MASTER THESIS

Title: Analysis and detection of synchronous behavior in neurons population depending on connectivity model

Student: Manuel López Martín
Tutor: Francisco de Borja Rodríguez Ortiz
Term: 2º Year: 2012/13
Date: 27/01/2013

Abstract

The study of synchronization is a hot topic of research in several fields of science and engineering. It is a difficult and evasive topic.

In a general way everyone has an understanding of synchronization, but trying to put the intuitive vision in a rigorous, mathematical form is not trivial.

Synchronization can be defined as the ¨coherent¨ behavior of the dynamic properties of several connected entities. The entities have to have some kind of connectivity defined between them. The connectivity structure plus the dynamic properties of each entity can be modeled as a network of interconnected entities.

This work tries to provide a general presentation on synchronization, with focus on (1) methods to detect it and (2) the intertwined between synchronization and graph properties of the connectivity network. The main part of the work is dedicated to the former aspect, plus the implementation of a generic Matlab Toolbox that implements some of the detection methods presented.

Additionally it has been developed two new methods to detect synchronization in a signals ensemble. Experimental results of applying the detection methods are included as part of the work.

Index

1 Introduction

The study of synchronization is a main topic of current research in different fields: Computational neuroscience[3], Temporal synchronization of image sequence [105], Data mining in multivariate data, Information retrieval by content [104], Analysis of geologic and atmospheric data[60], Human movements, gaits and gesture recognition [75], Financial applications (trends, similarities,..)[103][29], Genetics (DNA,..)[46], etc…

Synchronization is defined by Webster dictionary as: " (1) happening, existing, or arising at precisely the same time, (2) recurring or operating at exactly the same periods".

This apparently easy definition hides some troubles in its practical usage due to: (1) inherent signal noise, (2) imperfect signal coincidences, (3) alternatives in signal parameter used to establish signals similarity (instantaneous value, phase, frequency,…), (4) establish similarity between more than two signals, (5) similarity between two signals ensemble, (6) patterns recurrence (which pattern? which neurons? spatio-temporal?),… These difficulties imply more efforts dedicated to provide new and better methods for synchronization detection mechanisms.

These efforts are justified due to the importance of synchronization in the above mentioned fields.

Particularly, in neuroscience, synchronization is been studied as one of the main elements that the brain uses in the integration of dissimilar information to form a coherent perception (binding problem)[101] and as an important mechanism for information coding and learning[102].

Synchronization in the brain is the manifestation of dynamic behavior of a highly connected dynamical system made of a multitude of small nodes with its own dynamical behavior. Everything in this highly complex system can have an impact in the final synchronous state: (1) dynamics of the nodes (the neurons), (2) structure of connection between nodes (graph of connectivity), (3) weights and weights dynamic on connections and (4) inputs received by the system.

This work has the intention to explore the above mentioned problems related with synchronization and how synchronization can be produced in a system made of interacting dynamical nodes.

In order to start with, and considering the difficulties found to define and detect synchronization, a good effort has been dedicated to the study of both problems. A thorough review of detection techniques has been performed with the proposal of some additional new methods that can be used for non-stationary signals.

The highly connected multi-node system that forms any part of the brain is a neural ensemble. As mentioned before, spatio-temporal synchronization in neural ensembles

seems to be one of the main mechanisms of information coding. In order to explore further these aspects and to allow the study of synchronization in a neural ensemble, it has been developed a Matlab Toolbox which creates a simulation environment where the nodes dynamics and connectivity structure can be changed dynamically. The results can be analyzed and some of the detection methods presented previously can be applied directly to the generated data.

This work also provides some computational experiments made with the Matlab toolbox (and additional code) to show numerically some of the theoretical results of the thesis.

2 Objective

This work serves several objectives. The objectives and the work performed in order to satisfy them are presented below:

1. Analysis of the state of the art of available techniques to detect the synchronization of a population of neurons.
   To present the methods, organize and structure them.
   To study synchronization detection techniques not only between two signals but also multi-signal and for both continuous and spike data:
   - It has been analyzed over 60 detection techniques for the following groups: Correlation, Granger causality, Phase, State space, Information Theory, Divergence, Frequency flows, symbolic dynamics, Synchronization Clusters, Matrix similarity, Recurrence, Information distance, Point-process, Synfire chains and Unitary events.
2. To develop new methods of synchronization detection:
   - Two new methods has been proposed, one based on phase analysis for multivariate signals and the other based in the analysis of distance measures recurrence at different time shifts.
3. To study the dependence of synchronization with network connectivity (connection graph) in a dynamic network model:
   - It has been studied existing methods, based on both structural properties of the network and spectral properties (eigenvalues and eigenvectors of the Laplacian matrix associated with the adjacency matrix of the graph connectivity)
4. To have available an application which allows performing computational experiments related with synchronization in a population of neurons under several connectivity models:
   - It has been developed a Matlab Toolbox which allows dynamically to set a number of parameters that defines the simulation of a population of neurons:
     - ✓ Size and simulation time
     - ✓ Network connectivity: Small World, Random, Scale free or Regular (lattice).
     - ✓ The dynamic model of excitatory and inhibitory neurons, according to the Izhikevich neural model.
     - ✓ Coupling weights
     - ✓ Various input signals

To provide the following results:

- ✓ Simulated signals (continuous spikes)
- ✓ Signals averages (mean field, average number of spikes, ..)
- ✓ Adjacency matrices and network connectivity
- ✓ Synchronization indices based on network structure
- ✓ Phase analysis (bivariate and multivariate)
- ✓ Recurrence analysis based on time shifts.
- ✓ Analysis of divergence of two probability distributions (applicable to the probability distributions obtained with the toolbox)

5. To develop numerical experiments to validate some of the theoretical results given previously:
   - ➢ Numerical experiments has been conducted to demonstrate:
     - ✓ Synchronization dependence with node dynamical model
     - ✓ Synchronization dependence with the average degree of the network and synaptic weights
     - ✓ Synchronization dependence with the type of network
     - ✓ Synchronization dependence with synchronization index using spectral methods.
     - ✓ Results in the usage of Hilbert transform for continuous and spike signals.
     - ✓ Effectiveness of the two methods developed in this work, using a number of reference signals.

3 Document content

The present document is divided in the following parts:

- Chapter 4 provides a review of methods available to detect synchronization for two or more signals.
- Chapter 5 studies techniques to reveal recurrent patterns in a signals ensemble.
- Chapter 6 presents the relationship between synchronization and the network structure of a connected dynamical system.
- Chapter 7 shows the Matlab Toolbox developed as part of this work and which objective is to provide an application to study the synchronization phenomena in an ensemble of neurons under various connectivity structures.
- Chapter 8 presents several computational experiments performed in order to validate and study the correctness and efficiency of some of the theoretical results given previously. Great effort has been dedicated to study the suitability of the new methods presented in the work.
- Chapter 9 provides the summary of the work and associated conclusions.

- Chapter 10 gives the references to papers and books used in the work.

4 Synchronization

4.1 Synchronization definition

The definition of synchronization is not an easy task. In a general way, synchronization means correlated in-time behavior between different processes. The Oxford Advanced dictionary, defines synchronization as "to agree in time" and "to happen at the same time".

There are many ways to define synchronization, but using the way adopted in [2], synchronization requires: (1) Separating the dynamics of a large dynamical system into the dynamics of sub-systems. (2) A method for measuring properties of the sub-systems. (3) A method for comparing the properties of the subsystems. (4) A criteria for determining whether the properties agree in time. If they agree then the systems are synchronized.

The above definition implies:

- To divide a system in different subsystems
- To associate a mapping between any trajectory in a subsystem to a value in $\mathbb{R}^k$; $g(x): X \otimes \mathbb{R} \rightarrow \mathbb{R}^k$, where the first $\mathbb{R}$ (on the left) is time and X represent the trajectory of the first subsystem. Correspondingly, $g(y): Y \otimes \mathbb{R} \rightarrow \mathbb{R}^k$ represents the association of a vector in $\mathbb{R}^k$ to a trajectory in the second subsystem.
  We say that the value in $\mathbb{R}^k$ produced by $g(x)$ is a property of the subsystem.
- To define a comparison measure (h) between subsystems properties in $\mathbb{R}^k$ (corresponding to the two subsystems). $h: \mathbb{R}^k \otimes \mathbb{R}^k \rightarrow \mathbb{R}^k$
- The synchronization state between the two trajectories in the two subsystems is established when a norm defined in the above mentioned measure is below a threshold. $\left\| h\left[g(x),\ g(y) \right] \right\| < \varepsilon$
- Usually the comparison measure is based in a kind of subtraction:
  - $h[g(x), g(y)] \equiv g(x) - g(y)$
  - $h[g(x), g(y)] \equiv \lim_{t \to \infty} [g(x) - g(y)]$
  - $h[g(x), g(y)] \equiv \lim_{T \to \infty} \int_{t}^{t+T} [g(x(s) - g(y(s))] ds$

  And the norm is usually a Euclidean or Mahalanobis norm.

In summary, it is necessary to establish a measure to identify a trajectory of a part of the system and compare that measure with other parts, if both measures are similar (the comparison algorithm has to be defined), it is established that both parts are synchronized.

Using this schema can be identified the following (among others) classes of synchronization:

- Identical synchronization: Where $g(x) = x$, that is the metric will work on the direct signal, identifying the similarity of the signals in the broad sense.
- Phase synchronization: where $g(x)$ is the phase of the signal.

- Frequency synchronization: where $h\left[g(x), g(y)\right] = n\omega(x) - m\omega(y)$, and $\omega(x)$ is the frequency of the signal.
- Lag synchronization: where $g(x) = x$ and $g(y) = y(t+\tau)$, that is, the comparison is done between a signal and the shifted version of the other.
- Generalized synchronization: where $h\left[g(x), g(y)\right] = H\left[g(x)\right] - g(y)$, that is, when $y(t) = H\left[x(t)\right]$, where H is a smooth, invertible, time independent function and the state of one subsystem can be completely known from the state of the other.

The above definition of synchronization is generic and shows the difficulties in defining, and later, identifying synchronization. In addition, when synchronization is identified, that synchronization will depend on the selected subsystems partition and the metrics used in identifying synchronization. This is an important point, because in this study I will try to identify synchronization in a population of neurons, and we will see that one of the difficulties will be to identify the subsets of neurons to use.
To the difficulties mentioned it is necessary to add the issues of stationary or not stationary state of the signals, the identified linear or nor linear relationships, and whether the data used to compute synchronization will be direct data (e.g. voltage in time) or indirect (e.g. phase, power in frequency ...).

All these issues make the effort to identify synchronization a difficult pursuit.

In this work I will use synchronization mainly as:

- Identical synchronization: Where $g(x) = x$, that is the metric will work on the direct signal.
- Phase synchronization: where $g(x)$ is the phase of the signal.

In this work the signals used are either bivariate or multivariate time series. Some of the methods presented are used exclusively for bivariate signals, and others can be used for multivariate signals. The difference will be pointed out.
In case of using a bivariate method to identify the general synchronization state of a multivariate signal, a kind of combinatory bivariate exploration is necessary on all (or a set) of the $\binom{n}{2}$ possible combinations of the extracted bivariate signals.

In [15] is presented a complete review of types of synchronization, the work is applied to chaotic synchronization but the presentation is general. In this work is shown how the types of synchronization usually imply a gradient towards greater similarity in the signals. Being phase synchronization the lowest level of similarity (only the phase is similar but not the amplitudes), lag synchronization is the next step, where the signals are similar but they are time lagged with each other, and finally identical synchronization is the highest level of synchronization when all signals are identical. Usually the coupling strength of the system is the parameter that produces the displacement to higher levels of synchronization. Finally generalized synchronization is a way to show the interaction between two systems when there is an essential difference between them. These

systems are said to be synchronized in a generalized way when there is a function that relates the signals produced by the systems (this function does not imply identity or similarity but influence, or the capacity to determine the signals produced by one system from the results obtained in other).

4.2 Synchronization detection methods

This section goes through an exhaustive presentation of the available methods to detect synchronization between several signals (two or more). The methods are presented under specific groups.

4.2.1 Correlation based

4.2.1.1 Cross-Correlation Coefficient

As presented in [3][20] [21] the correlation coefficient is one of the best known mechanisms to detect linear dependence between two signals.
The correlation coefficient r is defined as:

$$r = \frac{1}{N}\sum_{k=1}^{N}\frac{(x(k)-\bar{x})}{\sigma_x}\frac{(y(k)-\bar{y})}{\sigma_y} \quad (1)$$

Where N is the length of the signals:

$x = \{x(1),...,x(N)\}$

$y = \{y(1),...,y(N)\}$

, $\bar{x}$ and $\bar{y}$ is the (sample) mean of x and y respectively, and σ_x and σ_y are the (sample) variance of x and y.

The correlation coefficient r quantifies the linear relationship between x and y. Besides, it is assumed that the signals are stationary.

If x and y are not linearly correlated, r is close to zero; on the other hand, if both signals are identical, then r=1.
If the signals have non-linear relationships, the correlation coefficient will not detect it, even when the relationship exists.

Related to the correlation coefficient are other methods, as the cross-correlation coefficient [22], defined as:

$$C_{xy}(\tau) = \frac{1}{N-\tau}\sum_{k=1}^{N-\tau}\frac{(x(k)-\bar{x})}{\sigma_x}\frac{(y(k+\tau)-\bar{y})}{\sigma_y} \quad (2)$$

Where k=1..N and τ is a time lag.

The cross-correlation is a measure of the linear synchronization between x and y. Its absolute value ranges from zero, no synchronization, to one, perfect synchronization, and is symmetric. This is also an extension of the autocorrelation function that is calculated as above with y(t) equal to x(t) (that is $C_{xx}(\tau)$).

4.2.1.2 Mean square coherence (MSC)

Given the cross-correlation function (see Section 4.2.1.1), the mean square coherence (MSC) is the magnitude of the FFT (Fast Fourier Transform) of the cross-correlation function ($C_{xy}(\tau)$), normalized by the magnitude of the FFTs of the autocorrelation functions of x and y. The MSC is defined as:

$$\gamma_{xy}(f) = \frac{\left|\left\langle S_{xy}(f)\right\rangle\right|^2}{\left|\left\langle S_{xx}(f)\right\rangle\right|\left|\left\langle S_{yy}(f)\right\rangle\right|} \quad (3)$$

Where $S_{xy}(f)$ is the FFT of equation (2), and, similarly, $S_{xx}(f)$ and $S_{yy}(f)$ are the FFT of the autocorrelation functions.

In order to compute this quantity, the signals x and y are subdivided in M segments of equal length L. The coherence function is computed by averaging over those segments. The symbol $\langle\cdots\rangle$ in equation (3) corresponds to this average.

The possible value of MSC at a certain frequency ranges between 0 (no linear dependence) and 1 (maximum linear dependence).

4.2.1.3 Coherence

The coherence function is defined as [21][20]:

$$c(f) = \frac{\left|\left\langle X(f)Y^*(f)\right\rangle\right|^2}{\left|\left\langle X(f)\right\rangle\right|\left|\left\langle Y(f)\right\rangle\right|} \quad (4)$$

Where $X(f)$ and $Y(f)$ denote the Fourier transform of x and y, $f \in [-T/2, T/2]$ is the frequency and the symbol * indicates complex conjugate.

In order to compute this quantity, the signals x and y are subdivided in M segments of equal length L. The coherence function is computed by averaging over those segments. The symbol $\langle\cdots\rangle$ corresponds to the average. The values in the denominator are normalization values corresponding with the individual Fourier transform of the signals.

The coherence function quantifies linear correlations in frequency domain [20]. It is mainly useful when synchronization is limited to a particular frequency band, which is usually the case in EEG [5][3][20].

The coherence function presented above is the magnitude square coherence function. There is another function called the phase coherence function, defined as:

$$\phi(f) = \arg(X(f)Y^*(f)) \quad (5)$$

Where $\arg(\mathrm{X})$ is the argument of X, where X is an imaginary number. Both $c(f)$ and $\phi(f)$ depend on the frequency f.

4.2.1.4 Corr-Entropy Coefficient

As presented in [23][6][7][3] the corr-entropy coefficient r_E is a proposed [23] non-linear extension of the correlation coefficient. The definition of the corr-entropy is:

$$r_E = \frac{\frac{1}{N}\sum_{k=1}^{N}\kappa(x(k),y(k)) - \frac{1}{N^2}\sum_{k,\ell=1}^{N}\kappa(x(k),y(\ell))}{\sqrt{K_X - \frac{1}{N^2}\sum_{k,\ell=1}^{N}\kappa(x(k),x(\ell))}\sqrt{K_Y - \frac{1}{N^2}\sum_{k,\ell=1}^{N}\kappa(y(k),y(\ell))}} \quad (6)$$

Where,

$$K_X = \frac{1}{N}\sum_{k=1}^{N}\kappa(x(k),x(k)) \qquad K_Y = \frac{1}{N}\sum_{k=1}^{N}\kappa(y(k),y(k)) \quad (7)$$

Where κ is a symmetric positive definite kernel function: a Gaussian, sigmoidal, or polynomial kernel. And it is assumed that the signals x and y consist of N samples:

$x = \{x_1,\ldots,x_N\}$

$y = \{y_1,\ldots,y_N\}$

Usually the kernel used is the Gaussian kernel:

$$\kappa(x,y) = \frac{1}{2\pi\sigma}e^{\frac{-|x-y|}{2\sigma^2}} \quad (8)$$

In the original paper [23], σ is given a value of σ=0.4.

The signals x and y need to be normalized before evaluating equation (6), by subtracting the mean and then dividing by the standard deviation. The normalization is important, since x and y may have significantly different dynamic ranges. After the normalization, both x and y are dimensionless.

If x and y are independent then r_E will be zero. The independence condition is stronger than uncorrelated. The closer r_E is to 1 the stronger the dependence.

The basic idea behind corr-entropy is to generalize the autocorrelation function to nonlinear spaces.

If $\{x_t\}$ is a strict stationary stochastic process and t is a discrete index, then the autocorrelation and corr-entropy functions for x_t are defined respectively as:

$$R(s,t) = E\{\langle x_s, x_t\rangle\}$$
$$V(s,t) = E\{\langle \phi(x_s), \phi(x_t)\rangle\} \quad (9)$$

Where ϕ is a nonlinear mapping from the input space to a projected higher dimensional space called the feature space.

Instead of explicitly defining a mapping, computing the mapping, and then taking an inner product of the mapping, corr-entropy makes use of the ‘‘kernel trick’’ which defines the inner product of the nonlinear mappings as a positive-definite Mercer kernel :

$$\kappa(x_s, x_t) = \langle \phi(x_s), \phi(x_t) \rangle \quad (10)$$

Implying that calculating the corr-entropy is reduced to obtain the expected value of the kernel.

When $V(s,t)$ (equation (9)) is extended to two signals x and y, then equation (6) is obtained.

An efficient algorithm to compute r_E is presented in [24] using an incomplete Cholesky decomposition of the kernel matrix, defined as:

$$\mathbf{K}_{XX}(i,j) := \kappa(x_i - x_j) \quad (11)$$

Where the signals x and y consist of n samples:

$$x = \{x_1, \ldots, x_n\}$$
$$y = \{y_1, \ldots, y_n\} \quad (12)$$

The indexes i and j in equation (11) refer to the sample indexes in (12).
The kernel function in equation (11) is defined as acting directly on the difference of the signals, as is usually the case for all kernel functions (e.g. Gaussian kernel (equation(8))).

In order to have a symmetric positive matrix where to apply the Cholesky decomposition, the kernel matrix is extended in the following way:

$$\mathbf{K}_{ZZ} = \begin{bmatrix} \mathbf{K}_{XX} & \mathbf{K}_{XY} \\ \mathbf{K}_{YX} & \mathbf{K}_{YY} \end{bmatrix} \quad (13)$$

Where

$\mathbf{K}_{YX} = \mathbf{K}_{XY}^{\top}$, is a symmetric matrix.

Once defined the extended kernel matrix, an alternative definition of corr-entropy (using a different symbol c for r_E) can be given as [24]:

$$c = \frac{\mathbb{E}_{XY}\kappa(X-Y) - \mathbb{E}_X\mathbb{E}_Y\kappa(X-Y)}{\sqrt{\left(1 - \mathbb{E}_{X_1}E_{X_2}\kappa(X_1 - X_2)\right)\left(1 - \mathbb{E}_{Y_1}E_{Y_2}\kappa(Y_1 - Y_2)\right)}} \quad (14)$$

And an estimate of corr-entropy can be calculated as:

$$\hat{c} = \frac{\frac{1}{n}\sum_{i=1}^{n}\kappa(x_i - y_i) - \frac{(\mathbf{e}_1^{\top}\tilde{\mathbf{G}}_{ZZ})(\tilde{\mathbf{G}}_{ZZ}^{\top}\mathbf{e}_2)}{n^2}}{\sqrt{\left(1 - \frac{(\mathbf{e}_1^{\top}\tilde{\mathbf{G}}_{ZZ})(\tilde{\mathbf{G}}_{ZZ}^{\top}\mathbf{e}_1)}{n^2}\right)\left(1 - \frac{(\mathbf{e}_2^{\top}\tilde{\mathbf{G}}_{ZZ})(\tilde{\mathbf{G}}_{ZZ}^{\top}\mathbf{e}_2)}{n^2}\right)}} \quad (15)$$

Where:

$$\mathbf{e}_1 = \{\underbrace{1,\ldots,1}_{n},\underbrace{0,\ldots,0}_{n}\}^\top \text{ and } \mathbf{e}_2 = \{\underbrace{0,\ldots,0}_{n},\underbrace{1,\ldots,1}_{n}\}^\top$$

And G and G^T form part of the incomplete Cholesky decomposition of K_{ZZ} .

K_{ZZ} can be decomposed using the Cholesky decomposition: $GG^T = K_{ZZ}$, which is a special case of LU decomposition for symmetric positive definite matrix.

If the eigenvalues of K_{ZZ} drops rapidly then the matrix can be approximated by a $n \times d$ $(d < n)$ lower triangular matrix G with arbitrary accuracy, meaning that, $\left\| K_{ZZ} - GG^T \right\| < \varepsilon$ where ε is any small positive number and $\|\cdot\|$ is a suitable matrix norm. This decomposition of $K_{ZZ} \approx GG^T$ is called the incomplete Cholesky decomposition of K_{ZZ} .

4.2.1.5 Coh-Entropy and Wav-Entropy Coefficient

As presented in [3] , and in a similar way to the linear case, a nonlinear magnitude square coherence function, which is again a normalized measure, can be defined. The coefficient obtained $c_E(f)$ is called coh-entropy (coherence-entropy) and is defined as:

$$c_E(f) = \frac{\langle \kappa(X(f), Y(f)) \rangle}{\sqrt{\langle \kappa(X(f), X(f)) \rangle}\sqrt{\langle \kappa(Y(f), Y(f)) \rangle}} \quad (16)$$

Where the averages $\langle . \rangle$ are computed over M segments of equal length L.

Before $c_E(f)$ can be computed, the Fourier transforms $FT(x)(f) = X(f)$ and $FT(y)(f) = Y(f)$ need to be normalized.

The wav-entropy is obtained as a new generalization of $c_E(f)$ when, instead of using $X(f)$ or $Y(f)$ as in equation(16), it is used the wavelet transform of x and y:

$$X(k,s) = \sum_{\ell} x(\ell)\psi^*\left(\frac{\ell - k}{s}\right) \quad (17)$$

Where $\psi(k)$ is the (complex) mother wavelet and s is a scaling factor.

Instead of using the notation $X(k,s)$ it is used the notation $X(k, f)$ referring to the frequency instead of the scale:

$$w_E(f) = \frac{\langle \kappa(X(k, f), Y(k, f)) \rangle}{\sqrt{\langle \kappa(X(k, f), X(k, f)) \rangle}\sqrt{\langle \kappa(Y(k, f), Y(k, f)) \rangle}} \quad (18)$$

In [3] the mother wavelet used is the complex Morlet wavelet:

$$\psi(k) = Ae^{-k^2/2\sigma_0^2} e^{2i\pi f_0 k} \quad (19)$$

Where the width σ_0^2 and frequency f_o jointly determine the number of oscillations in the wavelet

It is important that before computing the wav-entropy coefficient $w_E(f)$, the time–frequency transforms: $X(k, f)$ and $Y(k, f)$ need to be normalized, for any frequency f, by subtracting the average (over time) and then dividing by the standard deviation (over time).

Both $w_E(f)$ and $c_E(f)$ will have values between 0 (if the signals are independent) to 1 (if both signals are identical).

4.2.1.6 Nonlinear correlation coefficient

As presented in [20] this measure is primarily a nonlinear regression coefficient, which describes the dependency of X on Y in a most general way without any direct specification of the type of relationship between them.

The underlying idea is that if the value of X is considered as a function of the value of Y, the value of Y given X can be predicted according to a nonlinear regression curve.

The variance of Y according to the regression curve is termed as the explained variance, since it is explained or predicted by the knowledge of X. The unexplained variance is estimated by subtracting the explained variance from the original one. The correlation ratio h^2 describes the reduction of variance of Y that can be obtained by predicting the Y values from those of X according to the regression curve, so the ratio obtained is: h^2 = (total variance -unexplained variance)/total variance.

Accordingly the nonlinear correlation coefficient is finally defined as:

$$h_{y|x}^2 = \frac{\sum_{k=1}^{N} y(k)^2 - \sum_{k=1}^{N} (y(k) - f(x_i))^2}{\sum_{k=1}^{N} y(k)^2} \quad (20)$$

Where $\mathrm{f}(\mathrm{x_i})$ is the linear piecewise approximation of the nonlinear regression curve, which is calculated in the following way: starting from the scatterplot of Y versus X, the values of X are divided in a certain number of bins, the middle point of each bin is calculated (p_i) and the average value of Y ($\mathrm{q_i}$) for each bin. The curve of regression is approximated by connecting the resulting points ($\mathrm{p_i}, \mathrm{q_i}$) by segments of straight lines, which is precisely: $\mathrm{f}(\mathrm{x_i})$.

The measure $h_{y/x}^2$ has values ranging from 0 (Y is completely independent of X) to 1 (Y is fully determined by X).

If the relationship between X and Y is linear, then: $h^2_{y/x} = h^2_{x/y}$

For a nonlinear relationship, then: $h^2_{y/x} \neq h^2_{x/y}$ and the difference $h^2_{y/x} - h^2_{x/y}$ indicates the degree of asymmetry of the nonlinear coupling.

4.2.1.7 Partial coherence

As presented in [20] it is an extension to the bivariate (with signals X and Y) coherence function (see section 4.2.1.2) by incorporating a third signal (Z) into the estimation.

The underlying point is to subtract linear influences from other processes to obtain the partial cross-spectrum between X and Y given all the linear information of Z.

Partial coherence is defined as:

$$S_{xy|z}(f) = S_{xy}(f) - S_{xz}(f)S_{zz}^{-1}(f)S_{yz}(f) \quad (21)$$

Similarly, one can obtain the partial auto-spectra $S_{xx/z}(f)$ and $S_{yy/z}(f)$

From these previous definitions, the squared partial coherence is estimated as follows:

$$\kappa^2_{xy|z}(f) = \frac{|\langle S_{xy|z}(f)\rangle|^2}{|\langle S_{xx|z}(f)\rangle||\langle S_{yy|z}(f)\rangle|} \quad (22)$$

The symbol $\langle . \rangle$ indicates average over M segments (similar to the normal coherence).
The partial coherence $\kappa_{xy/z}(f)$ can be represented as the fraction of coherence between X and Y that is not shared with Z.

If Z contributes to the linear interdependence between X and Y, then the partial coherence $\kappa_{xy/z}(f)$ will be smaller than the ordinary coherence $c_{xy}(f)$. However, it must be noted that partial coherence is based on the assumption of linearity, so any failure in its reduction might be also caused by nonlinear interaction between signals.

4.2.2 Granger Causality

As referred in [20] Norbert Wiener, defined causality in a statistical framework as follows: "for two simultaneously measured signals, if one can predict the first signal better by incorporating the past information from the second signal than using only information from the first one, then the second signal can be called causal to the first one "(Wiener, 1956).
Using these ideas, Granger proposed a way to obtain a measure of causality between two signals X and Y by comparing the prediction performance of a univariate and bivariate AR model of the two signals. For the univariate model one obtains:

$$x(n) = \sum_{k=1}^{p} a_{xk} x(n-k) + u_x(n)$$

$$y(n) = \sum_{k=1}^{p} a_{yk} y(n-k) + u_y(n) \quad (23)$$

And the bivariate:

$$x(n) = \sum_{k=1}^{p} a_{xyk} x(n-k) + \sum_{k=1}^{p} b_{xyk} y(n-k) + u_{xy}(n)$$

$$y(n) = \sum_{k=1}^{p} a_{yxk} y(n-k) + \sum_{k=1}^{p} b_{yxk} x(n-k) + u_{yx}(n) \quad (24)$$

Where u_x, u_y, u_{xy}, u_{yx} are the model errors.

The prediction performance for both models can be assessed by the variances of the prediction errors:

$$V_{X|X\_} = \mathrm{var}(u_x) \text{ and } V_{Y|Y\_} = \mathrm{var}(u_y) \text{ for univariate AR model}$$
$$V_{X|X\_,Y\_} = \mathrm{var}(u_{xy}) \text{ and } V_{Y|Y\_,X\_} = \mathrm{var}(u_{yx}) \text{ for bivariate AR model}$$

Where $\mathrm{var}(.)$ indicates the variance operator.

$V_{X/X\_}$ indicates predicting X by its past values alone and $V_{X/X\_,Y\_}$ indicates predicting X by past values of X and Y.

If $V_{X/X\_,Y\_} < V_{X/X\_}$, then Y causes X in the sense of Granger causality, because knowing Y reduces the prediction variance of X.

The Granger causality of Y to X can be quantified as:

$$G_{Y \to X} = \ln\left(\frac{V_{X/X\_}}{V_{X/X\_,Y\_}}\right) = \ln\left(\frac{\mathrm{var}(u_x)}{\mathrm{var}(u_{xy})}\right) \quad (25)$$

Granger causality is an asymmetric measure.

This theory can be extended to a multivariate signal, applying an MVAR model instead of an AR model [3].

If one considers m signals, they can be expressed as a vector $x(n) = (x_1(n), \ldots x_m(n))^T$ where n is the time index. The MVAR model is:

$$x(n) = \sum_{l=1}^{p} A(l) x(n-l) + e(n) \quad (26)$$

This expression can be converted to:

$$e(n) = \sum_{l=0}^{p} A(l) x(n-l) \quad (27)$$

Where $A(0) = I$ and $A(l) = -A(l)$ for $l > 0$, having a final compact expression.

Equation (27) can be transformed to the frequency domain:

$$E(f) = A\ (f)X(f)$$
$$X(f) = A^{-1}(f)E(f) = H(f)E(f) \quad (28)$$

$A(f)$ is calculated applying the Fourier transform to equation (27), and $H(f)$ can be considered as a transfer function.

As an example, it is presented a bivariate scenario [29]; in this example the Fourier transform of matrix A is obtained:

$$X_1(t) = \sum_{j=1}^{p} A_{11,j} X_1(t-j) + \sum_{j=1}^{p} A_{12,j} X_2(t-j) + E_1(t)$$

$$X_2(t) = \sum_{j=1}^{p} A_{21,j} X_1(t-j) + \sum_{j=1}^{p} A_{22,j} X_2(t-j) + E_2(t)$$

Fourier transform gives:

$$\begin{pmatrix} A_{11}(f) & A_{12}(f) \\ A_{21}(f) & A_{22}(f) \end{pmatrix} \begin{pmatrix} X_1(f) \\ X_2(f) \end{pmatrix} = \begin{pmatrix} E_1(f) \\ E_2(f) \end{pmatrix}$$

Where:

$$A_{lm}(f) = \delta_{lm} - \sum_{j=1}^{p} A_{lm}(j) e^{(-i2\pi f j)}$$

$$A_{lm}(j) = A_{lm,j}$$

$$\delta_{lm} = 0 \quad (l = m)$$

$$\delta_{lm} = 1 \quad (l \neq m)$$

The delays (j) introduced in $X_1(t-j)$ are transformed to a shift in phase in $e^{-ij2\pi f}$. So the Fourier transform of A takes into account the values of $A_{lm}(j)$ and the delay associated to each value in a single complex number. In this way matrix $A(f)$ contemplates the strength of dependence and delays in a single point.

And the power spectrum matrix of the signal is defined as:

$$S(f) = \mathrm{E}[X(f)X(f)^{\dagger}] = H(f)\ V\ H^{\dagger}(f) \quad (29)$$

Where V is the covariance matrix of $e(n)$ and $H(f)^{\dagger}$ is the hermitian conjugate of $H(f)$

The Granger causality measures to detect synchronization are defined in terms of the matrices: $A, H\ and\ S$. These measures are applicable to multivariate signals.

The most common Granger causality measures are defined in the following sections.

4.2.2.1 Partial coherence

Partial coherence is defined [3]:

$$C_{ij}(f) = \frac{M_{ij}(f)}{\sqrt{M_{ii}(f)}\sqrt{M_{jj}(f)}} \in \mathbb{C} \quad (30)$$

Where $M_{i,j}$ is a minor (i, j) of S, that is, the determinant of S with row i and column j removed.
$C_{ij}(f)$ describes the amount of in phase components in signals i and j at the frequency f when the influence (i.e., linear dependence) of the other signals is statistically removed.

This measure is symmetric, $C_{i,j} = C_{j,i}$ the following ones are asymmetric or directed

4.2.2.2 Directed transfer function (DTF)

Defined as [3]:

$$\gamma_{ij}^2(f) = \frac{|H_{ij}(f)|^2}{\sum_{j=1}^{m} |H_{ij}(f)|^2} \in [0, 1] \quad (31)$$

Where $\gamma_{ij}^2(f)$ quantifies the fraction of inflow causality to channel i coming from channel j.

4.2.2.3 Full frequency directed transfer function (ffDTF)

Defined as [3]:

$$F_{ij}^2(f) = \frac{|H_{ij}(f)|^2}{\sum_f \sum_{j=1}^{m} |H_{ij}(f)|^2} \in [0, 1] \quad (32)$$

It is a variation of $\gamma_{ij}^2(f)$ with a global normalization in frequency

4.2.2.4 Partial directed coherence (PDC)

This is an extension of Partial Coherence [3]:

$$P_{ij}(f) = \frac{\tilde{A}_{ij}(f)}{\sqrt{\sum_{i=1}^{m} |\tilde{A}_{ij}(f)|^2}} \in \mathbb{C} \quad (33)$$

Where P_{ij} represents the fraction of outflow causality from channel j to channel i.

4.2.2.5 Direct directed transfer function (dDTF)

Defined as [3]:

$$\chi_{ij}^2(f) = F_{ij}^2(f) C_{ij}^2(f) \in [0, 1] \quad (34)$$

This is non-zero if the connection between channels i and j is causal and direct.

4.2.2.6 MVAR coherence

Defined a [3]:

$$K_{ij}(f) = \frac{S_{ij}(f)}{\sqrt{S_{ii}(f)}\sqrt{S_{jj}(f)}} \in \mathbb{C} \quad (35)$$

Where $K_{ij}(f)$ describes the amount of in-phase components in signals i and j at the frequency f.

The squared magnitude $\left|K_{ij}(f)\right|^2$ is an alternative to the magnitude square coherence function $c(f)$ (Eq.(4)).

The argument of $K_{ij}(f)$, is an alternative to the phase coherence function $\phi(f)$ (Eq.(5)).

4.2.3 Phase synchronization

Phase synchronization [1][3]refers to the relation between the instantaneous phases of two signals. Phase synchronization of coupled systems is defined as the appearance of a relationship between their phases, while the amplitudes can remain non-correlated.

The instantaneous phase of a signal can be extracted by several methods, which are presented in the following section, continuing with the different techniques available to detect the phase difference between two or more signals.

4.2.3.1 Phase extraction

4.2.3.1.1 Hilbert transform

The instantaneous phase can be obtained by computing the analytic signal $\xi(t)$ of signal x(t):

$$\xi(t) = x(t) + i\tilde{x}(t) = A(t)\exp[i\phi(t)] \quad (36)$$

Where $x(t)$ is the Hilbert transform of the signal, and $x(t)$ is the signal [20].

$$\tilde{x}(t) = \frac{1}{\pi}\mathrm{P.V.}\int_{-\infty}^{\infty}\frac{x(\tau)}{t-\tau}d\tau \quad (37)$$

and P.V. indicates that the integral is taken in the sense of the Cauchy principal value.

In equation (36), the instantaneous amplitude A(t) and the instantaneous phase $\phi(t)$ of the signal x(t) are uniquely defined by this function.

Therefore, the instantaneous phase can be obtained as:

$$\phi(t) = \arctan(\tilde{x}(t))/x(t)) \quad (38)$$

4.2.3.1.2 Wavelet transform

The instantaneous phase can be obtained from the wavelet transform of signal x(t).

Then the wavelet coefficients of the signal x(t) at time τ and frequency f are defined as [35] :

$$W_x(\tau, f) = \int_{-\infty}^{\infty} x(t) \cdot \Psi^*_{\tau,f}(t)dt \quad (39)$$

In this definition $\Psi^*_{\tau,f}(t)$ is the complex conjugate of the Morlet wavelet defined as:

$$\Psi_{\tau,f}(t) = \sqrt{f} \cdot \exp[i2\pi f(t-\tau)] \cdot \exp[-\frac{(t-\tau)^2}{2\sigma^2}] \quad (40)$$

Where $\Psi_{\tau,f}(t)$ is the product of a sinusoidal wave at frequency f and a Gaussian function centered at time τ and a standard deviation σ proportional to the inverse of f.

The instantaneous phase can be expressed as:

$$\phi_x(\tau, f) = \arg[W_x(\tau, f)] \quad (41)$$

The instantaneous phase obtained in this way depends on the frequency.

4.2.3.1.3 Value crossing

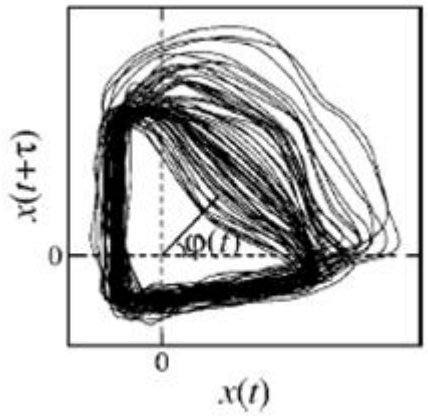
The phase is computed considering each time that the signal passes a threshold value in a direction. Every time this crossing happens the phase is increased in 2π. Between successive value crossings, the phase is increased in a linear way [27]:

$$\phi(t) = 2\pi \frac{t - t_k}{t_{k+1} - t_k} + 2\pi k, \; t_k \le t < t_{k+1} \quad (42)$$

Where t_k represent the times of crossing.

4.2.3.1.4 Phase portrait

It is the graph obtained when using the values of the signal and a shifted version in a 2-D graph:



Where $x(t+\tau)$ is a lag version of the signal.

The instantaneous phase is defined as:

$$\phi = \arctan \frac{x(t+\tau)}{x(t)} + 2\pi k \quad (43)$$

4.2.3.1.5 Instantaneous frequency

Once the instantaneous phases of two signals are obtained, the instantaneous frequency can be calculated as the derivative of the phases.

Synchronization is established in the time intervals when the ratio of frequencies is 1: $f_1 / f_2 = 1$

4.2.3.2 Phase synchronization detection

Once the phase for each signal is obtained, there are several methods to detect the synchronization between them:

4.2.3.2.1 Generalized phase difference

The generalized phase difference between two signals is [1]:

$$\varphi_{n,m} = n\phi_1 - m\phi_2 \quad (44)$$

Where n and m are integers.

That allows a fractional relationship between the instantaneous frequencies.

There is no way to anticipate the possible values of n and m, so a try an error exercise is needed. The usual values for n and m are n=m=1.

