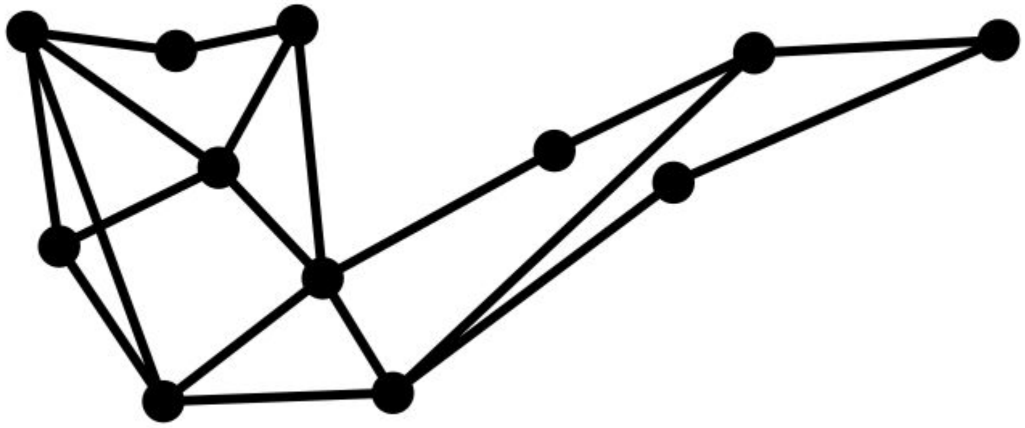
Topological Structure in the Study of Ablation and Desynchronization in Neural Networks

Mark Agrios
College of William & Mary

Quick Review of Graphs



Ceci n'est pas une graph

$$G = (V,E)$$

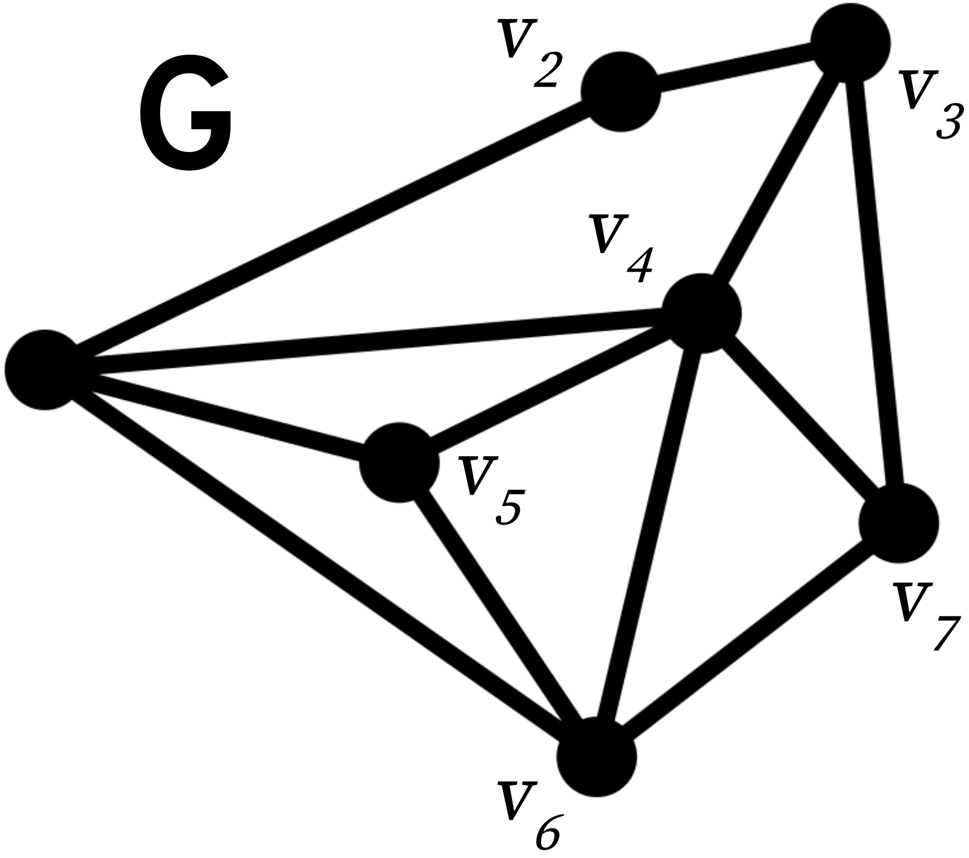
$$V = \{v_1,v_2,v_3,v_4,v_5,v_6,v_7\}$$

$$E = \boldsymbol{\{}\{v_1,v_2\},\{v_1,v_4\},\{v_1,v_5\},\{v_1,v_6\},$$
$$\{v_2,v_3\},\{v_3,v_4\},\{v_3,v_7\},\{v_4,v_5\},$$
$$\{v_4,v_6\},\{v_4,v_7\},\{v_5,v_6\},\{v_6,v_7\}\boldsymbol{\}}$$

$G = (V,E)$

$V = \{v_1,v_2,v_3,v_4,v_5,v_6,v_7\}$

$E = \boldsymbol{\{}\{v_1,v_2\},\{v_1,v_4\},\{v_1,v_5\},$
$\{v_1,v_6\},\{v_2,v_3\},\{v_3,v_4\},\{v_3,v_7\},$
$\{v_4,v_5\},\{v_4,v_6\},\{v_4,v_7\},\{v_5,v_6\},$
$\{v_6,v_7\}\boldsymbol{\}}$



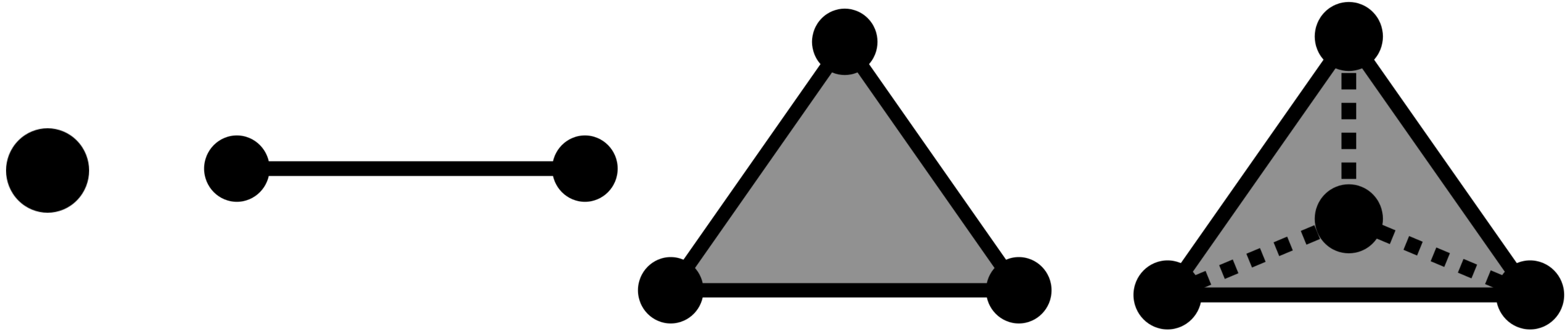
Simplex: a generalization of the notion of a triangle or tetrahedron to arbitrary dimensions

An n-simplex is an n-dimensional polytope that is the convex hull of n+1 vertices.

Simplex: a generalization of the notion of a triangle or tetrahedron to arbitrary dimensions

An n-simplex is an n-dimensional polytope that is the convex hull of n+1 vertices.

