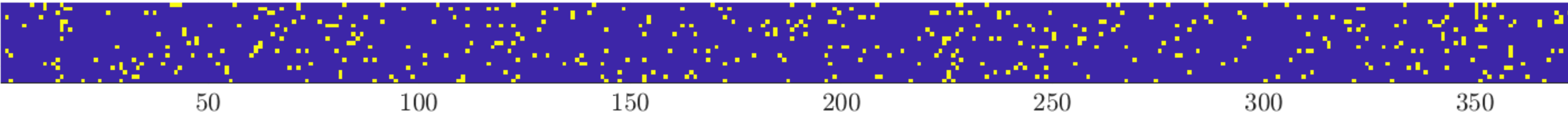
Transfer entropy for network reconstruction in a simple dynamical model

Roy Goodman
NJIT Dept. of Mathematical Sciences



How I Spent My Sabbatical

NYU | TANDON SCHOOL OF ENGINEERING

Dynamical systems laboratory

My location

My commute

My host: Mau Porfiri

$\mathcal{M}_{iN}^{\mathcal{E}}$

http://www.aimspress.com/journal/mine

Mathematics in Engineering, 2(1): 34–54.
DOI:10.3934/mine.2020003
Received: 12 April 2019
Accepted: 30 September 2019
Published: 31 October 2019

Research article

Topological features determining the error in the inference of networks using transfer entropy

Roy H. Goodman[1,*] and Maurizio Porfiri[2,*]

The Porfiri Lab

A big group working in a lot of areas, both theoretical and through laboratory experiments:

- Fluid mechanics: fluid structure interactions during water impact
- Artificial Muscles and Soft Robotics
- Telerehabilitation
- Network-based modeling of infectious diseases
- Fish schooling
- Using robotics and zebrafish to study substance-abuse disorders
- Information-theoretic analysis of social science datasets
- more...

A unifying theme in this work is using methods from information theory for modeling and analysis

What is this talk about?

A lot of words to define here:

Transfer entropy for network reconstruction in a simple dynamical model

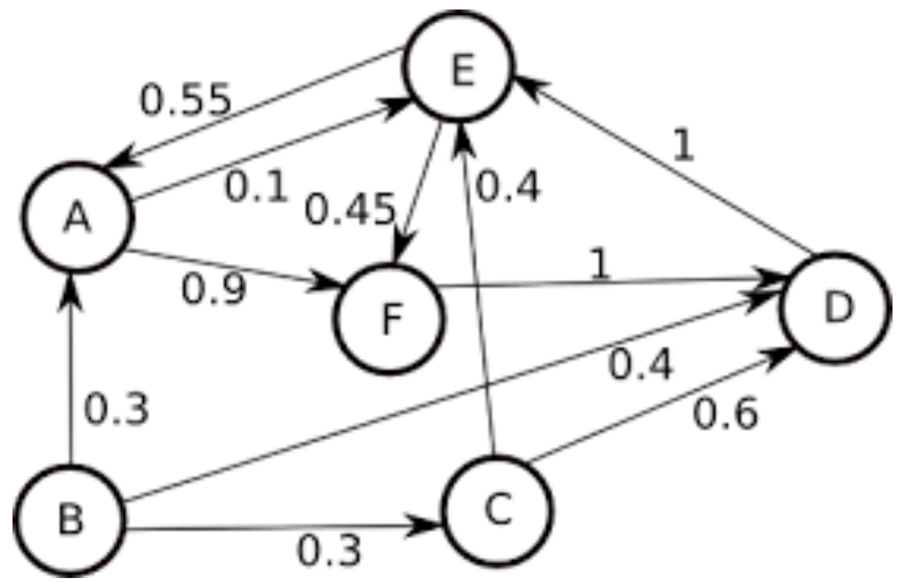
- Network: a *graph* composed of vertices and edges, the subject at the heart of *graph theory*
- Transfer entropy: a quantity describing the transfer of information from one evolving variable to another, from information theory
- The dynamical model: to be described, a simple probabilistic dynamics.
- Reconstruction: figure out properties of the graph based on the dynamics

18 months ago I knew ϵ about any of this

Graph Theory

Graph theory is central to the mathematics of Computer Science, describing the connections between interacting agents.

A graph is defined by a set $\mathcal{V}$ of vertices, connected by a set $\mathcal{E}$ of edges. The graph at right is both *directed* and *weighted*.



The weight matrix has entries W_{ij} defined as follows

- If an edge exists from node j to node i the entry is the associated weight
- If no edge exists, the entry zero.

$$W = \begin{bmatrix} 0 & 0.3 & 0 & 0 & 0.55 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 0 & 0 & 0 & 0 \\ 0 & 0.4 & 0.6 & 0 & 0 & 1 \\ 0.1 & 0 & 0.4 & 1 & 0 & 0 \\ 0.9 & 0 & 0 & 0 & 0.45 & 0 \end{bmatrix}$$

An important notion for us will be the *weighted incoming degree* $\delta_i = \sum_j W_{ij}$.

In this example $\delta_E = 1 + 0.4 + 0.1 = 1.5$

Information Theory

- Initiated by Claude Shannon's 1948 "A Mathematical Theory of Communication"
- Originally used to study the transmission of signals down noisy channels and to develop optimal strategies for encoding information
- Recently become popular tool for analyzing dynamical systems

Fundamental quantity:

Consider a discrete random variable X drawn from a sample space $\mathcal{X}$

The *information* associated with the event $X{=}x$ measures how "surprising" it is that $X{=}x$

$$I(x) = -\log\left(\Pr(X = x)\right)$$

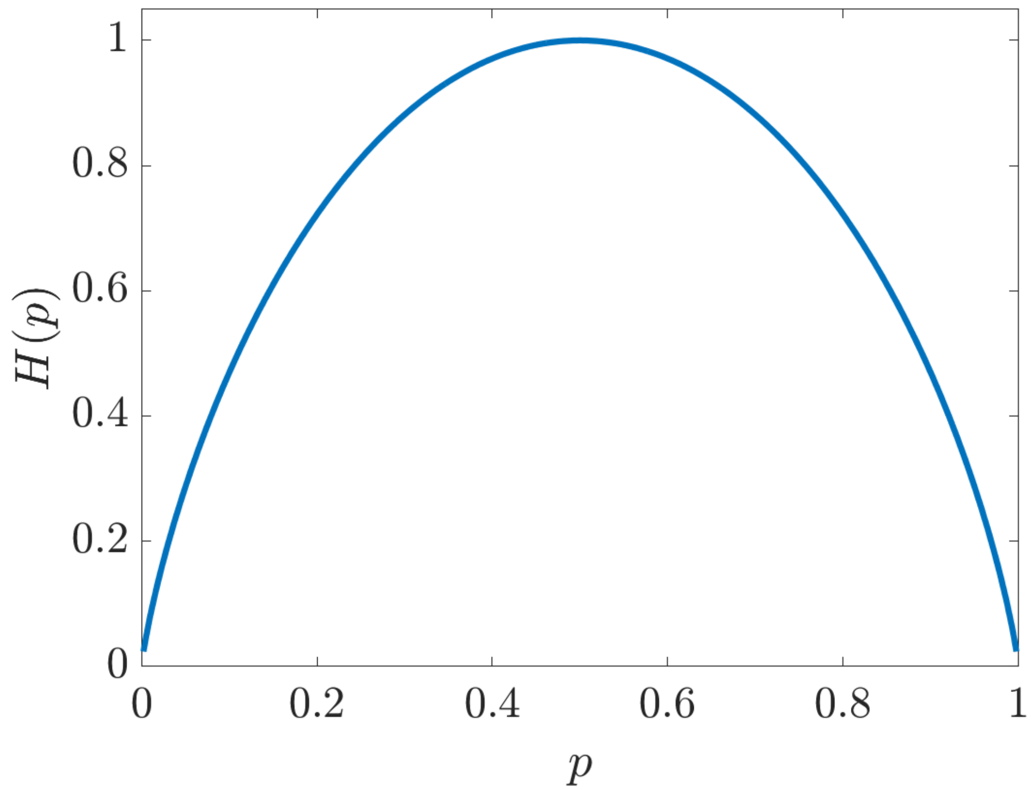
The *Shannon entropy* of the random variable is the expectation value fo the information

$$H(X) = E[I(X)] = -\sum_{x\in\mathcal{X}} \Pr(X = x) \log \Pr(X = x)$$

Basic example: biased coin toss

Consider a biased coin that gives heads with probability p and tails with probability 1−p

$$H(p) = -p \log p - (1-p) \log (1-p)$$



- When p≈0 or p≈1, entropy small since surprising outcomes rarely occur
- When p≈0.5, entropy large since both outcomes equally likely

Transfer entropy Schreiber 2000

Consider two random variables X and Y.