The presence of n:m phase synchronization is defined by the condition:

$$\left|\varphi_{n,m}\right| < cte \quad (45)$$

4.2.3.2.2 Relative phase

As presented in [26] , the phase difference is calculated as:

$$\varphi_1(t) - \varphi_2(t) = \arctan \frac{\tilde{s}_1(t)s_2(t) - s_1(t)\tilde{s}_2(t)}{s_1(t)s_2(t) + \tilde{s}_1(t)\tilde{s}_2(t)} \quad (46)$$

Where $\tilde{s}$ is the Hilbert Transform

In case of using the wavelet transform to obtain the phase, the phase difference can be defined as [35]:

$$\exp[i(\phi_y(f,\tau) - \phi_x(f,\tau))] = \frac{W_x(\tau,f)W_y^*(\tau,f)}{|W_x(\tau,f)|\,|W_y(\tau,f)|} \quad (47)$$

The phase difference ranges from 0 (perfect phase synchrony) to 2π (anti-phase synchrony), and it is dependent on time.

4.2.3.2.3 Phase index

As presented in [25][3] , the phase index is:

$$\gamma = |\langle e^{i(n\phi_x - m\phi_y)}\rangle| \in [0, 1] \quad (48)$$

n and m are integers and usually n=m=1. And $\langle . \rangle$ stands for average over time for the difference between both signals.

By construction, γ will be zero if the phases are not synchronized at all and will be one when the phase difference is constant implying perfect synchronization.

The key feature of γ is that it is only sensitive to phases, irrespective of the amplitude of each signal.

4.2.3.2.4 Phase index-mean absolute value

This index (developed in this work) is similar to the previous index, but the average is done only on the phase values (not on the complex values in the circle):

$$\gamma = \left\langle |\varphi_1(t) - \varphi_2(t)| \right\rangle_t \quad (49)$$

It measures the mean absolute value of phase difference. It provides a measure of total phase separation.

φ_1 and φ_2 are unbounded (Matlab unwrap command)

4.2.3.2.5 Phase index based on Entropy

As presented in [36] [22] [100] it can be defined another phase synchronization measure from the Shannon entropy of the distribution of $\phi^H_{x,y}(t) = \phi^H_x(t) - \phi^H_y$ (the instantaneous Hilbert phase difference).

The range of $\phi' = \phi^H_{x,y}(t) \bmod 2\pi$ is divided into M bins.

Let p_k be the probability that ϕ' is in bin k at any random time.

Then, γ_{Sh} is defined as:

$$\gamma_{Sh} = \frac{S_{\max} - S}{S_{\max}}; \qquad S = -\sum_{k=1}^{M} p_k . \ln p_k; \quad S_{\max} = \ln M \tag{50}$$

It ranges from zero for a uniform distribution of $\phi^H_{x,y}(t)$, to one if the distribution is a delta function.

The advantage over γ is that γ can underestimate phase synchronizations when the distribution of $\phi^H_{x,y}(t)$ has more than one peak.

4.2.3.2.6 Phase index-multivariate

This index (developed in this work) is applicable to multivariate signals and is computed via the instantaneous Hilbert phase of each signal.

To start with we assume having a set of N interconnected nodes (neurons). The activity of the N nodes produce N signals which we consider to be sampled at regular time intervals producing T samples per signal. Ending up with an activity matrix X of dimension $\mathrm{N}\times\mathrm{T}$ with the sampled values of all signals:

$$\mathrm{X} = \begin{pmatrix} x_1^1 & \cdots & x_1^T \\ \vdots & \ddots & \vdots \\ x_N^1 & \cdots & x_N^T \end{pmatrix} = \left(\overrightarrow{x^1}, \overrightarrow{x^2}, \ldots, \overrightarrow{x^T} \right) \tag{51}$$

Activity matrix X can be also represented as an array of column vectors $\overrightarrow{x^i}$, each column vector having the activity of all nodes at time i .

Computing the Hilbert transform of all signals in matrix X will provide matrix $\mathrm{X_H}$ (Hilbert transform along rows of X), from which we can compute the instantaneous matrix of signals phases Φ :

$$\mathrm{X_H} = \begin{pmatrix} x_{1H}^1 & \cdots & x_{1\,H}^T \\ \vdots & \ddots & \vdots \\ x_{NH}^1 & \cdots & x_{NH}^T \end{pmatrix};$$

$$\Phi = \begin{pmatrix} \tan^{-1}\frac{x_{1H}^1}{x_1^1} & \cdots & \tan^{-1}\frac{x_{1\,H}^T}{x_1^T} \\ \vdots & \ddots & \vdots \\ \tan^{-1}\frac{x_{NH}^1}{x_N^1} & \cdots & \tan^{-1}\frac{x_{NH}^T}{x_N^T} \end{pmatrix} = \begin{pmatrix} \phi_1^1 & \cdots & \phi_1^T \\ \vdots & \ddots & \vdots \\ \phi_N^1 & \cdots & \phi_N^T \end{pmatrix} = \left[\vec{\phi^1},\ \vec{\phi^2}..\ ,\vec{\phi^T}\right]$$

Considering the phase values in each column vector $\vec{\phi^i}$, we build its histogram, assigning the range of phase values $[0, 2\pi]$ into M bins.

Finally we will have T histograms (one per column vector $\vec{\phi^i}$) where in bin k of histogram i one will have the total number of times that any signal phase at time i has a value between $\frac{(\mathrm{k}-1)}{M}2\pi$ and $\frac{\mathrm{k}}{M}2\pi$ radians.

Normalizing the histograms to have a total area of 1, we end up with T probability mass functions p_t , one per time increment: $p_t(\mathrm{k}),\ 0 \le \mathrm{k} < \mathrm{M}$ (on M bins).

If the distribution of the probability mass functions is similar to a uniform distribution implies that the signals are not synchronized and the more the distribution is departed from the uniform distribution implies that the signals are more synchronized.

We define a synchronization index based in applying a measure to each probability distribution $p_t(\mathrm{k})$ to have a single number that represent how close is the distribution to a uniform distribution. In this form, the synchronization index can be defined in two ways:

- As the Kullback-Leibler divergence between the distributions and a uniform distribution $U(0,2\pi)$:

  $$\gamma_m(t) = KL(p_t(k), U(0,2\pi)) \quad (52)$$

- As the phase index-entropy comparing the entropy of the distribution to the uniform distribution which entropy is: $\ln(\mathrm{M})$

  $$\gamma_m(t) = \frac{\ln(M) - \sum_k p_t(k)\ln p_t(k)}{\ln(M)} \quad (53)$$

The above presented synchronization index is a point-wise index (there is one for each specific time). The final global index is defined as the average value of $\gamma_m(t)$.

This index can be used, as well to established the difference on the synchronization status of two different ensembles $\{x_1,...x_i,..x_n\}$ and $\{y_1,...y_i,..y_n\}$, just computing the difference

between their γ_m, or, otherwise computing the Kullback-Leibler divergence between their corresponding phase distributions.

4.2.3.2.7 Global Field Synchronization (GFS)

The above mentioned indexes are bivariate, while this index is multivariate [3].
This index is based on Principal Component Analysis in the complex plane.
To start is necessary to obtain the Fourier transform of the n signals. Then, a covariance matrix is calculated using the real and imaginary parts of the Fourier transform of the signals.
The difference between the first and second normalized eigenvalues of the covariance matrix gives the GFS:

$$\mathrm{GFS}(f) = \lambda_1 - \lambda_2 \quad (54)$$

Correlated phases will have a GFS close to one while uncorrelated phases will have a value close to zero.

4.2.3.2.8 Oscillators ensemble / Mean field

In [20] are presented two ways to use the mean field concept to identify synchronization.

One way is to compute the average of instantaneous phases (mean field applied to phases) of all signals of a multivariate ensemble, and instead of calculating the one-to-one bivariate synchronization index between every pair of signals, to calculate the bivariate index between the phase of each signal and the mean field phase. In this way, it is possible to establish the level of synchronization of each signal with the overall phase, performing only n (number of signals) calculations, instead of $\frac{n(n-1)}{2}$.

Another way is to use the classical connotation of mean field [1] to assume that every signal source (e.g. neuron) is being driven by a single signal formed by the aggregated average of all other signals. As in the former case, the synchronization between each signal and the mean field signal can be used as an easier intermediate state to identify global synchronization. Moreover, in cases of synchronization of an important number of elements of the ensemble, macroscopic oscillations of the mean field will appear, being an additional method to identify synchronization.

4.2.4 State-space analysis

State space based synchrony [3] tries to obtain a measure of synchronization by studying the comparative behavior of the signals in a reconstructed state-space.

It is assumed that the signals are the manifestation of an (unknown) deterministic, potentially high-dimensional, non-linear dynamical system:

$$ds/dt = f(s) \quad (55)$$

Where $s \in R^q$ is the state of the system, and q and f are usually unknown.

State space based synchrony measures assess the interdependence between two signals x and y by comparing local neighborhoods of their state space reconstructed mappings. More precisely, they quantify how local neighborhoods in one manifold are transformed into neighborhoods in the other manifold.

If the size of a neighborhood on one manifold is mapped unto a much larger neighborhood on the other manifold, the signals x and y are only weakly synchronized; on the other hand, if it is mapped to a neighborhood of about the same size, the two signals x and y are strongly coherent.

In the following subsection are presented several algorithms based on these methods.

We will consider that all signals are normalized to have zero mean and standard deviation equal to one.

4.2.4.1 Non-linear interdependence (embedding)

In order to reconstruct the system modeled in equation (55) from data coming from signal X, one considers delay vectors $X(k) = \left(x(k), x(k-\tau), \ldots, x(k-(m-1)\tau)\right)^T$ of the signal X, where m is the embedding dimension and τ denotes the time lag. If some generic conditions are fulfilled, the delay vectors lie on a smooth manifold ("mapping") in sub-space R^m [30]. That implies one can use the transformed distances to neighborhoods in this transformed sub-space in order to establish interdependence between two signals.

As presented in [22][3] the non-linear interdependence measures presented in this section are based in using the delay vectors that represent the original signal in an embedded sub-space of the original state-space.
These non-linear interdependence measures correspond to the computation of three measures: S^k, H^k, N^k.
In order to obtain these measures, it is necessary to compute previously the following additional measures:

$R_k^M(X)$: It is the average squared Euclidian distance for each delay vector X(k) to its M nearest neighbors.

$R_k^M(X/Y)$: It is the mean squared Euclidian distance between delay vector X(k) and the X-delay vectors corresponding to the M nearest neighbors of Y(k). It is referred to as the Y-conditioned mean squared Euclidian distance.

$R_k(X)$: It is the mean squared Euclidian distance between X(k) and the other points X(ℓ) (ℓ≠k).

The non-linear independence measures: S^k, H^k, N^k are defined in terms of those measures in the following way:

$$S^k(X|Y) = \frac{1}{N}\sum_{k=1}^{N} \frac{R_k^M(X)}{R_k^M(X|Y)}$$

$$H^k(X|Y) = \frac{1}{N}\sum_{k=1}^{N} \log\frac{R_k(X)}{R_k^M(X|Y)}$$

$$N^k(X|Y) = \frac{1}{N}\sum_{k=1}^{N} \frac{R_k(X) - R_k^M(X|Y)}{R_k(X)} \quad (56)$$

S^k can have values between zero and one, with values close to zero indicating independence between X and Y, and values close to one indicating synchronization.

H^k is zero if X and Y are completely independent, while it is positive if nearness in Y implies also nearness in X (in state space) for equal time partners. It would be negative if close pairs in Y would correspond mainly to distant pairs in X.
H^k only indicates that X and Y are independent, but does not prove it. This behavior is different to mutual information where a value of zero indicates independence. H^k is also more sensitive to weak dependence that mutual information, this can be useful in applications.

N^k is a normalized version of H^k.

The measure H^k is believed to be more robust against noise than S^k.

In relation to the number M of nearest neighbors, it is set [22] equal to 10.

The measures S^k, H^k, N^k are asymmetric that means for example that:
$S^k(X/Y) \neq S^k(Y/X)$

This asymmetry is very important at it provides information on directionality of dependence.

In the following figure (Figure 4.2.4.1-1) it is presented the basic idea behind non-linear interdependence measures. The size of the neighborhood in one of the systems, say X, is compared with the size of its mapping in the other system. The example shows a Lorenz system driven by a Rossler with zero coupling (upper case diagram) and with strong coupling (lower diagram). Below each attractor, the corresponding time series is shown.

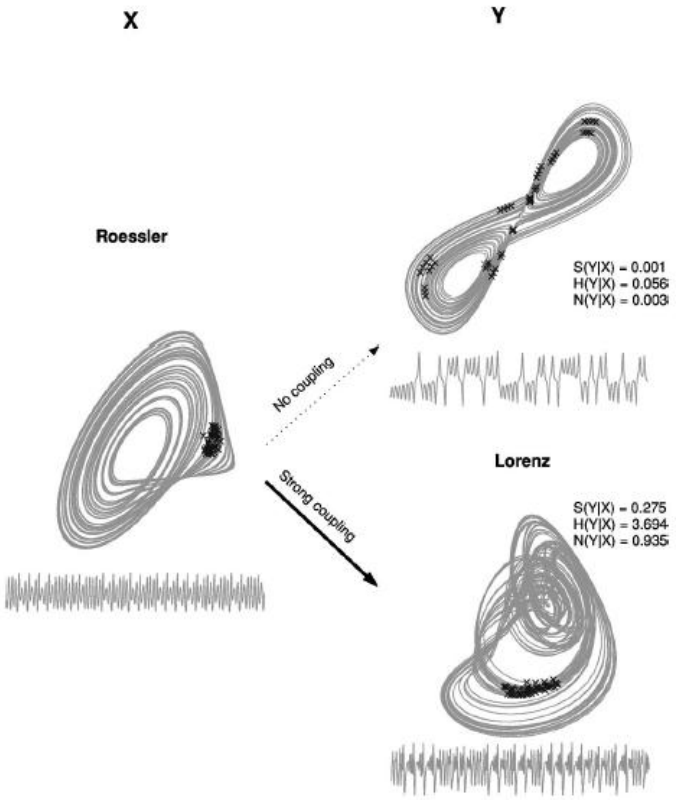


Figure 4.2.4.1-1 Basic idea of the nonlinear interdependence measures

Code is available (Matlab) at: http://www.vis.caltech.edu/~rodri/programs/synchro.m

4.2.4.2 Mean conditional dispersion

As presented in [19][37] a similar method as presented in section 4.2.4.1 exists.

The basic idea of the approach is that if two systems X and Y are coupled, then, due to the coupling, the states of X which are close to each other in the X reconstruction space should correspond to states of Y which are close neighbors in the Y reconstruction space.

So one starts from the assumption that if two states of the X system are close neighbors, $\|x_i - x_j\| < \varepsilon$, then it should be expected that the corresponding states of the Y system are also close, $\|y_i - y_j\| \approx \varepsilon$. If both systems are not related at all, the distance of the states y_i and y_j equals (on average) the mean distance. In order to measure this relation, it is introduced the so-called mean conditional dispersion:

$$\sigma_{xy}(\varepsilon) = \sqrt{\frac{\sum_{i \neq j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 \Theta(\varepsilon - \|\mathbf{x}_i - \mathbf{x}_j\|)}{\sum_{i \neq j} \Theta(\varepsilon - \|\mathbf{x}_i - \mathbf{x}_j\|)}} \tag{57}$$

where Θ is the Heaviside function with $\Theta(z > 0) = 1$ and $\Theta(z \leq 0) = 0$.

If the x and y systems are related then $\sigma_{xy}(\varepsilon)$ should decrease with decreasing ε. If they are not related then $\sigma_{xy}(\varepsilon)$ will not depend on ε. The difference between both cases should thus become visible in the limit of $\varepsilon \to 0$.

This measure is not symmetric, and that can be used to detect coupling directions.

4.2.4.3 Omega Complexity

Omega complexity Ω is a measure derived from principal component analysis (PCA) [3]

Considering that we have n signals $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_n$, with a covariance matrix $\mathrm{C} \in \mathrm{R}^{n \times n}$. Omega complexity is defined in terms of the normalized eigenvalues λ_i of C, as:

$$\Omega = \exp(-\sum_{i=1}^{n} \lambda_i \log \lambda_i) \quad (58)$$

Where the normalized eigenvalues λ_i of C are computed as:

$$\lambda_i = \frac{\lambda'_i}{\sum_{j=1}^{n} \lambda'_j} \quad (59)$$

Where λ'_i is the non-normalized eigenvalue of the covariance matrix C.

Omega complexity Ω is the entropy of the normalized eigenvalue distribution.

If the signals $\{\mathrm{X}_i\}$ are identical, all eigenvalues except one are zero, and Ω=1. On the other hand, if the signals $\{\mathrm{X}_i\}$ are independent, all n eigenvalues are identical, and hence Ω=n. This is because the covariance matrix of n independent signals will be diagonal with all the values of the diagonal equal to 1.

Therefore, omega complexity Ω is a dissimilarity measure.

4.2.4.4 S estimator

The S-estimator [14][3][13] extends omega complexity to state space embedded signals.

Considering that we have n signals $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_n$, first one constructs m-dimensional delay vectors for each signal, and then one computes the covariance matrix $\mathrm{C} \in \mathrm{R}^{mn \times mn}$ for the n sequences of delay vectors.

The S-estimator is defined in terms of the normalized eigenvalues λ_i of C, as:

$$S_{est} = 1 + \frac{\sum_{i=1}^{nm} \lambda_i \log \lambda_i}{\log(nm)} \quad (60)$$

Where the normalized eigenvalues λ_i of C are computed as:

$$\lambda_i = \frac{\lambda'_i}{\sum_{j=1}^{n} \lambda'_j} \quad (61)$$

Where λ'_i is the non-normalized eigenvalue of the covariance matrix C.

The numerator in the second term of equation (60) represents the negative entropy of the normalized eigenvalues λ_i, and the denominator is the negative entropy of the uniform distribution (which has maximum entropy). That means that $S_{est} \in [0,1]$.

If the signals $Xi(k)$ are statistically independent, all normalized eigenvalues tend to be equal to $1/mn$, and as a result, S-estimator approaches 0 (diagonal C, actually it is the identity matrix as we assumed all signals are normalized). On the contrary, if the signals are well synchronized, only a few numbers of eigenvalues will be different of zero, and as a result, S-estimator is then close to 1.

Code is available (Matlab) at: http://aperest.epfl.ch/docs/software.htm

4.2.5 Information theory based

4.2.5.1 Mutual information

Mutual Information is one of the best known measures of dependence of two random variables X and Y [31][20].
It is defined in terms of the Shannon Entropy of X and Y: H(X) and H(Y), and the joint Entropy of X and Y: H(X, Y).

It is defined as:

$$I(X;\ Y) = H(X) + H\ (Y) - H(X,\ Y) \quad (62)$$

It can also be defined as:

$$I(X;Y) = \sum_X \sum_Y p(x,y) \log \left[\frac{p(x,y)}{p(x)p(y)} \right] \quad (63)$$

Mutual information measures the information that X and Y share: it measures how much knowing one of these variables reduces uncertainty about the other.
It also measures the dependence between the variables. In equation (63) if X and Y are independent, then $p(x,y) = p(x).p(y)$, implying then I(X; Y) is equal to zero.

Mutual information is nonnegative (I(X; Y) ≥ 0) and symmetric (I(X; Y) = I(Y; X)).

Mutual information is usually done on the time domain, but it can also be defined on the time-frequency domain [3][90]. To do that, one starts from the normalized spectrograms:

$$C_x(k,f) = \frac{|X(k,f)|^2}{\sum_{k,f} |X(k,f)|^2} \quad (64)$$

The normalized spectrograms can be treated as probabilities distributions and define mutual information on these distributions, arriving to:

$$I_W\left(C_x, C_y, C_{xy}\right) = \sum_{k,f} C_{xy}(k,f) \log \frac{C_{xy}(k,f)}{C_x(k,f) C_y(k,f)} \quad (65)$$

Where the (normalized) cross time–frequency distribution of x and y is defined as:

$$C_{xy}(k,f) = \frac{|X(k,f)Y^*(k,f)|}{\sum_{k,f} |X(k,f)Y^*(k,f)|} \quad (66)$$

4.2.5.2 Multivariate mutual information

As presented in [8] mutual information can be extended to n variables $X_1, X_2, \ldots, X_n$ as:

$$I(X_1 X_2 \ldots X_N) = \sum I(X_i) - \sum I(X_i; X_j) + \ldots \ldots (-1)^n I(X_1; X_2; \ldots; X_N) \quad (67)$$

In equation(67), the summations are taken over all combinations of subscripts.

Equation(67) can also be expressed in terms of the Entropy:

$$I_N(X_1; X_2; \ldots X_N) = (H(X_1) + H(X_2) \ldots + H(X_N)) - \ldots + \ldots (-1)^{N-1} H(X_1, X_2, \ldots X_N)$$

(68)

Alternatively, I_N can be seen as:

$$I_N(X^N) = I_{N-1}(X^{N-1}) - I_{N-1}(X^{N-1}|X_N) \quad (69)$$

Where $X^k = (X_1; X_2; \ldots X_k)$. That is, it can be considered as the gain (or loss) in the information transmitted between a set of variables due to additional knowledge of an extra variable.

It can be positive or negative. This can be understood with an example: we can consider three variables X, Y and Z, where variables X and Y are independent when Z is not known, but become dependent given Z, so, in this case the mutual information of the three will be negative.

4.2.5.3 Transfer entropy

As presented in [32][33], Transfer Entropy is an extension of Mutual Information that tries to provide the directional relationship between two random variables. Mutual information doesn't provide this directionality as it is a symmetric measure.

Considering two signals X and Y, one defines an entropy rate which is the amount of additional information required to obtain the value of the next observation of one of the variables:

$$h_1 = -\sum_{x_{n+1}} p(x_{n+1}, x_n, y_n) \log_a p(x_{n+1}|x_n, y_n) \quad (70)$$

We can as well consider that the value of observation x_{n+1} is not dependent on the current observation y_n, arriving to another entropy rate:

$$h_2 = -\sum_{x_{n+1}} p(x_{n+1}, x_n, y_n) \log_a p(x_{n+1}|x_n) \quad (71)$$

We define transfer entropy as the difference between h_1 and h_2. The difference between the quantity h_1 that represents the entropy rate for the two systems, and h_2 that represents the entropy rate assuming that x_{n+1} is independent of y_n:

$$\begin{aligned} h_2 - h_1 &= -\sum_{x_{n+1},x_n,y_n} p(x_{n+1}, x_n, y_n) \log_a p(x_{n+1}|x_n) \\ &\quad + \sum_{x_{n+1},x_n,y_n} p(x_{n+1}, x_n, y_n) \log_a p(x_{n+1}|x_n, y_n) \\ &= \sum_{x_{n+1},x_n,y_n} p(x_{n+1}, x_n, y_n) \log_a \left(\frac{p(x_{n+1}|x_n, y_n)}{p(x_{n+1}|x_n)} \right) \end{aligned} \quad (72)$$

Actually this is the transfer entropy from Y to X, and it can be defined a similar transfer entropy from X to Y:

$$\sum_{y_{n+1},x_n,y_n} p(y_{n+1}, x_n, y_n) \log \left(\frac{p(y_{n+1}|x_n, y_n)}{p(y_{n+1}|y_n)} \right) \quad (73)$$

Considering Bayes rule, both transfer entropies can also be written as:

$$\sum_{x_{n+1},x_n,y_n} p(x_{n+1}, x_n, y_n) \log \left(\frac{p(x_{n+1}, x_n, y_n) \cdot p(x_n)}{p(x_n, y_n) \cdot p(x_{n+1}, x_n)} \right)$$

$$\sum_{y_{n+1},x_n,y_n} p(y_{n+1}, x_n, y_n) \log \left(\frac{p(y_{n+1}, x_n, y_n) \cdot p(y_n)}{p(x_n, y_n) \cdot p(y_{n+1}, y_n)} \right) \quad (74)$$

So, transfer entropy (an asymmetric measure) provides an indication of the flow of dependence between two variables.

Code is available (Matlab) at: http://code.google.com/p/transfer-entropy-toolbox/

4.2.6 Divergences methods

The divergences measures are measures that provide an assessment of distance between probability distributions [31][3], meaning how close are two probability distributions. They can be applied to distribution on time, or distributions defined on time and frequency.

They are not formally a distance, but can be used as one.

The definitions given in the following sections will consider as probability distributions the normalized spectrograms of the signals (equation(64)), but can be used for any kind of function with the properties of a probability distribution

4.2.6.1 Kullback–Leibler divergence

It is defined as [3][31] :

$$K\left(C_x, C_y\right) = \sum_{k,f} C_x(k,f) \log \frac{C_x(k,f)}{C_y(k,f)} \quad (75)$$

It is an asymmetric measure.
It can be done symmetric computing the mean between both measures:

$$K\left(C_x; C_y\right) = \frac{K\left(C_x, C_y\right) + K\left(C_y, C_x\right)}{2} \quad (76)$$

It is always positive and equal to zero if both distributions are equal.

4.2.6.2 Rényi divergence

It is defined as [3][7]:

$$D_\alpha\left(C_x, C_y\right) = \frac{1}{\alpha - 1} \log \sum_{k,f} \left[C_x(k,f)\right]^\alpha \left[C_y(k,f)\right]^{(1-\alpha)} \quad (77)$$

Where $\alpha \in [0,1]$ is the order of divergence. This measure tends to the Kullback-Leibler (KL) measure when α tends to 1.

It is an asymmetric measure and always positive (for positive values of alpha). It is equal to zero if both distributions are equal.

4.2.6.3 Jensen–Shannon divergence

It is defined as [3] :

$$J\left(C_x, C_y\right) = H\left(\frac{C_x + C_y}{2}\right) - \frac{H(C_x) + H\left(C_y\right)}{2} \quad (78)$$

Where H stands for the Shannon entropy.

It measures the deviation between the Shannon entropy of the mixture of values and the mixture of the entropies

It is a symmetric measure.

Its values are between [0-1]. It is a bounded measure. It is equal to zero if both distributions are equal.

4.2.6.4 Jensen–Rényi divergence

It is defined [3] in a similar way to the Jensen–Shannon divergence but using the Rényi entropy instead of the Shannon Entropy.

The Rényi entropy is defined as (using the normalized spectrogram):

$$H_\alpha(C_x) = \frac{1}{1-\alpha}\log\sum_{k,f}(C_x(k,f))^\alpha \quad (79)$$

Where $\alpha \in [0,1]$

Rényi entropy converges to Shannon entropy as α→1.

With these previous definitions, we arrive to the definition of Jensen–Rényi divergence as:

$$J_\alpha\left(C_x, C_y\right) = H_\alpha\left(\sqrt{C_x(k,f)C_y(k,f)}\right) - \frac{H_\alpha(C_x) + H_\alpha\left(C_y\right)}{2} \quad (80)$$

It is a symmetric measure.

It is equal to zero if both distributions are equal and its value is maximized when one of the distributions is degenerate (that is, its probability is concentrated in a single point).

It can be generalized to more than two distributions [91]:

$$JR_\alpha^\omega(p_1,\ldots,p_n) = R_\alpha\left(\sum_{i=1}^n \omega_i p_i\right) - \sum_{i=1}^n \omega_i R_\alpha(p_i)$$

Where $p_1, p_2, .., p_n$ are n probability distributions. $R_\alpha(p)$ is Renyi entropy. And $\omega = (\omega_1, \omega_2, ..., \omega_n)$ is a weight vector such that $\sum_{i=1}^n \omega_i = 1$ and $\omega_i \geq 0$.

It is a convex function on $p_1, p_2, .., p_n$ and is equal to zero when all the distributions are equal.

4.2.7 Frequency flows analysis (FFA)

As presented in [11][1] it is a multivariate method (global method section 4.3.1).

Frequency flows analysis (FFA) allows direct tracking and characterization of the non-stationary time-frequency dynamics of phase synchrony among groups of signals. It is based on the use of the one-to-one relationship between frequency locking and phase synchrony.

Phase synchrony implies identical instantaneous frequencies among synchronized signals, with possible time varying frequencies of synchronization. In this framework, synchronous groups of signals or neural assemblies can be identified as belonging to common frequency flows, and the problem of studying synchronization becomes the problem of tracking frequency flows.

Phase synchrony can be defined as the temporal adjustment of the rhythms of a pair of signals, evaluated within successive frequency bands fi, whereas the amplitudes can remain uncorrelated. The local stability of the phase adjustment within the band is then characterized over a window of time by means of various statistical dependence parameters (mean phase difference, circular variance, standard deviation, Shannon entropy, or mutual information). The frequency band decomposition can cover the totality of the relevant spectrum, thus constituting a time-frequency chart of synchrony for each pair of signal. Those time-frequency charts can be averaged across groups of pairs to provide a general map of the dynamics of synchronization through time.

The problems with the above methods are that the computation of global synchrony implies the computation of $n(n-1)/2$ relationships and the assumptions of stationarity of synchronization on the frequency band chosen to perform the analysis. Both problems are resolved with FFA.

When two signals are phase synchronized, their phase difference is constant which implies that the instantaneous frequencies of both signals are equal:

$$\Delta\phi_{1,2} = \phi_1 - \phi_2 = cte$$

$$\frac{d\Delta\phi_{1,2}}{dt} = \frac{d\phi_1}{dt} - \frac{d\phi_2}{dt} = \omega_1 - \omega_2 = 0 \quad (81)$$

These ideas can be seen more clear if one assumes that both signals correspond to two unit vectors rotating signals, and the idea can be extended to more than two signals, as can be seen in figure 4.2.7-1.

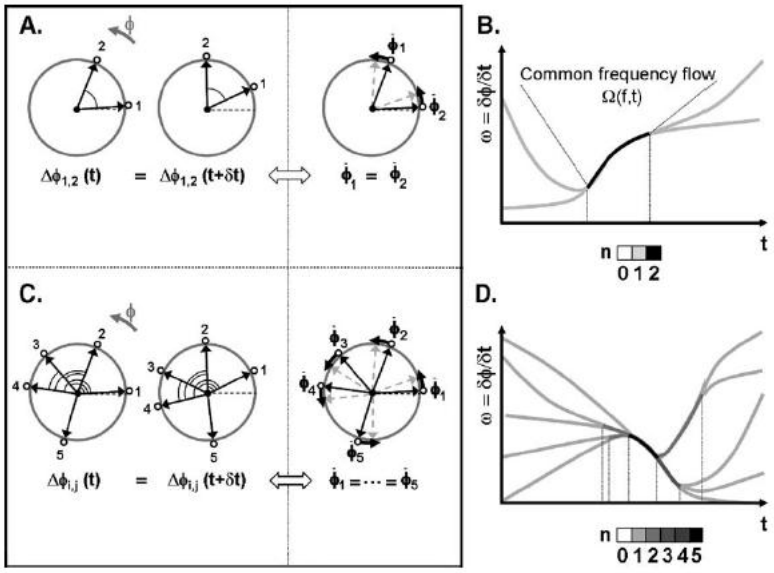


Figure 4.2.7-1. A. Phase synchronization of two unit vectors rotating signals corresponds to periods of equal instantaneous frequency. B. The same can be applied to more than two unit vectors rotating signals.

An important benefit obtained from using equal instantaneous frequency values as a measure of synchronization is that synchronization frequencies can change with time (as can be seen in figure 4.2.7-1). That implies that the signals will maintain phase difference in different frequencies over time (non-stationarity scenario).

In order to obtain the distribution of instantaneous frequencies of a signal, the method uses the analytic representation of the signal:

$$x(t) = \sum_i a_i(t) \cdot e^{j\phi_i(t)} \quad (82)$$

Then, a wavelet transform is applied to the analytic representation of the signal (using a Gabor wavelet), this transform is called an analytic wavelet transform (AWT) of the signal [11]:

$$Wx(u,s) = \int_{-\infty}^{+\infty} x(t)\ \psi_{u,s}^*(t)\mathrm{d}t \overset{\text{Parseval}}{=} \frac{1}{2\pi}\int_{-\infty}^{+\infty} \hat{x}(\omega) \cdot \hat{\psi}_{u,s}^*(\omega)\mathrm{d}\omega \quad (83)$$

Where ψ is the mother wavelet, which is taken as the Gabor wavelet. The Gabor wavelet is formed from a Gaussian function g as:

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)$$

$$g_{s,u,\xi}(t) = \sqrt{s}g\left(\frac{t-u}{s}\right) \cdot e^{i\xi t}$$

$$\psi_{u,s}(t) = g_{s,u,\xi}(t) \cdot e^{-i\xi u} \quad (84)$$

Where $\xi = \eta / s$ and η is the center frequency of the mother wavelet.

The AWT defines local time-frequency densities in boxes centered at (u, ξ).
The normalized energy density of the signal at (u, ξ) is:

$$\frac{\xi}{\eta} P_W x(u,\xi) = \frac{|Wx(u,s)|^2}{s} \quad (85)$$

Which is called normalized scalogram of x(t).
The points $(u, \xi(u))$ where the scalogram has its maxima are called wavelet ridges.
Where:

$$\xi(u) = \frac{\mathrm{d}\phi(u)}{\mathrm{d}t} \quad (86)$$

Then, it is possible to define a binary map, in the time-frequency plane, with ones where a significant instantaneous frequency has been found and zeros otherwise. The significant instantaneous frequency is taken for ridge values over a threshold. Such a map is usually called the ridgelet transform of the signal.

To extend these ideas to multivariate signals, one computes the ridgelet transform for each signal and latter does the summation of all them. The result is called the Instantaneous Frequency Histogram (IFH), reflecting the number of synchronous signals involved in each common frequency flow at a given time instant

As proven in [11], finally it can be defined an average interchannel correlation $\rho_{av}(t, f)$ as a function of time and frequency, and this correlation can be computed from the values of IFH, as:

$$\rho_{\mathrm{av}}(t,f) = \frac{\sum_{i,j \in \mathbf{W}(t,f)} \left(\mathbf{IFH}(i,j)^2 - \mathbf{IFH}(i,j) \right)}{n_{\mathrm{c}}(n_{\mathrm{c}} - 1) n_T} \quad (87)$$

Where **IFH** is the matrix containing the IFH of the signal, n_T is the number of time points spanned by the region and n_c is the number of signals.

A better way to compute equation (87) [11] is:

$$\rho_{\mathrm{av}} = \left(\frac{1}{n_{\mathrm{c}}(n_{\mathrm{c}} - 1) n_T(f)} \right) W(f) * \left(\mathrm{IFH}^2 - \mathrm{IFH} \right) \quad (88)$$

Where $W(f)$ is an adaptive window of integration and * is the convolution.

In the following figure (figure 4.2.7-2) is presented an example with 3 signals and 3 possible synchronization scenarios (named a, b and c). Scenario 'a' corresponds with the signals shown to the left. Scenarios 'b' and 'c 'correspond to other possible scenarios.

It can be seen the time-frequency regions where a multivariate synchronization (3 signals) is present. In the lower part is shown the computation of $\rho_{av}(t, f)$ using the convolution with a square integration window $W(f)$.

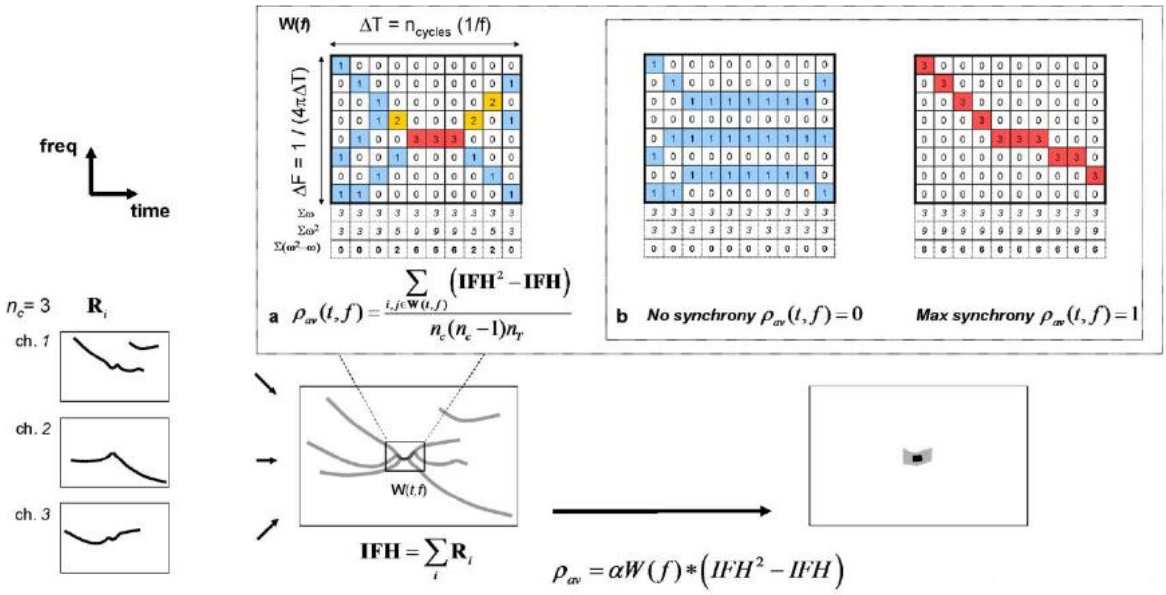


Figure 4.2.7-2. Measure of frequency locking based on the Instantaneous Frequency Histogram IFH.

The average interchannel correlation has values between [0-1], with a value of zero for no synchrony on the signals and a value of one for maximum synchrony.

4.2.8 Symbolic dynamics

Symbolic dynamics [43] consist of partitioning the state space of a dynamical system, discretizing it. Once discretized, each discrete value of the new (coarser) state space is associated with a symbol.

In this way, the path of a dynamical system can be represented as a string of symbols correspondent to the discrete values of the state space for which the path has moved.

The symbolic dynamics can be seen as a coarse granular projection of the original phase space.

This coarser grain transformation may be subject to loss of information resulting from the partition of state space in discretizing boxes, plus the effect of noise. However, the essential features (for example, the periodicity or chaotic behavior of a trajectory) are expected to be preserved in the resulting symbol sequences through an adequate phase space partition.

The following diagram (figure 4.2.8-1) shows the operational procedure for the symbolic dynamic analysis of time series

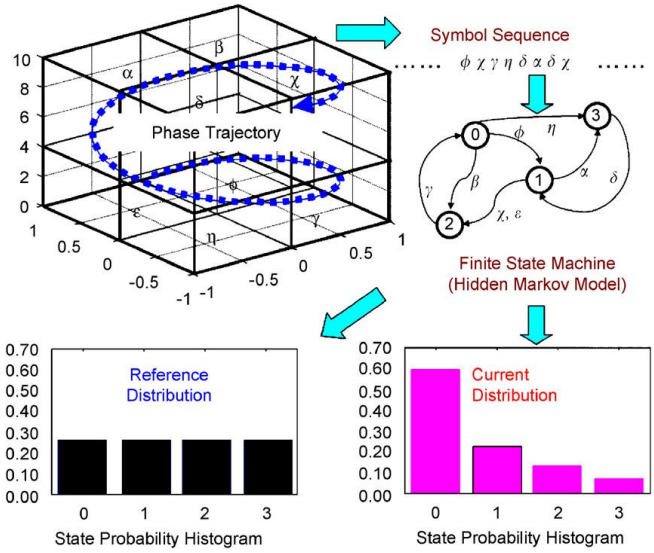


Figure 4.2.8-1 Dynamical system structure identification via symbolic time series analysis

As shown in the diagram, the state space is partitioned, and the trajectory of the dynamical system is moved to a sequence of symbols associated with the partitions visited. The behavior of the symbol sequence is translated to an HMM (Hidden Markov Model) representation, which models the transition probabilities between symbols.

Once the HMM model (the probability transition matrix) is obtained, it is possible to map the different dynamical behaviors of the system to changes in the HMM transition frequencies. Besides the transition matrix, another way to identify an HMM model is by its state probability vector: which is the left eigenvector corresponding to the unique unity eigenvalue of the transition matrix, that is, the stationary state probability vector of the system.

In more detail, this technique resolves the following issues:

• Partition of the state space: Not all partitions are similar. For example, we are interested in doing smaller partitions for the most visited areas. There are various ways to do the partition: variance based, entropy based, clustering based. The method proposed in [43] is the maximum entropy (ME) which uses the wavelet transform in its implementation.

• Decide on the number of symbols of the alphabet: [43] uses a specific technique also based on maximizing an entropy function.

• Represent the transitions between symbols: Uses a PFSA (Probabilistic finite state automata) based on D-Markov machines (Markov chains that consider D past values).

• The way to determine if the behavior of the data sequence has changed based on its probability structure: A distance measure is used. In [43] are presented two: one based on the angle between the state probability vectors and other based on the Kullback-Leibler distance.