0-simplex **1-simplex** **2-simplex** **3-simplex** **etc…**



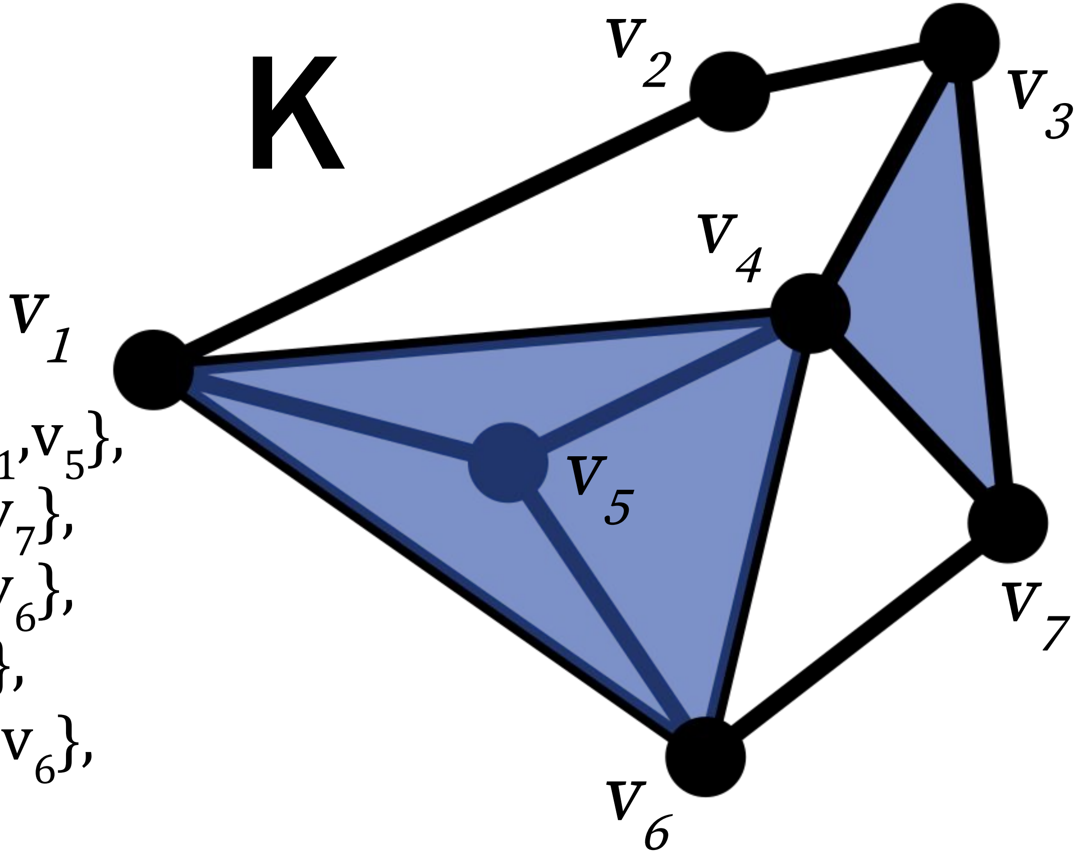
Simplicial Complex: A simplicial complex, *K*, is a set of "simplices" that are sets of finite order.

Simplicial Complex Condition:

1. If $a \in K$ then any subset (face) of a is also in K
2. The intersection of any two simplices $a,b \in$ K is either Ø or a face of both a and b

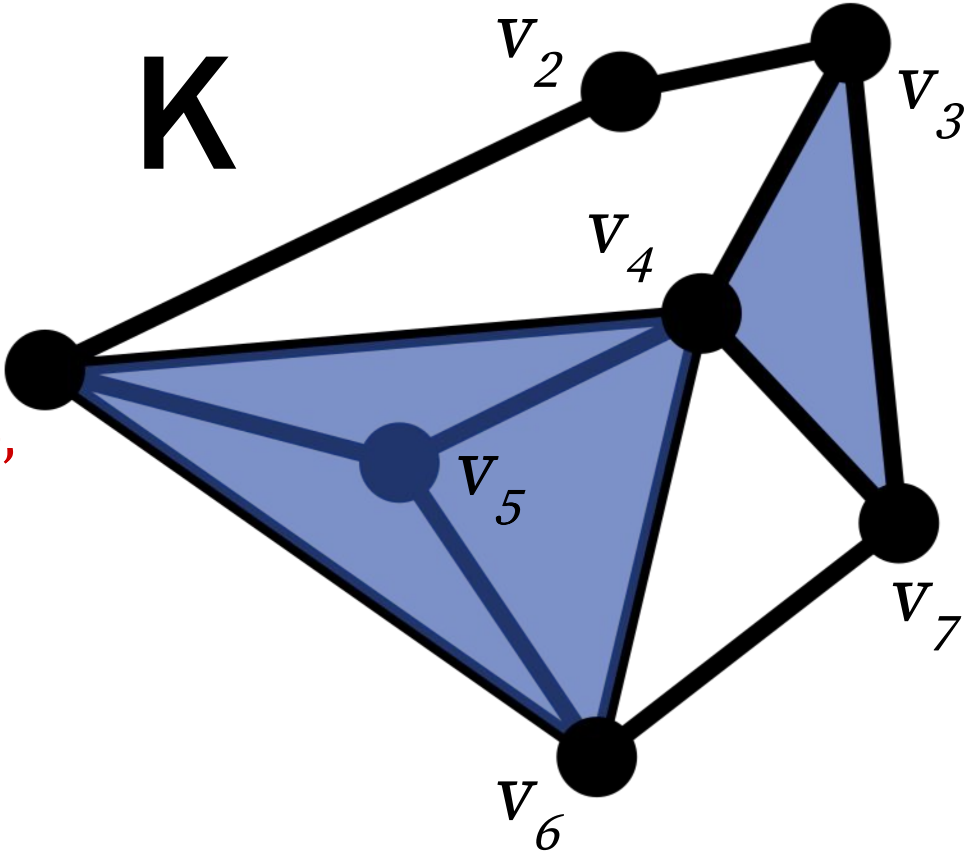
$K =$

$\{\{v_1\},\{v_2\},\{v_3\},\{v_4\},\{v_5\},$
$\{v_6\},\{v_7\},\{v_1,v_2\},\{v_1,v_4\},\{v_1,v_5\},$
$\{v_1,v_6\},\{v_2,v_3\},\{v_3,v_4\},\{v_3,v_7\},$
$\{v_4,v_5\},\{v_4,v_6\},\{v_4,v_7\},\{v_5,v_6\},$
$\{v_6,v_7\},\{v_1,v_4,v_6\},\{v_1,v_5,v_6\},$
$\{v_1,v_4,v_5\},\{v_1,v_4,v_6\},\{v_4,v_5,v_6\},$
$\{v_3,v_4,v_7\}\}$