Define the *joint entropy* and the *conditional entropy:*

$$H(X,Y) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} \Pr(X=x,Y=y)\log\Pr(X=x,Y=y)$$

$$H(X|Y) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} \Pr(X=x,Y=y)\log\Pr(X=x|Y=y)$$

Entropy can be defined analogously for stationary stochastic processes

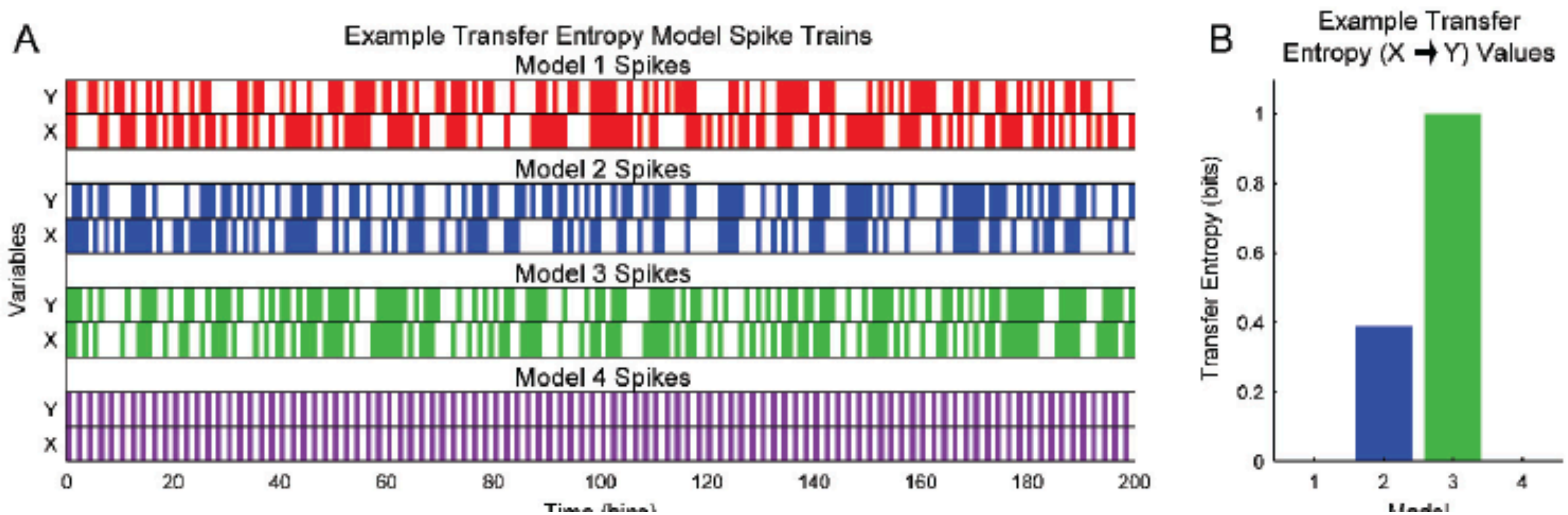
Transfer entropy from Y to X is the difference between the entropy of X(t+1) conditioned on X(t) and that conditioned on both X(t) and Y(t)

$$\begin{aligned}\mathrm{TE}^{Y\to X} &= H(X(t+1)|X(t)) - H(X(t+1)|X(t),Y(t)) \\ &= \sum_{\substack{x_+\in\mathcal{X}\\ x\in\mathcal{X}\\ y\in\mathcal{Y}}} \left\{\Pr\left[X(t+1)=x_+,X(t)=x,Y(t)=y\right]\times\log\frac{\Pr\left[X(t+1)=x_+|X(t)=x,Y(t)=y\right]}{\Pr\left[X(t+1)=x_+|X(t)=x\right]}\right\}\end{aligned}$$

Transfer entropy measures the *reduction in the uncertainty* of predicting X(t + 1) from both X(t) and Y(t) relative to predicting it from X(t) alone.

Contrived transfer entropy example

Source: A Tutorial for Information Theory in Neuroscience
Nicholas M. Timme and Christopher Lapish 2018

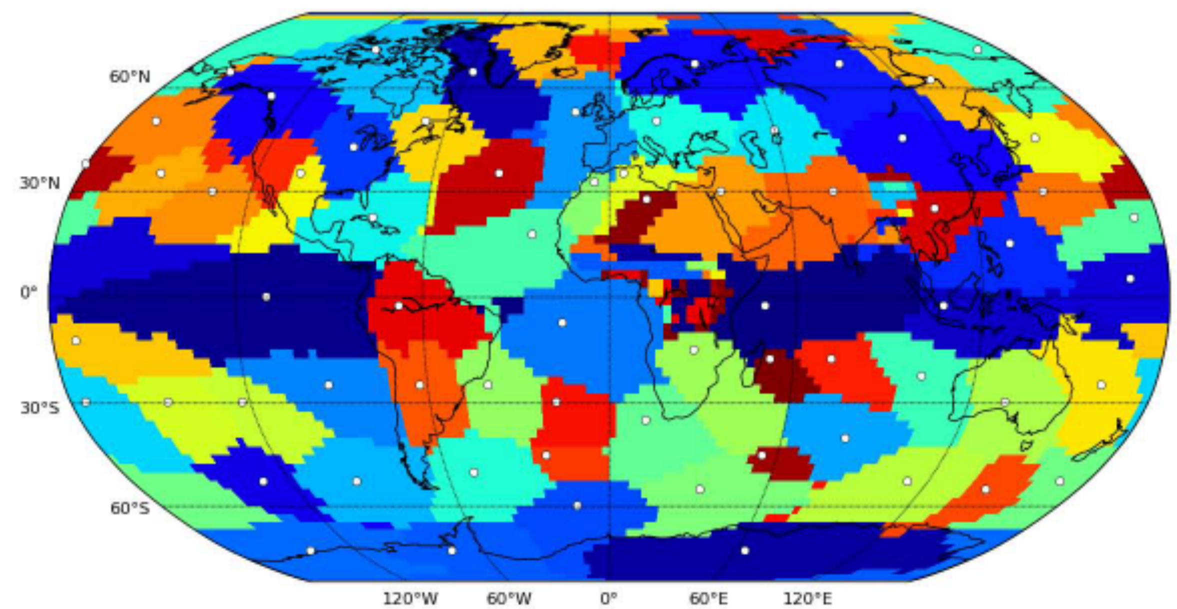


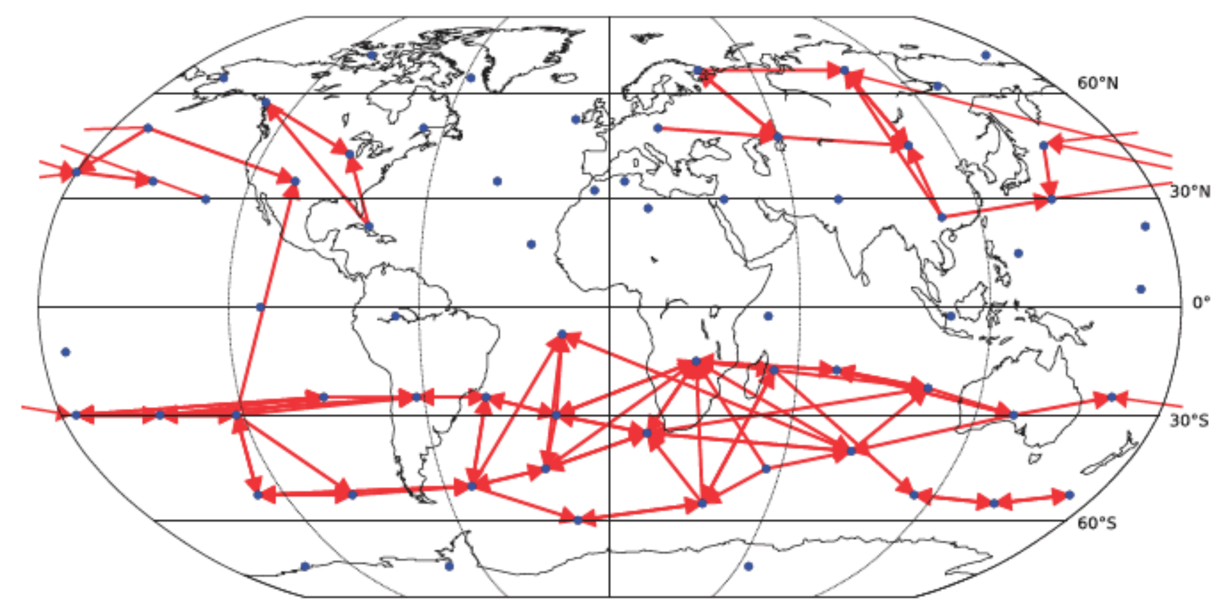
- Series 1: X & Y uncorrelated: TE≈0
- Series 2 & 3: X tends to fire before Y: TE large
- Series 4: $X(t+1)$ determined entirely from $X(t)$, no improvement from knowing $Y(t)$: TE=0