This technique has been used mainly to identify early signs of anomaly condition on dynamical systems, usually in mechanical systems. And it is presented as a univariate signal analysis technique.

Even when the technique seems promising to identify synchronization, as it is very appropriate to capture changes in system dynamical properties, I have not seen much work in applying it to sync detection, and to extend it to multivariate systems. An exception to the latter is [38], where it is presented a technique to apply symbolic dynamic to synchronization detection of a multivariate dynamical system; nevertheless the technique is tested with simulated systems and it is not clear the applicability to time series data coming from dynamical systems of unknown internal structure.

Another important application of symbolic dynamic (not related with sync detection) is to estimate event-related brain potentials (ERP) [44], meaning, to be able to associate an external sensory event to specific changes in signals behavior in the brain.

4.2.9 Synchronization cluster analysis

The methods presented in the following sections are based in analyzing the eigenvalues an eigenvectors of a matrix that represent the interaction between all the signals. The matrix considered for each method is different.

The objective is to use the distribution of eigenvalue and eigenvector to identify synchronized nodes forming clusters of synchronization, the number of clusters and the nodes integrated in each cluster.

The identification of synchronization clusters is here different to the recurrent patterns of section 5. Here, by synchronization clusters it is meant the identification of groups of nodes which are synchronized during all (or most) of the time (a similar dynamical property vs. time). In section 5, a cluster is a recurrent pattern which is a repeating spatiotemporal sequence of signals or events which occurrence is unusually high, that is we consider repeating patterns not similarity of nodes signals over time.

4.2.9.1 Eigenvalue Decomposition Method

As presented in [9][41] this method starts computing the phase difference matrix. This matrix is obtained from the instantaneous phase of each node ϕ_i with i=1…N, where N is the total number of nodes (neurons). The elements of the matrix are:

$$R_{ij} = \left| \frac{1}{n} \sum_{l=1}^{n} \exp\left(i\,(\phi_{jl} - \phi_{il}) \right) \right|$$

Where $l = 1 \ldots n$ corresponds with the discrete time index.
The matrix can be seen as a correlation matrix (the modulus of complex correlation coefficient of the instantaneous phases of the signals).

Then the eigenvalues and eigenvectors of R are computed. Being R a correlation matrix has real and non-negative eigenvalues and real valued eigenvectors. There are N eigenvalues and eigenvectors with:

$$\lambda_1 \leq \lambda_2 \leq .... \leq \lambda_N; \quad \sum \lambda_k = tr(R) = N;$$

The trace of R is always N (the diagonal are 1's), that means that the sum of eigenvalues is constant. In the case of uncorrelated signals $\lambda_k = 1$ for all k, but in other cases if an eigenvalue is greater than 1 has to be compensated by others being smaller than 1.

These properties of the eigenvalues are employed to identify the clusters in the following way:

1. Clusters with no inter-cluster synchronization: In this case, the eigenvalues and eigenvectors are computed, but we only consider the eigenvalues (λ_k) greater than 1 and its corresponding eigenvectors (v_k). In this case we can consider the following points:
   - Synchronization clusters are identified by eigenvalues with $\lambda_k > 1$. The eigenvalues themselves quantify the strength of the clusters.
   - For each cluster (associated with a $\lambda_k > 1$), the corresponding eigenvector describes its internal structure. That is, if we do the correspondence between the components value of the eigenvector to the corresponding node, then a high value of a component of the eigenvector is associated with a strong participation of the corresponding node in the cluster.
   - The involvement of channel i in cluster k can be quantified by the participation index $\lambda_k \mathrm{V}_{ik}^2$.

   These ideas are developed in the following figure 4.2.9.1-1:

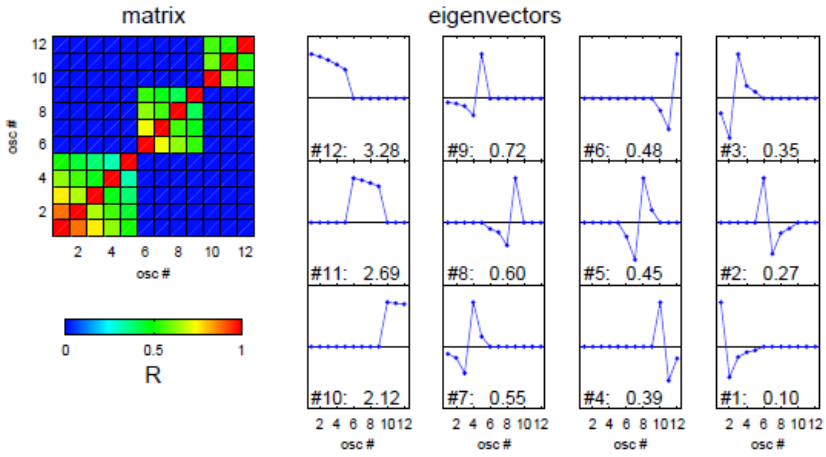


Figure 4.2.9.1-1. Three synchronization clusters with no inter-cluster synchronization [41]. Corresponding to 12 oscillators (nodes) separated in 3 clusters of 4 nodes each, with internal synchronization in each cluster as a separate entity.

In figure 4.2.9.1-1 we can see 3 clusters that correspond to 3 eigenvalues greater than 1. The eigenvectors for these eigenvalues provide information on the nodes

(oscillators) that belong to the cluster. The eigenvectors are vectors of 12 components, and the components are associated to the nodes, then, the components of higher value are associated to the nodes integrated in the cluster. For example, in the above figure, for eigenvector #12, only the five first components have a significant value (the rest have value zero), that means, that the oscillators #1-5 belong to that cluster.

2. Clusters with inter-cluster synchronization: The procedure is similar to the above method, but in this case, due to the inter-cluster synchronization there will be nodes belonging to more than a cluster (see figure 4.2.9.1-2)

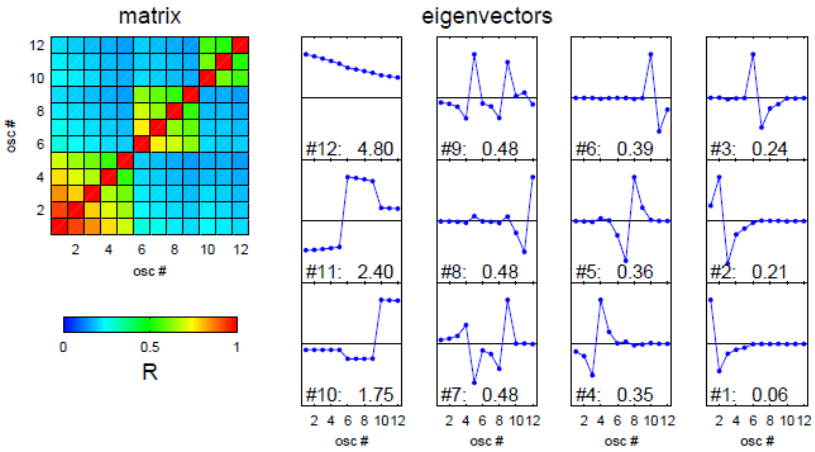


Figure 4.2.9.1-2. Three synchronization clusters with inter-cluster synchronization [41]. Corresponding to 12 oscillators (nodes) separated in 3 clusters of 4 nodes each, with internal synchronization in each cluster and inter-cluster synchronization between clusters.

In this case we can see three eigenvalues greater than 1 (3 clusters) but the components of the eigenvectors doesn't tell us which node belong to which cluster.

The way to proceed is to use the participation index $\lambda_k V_{ik}^2$ and to attribute each oscillator to that cluster for which its participation index is maximal. This can be seen in figure 4.2.9.1-3. Where: (1) each node is associated to the cluster with maximal participation index, (2) the new matrix R is calculated (with inter-cluster values removed) and (3) the recalculated eigenvalues and eigenvectors of the trimmed matrix R provide the correct assignment of clusters to nodes.

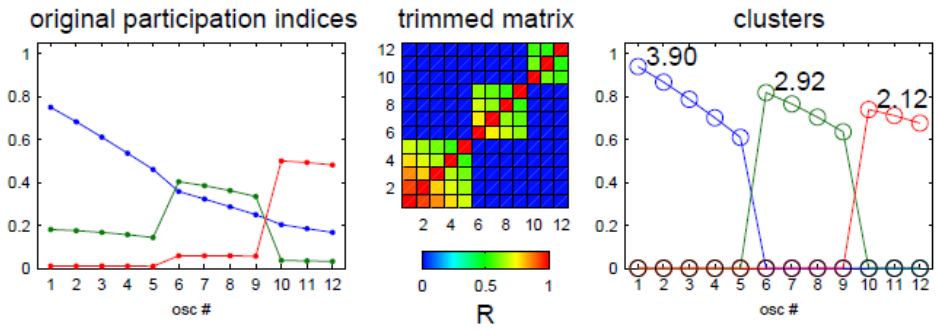


Figure 4.2.9.1-3. Three synchronization clusters with inter-cluster synchronization. The diagram presents the procedure to obtain the correct association between clusters and nodes [41].

4.2.9.2 Empirical Mode Decomposition

In [9][92] is presented an additional step to the method of section 4.2.9.1.

Traditional spectral decompositions applied to time series usually involve an assumption that the underlying signal dynamics consist of a linear superposition of complex exponentials as basis functions. These basis functions are always assumed a priori (as sine and cosines) rather than obtained adaptively from the signal.
Additionally, Fourier transform methods require piecewise stationarity of the time series under study. Thus any time-varying frequency content will be averaged out in the power spectrum.

A way to solve this problem is the computation of the Fourier power spectrum over short time windows, where the signal is considered to be stationary. This implies to assume a correct time-scale for the time-windows.

Another approach is to use a Wavelet Transform which accounts for non-stationarity, but the basis functions are, again, determined before the decomposition (Morlet, Gabor, …) and thus may be inappropriate for nonlinear signals.

Ideally, a decomposition method should be capable of extracting underlying oscillators from the signal without any assumptions of the underlying waveform or characteristic time-scales of the underlying oscillators.

The EMD (Empirical Mode Decomposition) method is capable of the extraction of (potentially non-linear and non-stationary) oscillators from any time series in an adaptive fashion.

The EMD method is also referred as the Hilbert-Huang transform.

The empirical mode decomposition (EMD) transforms the original signal x(t) in a series of intrinsic mode functions (IMFs) that together comprise the underlying oscillations (or basis functions) of the original signal.

The IMFs are obtained in the following way: The decomposition begins by identifying all maxima and minima of the data set followed by interpolation between the extreme values using a cubic spline to obtain a maximum envelope function $\mathrm{e}_{\max}(\mathrm{t})$ and a minimum envelope function $\mathrm{e}_{\min}(\mathrm{t})$. Next, the average of the envelopes $\mathrm{m}(\mathrm{t}) = \left[\mathrm{e}_{\max}(\mathrm{t}) + \mathrm{e}_{\min}(\mathrm{t})\right]/2$ is computed and subtracted from the original data x(t) to obtain a residual $\mathrm{r}(\mathrm{t}) = \mathrm{x}(\mathrm{t}) - \mathrm{m}(\mathrm{t})$. This process is then repeated on the residual in an iterative fashion to obtain the first IMF, which is the residual left after k iterations of the above described process. Once the first IMF: $c_1(t)$ is obtained, it is subtracted from the signal: $\mathrm{x}(\mathrm{t})$ and the process is started again on the new signal: $x(t) - c_1(t)$ to obtain the second IMF: $c_2(t)$. The process to obtain the third IMF starts from: $x(t) - c_1(t) - c_2(t)$ and goes on again to have the third IMF. The process finally stops when we arrive to an IMF which is a monotonic function which does not allow further decomposition.

Finally the signal can be decomposed as:

$$\mathrm{x}(\mathrm{t}) = \sum_{j=1}^{n} c_j(t) + \text{last residual} \quad (89)$$

There are different methods to establish the number k of iterations per IMF computation [92].

The method goes on from this point in a similar way as the Eigenvalue Decomposition method, but, instead of obtaining the instantaneous phase (using the Hilbert transform) from the original data x(t), it is obtained from the IMFs previously computed from the data. The IMFs are themselves time-series and can be the subject of the Hilbert transform.

To apply the Hilbert transform to a signal x(t) is necessary to filter previously the signal, because the method is only appropriate in the case of a narrowband signal. A wideband signal will yield a trajectory in the complex plane that has multiple centers of rotation, and for an unambiguous determination of phase, the complex trajectory of the analytic signal must have only a single center of rotation, thus this method requires prior filtering of the signal. The IMFs produced by the EMD have precisely this property. So, it is not required to filter the IMFs before applying the Hilbert transform.

4.2.9.3 SSA, MSSA

In [39][40][42][60] are presented the SSA (Singular Spectrum Analysis) an MSSA (Multichannel Singular Spectrum Analysis) methods. The MSSA method is a multivariate extension of SSA and it will be the one presented here.

In MSSA we assume to have $X_d(n);\ n = 1...N;\ d = 1...D;$ as a multivariate time-series of D channels of length N. The starting point of M-SSA is to embed each channel into an M-dimensional phase space, by using lagged copies:

$X_{\mathrm{d}}(\mathrm{n}) = (\mathrm{x}_{\mathrm{d}}(\mathrm{n}),...,\mathrm{x}_{\mathrm{d}}(\mathrm{n+M\text{-}1}));\ n = 1...N - M + 1$

From this we form the full augmented trajectory matrix $\mathrm{X} = (\mathrm{X}_1; \mathrm{X}_2; \ldots; \mathrm{X}_\mathrm{D})$ which has $\mathrm{D} \times \mathrm{M}$ columns of length N -M + 1. The M-SSA algorithm then computes the covariance matrix $\mathrm{C} = \frac{\mathrm{X}^T \mathrm{X}}{N}$ of X. Matrix C has size $\mathrm{DM} \times \mathrm{DM}$.
Next, one diagonalizes the covariance matrix:

$$\Lambda = E^T C E$$

To yield a diagonal matrix Λ that contains the real eigenvalues of C, and a matrix E whose columns are the associated eigenvectors e_k. The eigenvectors e_k form a new orthogonal basis in the embedding space of X, and the corresponding λ_k give the variance in the direction of e_k. The e_k is called empirical orthogonal function (EOFs).

The original signals can be projected into the eigenvectors producing the corresponding Principal Components (PC) matrix A:

$$A = XE$$

The PCs have the same length N-M+1 as X.

The signal can be reconstructed from the elements of matrix A and the eigenvectors. First we have the reconstructed components (RC): r_{dk} that represents the part of channel x_d that corresponds to (λ_k, e_k):

$$r_{dk}(n) = \frac{1}{M_n} \sum_{m=L_n}^{U_n} a_k(n-m+1) e_{dk}(m)$$

From the RCs the original signal can be reconstructed as:

$$x_d(n) = \sum_{k=1}^{DM} r_{dk}(n)$$

The principal property of MSSA is that the eigenvectors e_k are able to capture oscillatory behavior in time via oscillatory pairs. An oscillatory pair is a couple (λ_k, e_k) with the following properties:

- The two eigenvalues are nearly identical
- The two corresponding time sequences described by the EOFs (the sequence formed by the components of the eigenvector) are nearly periodic, with the same period and in quadrature.
- The associated PC's are in quadrature.

If this happens, then there is in the time series an oscillation whose period is the same as the period of the EOF itself and whose spatial pattern is the same as the one of the EOF.

The number of eigenvalue pairs with a significant value indicates the number of clusters and the EOFs provide the period and oscillation pattern (the particular nodes that participate on each cluster).

The significant eigenvalue pairs are taken as the eigenvalue in the upper part of the elbow point in a scree plot (eigenvalue index vs. eigenvalue value). In figure 4.2.9.3-2 is presented one scree plot with the sudden change in the value of the eigenvalues that indicate which ones are significant. In the horizontal axis are the eigenvalues in higher to lower value order.

In figure 4.2.9.3-1 is presented the evolution of the eigenvalues in a system with five Rossler coupled oscillators as the coupling strength is increased. We can see that the eigenvalues go in pairs and its number and evolution indicate the number of oscillation (synchronization) clusters that are in the system.

In figure 4.2.9.3-3 are presented the eigenvectors e_k corresponding to the 10 largest eigenvalues, after applying MSSA to 5 uncoupled Rossler oscillators (each oscillator being defined by three variables: x,y,z). Each eigenvector of length DM is composed of D = 15 consecutive segments, each of length M = 30, associated with the 15 channels.

Looking to eigenvectors of figure 4.2.9.3-3 we observe that it is not possible to identify which are the nodes that participate on each cluster. In order to allow a more clear identification the eigenvectors are further rotated. The intention of this posterior eigenvector rotation is to simplify the structure of the eigenvectors, to reduce mixture effects and to improve the physical interpretability. The usual methods are Varimax rotation and other alternative methods [39].

In figure 4.2.9.3-4 are presented the eigenvectors (same as in previous figure) after being rotated. After rotation it is possible to see clearly the oscillatory pairs with the corresponding eigenvectors in quadrature, and the identification of the oscillators that pertain to each oscillatory group, which are clearly separated as being part of 5 uncoupled systems.

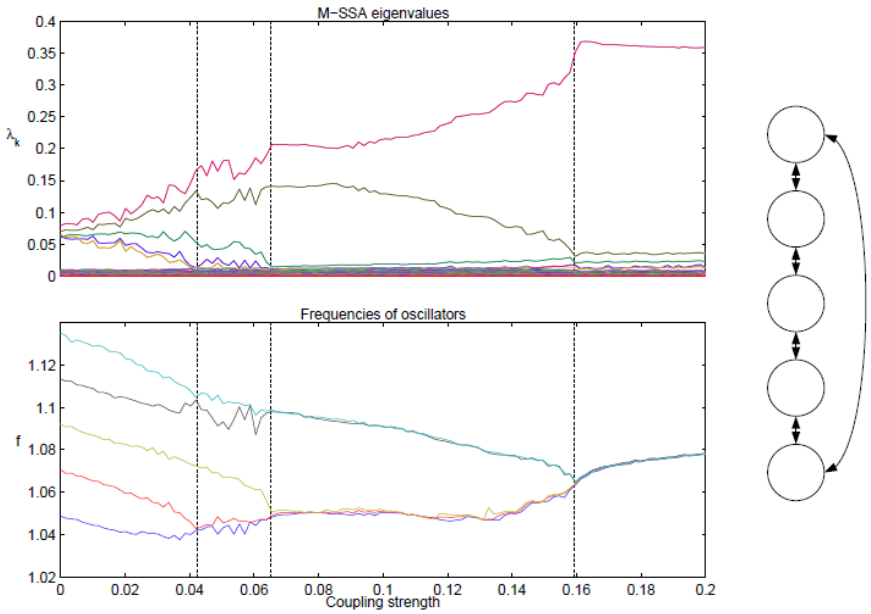


Figure 4.2.9.3-1. Evolution of eigenvalue pairs with number of oscillations in the system [61]. To the right of the diagram we can see that we have a single

frequency of oscillation when the coupling strength is high enough, at this time we have only one significant eigenvalue left.

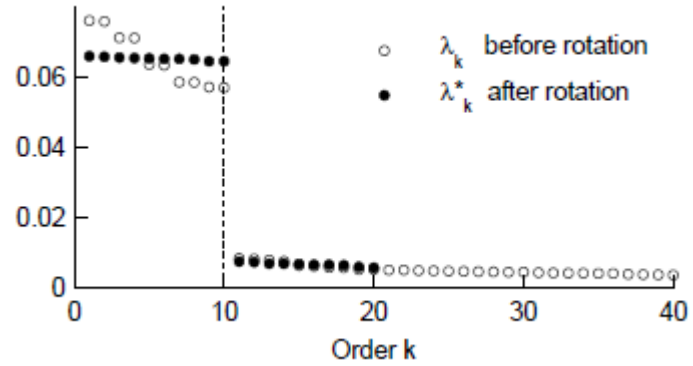


Figure 4.2.9.3-2. Plot of eigenvalue index vs. eigenvalue value (scree plot)[39].

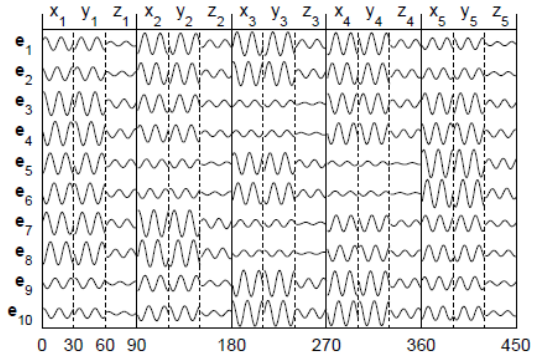


Figure 4.2.9.3-3. Eigenvectors corresponding to the 10 largest eigenvalues of 5 uncoupled Rossler oscillators (each oscillator being defined by three variables: x,y,z) with an embedding dimension M=30 which yields DM=450 [39].

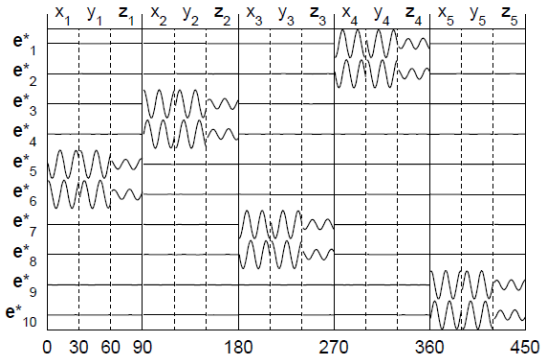


Figure 4.2.9.3-2. Eigenvectors of 5 uncoupled Rossler oscillators after being rotated (Varimax rotation or alternative method) [39].

In [53] is presented a method similar to MSSA but the matrix to which is applied the spectrum analysis is formed as the autocorrelation function of the normalized binned spike activity of the neurons.

4.2.10 Matrix similarity measures

If we consider a multivariate signal $X = \{x_{ij}\};\ i = 1...N;\ j = 1...M$ produced by N nodes and each signal having a length M, as a matrix, it is possible to apply norms defined on matrices to establish how close two multivariate blocks of signals are.

That is, the distance between multivariate time-series X and Y can be defined as the norm of $\mathrm{X} - \mathrm{Y}$: $\|X - Y\|$

The following methods apply these ideas to determine similarity between two multivariate blocks of signals.

4.2.10.1 Frobenius norm (raw values)

The Frobenius norm corresponds to the square of the absolute value of all elements in the matrix. This is a matrix norm and can be used to provide a single value measure to the matrix. When applied to the difference between two matrices provide a measure of the closeness of both matrices.

The Frobenius distance between matrices A and B is defined as [69]:

$$C = A - B;$$
$$\|C\|_F^2 = \sum_{ij} |c_{ij}|^2 = trace(C^T C) = \sum_j \langle c_{*j}, c_{*j} \rangle \quad (90)$$

Where $\langle c_{*j}, c_{*j} \rangle$ is the inner product of the columns vectors of matrix C.

The Frobenius norm of C can be written in terms of the columns vectors of A and B:

$$C = A - B;$$
$$\|C\|_F^2 = \sum_{ij} |c_{ij}|^2$$
$$= 2n - 2\sum_j \langle a_{*j}, b_{*j} \rangle \quad (91)$$
$$= 2n - 2\sum_j \cos\theta_i$$

Where $\cos\theta_i$ is the angle between the corresponding column vectors of A and B. And n is the number of columns.

The Frobenius norm can be extended to a weighted norm, as [70]:

$$C = A - B;$$
$$\|C\|_{F,W}^2 = \sum_j w_{jj} \sum_i |c_{ij}|^2$$
$$= 2 - 2\sum_j w_{jj} \langle a_{*j}, b_{*j} \rangle \quad (92)$$
$$= 2 - 2\sum_j w_{jj} \cos\theta_i$$

Where W is a weight matrix (symmetric diagonal positive semidefinite), with values not in the diagonal equal to zero, the values in the diagonal positive and with a matrix trace equal to 1.

Besides the Frobenius norm there are other matrix norms [69]:

- Matrix 2-norm: To obtain the distance between matrices A and B, the measure of the difference of the matrices is:

  $$C = A - B;$$
  $$\|C\|_2 = \sqrt{\lambda_{\max}}$$

  Where $\lambda_{\max}$ is the largest eigenvalue of $C^T C$.
- Matrix 1-norm: To obtain the distance between matrices A and B, the measure of the difference of the matrices is:

  $$C = A - B;$$
  $$\|C\|_1 = \max_j \sum_i |c_{ij}|$$

  That is, the largest absolute column sum (of the difference between the matrices)
- Matrix ∞-norm: To obtain the distance between matrices A and B, the measure of the difference of the matrices is:

  $$C = A - B;$$
  $$\|C\|_\infty = \max_i \sum_j |c_{ij}|$$

  That is, the largest absolute row sum (of the difference between the matrices)

All these raw value norms are based in the values of the matrix but do not consider any interrelation between values, neither any specific direction of maximum variance (information). That is the reason for not using these measures and, instead, using matrix norms over matrices obtained from the raw data after applying transformations that produce principal components or most significant (variance, energy) eigenvectors.

4.2.10.2 Frobenius norm (eigenvectors)

In [70] is presented a similarity distance method between two multivariate time series based in the weighted extended Frobenius norm (see section 4.2.10.1) where the matrices are not the raw value matrices (A and B) but two matrices obtained from the right eigenvectors after applying Singular Value Decomposition (SVD) to the original raw value vectors.
In this case the matrix W (of weights) is obtained after the application of a function to the eigenvalues obtained from the matrices. After applying SVD to A and B we will obtain two sets of eigenvalues, the values of matrix W is obtained after applying a function to each pair of corresponding eigenvalues (the pair of eigenvalues of the spectral decomposition of A and B). This function can be the mean or max value of the pair of eigenvalues.

That is, considering the SVD of A and B:

$$A = U_A \Sigma_A V_A^T$$
$$B = U_B \Sigma_B V_B^T \quad (93)$$

The similarity between A and B is:

$$C = V_A - V_B;$$
$$\|C\|_{F,W}^2 = \sum_j w_{jj} \sum_i |c_{ij}|^2$$
$$= 2 - 2\sum_j w_{jj} \left\langle v^a{}_{*j}, v^b{}_{*j} \right\rangle \quad (94)$$
$$= 2 - 2\sum_j w_{jj} \cos\theta_i$$

Where $\left\langle v^a{}_{*j}, v^b{}_{*j} \right\rangle$ is the inner product of the corresponding column vectors of the right eigenvectors of the SVD decomposition.

4.2.10.3 Weighted sum SVD (WSSVD)

This method [71] is similar to the Frobenius norm with eigenvectors but with some changes.

From the original matrices A and B, we first calculate the covariance matrices:

$$Q_1 = A^T A$$
$$Q_2 = B^T B \quad (95)$$

Applying SVD to Q_1 and Q_2 we have:

$$\mathrm{Q}_1 = \mathrm{V}_1\ \Sigma_1\ \mathrm{V}_1^T$$
$$\mathrm{Q}_2 = \mathrm{V}_2\ \Sigma_2\ \mathrm{V}_2^T \quad (96)$$

Being the covariance matrices symmetric the right and left eigenvector are equal.

The WSSVD similarity between A and B is defines as:

$$\theta_{WSSVD}(A,B) = \min(\theta_1(A,B), \theta_2(A,B));$$
$$\theta_1(A,B) = \frac{\sum_{i=1}^{r} c_i \left\langle e_{*i}, f_{*i} \right\rangle}{\sum_{i=1}^{r} |c_i|}; \quad (97)$$
$$\theta_2(A,B) = \frac{\sum_{i=1}^{r} d_i \left\langle e_{*i}, f_{*i} \right\rangle}{\sum_{i=1}^{r} |d_i|}$$

Where e_{*i} and f_{*i} are the vectors columns of V_1 and V_2 respectively, and, c_i and d_i are the diagonal values of Σ_1 and Σ_2 respectively.

This similarity metric compares the similarity along each maximum variance directions of both matrices, compensated with the variance level given by the eigenvalues.

Matrices A and B need to have the same number of nodes (columns) but can have different number of sampled times (rows). This can be an important feature in case we need to compare multivariate data blocks with different time samples.

4.2.10.4 PCA similarity factor

PCA Similarity Factor (PCASF) [70], is defined between two matrices of the same number of columns, but not necessarily the same number of rows.

First it is obtained the principal components for each matrix, and selected the first k principal components based on explained total variance. The explained total variance of a principal component is associated with the value of the associated eigenvalue. The sum of the normalized eigenvalues corresponding with a certain number of principal components matches with the explained total variance of those principal components. A normalized eigenvalue is the value of the eigenvalue divided by the sum of all eigenvalues.

PCASF then computes the similarity between the first k principal components, as:

$$PCASF = trace(LM^T ML^T) = \sum_{i=1}^{k}\sum_{j=1}^{k}\cos^2\theta_{ij} \quad (98)$$

Where L and M are the matrices formed with the first k principal components of A and B, respectively. And θ_{ij} is the angle between the ith principal component of A and the jth principal component of B.

In PCASF we consider the angles between any combinatorial pair of selected principal components of A and B, and in Frobenius norm (eigenvector) we consider the angles between the corresponding eigenvectors of A and B (not combinatorial).

4.2.10.5 Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW) can also be used to calculate a distance between two multivariate ensembles [73] [74] [75].

DTW is usually applied to two univariate signals X and Y with possibly different length (n and m). The application of the method consist in creating a matrix $D(n\times m)$ of all pair distances between points X(i) and Y(j), $i=1..n; j=1...m$. The distance between points can be calculated using different distance norms (Euclidean, city-block,..).

The total distance between X and Y is calculated by computing the optimal warping path that connect point D(1,1) with D(n,m) and which minimizes the sum of visited values in D. The warping path is denoted by [p(1), p(2),.., p(l)] where p(1)=D(1,1) and p(l)=D(n,m).

To build the path some rules have to be followed [73]:

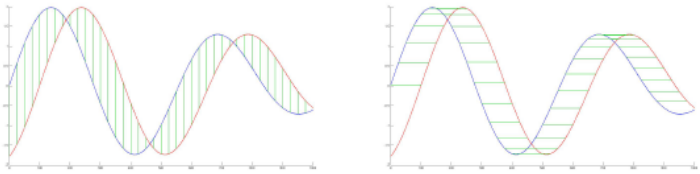
- Boundary condition: The path has to start in D(1,1) and end in D(n,m) .

- Monotonicity: The path has to be monotonous, i.e. always heading from D(1;1) to D(n,m) . If p(k) = D(i, j) and p(k +1) = D(i', j') then i'−i ≥ 0 and j'− j ≥ 0.

- Continuity: The path has to be continuous. If p(k) = D(i, j) and p(k +1) = D(i', j') then i'−i ≤ 1 and j'− j ≤1.

The sum of D values of the optimal warping path provides the distance between X and Y. To calculate the distance it is used an algorithm based in dynamic programming based upon an intermediate matrix D of accumulated distances from each point to all the other reachable points.

In figure 4.2.10.5-1 is presented DTW applied to two signals out of phase and the different results of applying the Euclidean and DTW distance. In these cases, of similar but not identical signals (shifted, scaled,…) the application of DTW is a very suitable distance measure.



(a) Euclidean Distance (b) Dynamic Time Warping

Figure 4.2.10.5-1 DTW applied to two similar but shifted signals

The application of DTW to two multivariate ensembles is based in calculating the distances between multidimensional points, in the same way that DTW calculates distances between corresponding scalar points of both time series, in the case of multivariate signals, the distances are calculated between multidimensional points that are obtained from the original matrices. Depending of the procedure to extract the multidimensional points, we can have:

- The multidimensional points are the rows or columns of the matrices, and the distances are the Euclidean or other norm distance between vectors [75].
- The multidimensional points are matrix segments obtained from the original matrices and the distances between these matrix segments can be one of the previous revised matrix distances (previous sections). To obtain the segments some algorithm based in homogeneity of the segments can be used [73]

There is Matlab code for DTW:
http://www.mathworks.com/matlabcentral/fileexchange/16350-continuous-dynamic-time-warping
http://www.mathworks.com/matlabcentral/fileexchange/6516-dynamic-time-warping

http://www.ee.columbia.edu/~dpwe/resources/matlab/dtw/

4.2.11 Recurrence based

4.2.11.1 Recurrence Plots

Recurrence plots [18][77] are a powerful tool for the study of synchronization in dynamical systems and, furthermore, for the study of the general properties of such systems.

Recurrence plots (RP) methods are based in studying the properties of the recurrence matrix, defined on a dynamical system which follows a trajectory $\left[\vec{x}_i\right]_{i=1}^{N}$ in its phase space, where $\vec{x}_i$ are the vectors visited by the trajectory. The recurrence matrix is defined as:

$$\mathbf{R}_{i,j} = \begin{cases} 0 : \vec{x}_i \neq \vec{y}_j \\ 1 : \vec{x}_i \approx \vec{y}_j \end{cases} \quad i,\ j = 1,....\mathrm{N} \quad (99)$$

Where N is the maximum index (discrete time) and $\vec{\mathrm{x}}_\mathrm{i} \approx \vec{\mathrm{x}}_\mathrm{j}$ means equality up to an error (or distance) ε. That is the recurrence matrix indicates with a 1 when the system has a similar state considering two different times (indices i and j).

The recurrence matrix is a symmetric matrix with values {0, 1}, and previous to its computation is necessary to define:

- The value ε that will be adopted to establish two states as equal. To use a proper value of ε is crucial, and in [18] are proposed several methods.
- The distance (norm) to be adopted to compare states $\vec{\mathrm{x}}_\mathrm{i}$ and $\vec{\mathrm{x}}_\mathrm{j}$ for equality. The usual norms are the Euclidean, L_1 or L_∞ norms (Maximum or Supremum norm). The L_∞ is usually the norm adopted as it is easy to compute and allows to study some features in RPs analytically.

Typical representations of RPs are given in figure 4.2.11.1-1 for different systems: a periodic motion with one frequency (A), the chaotic Rossler system (B) and a uniformly distributed noise process (C).

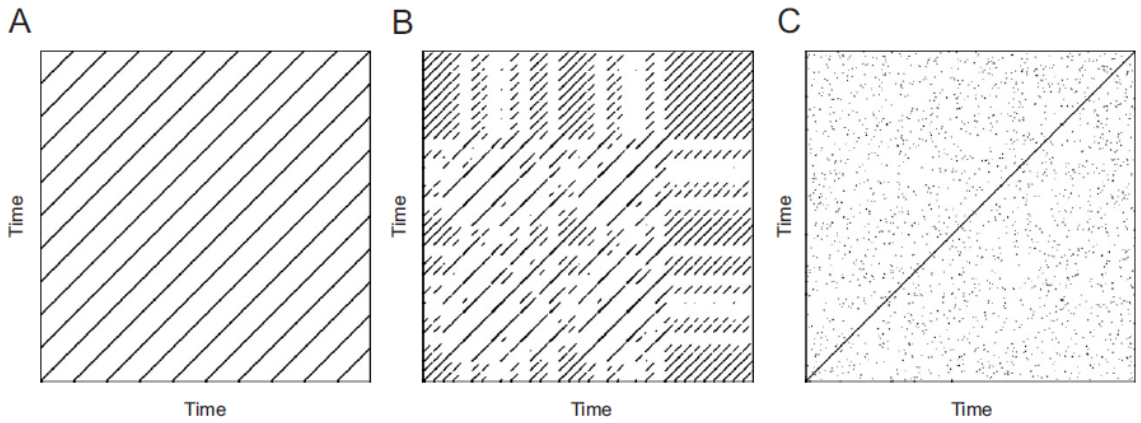


Figure 4.2.11.1-1 RPs for different dynamical systems [18]

The study of the properties of RPs can provide valuable information on the underlying dynamical system. For example, the length of the diagonal lines provides information on predictability of the system and the inverse of the longest diagonal is proportional to the largest Lyapunov exponent of the system [18].

The diagonal line of the RP is usually not considered as it will be always present ($R_{i,i} = 1, \ \forall i=1,..,N$).

Actually, as can be seen in figure 4.2.11.1-2, a diagonal line in a RP corresponds with a section of a trajectory (dashed) which stays within a ε-tube around another section (solid).

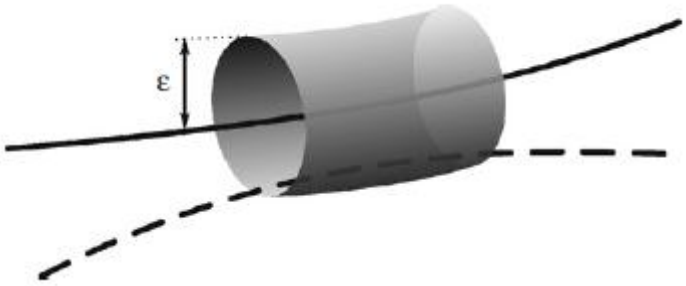


Figure 4.2.11.1-2 Recurrent trajectory captured by an RP [18]

To obtain the recurrence matrix is necessary to have a representation of the system in phase-space. In case we have only a scalar and discrete time-series of the system, it is necessary to extract the phase-state representation from this data. This is done using the time delay method.

To apply the time delay method, we assume that our system is represented by the following scalar and discrete time-series: $u_i = u(i\Delta t)$, where $i = 1,.\ ,\ N$ and Δt is the sampling rate of the measurement, then the vector states of the system can be reconstructed as:

$$\hat{\vec{x}}_i = \sum_{j=1}^{m} u_{i+(j-1)\tau}\, \vec{e}_j, \qquad (100)$$

Where m is the embedding dimension and τ is the time delay. The vectors $\vec{e}_i$ are unit vectors and span an orthogonal coordinate system ($\vec{e}_i.\vec{e}_j = \delta_{i,j}$). If $m \geq D_2 + 1$, where D_2 is the correlation dimension of the attractor, then Takens theorem guarantees the existence of a correspondence between the original and the reconstructed attractor. This means that both attractors can be considered to represent the same dynamical system in different coordinate systems. That is, we are able to build an associated phase space for our system which reconstructs the dynamical properties of the system.

In an RP we can distinguish between typology and texture:

- Typology: Large scale patterns in RPs are designated as typology, and can be classified in homogeneous, periodic, drift and disrupted ones. In figure 4.2.11.1-3 are represented the different types.
    - Homogeneous: A homogeneous RP represents a stationary system, in which the relaxation times are short in comparison with the time spanned by the RP. See Figure 4.2.11.1-3 (A) homogeneous (uniformly distributed white noise)
    - Periodic: The RP presents diagonal lines and/or checkerboard structures. They are periodic o quasi-periodic systems. See Figure 4.2.11.1-3 (B) which represents a periodic system with two harmonic frequencies and with a frequency ratio of four.
      Irrational frequency ratios cause more complex quasi-periodic recurrent structures (the distances between the diagonal lines are different).
    - Drift: The RP pales away from the main diagonal. It is associated to systems with slowly varying parameters. That is, not stationary. See Figure 4.2.11.1-3 (C) represents a logistic map corrupted with a linearly increasing term.
    - Disrupted: Abrupt dynamics or extreme events cause white areas or bands in the RP. See Figure 4.2.11.1-3 (D) represents a Brownian motion.

- Texture: Small scale structures in RPs correspond to the texture, and can be typically classified in single dots, diagonal lines , vertical and horizontal lines (forming rectangular clusters of recurrence points) and bowed lines
    - Single dots: Represent recurrence points that persist only for a very short time.
    - Diagonal lines: A diagonal line of length L occurs when a segment of the trajectory runs almost in parallel to another segment (i.e. through a ε tube around the other segment, figure 4.2.11.1-2) for L time units. The length of the diagonal lines is related with the positive Lyapunov exponents and the Renyi entropy of second order K_2 [18]
    - Vertical and horizontal lines: Marks a time interval in which a state does not change or changes very slowly
    - Bowed lines: They are lines with a non-constant slope.