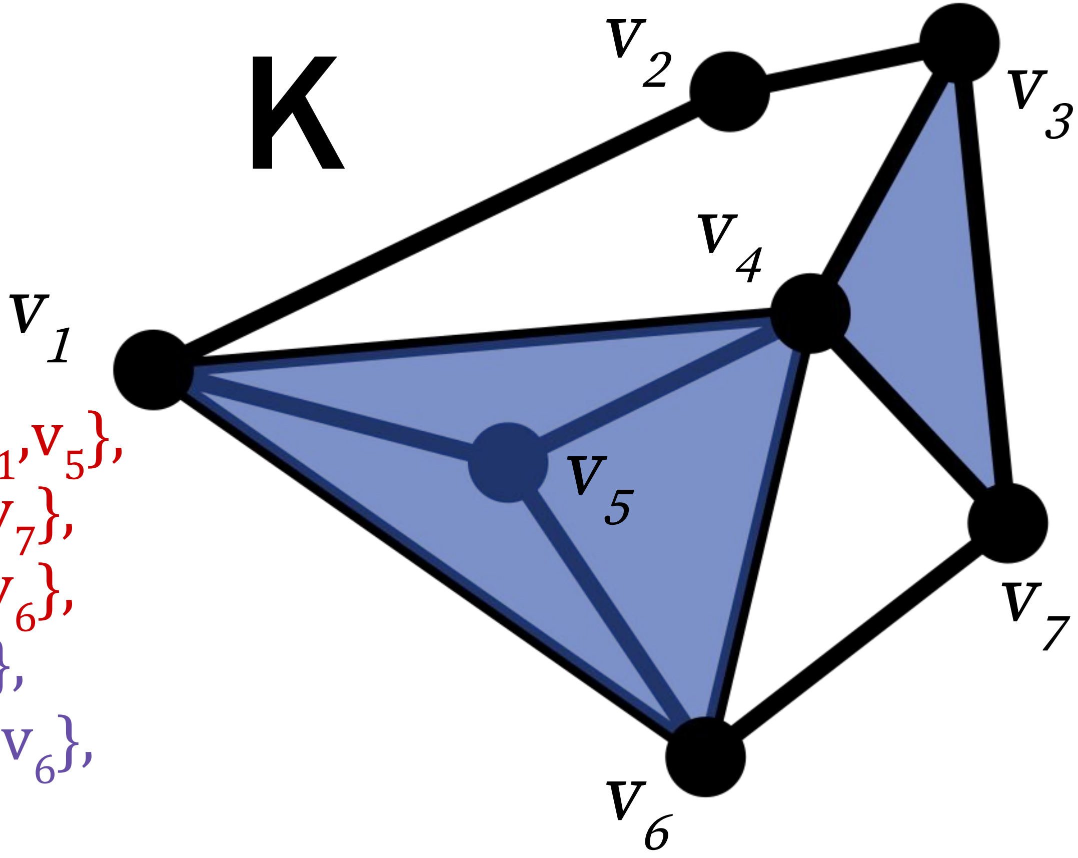
$K =$

$\{\{v_1\},\{v_2\},\{v_3\},\{v_4\},\{v_5\},$
$\{v_6\},\{v_7\},\{v_1,v_2\},\{v_1,v_4\},\{v_1,v_5\},$
$\{v_1,v_6\},\{v_2,v_3\},\{v_3,v_4\},\{v_3,v_7\},$
$\{v_4,v_5\},\{v_4,v_6\},\{v_4,v_7\},\{v_5,v_6\},$
$\{v_6,v_7\},\{v_1,v_4,v_6\},\{v_1,v_5,v_6\},$
$\{v_1,v_4,v_5\},\{v_1,v_4,v_6\},\{v_4,v_5,v_6\},$
$\{v_3,v_4,v_7\}\}$



$K =$

$\{\{v_1\},\{v_2\},\{v_3\},\{v_4\},\{v_5\},$
$\{v_6\},\{v_7\},\{v_1,v_2\},\{v_1,v_4\},\{v_1,v_5\},$
$\{v_1,v_6\},\{v_2,v_3\},\{v_3,v_4\},\{v_3,v_7\},$
$\{v_4,v_5\},\{v_4,v_6\},\{v_4,v_7\},\{v_5,v_6\},$
$\{v_6,v_7\},\{v_1,v_4,v_6\},\{v_1,v_5,v_6\},$
$\{v_1,v_4,v_5\},\{v_1,v_4,v_6\},\{v_4,v_5,v_6\},$
$\{v_3,v_4,v_7\}\}$



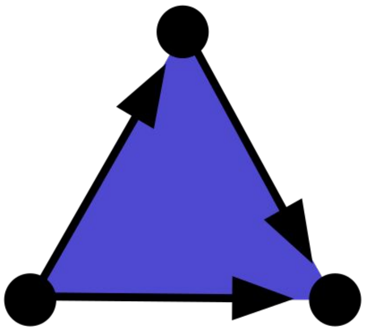
$\beta_0 = 1, \quad \beta_1 = 2, \quad \beta_2 = 1$

Directed cliques and why we care.

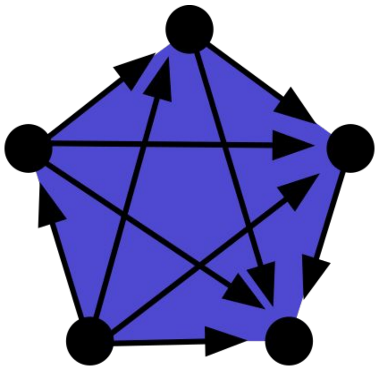
Directed clique: An all-to-all connected subgraph of the network with a clear “source” and “sink”

Directed cliques and why we care.

Directed clique: An all-to-all connected subgraph of the network with a clear “source” and “sink”



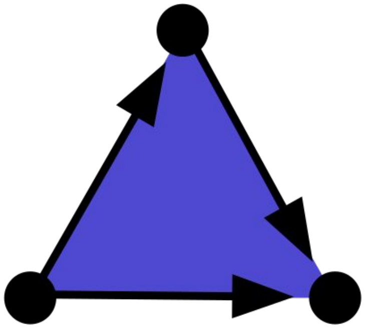
Directed 3-clique



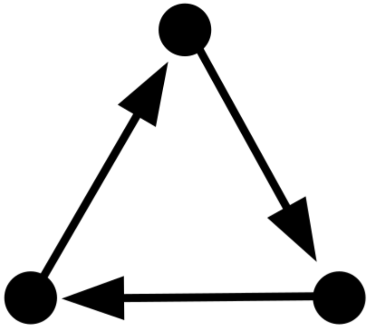
Directed 5-clique

Directed cliques and why we care.

Directed clique: An all-to-all connected subgraph of the network with a clear “source” and “sink”



Directed
3-clique

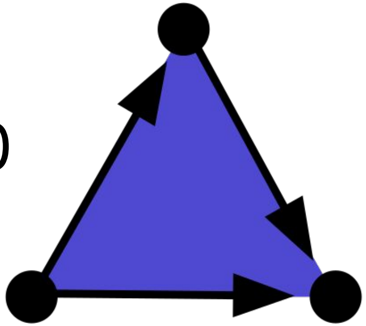


Directed
3-cycle

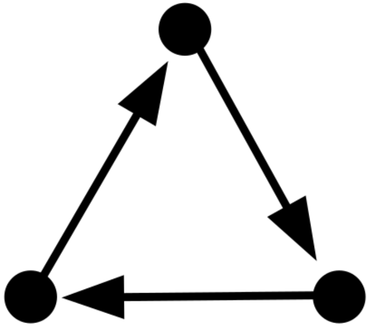
Directed cliques and why we care.

Directed clique: An all-to-all connected subgraph of the network with a clear “source” and “sink”

$\beta_1 = 0$



Directed 3-clique



$\beta_1 = 1$

Directed 3-cycle

Directed cliques give us a condition to create simplicial complexes which emphasize the direction that information flows in neural networks.

