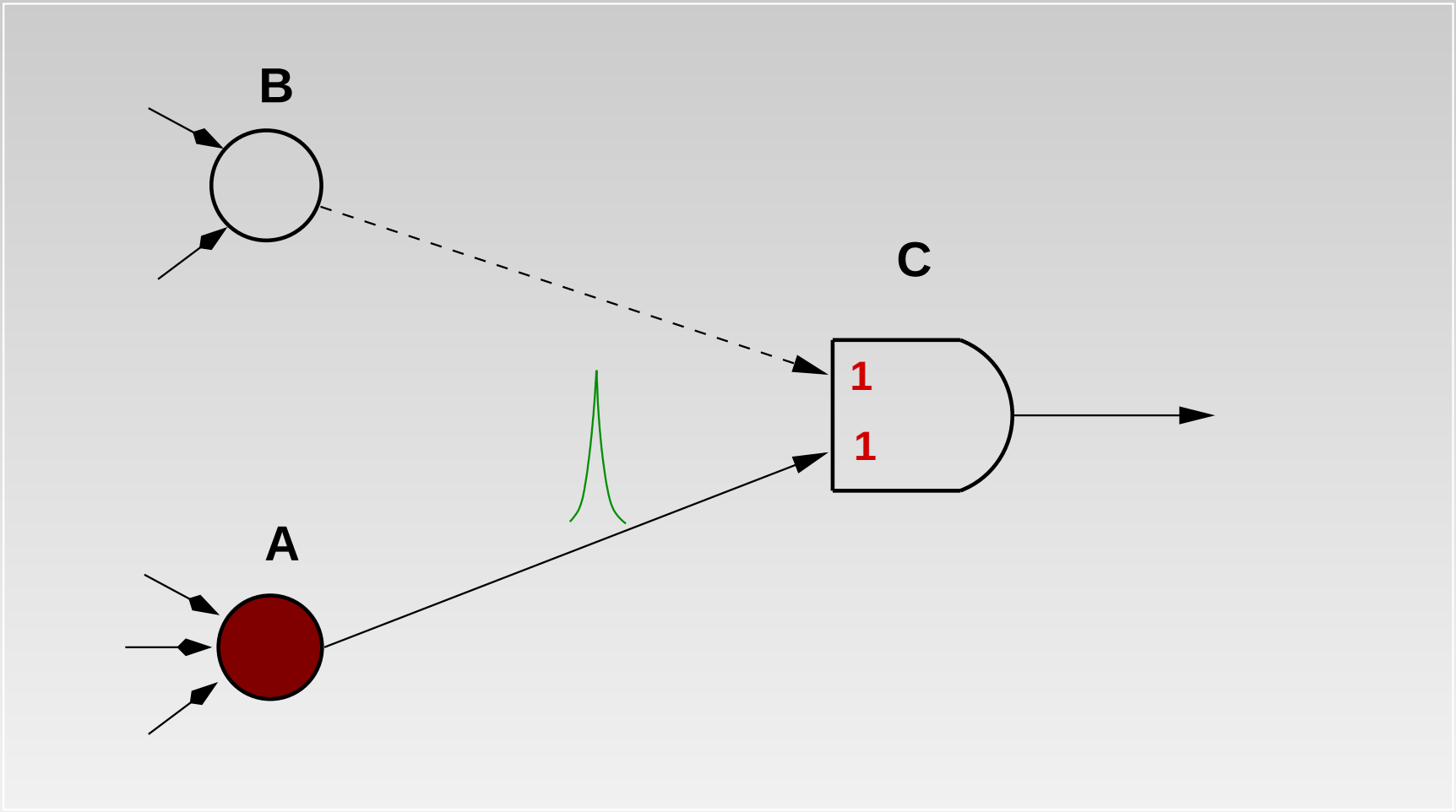
Nodes A, B evolve (continuous dynamics). If state of either A or B reaches a threshold, then node fires – a spike or pulse – towards target node C (stopped).

Spiking neuron models; switching, ctd



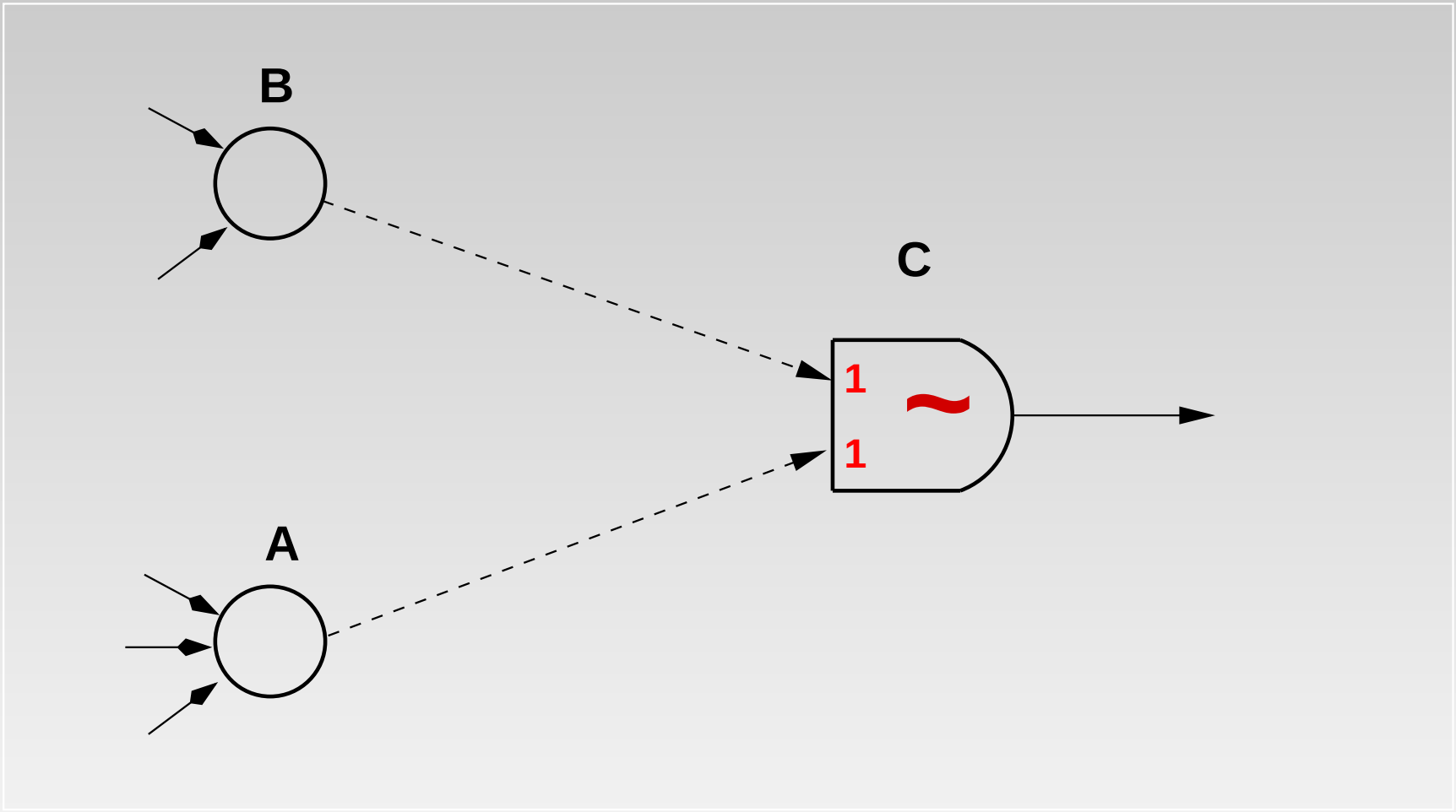
Node B fires a spike towards C – receipt registered by changing input state to 1. Nodes A and B continue to evolve, Node C stopped.

Spiking neuron models; switching, ctd



Node A fires a spike towards C – Both inputs of C are now activated.

Spiking neuron models; switching, ctd



Various possibilities: (A) (Shown) With both inputs filled, C starts and further inputs blocked (inputs set to zero after fixed time, or decay to zero LIF).

Spiking neuron models; switching, ctd

(B) Order of filling inputs may matter: C only starts if B fires *before* A. Similar to passing loop with branch example or in distributed production systems. Order that parts/chemicals/signals are received may be critical for functionality.

Spiking neuron models; switching, ctd

(B) Order of filling inputs may matter: C only starts if B fires *before* A. Similar to passing loop with branch example or in distributed production systems. Order that parts/chemicals/signals are received may be critical for functionality.

In large complex systems, the asynchronous logic (handshaking protocols) involved in running a system where order of inputs matters is likely to make the system very fragile and susceptible to deadlocks (eg timetable disruption).

Spiking neuron models; switching, ctd

(B) Order of filling inputs may matter: C only starts if B fires *before* A. Similar to passing loop with branch example or in distributed production systems. Order that parts/chemicals/signals are received may be critical for functionality.

In large complex systems, the asynchronous logic (handshaking protocols) involved in running a system where order of inputs matters is likely to make the system very fragile and susceptible to deadlocks (eg timetable disruption).

Adaptation and randomness are likely to play a major role in any complex asynchronous network; in particular, to avoid deadlocks. Note that spikes avoid race conditions “a.s.”.

Asynchronous Networks

A general definition is given in terms of events – which may be deterministic or stochastic – and local times (in the non-autonomous case) and continuous or discrete dynamics. We give a formal definition in the simplest case: discrete, deterministic and autonomous.

Asynchronous Networks

A general definition is given in terms of events – which may be deterministic or stochastic – and local times (in the non-autonomous case) and continuous or discrete dynamics. We give a formal definition in the simplest case: discrete, deterministic and autonomous.

Assume a fixed $\mathcal{N}$ set of nodes: $N_0, \cdots, N_n$ (N_0 denotes *null node*).

Asynchronous Networks

A general definition is given in terms of events – which may be deterministic or stochastic – and local times (in the non-autonomous case) and continuous or discrete dynamics. We give a formal definition in the simplest case: discrete, deterministic and autonomous.

Assume a fixed $\mathcal{N}$ set of nodes: $N_0, \cdots, N_n$ (N_0 denotes *null node*).

Let $\mathcal{C}$ denote the set of all directed connection structures on $\mathcal{N}$ (no self- or multiple connections). Note that $\emptyset \in \mathcal{C}$.

Asynchronous Networks

A general definition is given in terms of events – which may be deterministic or stochastic – and local times (in the non-autonomous case) and continuous or discrete dynamics. We give a formal definition in the simplest case: discrete, deterministic and autonomous.

Assume a fixed $\mathcal{N}$ set of nodes: $N_0, \cdots, N_n$ (N_0 denotes *null node*).

Let $\mathcal{C}$ denote the set of all directed connection structures on $\mathcal{N}$ (no self- or multiple connections). Note that $\emptyset \in \mathcal{C}$.

Fix a non-empty subset $\mathcal{A}$ of $\mathcal{C}$. Every $\mathbf{C} \in \mathcal{A}$ gives $\mathcal{N}$ the structure of a directed graph (the connection matrix is a 01 matrix with diagonal elements zero).

