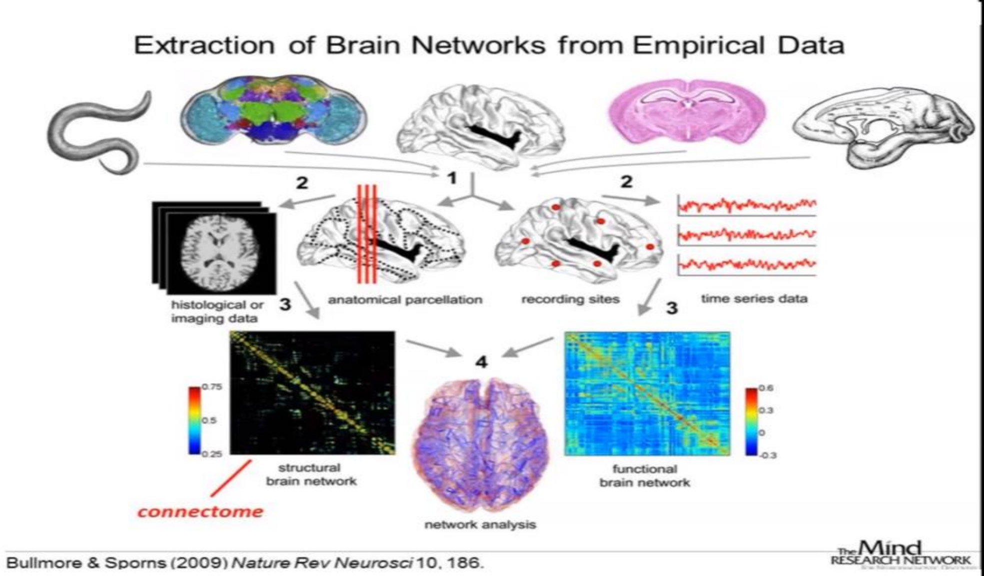
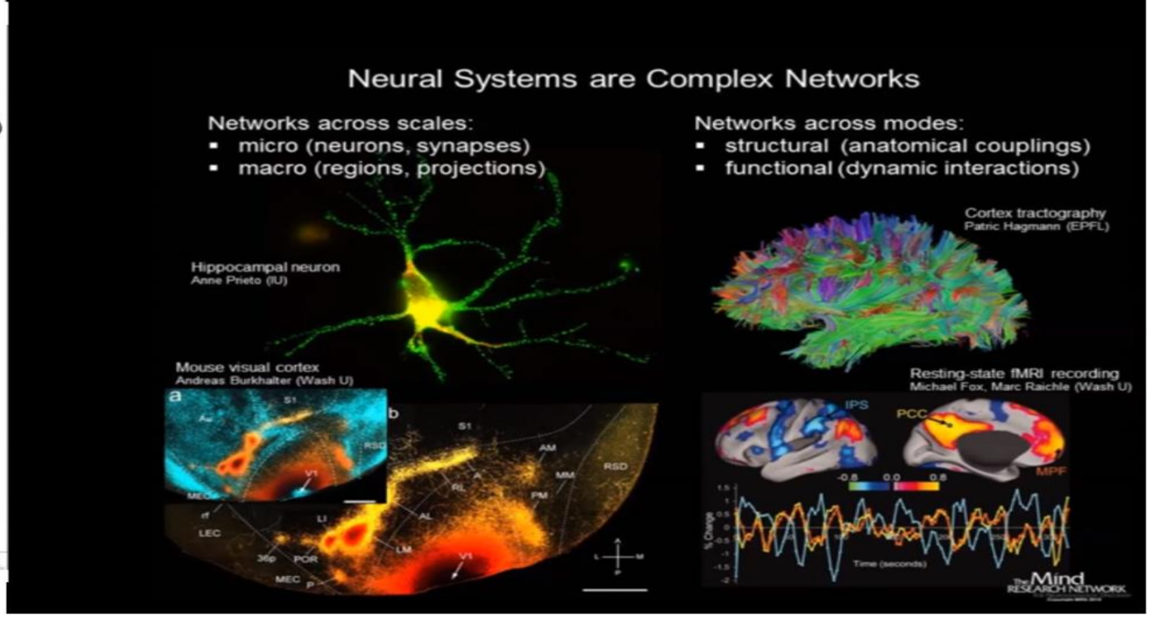




Introductory Lecture on Recurrence Quantification Analysis

Prof. Maria Papadopouli

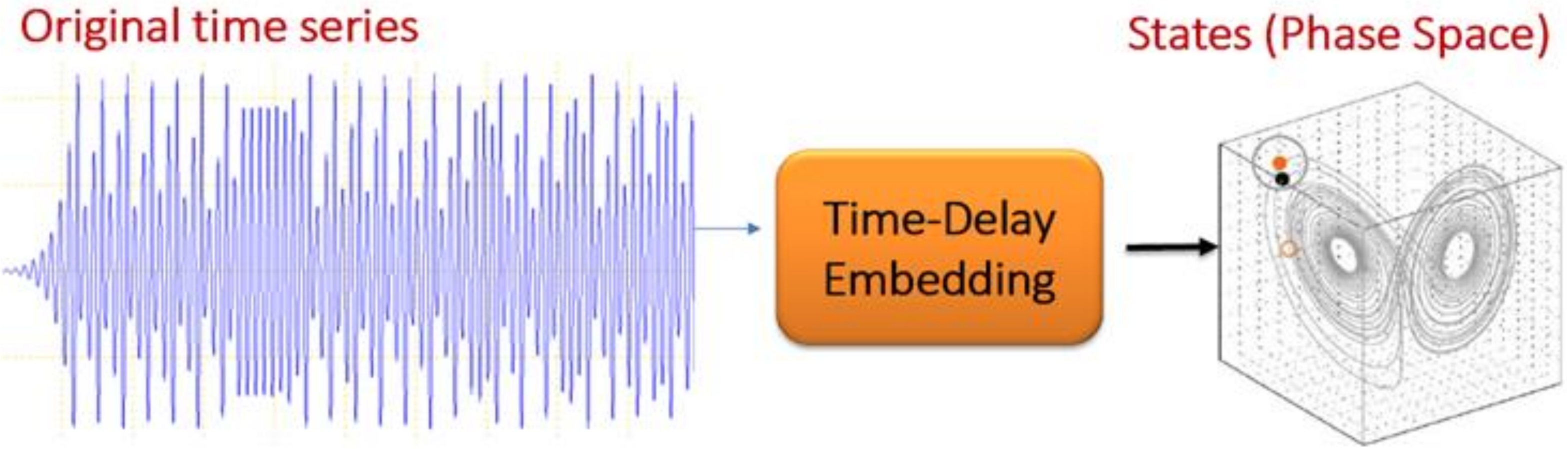
CS – 590.21 Analysis and Modeling of Brain Networks

[Department of Computer Science](#)
University of Crete

Recurrence Quantification Analysis (RQA)

- Powerful tool that uses theory of **non-linear dynamics** based on the topological analysis of the phase space of the underlying dynamics
- Enables the understanding of the **behavior of a complex dynamic system,** e.g., deterministic, random, chaotic
- Does **not make any assumption about the model** that governs the system or the data (e.g., linearity, convexity, stationarity)
- Can handle short time-series, non-stationary data
- Is robust to outliers

Phase Space Representation



$$\mathbf{r} = \{r_i\}_{i=1}^{n}$$

$$\mathrm{x}_i = [r_i, r_{i+\tau}, \ldots, r_{i+(m-1)\tau}] \quad i = 1, \ldots, N$$