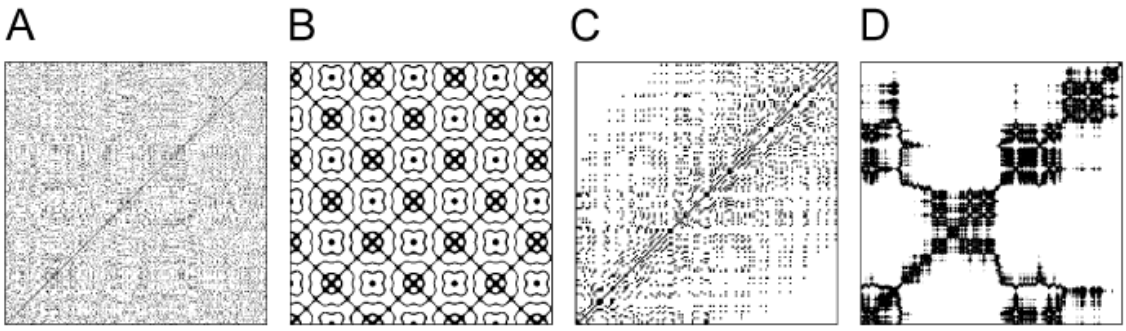


Figure 4.2.11.1-3 Different types of RPs according to typology [18]

Diagonal and vertical lines are the base for a quantitative analysis of RPs.

Till now we have considered only the application of RP to a single time series, the bivariate extension produces the following methods:

- Cross recurrence plot (CRP): It can be considered as a generalization of the linear cross-correlation function. Suppose we have two dynamical systems, each one represented by the trajectories x_i and y_i in a d-dimensional phase space. Analogous to RP, we can define a corresponding cross recurrence matrix as:

$$\mathbf{CR}_{i,j}^{\vec{x},\vec{y}}(\varepsilon) = \Theta(\varepsilon - \|\vec{x}_i - \vec{y}_j\|), \quad i = 1, \ldots, N, \quad j = 1, \ldots, M, \tag{101}$$

  Where Θ represents the Heaviside step function and both time series can have different lengths. The value ε is the minimum distance to establish two states as equal. An important point to consider is that both time series have to be similar in the range of values, otherwise it won't be possible to compare the values.
- Joint recurrence plot (JRP): CRPs allows studying the relationship between two different systems by examining the occurrence of similar states. However, CRPs cannot be used for the analysis of two physically different time series, because the difference between two vectors with different physical units or even different phase space dimension does not make sense. Another possibility to compare different systems is to consider the recurrences of their trajectories in their respective phase spaces separately and look for the times when both of them recur simultaneously, i.e. when a joint recurrence occurs.

  By means of this approach, the individual phase spaces of both systems are preserved. Formally, it is related to an extension of the phase space to R^{dx+dy}, where dx and dy are the phase space dimensions of the corresponding systems, which are in general different. In this way, we can define the joint recurrence matrix for two systems $\vec{x}$ and $\vec{y}$, as:

$$\mathbf{JR}_{i,j}^{\vec{x},\vec{y}}(\varepsilon^{\vec{x}}, \varepsilon^{\vec{y}}) = \Theta(\varepsilon^{\vec{x}} - \|\vec{x}_i - \vec{x}_j\|)\Theta(\varepsilon^{\vec{y}} - \|\vec{y}_i - \vec{y}_j\|), \quad i, j = 1, \ldots, N, \tag{102}$$

  This definition can be extended to n systems $\vec{x}_{(1)}, \vec{x}_{(2)}, \ldots, \vec{x}_{(n)}$ (n time series) and defines the multivariate joint recurrence matrix:

$$\mathbf{JR}_{i,j}^{\vec{x}_{(1,\ldots,n)}}(\varepsilon^{\vec{x}_{(1)}},\ldots,\varepsilon^{\vec{x}_{(n)}}) = \prod_{k=1}^{n} \mathbf{R}_{i,j}^{\vec{x}_{(k)}}(\varepsilon^{\vec{x}_{(k)}}), \quad i,j = 1,\ldots,N. \tag{103}$$

All the time series need to have the same length. The value ε is the minimum distance to establish two states as equal and can be different for each time series. A value of 1 in position (i,j) in the multivariate joint recurrence matrix will occur when all the vector states of all the time series return in time j to a position similar to the position they had in time i.

In the following lines are presented different measures of complexity that can be obtained from the RPs. These measures provide a quantifiable number to the diagrammatic aspects that can be studied from the RP of the system:

- Recurrence density: The recurrence density is obtained using the recurrence rate (RR):

$$RR(\varepsilon) = \frac{1}{N^2} \sum_{i,j=1}^{N} \mathbf{R}_{i,j}(\varepsilon) \tag{104}$$

  This is a measure of the density of recurrence points in the RP. It corresponds to the definition of the auto-correlation sum. Similarly the recurrence density of the cross recurrent plots and joint recurrent plots provides the cross correlation and joint correlation sums.
  The value ε is the minimum distance to establish two states as equal.
- Diagonal lines: There are different measures that try to gain information from the diagonal lines. These measures are based in histograms of total number of diagonal lines of length ℓ in the RP.
  Processes with uncorrelated or weakly correlated, stochastic or chaotic behavior cause none or very short diagonals, whereas deterministic processes cause longer diagonals and less single, isolated recurrence points. That's the reason for calling "determinism" to one of the main expressions related with diagonal [18].
  Diagonal line measures are based in the previous calculation of the histogram $\mathrm{P}(\varepsilon,\ell)$ of diagonal lines of length ℓ:

$$P(\varepsilon,l) = \sum_{i,j=1}^{N} (1 - \mathbf{R}_{i-1,j-1}(\varepsilon))(1 - \mathbf{R}_{i+l,j+l}(\varepsilon)) \prod_{k=0}^{l-1} \mathbf{R}_{i+k,j+k}(\varepsilon) \tag{105}$$

  Once $\mathrm{P}(\varepsilon,\ell)$ is obtained, it can be computed other measures based on it and which provide information on [18]:
  - Determinism
  - Trajectory divergence
  - Mean prediction time
- Vertical lines: Similarly to the diagonal lines, the measures based in vertical lines obtain histograms of total number of vertical lines of length ν in the RP.
  Vertical line measures are based in the previous calculation of the histogram $\mathrm{P}(\nu)$ of diagonal lines of length ν:

$$P(v) = \sum_{i,j=1}^{N} (1 - \mathbf{R}_{i,j})(1 - \mathbf{R}_{i,j+v}) \prod_{k=0}^{v-1} \mathbf{R}_{i,j+k}. \quad (106)$$

Based on $\mathrm{P}(\nu)$ there are additional diagonal line measures which provide information on [18]:

- Laminarity of the dynamics
- Trapping time (in a vertical line an state is trapped for a certain amount of time)

The above measures are called as RQA (Recurrence quantifiable analysis)

There are also a number of dynamical invariants derived from RPs. These invariant measures are independent of the embedding chosen (that is not happening in RQA measures which depend on the embedding). The underlying dynamic system invariants that can be estimated from RPs are generalized entropies, dimensions, mutual information and generalized mutual information (redundancies). All these invariants can be obtained from the generalized correlation sum, defined as:

$$C_q(\varepsilon) = \frac{1}{N} \sum_{i=1}^{N} \left[\frac{1}{N} \sum_{j=1}^{N} \Theta(\varepsilon - \|\vec{x}_i - \vec{x}_j\|) \right]^{q-1} \quad (107)$$

which can be calculated from the RP.

RPs can be used to detect different transitions to synchronization and also to detect synchronization in cases where known methods fail.

There are three basic types of synchronization in coupled complex systems: if the trajectories of the systems evolve due to coupling on the same trajectory, they are completely synchronized (CS). If there is a functional relationship between the systems, they are generalized synchronized (GS), and if their phases adapt to each other so that they evolve in the same manner, the systems are phase synchronized (PS).

The usual path to synchronization of two systems as we increase the coupling strength between them is to start with no synchronization, then to move to PS synchronization, where the phases match but not the amplitudes, if we further increase coupling we obtain Lag Synchronization (LS) where the signals coming from the systems match but one of them is delayed in time. After LS is achieved, a higher level of coupling will produce CS. This path to synchronization is applicable when both systems are identical or similar; if the systems are non-similar there is no way to synchronize them. Nevertheless, in this case, if the level of coupling is strong enough we will obtain GS, where there is a functional relationship between both signals that is dependence relation between them without a direct similarity in the signals.

If two systems synchronize in some way, their recurrences in phase space are not independent from each other. Hence, comparing the recurrences of each single system

with the joint recurrences of the entire system, we can expect to get indications about their synchronization type and degree.

Another application of RPs is to determine the level of PS synchronization of two systems where it is possible to compare the probability of recurrence of both systems after a certain time delay τ. If the probability distribution for both systems is similar, and, in particular, the peaks of probability are situated in the same places we can determine that both systems are PS synchronized. The probability of recurrence after a certain time delay τ can be calculated as:

$$\hat{p}(\varepsilon, \tau) = RR_\tau(\varepsilon) = \frac{1}{N-\tau} \sum_{i=1}^{N-\tau} \Theta(\varepsilon - \|\vec{x}_i - \vec{x}_{i+\tau}\|)$$

Which can be considered as a generalized auto-correlation function, and can be obtained from the RP of the system.

Once the probability of recurrence is found we can define the correlation coefficient between the probabilities of recurrence of two series: $p_{\bar{x}}(\tau)$ *and* $p_{\bar{y}}(\tau)$ are as their inner product:

$$CPR = \left\langle p_{\bar{x}}(\tau), p_{\bar{y}}(\tau) \right\rangle \quad (108)$$

CPR can be used in order to quantify PS, where $p_{\bar{x}}(\tau)$ *and* $p_{\bar{y}}(\tau)$ are the probabilities normalized to zero mean and standard deviation of one. If both systems are in PS, the probability of recurrence will be maximal at the same time and CPR ≈ 1. On the other hand, if the systems are not in PS, the maxima of the probability of recurrence will not occur simultaneously and then we expect low values of CPR.

There is a recurrence plots toolbox for Matlab:
CRP toolbox for Matlab http://tocsy.agnld.uni-potsdam.de

4.2.11.2 Shifted recurrence diagram

This method (developed on this work) is related with recurrence plots [18] and intersection matrices [66]. In this method it is built a matrix of similarity distances between neurons activity at different times, so, it is different to a recurrence plot because here the matrix considers the distance (difference) on the activity of N neurons and not the distance of the state space vectors of the associated dynamical systems. It is also different to the intersection matrix method because here the focus is placed in identifying, with a probability distribution, the whole ensemble activity and not in identifying synfire chains activations [66].

Assuming that we have N neurons, each producing a time-series (spikes with values 0/1 or voltages with values [Vmin,Vmax]) of length T.
The element $\mathrm{SRD}(\mathrm{i},\mathrm{j})$ $\mathrm{i},\mathrm{j}=1.\mathrm{T}$, of matrix SRD, is defined as:

$$SRD(i,\ j) = \mathrm{distance}(\overrightarrow{act}(i), \overrightarrow{act}(j)) \quad (109)$$

That is, $\mathrm{SRD}(\mathrm{i},\mathrm{j})$ corresponds to a normalized distance between the activity vectors of the neurons ensemble at time i and j. The activity vector at time i: $\overrightarrow{act}(i)$ is a vector of length N with component values equal to the activity value of all neurons (0/1 in case of spike trains or voltages values otherwise).

Building a matrix A for the activity of all neurons, the matrix dimensions of A will be $N \times T$. Then:

$$\mathrm{A} = \left[\vec{a}_1 \ \vec{a}_2 \ ..... \ \vec{a}_T \right]$$

Where $\vec{a}_i$ will be a vector of length N which correspond to the activity of all neurons at time i.

Using A, matrix SRD can be defined as:

$$SRD(i,\ j) = \mathrm{distance}(\vec{a}_i, \vec{a}_j) \quad (110)$$

The distance function can be chosen. Several distances are proposed:

- Euclidean
- Correlation
- Cosine
- Jaccard
- Mahalanobis
- Spearman correlation

Depending on the distance function, the chosen time-series activity representation will be also different. For example, for the jaccard distance we will chose the spike train data and for correlation distance we will chose the voltage data.

SRD is a symmetric matrix with values specifying the similarity between the activity of all neurons in the ensemble at time i and at time $\mathrm{j} = \mathrm{i} + \Delta\mathrm{i}$, that is, between a certain time and other incremental time.

In figure 4.2.11.2-1 is presented the SRD for different distance measures. Actually, what is really presented in the figure is the Similarity Matrix which is calculated as one minus the SRD. That is a zero value in the similarity matrix indicates no similarity and a one indicates maximum similarity (the opposite to distance). The colors in the figure are: white is maximum similarity and black is no similarity.

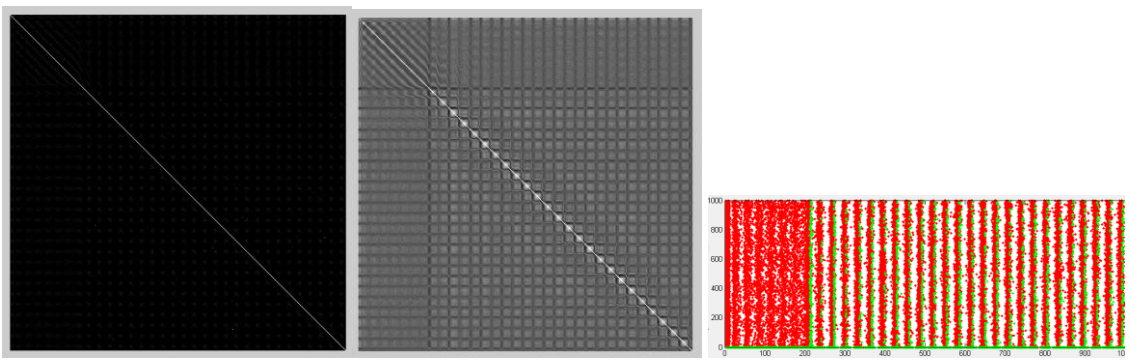


Figure 4.2.11.2-1 Similarity matrix for jaccard (left) and correlation (right) distance measures of same ensemble of signals (further to the right).

In figure 4.2.11.2-2 is shown the probability distribution of values in similarity matrices for several distance measures.

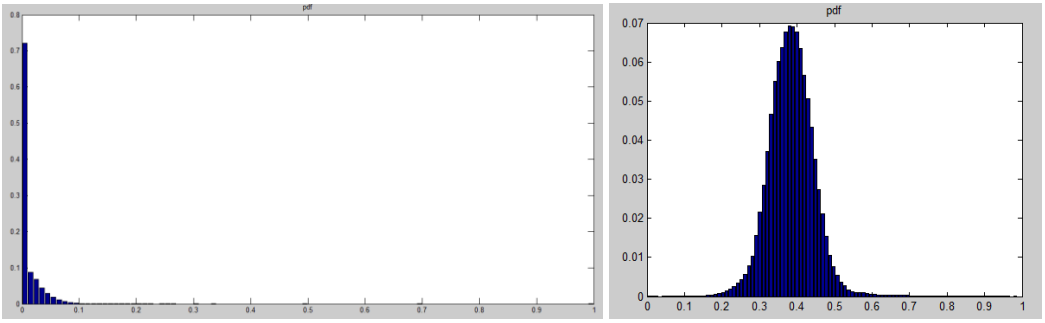


Figure 4.2.11.2-2 Probability distribution of values in similarity matrices for jaccard (left) and correlation (right) distances.

Once SRD is built we can establish a threshold value Γ and consider as significant only the values of the matrix over that threshold.
We build the matrix of significant values, defined as:

$$SRDs(i,j) = \begin{cases} 0; & \text{distance}(\vec{a}_i, \vec{a}_j) \leq \Gamma \\ \text{distance}(\vec{a}_i, \vec{a}_j); & \text{distance}(\vec{a}_i, \vec{a}_j) > \Gamma \end{cases} \quad (111)$$

With the new matrix of significant values we can compute the recurrence probability distribution over $\Delta \mathrm{i} = 1 \ldots \mathrm{T}$ of the number of times that we have a significant value (value in SRDs over zero) at a certain incremental (shifted time) $\Delta \mathrm{i}$. We call $p_{\Delta i}$ the recurrence probability distribution, with $\Delta \mathrm{i} = 1 \ldots \mathrm{T}$. The counting process, to obtain the recurrence probability distribution at shifted times is performed over all the rows of the matrix (starting at the diagonal position for each row). The selection of the threshold value Γ is an important aspect of the method.

An alternative method to compute the recurrence probability distribution is a weighted variant which in the counting process does not add the number of significant values (values over zero) but the value of the SRD element itself (not the SRDs) for that particular $\Delta \mathrm{i}$. When the resulting histogram is divided by the total sum of values the result is a probability distribution that takes into account the bigger weight provided by time increments where a greater synchronization is produced.
In this alternative method we can opt for a threshold value Γ of zero, as the noise from the smaller values is naturally reduced due to its low value contribution.

In figure 4.2.11.2-3 is shown the recurrence probability distribution for several distance measures.

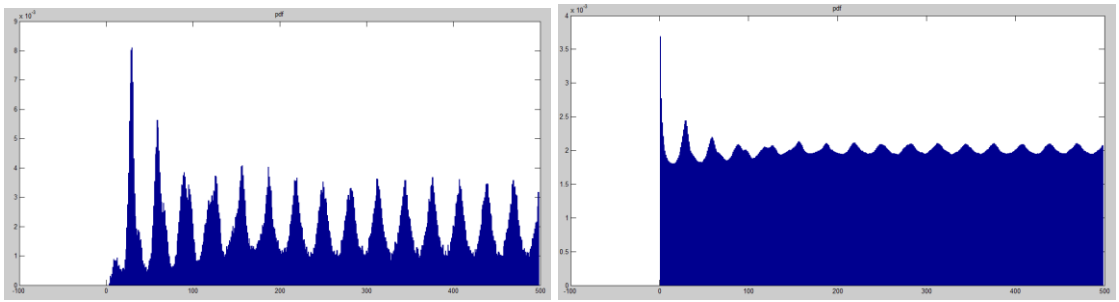


Figure 4.2.11.2-3 Recurrence probability distribution for jaccard (left) and correlation (right) distances, for a threshold of zero.

The peaks of the recurrence probability distribution will show the periods of the most important oscillatory frequencies of the system. And the form of the probability distribution corresponds with the distribution of the significant frequencies of the system (considered as an ensemble).

The above computed probability distribution represents, in some way, the complete activity of the ensemble (at least of the statistical distribution of its recurrent patterns of activity). So, it seems representative of the synchronous activity of the ensemble, and can be used to represent it. If this approach is taken, we can use the recurrence probability distribution of two different ensembles (or the same ensemble under different input signals) and perform a Kullback-Leibler divergence on both probability distributions to establish the distance between them. In that way, the distance between their probability distribution can be considered as a distance between their respective levels of synchronous activity. This approach can be used as a measure of distance of two multivariate signals (block-multivariate measure).

Additionally, it can be defined another variant of the SRD matrix which is calculated over blocks of activity of neurons over a certain number of consecutive times.
If we define a block activity with a block of size b as: $\overrightarrow{act}(i,b) = \left[\overrightarrow{act}(i), \overrightarrow{act}(i+1), ..., \overrightarrow{act}(i+b-1) \right]$
$\overrightarrow{act}(i,b)$ is itself a matrix of size $N \times b$ and corresponds to the b-1 consecutive activity vectors (column vectors of matrix A) starting with time i.
And the corresponding new SRDb matrix is defined as:

$$SRDb(i,j) = dist(\overrightarrow{act}(i,b), \overrightarrow{act}(j,b))$$

This matrix will provide a distance matrix over the time shifted activity of the ensemble but considering time sliced blocks of activity. Actually, the distance function is computed on two matrices and it can be performed using the distances defined for matrices or building two vectors formed with the concatenation of the columns of each matrix (a vector of length $N \times b$) and computing the distance between both resulting vectors.

All the computations performed on SRD can be extended to SRDb, with the needed care in interpretation of results.

The use of the SRDb matrix is more appropriate if we try to study patterns formation embedded in b columns of neurons activity.

In order to have a global number for the synchronicity of the ensemble (multivariate measure) we can calculate the entropy of the distribution obtained (either normal or weighted and with any block size) and compare the value with the entropy of a uniform distribution. In this manner, the final multivariate synchrony index will be:

$$I_{SRD} = \frac{\ln(\mathrm{M}) - \left(-\sum_{\Delta i} p_{\Delta i} \ln p_{\Delta i} \right)}{\ln(\mathrm{M})};\ \Delta \mathrm{i} = 1 \ldots \mathrm{T} \quad (112)$$

Where M is the number of bins used for the histogram to calculate the probability distribution.

This index has values between 0-1, with zero for no synchrony (probability distribution is a uniform distribution) and one for complete synchrony (probability distribution is a delta distribution)

An interesting point is the dependence of the probability distribution of similarity values in the Similarity Matrix (or similarly the Distance Matrix) with the value chosen for the threshold: Γ.

Figure 4.2.11.2-4 shows the spikes of 100 neurons, where, at the middle of the registered time one can observe an increase in synchronization.

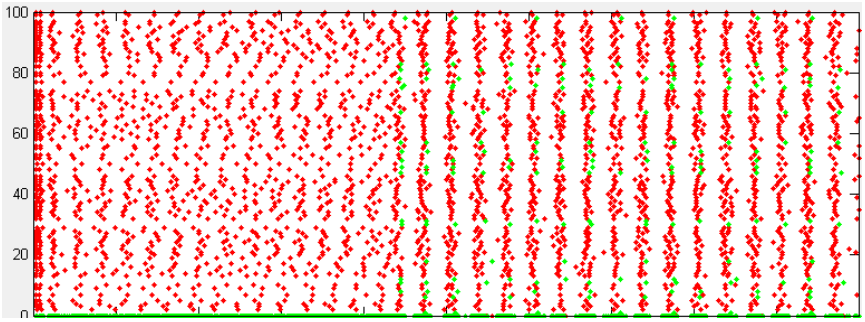
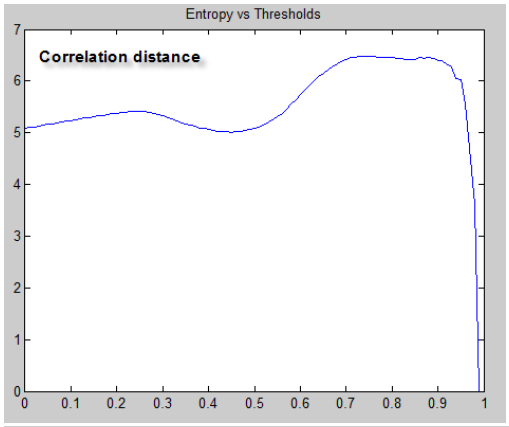


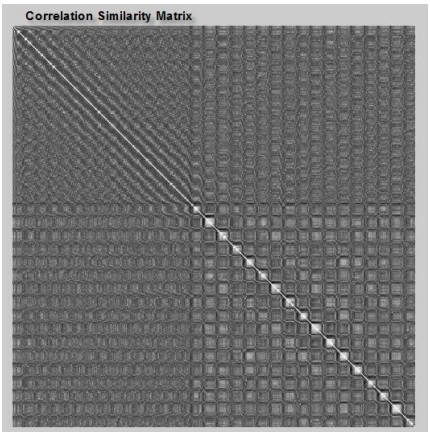
Figure 4.2.11.2-4 Spikes signals (100 neurons)

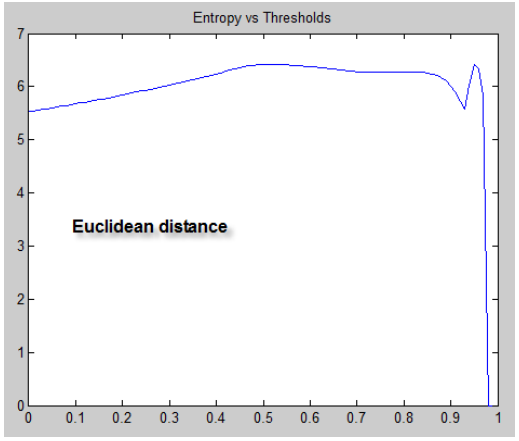
The following figure 4.2.11.2-5 corresponds to the similarity matrices for this ensemble of signals; they are obtained with different distance measures. At the right can be seen the similarity matrix for each distance measure and to the left the function of entropy change versus the threshold value. The entropy is applied to the distribution of similarity values of the Similarity Matrix (to the right).

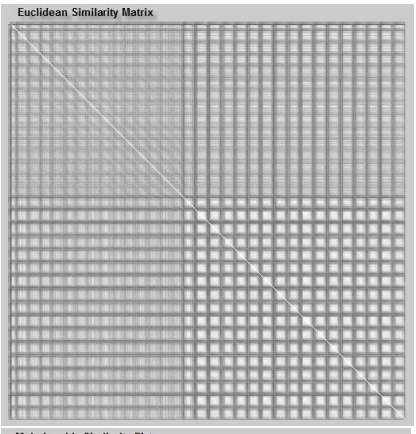
Interestingly all the entropy vs. threshold functions show a down-up-down behavior, except for the jaccard and cosine distances. In particular the jaccard distance present a monotonic decreasing function which is the most intuitive result, as one can expect that as the number of values in the matrix is reduced the value options (uncertainty) will decrease.

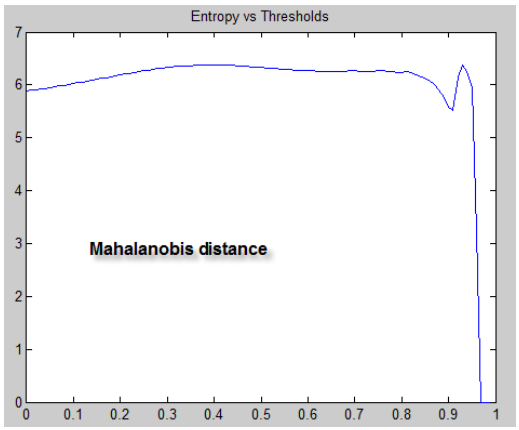
The increase in entropy for an intermediate range of values is a non-intuitive result and can be used to find thresholds which have low entropy at intermediate values, that is, they don't take away too many values in the Matrix, and at the same time can be used to get rid of noise. In this case, the threshold would be chosen at the minimum intermediate value for the function entropy vs. threshold.

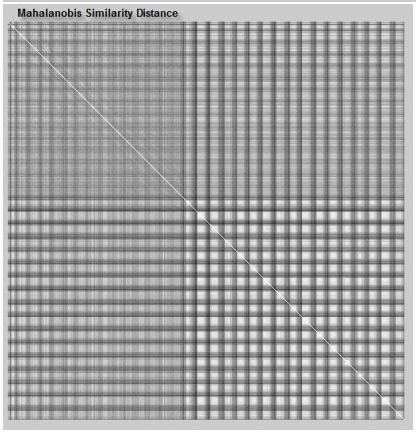

Entropy vs Thresholds
Correlation distance
0 1 2 3 4 5 6 7
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

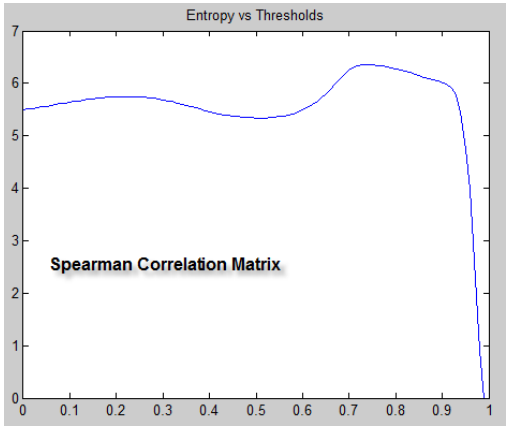

Correlation Similarity Matrix

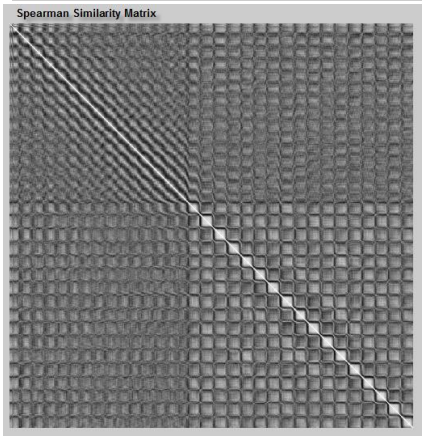

Entropy vs Thresholds
Euclidean distance
0 1 2 3 4 5 6 7
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1


Euclidean Similarity Matrix


Entropy vs Thresholds
Mahalanobis distance
0 1 2 3 4 5 6 7
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1


Mahalanobis Similarity Distance


Entropy vs Thresholds
Spearman Correlation Matrix
0 1 2 3 4 5 6 7
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1


Spearman Similarity Matrix

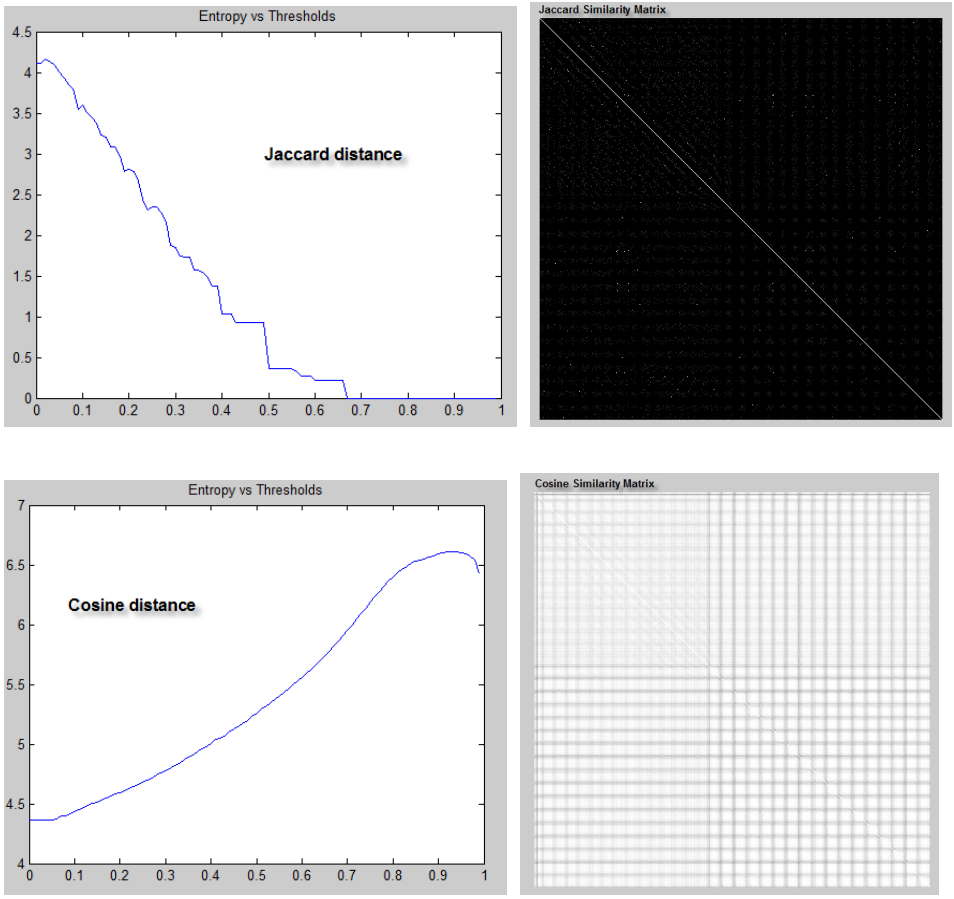


Figure 4.2.11.2-5 Entropy vs. threshold behavior under different distance measures.

4.2.12 Information distance

There are a group of measures that try to establish a distance between two signals (or symbol strings) by the comparison of the information quantity inside the signals.

In order to do the comparison we use the Kolmogorov complexity K(x) of a string x. The Kolmogorov complexity [96][97] is defined as the length of the shortest Turing machine program, that, given the empty string as input, generates x on its output tape and then halts.

Kolmogorov complexity is not computable, and it can be considered as an idealized measurement of the informational quantity given in a string (or signal).

To use this concept to compare the information quantity of two signals we define the Normalized Information Distance (NID)[95]:

$$NID(x,y) = \frac{\max\{K(x/y), K(y/x)\}}{\max\{K(x), K(y)\}} \quad (113)$$

Where $K(x/y)$ is the conditional Kolmogorov complexity, i.e. the size of the smallest Turing machine, that, given y as input, generates x on its output tape and then halts.

NID can have values between [0-1], with zero for the comparison of two equal signals and 1 for the comparison of a signal with the empty signal.

Neither Kolmogorov complexity nor NID are computable.

4.2.12.1 Normalized compression distance (NCD)

In order to generate a computable measure that approximate NID, the normalized compression distance (NCD) was developed.

Considering the smallest program in the Turing machine as the maximum possible compression of the string x, we can approximate K(x) with C(x) which is the size of the output of a compressor (Huffman, BZIP2, Lempel-Ziv, or any commercial one) when applied the signal x.

Additionally the expression $\max\{K(x|y), K(y|x)\}$ could be approximated by $C(xy) - \min\{C(x), C(y)\}$, where xy is the concatenation of x and y. In this way, it can be seen how good the compression algorithm can use the information given in one string to better compress the other one and vice-versa. By subtraction of $\min\{C(x), C(y)\}$ the term approximates the remaining compressed size of the more complex string if the information in the other one was used. Therefore the term should be smaller, the more similar the two strings are.

Finally, the Normalized Compression Distance is defined as [93][98]:

$$NCD(x,y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}} \quad (114)$$

And, an alternative formulation [94]:

$$NCD(x,y) = \frac{\max\{C(xy) - C(x), C(yx) - C(y)\}}{\max\{C(x), C(y)\}} \quad (115)$$

Similarly as NID, the NCD can have values between [0-1], with zero for the comparison of two equal signals and 1 for the comparison of a signal with the empty signal.

4.2.13 Point process synchronization

The following methods are synchronization measures applied to point process signals. Examples of point processes are spike trains produced by neurons, arrival of customers to a shopping line, etc…

The point process signal is usually associated to the occurrence of an event.

The methods presented are based in considering the spike train as a point in an abstract metric space and to determine the distance between two spike trains as a dissimilarity

distance measure in this metric space. Inferring that, two spike trains highly synchronized will have a small distance between them.

We divide the methods presented in two groups [45]:

- Time-scale dependent: in these methods there is a parameter which determines the temporal scale in the spike trains to which the distances are sensitive. While in one limit of the parameter range these distances are sensitive to the difference in spike number, they detect spike coincidences in the other limit. These limits reflect the characteristics of a rate code and a coincidence code, respectively.
  Methods belonging to this group are: Victor-Purpura distance, Van Rossum distance and Schreiber similarity measure.
- Time-scale independent: in these methods there is no parameter that defines the temporal scale associated to the defined distance. These methods are usually simple to compute, and can be preferable in applications to real data for which there is no validated knowledge about the relevant time scales, or there are several time scales present in the same data (e.g. regular spikes plus bursting).
  Methods belonging to this group are: Event synchronization, ISI and SPIKE distance and SES.

4.2.13.1 Victor-Purpura distance

In [10] [12] is presented a distance defined between two spike trains in terms of the minimum cost of transforming one spike train into the other by means of just three basic operations:

- spike insertion which has a cost of 1
- spike deletion which has a cost of 1
- and shifting a spike by some interval Δt, which has a cost of $q \times |\Delta t|$

The cost per time unit q sets the time scale of the analysis. For q=0 the distance is equal to the difference in spike counts, while for large q the distance approaches the number of non-coincident spikes, as it becomes more favorable to delete and reinsert all non-coincident spikes rather than shifting them. Thus, by increasing the cost, the distance is transformed from a rate distance to a temporal distance.

Additionally to this distance which is sensitive to the timing of individual spikes, two other cost-distances are defined in terms of interspike intervals and groups of spikes with a specific temporal pattern (“motif”). In these cases, it has been extended the set of operations allowed. In the former case, it has been added an operation that consists of changing the length of an interspike interval by an infinitesimal amount Δt. This step has cost $q \times \Delta t$. In the latter case, the operation added allows joint shifting of any number of spikes, all in the same direction, by the same amount Δt. This step has also a cost $q \times \Delta t$.

In order to compute the cost it is used the Sellers algorithm [46] which was introduced by Sellers for calculating the distance between two genetic sequences (i.e. a sequence of nucleic acid codons). To apply the Sellers algorithm, the spike train, considered as a sequence of interspike intervals, corresponds to a sequence of nucleotides in a DNA segment.

In the following diagram (figure 4.2.13.1-1) is presented a minimal-cost path of elementary operations connecting two spike trains S_a and S_b

Figure 4.2.13.1-1 Operations performed to transform spike train S_a in S_b

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html

4.2.13.2 Victor-Purpura distance -Multivariate

In [50] is presented a method that applies the Victor-Purpura measure to signals coming from a population (ensemble or multi-unit) of neurons.

This method is a block-multivariate method (see Section 4.3.1).

The block-multivariate measures are mainly used to identify and uncover the coding of different sensory inputs in a population of neurons. That is, it is used to try to determine if a neuron population is able to produce low block-multivariate distances for similar inputs and big distances for very different inputs (discrimination vs. repeatability). The signals obtained for each sensory input are called the responses.

To extend the analysis to multiple neurons, two changes are required. First, we represent a simultaneously recorded response of multiple neurons as a single sequence of labeled events. That is, a multi-unit response is considered as a “labeled spike train,” where a label assigned to each spike indicates its neuron of origin. For a population of L neurons, we use the integers 1, 2,. . ., L as labels. These labels are merely abstract tags—no serial ordering is implied. Second, we introduce a new parameter k into the metric. The new parameter k describes the importance of the neuron of origin of a spike, much as q describes the importance of the time of a spike (see Section 4.2.13.1).

Metrics used to characterize coding in such labeled spike trains allow all of the elementary transformations used by the single neuron metric (insertion of a spike,

deletion of a spike, and moving a spike). In addition, these metrics include an elementary step in which a spike's label (i.e., neuron of origin) is changed. This transformation is assigned a cost of k.

The dimensionless parameter k thus indicates the importance of distinguishing spikes that are fired by different neurons. When k = 0, the multi-unit metric reduces to the single-unit metric described earlier because there is no cost for reassigning the neuron of origin of a spike. This amounts to considering the neuron of origin to be irrelevant to coding, a situation we designate as a "summed population" code. When $k \geq 2$, spikes from different neurons are never considered to be similar because the cost of matching them (by changing the label of either one) is no less than the cost of deleting a spike from one neuron's response and inserting it into another neuron's response. Thus for $k \geq 2$, the distance between two multi-unit (two ensembles) responses becomes equal to the sum of the distances between responses of corresponding individual neurons computed by the single-unit metric (that is, sum for all neurons of bivariate cases between neuron j in ensemble 1 and neuron j in ensemble 2). Because each neuron's activity is considered independently and no interchangeably labeled, we designate this situation a "labeled line" code.

The range of values of k from 0 to 2 provides a continuum between the extremes of the "summed population code" (k = 0) and the "labeled line code" (k = 2). For intermediate values, spikes in two multi-unit responses that occur sufficiently close to each other in time can be seen as similar even if they are fired by different neurons.

Summing up, in the "summed population" scenario, for each response, the spike trains from different neurons are superimposed before the distances between different responses are calculated. Meanwhile, in the "labeled line code" scenario, the distances between different responses are calculated separately for each neuron and then added.

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html

4.2.13.3 Van Rossum distance

In this method [45] the two time series $x(n)$ and $y(n)$ are transformed to continuous time series by performing a convolution of the discrete time series with the exponential kernel:

$$H(t)\exp(-\frac{t}{\tau_R}) \quad (116)$$

Where $H(t)$ is the Heaviside function and τ_R is the time scale parameter.

After applying the kernel to the spike train signal we obtain two continuous signals: $x(t)$ and $y(t)$. From these resulting signals, the Van Rossum distance D_R can be calculated as:

$$D_R(\tau_R) = \frac{1}{\tau_R}\int_0^{\infty}[\tilde{x}(t) - \tilde{y}(t)]^2 dt \quad (117)$$

The closer this distance is to zero, the bigger the similarity of the two signals.

τ_R is the time scale parameter and it is inversely related to Victor-Purpura cost parameter: q.

In [51] is presented an interesting relationship between the Victor-Purpura distance and the kernel method presented in Van Rossum. Actually, the Victor-Purpura method can be translated to a kernel method, under the following kernel:

$$\delta_{2/q}(t) = \begin{cases} q/2 & 0 \le t < 2/q \\ 0 & \text{otherwise} \end{cases} \quad (118)$$

And choosing the L^1 distance, instead of the L^2 as in Van Rossum, that is:

$$D_{VP}(q) = \int dt \left| x(t) - y(t) \right| \quad (119)$$

Where $x(t)$ and $y(t)$ are the spike trains transformed signal under the kernel in equation (118).