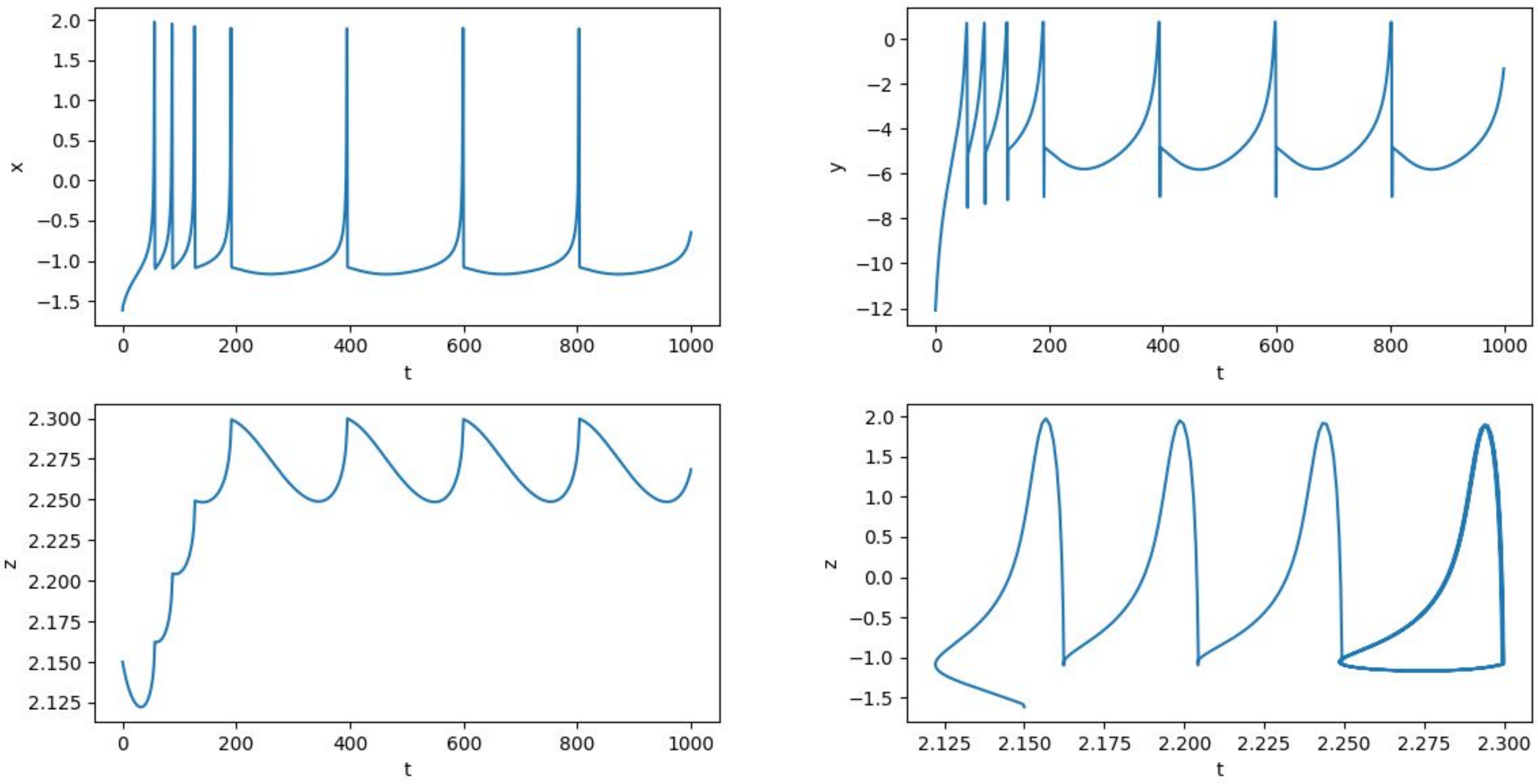
Hindmarsh-Rose Neuron Model

Hindmarsh-Rose Neuron Model

$$\dot{x} = y - ax^3 + bx^2 + I - z$$

$$\dot{y} = c - dx^2 - y$$

$$\dot{z} = r(s(x - x_{rest}) - z)$$


x
t
y
t
z
t
z
t

Hindmarsh-Rose Neuron Model

$$\dot{x}_{post} = y_{post} - ax_{post}^3 + bx_{post}^2 + I - z_{post} - gA_{i,j}(x_{post} - V_0) \sum \Gamma(x_{pre})$$

$$\dot{y}_{post} = c - dx_{post}^2 - y_{post}$$

$$\dot{z}_{post} = r(s(x_{post} - x_{rest}) - z_{post})$$

Hindmarsh-Rose Neuron Model

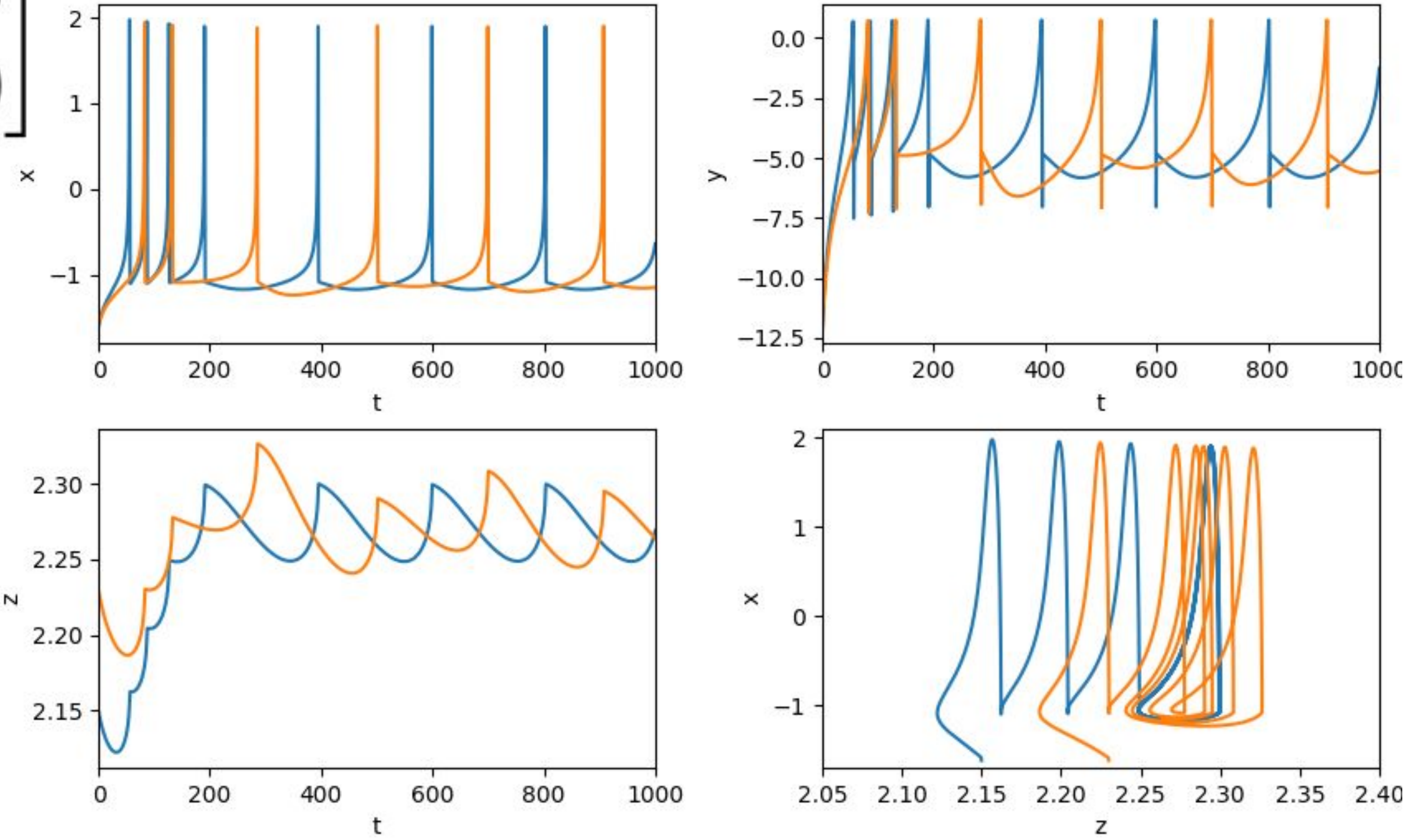
$$\dot{x}_{post} = y_{post} - a x_{post}^3 + b x_{post}^2 + I - z_{post} - \underline{g A_{i,j}(x_{post} - V_0) \sum \Gamma(x_{pre})}$$