Asynchronous Networks ctd

Assume each node N_i, $i \neq 0$, has associated phase space M_i. Set $M = \prod_{i=1}^{n} M_i$

Asynchronous Networks ctd

Assume each node N_i, $i \neq 0$, has associated phase space M_i. Set $M = \prod_{i=1}^{n} M_i$

Assume that each $\mathbf{C} \in \mathcal{A}$ determines a smooth (enough) map $f_{\mathbf{C}} : M \to M$ satisfying

- For $i \in \{1, \cdots, N\}$, $j \neq i$, $f^i_{\mathbf{C}}$ depends on $x_j \in N_j$ only if there is an edge $N_j \to N_i$ in $\mathbf{C}$.
- If there is an edge $N_0 \to N_i$, then $f^i_{\mathbf{C}}$ is constant (stopped node).

Asynchronous Networks ctd

Assume each node N_i, $i \neq 0$, has associated phase space M_i. Set $M = \prod_{i=1}^{n} M_i$

Assume that each $\mathbf{C} \in \mathcal{A}$ determines a smooth (enough) map $f_{\mathbf{C}} : M \to M$ satisfying

- For $i \in \{1, \cdots, N\}$, $j \neq i$, $f^i_{\mathbf{C}}$ depends on $x_j \in N_j$ only if there is an edge $N_j \to N_i$ in $\mathbf{C}$.
- If there is an edge $N_0 \to N_i$, then $f^i_{\mathbf{C}}$ is constant (stopped node).

Assume given an *event map* $\mathcal{E} : M \to \mathcal{A}$.

Asynchronous Networks ctd

Assume each node N_i, $i \neq 0$, has associated phase space M_i. Set $M = \prod_{i=1}^{n} M_i$

Assume that each $\mathbf{C} \in \mathcal{A}$ determines a smooth (enough) map $f_{\mathbf{C}} : M \to M$ satisfying

- For $i \in \{1, \cdots, N\}$, $j \neq i$, $f^i_{\mathbf{C}}$ depends on $x_j \in N_j$ only if there is an edge $N_j \to N_i$ in $\mathbf{C}$.
- If there is an edge $N_0 \to N_i$, then $f^i_{\mathbf{C}}$ is constant (stopped node).

Assume given an *event map* $\mathcal{E} : M \to \mathcal{A}$.

This data defines the structure of a discrete asynchronous network – synchronous if $|\mathcal{A}| = 1$.

Dynamics

Given data for a discrete asynchronous network as above, we define $F : M \to M$ by

$$F(\mathbf{X}) = (f^1_{\mathcal{E}(\mathbf{X})}(\mathbf{X}), \cdots, f^n_{\mathcal{E}(\mathbf{X})}(\mathbf{X})),\ \mathbf{X} \in M.$$

Provided the event map is not constant (synchronous case) and we avoid trivial cases (eg the maps f_{C} are identical), the operator F will not be analytic (switching is *forced* in asynchronous networks).

In practice, we add conditions to avoid degeneracies. In many situations (eg passing loop), the event map will be constant on an open dense set.

An example of a *state dependent* dynamical system (engineers terminology).

Examples

- Random connection structure (RDD network). ▶
- Adaptive network and sloppy asynchronous logic. ▶
- STDP in a spiking neural network. ▶

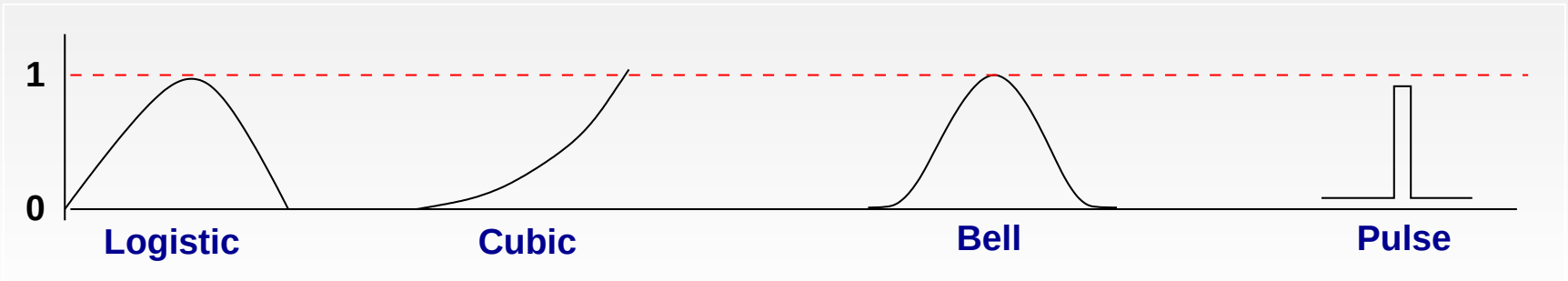
Dynamics, Example I

1. Random (in time) connection structure.
2. Discrete phase oscillator dynamics.

Inhomogeneous ‘Poisson neuron’ firing model: probability of a node firing is state dependent.

$$p(\theta) = 16\theta^2(1-\theta)^2, \quad \textbf{Bell}.$$

$$p(\theta) = \begin{cases} 0.05, & \theta \leq 0.5 - d, \\ 0.05, & \theta \geq 0.5 + d, \\ 0.95, & \theta \in (0.5 - d, 0.5 + d) \end{cases} \qquad \textbf{Pulse}$$



Maps

Set $\mathbf{N} = \{1, \cdots N\}$. Fix $\omega_i > 0$, $i \in \mathbf{N}$ and constants $a, b, c \in \mathbb{R}$. For $i \neq j \in \mathbf{k}$, $\boldsymbol{\theta} \in \mathbb{T}^N$, define

$$F_{ij}(\boldsymbol{\theta}) = a\mathbf{Sin}(\theta_i - \theta_j) + b\mathbf{Sin}(2(\theta_i - \theta_j + c))$$

As iterative scheme, take

$$\theta_i^{n+1} = \theta_i^n + \omega_i + \frac{1}{k}\sum\nolimits_j^\star F_{ji}(\boldsymbol{\theta}^n)$$

where the sum is over all j such that cell j fired and there is a connection $j \to i$.

This system can be modelled as a place dependent RDS (the number of symbols grows super-exponentially fast in N: $\sim 2^{N^2}$).

Visualization of dynamics

Use a system of contractive cocycles forced by (firing) dynamics.

If the system has N nodes, regard the nodes as vertices of a regular polygon, centered at the origin of $\mathbb{R}^2 \approx \mathbb{C}$. Denote the coordinates of C_j by Z_j.

Associated to the node C_j we define a contraction mapping f_j with fixed point Z_j by

$$f_j(z) = \frac{1}{2}(z + Z_j).$$

Take the initial point $z_0 = 0 \in \mathbb{C}$.

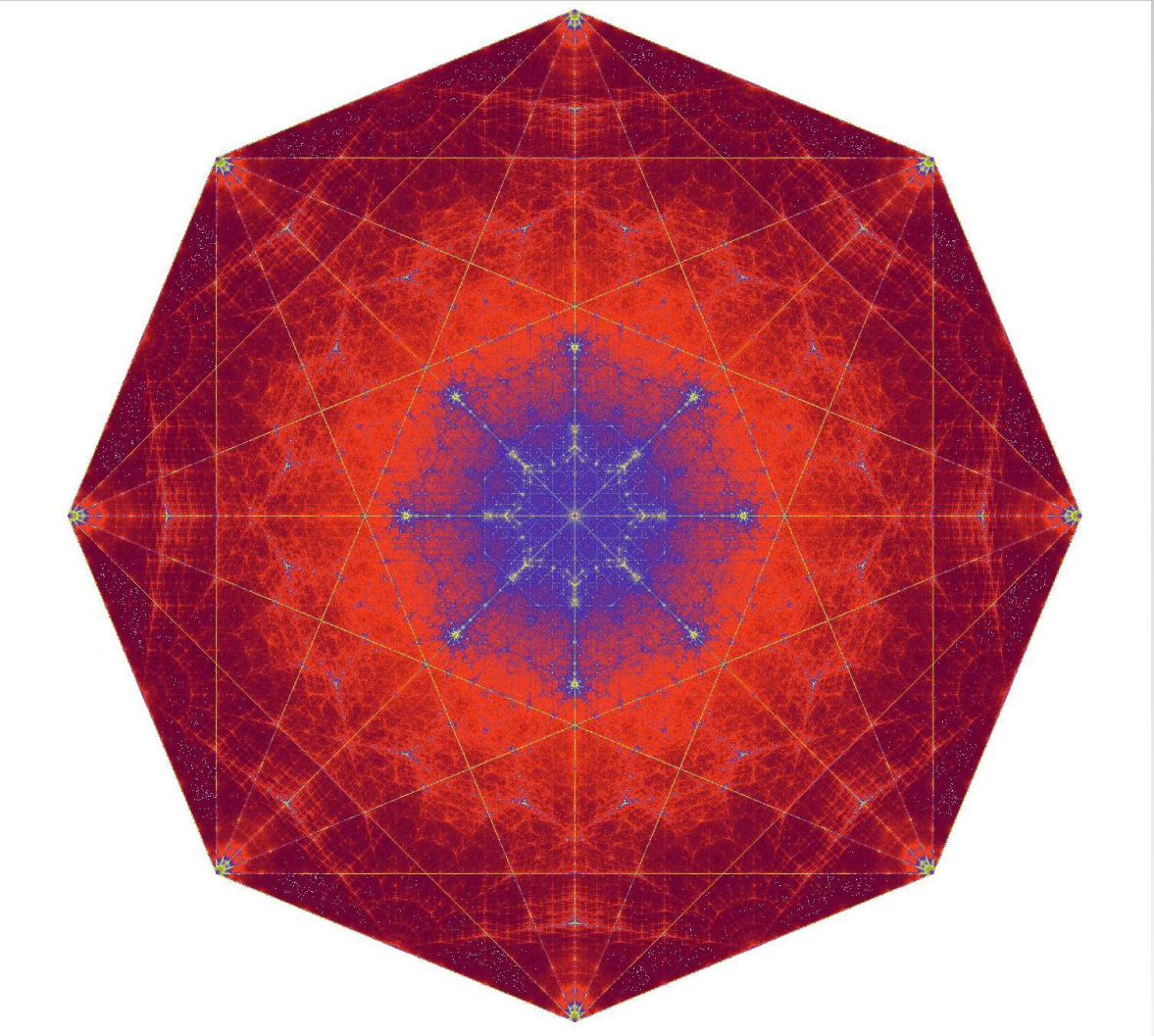
Measurement

Suppose constructed the sequence $z_0, z_1, \ldots, z_n$ after $m \geq n$ time steps. At the $(m+1)$th step of the iteration, suppose that the nodes $C_{j_1}, \ldots, C_{j_k}$ fire (if no nodes fire, do nothing, go to the next iteration). Define