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/Source-Code/Event-Sync.html
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html

4.2.13.4 Van Rossum distance -Multivariate

In [51] is presented an alternative multivariate method to the Van Rossum method, which I call the Van Rossum distance-Multivariate (VRM).

This method is a block-multivariate method (see Section 4.3.1).

The block-multivariate measures are mainly used to identify and uncover the coding of different sensory inputs in a population of neurons. That is, it is used to try to determine if a neuron population is able to produce low block-multivariate distances for similar inputs and big distances for very different inputs (discrimination vs. repeatability). The signals obtained for each sensory input are called the responses.

In the same way as in Victor-Purpura (see Section 4.2.13.2), in this measure it is used a parameter that determines the importance of distinguishing spikes fired in different neurons against only focus on the spikes disregarding the neuron of origin. These two

extremes were called in Victor-Purpura (see Section 4.2.13.2) as the "summed population" (SP) and "labeled line" (LL) extremes.

First we will consider the case of an ensemble with just two neurons (X and Y), latter, it will be extended to more than two neurons. The intention is to use VRM to determine the distance in the response of the two neurons (as a block) to two different sensory inputs (two responses).

VRM starts by transforming the X and Y spike trains coming from the responses into the Van Rossum kernel transformed signals.

X^1 will be the spike train for the first response and X^2 will be the spike train for the second response, for Y it is done accordingly.

X^n will be the transformed spike train, n=1,2

The idea behind the new metrics is to map the filtered spike trains X^1 and X^2 into the space of 2-dimensional vector fields. That is, we will map the two transformed signals into two vectors in 2-D. The distance between the two resulting vectors will be the Euclidean distance.

The basis for the new vector space will be:

$$e_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (120)$$

And

$$e_\theta = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} (121)$$

The parameter θ will be the time-scale parameter, which allows to move between the SP and LL scenarios.

And the result of the mapping to the 2-D vector space will be the r_1 and r_2 vectors:

$$\begin{aligned} r_1 &= X^1 e_0 + Y^1 e_\theta \\ r_2 &= X^2 e_0 + Y^2 e_\theta \end{aligned} \quad (122)$$

Where

$$\begin{aligned} X^1 e_0 &= \begin{pmatrix} X^1(t) \\ 0 \end{pmatrix} \\ X^1 e_\theta &= \begin{pmatrix} X^1(t) \times \cos\theta \\ X^1(t) \times \sin\theta \end{pmatrix} \end{aligned} \quad (123)$$

Finally the distance is defined as:

$$D = \|r_1 - r_2\|_{L^2} = \sqrt{\int dt \left\| X^1 e_0 + Y^1 e_\theta - X^2 e_0 - Y^2 e_\theta \right\|^2}$$
$$= \sqrt{\int dt \left\| (X^1 - X^2) e_0 + (Y^1 - Y^2) e_\theta \right\|^2}$$
$$= \sqrt{\int dt \left[(X^1 - X^2) e_0 + (Y^1 - Y^2) e_\theta \right] \left[(X^1 - X^2) e_0 + (Y^1 - Y^2) e_\theta \right]^T} \quad (124)$$
$$= \sqrt{\int dt \left[(X^1 - X^2)^2 + (Y^1 - Y^2)^2 + 2\cos\theta (X^1 - X^2)(Y^1 - Y^2) \right]}$$

Where

$$e_0 e_0^T = 1; \quad e_\theta e_\theta^T = 1; \quad e_0 e_\theta^T = \cos\theta; \quad (125)$$

In this measure a value of $\theta = 0$ produce the SP scenario and a value of $\theta = \frac{\pi}{2}$ produce the LL scenario.

To extend the technique to more than two neurons is immediate, taking care only to select adequate basis vectors in the new N-Dimensional vector space.

The way to select the basis vectors is:
For 3-D

$$\mathbf{e}^1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{e}^2 = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix}, \quad \mathbf{e}^3 = \begin{pmatrix} \cos\theta \\ \sin\theta\cos\theta' \\ \sin\theta\sin\theta' \end{pmatrix} \quad (126)$$

For 4-D

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \cos\theta \\ \sin\theta\cos\theta' \\ \sin\theta\sin\theta' \\ 0 \end{pmatrix}, \begin{pmatrix} \cos\theta \\ \sin\theta\cos\theta' \\ \sin\theta\sin\theta'\cos\theta'' \\ \sin\theta\sin\theta'\sin\theta'' \end{pmatrix} \quad (127)$$

From these examples is clear how to obtain a higher dimensional basis. When the basis vector are chosen in this way, the inner product between any two different vectors is always $\cos\theta$, which is the distance parameter. Like this, we maintain a single parameter, even in a dimension higher than 2-D.

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/Source-Code/Event-Sync.html
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html

4.2.13.5 Schreiber similarity measure

In this method [45] the two time series $x(n)$ and $y(n)$ are transformed to continuous time series by performing a convolution of the discrete time series with the Gaussian kernel:

$$\frac{1}{\sqrt{2\pi\sigma_S^2}}\exp\left(-\frac{t^2}{2\sigma_S^2}\right) \quad (128)$$

Where σ_S is the time scale parameter.

After applying the kernel to the spike train signal we obtain two continuous signals: $x(t)$ and $y(t)$. From these resulting signals, the Schreiber similarity measure can be calculated as the correlation between the transformed signals:

$$C_S(\sigma_S) = \frac{xy}{\left|x\right|\left|y\right|} \quad (129)$$

The closer this distance is to one, the bigger the similarity of the two signals. Reciprocally, the closer this distance is to zero, the smaller the similarity of the two signals

4.2.13.6 ISI and SPIKE distance

In [48][45] are presented the ISI and SPIKE distances. To compute these distances it is performed a previous transformation of the event data (bivariate X and Y) into a quasi-continuous signal, which is discrete in time, but continuous in value according to a, so called, temporal profile $I(t)$, which provides an indication, for each time, of the instantaneous interspike intervals difference between the two signals (for the case of the ISI distance) or the spike times coincidence between the two signals (for the case of the SPIKE distance).
In both cases, from the temporal profile function $I(t)$, the distance measure is obtained by integrating over all the observed period:

$$D = \frac{1}{T}\int_{t=0}^{T} dtI(t) \quad (130)$$

To calculate the temporal profile function $I(t)$ for the ISI distance, the following time-functions are calculated previously:

for each neuron u=X,Y one assigns to each time instant the time of the previous spike

$$t_{\mathrm{P}}^{u}(t) = \max(t_i^u | t_i^u \le t) \quad (131)$$

and the time of the following spike

$$t_{\mathrm{F}}^{u}(t) = \min(t_i^u | t_i^u > t) \quad (132)$$

as well as the interspike interval

$$\nu^u_{\mathrm{ISI}}(t) = t^u_{\mathrm{F}}(t) - t^u_{\mathrm{P}}(t) \quad (133)$$

And from these functions we can obtain the temporal profile:

$$I(t) = \begin{cases} \nu^x_{\mathrm{ISI}}(t)/\nu^y_{\mathrm{ISI}}(t) - 1 & \text{if } \nu^x_{\mathrm{ISI}}(t) \le \nu^y_{\mathrm{ISI}}(t) \\ -(\nu^y_{\mathrm{ISI}}(t)/\nu^x_{\mathrm{ISI}}(t) - 1) & \text{otherwise.} \end{cases} \quad (134)$$

This quantity becomes zero for identical ISI of the two spike trains, and approaches −1 and 1, respectively, if the first or the second spike train is much faster than the other. Since all deviations from identical ISI count equally, the ISI-distance is calculated by temporal averaging over the absolute value $\left|\mathrm{I}(\mathrm{t})\right|$.

In a similar way, it is calculated the SPIKE distance but taking into account the spike timing in the computation of $I(t)$ [48]

The ISI distance is based on the relative length of simultaneous interspike intervals and is thus well-designed to quantify similarities in the neurons' firing rate profiles, however, it is not optimally suited to track complete similarity of spike trains. The SPIKE distance is intended to capture identical synchronization between the two signals.

Both the ISI and the SPIKE distances are bounded in the interval [0,1] . For the latter the limit value 0 is obtained only for perfectly identical spike trains while for the former it is also obtained for periodic spike trains with the same period.

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html
http://www.fi.isc.cnr.it/users/thomas.kreuz/Source-Code/Event-Sync.html

4.2.13.7 SPIKE distance -Multivariate

In [48] is presented a multivariate alternative to the ISI and SPIKE distances, actually, are presented two, but only the second is truly multivariate. The two alternatives are:

1. To calculate all pairwise bivariate distances over all signals and average the obtained value. That implies to calculate n(n-1)/2 bivariate distances.
2. The second method is based in computing, for each time instant, the standard deviations over all previous and all following spike times and divide their mean by the mean instantaneous ISI. And, in order to be more local in time we weight the two terms such that the standard deviation of the spike times that are closer dominates. This way we obtain the instantaneous spike standard deviation:

$$S^{\mathrm{m}}(t) = \frac{\sigma[t^{(n)}_{\mathrm{P}}(t)]_n \langle x^{(n)}_{\mathrm{F}}(t)\rangle_n + \sigma[t^{(n)}_{\mathrm{F}}(t)]_n \langle x^{(n)}_{\mathrm{P}}(t)\rangle_n}{\langle x^{(n)}_{\mathrm{ISI}}(t)\rangle^2_n} \quad (135)$$

Where $\sigma[..]_n$ is the standard deviation over all neurons (1..n) and $\langle .. \rangle_n$ is the average over all neurons.
And

$$\langle x_{\mathrm{ISI}}^{(n)}(t)\rangle_n = \frac{1}{N}\sum_{n=1}^{N} x_{\mathrm{ISI}}^{(n)}(t)$$

Finally, the multivariate SPIKE-distance D_S^m is obtained by integrating over time:

$$D_S^{\mathrm{m}} = \frac{1}{T}\int_{t=0}^{T} S^{\mathrm{m}}(t)\,\mathrm{d}t \qquad (136)$$

The multivariate SPIKE-distance has 0 as a lower bound but does not have an upper bound and in particular can attain values larger than 1.

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html
http://www.fi.isc.cnr.it/users/thomas.kreuz/Source-Code/Event-Sync.html

4.2.13.8 Event synchronization

As presented in [20] there is a computational light method to establish the synchronization of two point process signals.

Considering two event time series x_n and y_n, where n is the sampling interval index $\mathrm{n}=1,\ldots\mathrm{N}$, and event times t_i^x and t_j^y $\left(\mathrm{i}=1,\ldots\mathrm{m_x};\mathrm{j}=1,\ldots\mathrm{m_y}\right)$, event times are the times when the event occurs, and allowing a time lag τ between two 'synchronous' events (which should be smaller than half the minimum inter-event distance to avoid double counting), we can define a measure of the number of times an event appears in x shortly after it appears in y:

$$c^{\tau}(x|y) = \sum_{i=1}^{m_x}\sum_{j=1}^{m_y} J_{ij}^{\tau} \qquad (137)$$

where

$$J_{ij}^{\tau} = \begin{cases} 1 & \text{if } 0 < t_i^x - t_j^y \le \tau \\ 1/2 & \text{if } t_i^x = t_j^y \\ 0 & \text{otherwise} \end{cases} \qquad (138)$$

Similar computations can be done for $\mathrm{c}^{\tau}(\mathrm{y/x})$. Therefore, it is possible to define the following symmetrical and anti-symmetrical combinations:

$$Q_\tau = \frac{c^\tau(y|x) + c^\tau(x|y)}{\sqrt{m_x m_y}}, \qquad q_\tau = \frac{c^\tau(y|x) - c^\tau(x|y)}{\sqrt{m_x m_y}} \tag{139}$$

Which measure, respectively, the synchronization of the events and their delay behavior.

They are normalized to $0 \le Q_\tau \le 1$ and $-1 \le q_\tau \le 1$. We have $Q_\tau = 1$ if and only if the events of the signals are fully synchronized. In addition, if the events in X always precede those in Y, then $q_\tau = 1$.

Then, both parameters can identify the synchronization and delay behavior of two signals.

There is a Matlab code available for this method:
http://www.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html
http://www.fi.isc.cnr.it/users/thomas.kreuz/Source-Code/Event-Sync.html

4.2.13.9 Stochastic Event Synchrony (SES)

As presented in [3][34], the "stochastic event synchrony" (SES) is an approach to measure the interdependence of two time series. It quantifies the alignment of two point processes (event strings) by means of the following three parameters (for one-dimensional signals): time delay, variance of the timing jitter and fraction of "spurious" events.

This measure is related to "Metric-space analysis" [12], but in SES the synchronization is defined with three parameters instead of one as in [12].

SES may be applied to generic one-dimensional and multi-dimensional point processes.

The pairwise alignment of point processes is tackled as a statistical inference problem, which is solved by applying the max-product algorithm on a graphical model. The SES parameters are determined from the resulting pairwise alignment by MAP (maximum a posteriori) estimation.

To provide an intuitive understanding of the technique, let us consider [34] the one-dimensional point processes X and X' in Figure 4.2.13.9-1 (ignore Y and Z for now). We wish to quantify to which extent X and X' are synchronized. Intuitively speaking, two event strings can be considered as synchronous (or "locked") if they are identical apart from: (1) a time shift δt; (2) small deviations in the event occurrence times ("event timing jitter"); (3) a few event insertions and/or deletions ("spurious events"). More precisely, for two event strings to be synchronous, the event timing jitter should be significantly smaller than the average inter-event time, and the number of deletions and insertions should comprise only a small fraction of the total number of events.

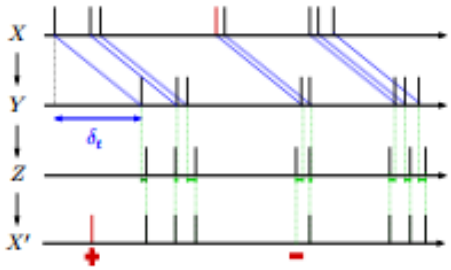


Figure 4.2.13.9-1 Synchronization of point processes X and X'

This intuitive concept of synchrony is illustrated in Figure 4.2.13.9-1. The event string X' is obtained from event string X by successively shifting X over δt (resulting in Y), slightly perturbing the event occurrence times (resulting in Z), and eventually, by adding (plus sign) and deleting (minus sign) events, resulting in X'. Adding and deleting events in Z leads to "spurious" events in X and X' (see Figure 4.2.13.9-1; spurious events are marked in red): a spurious event in X is an event that cannot be paired with an event in X' and vice versa.
The above intuitive reasoning leads to SES, defined as the triplet (δt, st, ρspur), where st is the variance of the (event) timing jitter, and ρspur is the percentage of spurious events.

To obtain these parameters (δt, st, ρspur) we apply an iterative algorithm using the Viterbi algorithm, or alternatively, by applying the max-product algorithm [34].

When the technique is applied to multidimensional signals (usually two dimensional: time-frequency) the method is similar but more complex.

For time-frequency signals: first, the time–frequency transform (Wavelet transform) of each signal is approximated as a sum of half-ellipsoid basis functions, referred to as "bumps" (see Figure 4.2.13.9-2); each bump may be considered as an event on the time–frequency plane, and the resulting bump models E (for signal X) and E′ (for signal X') may be considered as two dimensional point processes ("event sequences in 2-D"), representing the most prominent oscillatory activity. Only bumps whose energy is larger than a threshold T are retained in the bump model.

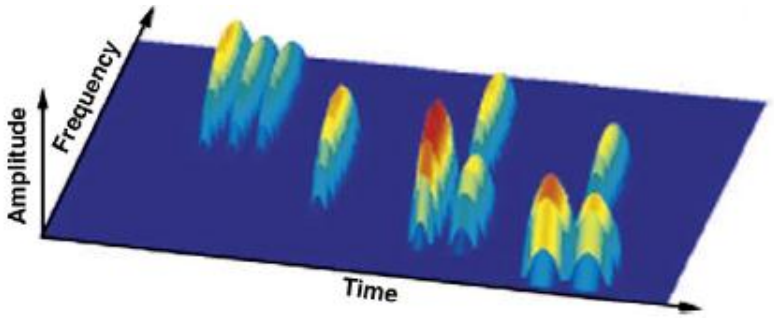


Figure 4.2.13.9-2 Bump modeling. 2-D time-frequency signal (usually obtained with a wavelet transform)

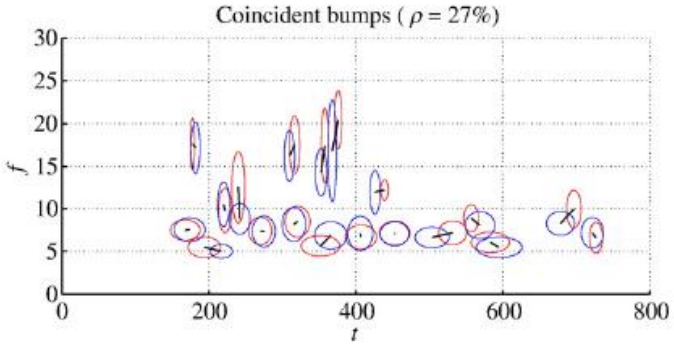


Figure 4.2.13.9-3 Bump models of two signals (red and blue)

Next the two bump models (E and E') are aligned (see Figure 4.2.13.9-3): bumps in one time–frequency map may not be present in the other map ("non-coincident" bumps); other bumps are present in both maps ("coincident" bumps), but appear at slightly different positions on the maps.

Stochastic event synchrony (SES) consists of five parameters (for two dimensional time-frequency signals) that quantify the alignment of two bump models:
• ρ: fraction of non-coincident bumps,
• δt and δf: average time and frequency offset respectively between coincident bumps,
• st and sf: variance of the time and frequency offset respectively between coincident bumps.

The parameters ρ and st are the most relevant for SES, since they quantify the synchrony between bump models and hence the original time–frequency maps.

Low ρ and st implies that the two time–frequency maps at hand are well synchronized.

4.3 Methods classification

In this section it is presented some global characteristic of the methods and the relationships between them.

4.3.1 Global vs. Local

The global methods to detect synchrony in multivariate signals are the methods that provide a measure of the whole ensemble of signals. The local methods are the methods that provide a measure on pair or sets of signals.

From the list of methods, the following are global methods [3]:

- GFS
- Omega complexity
- S-estimator
- Multivariate mutual information
- Phase index-multivariate
- Frequency flows analysis (FFA)
- SPIKE distance - Multivariate
- Victor-Purpura -Multivariate

- Van Rossum -Multivariate
- Eigenvalue decomposition
- Empirical mode decomposition
- MSSA
- Matrix similarity measures (Frobenius, WSSVD,..)
- Joint recurrence plot (JRP)
- Shifted recurrence diagram

All other methods are local.

From the list of global methods it is necessary to do a further distinction between two different methods:

1. Multivariate: Multivariate methods provide a measure of synchronization among all the signals in the ensemble of signals, considering the signals ensemble as a block. These measures provide a single scalar value or a single vector value to the level of synchrony of the block, taken as a unity.
2. Block-multivariate: Another different multivariate type of measures tries to measure the level of similarity between two ensembles (the level of synchrony between two blocks). These latter measures could be considered as a bivariate case when looking at each ensemble as a univariate (but multidimensional) signal. Nevertheless, if this approach is taken, the classical bivariate methods cannot be applied directly in order to calculate the distance, as the position of each specific signal (and the interrelationship between the signals) in the block can be important. This is the reason for the necessity of specific method to calculate a distance measure between signals coming from two ensembles.
   These latter multivariate measures will be called in this work as block-multivariate.
   The block-multivariate measures are mainly used to identify and uncover the coding of different sensory inputs in a population of neurons. That is, it is used to try to determine if a neuron population is able to produce low block-multivariate distances for similar inputs and big distances for different inputs (discrimination v.s. repeatability).

From the list of global methods, the block-multivariate ones are:

- Victor-Purpura -Multivariate
- Van Rossum –Multivariate
- Phase index-multivariate
- Matrix similarity measures
- Shifted recurrence diagram

4.3.2 Target information

An additional distinction can be done considering the target information that the technique try to obtain from the data:

1. Ensemble synchrony over space: These methods correspond to the multivariate methods when the signals analyzed are: $X_n(d);\ n=1...N;\ d=1...D;$ where D is the number of nodes (neurons) and N is the maximum time, and d and n are the index for node and time, respectively.
2. Ensemble synchrony over time: These methods correspond to the multivariate methods when the signals analyzed are: $X_d(n);\ n=1...N;\ d=1...D;$ where D is the number of nodes (neurons) and N is the maximum time, and d and n are the index for node and time, respectively. The relationship between synchrony over time and space is:

$$X_d(n) = X_{dn}$$
$$X_n(d) = X_{nd}$$
$$X_{dn} = X_{nd}^T$$

3. Ensemble block synchrony: These methods correspond to block-multivariate methods (see section 4.3.1)
3. Event detection: In these types of measures we are interested in identifying if a signal (or an ensemble of signals) is encoding the presence of a particular input, that is, if the behavior of neural signals has changed after the application of the input. These measures are usually employed in identifying ERPs (Event Related Potentials) produced in a neural population after receiving a sensory stimulus. Symbolic dynamics is usually employed for event detection.
4. Pattern repetition: These methods correspond to the methods presented in section 5. The objective here is to discover patterns which occurrence is unusually high. The pattern can be distributed over time and space, and may not be periodic.
5. Cluster identification: The target is to identify synchronized nodes forming clusters of synchronization, the number of clusters and the nodes integrated in each cluster. These methods correspond to the methods presented in section 4.2.9 .

4.3.3 Correlation between methods

As presented in [3] the synchronization detection methods form clusters (families) when considering correlation between measures produced by the different methods.

It is clear [3] that the majority of measures are strongly correlated (or anti-correlated) with each other. In other words, the measures can easily be classified in different families (See Figure 4.3.3-1).

The following 9 families of measures are identified [3]:

(1) Correlation coefficient, cross-entropy, wav-entropy, state space measures (Nk, Hk, Sk, Ω, S-estimator), mutual information (in time and frequency domain), divergence measures, MVAR coherence, Hilbert and wavelet phase
(2) GFS,
(3) Phase coherence
(4) EMA and IPA (phase measures),

(5) Partial coherence (PC) and direct DTF (Granger),
(6) PDC and DTF (Granger),
(7) Full-frequency DTF (Granger),
(8) st (SES),
(9) ρ (SES).

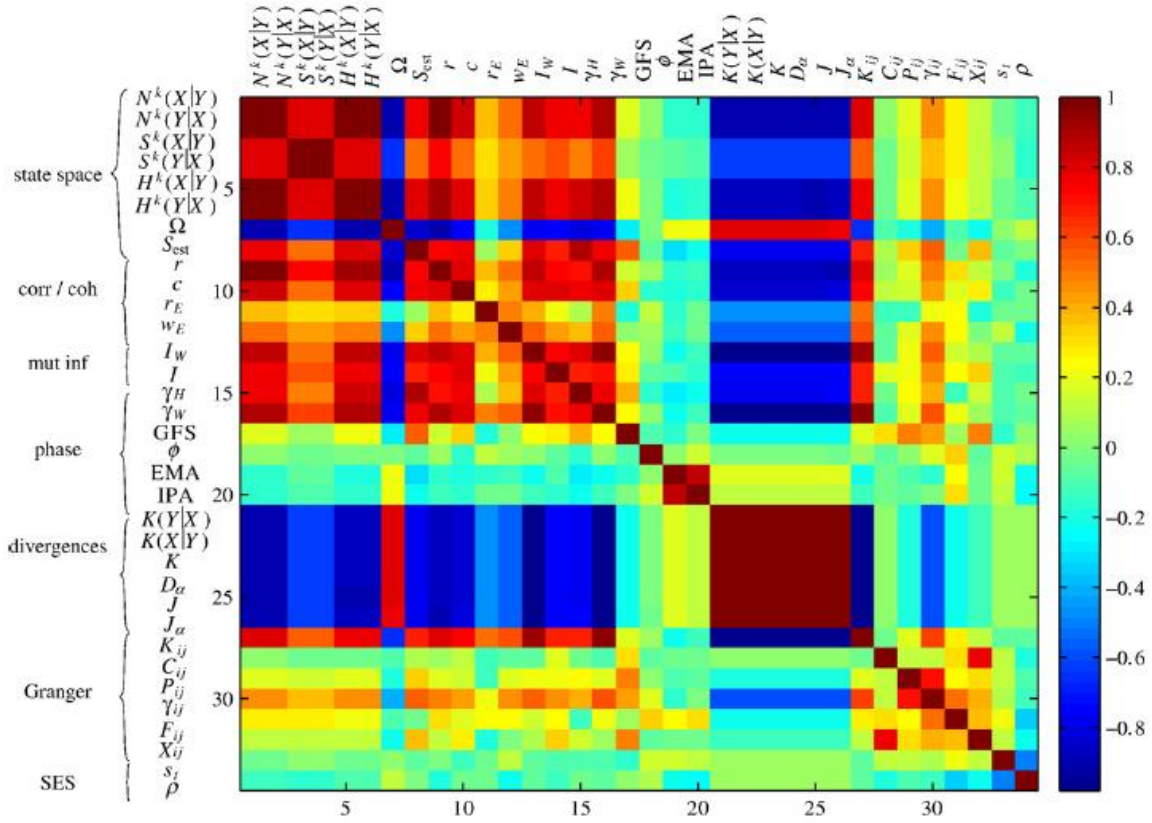


Figure 4.3.3-1 Correlation between the synchrony measures (red and blue indicate strong correlation and anti-correlation respectively)

Each family of measures captures a different kind of interdependence. So, it is not necessary to apply all the measures to a specific scenario, but it may be enough to evaluate only a few measures from each of the families. The proposed measures are [3]:

- Correlation coefficient
- Phase synchrony indices
- Granger measures
- SES

The above presented information on correlation between methods corresponds to methods presented in [3], the study has not been extended to other additional methods presented in this work. It is considered as a possible extension to the present work.

5 Recurrent patterns

We consider a recurrent (usually periodic but not necessarily) pattern as a repeating spatiotemporal sequence of signals or events which occurrence is unusually high. To consider what is a normal occurrence rate, it is used, as reference, an stochastic process which assumes independence in the generation process (usually Bernoulli, Poisson,…)

The identification and nature of recurrent patterns is different (an even more difficult) that the identification of synchronization.

A system may not present any aspect of previously defined synchronization but can present, nevertheless, interesting recurrent patterns.

The additional difficulty with the identification of recurrent patterns is:

- To identify the pattern to look for: usually this pattern is not known in advance, what can be translated to a combinatorial explosion in the search space.
- The complex spatiotemporal structure of the searching process: the pattern can be present in space, time or both.
- Repetition period can change with time (what makes it recurrent and not periodic).

5.1 Synfire chains

Patterns of precisely timed and spatially distributed spikes are observed in different neuronal systems. The patterns respond to external stimuli (events) and are so candidates to explain neural computation [28]. Their dynamical origin, however, is unclear. One possible explanation for their occurrence is the existence of excitatory coupled feed-forward structures, which are called synfire chains [64][28], which are embedded in a network of otherwise random connectivity and receive a large number of random external inputs. Other possible explanation is that this kind of behavior arises naturally in a recurrent neural network.

Synfire chains are also good candidates to explain signal propagation inside a network without incurring in avalanche or die out phenomena. If we have an input signal presented to a network and we want that the network respond to the signal it is needed that the signal is propagated inside the network. This propagation is difficult, as it is shown with an easy model [55]: If a neuron produces an spike in a network and the spike produces a new spike with a probability p in each of the n neurons connected to it, after the first spike we will have (np) new spikes, and at the next stage we will have $(np)^2$, and after m stages we will have $(np)^m$. If (np) is greater than 1 we will get an avalanche and if smaller the signal will die out. So (np) has to be very close to 1, implying an unstable condition.

Synfire chains can resolve this problem as they provide a path for a signal to propagate inside a network. They are groups of neurons coupled in a feed forward manner that support synchronous signal propagation. They are arranged in several layers, where one particular neuron can belong to various layers, and in which every neuron of a given layer makes synapses onto every neuron of the next layer. With this architecture it is possible to propagate a signal through the network.

Focusing in synfire chains activity, in [67][65][66] are presented various methods to detect the synfire patterns activity.

One neuron may be involved simultaneously in various synfire chains and a synfire chain may include the same neuron twice or even a higher number of times. That makes especially difficult the detection of significant patterns (not produced by chance).

In figure 5.1-1 is presented the process of synfire chains activation and periodic patterns formation.

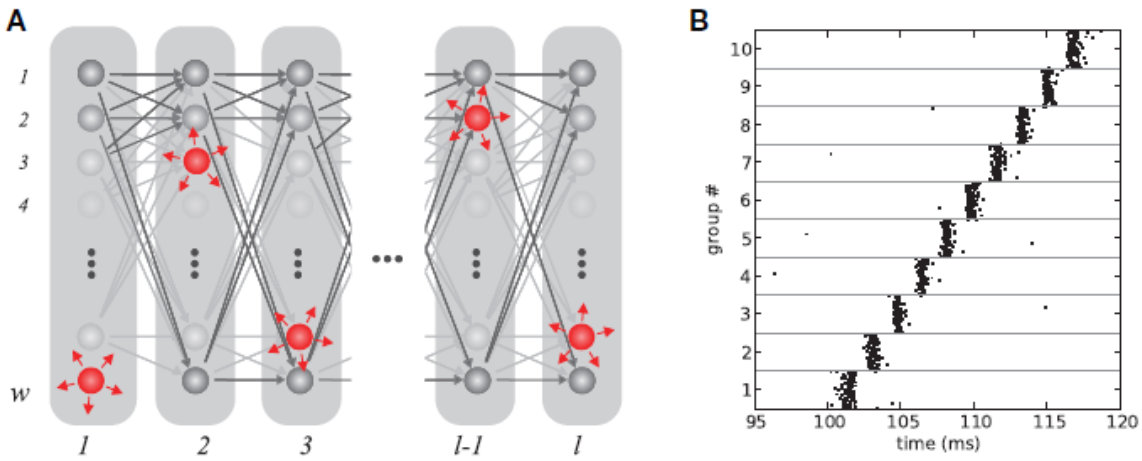


Figure 5.1-1 Synfire chains activation and periodic patterns formation [99]. Number on bottom-left is the group index. Each group in the right diagram presents the activation of each neuron (1...w).

The methods used to detect patterns usually are based in sorting the spike patterns whose duration is less than some criterion into groups according to pattern complexity and repetition number and identify those groups whose count significantly (using statistical hypothesis test methods) exceeds corresponding counts found in surrogate data [67][65].

In [66] is presented a different method to detect synfire chains which is based in the construction of an intersection Matrix M. Assuming that we have N neurons each producing a time-series of spikes (values 0/1) of length T. The element $\mathrm{M}(\mathrm{i},\mathrm{j})\ \mathrm{i},\mathrm{j}=1..\mathrm{T}$, of matrix M is computed as:

$$M(i,j) = \frac{|S(i) \cap S(j)|}{\min(|S(i)|, |S(j)|)} \quad (140)$$

Where $S(i)$ are the indices of neurons fired at time i. And |..| indicates the number of elements. That is, $\mathrm{M}(\mathrm{i},\mathrm{j})$ corresponds to the number of neurons which are commonly being fired in time i and j, divided by the size of the minimum set of total neurons fired in time i or j.

$\mathrm{M}(\mathrm{i},\mathrm{j})$ can also be calculated as the inner product of two vectors $\mathrm{V}(\mathrm{i})$ and $\mathrm{V}(\mathrm{j})$ $\mathrm{i},\mathrm{j} = 1..\mathrm{T}$, built as vectors of length N with component values 0/1 depending on the corresponding neuron being fired.

This matrix is related to recurrence plots presented in section 4.2.11.1 but here the matrix consider the difference in the group activity of the N neurons and not the distance of the state space vectors of the associated dynamical system. It is also related to the shifted recurrence diagram in section 4.2.11.2 with a jaccard distance.

In figure 5.1-1 is presented the intersection matrix of an ensemble of neurons with a number of synfire chains artificially built in the network. The diagonal lines correspond with the activation of synfire chains and the analysis of these lines provides information on number and characteristics of the synfire chains [66].

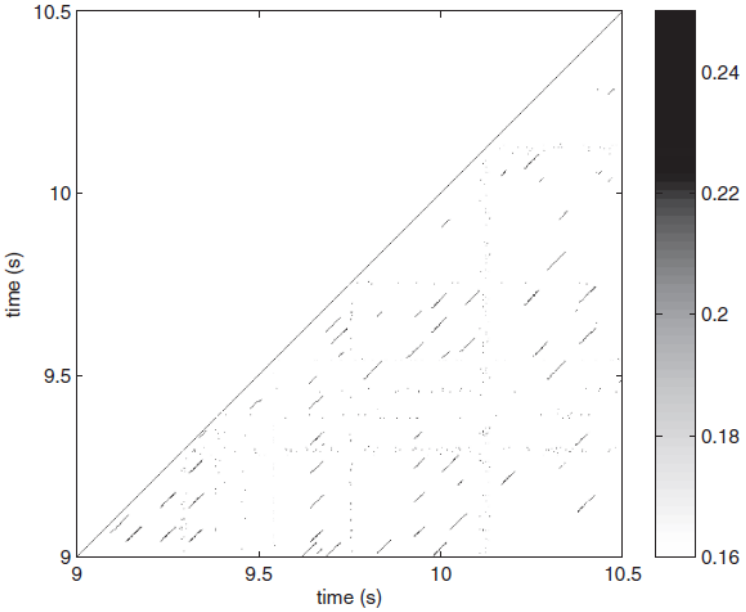


Figure 5.1-1 Illustration of the Intersection matrix M (being M symmetric only half is showed)

5.2 Unitary events

In [49] is presented a method to detect unusual occurrence of simultaneous spikes in a neural ensemble. It is developed as a method to detect the presence of unusual spike coincidences in simultaneously recorded multiple single-unit spike trains and to evaluate their statistical significance.

We refer to such unusual coincidences as unitary events and define them as those joint spike instances that recur more often than expected by chance.

Each unitary event is denoted by v^k, with $\mathrm{k}=1,..,2^N$, that is, a unitary event is one of the possible patterns of 1 and 0 that can be formed at each particular time in the simultaneous recording of the N neurons (see figure 5.2-1).

The probability of such a unitary event is (considering that the spikes of different neurons are independent and the spikes from a neuron at a particular time are independent of previous times):

$$P_k = \prod_{i=1}^{N} P\left(v_i^k\right),$$

$$\text{with } P\left(v_i^k\right) = \begin{cases} P\left(v_i = 1\right), & \text{if } v_i^k = 1 \\ 1 - P\left(v_i = 1\right), & \text{if } v_i^k = 0. \end{cases} \quad (141)$$

Where $P(\mathrm{v}_i = 1)$ is the probability of a spike in neuron i. This probability is presented to the right in figure 5.2-1 and can be different for different neurons. It is obtained by the ratio of spikes present in a predetermined window of time for that particular neuron.

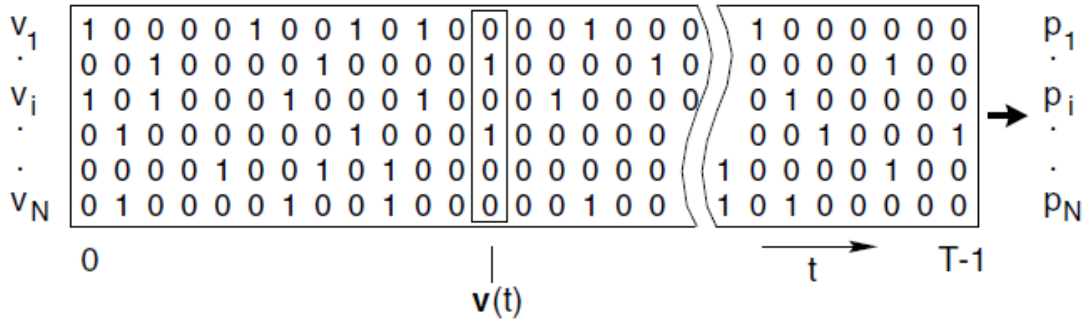


Figure 5.2-1 Illustration of unitary events in a neural population of N neurons recorded in T time intervals.

The task now is to develop a tool that enables us to judge whether the empirical number of occurrences n_k^{emp} of a particular coincidence pattern v^k deviates significantly from the expected number n_k^{pred}

We can write the probability of finding each pattern v^k exactly n_k times in the observation interval (T) directly in terms of the P_k:

$$\psi(n_k; P_k; T) = \frac{T!}{n_k! \cdot (T - n_k)!} \cdot P_k^{n_k} \cdot (1 - P_k)^{T - n_k}, \quad k = 1, \ldots, m. \tag{142}$$

Since the number of time steps T is usually large and the associated probabilities P_k are small, while their product $P_k \times T$ remains moderate, equation (142) can be approximated by the Poisson distribution:

$$\psi(n_k; P_k; T) = \frac{(P_k \cdot T)^{n_k}}{n_k!} \cdot \exp(-P_k \cdot T), \quad k = 1, \ldots, m.$$

Here, $P_k \times T$ is the rate parameter of the Poisson distribution, defining the expected number of occurrences $n_k^{pred} = P_k \times T$ of the joint spike pattern v^k.

Being able to model the expected number of pattern occurrences under the assumption of independence, it is possible to establish a hypothesis testing to determine the significance (surprise) of a particular repetition number coming from the data. In [49] is presented the hypothesis test to be used.

This method is a multivariate method but detect unexpected number of patterns occurrences, instead of synchronization.

6 Network connectivity and synchronization

6.1 Complex networks

In many works on network synchronization [76][57][16][17][59][68] has been established the crucial influence of the underlying network structure in the synchronization capacity of a multi-node dynamical system.

To study network influence the important aspects of the network are abstracted by the network graph.

The graph of a network can be undirected or directed, and weighted or unweighted. The graphs to be considered here will be, usually, directed and weighted, as neural networks present directed connectivity and the synaptic strength between neurons introduce a weight in the connection.

Directed and weighted graphs are more difficult to study and there are fewer works available that analyze them, so, I will try to present results on directed and weighted graphs as far as possible, or, in case they are not available (or I am not aware of them) I will present results on the alternative undirected/unweighted graphs.

In order to study a weighted network, the values of the weights given in the graph are very important, in fact, as presented in previous sections (4.1 and 4.2.11.1) the typical

path to synchronization from phase synchronization, to lag and finally complete synchronization is usually performed with successive increases in the coupling strength of the network what is done increasing the weight values.

In order to establish the more important network parameters that can influence synchronization, the following are considered:

Average path length

In a network, the distance d_{ij} between two nodes i and j is defined as the number of edges along the shortest path connecting them. The diameter D of a network, on the other hand, is the maximal distance between any pair of its nodes.

The average path length L of the network, then, is defined as the distance between two nodes, averaged over all pairs of nodes. Therefore, L determines the effective size of a network, the typical separation of pairs of nodes.

One of the most important discoveries in the field of complex networks is that the average path lengths of most large-scale real complex networks are relatively small, even though they have much fewer edges than a globally coupled network with equal number of nodes. This smallness is usually referred to as the small-world effect.

Degree distribution

The degree k_i of a node i is the total number of its connections. The average of k_i on all i (all nodes) is called the average degree of the network, and is denoted as $\langle k \rangle$. The spread in node degree is characterized by a distribution function P(k), which gives the probability that a randomly selected node has exactly k edges.

In case of directed network it has to be taken into account both in and out-degree.

For a regular lattice network, all nodes will have the same number of connected nodes, so the degree distribution will be a single peak in the number of nodes connected to any single node.
For a random network the degree distribution will be that of a Poisson distribution:

$$P(k) \simeq e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!}$$

This distribution has not a fat tail, meaning that for big k the probability of having a node with this degree is negligible small.
For real networks what has been observed is that the degree distribution does not follow this law, but they have a fat tail distribution that follows a power law:

$$P(k) \sim k^{-\gamma}$$

This distribution allows for a few nodes of very large degree. Since power-laws are free of a characteristic scale, a network with a power-law degree distribution is called a scale-free network.