	Critical Parameters
m	Embedding dimension
τ	Delay

Time-delay Embedding

- Phase-space reconstruction
- Objective: Unfold the projection back to a **multivariate state-space** that is representative of the original system
- Parameters
  - Embedding dimension (**m**)
  - Time delay (**τ**)

Estimating the Optimal **m using the** **False Nearest Neighbors (FNN) algorithm**

Eliminate 'false neighbours' by **checking the neighbourhood** of points embedded in projection manifolds of **increasing dimension**:

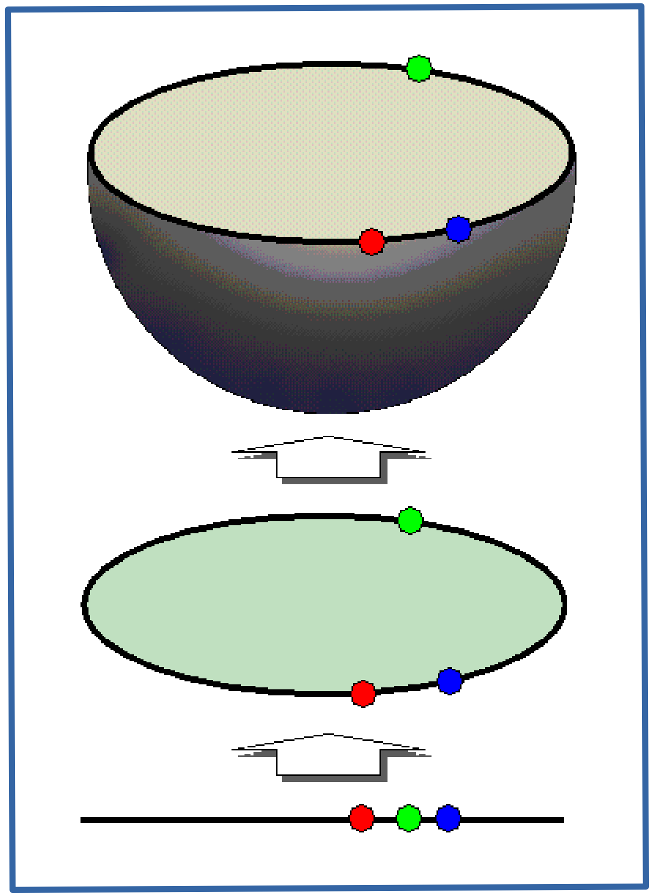
Intuition: when we increased the dimension, the vectors capture a richer (or more complete, more accurate) "picture" of the state

- Points lying close together due to projection are further away in higher embedding dimensions
- If two points are genuine neighbours, they become close due to the system dynamics & separate (relatively) slowly.

False nearest neighbors (FNN) algorithm

- eliminates 'false' neighbours by checking the neighbourhood of points embedded in projection manifolds of increasing dimension:

This means that points apparently lying close together **due to projection** are separated in higher embedding dimensions.



Source: http://people.virginia.edu/~smb3u/NASPSPA9506a/node5.html

Estimating the Optimal **m (con’t)**

- Embed the scalar time series x_d in increasingly higher dimensions
- At each stage compare the number of pairs of vectors v_d and v_d^{NN} (i.e., the nearest neighbour of v_d) which are close when embedded in $\mathbf{R}^d$ but ***not*** when in $\mathbf{R}^{d+1}$.

If two points are genuine neighbours, they become close due to the system dynamics & separate (relatively) slowly.

However, these two points may have become close because the embedding in $\mathbf{R}^d$ has produced trajectories that cross (or become close) due to the “embedding” structure-formation rather than the system dynamics.

Estimating the Optimal **m**

The increase of the distance

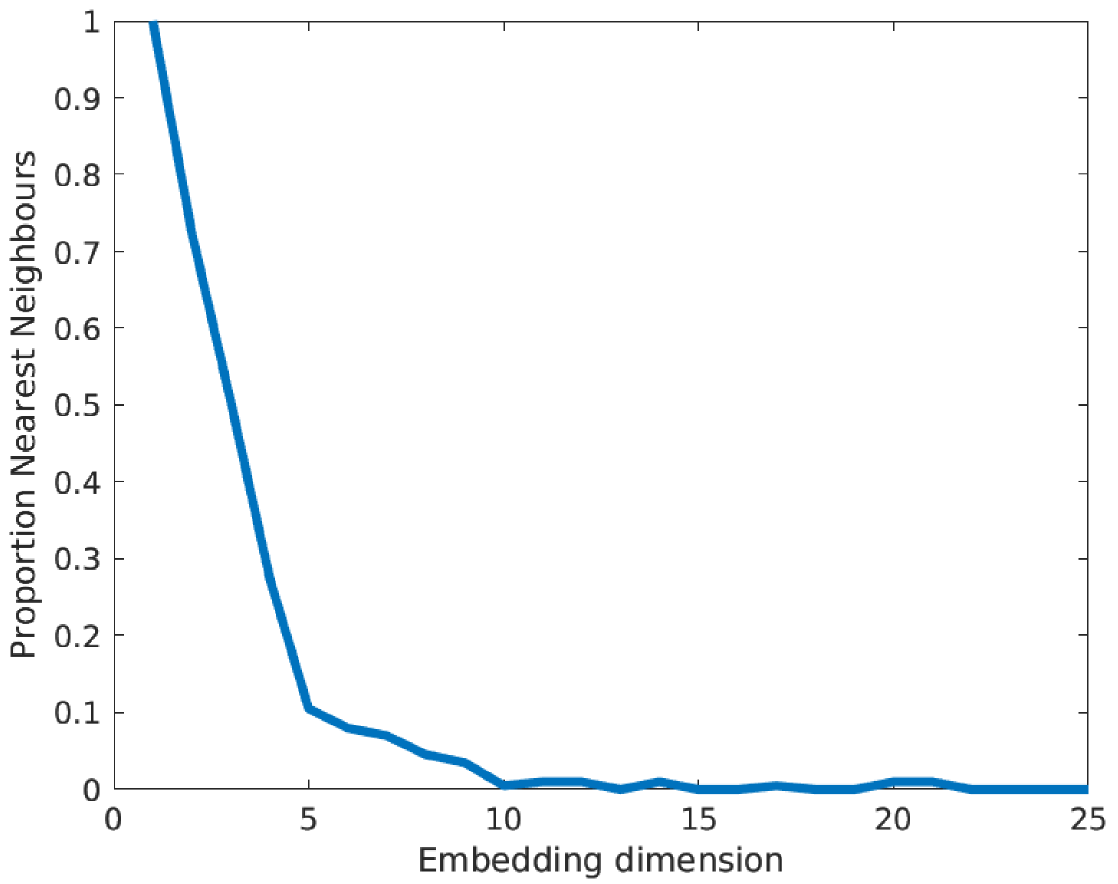
$$\frac{R_{d+1} - R_d}{R_d} > R_{tol}$$

R_d :**Euclidean distance** of two nearest neighbors in the phase space representation of **d** dimension

R_{tol} : threshold above which, false neighbors are identified (typically $10 \le R_{tol} \le 30$)

Estimating the Optimal **m (con’td)**

- The output is the percentual amount of FNN vs embedding dimension
- Monotonically decreasing
- Choose the **smallest m** with the FNN proportion:
  - under a threshold
  - converging



Example: Estimating the optimum **m**

Let's check if the third element of $X_{d=1}(t)$ has a FNN

$X_{d=1}(t) = (7, 5, 1, 3, 5, 9, 12, 10, 9, 9)$

$X_{d=2}(t) = (\{7,5\}, \{5,1\}, \{1,3\}, \{3,5\}, \{5,9\}, \{9,12\}, \{12,,10\}, \{10,9\}, \{9,9\})$