$$z_{n+1} = \frac{1}{k} \sum_{i=1}^{k} f_{j_i}(z_n).$$

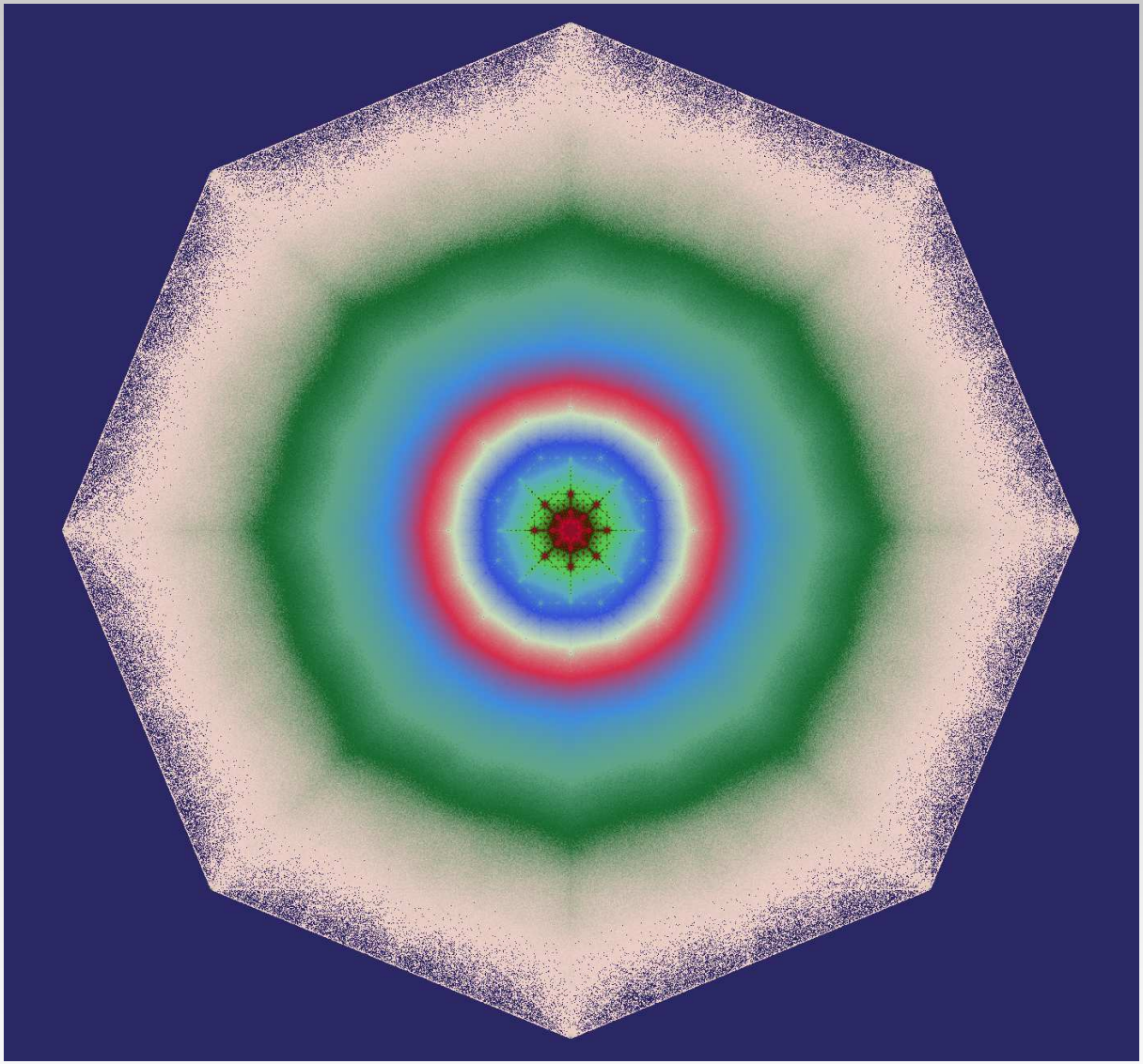
At least numerically, (z_n) converges (often slowly) to an attractor with associated invariant measure (and usually $\mathbf{D}_N$ symmetry!). The attractor and measure reflect statistical properties of the node dynamics (eg statistics of synchrony patterns).

8-node example: Pulse probability



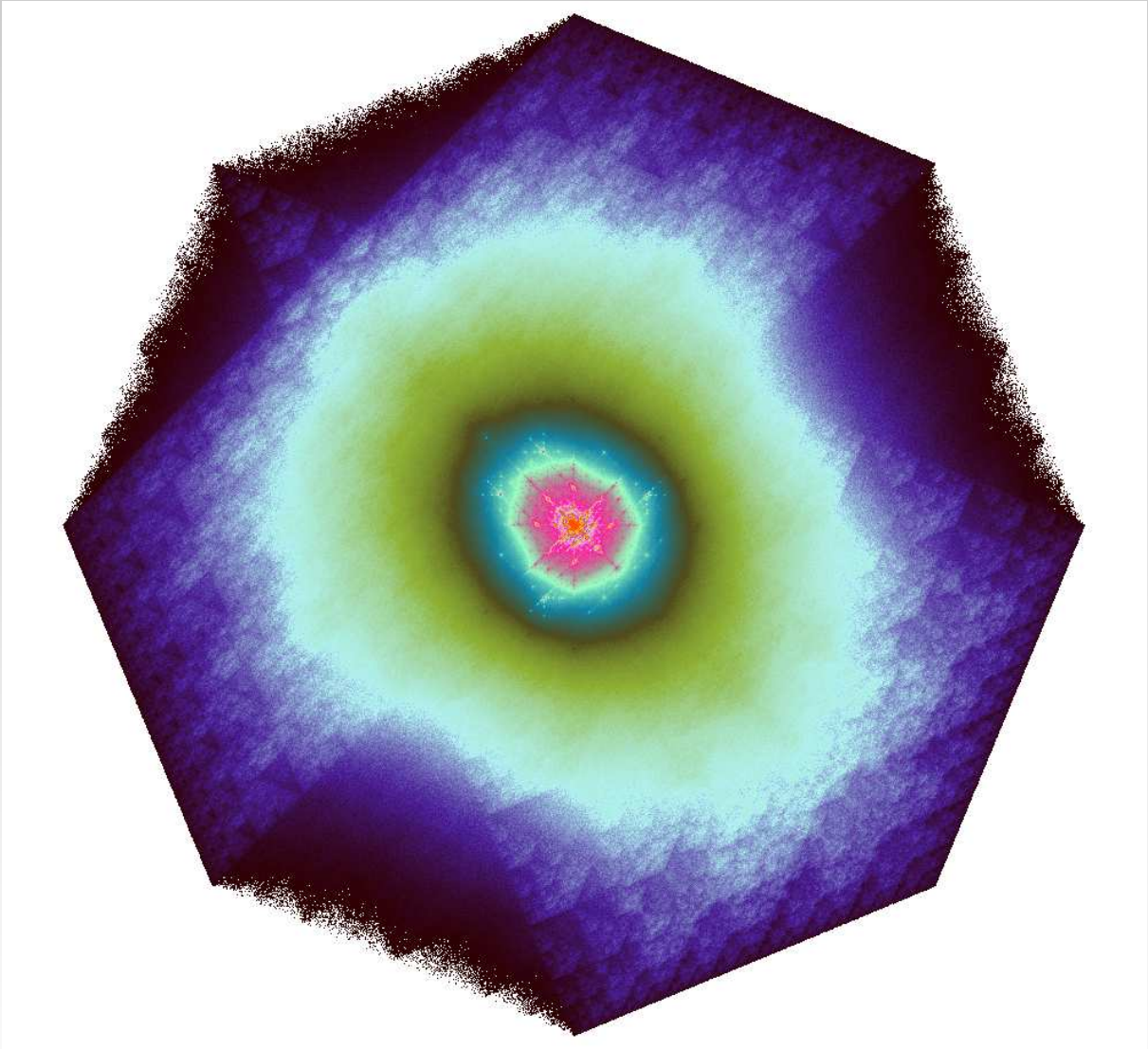
Visualization of clustering and synchronization: random connection structure, $\omega = 0.0001$, $a = 0.1301$, $b = -0.15$, $c = 0$.

8-node example: Bell probability



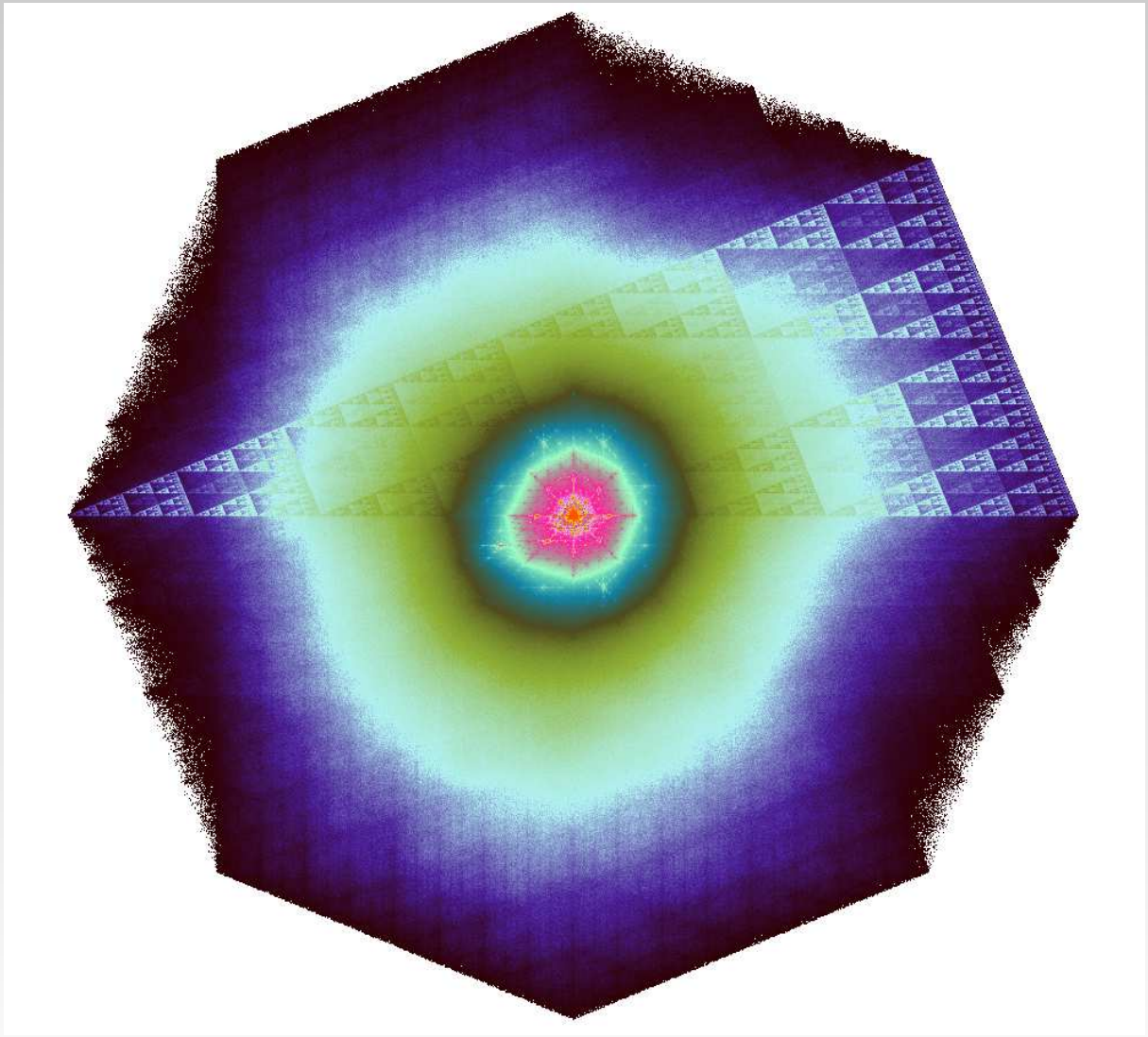
$\omega = 0.0002$, $a = 0.16$, $b = -0.0336$, $c = 0$.