Clustering coefficient

The clustering coefficient C of a network is the average fraction of pairs of neighbors of a node that are also neighbors of each other. Suppose that a node i in the network has k_i edges, which connect it to other nodes, which are the neighbors of node i. Clearly, at most $k_i(k_i - 1)/2$ edges can exist between them, and this occurs when every neighbor of node i is connected to every other neighbor of node i. The clustering coefficient C_i of node i is defined as the ratio between the number E_i of edges that actually exist between these k_i nodes and the total possible number $k_i(k_i - 1)/2$, that is:

$$C_i = \frac{2E_i}{k_i(k_i - 1)}$$

The clustering coefficient C of the whole network is the average of C_i over all i. Thus, C measures the cliquishness of the network. Clearly, $C \leq 1$, and C equals 1 if and only if the network is globally coupled. In a completely random network $C \sim N^{-1}$, which is very small as compared to most large networks. It has been found that most large-scale real networks have a tendency to clustering, in the sense that their clustering coefficients are much greater than $O(N^{-1})$, although they are significantly less than one.

To study network influence usually are considered the following main network categories:

Regular networks

Regular graphs have all nodes with the same degree (number of neighbors). The degree distribution has a peak at the number K of neighbors per node. Among the possible regular networks some are especially important:

- Globally coupled network: The network has a K=N, where N is the number of nodes.
- Nearest-neighbors coupled network (lattices): in this graph each node is connected to a certain number of neighbors. It can be built using a ring to arrange the nodes and to connect each node to the K following nodes. Or, the nodes can be arranged in a square grid where each node is connected to the nearest 4 or 8 neighbors.

These networks have high clustering coefficients and high average path lengths. An exception to this behavior is the star configuration which has high clustering coefficient and low average path length.

Random networks

They are also called Renyi-Erdos (RE) networks. They are at the opposite end of the regular networks. They are defined by two parameters: N, number of nodes and p, probability of having an edge connecting any two nodes. The edges that connect the nodes are random and the way to construct them is to peak every possible edge between two nodes and with probability p leave the edge and with probability (1-p) to erase it.

The average degree of these networks is: $\langle k \rangle \cong pN$ and the average path length is: $L \sim \frac{\ln(N)}{\ln\langle k \rangle}$ and the clustering coefficient is: $C = p = \frac{\langle k \rangle}{N} \ll 1$. The degree distribution follows a Poisson distribution.

That is, they show a low clustering coefficient and a low average path length. The logarithmic increase in the average path length with the size of the network is typical of the small world effect.

Small-world networks

The regular lattice is clustered, but does not exhibit the small-world effect. On the other hand, the random graph shows the small-world effect, but does not show clustering.

The networks that are usually found in nature show high clustering and low average path length. To try to reproduce these characteristics Watts and Strogatz (WS) introduced a small-world network model, now referred to as WS small-world model.

This model starts with a nearest-neighbor coupled network consisting of N nodes arranged in a ring and each node i is adjacent to its K next neighbor nodes. In order to have a sparse but connected network at all times, consider $\mathrm{N} \gg \mathrm{K} \gg \ln(\mathrm{N}) \gg 1$ in general. Next, we randomly rewire each edge of the network with probability p. Rewiring in this context means shifting one end of the connection to a new node chosen at random from the whole network, with the constraint that no two different nodes can have more than one connection in between, and no node can have a connection with itself.

In this way we can control the final structure of the network from a complete regular network (p=0) to a random network (p=1).

The important characteristic of these networks is that they have high clustering coefficient and low average path length.

And alternative construction method of a small-world network was proposed by Newman and Watts (NW) [17]. In this method we start from a nearest-neighbor coupled network consisting of N nodes, as in the previous method, but in this case, instead of breaking existing connections with probability p, we add new connections with probability p. In this model a p=0 produces the original nearest-neighbor coupled network and a p=1 produces a global coupled network.

Scale free networks

The degree distribution of the random and small world networks follows an exponential decaying distribution. Such networks are called exponential networks. Nevertheless, complex network in nature seems to have a different degree distribution which is scale frees, that is, they have a degree distribution that follows a power law. Networks with this property are called scale-free.

Barabasi and Albert proposed a method to generate this network (BA network) which is based in starting with a graph with N_0 nodes and m_0 edges, and to add successively new

nodes one by one. Each new node is connected to $\mathrm{m} \leq N_0$ existing nodes (without repeating connections). When choosing the nodes to which the new node connects, it is assumed that the probability Π_i that a new node will be connected to node i depends on the degree k_i of node i, in such a way that:

$$\Pi_i = \frac{k_i}{\sum_j k_j} \quad (143)$$

After n time steps, this algorithm results in a network with $N = n + N_0$ nodes and $\mathrm{m} \times \mathrm{n}$ edges. Growing according to these two rules, the network evolves into a scale-invariant state: The shape of the degree distribution does not change over time
The degree distribution of BA networks is approximately: $P(k) \sim \mathrm{k}^{-3}$, that is a power law with exponent 3.

6.2 Spectral graph methods

This section is dedicated to the study of the eigenvalues and eigenvectors of the adjacency matrix of the network or its Laplacian matrix.

The adjacency matrix A of a network has values $\mathrm{A}_{ij} = 0$ if nodes i and j are not connected and $\mathrm{A}_{ij} = 1$ if nodes i and j are connected. For a weighted network the values of connection are the weights of connections instead of 0/1. Matrix A is symmetric for an unweighted undirected network. The Laplacian of a network is: L=D-A, where D is a diagonal matrix with diagonal values equal to the degree of the corresponding node. In the case of directed graphs, either the in-degree or out-degree might be used, depending on the application.

The eigenvalues of the Laplacian are real and positive (Laplacian matrix is positive semi-definite): $0 = \lambda_1 \leq \lambda_2 .. \leq \lambda_N$, where N is the number of nodes.

The first eigenvalue is always equal to zero (zero sum rows), and the number of times that zero appears as an eigenvalue in the Laplacian is the number of connected components in the graph. The smallest non-zero eigenvalue of L is called the spectral gap. The second smallest eigenvalue of L is the algebraic connectivity (or Fiedler value), this eigenvalue is greater than 0 if and only if G is a connected graph.

It is now clear [58] that commonly used statistical properties of the underlying network of a multi-node dynamical system, including the degree distribution (in and out-degree), degree homogeneity, average degree, average distance, degree correlation, path distances, and clustering coefficient, can fail to characterize the synchronizability of such networks.

The spectral properties of the graph Laplacian of the network provides additional information that is crucial to explain the synchronization capacity of a network [58].

For undirected networks there are bounds defined for the eigenvalues of the Laplacian in terms of degree distribution values. Let d_{min} and d_{max} denote, respectively, the smallest and the largest degree, and let λ_N be the largest eigenvalue of the Laplacian. The following relations are obtained [58]:

$$\lambda_2 \leq \frac{n}{n-1} d_{\min} \leq \frac{n}{n-1} d_{\max} \leq \lambda_N \leq 2d_{\max} \quad (144)$$

Similarly, in terms of the average degree d_{avg} it can be shown that: $d_{avg} < \lambda_N$

From equation (144) we can see that the second eigenvalue does not have a simple bound from below in terms of the vertex degrees.

λ_2 is further bounded by the isoperimetric number $i(A)$ of a matrix A [58]. Let V denote the set of vertices of an undirected network A, and let $S \subset V$ be a subset of vertices, with V−S denoting its complement, and |S| its cardinality. We can define:

$$|\partial S| = \sum_{i \in S} \sum_{j \in V-S} a_{ij} \quad (145)$$

That is, $|\partial S|$ is the (weighted) number of edges between S and its complement. The isoperimetric number $i(A)$ of a graph A is defined by:

$$\min \left\{ \frac{|\partial S|}{|S|} : S \subset V,\ 0 < |S| \leq \frac{n}{2} \right\} \quad (146)$$

To calculate $i(A)$ is an NP-hard problem, but there is a lower bound for it in terms of the second smallest eigenvalue of the Laplacian of A:

$$i(A) \geq \frac{\lambda_2}{2} \quad (147)$$

That implies that:

$$\lambda_2 \leq 2\frac{|\partial S|}{|S|} \quad (148)$$

Where S is any subset of vertices satisfying 0 < |S| ≤ n/2.

This result connecting λ_2 with $i(A)$ is very important as it establishes that λ_2 (which holds the key for synchronization) will be affected by the $|\partial S|$ with the smallest value for any subset of vertices of A. That is the reason why synchronization is not so much dependent on global graph properties (for the complete graph A), and it is especially dependent on properties that sometimes come from a small sub graph embedded in it. And this is also the reason that explains why two networks with essentially the same degree distribution can have very different synchronization characteristics, as the degree distribution is a global property for the complete graph and does not consider the value of λ_2 [58] (see figure 6.2-1)

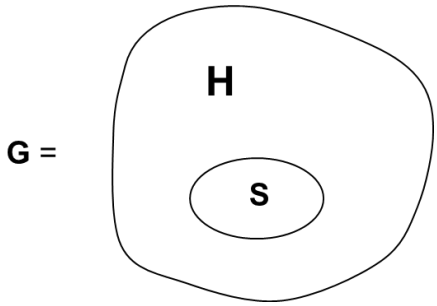


Figure 6.2-1 The statistical properties of the graph G is determined by the huge part H, while the value of λ_2 is independently constrained by the small subgraph S [58].

Furthermore, it can be shown [85] that for any given degree distribution (under quite generic assumptions), there is a connected graph having that degree distribution and which has a second smallest eigenvalue of the graph Laplacian arbitrarily close to zero, implying that we can build a very bad synchronizer for any degree distribution, meaning that the degree distribution does not control synchronization

There are three methods to establish synchronization based on the eigenvalues[84][82]: (1) The Wu–Chua conjecture, (2) Master stability function based on the stability of the synchronous state and (3) Master stability function based on the instability of the asynchronous state

6.2.1 The Wu–Chua conjecture

The Wu-Chua conjecture establish [84] that if we have a certain network of n1 nodes with a certain connectivity which synchronizes at a certain coupling strength ε_{n1}, then another network with n2 nodes and similar connectivity will synchronize at a certain coupling strength ε_{n2} which satisfy the following relationship:

$$\varepsilon_{n1} \times \lambda_2(n1) = \varepsilon_{n2} \times \lambda_2(n2) \quad (149)$$

Where $\lambda_2(n)$ is the second smallest eigenvalue of the Laplacian matrix associated with the network graph. If we apply equation (149) to the specific case of a network of two nodes, where $\lambda_2(2) = 2;$ we obtain the relation:

$$\varepsilon_n = \frac{2\varepsilon_2}{\lambda_2(n)} \quad (150)$$

Which says that a network of n nodes will synchronize with a coupling strength directly proportional to the coupling strength of a synchronizing network of 2 nodes and inversely proportional to the second smallest eigenvalue of the n nodes network.

This conjecture does not work in all cases but for networks of limit cycle and chaotic oscillators (e.g. with Hodgkin–Huxley nodes) it does work and provide a good guide to global synchronization.

6.2.2 Master stability function based on the stability of the synchronous state

Synchronization stability analysis of the network can be conducted using the master stability function formalism [58] [16] [17]. In this formalism we try to determine how stable will be a synchronous state in a networked dynamical system composed of many nodes, connected with the adjacency matrix $\mathrm{A} = \{a_{ij}\}$, with possible weights for each connection $\mathrm{W} = \{w_{ij}\}$, a generic coupling strength parameter for the whole network σ and a vector function H that maps in a generic way input and output state vectors. The dynamic of the system can be represented as:

$$\dot{\mathbf{x}}_i = \mathbf{F}(\mathbf{x}_i) + \sigma \sum_{i=1}^{N} a_{ij} w_{ij} [\mathbf{H}(\mathbf{x}_j) - \mathbf{H}(\mathbf{x}_i)] \quad (151)$$

And the right hand side can be finally transformed to:

$$= \mathbf{F}(\mathbf{x}_i) - \sigma \sum_{j=1}^{N} G_{ij} \mathbf{H}(\mathbf{x}_j) \quad (152)$$

Where G is the Laplacian matrix of the weighted adjacency matrix.
H indicates which state vectors we will use to perform the connection between nodes (independently of the graph connectivity which is controlled by A and W).
Finally, $\mathbf{x}_i$ is the state vector of node i.

The synchronized state of the network is characterized by:

$$\mathbf{x}_1(t) = \mathbf{x}_2(t) = \ldots = \mathbf{x}_N(t) = \mathbf{s}(t) \quad (153)$$

The stability and achievability of this synchronous state is determined, for a specific value of σ, by the eigenvalues of the Laplacian matrix G (and mainly by λ_2 and λ_N), and a sufficient condition for synchronization is given by [58]:

$$\max\{|1 - \lambda_i| : i = 2, \ldots, n\} < e^{-\mu} \quad (154)$$

Where μ is the Lyapunov exponent of the local (isolated dynamic of one node) while the left hand side depends exclusively on the network connectivity. In this way, it is possible to separate the connectivity from the dynamics of the nodes and to establish conditions to achieve synchronization. Further elaborating on these principles [58][89], it can be concluded that the largest and second smallest eigenvalues influence synchronization in the following way:

- For undirected networks: synchronizability depends on λ_2, higher values of λ_2 implies better synchronizability.
  The ratio of the largest to the second smallest eigenvalue is also used as a synchronization index and is called spread of eigenvalues:
  $R = \lambda_N / \lambda_2$

R is determinant for synchronizability. The smaller R the better synchronization is achieved. Therefore, a network whose smallest and largest degrees are very different is a bad synchronizer.

- For directed network, synchronizability depends on the normalized spread of eigenvalues, defined by [79]:

$$\sigma^2 := \frac{1}{d^2(n-1)} \sum_{i=2}^{n} |\lambda_i - \bar{\lambda}|^2$$

With

$$\bar{\lambda} := \frac{1}{n-1} \sum_{i=2}^{n} \lambda_i$$

$$d := \frac{1}{n} \sum_i \sum_{j \neq i} A_{ij}$$

Where A is the adjacency matrix. The smaller the normalized spread of eigenvalues, the more synchronizable the network will generally be.

Considering that for a directed network the eigenvalues of the connectivity matrix are complex values (not symmetric), another method to establish the best propensity of a directed network for synchronization is obtained [89] when the ratio of the largest to the smallest non-zero real parts of the eigenvalues, $\lambda^r_{\max} / \lambda^r_2$, and the quantity $\mathrm{M} = \max_j \{| \lambda^i_j |\}$ (the maximum absolute value of the imaginary parts) are simultaneously made as small as possible, where λ^r and λ^i are respectively the real and imaginary parts of the eigenvalues.

Applying these concepts to the previously defined networks we obtain the following conditions for synchronization:

- Global coupled network: A network with N nodes has a single eigenvalue at 0 and all the others equal to N, therefore λ_2 increases with N. Implying that a globally coupled network can synchronize as far as N is large enough [17].
- Regular ring with N nodes each coupled to its 2k nearest neighbors: In this network: $R \approx N^2 / k^2$, implying that synchronization is worst as N increases and better as k increases [18].
- Star coupled network: The eigenvalues are: 0, N and 1. 0 and N with multiplicity 1 and 1 with multiplicity N-2. In this network synchronization will depend mainly on the coupling strength as λ_2 does not depend on N [17].
- Small world network: Starting with a NW small world network, the second smallest eigenvalues follows these rules [17]:
    - For any given value of N: $\lambda_2(\mathrm{p},\mathrm{N})$ increases to N as p increases from 0 to 1.
    - For any given value of p : $\lambda_2(\mathrm{p},\mathrm{N})$ increases to 1 as N increases to 1.

  The above results imply that, for any given coupling strength $\sigma > 0$:
    - For a high enough N, there exists a critical value, $\bar{\mathrm{p}}$, so that if $\bar{\mathrm{p}} \leq \mathrm{p} \leq 1$, then the small world dynamical network will synchronize.

- For any given p there exists a critical value $\overline{N}$, so that if $N > \overline{N}$, then the small-world dynamical network will synchronize.

That implies that a nearest-neighbor coupled network consisting of N nodes, with a high N, that could not synchronize, will synchronize by adding some long-range connections, then improving synchronization.

- Scale free network: The second smallest eigenvalues [17] decreases to a negative constant as N increases to infinity. This implies that adding new nodes in a scale free network cannot decrease the synchronizability of the network, and the synchronizability of a large size scale-free dynamical network will remain almost unchanged by the constant adding of new nodes. From the point of view of synchronizability, the behavior of a scale free network is similar to the behavior of a star coupled network.

For weighted networks there is an important result [16] that connects the value of the synchronization index R with the ratio S_{max}/S_{min} where S is the intensity, which is defined for a node i as:

$$S_i = \sum_{j=1}^{N} a_{ij} w_{ij}$$

That is, the intensity of a node is the total input weight of the node. The main finding is that the synchronizability is sensitively controlled by the heterogeneity of the intensity S_i. The smallest the heterogeneity implies the highest synchronizability.

This is in accordance with the influence of degree fluctuations in synchronization and the paradox of heterogeneity [89]. The fact that networks with heavy-tailed degree distributions and heterogeneous connectivity patterns are more difficult to synchronize than homogeneous networks has been dubbed the paradox of heterogeneity. Using this knowledge, a possibility to increase synchronization in a free-scale network (or a network with hubs) is to reduce the weight of the hub connections in a manner inversely proportional to the degree of the hub node.

6.2.3 Master stability function based on the instability of the asynchronous state

An alternative analysis to the one followed by the master stability formalism, where we try to find the parameters that maintain the stability of a synchronization state in a network, is to find the parameters that influence the instability of the complete asynchronous (incoherent) state in a network. Following these ideas, it has been shown [80] that the largest eigenvalue λ_N of the adjacency matrix influence the stability of the asynchronous state. In particular, the larger λ_N, the more likely the incoherent state is unstable [80][81], implying it is also the more likely a synchronous state can be reachable.

6.2.4 Random matrix methods

It is interesting to see the application of random matrix theory to obtain results on the eigenvalues distribution of big random matrices (associated to big neural connectivity matrices). The distribution of these eigenvalues under several considerations can provide light on which elements have an influence in network synchronization (considering the impact of the eigenvalues distribution on synchronization of the network).
In [54] is done an analysis of the distribution of eigenvalues in big connectivity matrices with excitatory and inhibitory connections (positive and negative coupling strength), where the excitatory coupling strengths are drawn from a probability distribution different to the probability distribution of the inhibitory connections. The result of the study is that the average value of the excitatory and inhibitory coupling strengths has no major effect on stability or spontaneous activity of the network, instead it is the width of the strength distribution which have an strong impact, being more important if the distribution of both excitatory and inhibitory coupling strength are different.

6.3 Graph properties methods

The different networks presented in section 6.1, and, any network in general, has a structure defined by its graph. This structure has an impact on the synchronization capacity of the network. The structure can be defined using graph properties: average path length, degree distribution, connection weights distribution, etc… or using the distribution of values associated to the eigenvalues/eigenvectors of the associated adjacency matrix defined by the network (see section 6.2).

This section is related to the study of the graph properties of the network (no spectral graph methods).

In [68] are presented a series of general results on synchronization using graph properties which are worth to mention:

- For sufficiently strong connected networks, the network synchronizes when the underlying undirected graph is connected [68].
- For directed graphs, the network synchronizes, for sufficiently strong connected networks, when the underlying graph is strongly connected [68].
- For connected directed graphs which are not strongly connected, the network synchronizes, for sufficiently strong connected networks, if the underlying graph contains a spanning directed tree [68]. This is an intuitive result since this graph-theoretical condition implies the existence of a node (located at the root of the tree) which directly or indirectly influences all other systems.

6.3.1 Network motifs

There is an interesting influence of network structure in the correlation of node activities (synchronization) considering specific network motif which are present in the network graph [82][83].

Network motifs are specific connectivity patterns present in the network, e.g. figure 6.3-1 shows 2 network motifs: divergence and chain motifs.

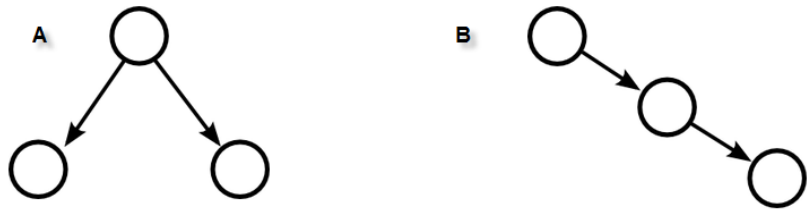


Figure 6.3-1 Network motifs: (A) divergence motif and (B) chain motif [82]

In the same network, the motifs considered can be simple or complex (see figure 6.3-2).

These motifs have an impact in the correlation of signals in the network [83][82].
To determine the relationship between motif structure and correlation of signals, we can start from the uncoupled network, in which the value of the signals from the nodes can be put together in a vector of length equal to the number of nodes: X_0.
Then, we can reestablish coupling between nodes, resulting in nodes activity at any time t given by: $X(t+1) = X_0 + \mathrm{w}\mathrm{A}X(t)$ (155)

Where A is the adjacency matrix and w is a generic coupling strength.
The generic expression of activity can then be obtained for stationary conditions:

$$X(t) = (I - wA)^{-1} X_0 \quad (156)$$

The correlation of activity in the network can be stated as:

$$\begin{aligned} E(XX^T) &= (I - wA)^{-1} E(X_0 X^T{}_0)(I - wA^T)^{-1} \\ &= (I - wA)^{-1} C_0 (I - wA^T)^{-1} \\ &= (I - wA)^{-1} c_0 I (I - wA^T)^{-1} \\ &= c_0 (I - wA)^{-1} (I - wA^T)^{-1} \\ &= c_0 \sum_{i,j=0}^{\infty} \mathrm{w}^{\mathrm{i+j}} \mathrm{A}^{\mathrm{i}} (\mathrm{A}^{\mathrm{T}})^{\mathrm{i}} \end{aligned} \quad (157)$$

Where the last expression is obtained using a power expansion assuming that wA is small (weak coupling). C_0 is a diagonal matrix as it is the covariance matrix of the uncoupled nodes (independent), we further assume that the diagonal entries are equal (statistically homogeneous network).

The result coming from equation (157) is that (considering a w small), the main influence in signal correlation in the network comes from the small powers of A and A^T ($\mathrm{A}^{\mathrm{T}}\mathrm{A}$, A^2, $(\mathrm{A}^{\mathrm{T}})^2$), which correspond to the simplest motifs in the network.

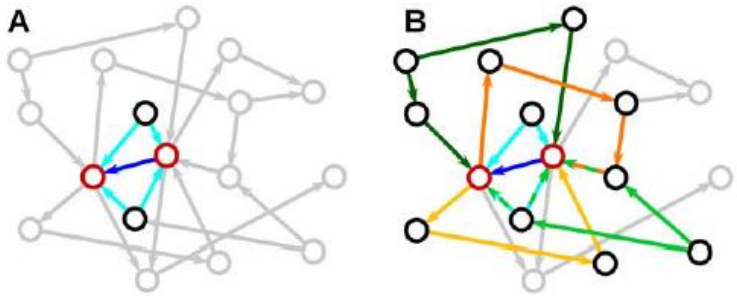


Figure 6.3-2 Increasing complexity of network motifs. It can be only considered only the simplest motifs (A) or more complex ones (B) [83]

In [83] is presented an interesting connection between these results and the influence of the largest eigenvalue of the adjacency matrix.
After diagonalizing A:

$$\left\| A^i (A^T)^j \right\| = \left\| U^{-1} \Lambda^{j+i} U \right\| \le \left\| U^{-1} \right\| \left\| U \right\| \left\| \lambda_{\max} \right\|^{i+j} \quad (158)$$

Then a large value of $\lambda_{\max}$ will imply that more higher-order terms (higher powers of A) contribute to correlation. That is, a high value of $\lambda_{\max}$ will mean that correlation depends more on complex connection structures in the network (higher powers of A).

6.3.2 CGS

Another important method to establish synchronization (not based in the graph spectrum) is based on the connection graph stability method (CGS)[84].
This method is better than spectral methods for time varying coupling networks where the Laplacian of the connectivity matrix is time varying and the techniques presented in section 6.2 are not applicable in general.

The CGS method establishes that in order to achieve synchronization, the coupling strength for each edge of the network has to fulfill the following theorem due to Belykh [84]:

$$\varepsilon_k(t) > \frac{2\varepsilon_2}{n} b_k(n,m) \quad (159)$$

Where $\varepsilon_k(t)$ is the needed coupling strength in edge k and at each time t. The number n is the number of nodes, m is the number of edges, ε_2 *is th*e coupling strength sufficient for global synchronization of two nodes, and $b_k(n,m)$ is the sum of the lengths of all chosen paths which connect any pair of nodes and which pass through the edge k.

This approach allows having a sufficient condition to obtain synchronization based exclusively in graph theory methods (length of all paths going through a specific edge) and the necessary coupling strength to ensure synchronization for a network of only two nodes.

In equation (159) the value $b_k(n,m)/n$ can be replaced with $1/\lambda_2(n)$ which produces equation(150). That is, $b_k(n,m)$ can be used as an estimate of the second smallest eigenvalue of the connectivity matrix.

Equation (159) provides a suboptimal estimate to achieve synchronization but it is affordable to obtain, in terms of computing time, for structural changing networks.

6.4 Speed of synchronization

The synchronization capacity of a network seems to be determined more by the spectral characteristic of the network [58] but the speed to achieve this synchronization seems to be determined by:

- The second smallest eigenvalue of the Laplacian (λ_2) has an important role in the time needed for complete synchronization, in particular the synchronization time scales with the inverse of the smallest nonzero eigenvalue of the Laplacian matrix [16]. This relation qualitatively holds for very different networks where synchronization is achieved, indicating that this eigenvalue alone might be a relevant topological property for synchronization phenomena.
- The typical in-degree of the digraph associated to the network [56]. Even with arbitrarily strong interaction strengths, neurons cannot synchronize faster than at a certain maximal speed determined by the typical in-degree [56].

In [56] [63] is studied the collective synchronization of pulse-coupled oscillators interacting on asymmetric random networks. Random matrix theory is applied to predict the speed of synchronization of this network, and to find that there is a limit in the speed of synchronization, being established by the network connectivity and that remains finite, even if the coupling strength becomes infinite.

7 Toolbox

In the following section it is presented a Matlab toolbox (SyncLab) made to study the synchronization phenomena of N neurons connected under different graph structures.

The toolbox allows modifying several parameters of neuron dynamics and network connectivity and monitoring different results of network activity

The toolbox implements some of the synchronization detection techniques presented in this thesis.

7.1 SyncLab

SyncLab is a Matlab toolbox implemented specifically to study synchronization phenomena of a network of N neurons connected in different ways.

The program simulates the N neurons connected in a specific connection graph, with specific inhibitory and excitatory dynamics. The program allows observing the spiking activity of all N neurons in real time with several diagrams. It also shows the voltage activity of two chosen neurons, plus some average activity.

It has been also implemented several methods to measure correlation activity among the neurons (both in time and space).

In figure 7.1-1 is presented the main window of SyncLab application.

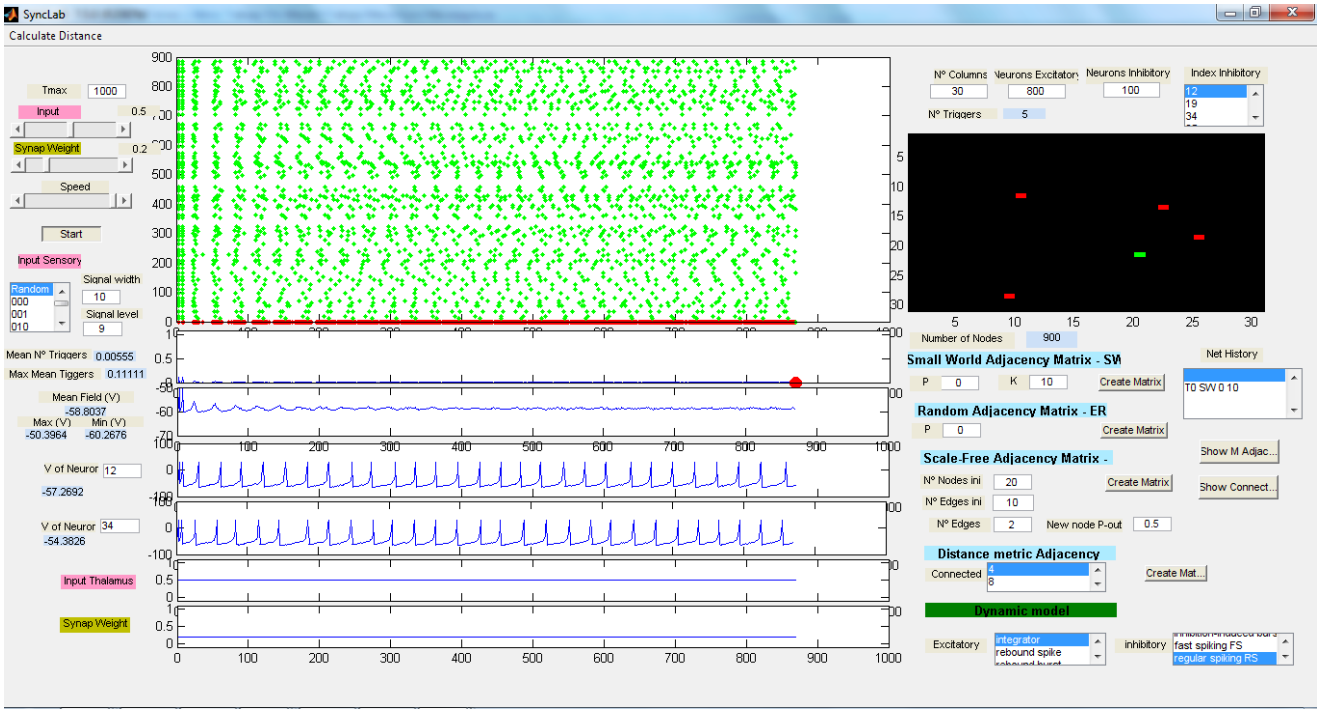


Figure 7.1-1 SyncLab application

SyncLab allows modifying the following input parameters:

- Number of neurons
- Number of excitatory and inhibitory neurons
- Strength of input noise (input thalamus) for all neurons
- General synapse coupling strength (for all the edges in the network)
- Maximum time of simulation execution.
- Type of connectivity network:
    - Small World
    - Random
    - Scale-Free
    - Regular 2D lattice
- Type of neuron dynamic for each inhibitory and excitatory neuron.

Once, the input parameters are set out, the following information is obtained:

- A raster diagram: of real time spiking activity of the N neurons (main diagram in green in figure 7.1-1). Spikes of inhibitory neurons are in green and excitatory in red.
- A 2D spiking diagram: which shows spikes, similar to previous diagram, but with the neurons arranged in a 2D disposition (right diagram in figure 7.1-1).
- A plot of total number of spikes vs. time
- A plot of average voltage (considering all neurons) vs. time.
- Two plots for the voltage activity of two selected neurons vs. time
- Plot of strength of input noise (input thalamus) vs. time
- Plot of general synapse coupling strength (for all the edges in the network) vs. time
- List of index number of all inhibitory neurons

The application allows modifying all the input parameters in real time, as the simulation proceeds, obtaining the corresponding real time changes in the output diagrams.

In order to track the changes made, there are available several resources:

- Plot of input noise strength vs. time
- Plot of general coupling strength vs. time
- Listbox with network history. In this listbox is provided the time of any change done to the network structure plus the parameters that defines the new structure.

7.1.1 Detailed description

In this section is presented in detail the toolbox features and dynamical model used in the simulation.

7.1.1.1 Simulation time

This parameter is set with ¨Tmax¨ input textbox. It has to be set at the beginning of each simulation.

7.1.1.2 Input Thalamus

In the input Thalamus section can be set out the input to the neurons.

The ¨input thalamus¨ slider allows setting the level of signal provided by the Thalamus.

The level is set differently for excitatory and inhibitory neurons.

The minimum level is 0 and the maximum is 1, so the slider allows choosing a value between [0-1]. To the value obtained in this way, a multiplicative factor is applied to adjust the inputs. The multiplicative factors for excitatory and inhibitory neurons are the values given in the textboxes “Exc factor” and “Inh factor” in the Input Thalamus section.

The listbox in this section controls the input provided to the neurons and can have one of 3 values:

- All random: The neurons will receive a random input. In this case, the input to all neurons will be:

  ```
  I=slidervalue*randn(num_neurons,1);
  I([loc_exc],1)=I([loc_exc],1)*factor_exc;
  I([loc_inhib],1)=I([loc_inhib],1)*factor_inh;
  ```

  Where:

  ```
  num_neurons: is the total number of neurons. slidervalue: is the
  value [0-1] selected by the Input Thalamus Slider. loc_exc: are the
  indexes of all excitatory neurons. loc_inhib: are the indexes of
  all inhibitory neurons. factor_exc: is the factor in textbox “Exc
  factor”. factor_inh: is the factor in textbox “Inh factor”.
  ```

- Select neurons: The neurons between the indexes given by the textboxes “Ini” and “End” will receive an input given by:

  ```
  I=zeros(num_neurons,1);
  I(neuron_ini:neuron_end,1)=1
  I([loc_exc],1)=I([loc_exc],1)*factor_exc;
  I([loc_inhib],1)=I([loc_inhib],1)*factor_inh;
  ```

  Where:

  ```
  num_neurons: is the total number of neurons. loc_exc: are the
  indexes of all excitatory neurons. loc_inhib: are the indexes of
  all inhibitory neurons. factor_exc: is the factor in textbox “Exc
  factor”. factor_inh: is the factor in textbox “Inh factor”
  ```

- All zeros: The input for all neurons will be zero.

7.1.1.3 Synaptic weights

The Synap Weight section allows controlling de synaptic weights applied to all neuron connections.

The ¨Synap Weight¨ slider allows setting the general multiplicative factor applied to the synaptic weights of all synapses.

The minimum level is 0 and the maximum is 1, so the slider allows choosing a value between [0-1].

Additionally the excitatory synapses have another multiplicative factor that is set out in the “cfactor” textbox and, similarly for the inhibitory in the “inhf” textbox.

Finally all synapses have a final factor of $\mathrm{cfactor} = 100/\left(\mathrm{Ne}*\mathrm{excf} + \mathrm{Ni}*\mathrm{inhf}\right)$ which is a correction factor to generate an input signal of as much as 100 at any neuron. This factor is applied automatically by the program.

The inhibitory or excitatory behavior is associated to the neuron (not to the synapse). That means that all synapses coming out of an excitatory neuron will be excitatory (similarly for inhibitory).

7.1.1.4 Simulation launch

With the slider ¨Speed¨ it is controlled the simulation speed. It is usually set to maximum.

The button ¨Start¨ launches the simulation.

After the simulation is finished, several matrices are created in the working directory (.mat) and additionally their values are stored in the Matlab workspace. The matrices are:

- VV: Contains all the simulation values for all neurons (rows) and time (columns)
- Spike: Similar to VV but is a matrix of 0-1 with a 1 when a spike occurs.
- MExterna: Adjacency matrix of the network being used (final one in case of changes).
- EigenA: All the eigenvalues of Adjacency matrix.
- EigenL: All the eigenvalues of Laplacian matrix.

7.1.1.5 Activity diagrams

The following plots are available, starting from left-top to left-bottom:

- A raster diagram: of real time spiking activity of the N neurons. Spikes of inhibitory neurons are in green and excitatory in red. Time runs on the horizontal line and neurons (from 1 to N) in vertical. The number given to each neuron is sequential and does not provide further information. This number is the same used for the network construction and representation.
- A plot of average number of neurons firing vs. time
- A plot of average voltage (considering all neurons) vs. time.
- Two plots for the voltage activity of two chosen neurons vs. time. The neurons to show are chosen by their index number, using a textbox.
- Plot of strength of input noise (input thalamus) vs. time. This plot is used to track changes made to this value using the slider as the simulation was running.
- Plot of general synapse coupling strength (for all the edges in the network) vs. time. This plot is used to track changes made to this value using the slider as the simulation was running.

At the left of these plots are presented some instantaneous values of voltage, mean field, average number of spikes and maximum and minimum values of the corresponding running plots.

On the right:

- A 2D spiking diagram: which shows spikes similarly to the spiking raster diagram but with the neurons arranged in a 2D disposition. Spikes of inhibitory neurons are in green and excitatory in red. The number of columns to create the diagram is set to the value present at the ¨Nº Columns¨ textbox at the right upper part of the window.
  Neurons are arranged on this diagram in a sequential way, filling each row at a time. If the number of neurons is not a multiple of the number of columns, there will be some black cells at the last row.

  At the top of this plot is shown the instantaneous number of spikes (not the average)

7.1.1.6 Number of neurons

At the top upper part of the window is set the number of excitatory and inhibitory neurons. The total number of neurons is calculated as the sum of these values.

There are specific textboxes to set these values. At the right of these textboxes there is a listbox which is automatically filled up as soon as the simulation starts. This listbox contains the indices of the inhibitory neurons which are given randomly.

The number of neurons cannot be changed after the simulation starts.

7.1.1.7 Network structure

At the right middle part of the window is established the network type to be used in the simulation:

- Small World: Using Watts-Strogatz method (section 6.1), where the inputs are the probability of rewiring and the initial number of connections to the k adjacent nodes. The initial k connections are unidirectional from node I to nodes i+1,….,i+k.
- Random: Using Erdos-Renyi method (section 6.1), where every node (N in total) is connected to every other node with a probability p.
- Scale-Free: Using Barabasi-Albert method (section 6.1), where starting with an initial number of nodes and edges (connected randomly), a node is added one by one, connecting each new node with m nodes (m new edges) in the network.
  In order to select the nodes to be connected, it is used the degree distribution of the nodes. The higher the degree, the higher the probability of receiving a new connection.
  As the connection network is directed, the model allows determining the probability of each new edge to be an input or output edge to the new node.
- Regular 2D lattice: In this mode, the N neurons are arranged in m columns forming a 2D lattice. From this arrangement it can be selected two modes of connection: 4-edges or 8-edges connection. In the 4-edges each neuron is connected to the

upper, lower, left and right neurons, in the 8-edges connection, each neuron is connected as in the 4-edges plus to each of the 4 diagonal neurons.

The network type can be changed **before** and **during** simulation running. In order to change the network, it is necessary to set the corresponding network parameters (depending of the network type) and to click the button ¨Create Matrix¨ at the right of the associated type.

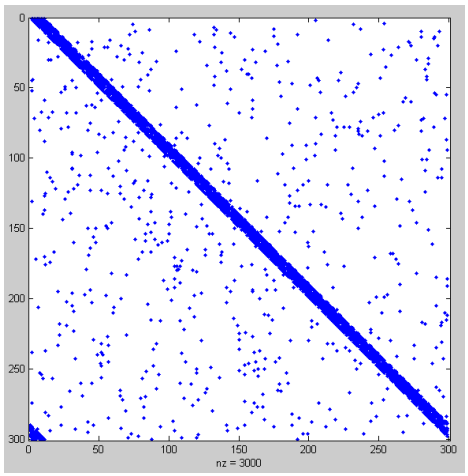
At the right middle part of the window, and right to the network modeling section, there is a listbox with the network history. In this listbox is provided the time of any change done to the network structure plus the parameters that defined the new structure.
To clarify it, as an example, it follows the meaning of a possible network history:

T0 SW 0 10
T115 SW 0.9 50
T176 ER 0.7
T288 BA 20 10 6 0.8
T335 SP 4
T386 SP 8

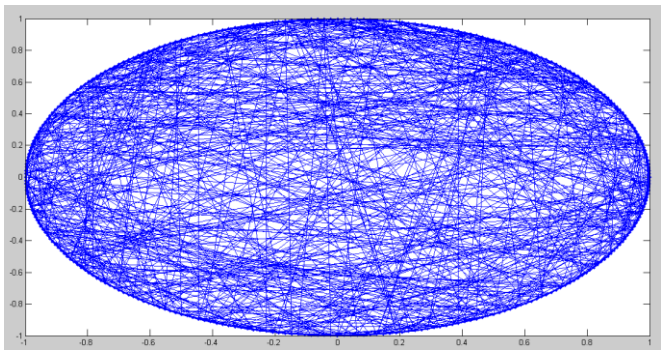
- It will inform that at time 0 the network type was Watts-Strogatz with rewiring probability 0 and 10 unidirectional connected nodes.
- At time 115 the network type was Watts-Strogatz with rewiring probability 0.9 and 50 unidirectional connected nodes.
- At time 176 the network type was Erdos-Renyi with probability 0.7.
- At time 288 the network type was Barabasi-Albert starting with an initial number of 20 nodes and 10 edges (connected randomly), a node is added one by one, connecting each new node with 6 existing nodes and with a probability of each new edge to be an output edge (to the new node) of 0.8.
- At time 335 the network type was a regular 2D lattice with each node connected to the 4 nearest nodes.
- At time 386 the network type was a regular 2D lattice with each node connected to the 8 nearest nodes.