$X_{d=1}(3) = 1$

$X_{d=1}^{NN}(3) = X_{d=1}(4) = 3$

$R_{d=1} = \sqrt{(3-1)^2} = 2$

$$\frac{R_{d+1} - R_d}{R_d} > R_{tol}$$

$R_{tol} = 10$

$X_{d=2}(3) = \{1,3\}$

$X_{d=2}(4) = \{3,5\}$

$R_{d=2} = \sqrt{(3-1)^2 + (5-3)^2} = 2.83$

$$\frac{2 - 2.83}{2} = 0.42 < 10$$

$X_{d=1}(3)$ has no FNN

Estimating the Optimal τ

Employ the **mutual information** (or auto-correlation function)

Intuition: obtain coordinates/"points" for the time delayed trajectory that are **as uncorrelated as possible**

- Auto Mutual Information Function **AMIF(x(t), x(t + τ))**: dependence between the original **time series x(t) & the shifted by τ, x(t + τ)**

$$AMIF(\tau) = \sum_{i,j} p_{ij}^{X(t)X(t+\tau)} \log_2\left(\frac{p_{ij}^{X(t)X(t+\tau)}}{p_i^{X(t)} p_j^{X(t+\tau)}}\right)$$

Select **τ that produces the first local minimum in the AMIF**

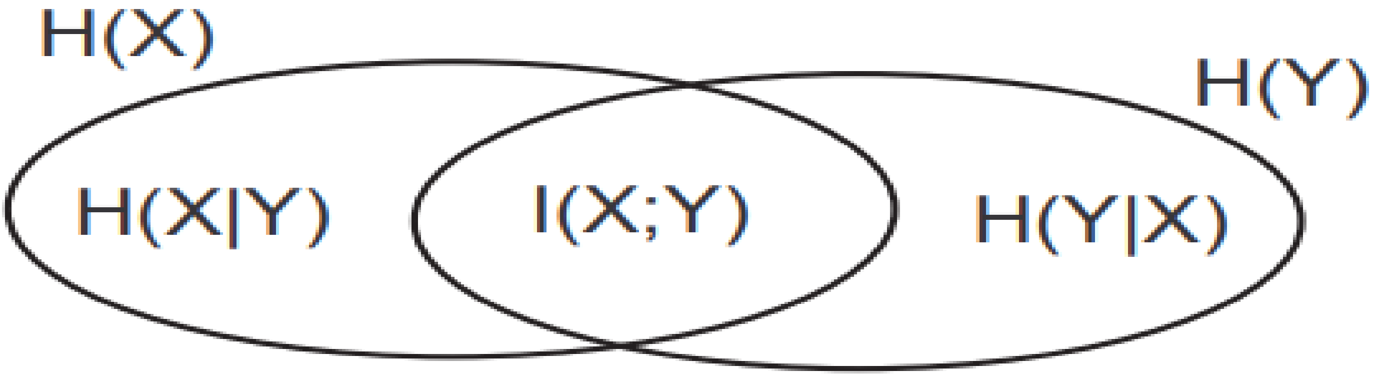
Mutual Information

Quantifies the reduction of uncertainty **relative to a X given the knowledge of Y**

$$I(X;Y) \equiv H(X) - H(X|Y) \quad \text{bit/symbol}$$

or using (1)

$$I(X;Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X,Y)$$



Mutual Information

(Mutual Information) $I(X ; Y) = H(X) - H(X \mid Y)$

$= H(Y) - H(Y \mid X)$

$= I(Y ; X)$

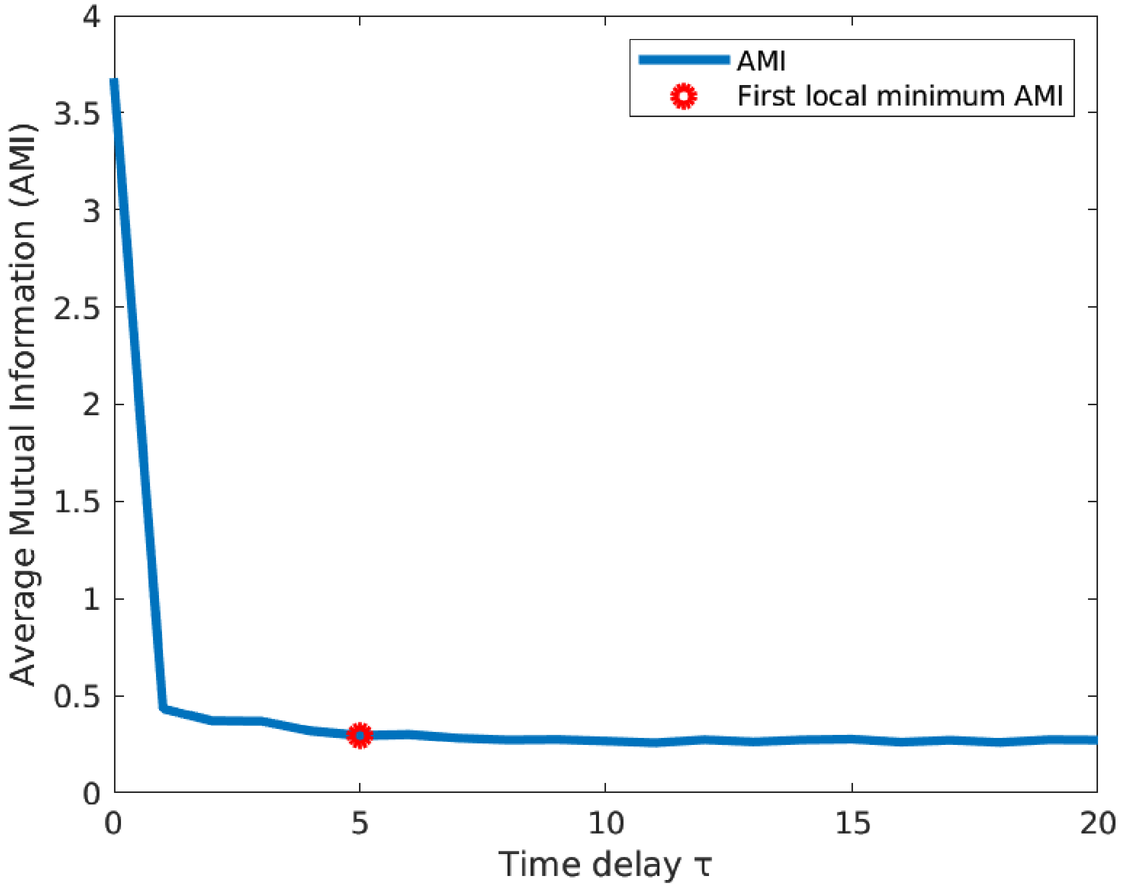
Note: $I(X ; X) = H(X) - H(X \mid X) = H(X)$

(Conditional Mutual Information)

$$I(X ; Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z)$$

Estimating the Optimal **τ**

- **Select time delay τ** which gives least compression of trajectories (the **first local minimum** in the AMIF).
- Obtain coordinates/"points" for the time delayed trajectory that are as independent as possible.



Estimating the Optimal τ

The amplitude distributions are estimated from histograms.