Example A



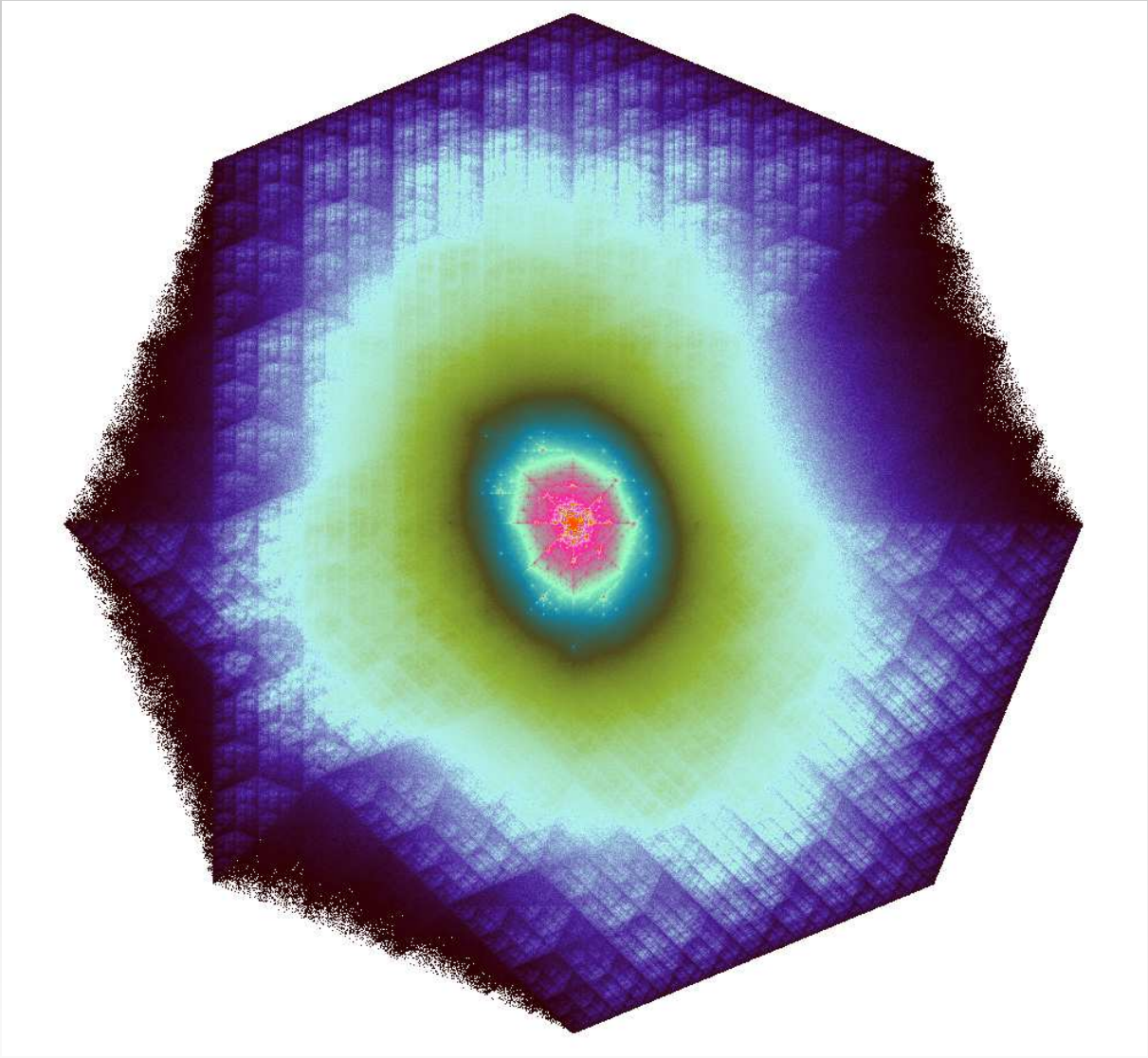
$\omega = 0.0002$, $a = 0.06$, $b = -0.0336$, $c = 0$.

Example B



$\omega = 0.0002$, $a = 0.06$, $b = -0.0336$, $c = 0$.

Example C



$\omega = 0.0002$, $a = 0.06$, $b = -0.0336$, $c = 0$.

Invariant subspaces

We recall that the map used for these examples is

$$\theta_i \mapsto \omega + \theta_i + \frac{1}{k}\sum{}^{\star} 0.06 \sin 2\pi(\theta_j - \theta_i) - 0.0336 \sin 4\pi(\theta_j - \theta_i),$$

where the sum is over the k 'fired' cells connected to the θ_i-cell and $\omega = 0.00002$.

In this case the cell states synchronize into *either* two clusters of 4 cells, *or* one cluster of 5 and one of 3 cells. So either $\theta_j = \theta_i$ or $\theta_j - \theta_i = \alpha$, where

$$0.06 \sin 2\pi\alpha - 0.0336 \sin 4\pi\alpha = 0.$$

(Hence $\alpha = \frac{1}{2\pi}\cos^{-1}(0.89285) = 0.074$.)

Intermingled basins of attraction

The invariant $4:4, 3:5$ subspaces defined by $\theta_j - \theta_i = 0, \alpha$ can be shown to be normally hyperbolic attracting (neutral stabilities in the subspace, phase shift directions). Given any initial point, there is almost sure convergence to one of the 252 different attractors corresponding to $4:4$ or $5:3$ clustering.

More precisely, let $\mathcal{E}$ denote the set of $4:4, 3:5$ subspaces. For $x_0 \in \mathbb{T}^8$, $\omega(x_0)$ exists a.s. Define

$$\begin{aligned} B_0 &= \{x_0 \in \mathbb{T}^8 \mid \omega(x_0) \subset \cup_{E\in\mathcal{E}} E\}, \\ B_1 &= \{x_0 \in \mathbb{T}^8 \mid \exists! E \in \mathcal{E}, \omega(x_0) \subset E\}. \end{aligned}$$

Intermingled basins of attraction ctd.

That is, if $x_0 \in B_1$, $\omega(x_0)$ is always subset of *same* E. For $x_0 \in B_0$, we may get different E each time iteration is run (case of intermingled basins of attraction).

$\mu(B_0) = 1$, $B_0 \neq \mathbb{T}^8$ and $0 < \mu(B_1) < 1$.

Intermingled basins of attraction ctd.

That is, if $x_0 \in B_1$, $\omega(x_0)$ is always subset of *same* E. For $x_0 \in B_0$, we may get different E each time iteration is run (case of intermingled basins of attraction).

$\mu(B_0) = 1$, $B_0 \neq \mathbb{T}^8$ and $0 < \mu(B_1) < 1$.

One way of breaking the invariant subspace structure is by using the term $\sin(4\pi(\theta_j - \theta_i - c))$, $c \neq 0$, rather than $\sin(4\pi(\theta_j - \theta_i))$. Alternatively, we may assume that $\omega = \omega_i$ (say with uniform distribution in $[\omega - \delta, \omega + \delta]$, $0 < \delta/\omega \ll 1$). We show a movie of the result (either case). ◀ ▶

Dynamics, Example 2

We want to address the problem of asynchronous logic in large asynchronous networks. We present an example of a synchronous adaptive network as an illustration of one way to overcome the problem of the fragility of and complexity of asynchronous logic. Two of the illustrations we present are really asynchronous.

1. All-to-all connection structure.
2. Node dynamics given by odd logistic maps.

$$f_\lambda(x) = \lambda x(1 - x^2).$$

3. Adaptive network – spatial verion of Spike-Timing Dependent Plasticity (STDP).

Adaptive network of odd logistic maps

We assume N nodes where $N \in [2, 10^4]$ and node dynamics given odd-logistic maps. We rescale to $[0, 1]$ and take

$$F_\lambda(x) = \frac{\lambda}{2}(1 - 18x + 48x^2 - 32x^3) + \frac{1}{2},\ \lambda \in [-1, 1]$$

Denote weight of connection from node j to node i by w_{ij} and assume $w_{ij} \in [0, 2]$. State update rule given by

$$x_i^{n+1} = F_{\lambda^n}(x_i^n) + \frac{\alpha}{N} W_i(\mathbf{x}^n),$$

where $W_i(\mathbf{x}^n) = \sum_{j \neq i} w_{ij}^n x_j^n$. In our example, we take $\alpha = 0.45$

Weight update rule

If at time n states and weights are given by x_i^n, w_{ij}^n, then

$$w_{ij}^{n+1} = \max\{0, \min\{2, w_{ij}^n + \Delta(w_{ij}^n)\}\},$$

where

$$\Delta(w_{ij}^n) = F(w_{ij}^n, x_i^n, x_j^n),$$

and $F(w, x, y) = G(w)H(x, y)$. For our example, we take

$$\begin{aligned} G(w) &= w, \text{ (Multiplicative)} \\ H(x, y) &= 0.2(1 - 4.5 \min\{|x - y|, 1 - |x - y|\}), \\ &\quad \text{(distance on } \mathbb{T}\text{)}. \end{aligned}$$

Notes on adaptation

Observe the adaptation strengthens w_{ij} if $|x_i - x_j|$ is small. For example if $x_i^n = x_j^n$, then