Transfer entropy used to infer climate network

Hlinka et al. 2013

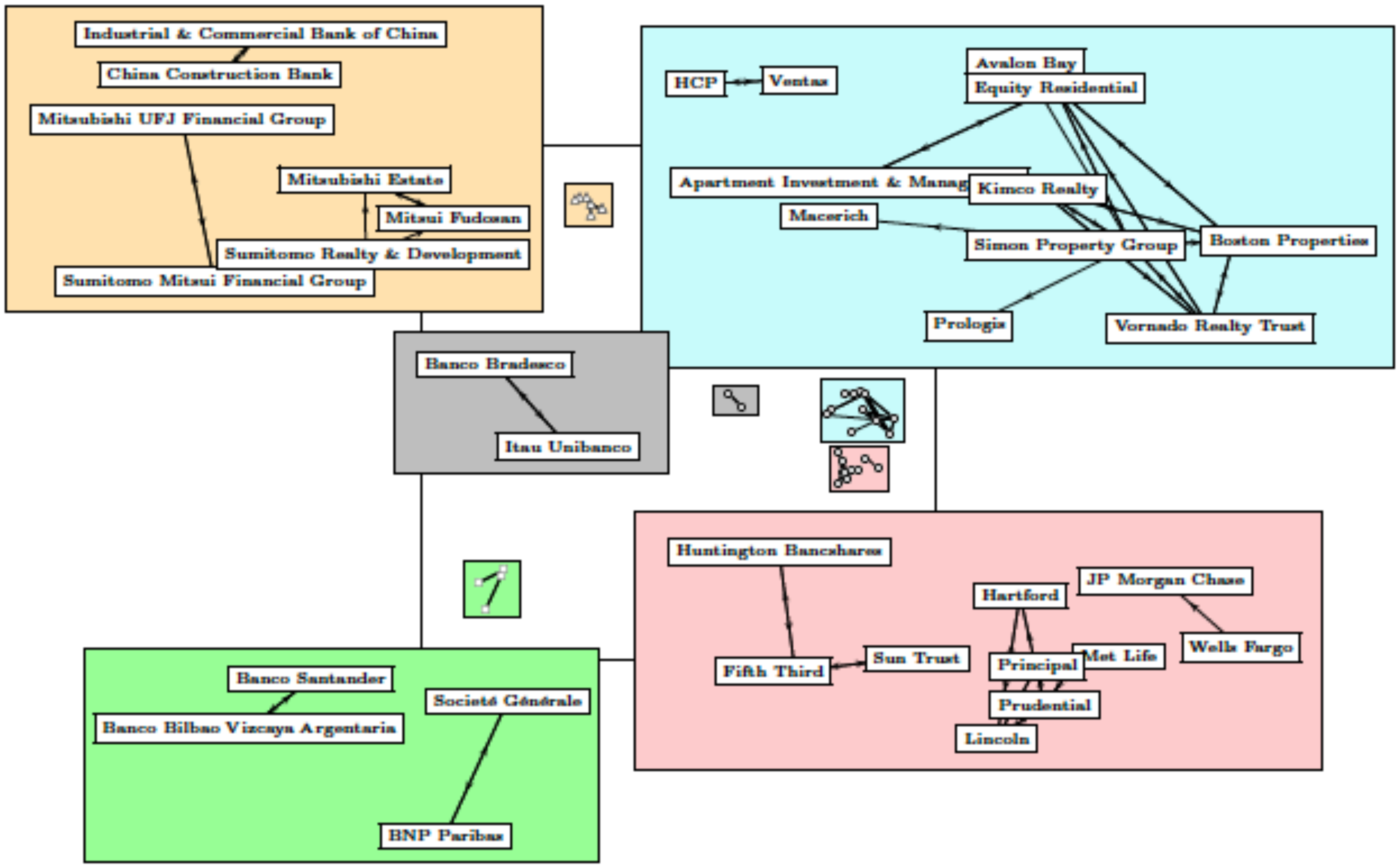
- Divide the earth into N patches using principal component analysis
- Look at time series of surface-area temperature deviations
- Compute transfer entropies, estimating PDFs using specially-tuned kernel density estimators
- Threshold to find links with non-negligible transfer entropy





Transfer Entropy used to analyze connections between corporations

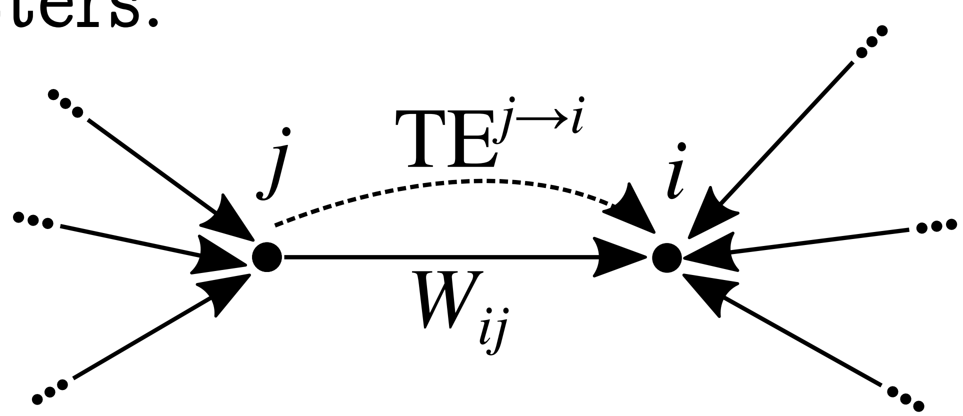
Sandoval 2014



Analysis based on time series of stock prices. Transfer entropy predicts that the price of a stock has an influence on other stocks in its sector and geographic area

Advantages and disadvantages of Transfer Entropy

- Model free
- Nonlinear
- Relatively simple to compute
- Requires the estimation of underlying probability distribution, e.g. by binning or kernel density estimation
- Lots of hidden parameters to fiddle with that might effect the computation
- Inherently dyadic: quantifies the interaction between two agents while ignoring the effects of others. Unclear how much this matters.



Random Boolean Network (RBN) Model

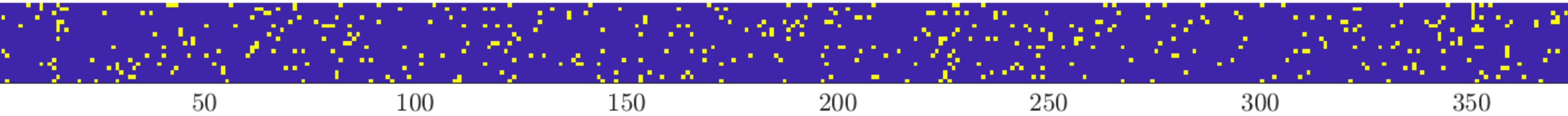
Porfiri & Ruiz Marín 2018

A system of nodes $X^i(t)$, i=1...n, each of which can take values in $\{0,1\}$

At time step t+1, the state of node i depends on the state of the system at time step t, according to

$$\Pr\left[X^i(t+1) = 1 | X^1(t) = x^1, \ldots, X^N(t) = x^N\right] = \epsilon \left[1 + \sum_{j=1}^{N} W_{ij} x^j\right]$$

Conceived as a model "Policy Diffusion"—How the passage of laws in one jurisdiction influences the passage of laws in other jurisdictions. The terms W_{ij}≥0 represent the "network of influence."



Sample time series

Porfiri & Ruiz Marín's result (2018)

Setting $\epsilon \ll 1$ allows the use of perturbation methods to approximately calculate Transfer Entropy in terms of the weights

Inverting gives an approximate formula for W_{ij} in terms of transfer entropy

$$\mathrm{TE}^{j \to i} = \epsilon^2 G^{(2)}(W_{ij}) + \mathcal{O}(\epsilon^3)$$

where

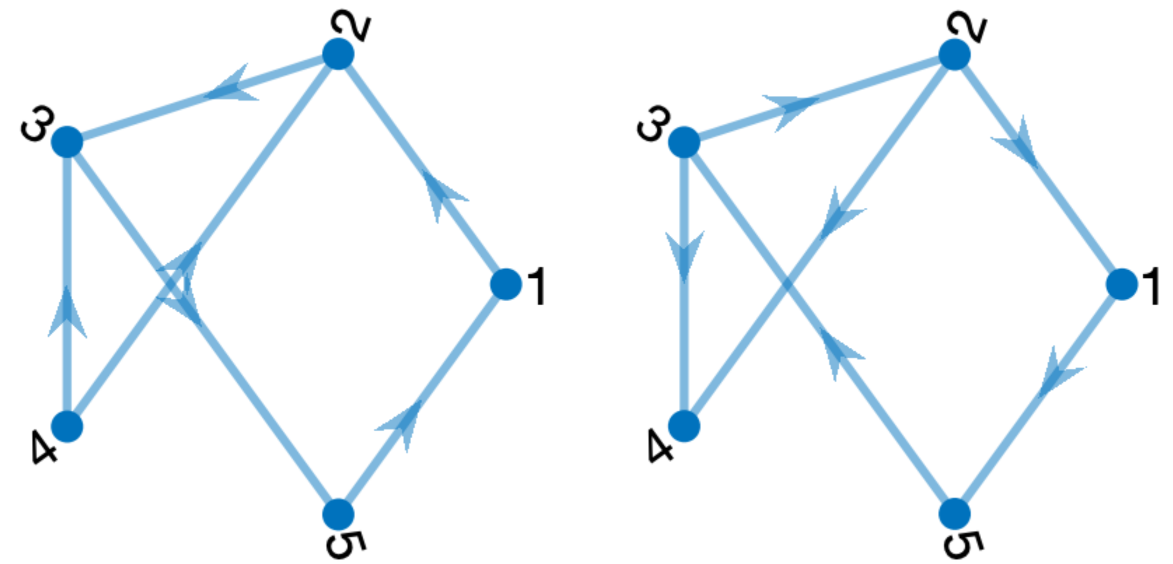
$$G^{(2)}(x) = -x + (1+x)\log(1+x) \approx \frac{1}{2}x^2 \text{ for } x \ll 1$$

Takeaway: To leading order transfer entropy from j to i depends on the strength of the weight from j to i

Obvious next question: By calculating next term in the expansion, can we quantify the effect the global topology of the network has on the computed transfer entropy?