Mutual information:

$$MI(X,Y) = \sum_{i,j} p_{ij}^{XY} \log_2\left(\frac{p_{ij}^{XY}}{p_i^X p_j^Y}\right)$$

Equivalently:

$$MI(X,Y) = H(X) - H(X|Y) = H(X) + H(Y) - H(X,Y)$$

where $H(X,Y) = -\sum_{i,j} p_{ij}^{XY} \log_2\left(p_{ij}^{XY}\right)$

Estimating the Optimal τ

Mutual information (or auto-correlation function)

- Obtain coordinates/"points" for the time delayed trajectory that are as independent as possible

Auto Mutual Information Function **AMIF(x(t), x(t + τ))**: dependence between the original time series & corresponding time shifted

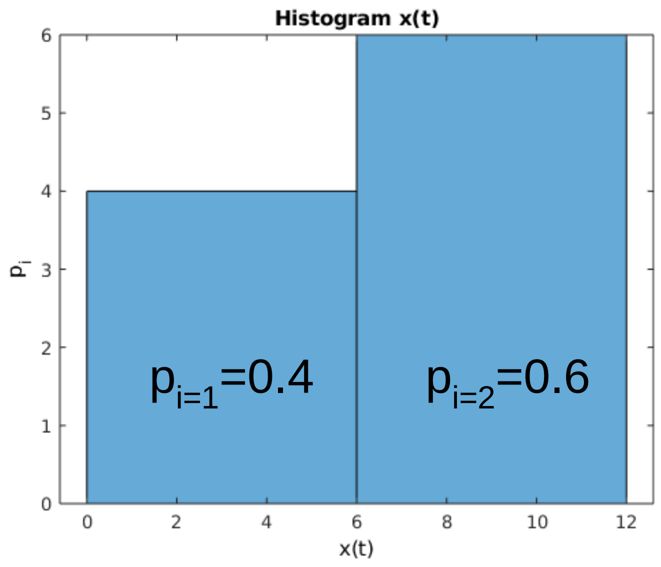
$$AMIF(\tau) = \sum_{i,j} p_{ij}^{X(t)X(t+\tau)} \log_2 \left(\frac{p_{ij}^{X(t)X(t+\tau)}}{p_i^{X(t)} p_j^{X(t+\tau)}} \right)$$

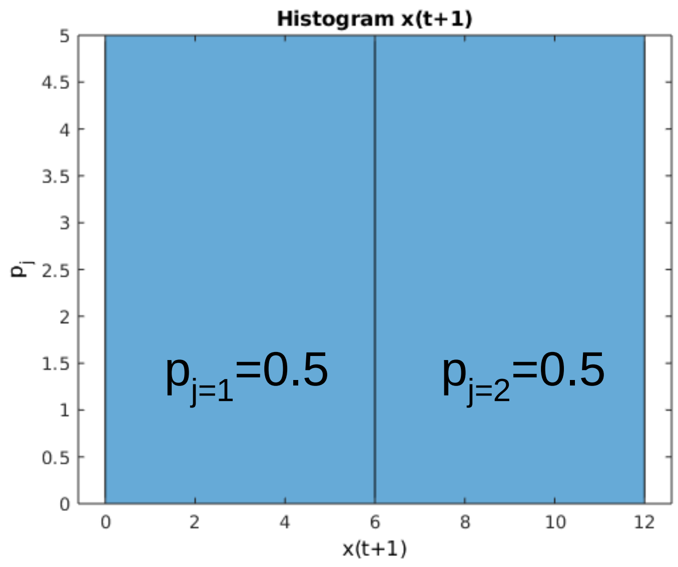
Estimating the optimum **τ**

Example – Calculate the AMI between x(t) and x(t+1)

- $x(t) = (7, 5, 1, 3, 5, 9, 12, 10, 9, 9),$
- $x(t+1) = (5, 1, 3, 5, 9, 12, 10, 9, 9, 0)$

p_{ij}	$p_{j=1}$	$p_{j=1}$
$p_{i=1}$	0.3	0.1
$p_{i=2}$	0.2	0.4





Example: Estimate the Optimum **τ**

Calculate the Auto Mutual Information (AMI) between x(t) & x(t+1)

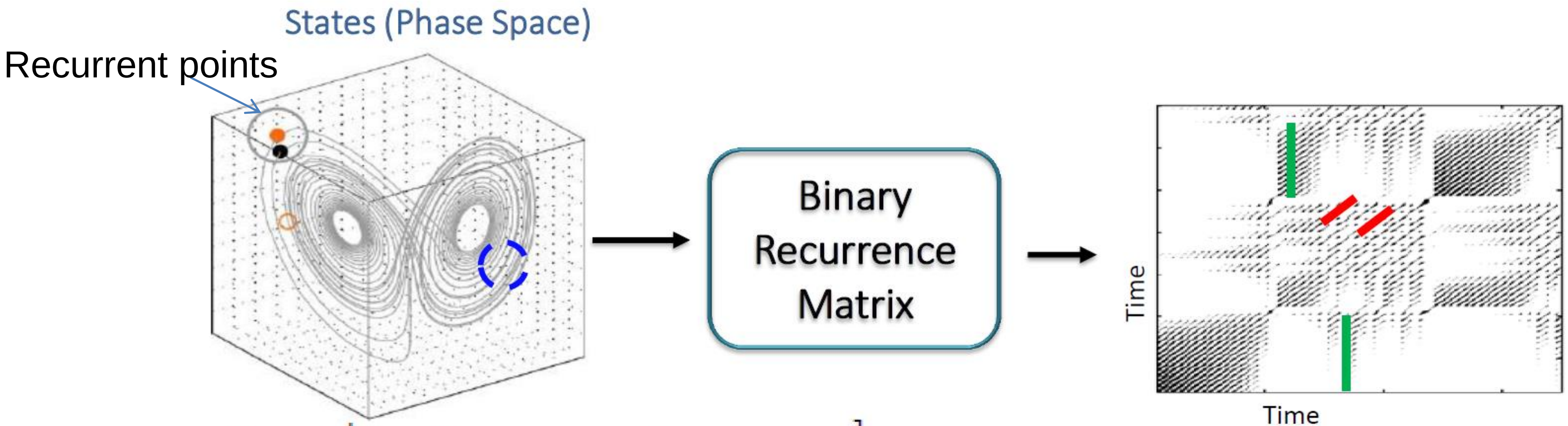
$$MI\left(x(t), x(t+\tau)\right) =$$

$$= p_{11} \log \frac{p_{11}}{p_{i=1} p_{j=1}} + p_{12} \log \frac{p_{12}}{p_{i=1} p_{j=2}} + p_{21} \log \frac{p_{21}}{p_{i=2} p_{j=1}} + p_{22} \log \frac{p_{22}}{p_{i=2} p_{j=2}} =$$

$$= 0.3 \cdot \log \frac{0.3}{0.2} + 0.1 \cdot \log \frac{0.1}{0.2} + 0.2 \cdot \log \frac{0.2}{0.3} + 0.4 \cdot \log \frac{0.4}{0.3} =$$

$$= 0.124$$

Recurrence Plot (RP)



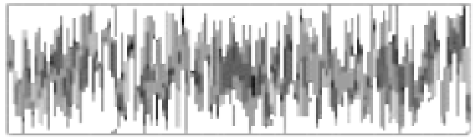
$$\mathbf{x}_i = [r_i, r_{i+\tau}, \dots, r_{i+(m-1)\tau}]$$

$$\mathbf{R}_{i,j} = \Theta\left(\varepsilon - d(\mathbf{x}_i, \mathbf{x}_j)\right) \quad i, j = 1, \dots, N$$