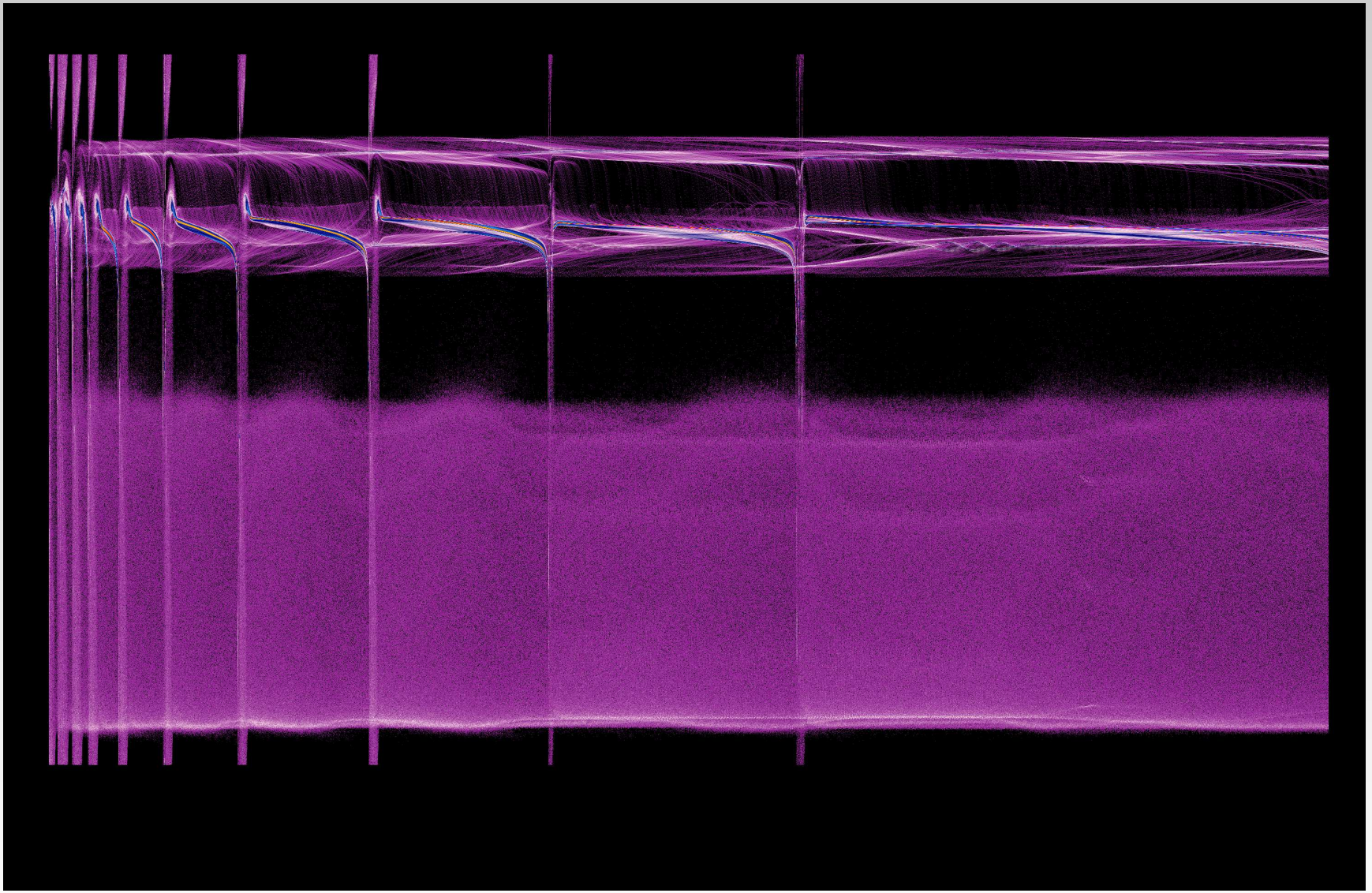
$$w_{ij}^{n+1} = \min\{2, 1.2w_{ij}^n\}.$$

Conversely if $|x_i - x_j|$ is large (close to 0.5), weights are weakened. For example, if $|x_i - x_j| = 0.5$, then

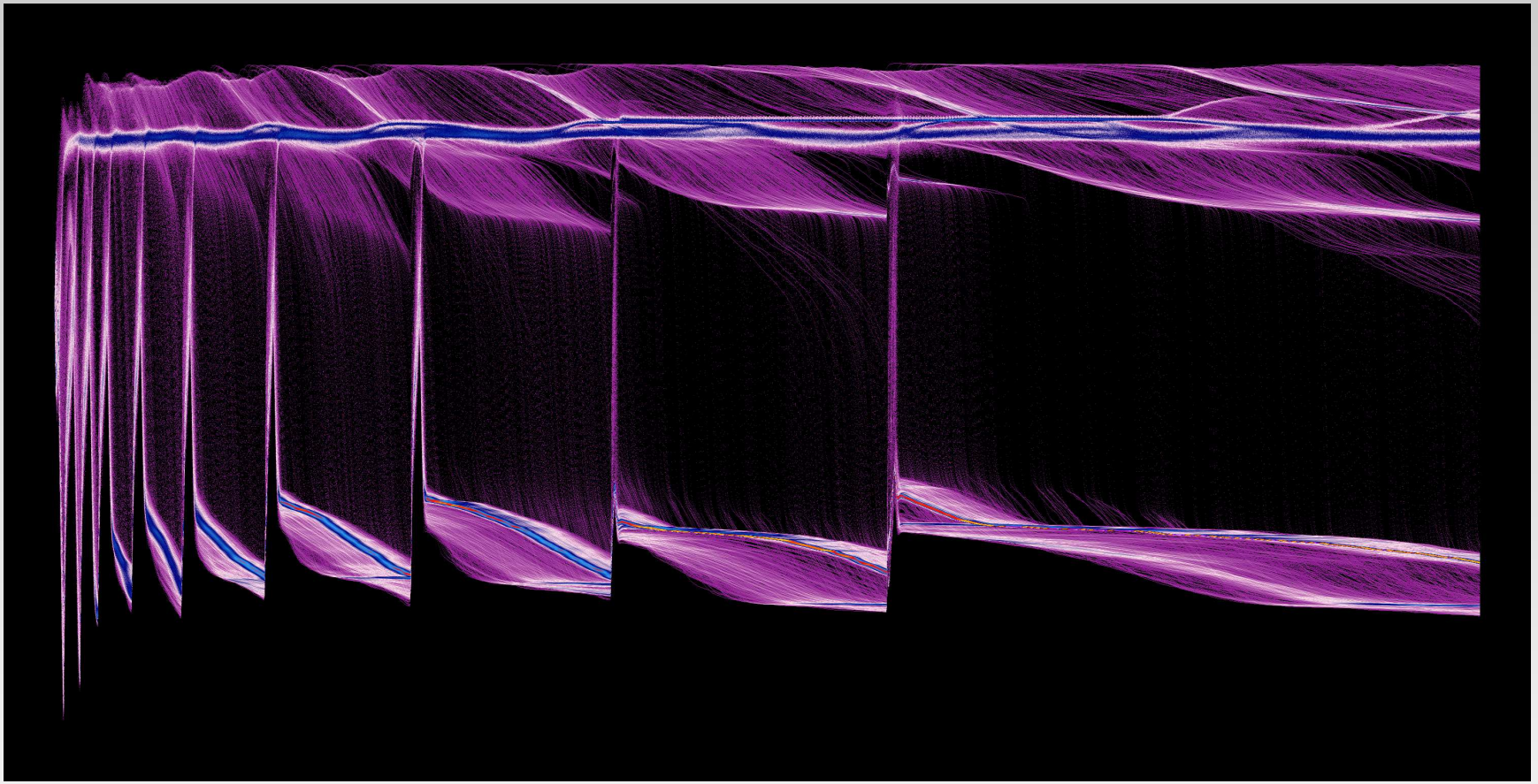
$$w_{ij}^{n+1} = 0.75w_{ij}^n.$$

In the next two slides we show dynamics and weight dynamics over about 5600 iterations for a 6000 node network.