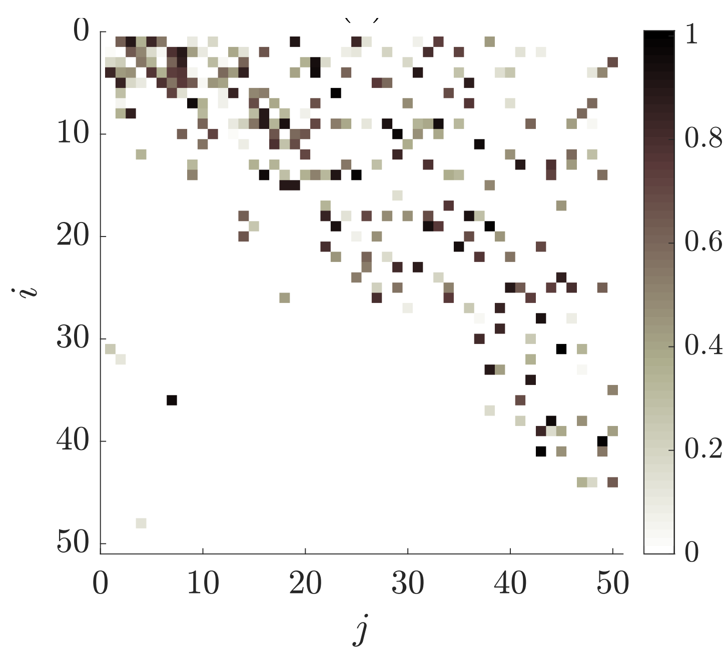
An example to show that next-order terms matter

The transpose $\Gamma^\top$ of a directed graph Γ has the same vertices, oppositely-directed edges, and a weight matrix $W^\top$



Two networks Γ and $\Gamma^\top$ that behave differently

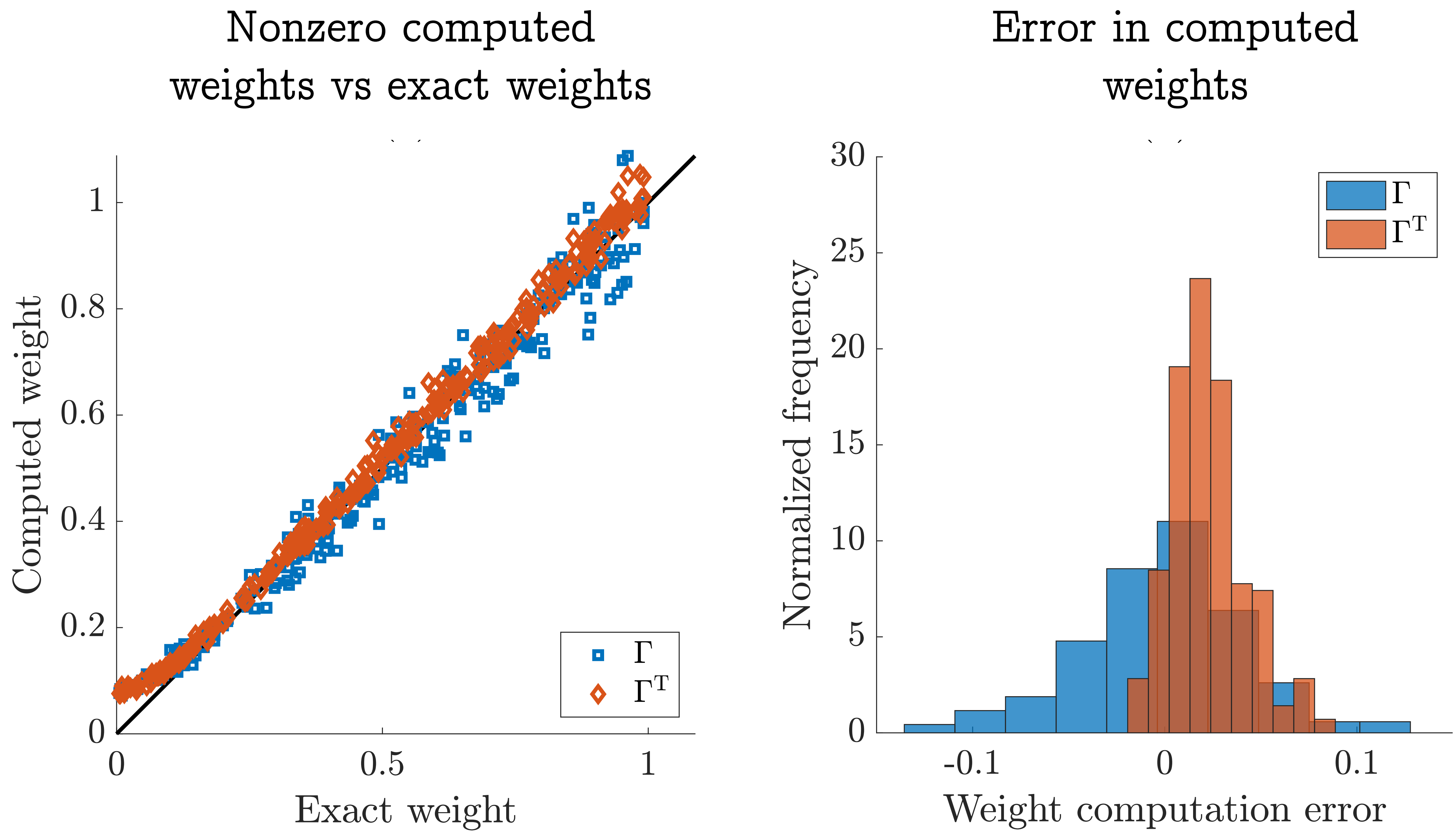
A 50-vertex, 282-edge directed Barabási-Albert network with weight matrix W, random weights



Procedure:

- Compute dynamics for 10^5 steps
- Repeat 100 times
- Compute TE from time series
- Estimate W_{ij} from formula

Result: Distribution of error much wider for Γ than $\Gamma^{\top}$



Strategy for analyzing the dynamics

- Recast the system as a Markov chain, dependent on a small parameter ϵ
- Calculate the stationary vector of the Markov chain via perturbation theory in ϵ
- Use the stationary probability vector and the transition law to derive a formula for transfer entropy in terms of the weights W_{ij}
- Invert to get approximation for the weights in terms of the pairwise transfer entropy
- Apply to numerically-generated time series of the model

To analyze: recast as a Markov Chain

So, a quick review

Consider a discrete-time finite-state Markov chain $Z(t)$, i.e.

- $Z(t), t \in \mathbb{N}$, takes values $z_1, \ldots, z_M$ in a space $\mathcal{Z}$ of cardinality M
- $\nu(t) \in \mathbb{R}_+^M$ is the probability vector $\nu_i(t) = \Pr[Z(t) = z_i]$
- Transition matrix P with entries

$$P_{ij} = \Pr[Z(t+1) = z_j | Z(t) = z_i]$$

- Then ν evolves according to

$$\nu(t+1) = \nu(t)P$$

- The long term behavior is $\nu(t) \xrightarrow[t\to\infty]{} \pi$, where the stationary vector π satisfies

$$\pi = \pi P$$

under mild assumptions on P

Recasting as a Markov chain: Main Idea

The states are binary vectors

$$\left\{ Z_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, Z_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, Z_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, Z_4 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, Z_{2^N} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \right\}$$

The state vector is a 2^N-dimensional probability vector ν with components

$$\nu_i(t) = \Pr[X(t) = Z_i]$$

The transition law is determined from the dynamics

Recasting as a Markov Chain: Ugly Details

Our RBN Model:

$$\Pr\left[X^i(t+1) = 1 | X^1(t) = x^1, \ldots, X^N(t) = x^N\right] = \epsilon\left[1 + \sum_{j=1}^{N} W_{ij} x^j\right]$$

Letting z=[x^1,...,x^N] this becomes

$$\Pr\left[X^i(t+1) = 1 | X(t) = z\right] = \epsilon\left[1 + e_i^\top W z\right]$$

Allowing $x^i{}_+ \in \{0,1\}$

$$\Pr\left[X^i(t+1) = \underbrace{x^i_+}_{\in\{0,1\}} | Z(t) = z\right] = \underbrace{\left(1 - x^i_+\right)}_{=\begin{cases} 1, & x^i_+ = 0 \\ 0, & x^i_+ = 1 \end{cases}} + \epsilon \underbrace{\left(2x^i_+ - 1\right)}_{=\begin{cases} -1, & x^i_+ = 0 \\ 1, & x^i_+ = 1 \end{cases}} \left[1 + e_i^\top W z\right]$$

Taking a product of such terms yields the Markov transition matrix

$$P_{ij}(t) = \Pr\left[Z(t+1) = z_j | Z(t) = z_i\right] = \prod_{k=1}^{N} \left\{\left(1 - e_k^\top z_j\right) + \epsilon\left(2e_k^\top z_j - 1\right)\left[1 + e_k^\top W z_i\right]\right\}$$

$$= P^{(0)} + \epsilon P^{(1)} + \epsilon^2 P^{(2)} + \mathcal{O}(\epsilon^3)$$

Setting up perturbation calculation

Expand $\boldsymbol{\pi}=\boldsymbol{\pi}^{(0)}+\boldsymbol{\epsilon}\boldsymbol{\pi}^{(0)}+\boldsymbol{\epsilon}^2\boldsymbol{\pi}^{(2)}+\ldots$

$$\left(\pi^{(0)}+\epsilon\pi^{(1)}+\epsilon^2\pi^{(2)}\right)\left(P^{(0)}+\epsilon P^{(1)}+\epsilon^2 P^{(2)}\right)=\pi^{(0)}+\epsilon\pi^{(1)}+\epsilon^2\pi^{(2)}+\ldots$$

Separating by orders yields

$$\mathcal{O}(1): \quad \pi^{(0)}\left(I-P^{(0)}\right)=0 \qquad \sum_{j=1}^{N}\pi_j^{(0)}=1,$$

$$\mathcal{O}(\epsilon): \quad \pi^{(1)}\left(I-P^{(0)}\right)=\pi^{(0)}P^{(1)}, \qquad \sum_{j=1}^{N}\pi_j^{(1)}=0$$

$$\mathcal{O}(\epsilon^2): \quad \pi^{(2)}\left(I-P^{(0)}\right)=\pi^{(0)}P^{(2)}+\pi^{(1)}P^{(1)} \qquad \sum_{j=1}^{N}\pi_j^{(2)}=0$$

Actually solving this was really hard...