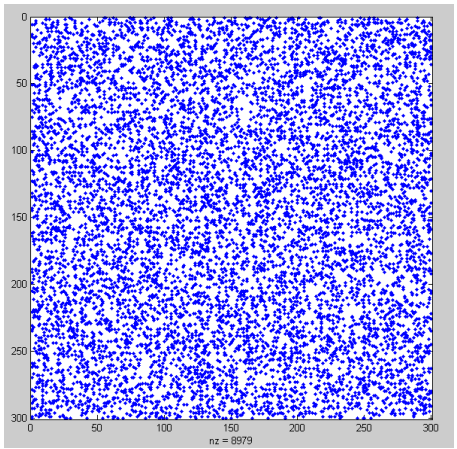
Below the ¨network history¨ listbox there are two buttons which generate a graph for the adjacency matrix and connectivity matrix of the currently active network structure.

The following diagrams show different adjacency and connectivity matrices generated by the Toolbox:

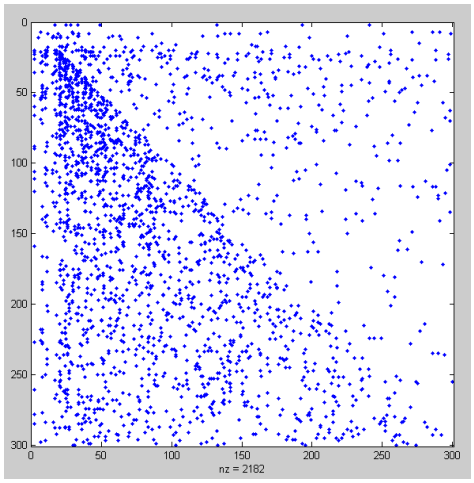


Adjacency matrix: 300 neurons, Watts-Strogatz with P= 0.2 and K=10

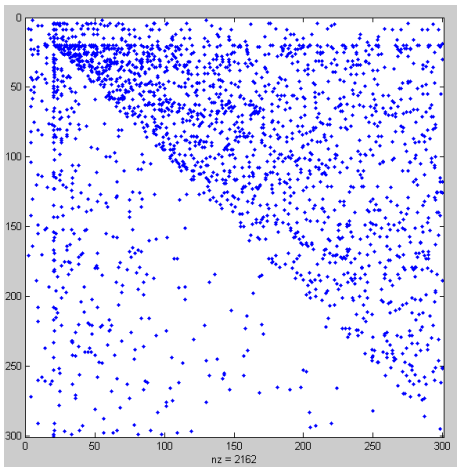

Connectivity matrix: 300 neurons, Watts-Strogatz with P= 0.2 and K=10



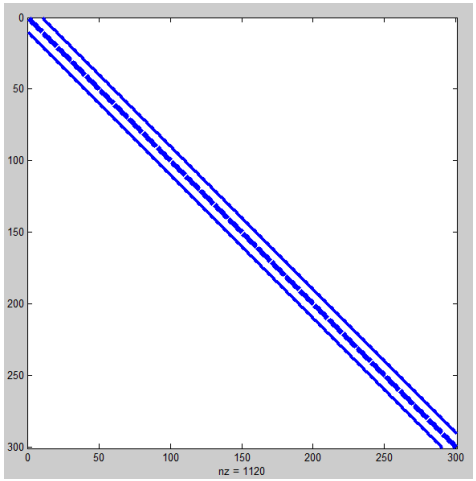
Adjacency matrix: 300 neurons, Erdos-Renyi with probability 0.7

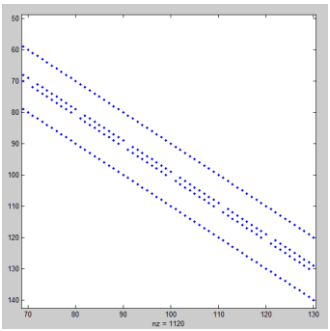


Adjacency matrix: 300 neurons, Barabasi-Albert, ini-nodes 20, ini-edges 10, added edges 8, probability output edge 0.8

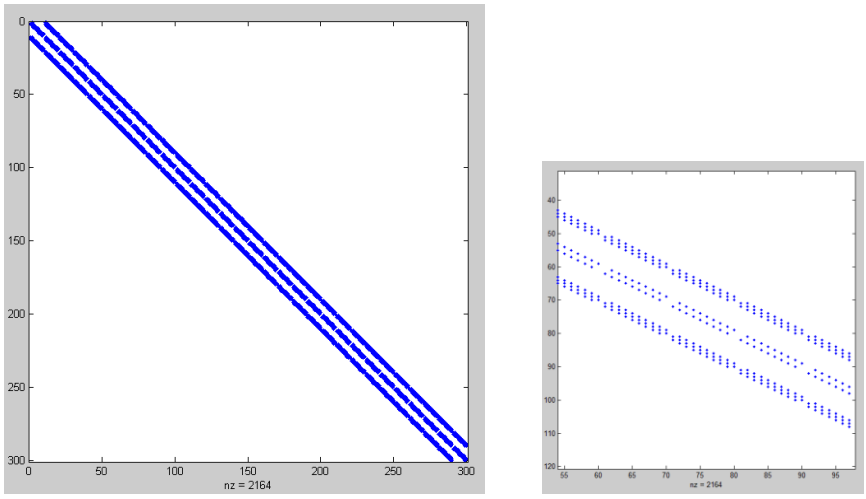


Adjacency matrix: 300 neurons, Barabasi-Albert, ini-nodes 20, ini-edges 10, added edges 8, probability output edge 0.2

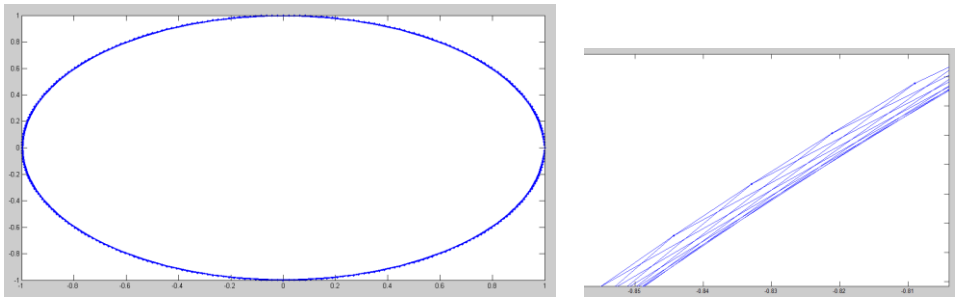




Adjacency matrix: 300 neurons, regular 2D lattice 4-edges, with a detailed view to the right.



Adjacency matrix: 300 neurons, regular 2D lattice 8-edges, with a detailed view to the right.


Connectivity matrix: 300 neurons, a regular 2D lattice 8-edges, with a detailed view to the right.

7.1.1.8 Neurons dynamic model

At the right bottom part of the window is set the type of neuron dynamic for each inhibitory and excitatory neuron group.

The dynamic can be different for excitatory and inhibitory neurons, but equal for all neurons inside each group (excitatory or inhibitory).

The inhibitory or excitatory behavior is associated to the neuron (not to the synapse). That means that all synapses coming out of an excitatory neuron will be excitatory (similarly for inhibitory).

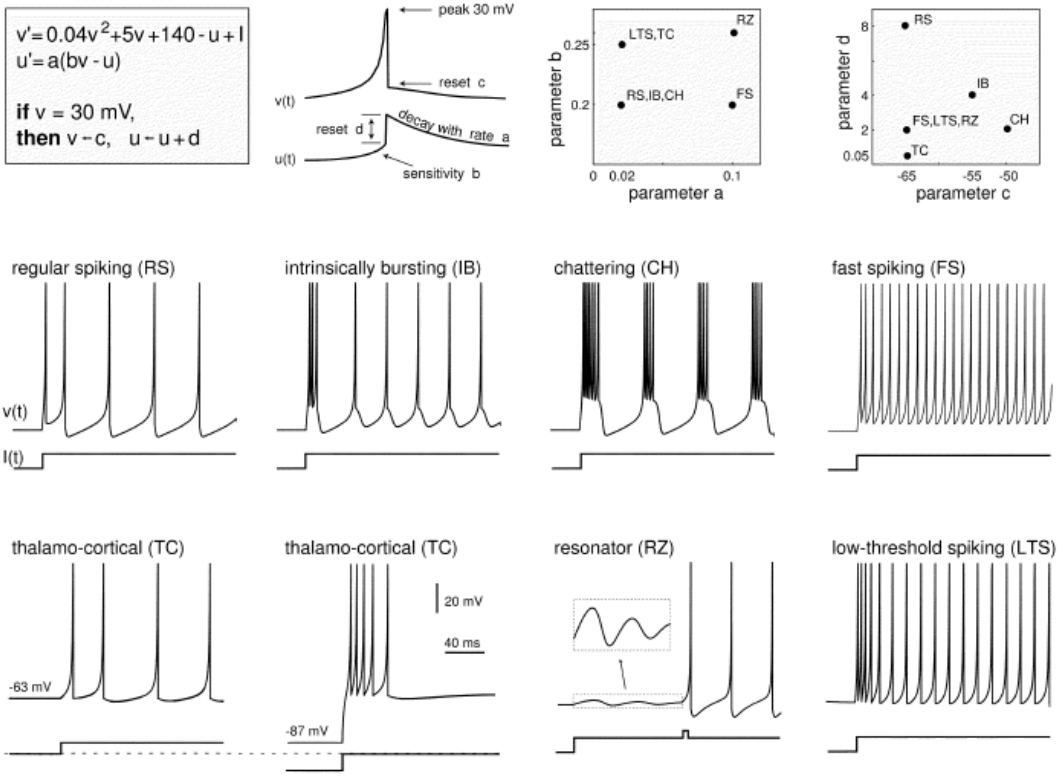
Over the list boxes which allow selecting the dynamic model there is a checkbox: "Random model" which is not checked by default. When this checkbox is checked the dynamic model for excitatory and inhibitory neurons will be randomly selected, following the same model presented in [87].

The neurons dynamic can be changed **before** and **during** simulation running.

The types and neuron dynamic model are chosen from the different types defined in [87][88] (Izhikevich neuron model). This model is a phenomenological spiking model as

computationally efficient as the integrate-and-fire model and that depending only on four parameters can reproduce the spiking and bursting behavior of known types of cortical neurons.

In the following figure [87] are presented (1) the model equations, (2) the four parameters that control the neuronal model and (3) some signal examples of possible signals generated by the model.



7.1.1.9 Output files

The application generates different files with the more important information available:

- Matrix with all neurons activity (rows are neurons and columns is time). It is stored in the file: VV.mat
- Matrix with all neurons spikes (rows are neurons and columns is time). It is stored in the file: Spikes.mat. A value of one indicates the occurrence of a spike and the contrary for a zero.
- Adjacency matrix of the last associated network. It is stored in the file: MExterna.mat

The files are available at the following directory:
[working directory]\files

7.1.1.10 Run the application

In order to run the application it is necessary to open Matlab and at the command window use the following command:

>> Neurosync

7.1.2 Network Eigenvalues application

As a specific menu it is available a range of functionalities around the study of the eigenvalues of the adjacency and Laplacian matrices associated to the network graph.

To run this application it is necessary to have launched the application and it can be activated as the simulation is running.

In figure 7.1.2-1 is presented the window of this application.

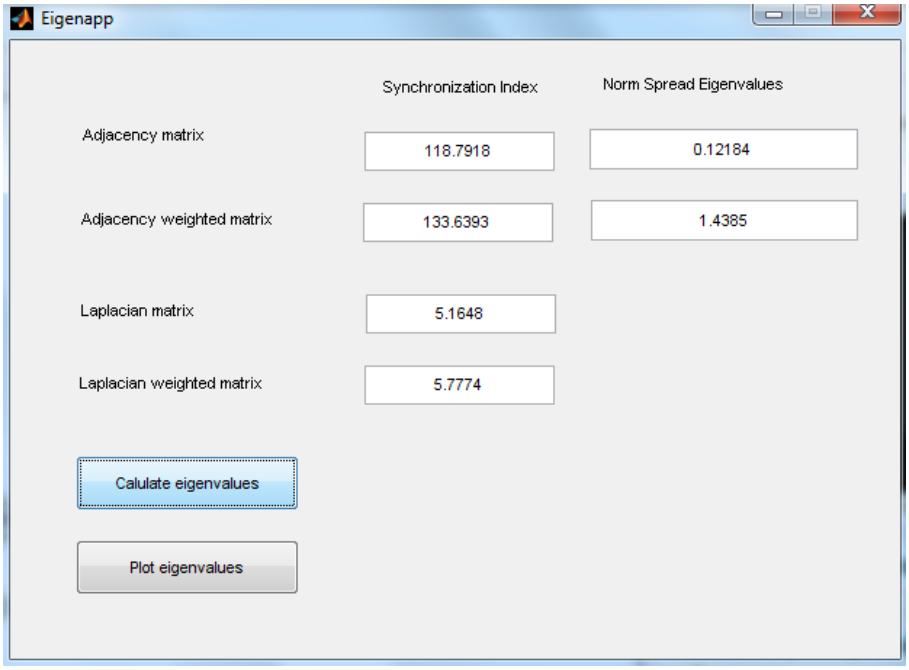


Figure 7.1.2-1 Network eigenvalues application

The application provides:

- Synchronization index associated to different matrices:
  - Adjacency matrix (matrix only with 0-1 values).
  - Adjacency weighted matrix (matrix with real values associated to the weight of each network edge).
  - Laplacian of the Adjacency matrix (matrix only with 0-1 values).
  - Laplacian of the Adjacency weighted matrix (matrix with real values associated to the weight of each network edge).
- Normalized spread of eigenvalues for:
  - Adjacency matrix (matrix only with 0-1 values).
  - Adjacency weighted matrix (matrix with real values associated to the weight of each network edge).

- Plot of the eigenvalues distribution for different matrices:
    - Adjacency matrix (matrix only with 0-1 values).
    - Adjacency weighted matrix (matrix with real values associated to the weight of each network edge).
    - Laplacian of the Adjacency matrix (matrix only with 0-1 values).
    - Laplacian of the Adjacency weighted matrix (matrix with real values associated to the weight of each network edge).

To calculate the values is necessary to press the button “Calculate eigenvalues”, each time it is pressed the values will be calculated with the current available matrices. New matrices will be available each time the network is changed in the main application. In this way, it can be calculated the evolution of the indices as the network structure is changed as part of a simulation
To plot the eigenvalues distribution it is necessary to press the button “Plot eigenvalues”. The following plots will be obtained (figure 7.1.2-2):

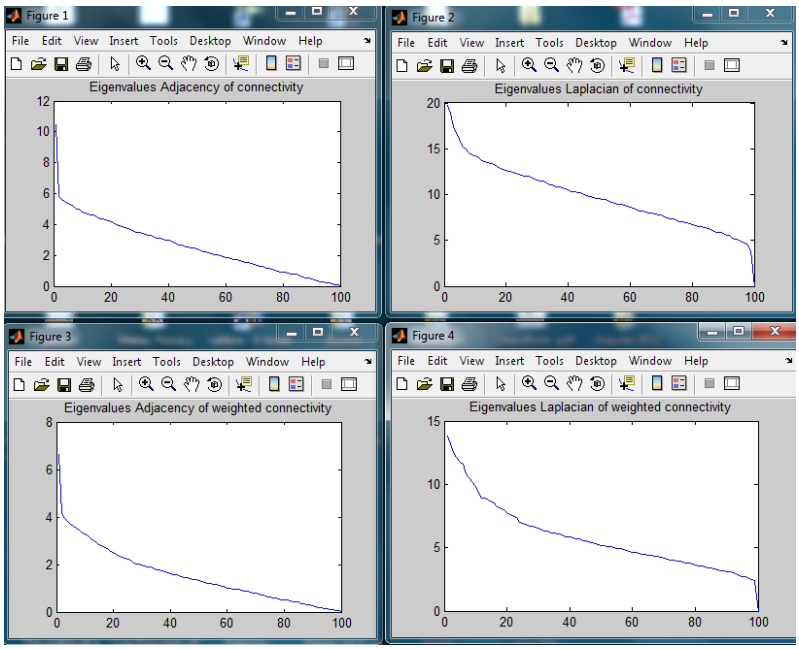


Figure 7.1.2-2 Plots of eigenvalues distributions for different matrices

In some cases the indices cannot be calculated because the denominators of the ratios are too close to zero, in these cases a message will appear at the textboxes, as presented in figure 7.1.2-3

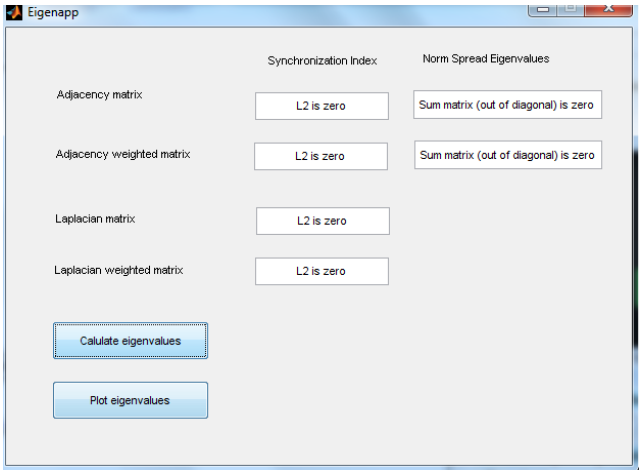


Figure 7.1.2-3 Error messages for the indices.

The synchronization index is useful for undirected networks, while the normalized spread of eigenvalues is useful for directed ones.

The files with the eigenvalues distribution are available at the following directory: [working directory]\files\eigen

7.1.3 Calculate distances application

There are several “distances” that can be calculated using the Calculate distances” menu (see figure 7.1.3-1) .

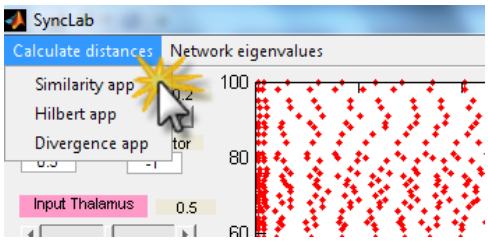


Figure 7.1.3-1 Calculate distance application

The applications available from these menu are presented in the following sections.

7.1.3.1 Similarity application

The “Similarity application” is available as a sub-menu from the “Calculate distances” menu.

In figure 7.1.3.1-1 is presented the main window of the Similarity application as it is shown the first time the window is used.

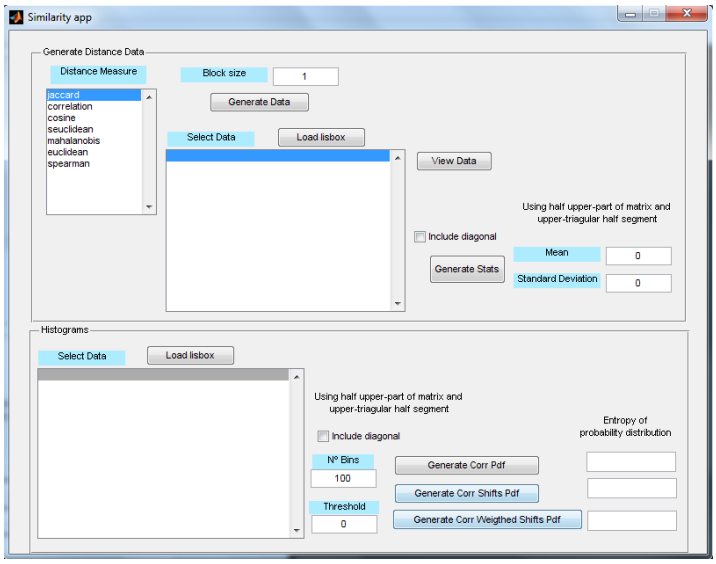


Figure 7.1.3.1-1 Similarity application

This application allows to have a representation of the distances between columns (or rows) of the activity of the ensemble of neurons (presenting the activity as a matrix). The distance between columns i and j is shown as the value in position $D_{i,j}$ of a new distances matrix. Different distances can be applied and the correspondent distances matrices can be visualized.
Associated to the distance matrix, the similarity matrix is the opposite of the distance matrix and it is calculated subtracting one from the distance matrix values

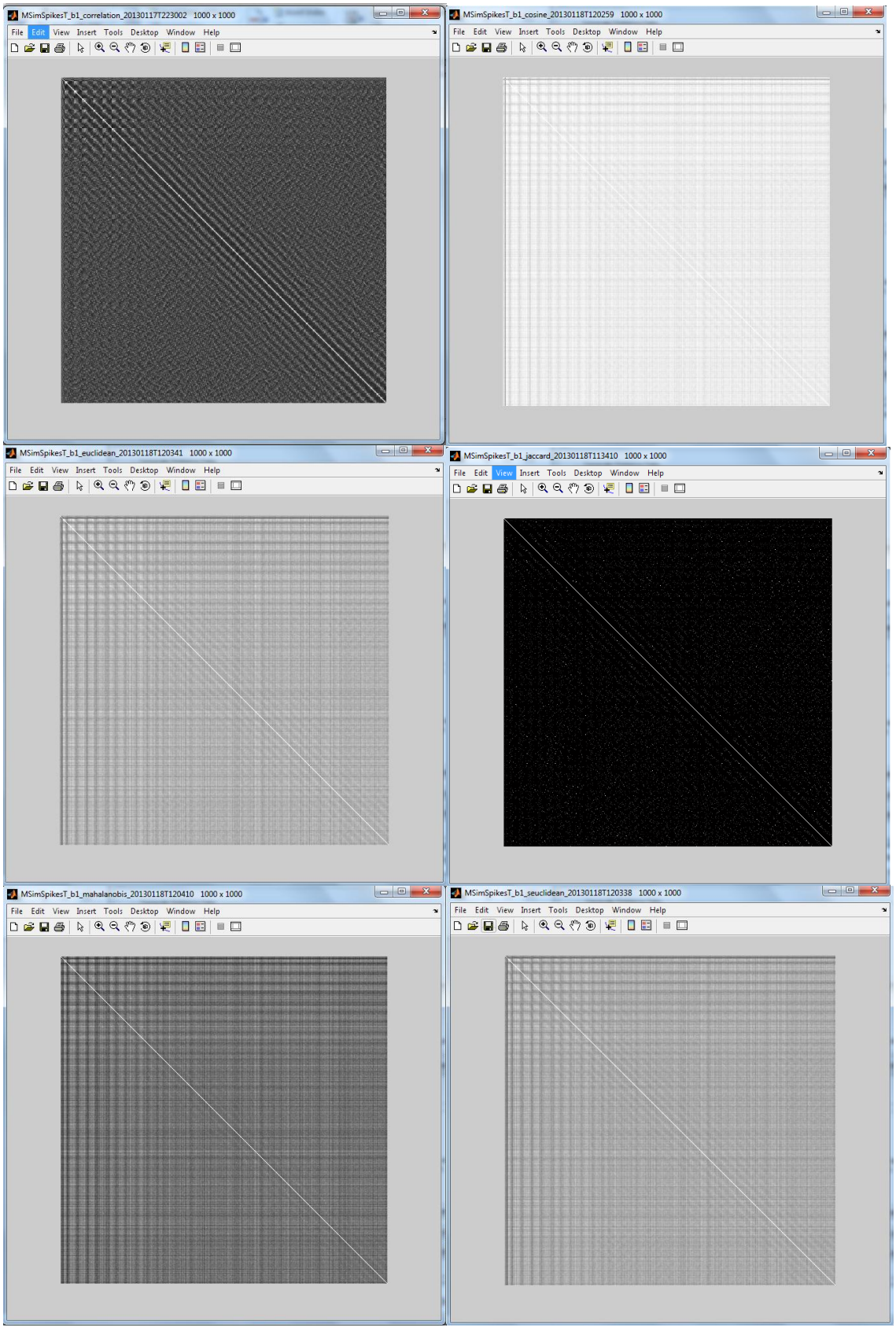
The mean and standard deviation of the values of the similarity matrix (obtained from the distance measure selected) is also presented. These values can be used as a global index: average value of all distance values between columns or rows.

For each type of distance the application calculates the distance and similarity matrices for the activity matrix and its transpose. The transpose is indicated by a T at the end of the first name. If the transposed matrix is selected, all values and plots will be from the transposed matrix. That is, distances between rows will be calculated.

Once the window is opened, we have to select the distance measure and the block size (this is the number of columns or rows that will be used concatenated as a block to compare consecutives blocks in the ensemble, see section 4.2.11.2). Then, we have to press the buttons “generate Data” and “Load Listbox”. The listbox will be populated with the similarity matrix (and its transposed). In the listbox we can select one of the matrix previously calculated.
To calculate the mean and standard deviations of the selected distance matrix we have to press the button: ”Generate Stats”. We can also choose if the values from the diagonal are used in the calculations.

Pressing on button “View Data” we have a plot of the similarity matrix selected on the listbox.

In figure 7.1.3.1-2 is presented an example of similarity matrix for the different distance measures available. In all cases the neurons activity values are the same.


MSimSpikesT_b1_correlation_20130117T223002 1000 x 1000
MSimSpikesT_b1_cosine_20130118T120259 1000 x 1000
MSimSpikesT_b1_euclidean_20130118T120341 1000 x 1000
MSimSpikesT_b1_jaccard_20130118T113410 1000 x 1000
MSimSpikesT_b1_mahalanobis_20130118T120410 1000 x 1000
MSimSpikesT_b1_seuclidean_20130118T120338 1000 x 1000
File Edit View Insert Tools Desktop Window Help

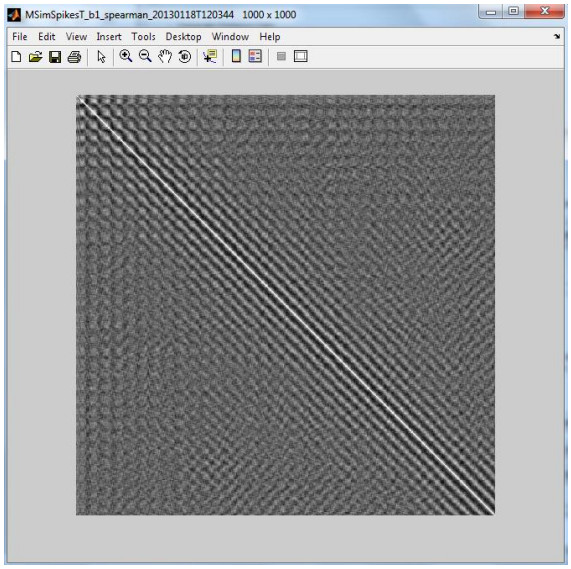


Figure 7.1.3.1-2 Similarity matrices for different distance measures. Color white indicates higher values.

Once the distance and similarity matrices are calculated, the data is available in a file at the following directory:
[working directory]\files\distances
In this directory will be available four files after each simulation: (1) Distance matrix (2) Distance matrix transposed (3) Similarity matrix and (4) Similarity matrix transposed.
The name of the files indicates the distance used, block size, time of file creation and if the distances are from rows or columns, with a "T". The "T" indicates row distances. Additionally, the similarity and distance matrices are identified with the strings: "Sim" and "Dist", respectively.

The second section of the window: "Histogram section", allows obtaining several probability distributions and their associated entropies.

To calculate these probability distributions is necessary to load the second listbox, and to select the similarity matrix to work with. Once selected, there are three probability distributions (pdf) that can be calculated. In all cases, we can choose:

- If the diagonal values of the similarity matrix are considered for the computation
- The number of bins to be used in the histogram and the resulting pdf
- The threshold value. All values in the similarity matrix below this value will be set to zero previously to start the computation.

The first probability distribution is obtained pressing the button: "Generate Distance Pdf" (see figure 7.1.3.1-3). This probability distribution gives the probability of having a certain similarity value in the similarity matrix.

The values used to compute the probability distribution are the values in the upper triangular part of the matrix (symmetric matrix).

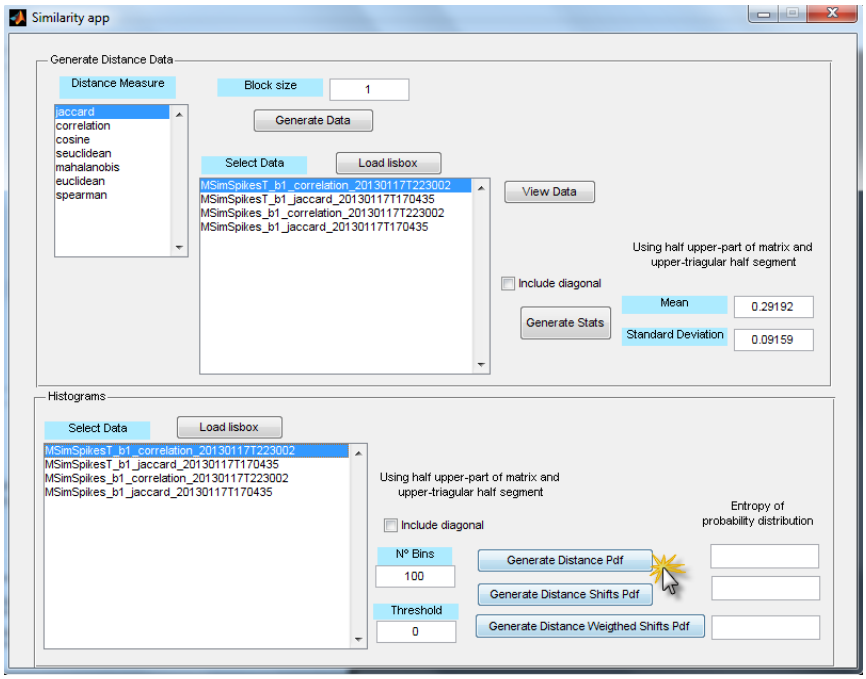


Figure 7.1.3.1-3 First probability distribution (Distance pdf).

In figure 7.1.3.1-4 is presented an example of distance pdf for a correlation distance.

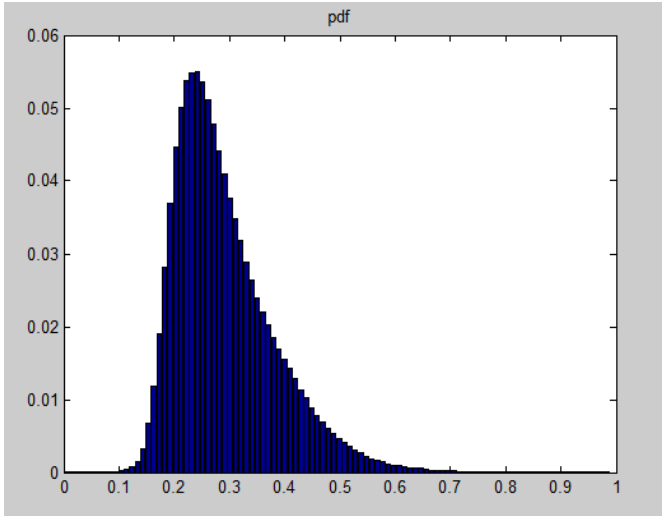


Figure 7.1.3.1-4 Distance pdf.

The second probability distribution is obtained pressing the button: “Generate Distance Shifts Pdf”. This probability distribution gives the probability of having a similarity value (in the similarity matrix) over the threshold but at a specific shift or lag starting from the diagonal. The values of the random variable (the random variable for which the pdf is calculated) are the shift values and the probability obtained is the probability of having a similarity over the threshold at a certain lag starting at any column (or row if using the transposed).

In this case, the values used to compute the probability distribution are the values of the half upper-part of the similarity matrix and upper-triangular half segment. The values

used are shown in figure 7.1.3.1-5 in grey tone. Only these values are used to give equal chance of having a specific value at a certain lag, so only half of the values can be used.

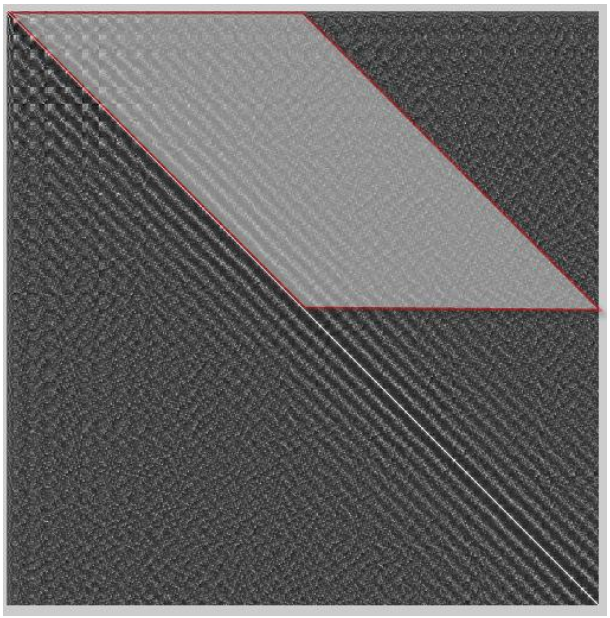


Figure 7.1.3.1-5 Values used for Distance shift pdf.

In figure 7.1.3.1-6 is presented an example of Distance Shift Pdf for a jaccard distance.

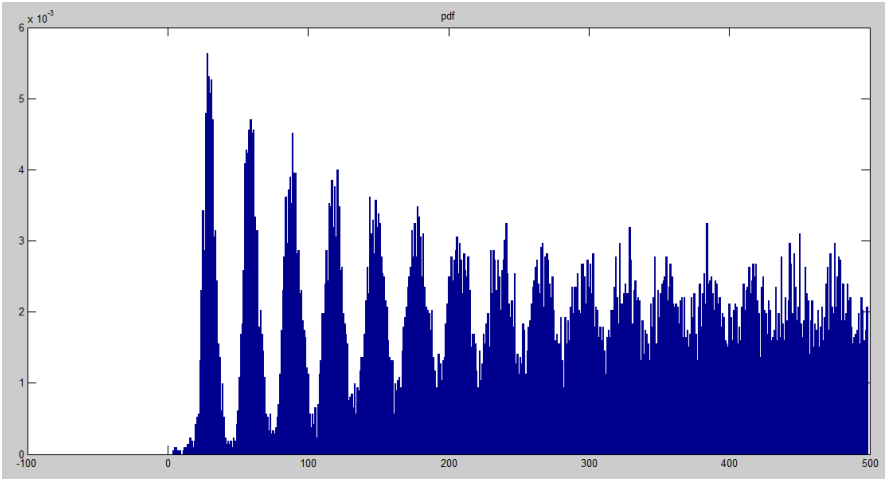


Figure 7.1.3.1-6 Distance shift pdf.

The third probability distribution is obtained pressing the button: “Generate Distance Weighted Shifts Pdf”. This probability distribution is similar to the previous one, but instead of adding one in case of a value over threshold in the computation of values over threshold at a certain lag, it is added the specific similarity value (not one). In this way, greater values contribute with a bigger weight (so the name).

Similarly to the previous case, the values used to compute the probability distribution are the values of the half upper-part of the similarity matrix and upper-triangular half segment. The values used are shown in figure 7.1.3.1-5 in grey tone. Only these values are

used to give equal chance of having a specific value at a certain lag, so only half of the values can be used.

In figure 7.1.3.1-7 is presented an example of Distance Weighted Shifts Pdf for a jaccard distance.

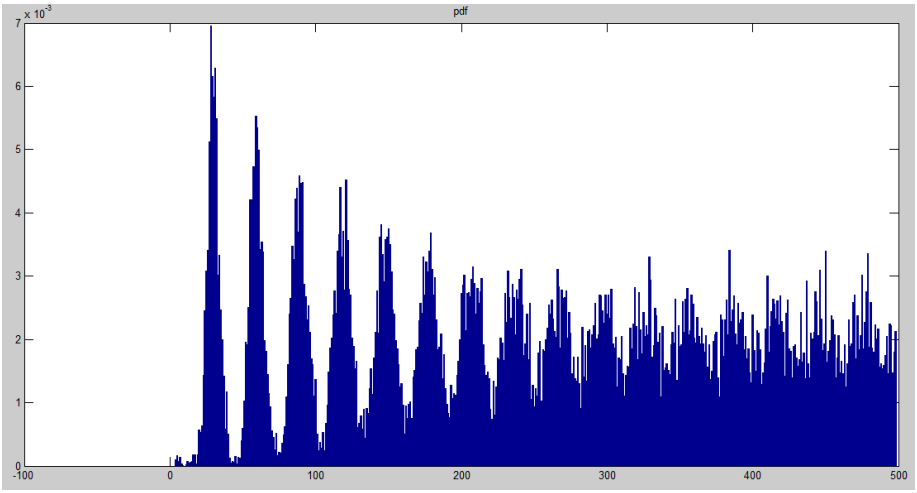


Figure 7.1.3.1-7 Distance Weighted Shifts Pdf.

Once the pdf distributions are calculated, the data is available in a file at the following directory:
[working directory]\files\dist_pdfs

The names of the files contain information on the threshold value used (th), if the diagonal is included or not (ndi), and, if the probability obtained was computed using: (1) the values of the similarity matrix, (2) the shifted values (shift) or (3) the shifted weighted values (shifw).

7.1.3.2 Hilbert application

The Hilbert application is available as a submenu from the calculate distance menu.

In figure 7.1.3.2-1 is presented the window of Hilbert application when it is first shown.

The application has to sections: Bivariate and Multivariate.

Bivariate section

The bivariate section allows choosing two different signals from the previously calculated signal matrix (VV data object in Matlab). So, it is necessary to run previously the main application to calculate the neurons signals.

Once the signals are selected, the “Calculate” button shows the signal difference in the plotting area plus two phase indices: one based in entropy and other based in mean absolute value (see sections 4.2.3.2.4 and 4.2.3.2.5)

The window of the bivariate section is shown in figure 7.1.3.2-2

Multivariate section

The multivariate section allows choosing the phase index measure to obtain an average value for the global phase synchronization of the complete ensemble of signals.

The two measures available are entropy and KL divergence. A radio-button box allows choosing the measure, and after pressing the "Calculate" button two results are available:

- The global average measure (based on entropy or KL divergence)
- A plot of the phase distribution measure vs. time.

The application allows also visualizing the phase distribution at specific times. There are two ways to choose the time:

- Button "Plot phase distribution at time": Pressing this button, the plot for the phase distribution at the specific time is provided in the plotting area to the right. The time is set in the textbox at the right of the button (see figure 7.1.3.2-3)
- Slider: The slider allows changing the time in a dynamic way. The time chosen appears to the right. The plotting area is updated as the time is changed (see figure 7.1.3.2-4)

To use this part of the window it is necessary first to do the computation of phase distribution for each time, which is done when pressing the "Calculate" button. So, before using this section is necessary to press the button.