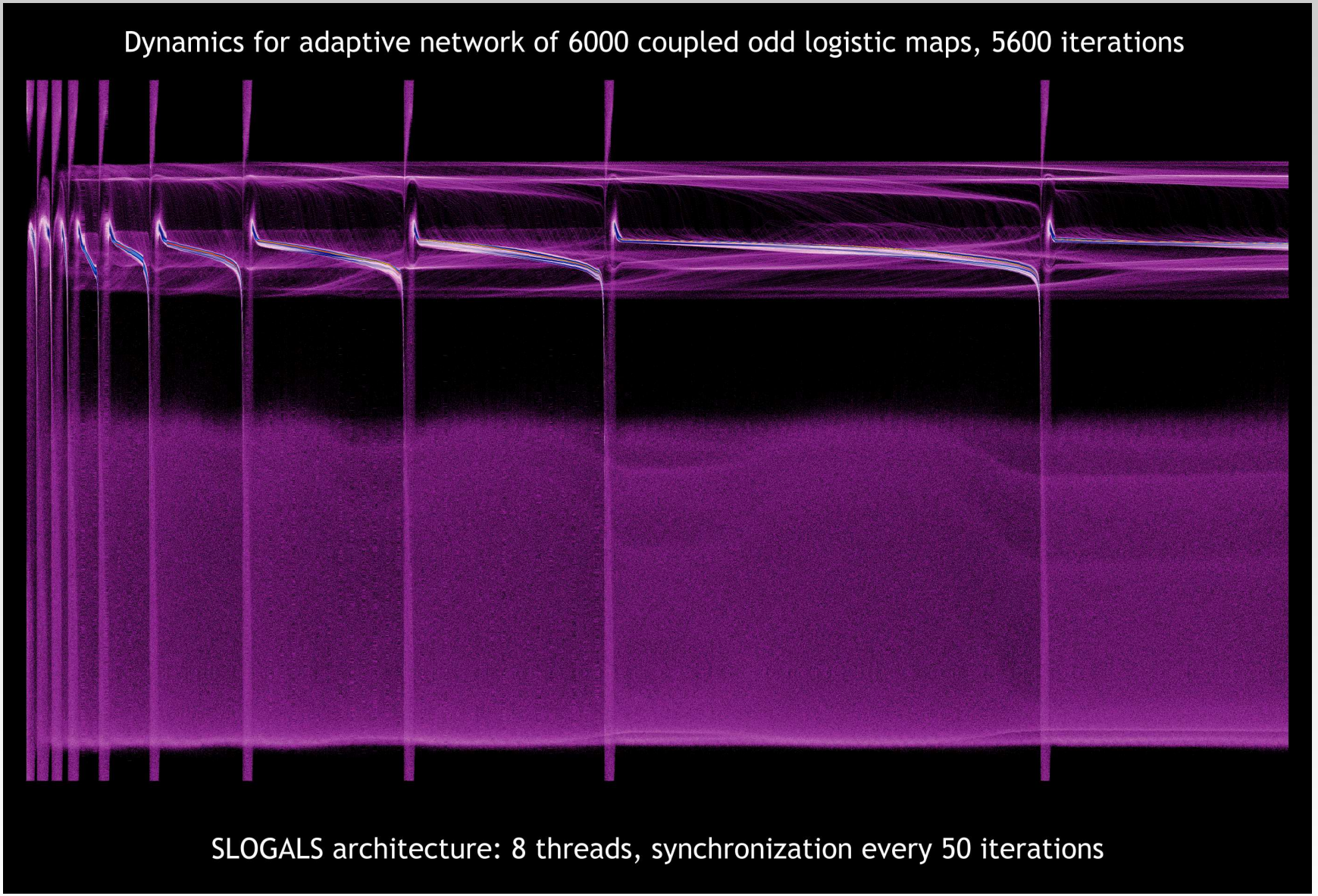
Dynamics: 6000 nodes



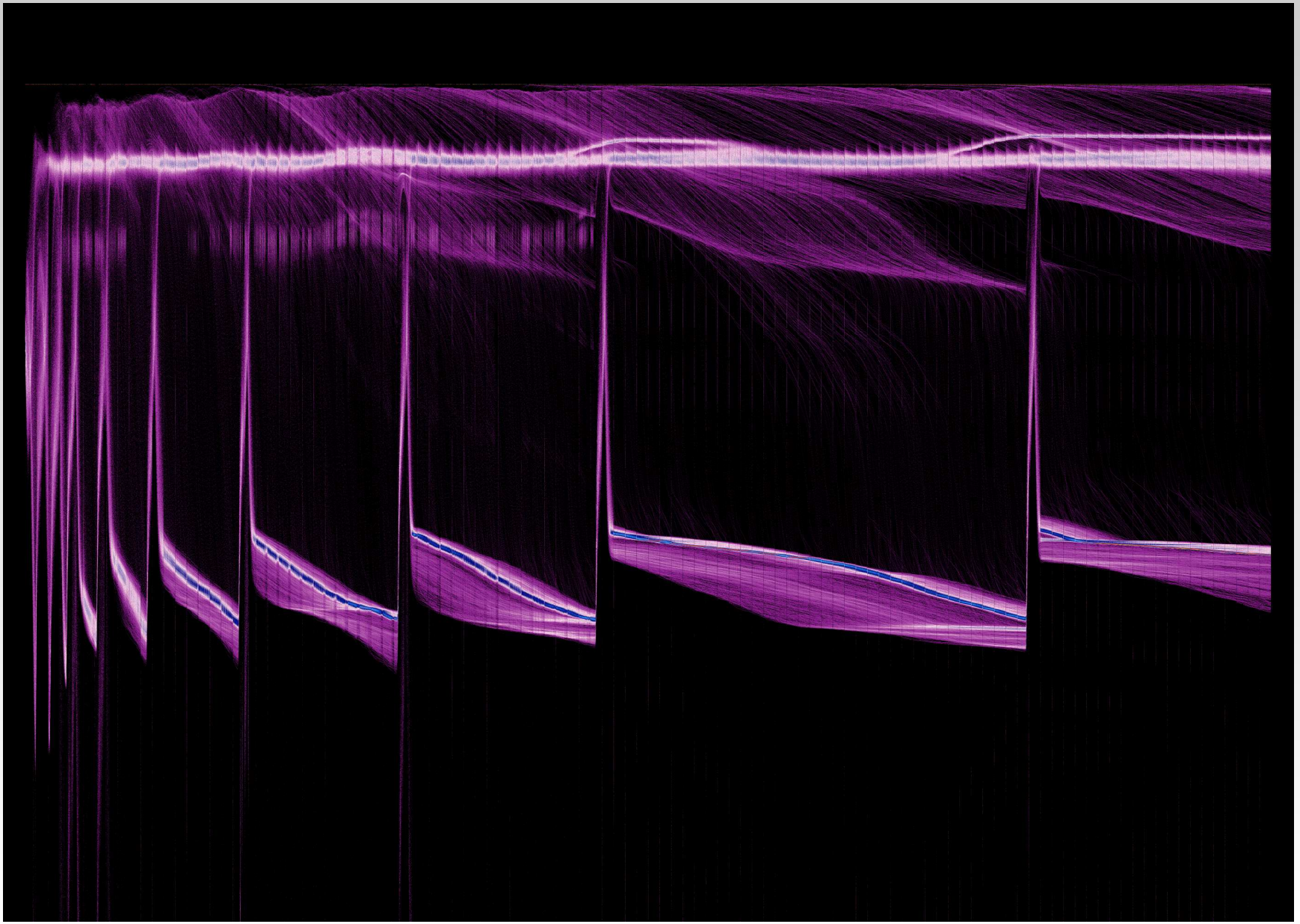
Weight Dynamics: 6000 nodes



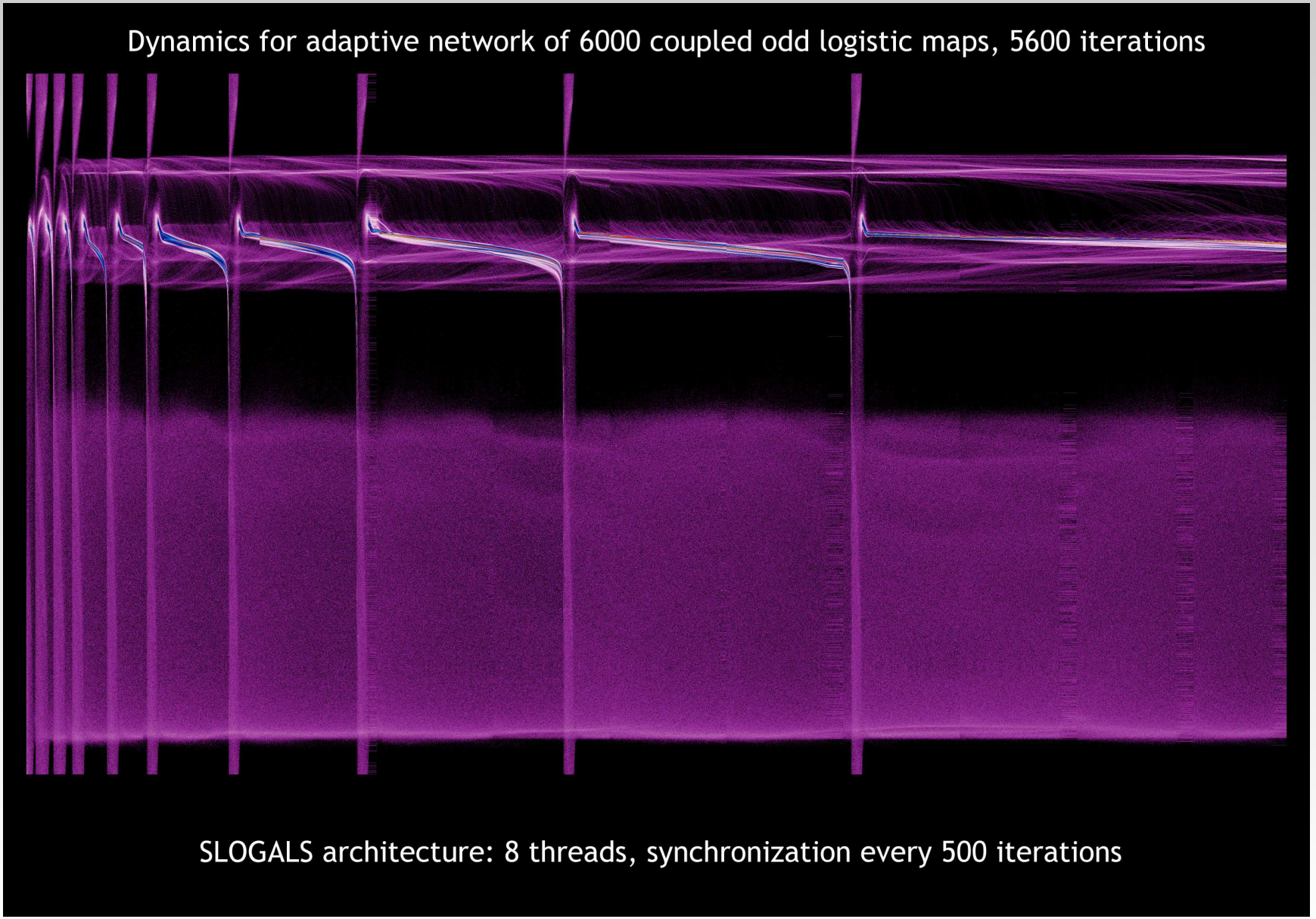
Dynamics: 6000 nodes



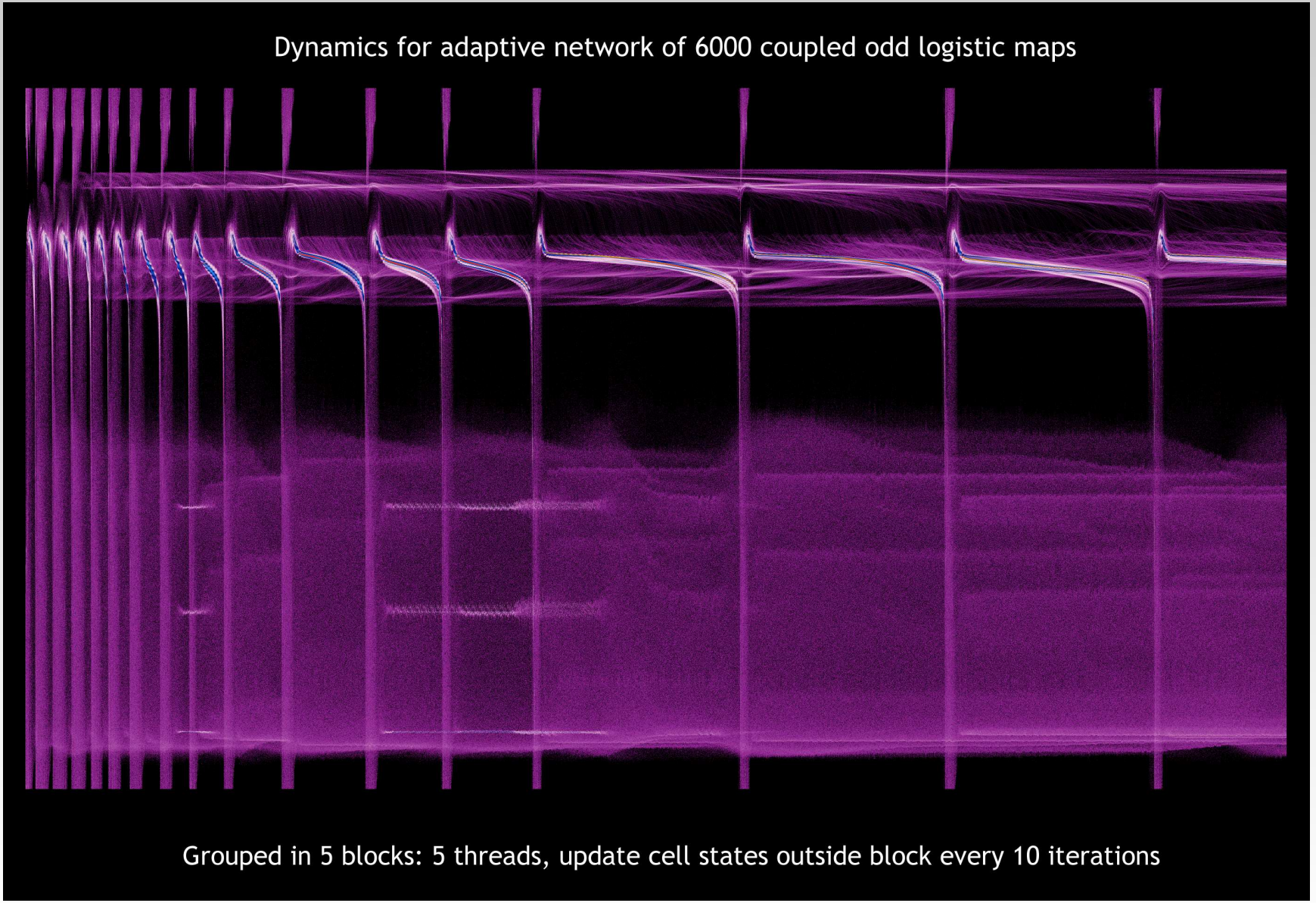
Weight Dynamics: 6000 nodes



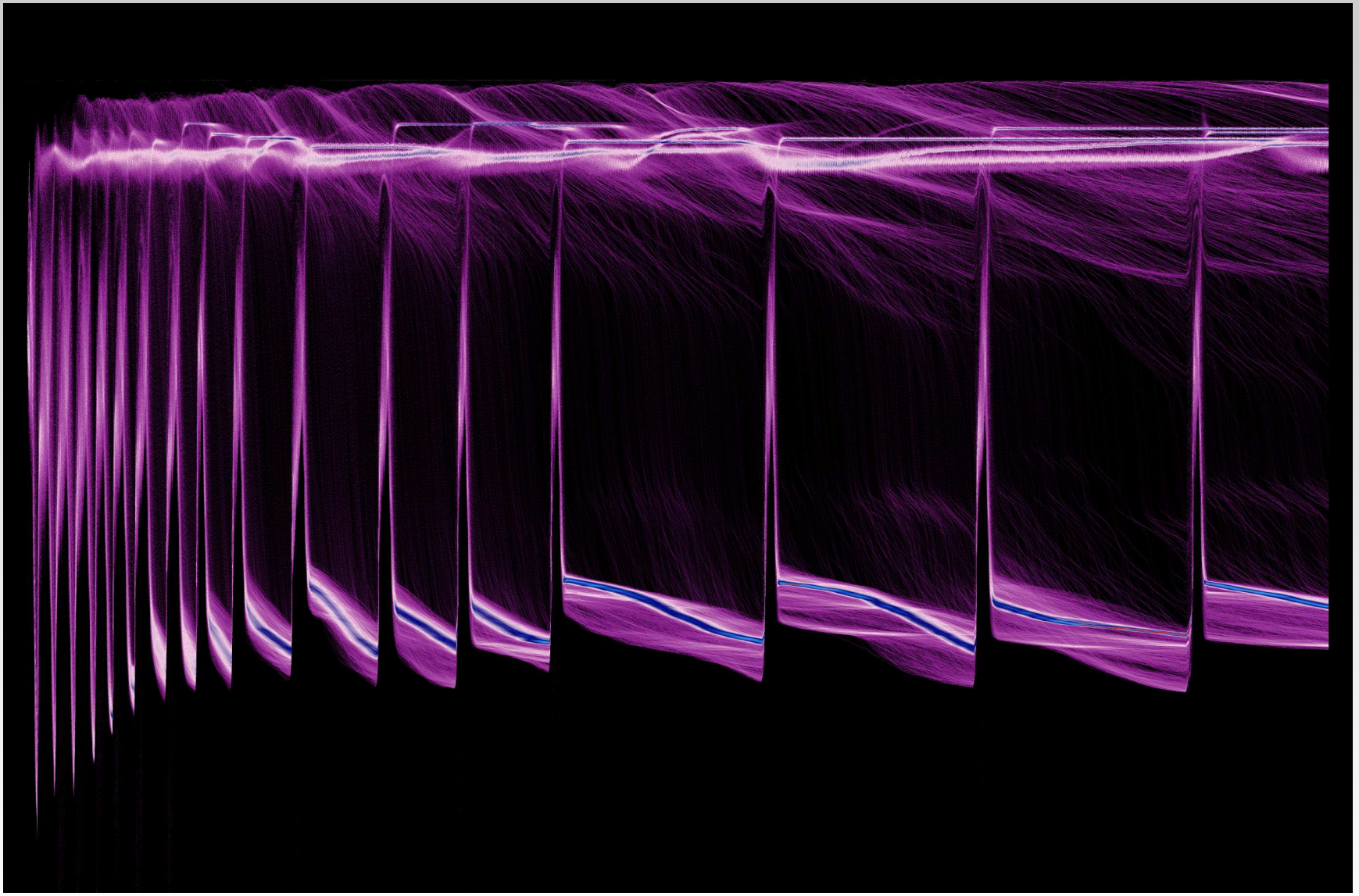
Dynamics: 6000 nodes



Dynamics: 6000 nodes



Weight Dynamics: 6000 nodes



Dynamics: STDP

STDP is short for *Spike-Timing Dependant Plasticity*.

STDP is a mechanism for adaptivity in (biological) networks which depends on relative timings. It is an example of a *Hebbian* learning rule (unsupervised or correlation based learning):

Cells that fire together wire together

Dynamics: STDP

STDP is short for *Spike-Timing Dependant Plasticity*.

STDP is a mechanism for adaptivity in (biological) networks which depends on relative timings. It is an example of a *Hebbian* learning rule (unsupervised or correlation based learning):

Cells that fire together wire together