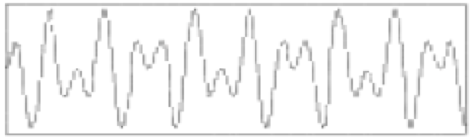
$$\Theta(n) = \begin{cases} 1, & \text{if } n \geq 0 \\ 0, & \text{if } n < 0 \end{cases} \longrightarrow \mathbf{R}_{i,j}(\varepsilon) = \begin{cases} 1, & d(\mathbf{x}_i, \mathbf{x}_j) \leq \varepsilon \quad \bullet \\ 0, & \text{otherwise} \quad \circ \end{cases}$$

Time Series

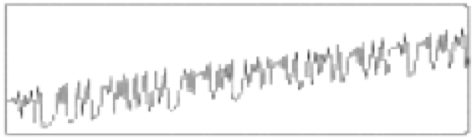
Uncorrelated stochastic data **(white noise)**



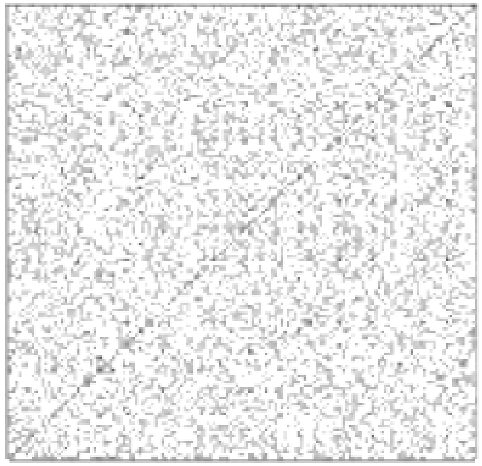
Harmonic oscillation with two frequencies

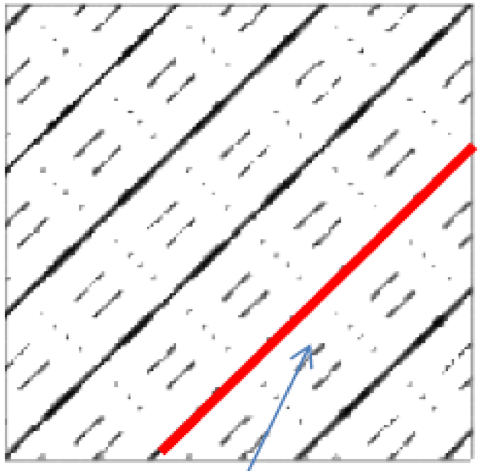


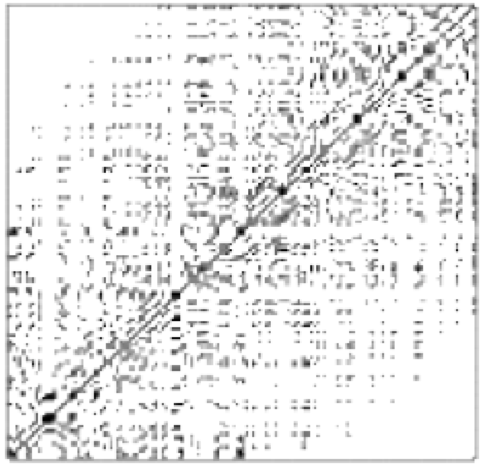
Chaotic data with linear data

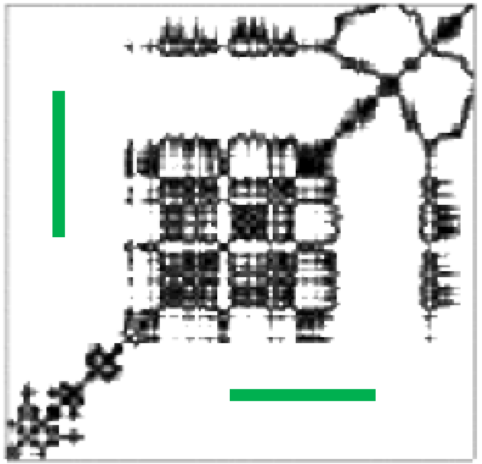


Corresponding recurrence plots









Diagonal lines: State evolution is similar at distinct times; the duration of similar local evolution

Vertical/horizontal lines: Laminar states; State changes slowly

RQA Measures

Laminarity (LAM)

$$LAM = \frac{\sum_{v=v_{min}}^{N} v\,P(v)}{\sum_{v=1}^{N} v\,P(v)}$$

LAM decreases if the RP consists of **more single recurrence points than vertical structures** (e.g., **chaos-order transitions**)

Trapping time (**TT**)

$$TT = \frac{\sum_{v=v_{min}}^{N} v\,P(v)}{\sum_{v=v_{min}}^{N} P(v)}$$

Mean time that the system remains at a specific state

Recurrence Rate (**RR**) based on recurrence density

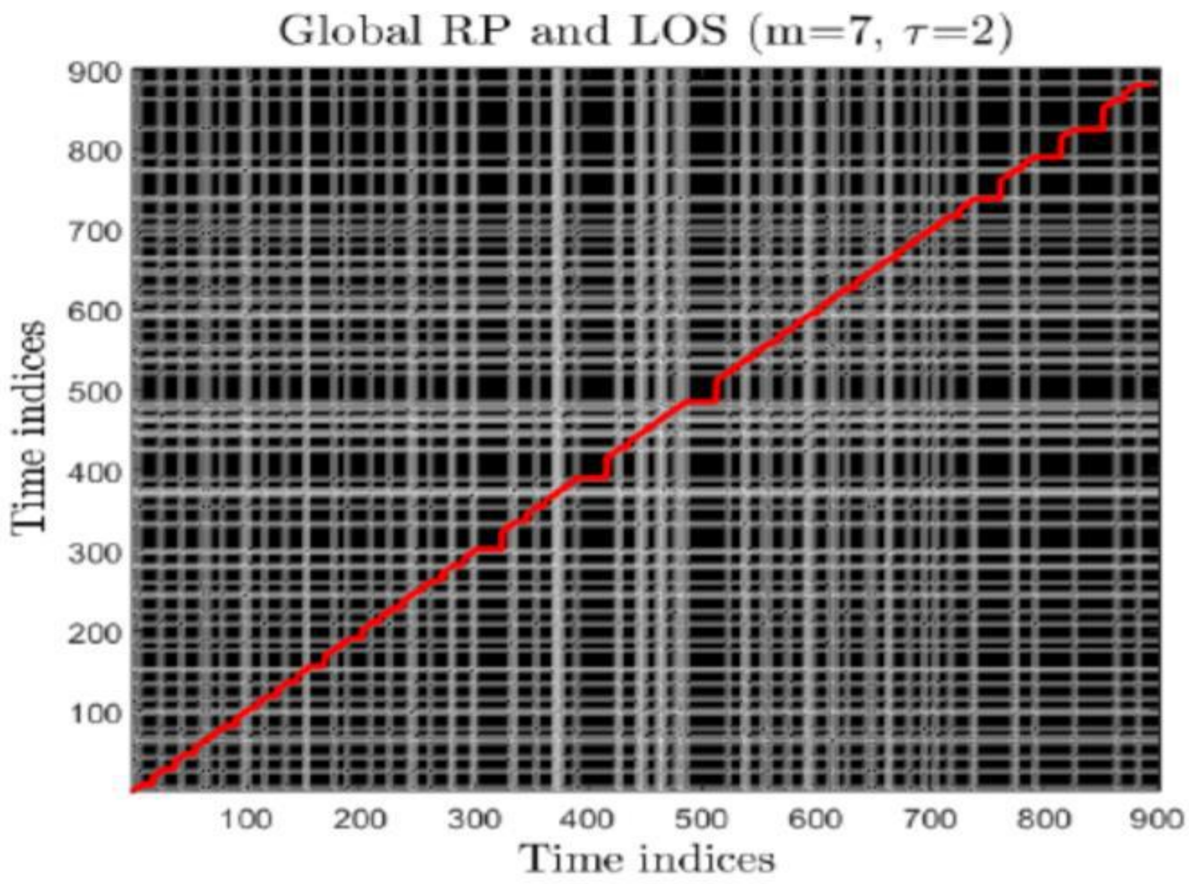
$$RR(\varepsilon) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{R}_{i,j}(\varepsilon)$$