$$\dot{y}_{post} = c - d x_{post}^2 - y_{post}$$

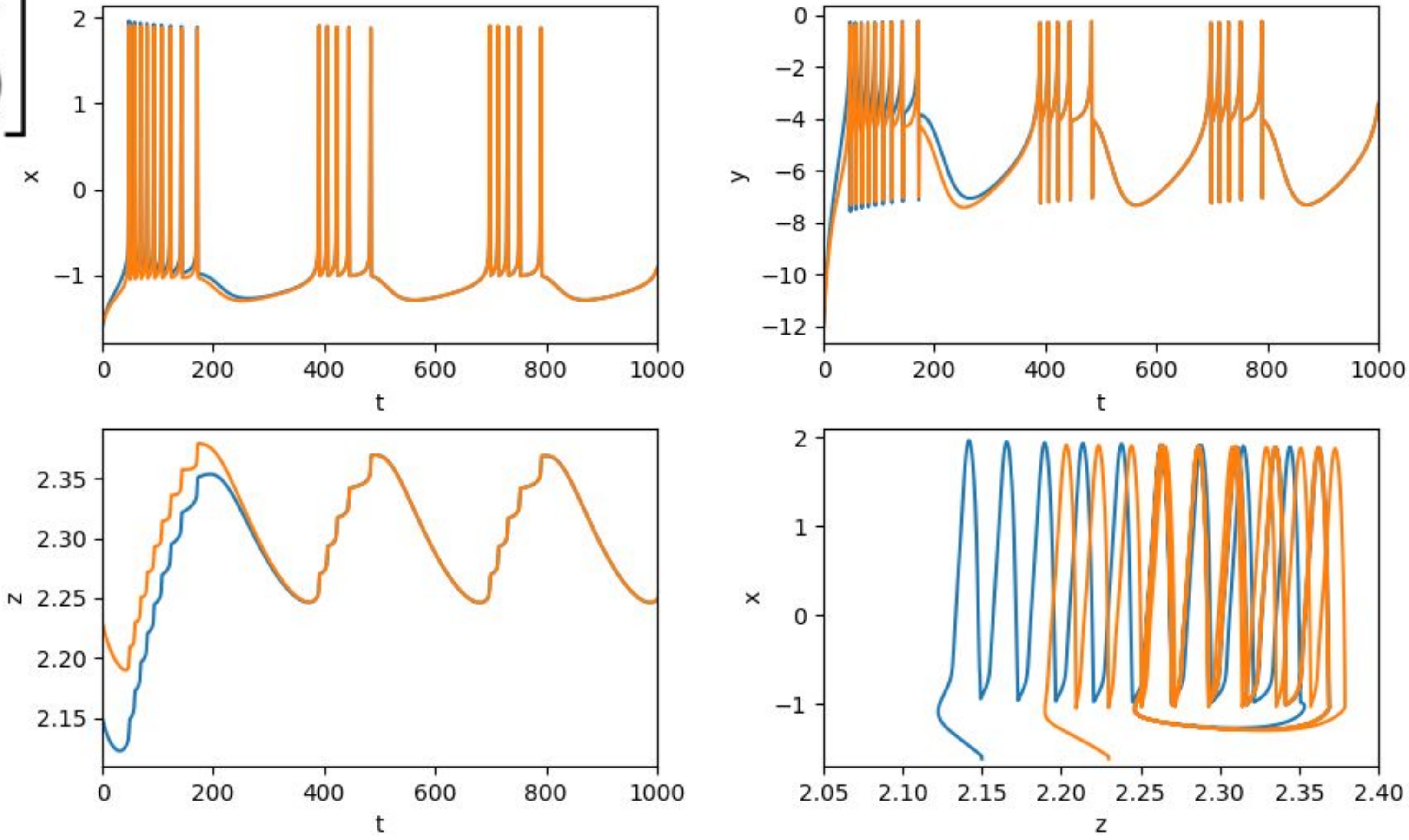
$$\dot{z}_{post} = r(s(x_{post} - x_{rest}) - z_{post})$$

$$\Gamma(x) = \frac{1}{1+e^{-\lambda(x-\Theta)}}$$

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$



$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$



Measuring synchrony of a network

For a time t, we can measure the synchrony of a network by looking at the variable z of each neuron because this one dictates when the neuron will spike or burst.

Measuring synchrony of a network

First, transform each neuron's z variable to a number between 0 and π using the following formula, where z_{min} and $z_{\max}$ are the extreme value z reaches during the simulation.

$$f(z) = \frac{z - z_{min}}{z_{max} - z_{min}} \cdot \pi$$

Measuring synchrony of a network

Now with each $\Theta_i = f(z_i)$ for each neuron i in the network, $\Theta_i \in [0,\pi]$ and so we can map each to a vector on the complex unit circle by $v_i = e^{i\theta_i}$

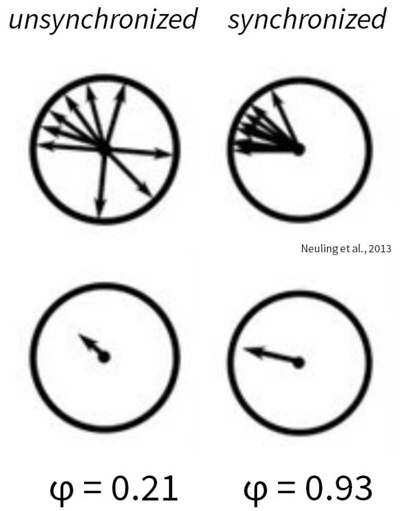
Measuring synchrony of a network

Then when all the vectors are averaged, the resulting vector will have a magnitude close to 1 if the starting vectors all point in the same direction and 0 if they point in different directions.

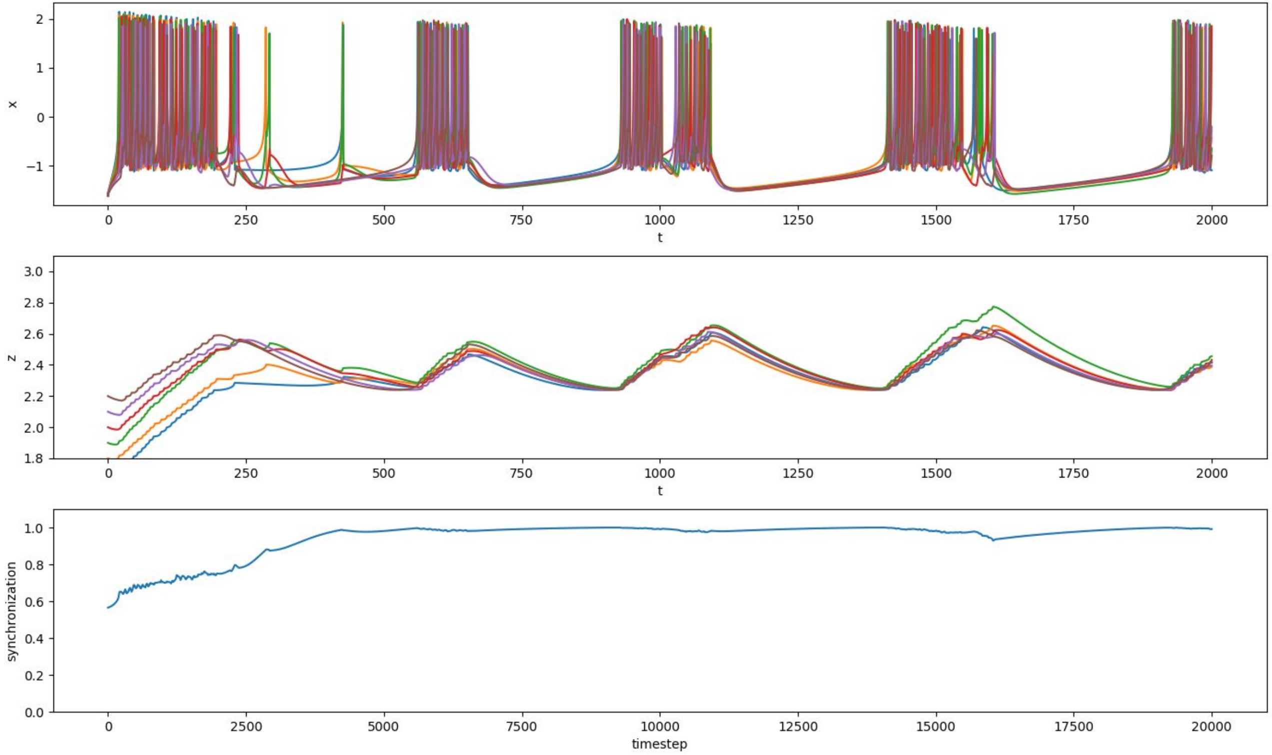
Measuring synchrony of a network

Then when all the vectors are averaged, the resulting vector will have a magnitude close to 1 if the starting vectors all point in the same direction and 0 if they point in different directions.

We take the resulting vector's magnitude as φ which gives us a numerical value of synchrony between 0 and 1 at each timestep in the simulation.