Solving the perturbation series I

The matrices $P^{(j)}$ are $2^N \times 2^N$ so we can only write them down explicitly for small N, need to work "in the abstract," hybrid pencil&paper/Mathematica workflow

$$P^{(0)}_{ij} = \prod_{k=1}^{N} \left[\left(1 - e_k^\top z_j\right)\right] = \begin{cases} 1, & \|z_j\| = 0, \\ 0, & \|z_j\| > 0. \end{cases} \qquad \text{where } \|Z\| = \textstyle\sum_{k=1}^{N} x_k$$

$$P^{(1)}_{ij} = \sum_{r=1}^{N} \left\{ \left(2e_r^\top z_j - 1\right)\left[1 + e_r^\top W z_i\right] \prod_{\substack{k=1 \\ k \neq r}}^{N} \left(1 - e_k^\top z_j\right) \right\} = \begin{cases} -\left[N + 1_N^\top W z_i\right], & \|z_j\| = 0 \\ \left[1 + z_j^\top W z_i\right], & \|z_j\| = 1 \\ 0, & \|z_j\| > 1 \end{cases}$$

$$P^{(2)}_{ij} = \sum_{\substack{r,s=1 \\ r>s}}^{N} \left\{ \left(2e_r^\top z_j - 1\right)\left[1 + e_r^\top W z_i\right]\left(2e_s^\top z_j - 1\right)\left[1 + e_s^\top W z_i\right] \prod_{\substack{k=1 \\ k \neq r,s}}^{N} \left(1 - e_k^\top z_j\right) \right\}$$

$$= \begin{cases} \sum_{\substack{r,s=1 \\ r>s}}^{N} \left\{\left[1 + e_r^\top W z_i\right]\left[1 + e_s^\top W z_i\right]\right\}, & \|z_j\| = 0, \\ -\left[1 + z_j^\top W z_i\right]\left[N - 1 + \left(1_N^\top - z_j^\top\right) W z_i\right], & \|z_j\| = 1, \\ \left\{1 + z_j^\top W z_i + \left[e_{\mathcal{I}_1(z_j)}^\top W z_i\right]\left[e_{\mathcal{I}_2(z_j)}^\top W z_i\right]\right\}, & \|z_j\| = 2. \\ 0, & \|z_j\| > 2, \end{cases}$$

Solving the Perturbation Series II

Solve for the stationary vector order by order

$$\pi_i^{(0)} = \begin{cases} 1, & \|z_i\| = 0, \\ 0, & \|z_i\| > 0, \end{cases}$$

$$\pi_i^{(1)} = \begin{cases} -N, & \|z_i\| = 0, \\ 1, & \|z_i\| = 1, \\ 0, & \|z_i\| > 1, \end{cases}$$

$$\pi_i^{(2)} = \begin{cases} \binom{N}{2} - 1_N^\top W 1_N, & \|z_i\| = 0, \\ -(N-1) + z_i^\top W 1_N, & \|z_i\| = 1, \\ 1, & \|z_i\| = 2, \\ 0, & \|z_i\| > 2, \end{cases}$$

Calculating Transfer Entropy

from node 2 to node 1

$$\mathrm{TE}^{2\to 1} = \sum_{x^1_+, x^1, x^2} \Pr\left[X^1_{(t+1)} = x^1_+, X^1_t = x^1, X^2_t = x^2\right] \log \frac{\Pr\left[X^1_{(t+1)} = x^1_+ | X^1_t = x^1, X^2_t = x^2\right]}{\Pr\left[X^1_{(t+1)} = x^1_+ | X^1_t = x^1\right]}$$

- Use the transition rule and the asymptotic expansions to approximate the various probabilities and conditional probabilities
- Use some tricks to avoid dividing by small numbers
- Get terms like

$$\Pr\left[X^1_{t+1} = x^1_+ | X^1_t = x^1, X^2(t) = x^2\right] =$$

$$(1 - x^1_+) + (2x^1_+ - 1)\left(\epsilon\left[1 + W_{11}x^1 + W_{12}x^2\right] + \epsilon^2 \sum_{j=3}^{N} W_{1j}\right) + \mathcal{O}(\epsilon^3)$$

The next-order correction

Finally, we arrive at

$$\mathrm{TE}^{j\to i} = \epsilon^2 G^{(2)}(W_{ij}) + \epsilon^3 G^{(3)}_{ij}(W) + \mathcal{O}(\epsilon^4)$$

where

$$G^{(2)}(x) = -x + (1+x)\log(1+x)$$

$$G^{(3)}_{ij}(W) = W_{ij}\left(W_{ij} - d_i - d_j\right) + \log\left(1 + W_{ij}\right)\left(d_i - W_{ij} + (1+W_{ij})d_j\right)$$

and d_j is the weighted in-degree of node j

$$d_j = \sum_{k=1}^{N} W_{jk}.$$

Solving for the weights: one more perturbation expansion

Letting

$$T_{ij} \equiv \frac{\mathrm{TE}^{j\to i}}{\epsilon^2} = G^{(2)}(W_{ij}) + \epsilon G^{(3)}_{ij}(W) + \mathcal{O}(\epsilon^2)$$

and

$$W = W^{(0)} + \epsilon W^{(1)} + \mathcal{O}(\epsilon^2)$$

we arrive at desired formula

$$W^{(0)}_{ij} = \left[G^{(2)}\right]^{-1}(T_{ij})$$

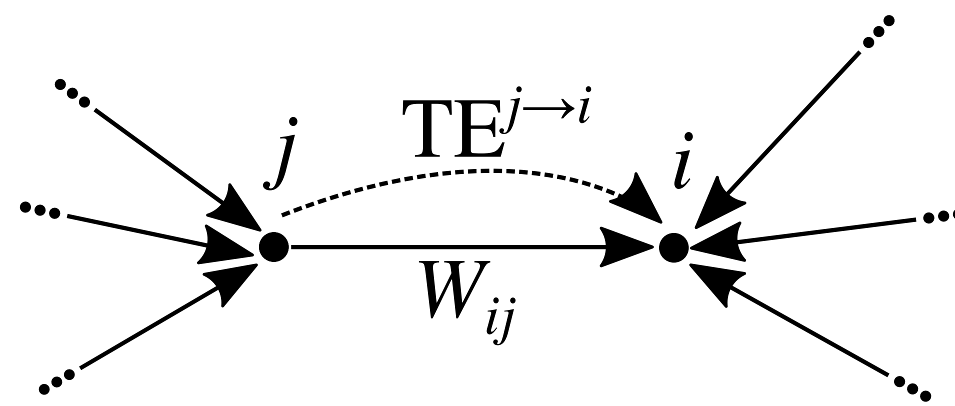
$$W^{(1)}_{ij} = -\frac{G^{(3)}_{ij}(W^{(0)})}{\frac{d}{dw}G^{(2)}(w)\Big|_{w=W^{(0)}_{ij}}}$$

Interpreting the correction term

If $W_{ij} \ll 1$ then the correction to the computed transfer entropy is

$$G^{(3)}_{ij}(W) \sim \frac{W_{ij}^2}{2}\left(d_j - d_i + W_{ij}\right) + \mathcal{O}(\|W\|^3)$$