The Barn Owl: Gerstner, Kemptner, Van Hemmen & Wagner, *Nature* 1996. Rapid direction finding to within $1-2$ degrees by encoding signals requiring a time resolution beyond $5\mu s$ – an order of magnitude faster than time constants of owl's neurons.

Dynamics: STDP

STDP is short for *Spike-Timing Dependant Plasticity*.

STDP is a mechanism for adaptivity in (biological) networks which depends on relative timings. It is an example of a *Hebbian* learning rule (unsupervised or correlation based learning):

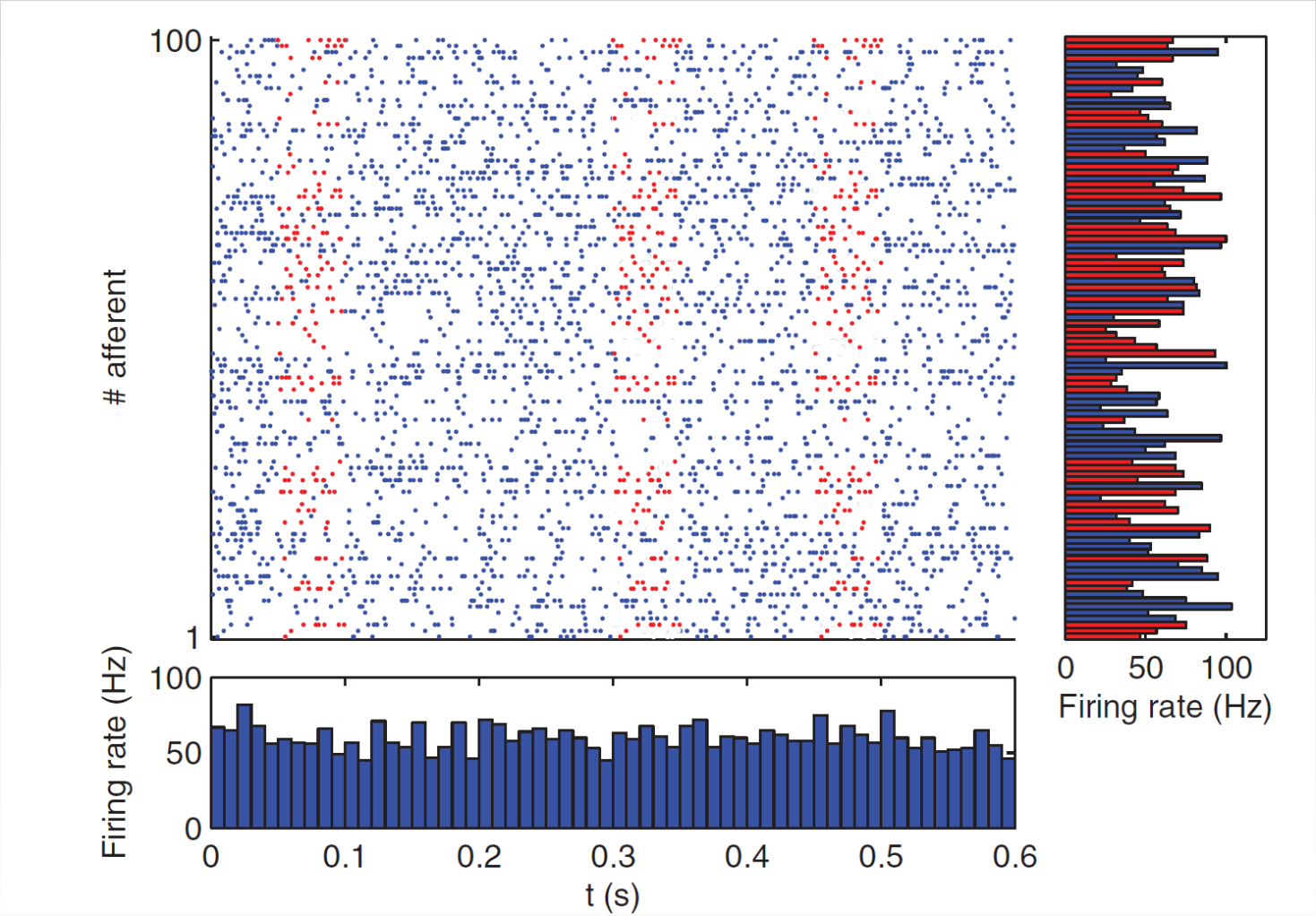
Cells that fire together wire together

The Barn Owl: Gerstner, Kemptner, Van Hemmen & Wagner, *Nature* 1996. Rapid direction finding to within $1-2$ degrees by encoding signals requiring a time resolution beyond $5\mu s$ – an order of magnitude faster than time constants of owl's neurons.

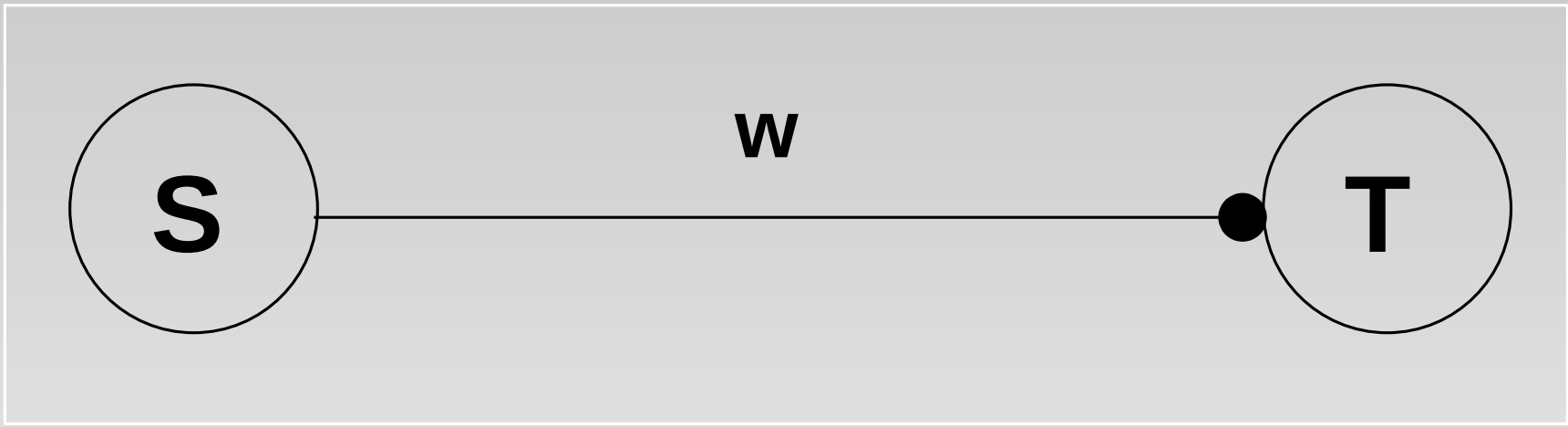
Proposed mechanism: STDP – based on delays & interaural time differences.