Measuring synchrony of a network



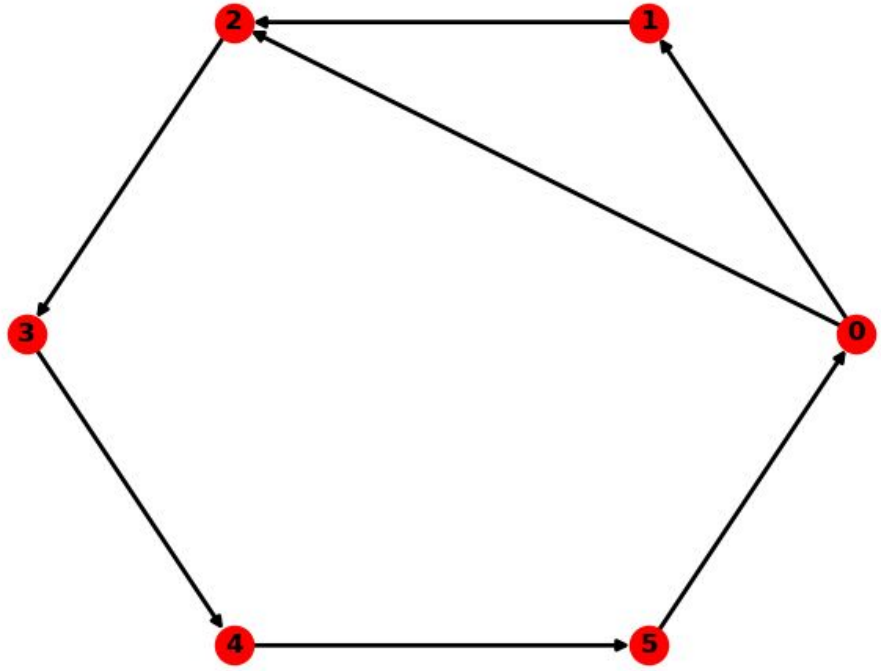
Measuring synchrony of a network extra part

To get another measure of synchrony for a network, we can also take the total variation of our time series of synchronization values. For $t = 0,1,\ldots,T$

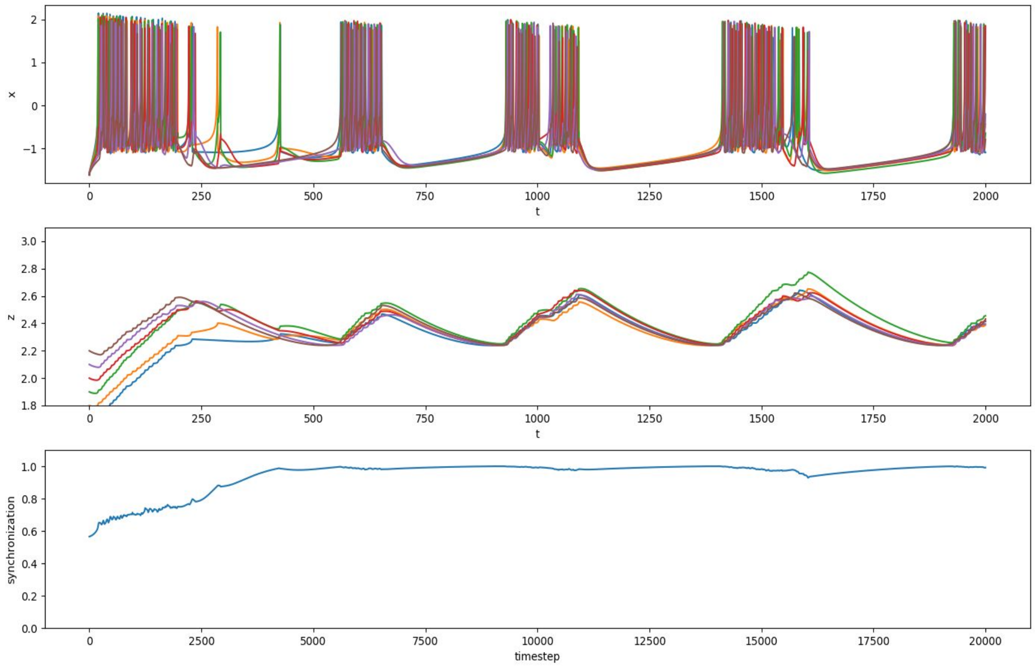
$$\Phi = \sum_{i=0}^{T} |\phi_{i+1} - \phi_i|$$

Small networks

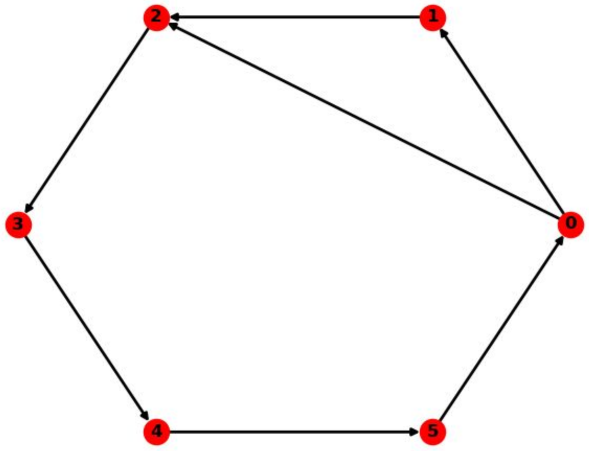
We start by running simulations on very small networks because ablating single synapses has a very small chance to have any detectable effect on large networks.