The difference in weighted in-degree of the two nodes

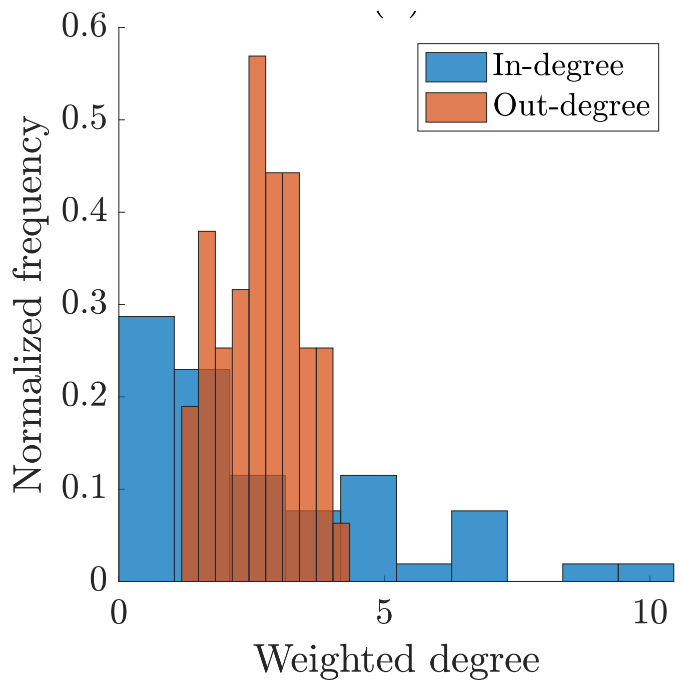


This yields a correction to the computed weight

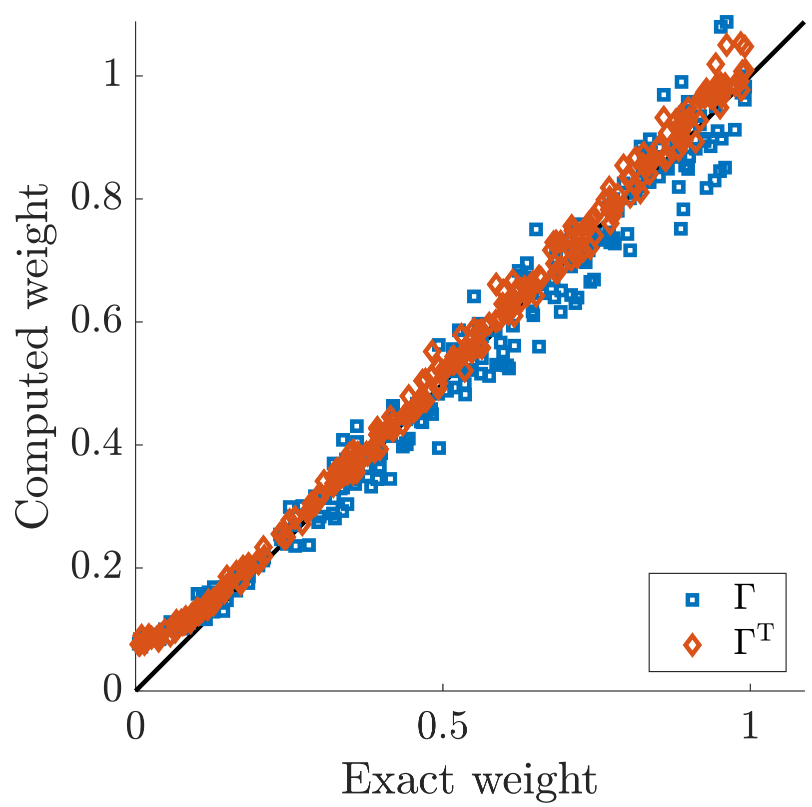
$$W^{(1)}_{ij} \sim \frac{W^{(0)}_{ij}}{2}\left(d^{(0)}_j - d^{(0)}_i + W^{(0)}_{ij}\right) + \mathcal{O}\left(\left\|W^{(0)}\right\|^2\right)$$

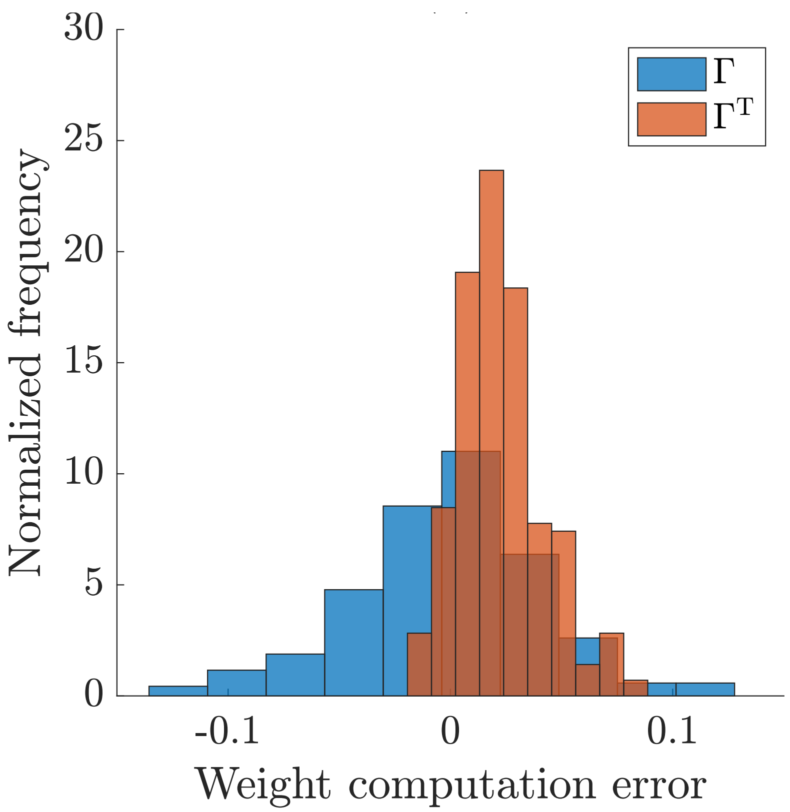
Return to numerical example

The network Γ was constructed so that its in-degrees vary more widely than its out-degrees



This leads to a larger variance in the computed weights for Γ than for $\Gamma^\top$