Line Of Synchronization (LOS)

RP reveals the **time instants where the dynamics of the timeseries are somehow constant**

- To identify these instants, apply a specific algorithm which produces the Line of Synchronization

  This line is **distorted,** in comparison to the main diagonal



Construction of the LOS

Start **at the recurrence point (1, 1)** next to the axes origin (**first point of the LOS)**

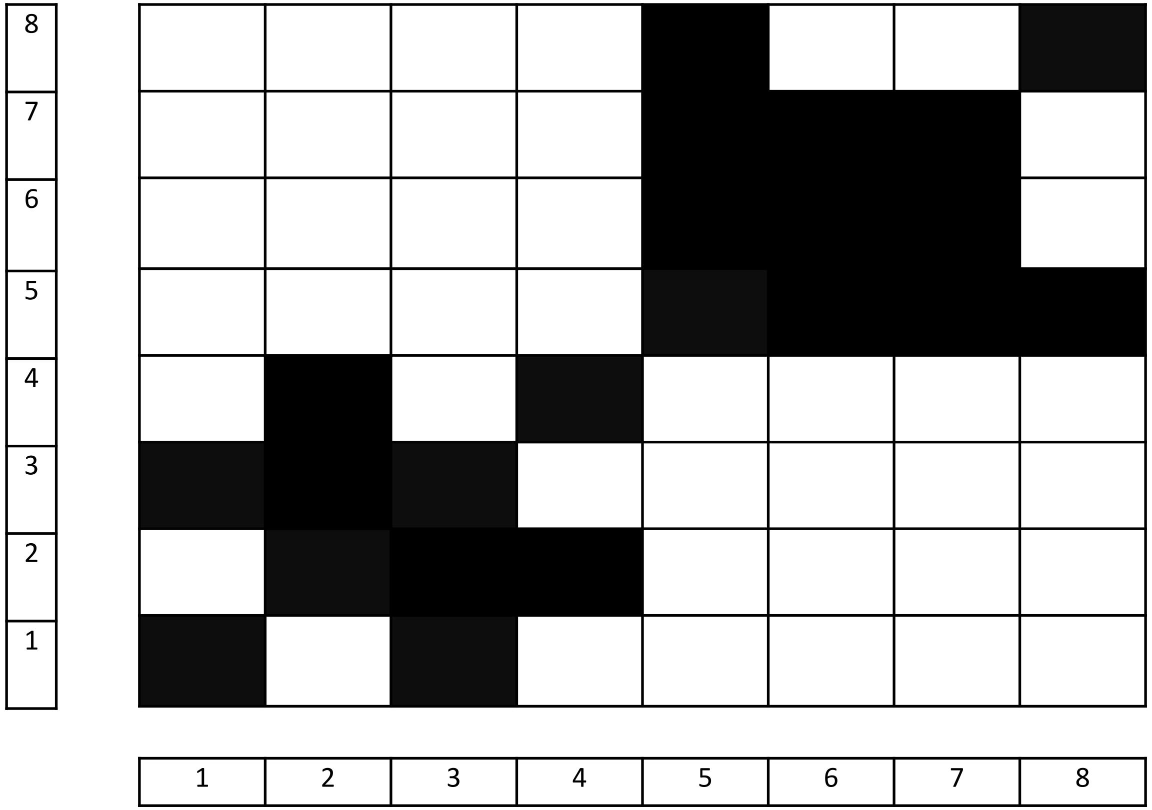
Apply **iteratively** the following steps:

1. At the recurrence point, look for **recurrences in a squared W x W** window (initially W=2)
2. If the squared window contains recurrence points
   a. Increase the window in the horizontal (vertical) direction by a **predefined δx<dx (δy<dy)** parameter OR until no new recurrence points are met
   a. Estimate the **centroid of the cluster of recurrence points**
   b. Set this **centroid as the next point of the LOS**
   Go to step 1

   Else, increase the size of the window to W+1 and go to step 2

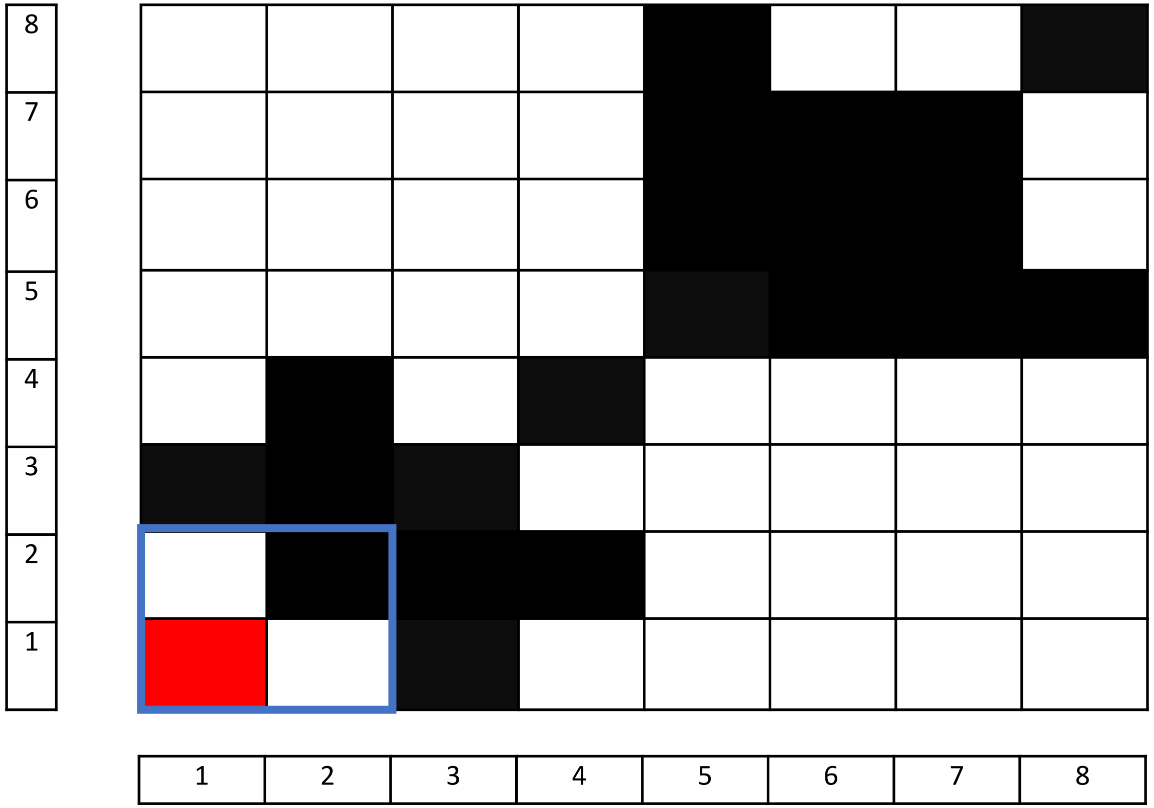
A step by step illustration (1 of 7)