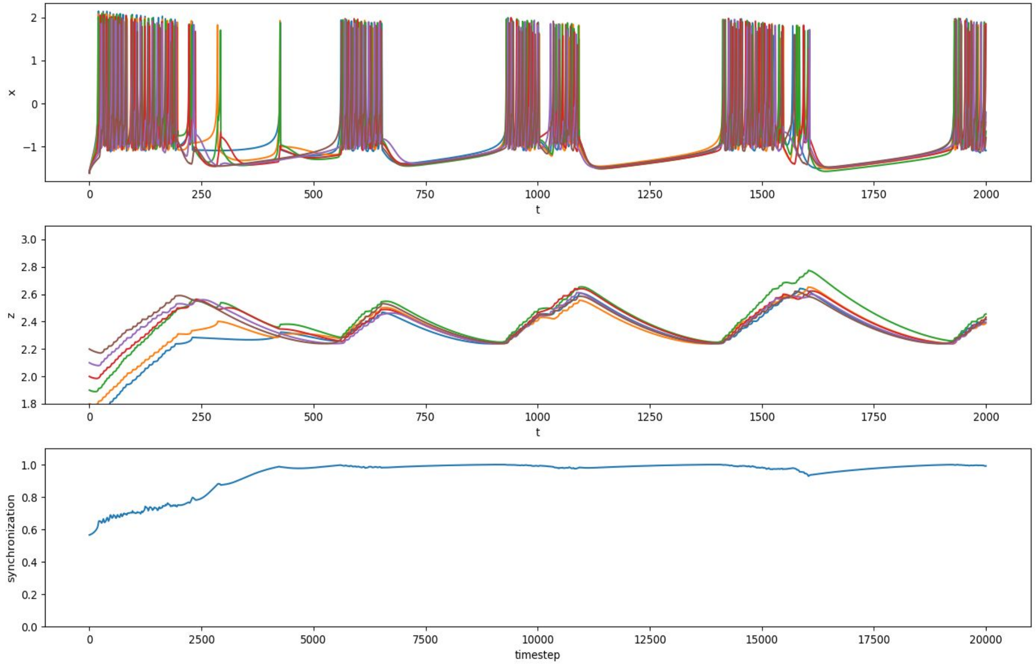
$\beta_0 = 1 \quad \beta_1 = 1$

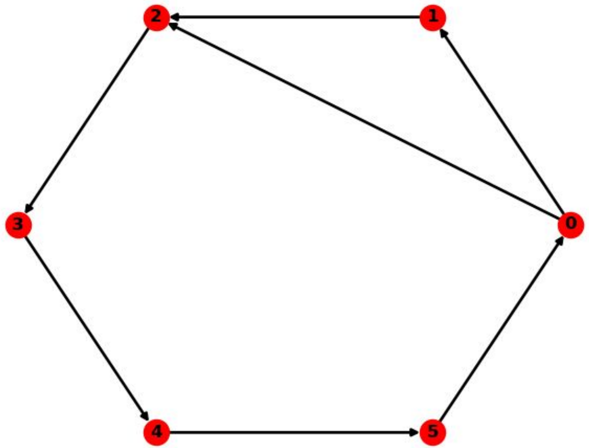


x
t
z
t
synchronization
timestep



0
1
2
3
4
5

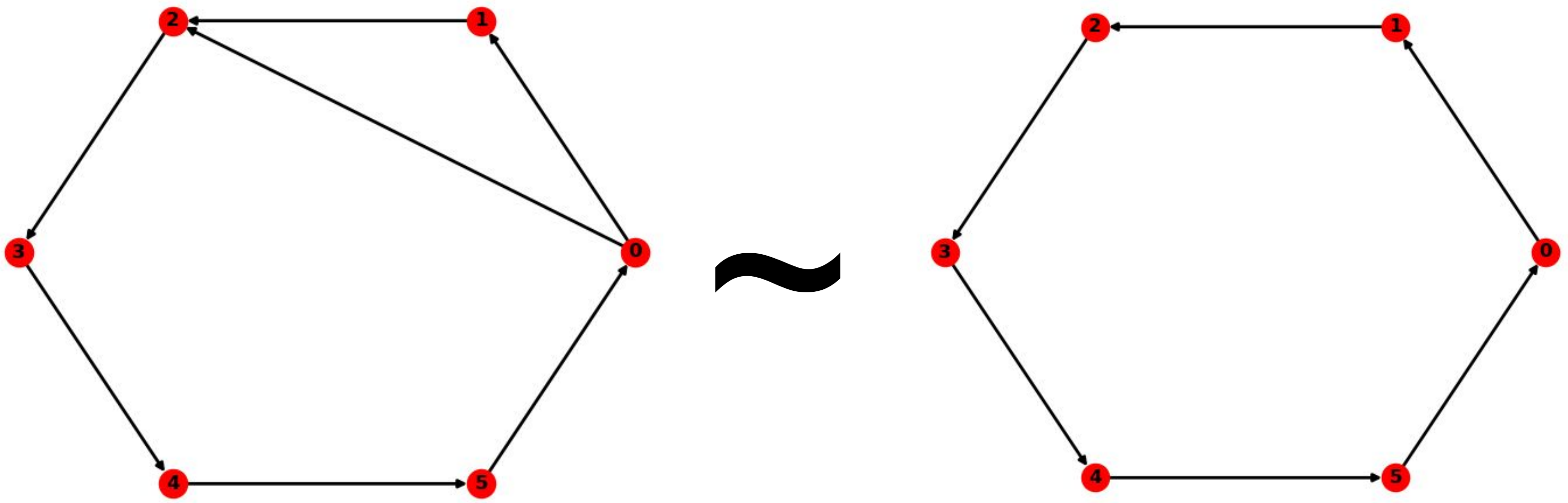


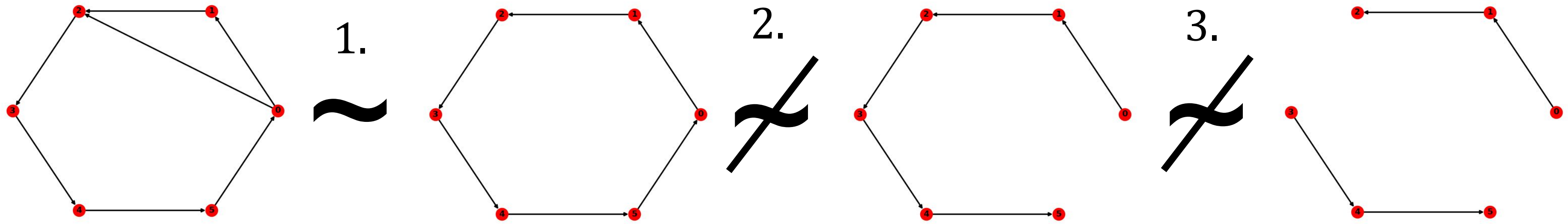


Let’s ablate a synapse that preserves the homology of this simplicial complex

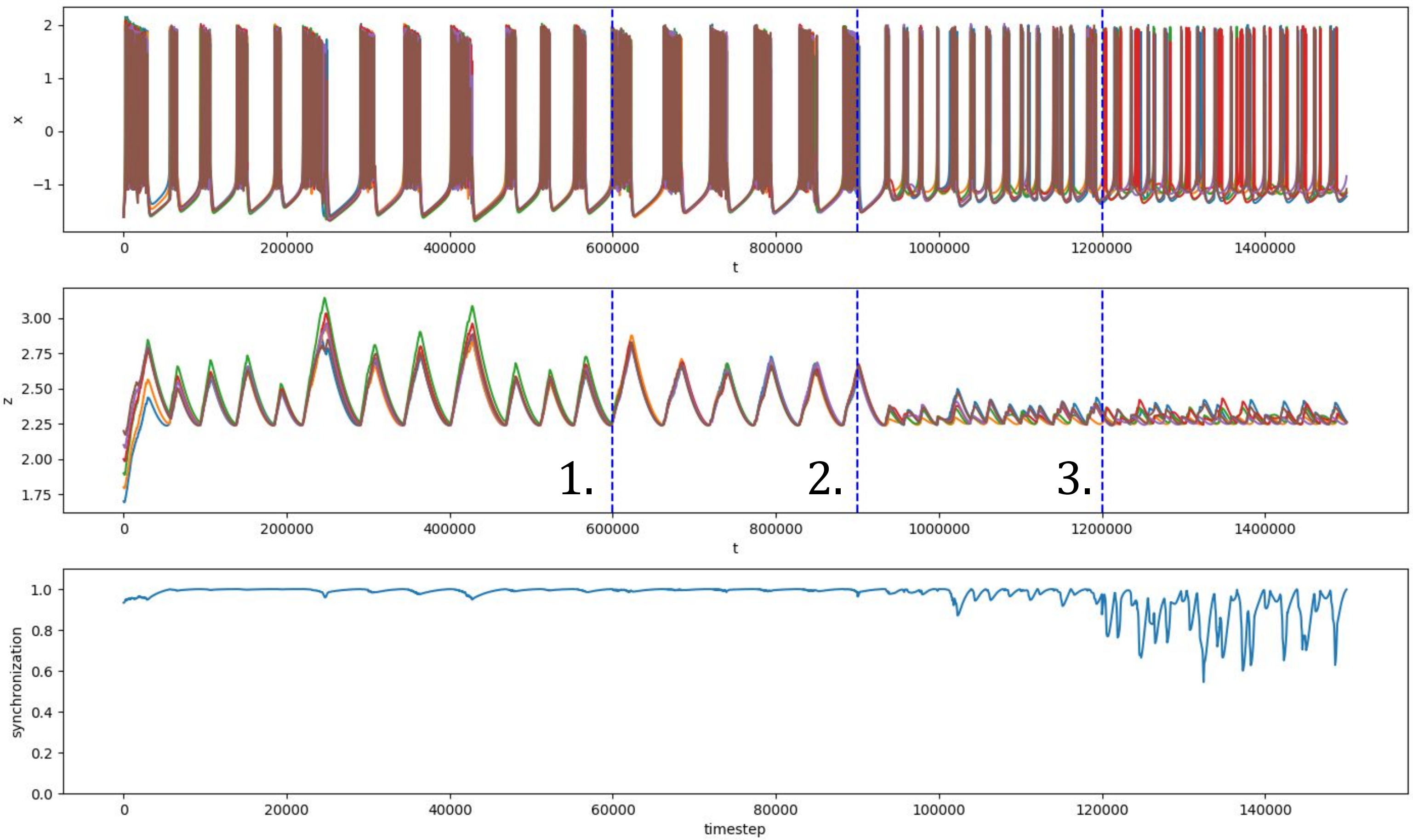
$\beta_0 = 1$

$\beta_1 = 1$

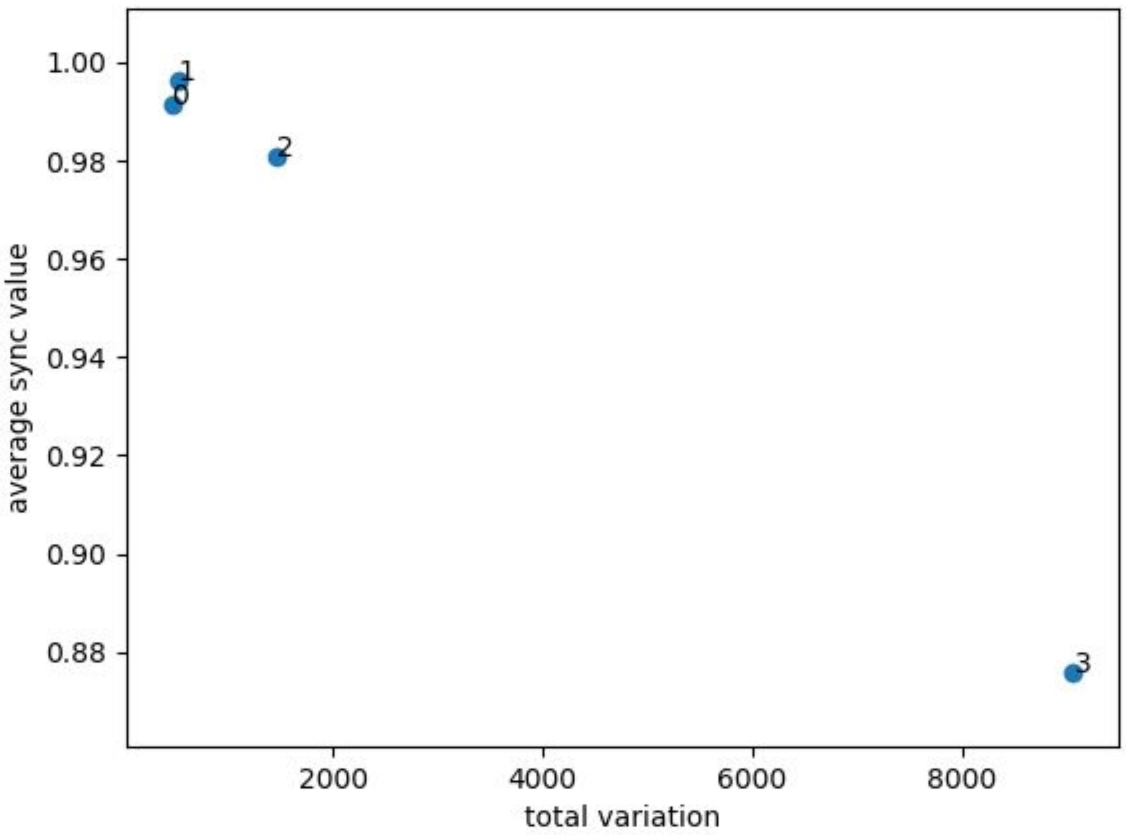




$\beta_0 = 1$	$\beta_0 = 1$	$\beta_0 = 1$	$\beta_0 = 2$
$\beta_1 = 1$	$\beta_1 = 1$	$\beta_1 = 0$	$\beta_1 = 0$