STDP: Pattern detection

Image from Masquelier, Guyonneau & Thorpe (2008, PLoS One)



Some details



Assume the neuron S emits spike train

$$S(t) = \sum \delta(t - t_i^S),$$

where $\cdots < t_i^S < t_{i+1}^S < \cdots$.

STDP ctd

Similarly assume the neuron T has spike train

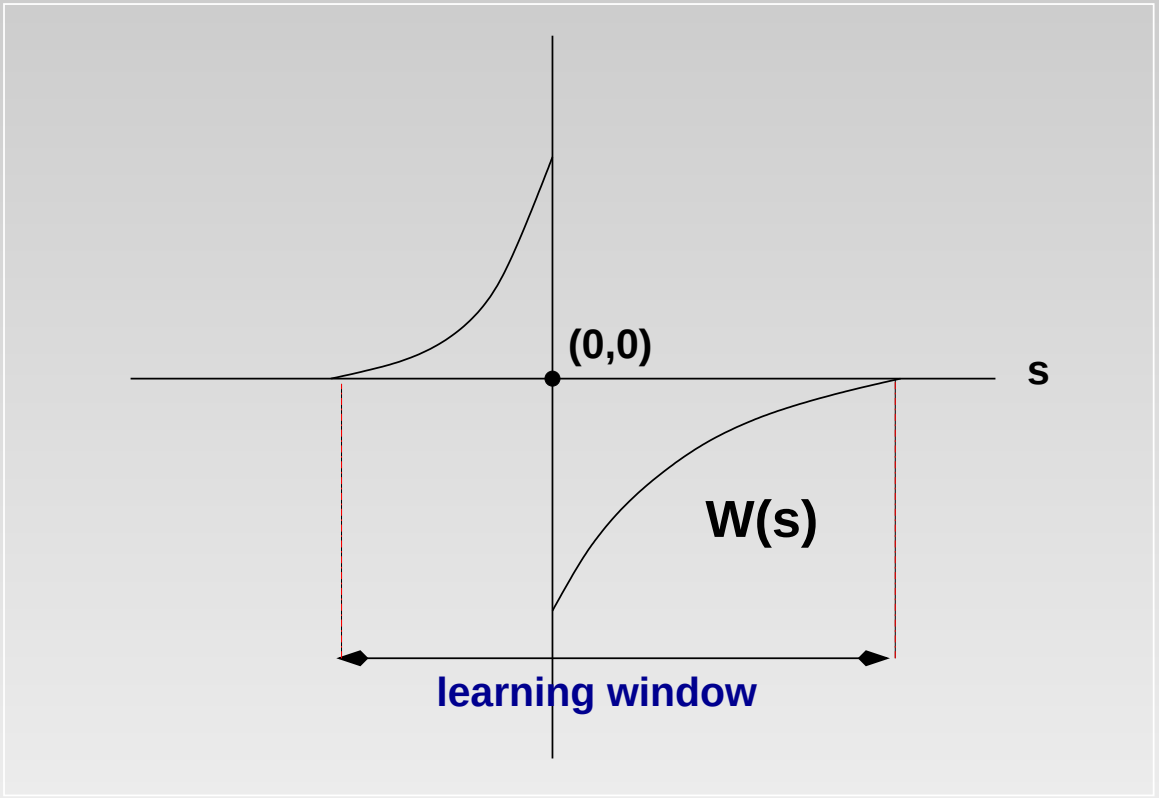
$$T(t) = \sum \delta(t - t_i^T),$$

where $\cdots < t_i^S < t_{i+1}^S < \cdots$.

Assume the connection is excitatory ($w > 0$). The basic idea is that if T fires just after S, we regard the firing as having been 'caused' by S and increase the coupling strength w; if T fires just before S, there is no causality and we weaken the coupling strength w.

More formally, we use a function $W(s)$ that defines a *learning window*.

STDP ctd



Note: usually assume $\int W < 0$. If $t_i^S - t_j^T$ is in the learning window, then we change w by

$$\Delta(w) = \eta H(w) W(t_i^S - t_j^T).$$

STDP ctd

Here $\eta > 0$ (typically $\eta \ll 1$) and $H(w) = w^\mu$. If $t_i^S < t_j^T$ (causality), then $\Delta(w) > 0$.

Assume (simpler) additive case: $H(w) = 1$. Over a learning session of time T_ℓ, we take

$$\Delta(w)(t) = \eta \sum_{t_i^S, t_j^T \in I} W(t_i^S - t_j^T),$$

where $I = [t - T_\ell, t]$.

One approach to developing a mean field model of STDP, due to Burkitt, Gilson, Hemmen et al., is to assume that firings follow an inhomogeneous Poisson statistic ('Poisson neurons'):

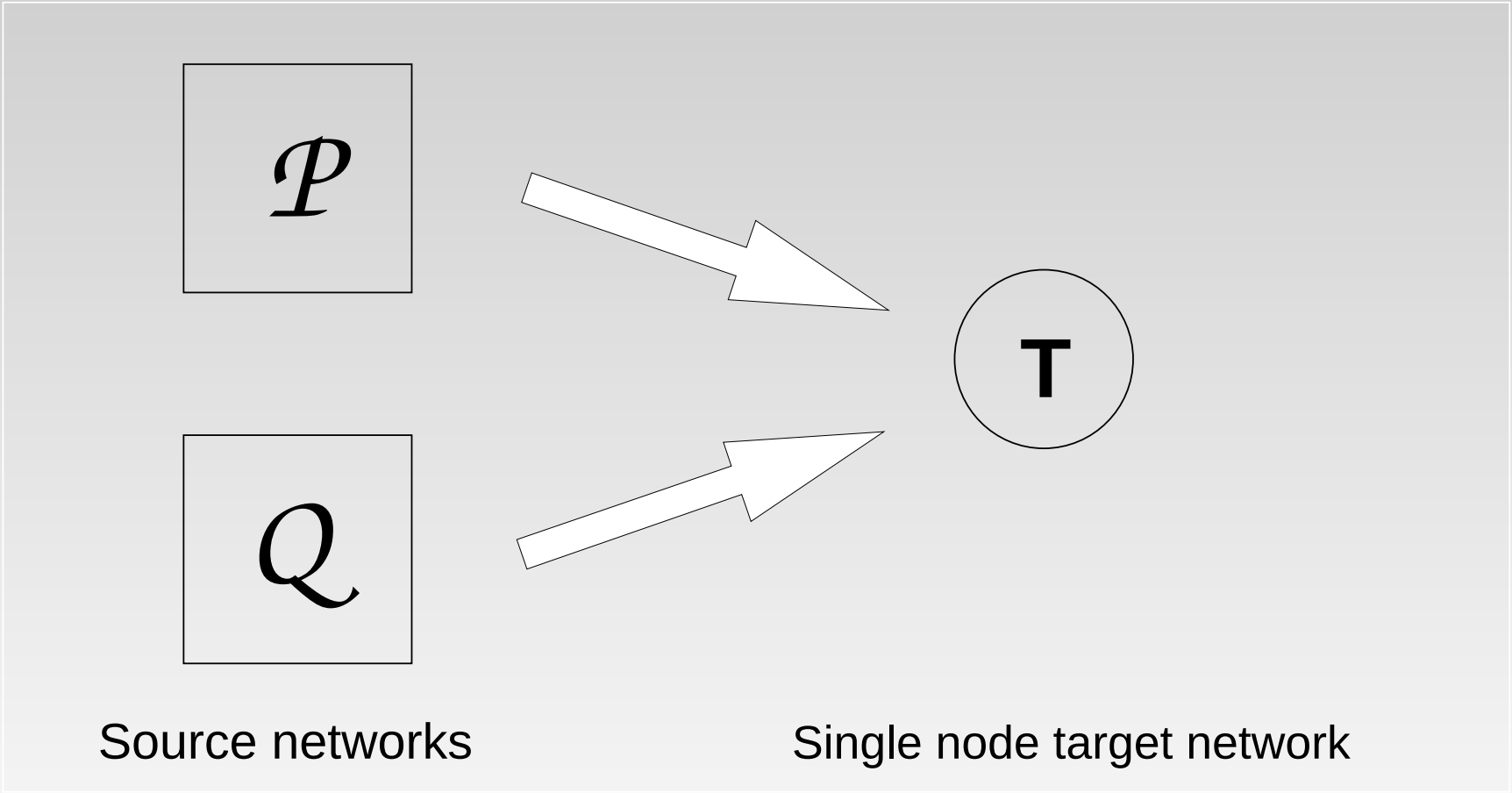
STDP: Mean Field Model

Probability of 1 firing in $[t, t+\Delta t] = \lambda_i(t)\Delta t$,
Probability of ≥ 2 firings in $[t, t+\Delta t] = o(\Delta t)$.
Firings in disjoint intervals independent.

Under appropriate assumptions on the time scales (eg slow learning compared with firing rates and changes in $\lambda_i(t)$ small in learning session: adiabatic hypothesis) Burkitt et al develop a mean field model of STDP for quite general recurrent networks of spiking neurons subject to inputs from Poisson neurons (the latter with fixed Poisson rates). Their model can and does incorporate delays and yields an ODE model for evolution of weights.

Adaptation & Dynamics Detection

We consider dynamics detection using STDP.



Two source networks connected all-to-1 to a target network consisting of a single node.

Dynamics

Both networks $\mathcal{P}$, $\mathcal{Q}$ consist of coupled phase oscillators — in regimes where the oscillators will eventually frequency synchronize or do something else “interesting”...

Each time a node state passes through 1, the oscillator fires a spike.

All oscillators are connected to target node – each connection has weight $w \in [0, 1]$.

Sum inputs into $\mathbf{T}$. If sum exceeds a threshold, $\mathbf{T}$ fires and its state is reset to 0. (Various protocols allowed: SRM_0, SRM & gated.)

Adaptation: STDP

We adapt weights according to STDP.

If $\mathbf{T}$ fires (shortly) after a node $N \in \mathcal{P} \cup \mathcal{Q}$ fires, we regard the firing of N as having caused the firing of $\mathbf{T}$ and strengthen the weight of the connection between N and $\mathbf{T}$. Conversely if $\mathbf{T}$ fires (shortly) before a node $N \in \mathcal{P} \cup \mathcal{Q}$ fires, we regard the events as uncorrelated and weaken the weight of the connection between N and $\mathbf{T}$.

This form of adaptation is called Spike-Timing Dependent Plasticity in computational neuroscience.

We illustrate with some numerical examples. ◀ ▶

Mathematical challenges

1. How does asynchronicity impact dynamics?
2. How do we analyze without the assumption that equations are analytic?
3. Bifurcation theory in adaptive spiking networks (STDP) – many interesting questions and phenomena already.
4. Understanding how & why asynchronous networks can work correctly (most of the time) notwithstanding the fragility and complexity of asynchronous logic. On the neuro-computation side, the basic mechanisms may not be so hard to understand – evolution can lend a helping hand. With stochastic asynchronous networks, analysis may be much easier than in the deterministic case! Applications to 'Qualitative Computing'.