Suppose the Recurrence Plot below has been produced by applying RQA to our timeseries
We set dx = dy = 2 for the algorithm estimation



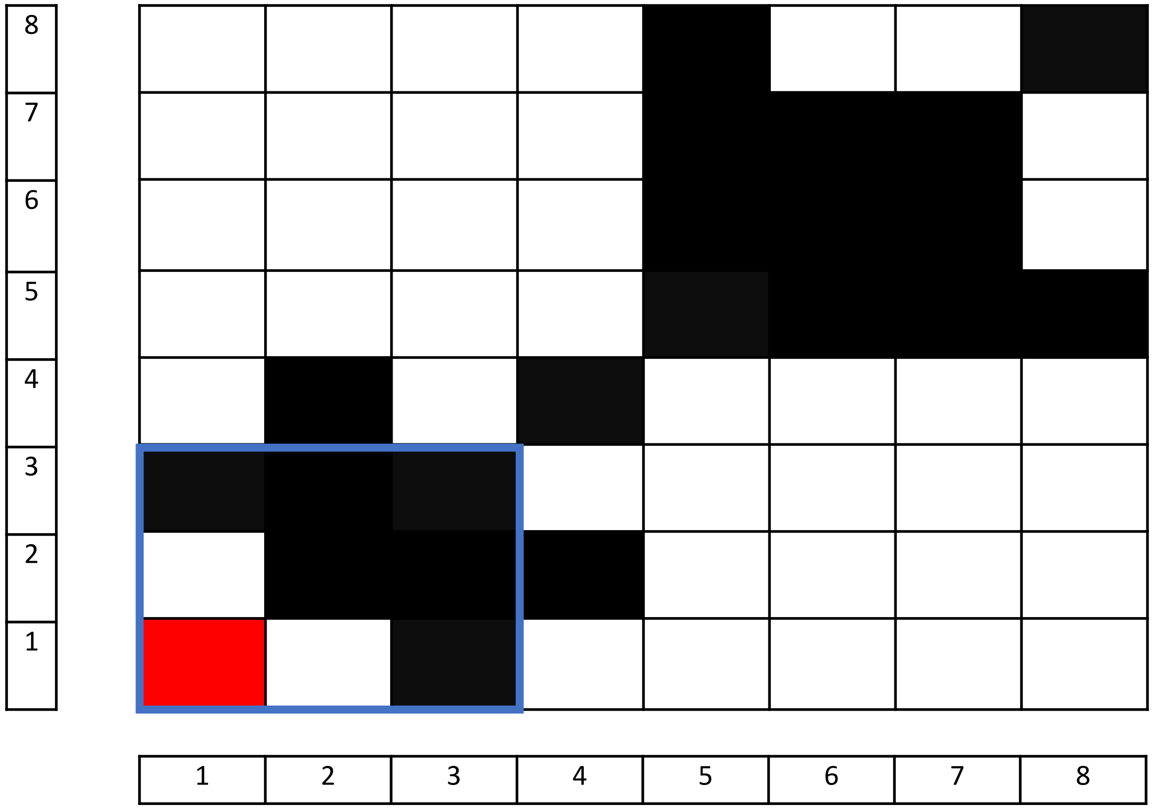
A step by step illustration (2 of 7)

- Set the first recurrence point as (1,1)
- Look if the edge of a squared 2 x 2 window (in blue) contains recurrence points



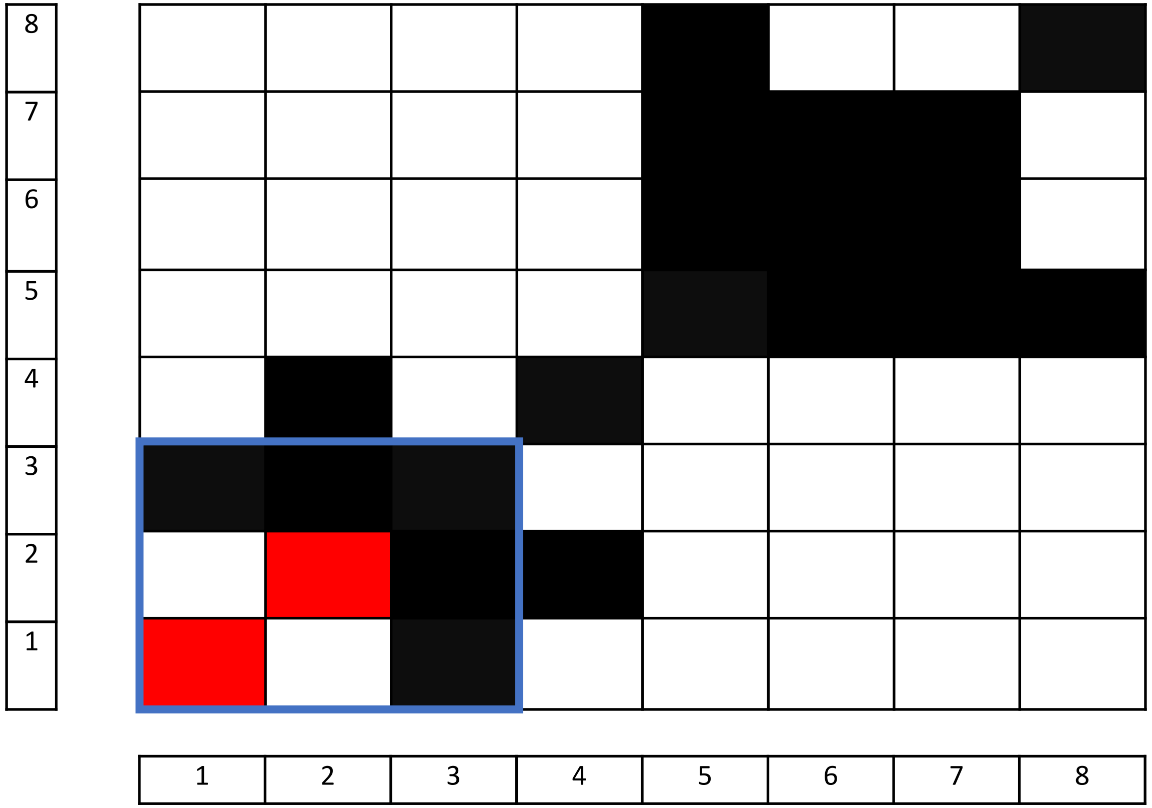
A step by step illustration (3 of 7)

- Set the first recurrence point as (1,1) V
- Look if the edge of a squared 2 x 2 window (in blue) contains recurrence points V
- Increase the window in the x-direction until a predefined δx < dx (δx=1) or until no new recurrence points are met
- Similarly, do this in the y-direction