x
t
z
t
1.
2.
3.
synchronization
timestep

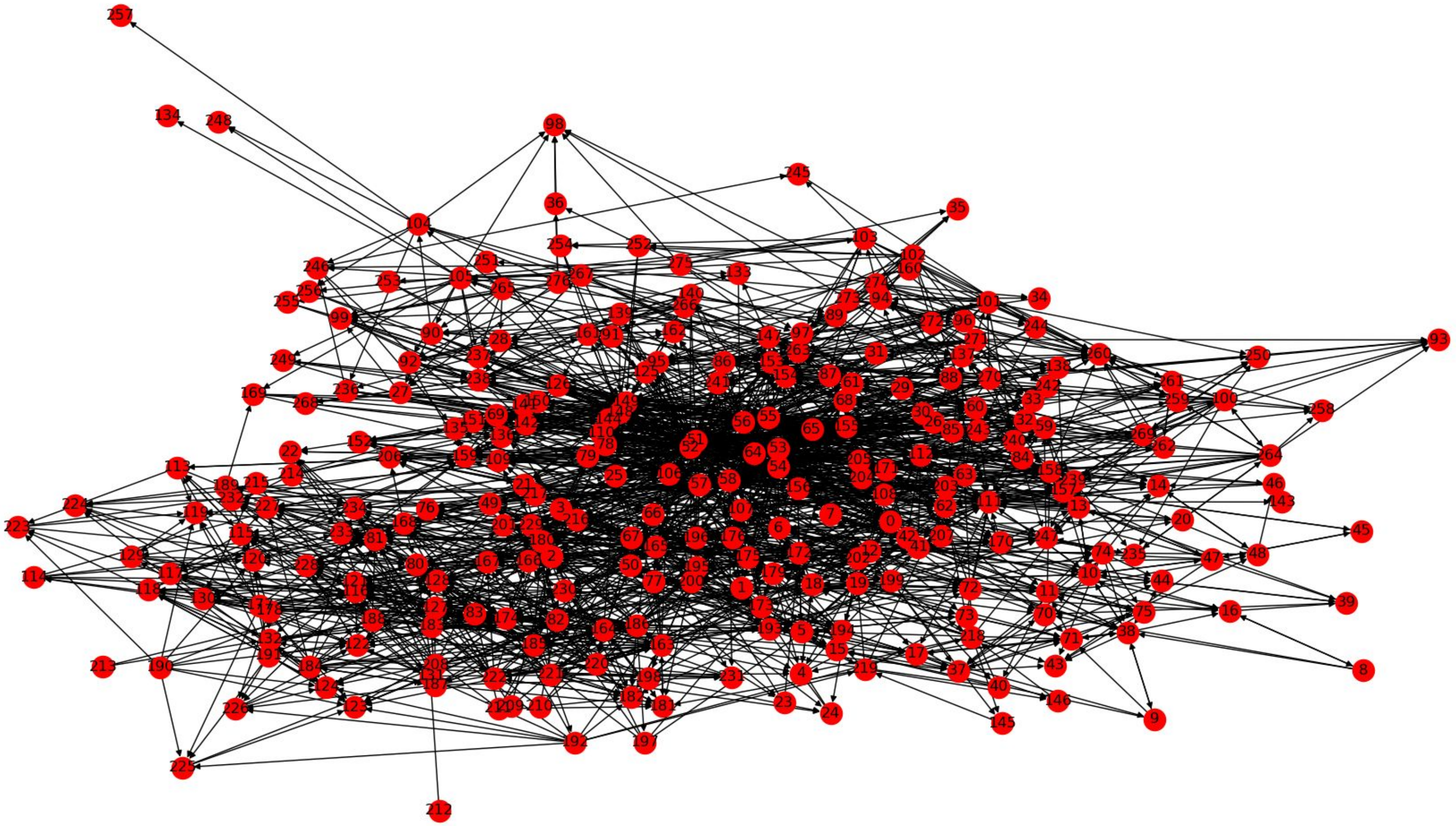


1.00
0.98
0.96
0.94
0.92
0.90
0.88
average sync value
1
0
2
3
2000
4000
6000
8000
total variation

Who does this help?

Computationally, running a neural network simulation is a very intense process.

If we could conclude that “homology matters” for any sized network, then reducing a network to its essential connections while preserving the homology would increase computability while exhibiting the same dynamic behavior.



Thank you!