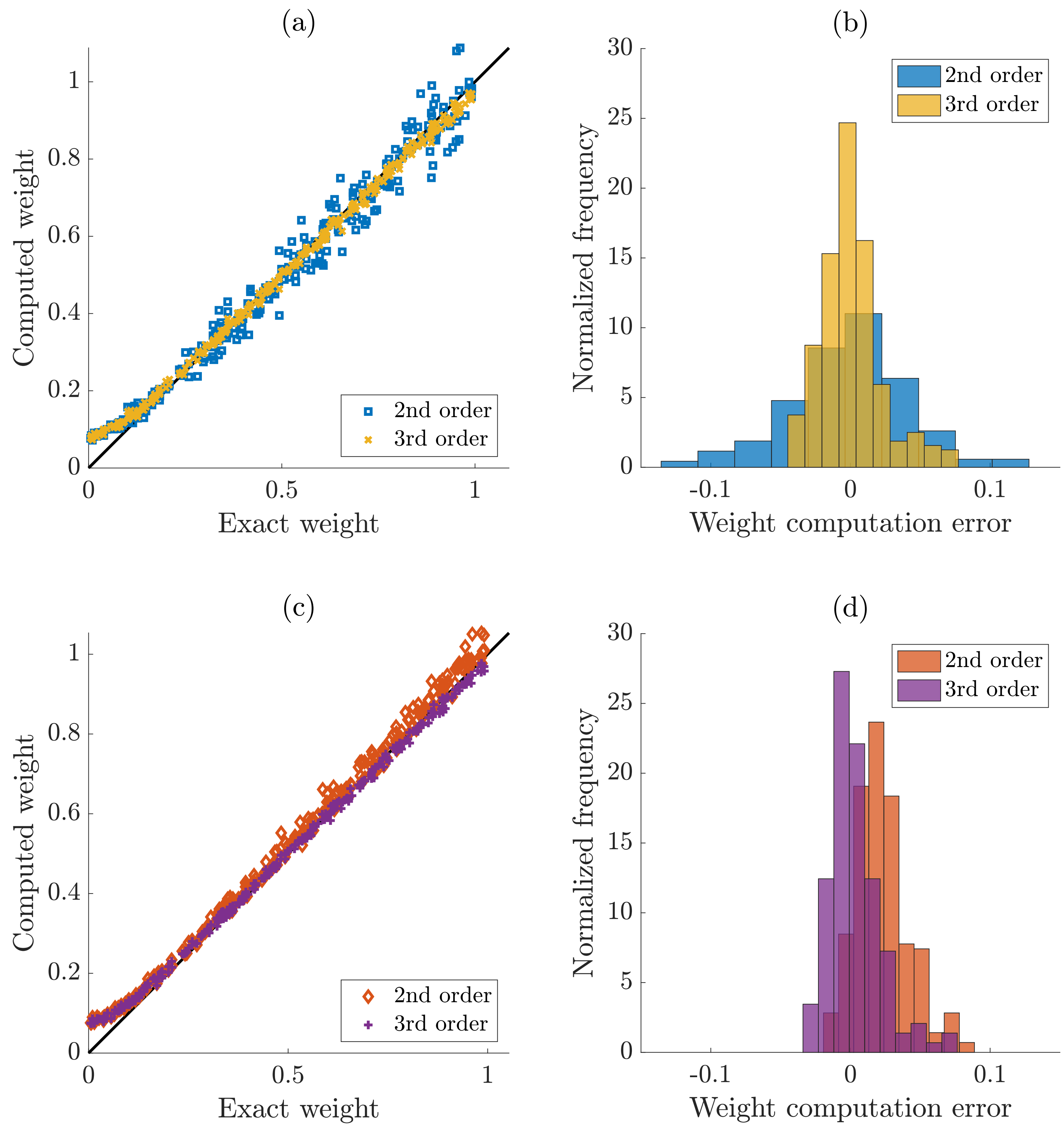




Putting in the correction

Γ

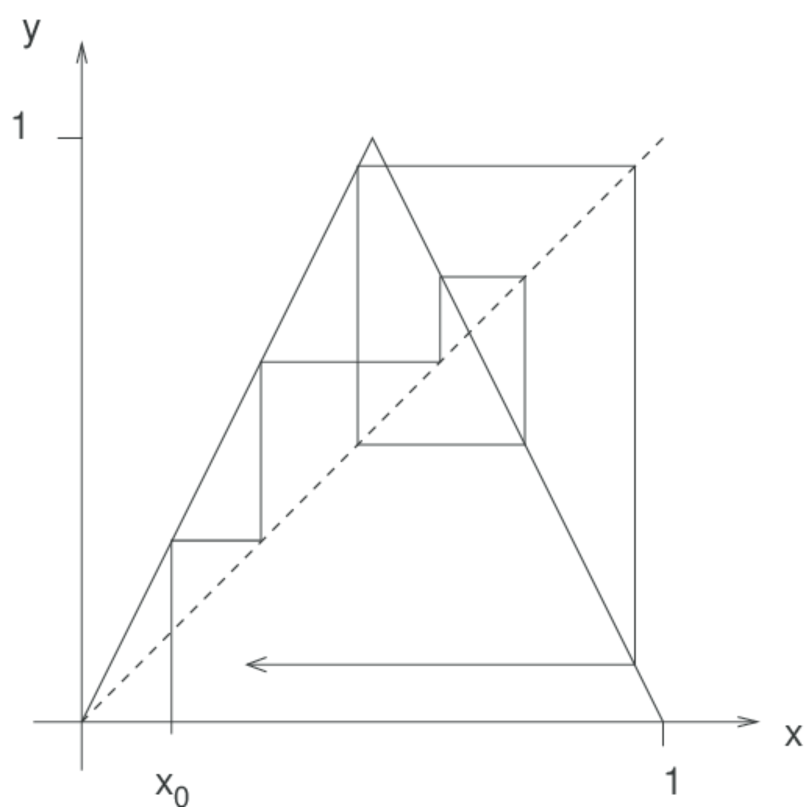
$\Gamma^\top$



Question: Is this a general phenomenon?

Does this long and painful calculation actually tell us anything about connection between the network structure and the accuracy of the computation?

Let's look at another model.



The *tent map* is a simple chaotic system

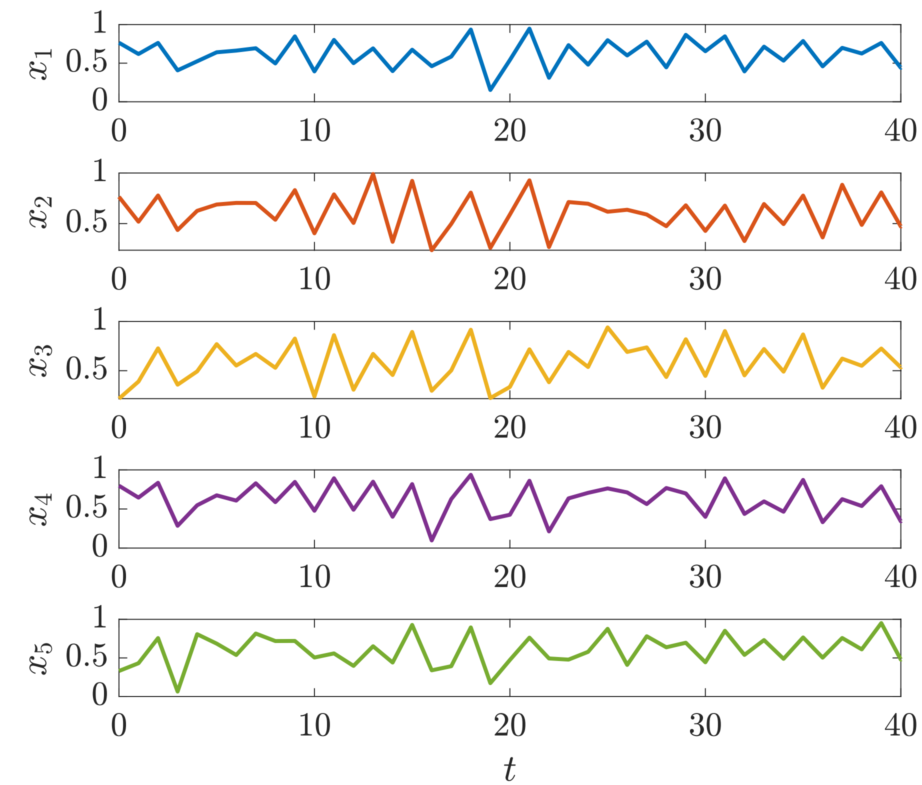
$$x_{t+1} = F(x_t) \equiv \begin{cases} 2x_t, & 0 \leq x_t < \frac{1}{2} \\ 2(1 - x_t) & \frac{1}{2} \leq x_t \leq 1 \end{cases}$$

We consider a system of coupled tent maps with noise

$$x^i_{t+1} = F\left(x^i_t\right) + \epsilon \sum_{j=1}^{N} W_{ij} \left(F\left(x^j_t\right) - F\left(x^i_t\right)\right) + \sigma n^i(t)$$

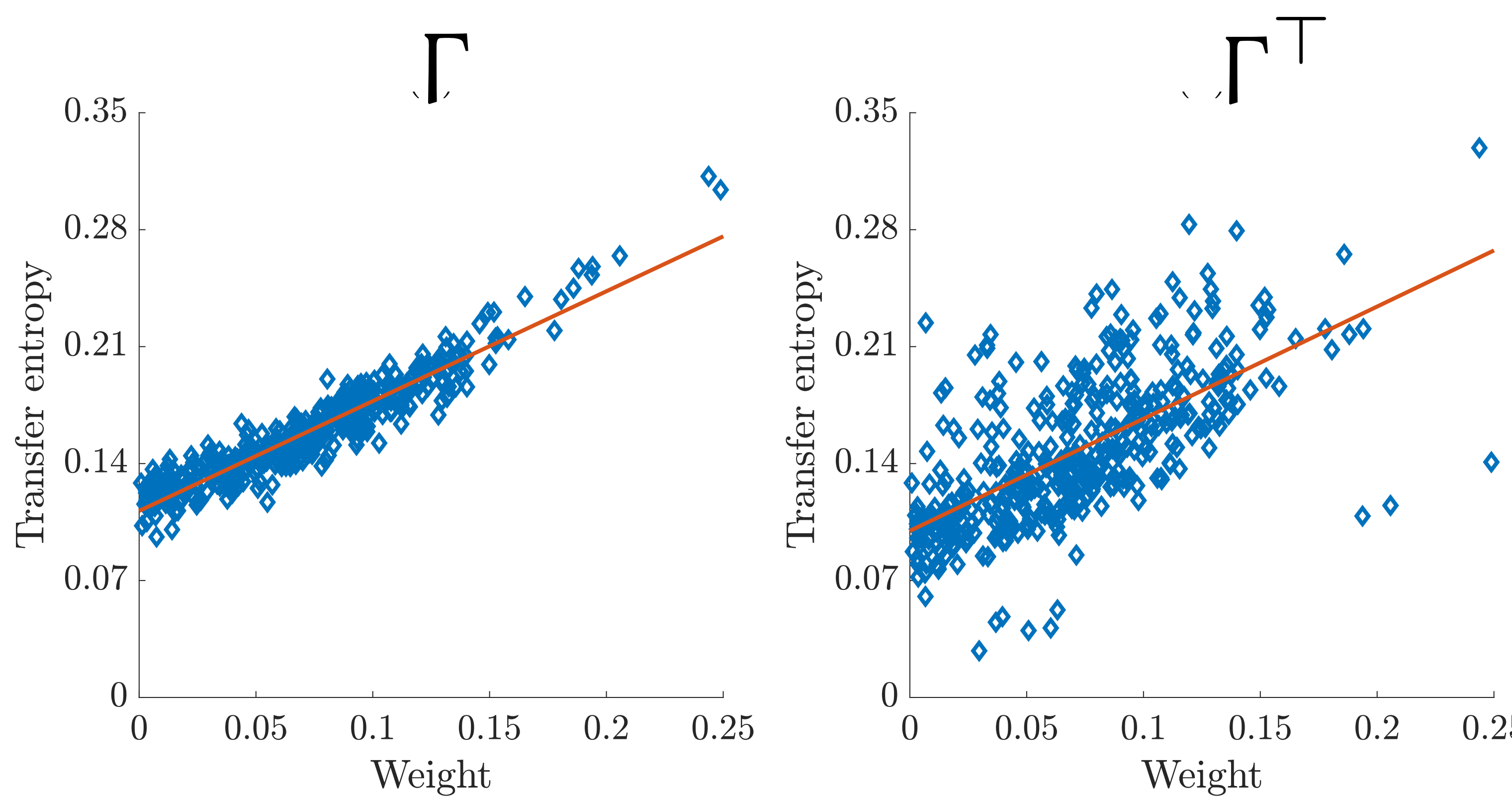
The coupled tent map example

Simulate for two 30-node networks Γ and $\Gamma^\top$ and observe oscillators going in and out of synchronization