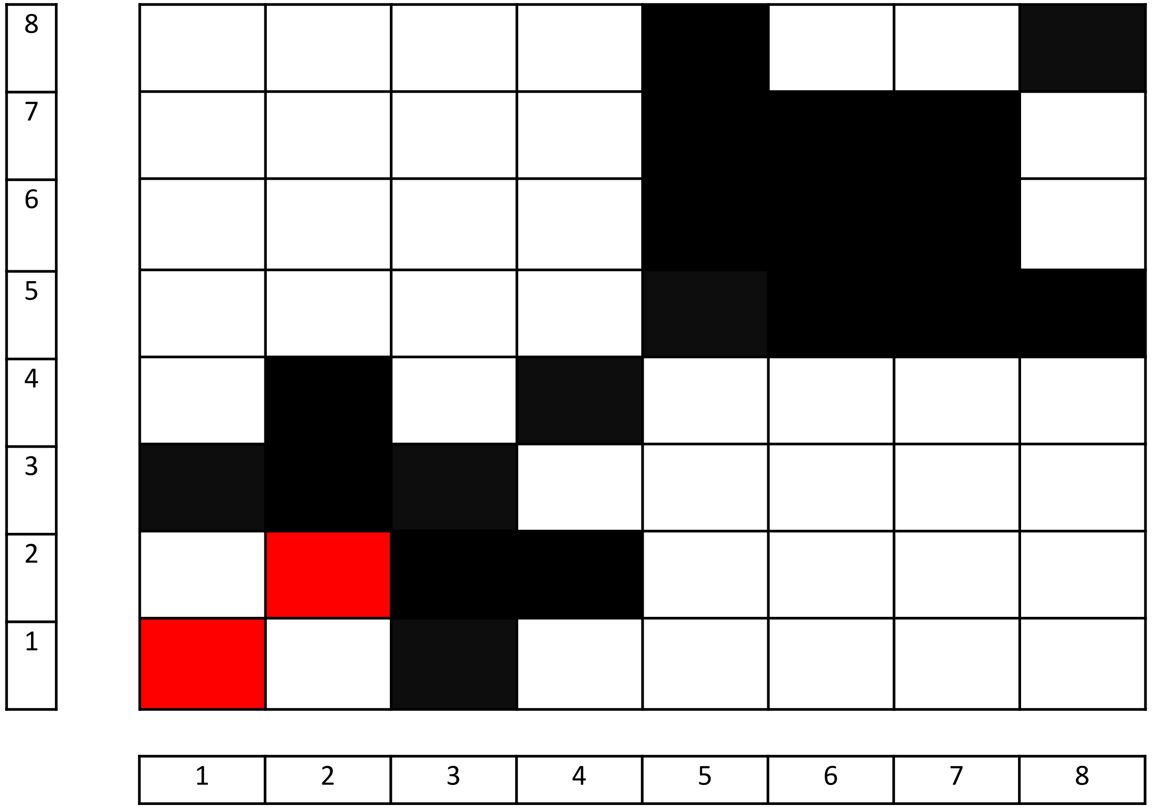
A step by step illustration (4 of 7)

- Set the first recurrence point as (1,1) V
- Look if the edge of a squared 2 x 2 window (in blue) contains recurrence points V
- Increase the window in the x-direction until a predefined δx < dx < 2 (δx=1) or until no new recurrence points are met V
- Similarly, do this in the y-direction V
- Determine the centroid of the cluster of points & set it as the next recurrence point



A step by step illustration (5 of 7)

- Start from the previously set recurrence point
- Follow again the aforementioned steps in order to determine the next recurrence point
- Do this iteratively until you reach the end of the Recurrence Plot



A step by step illustration (6 of 7)

Connect all the recurrence points which have been defined by the algorithm

Then, the Line of Synchronization is produced (below in red)

A step by step illustration (7 of 7)

In addition, the LOS vector that contains the position of each centroid in the RP is derived

- Each position of the vector is connected to the x-dimension of the RP
- Each value of the vector is connected to y-dimension of the RP

The LOS vector for our example is (1, 2, 2, 4, 5, 5)

Interpretation of the LOS (1 of 2)

We can say that:

- The state of time instant 1 is similar to itself
- The state of time instant 2 is similar to itself
- The state of time instant 3 is similar to the state of time instant 2, etc.

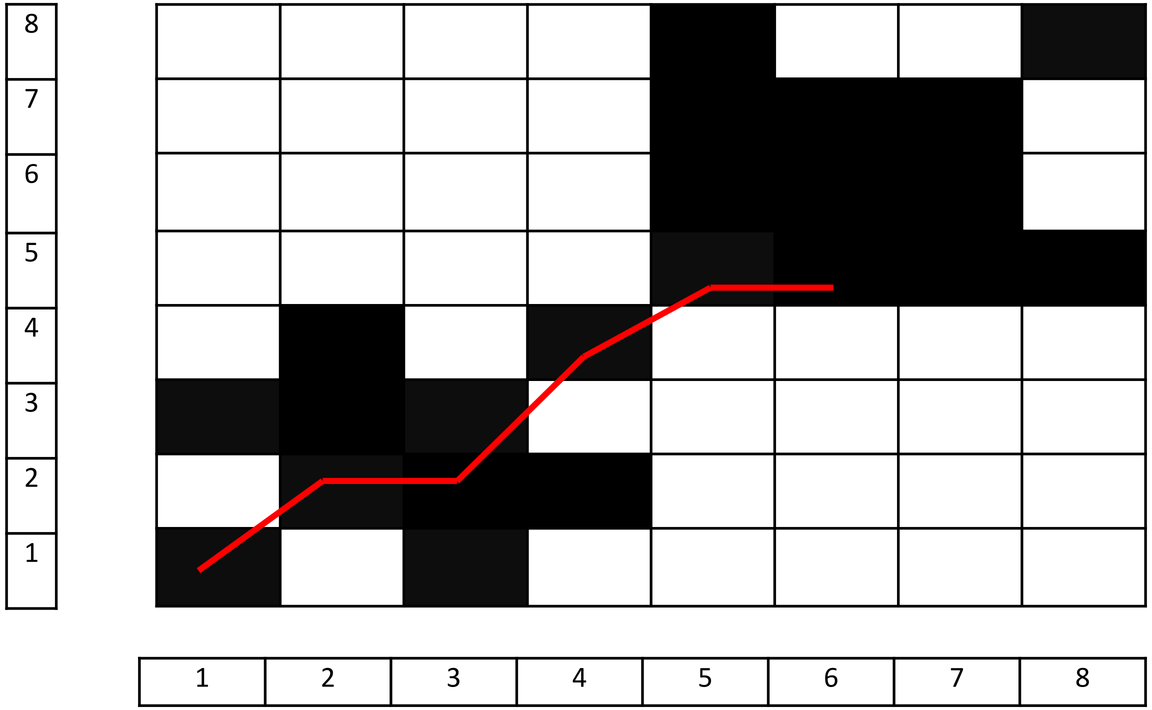
Interpretation of the LOS (2 of 2)

If we now subtract each element of the LOS vector with its predecessor (**calculate pairwise differences**), some segments are formed which contain zero differences.

(1, 2, 2, 4, 5, 5) ⟶ (1, 0, 2, 1, 0)

During these segments, obviously the LOS is constant

- Implying that the **dynamics of the timeseries during these time instants, are also constant**
  In other words, they are **mapped to a specific event**
- The **length of these segments is connected to the duration of the event**



References

- G. Tzagkarakis, G. Palagina, I. Smirnakis, S. M. Smirnakis, M. Papadopouli, Detection Of Stimuli Changes In Neural Eventograms Using The Line Of Synchronization Of Global Recurrence Plots
- M.Kennel, R. Brown, H. Abarbanel, Determining embedding dimension for phase-space reconstruction using a geometrical construction
- S. Wallot, D. Monster, Calculation of Average Mutual Information (AMI) and False-Nearest Neighbors (FNN) for the Estimation of Embedding Parameters of Multidimensional Time Series in Matlab
- N. Marwan, M. Thiel, N.R. Nowaczyk, Cross Recurrence Plot Based Synchronization of Time Series
- A. Fabretti, M. Ausloos, Recurrence Plot and Recurrence Quantification Analysis Techniques for Detecting A Critical Regime. Examples from Financial Market Indices
- A. Fraser, H. Swinney, Independent coordinates for strange attractors from mutual infoiiiiation
- G. Ouyang, Y. Wang, X. Li, Auto Mutual Information Analysis with Order Patterns for Epileptic EEG