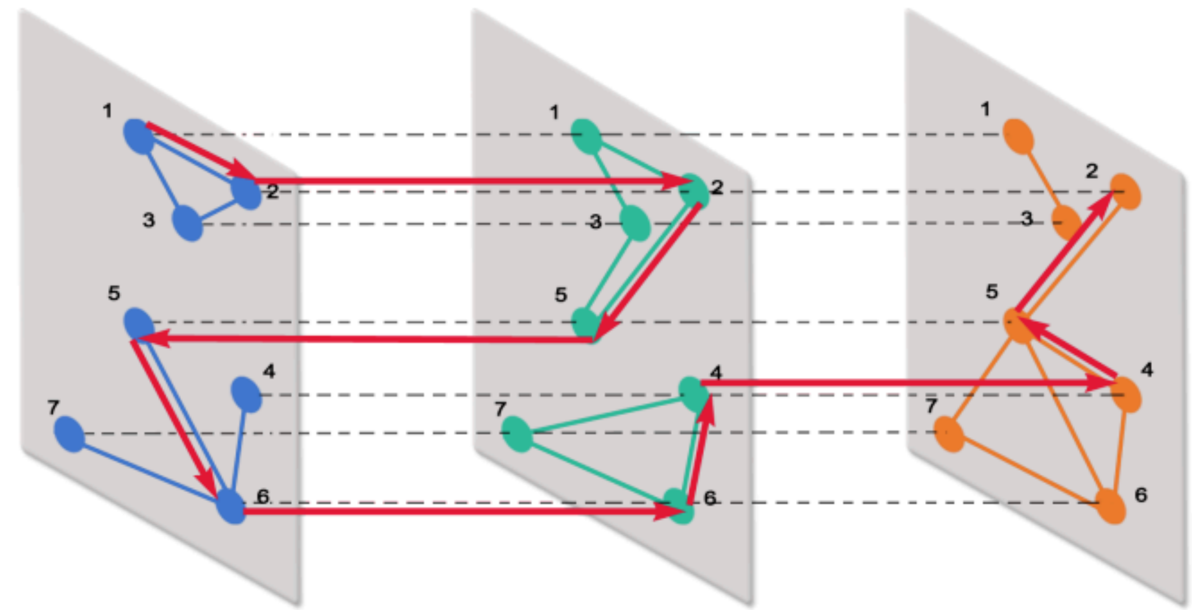
The networks constructed such that their distribution of weighted in-degrees very different

Variance in computed transfer entropy increases with variance in weighted in-degree



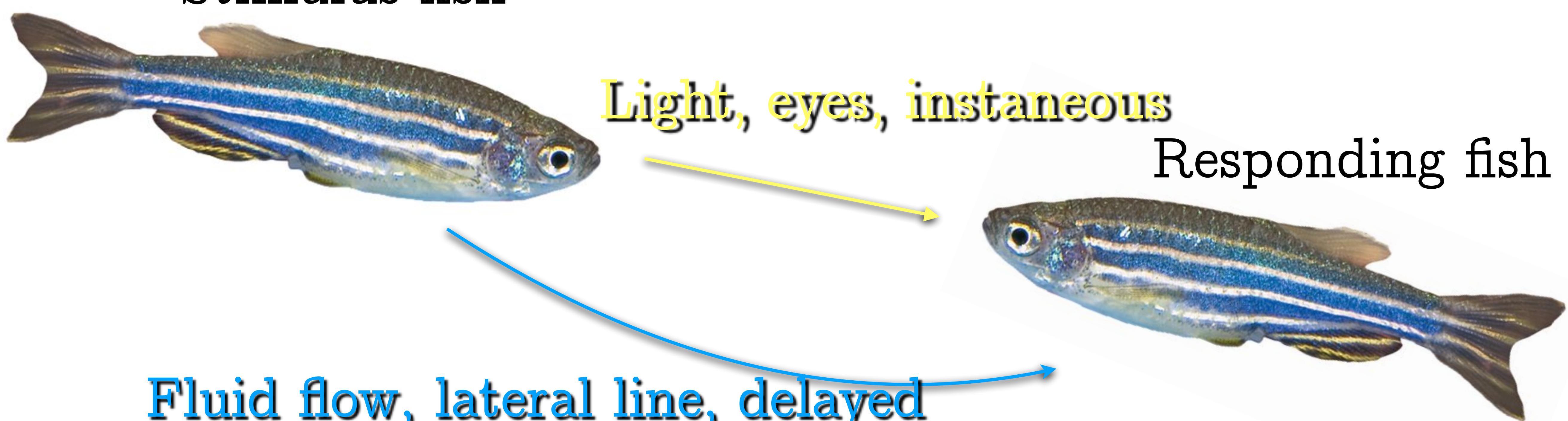
Extending the model: a multi-layered graph

There are many generalizations of the concept of a graph, e.g. multigraph in which there exist different types of edges. organized in layers



Maurizio's interest: Fish communicate over multiple channels

Stimulus fish



Multilayered delay RBN model

$$\Pr\left(x_i(t)=1\right)=\epsilon\left(1+\sum_{m=1}^{M}\sum_{j=1}^{N}W_{ij}^{(m)}x_j(t-m)\right)$$

Rewrite as a suspended system with coordinate

$$y(t)=\begin{pmatrix} x(t) \\ x(t-1) \\ \vdots \\ x(t+1-M) \end{pmatrix}=\begin{pmatrix} y_{(1)}(t) \\ y_{(2)}(t) \\ \vdots \\ y_{(M)}(t) \end{pmatrix}\in\{0,1\}^{MN}$$

The first N coordinates are updated probabilistically

$$\Pr\left(y_i(t)=1\right)=\epsilon\left(1+\sum_{j=1}^{MN}W_{ij}y_j(t-1)\right),\ \text{where } W=\left[W^{(1)}\,W^{(2)}\ \cdots\ W^{(M)}\right]$$

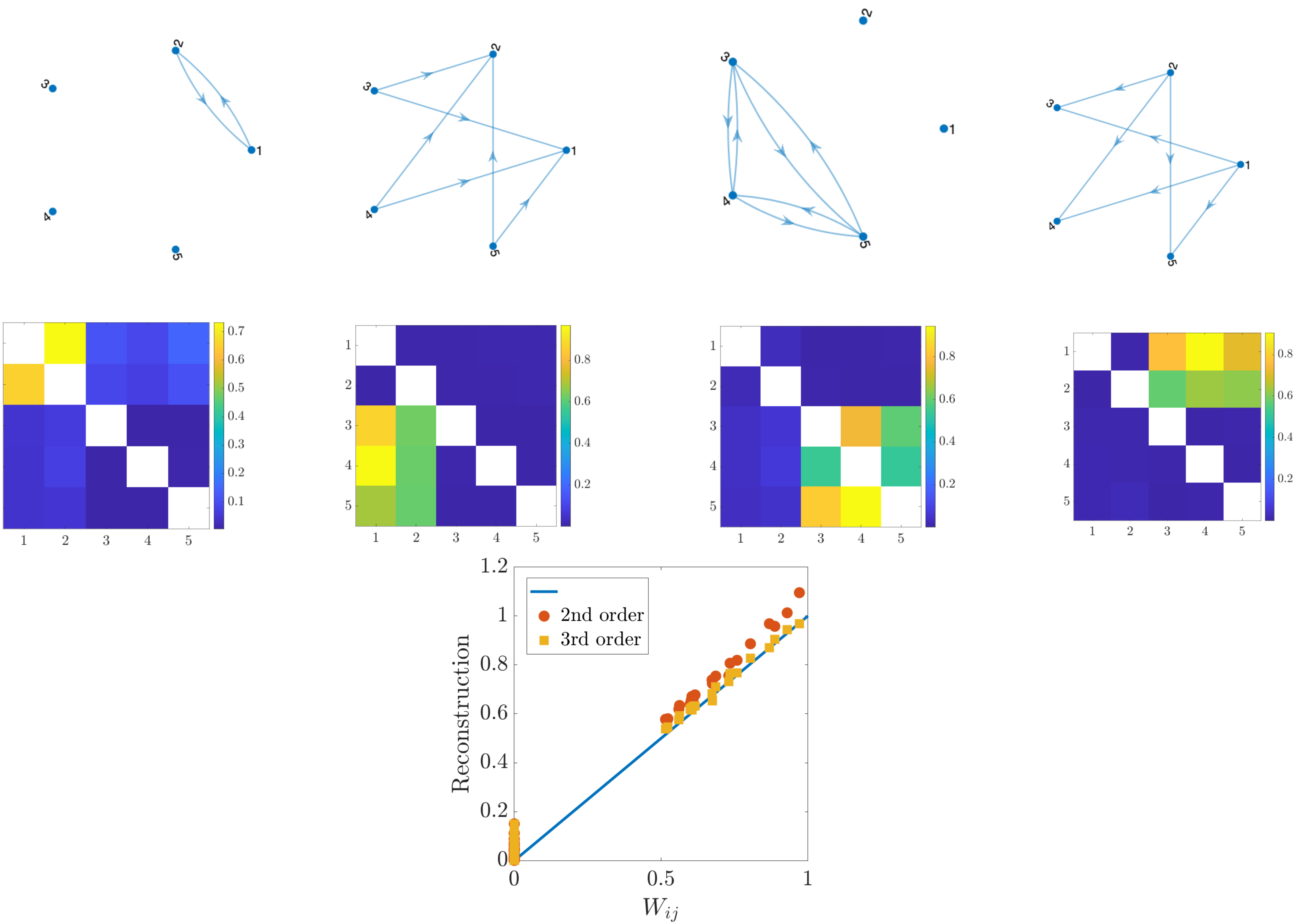
and the remaining $(N-1)M$ deterministically

$$y_{(m)}(t)=y_{(m-1)}(t-1),\ \text{for } m=2,\ldots,M$$

The result of the calculation

- Repeat our procedure
  - Reformulate as a Markov chain on a 2^{MN} dimensional state space
  - Calculate stationary vector
  - Use the transition law and the stationary vector to compute transfer entropy in terms of weights W^{ij}
- This is the worst calculation I had to do in my entire life
- The result is entirely analogous to what I obtained in the first problem

One last numerical experiment




why, thank you.