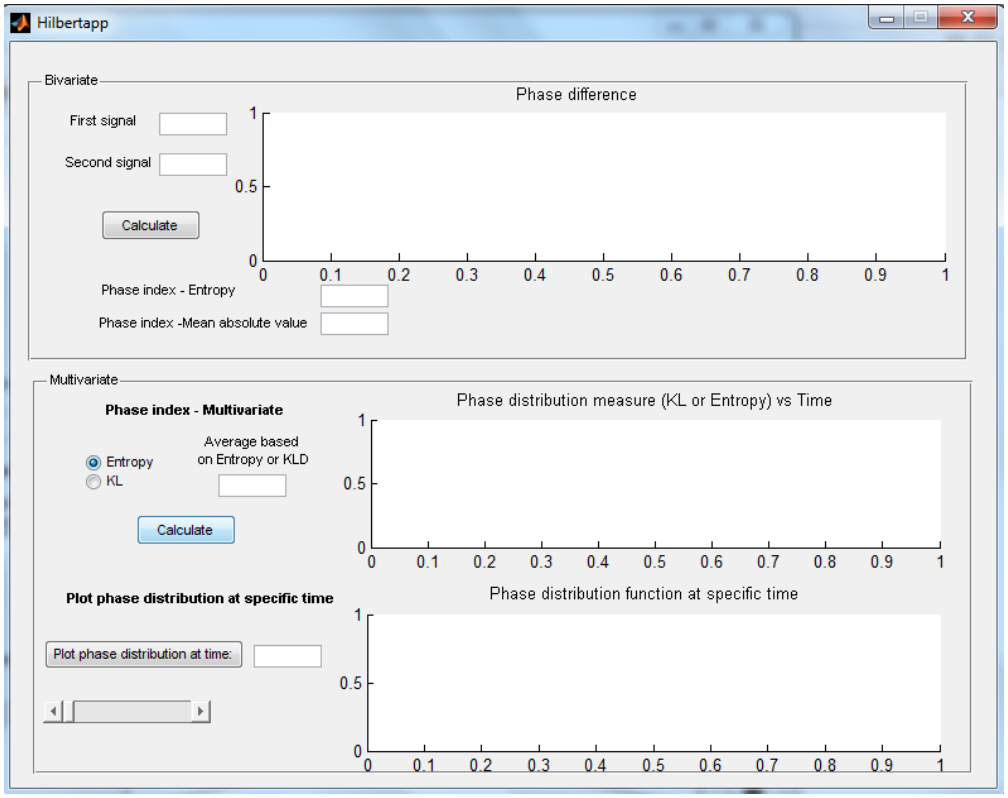


Figure 7.1.3.2-1 Hilbert application.

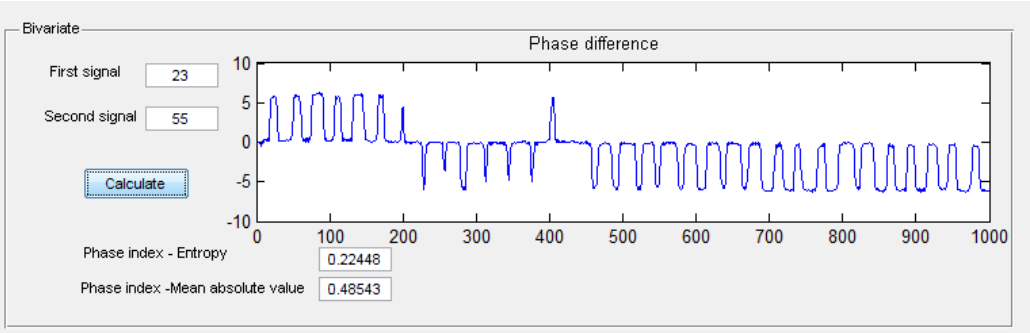


Figure 7.1.3.2-2 Bivariate section with data.

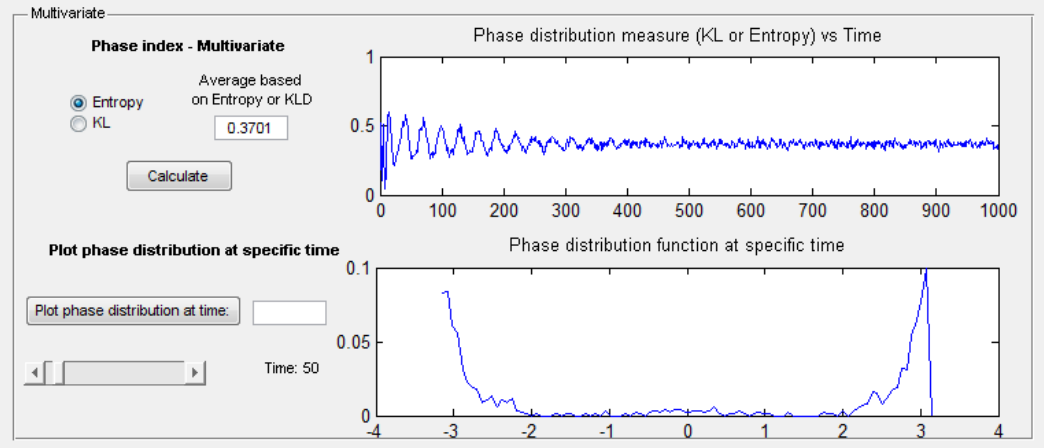


Figure 7.1.3.2-3 Multivariate section with data. It is used the slider to show phase distributions at specific times.

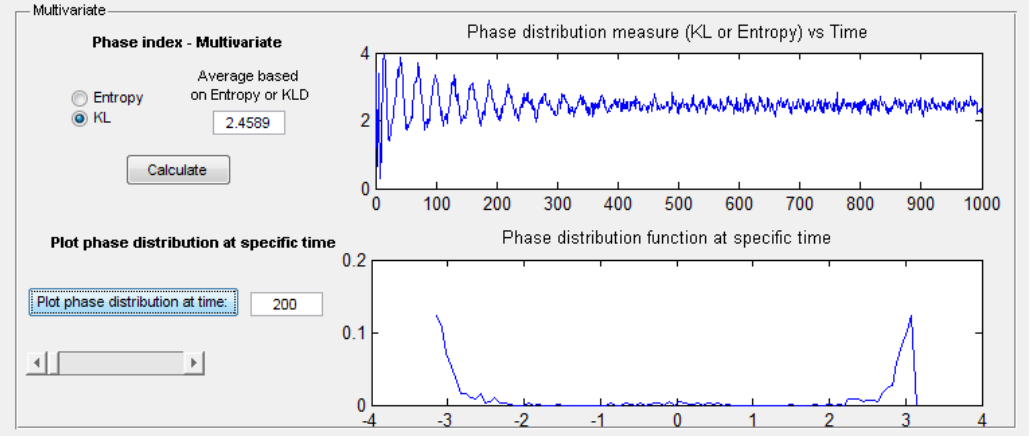


Figure 7.1.3.2-4 Multivariate section with data. It is used the plot button to show phase distributions at specific times.

As an example of how the application works, assuming we choose the entropy measure, at each specific time the phases of all neurons are obtained by the application. From the phases at that time, a histogram and associated pdf is computed, the pdf will give the probability of having a specific phase in all neurons at that specific time. The graph for this pdf (phase distribution at specific time) is given in the second plot, the one named: “Phase distribution function at specific time”.

Once the application has got all the pdf (phase distribution at each time), it applies an entropy computation to each specific pdf (one per time index). The entropy for each pdf at each specific time is the function shown at the first plot, the one named: “Phase distribution measure (KL or Entropy) vs. Time”.

7.1.3.3 Divergence application

The divergence application is available as a submenu from the calculate distance menu.

In figure 7.1.3.3-1 is presented the application’s window when it is first shown.

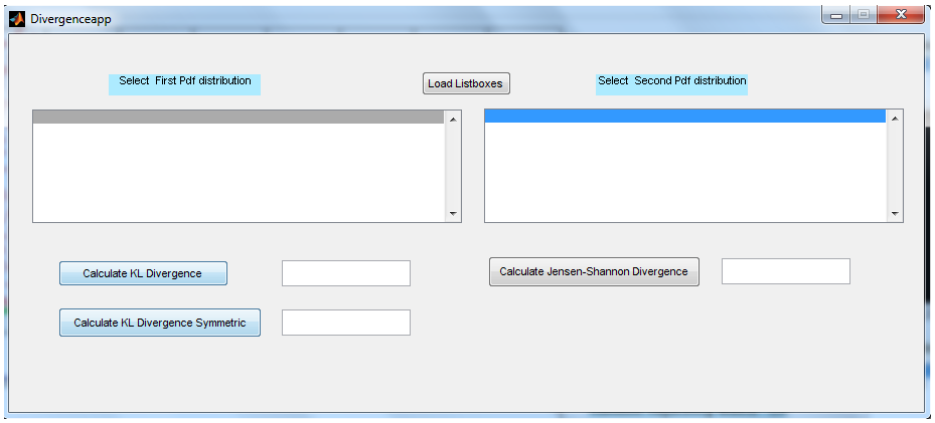


Figure 7.1.3.3-1 Divergence application.

First it is necessary to load both list boxes from the probability distributions previously calculated. So, to launch this application it is first necessary to run the Similarity application or Hilbert application to generate some of the probability distributions available there.

Then, it is necessary to select the first and second probability distributions to apply the divergence method chose. Once the distributions are selected, the buttons below provides different divergences:

- Kullback–Leibler divergence
- Kullback–Leibler symmetric divergence
- Jensen-Shannon divergence

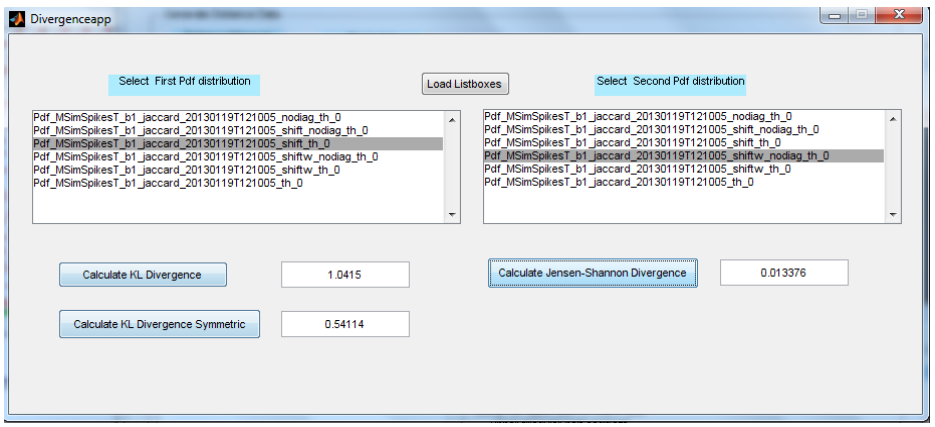


Figure 7.1.3.3-2 Divergence application with data.

8 Computational results

In the following sections are presented several computational experiments made in order to validate some of the theoretical results given previously.

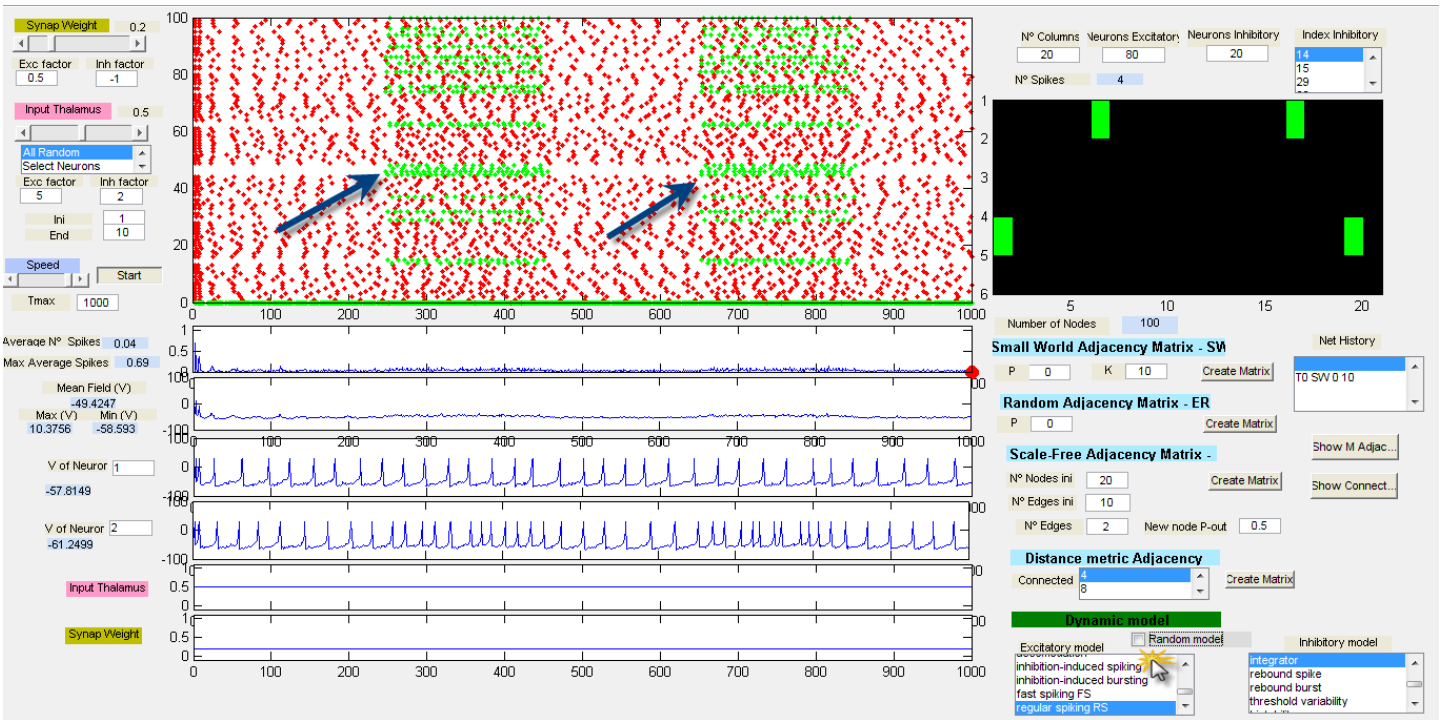
All the computational work has been done using the SyncLab toolbox and additional code in Matlab which is made available as part of this work.

8.1.1 Dependence on dynamic model

The following results show the dependence of synchronization on the neuron dynamic model.
The results seem to indicate that the more random the dynamic model is selected for the different neurons, the more synchronization will be obtained.

In the following figure is shown the simulation of 100 neurons. The changes of behavior in the diagram (blue arrows) are consequence of changes in the neurons dynamic model (tick box at bottom-right). The ranges of more synchronization correspond to changes from a specific model (regular spiking for excitatory and integrator for inhibitory) to a random model.

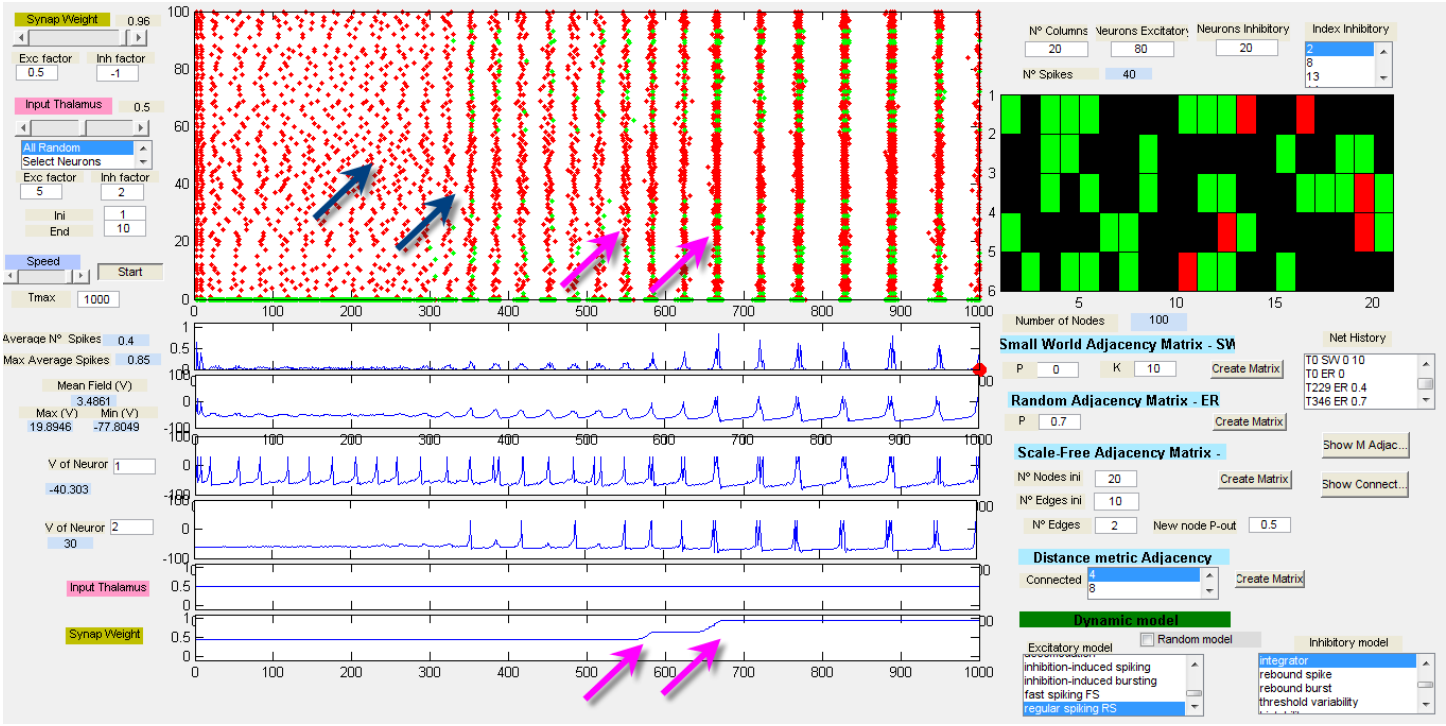
The set of parameters for the simulation appear also at the figure.



8.1.2 Dependence on average degree of network and synaptic weights

The following results show the dependence of synchronization on connectivity level and synaptic weights.

As it can be seen we have a random connectivity, starting with a probability of connection (p) of zero and increasing it to 0.4 and 0.7 (blue arrows). With a p=0.7 the increase in synchronization can be appreciated. With a further increase in the connectivity weight (purple arrows) the increase in synchronization is even more pronounced.

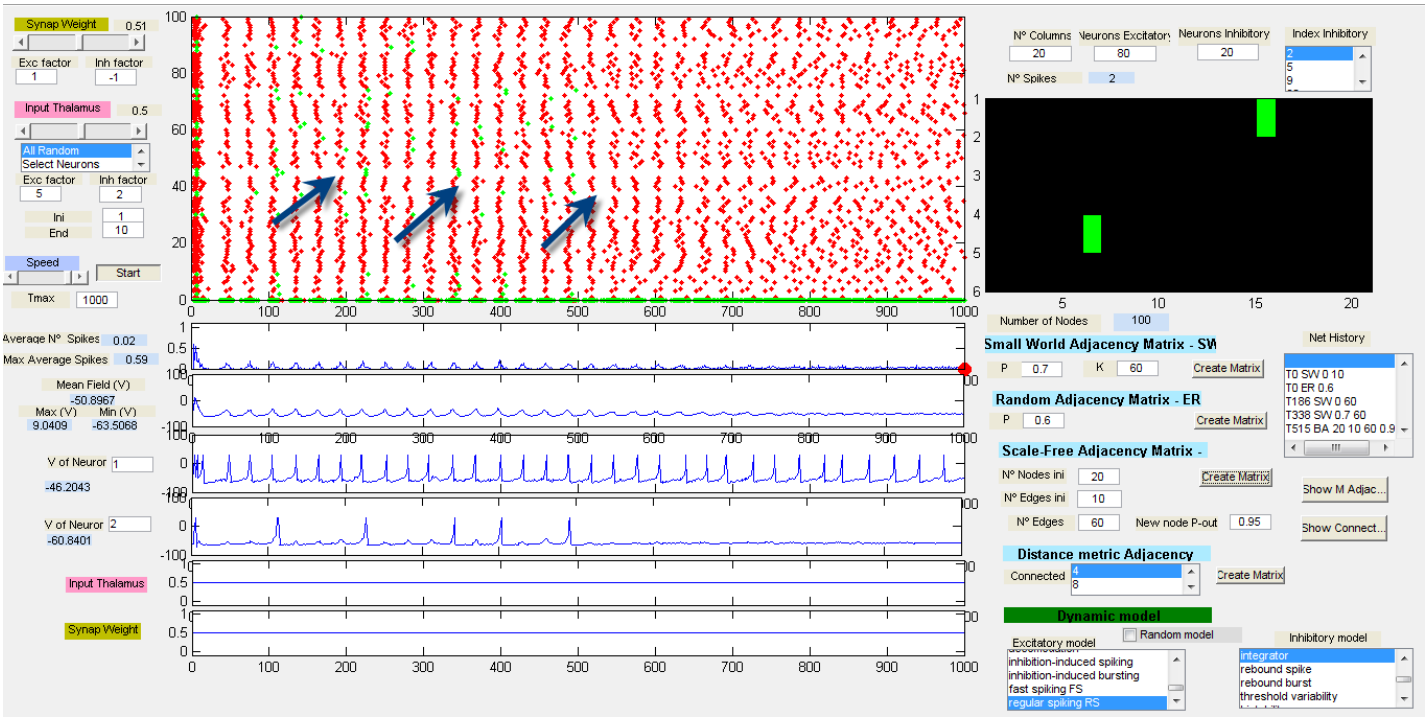


8.1.3 Dependence on network type

In the following figure can be seen as the ring, small world and random network are more stable under synchronization that the BA scale free network.

Starting with a random network with a p=0.6, at the points indicated (blue arrows) the network changes to small world p=0,K=60, and then to small world p=0.7,K=60 and finally to BA scale free with a N=60 for the number of edges (most of them outward). So, the average network degree is maintained.

It can be seen that the BA network is the less stable to maintain synchronization.



8.1.4 Dependence on synchronization index (spectral graph methods)

The “network eigenvalues” application can be used to prove that the “synchronization index” and the “normalized spread of eigenvalues” can be used to test the synchronizability of a network structure.

In general the smaller these values are, the greater the synchronizability of the network.

The synchronization index is mainly useful for undirected networks, while the normalized spread of eigenvalues is useful for directed ones.

For a random matrix, the bigger the probability of nodes connections, the bigger the synchronization of the associated system.

The following figure shows the values obtained for the synchronization indices for different values of probability of an ER-Random network.

It can be seen that the bigger the probability associated to the ER-Random network, the smaller the synchronization indices for the Laplacian matrices and the smaller the normalized spread of eigenvalues (in accordance with the theory presented in this work)

	Synchronization Index	Norm Spread Eigenvalues
Adjacency matrix	L2 is zero	Sum matrix (out of diagonal) is zero
Adjacency weighted matrix	L2 is zero	Sum matrix (out of diagonal) is zero
Laplacian matrix	L2 is zero	p=0
Laplacian weighted matrix	L2 is zero	

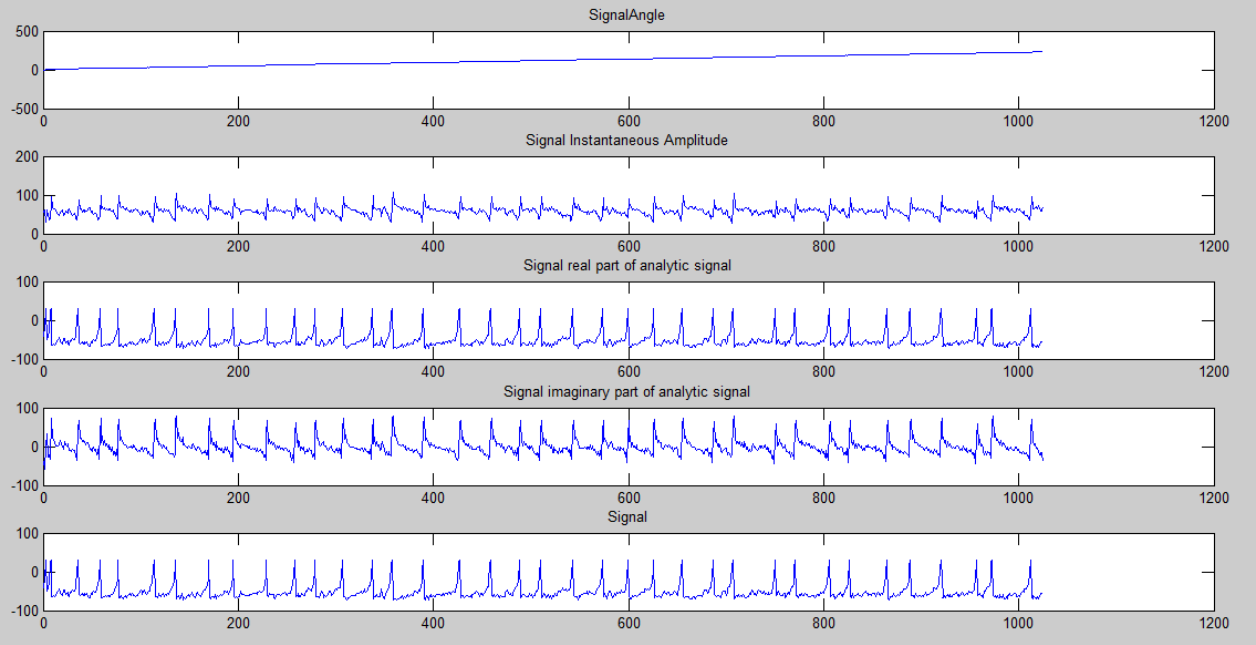
	Synchronization Index	Norm Spread Eigenvalues
Adjacency matrix	112.9793	0.13433
Adjacency weighted matrix	136.8957	4.2921
Laplacian matrix	6.8966	p=0.1
Laplacian weighted matrix	11.3889	

	Synchronization Index	Norm Spread Eigenvalues
Adjacency matrix	414.5626	0.013657
Adjacency weighted matrix	504.1851	1.2532
Laplacian matrix	1.6119	p=0.5
Laplacian weighted matrix	3.1396	

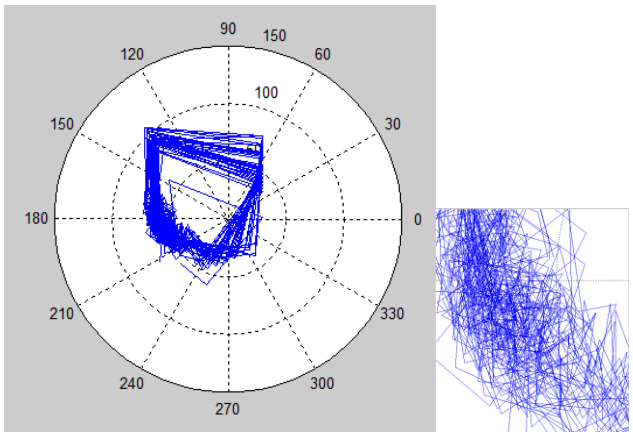
	Synchronization Index	Norm Spread Eigenvalues
Adjacency matrix	1585.7302	0.0017821
Adjacency weighted matrix	1790.1698	1.0721
Laplacian matrix	1.2152	p=0.9
Laplacian weighted matrix	2.4102	

8.1.5 Application of Hilbert transform to continuous and spike signals

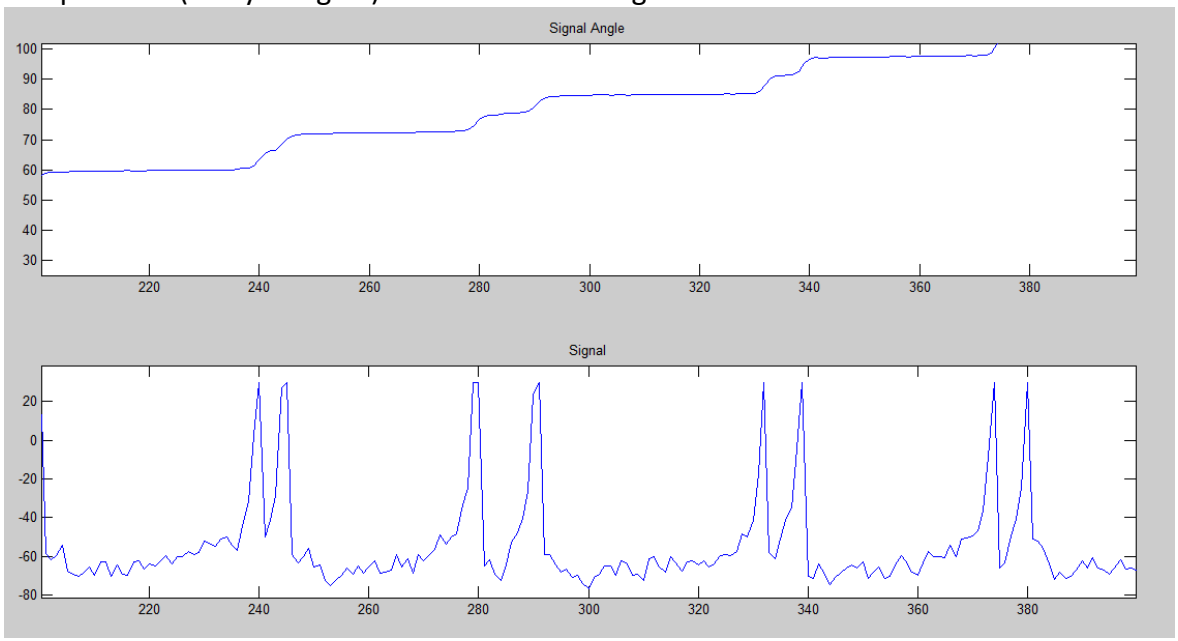
In the following figure it is presented the different signals obtained after computing the analytic signal of a signal:

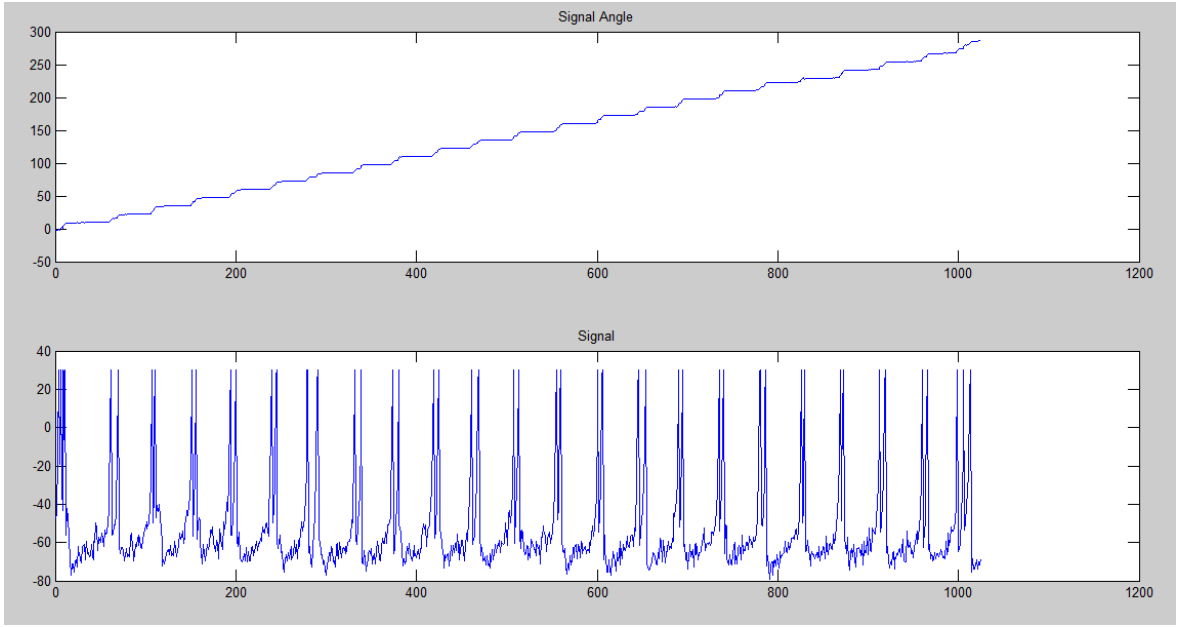


The following one provides the polar representation of the analytic signal with a detail (right) of the more dense area.



In the following figures are presented the signal angle in detail after Hilbert transform computation (analytic signal) for two different signals:



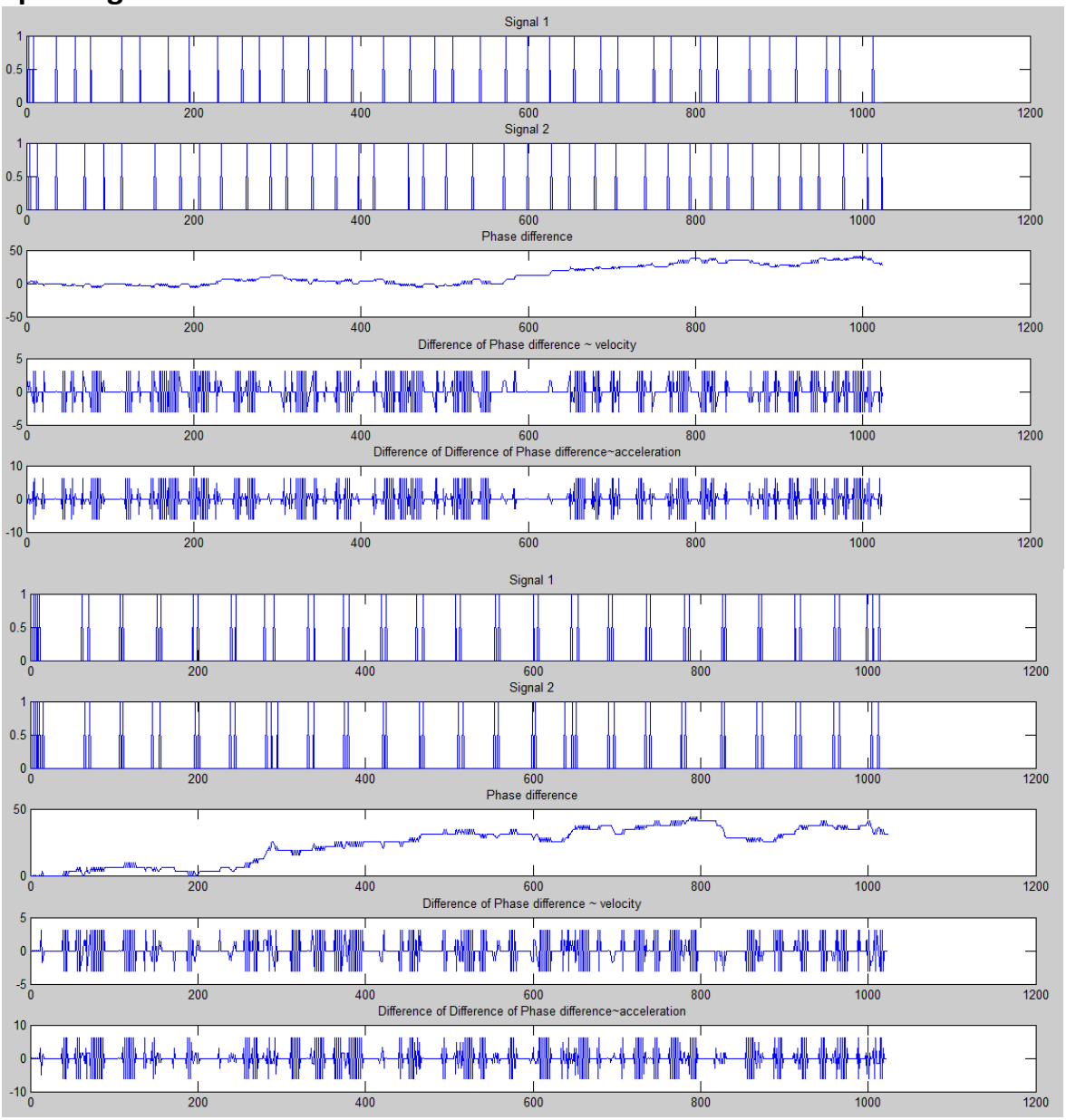


Hilbert transform does not provide adequate results when the signals are represented as spikes (0-1). In the following diagrams are presented the results when comparing two different sets of signals. In the first set the two signals are poorly synchronized, in the second the signals are much more synchronized.

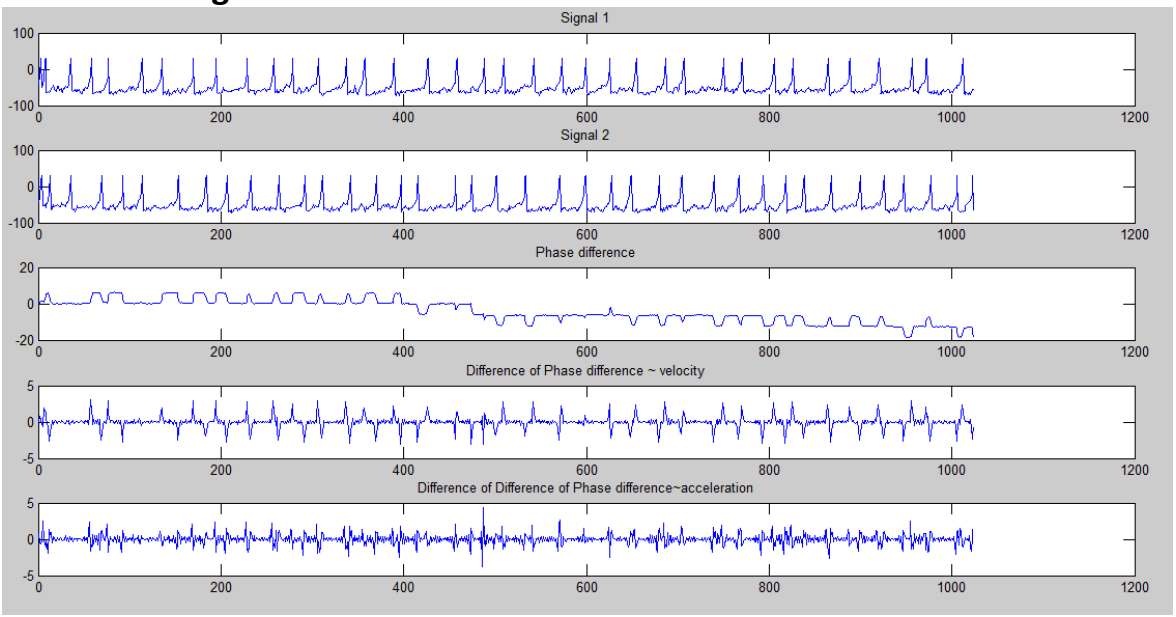
The results obtained when one applies the Hilbert transform to the analog signals are in accordance with the expected results. When the same analysis is applied to the correspondent spike (0-1) transformed signals, the results are not adequate.

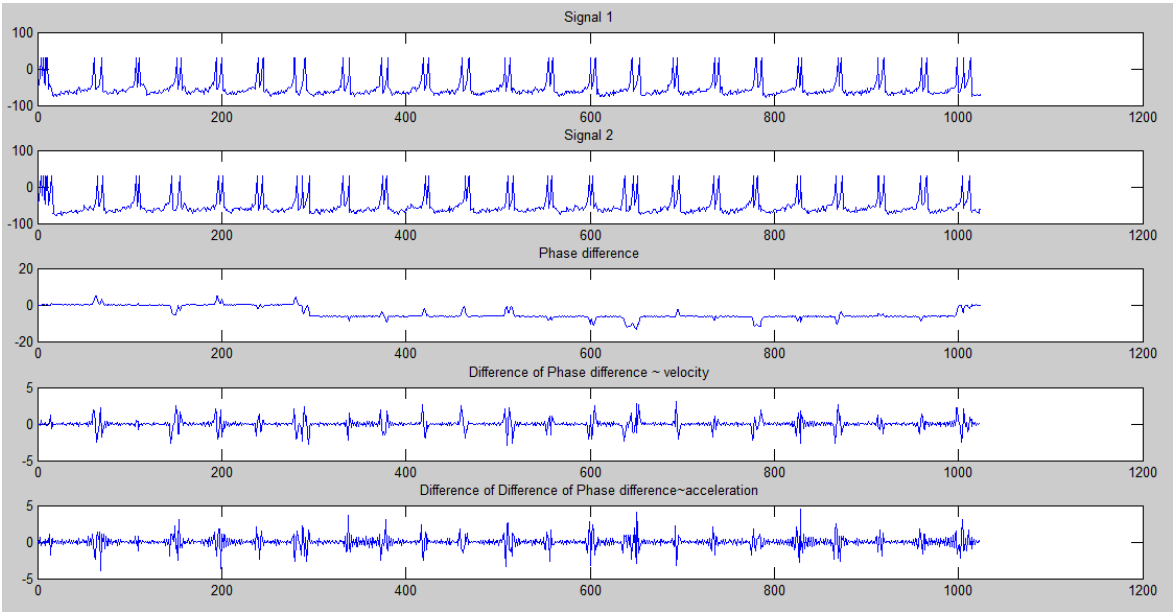
The following results include the calculation of the velocity and acceleration of the phase difference to observe also the phase difference behavior.

Spike signals



Continuous signals





Using spikes (0-1)

	First set –poorly synchronized	Second set – mostly synchronized
Mean of absolute value of phase difference	13.7742	23.5972
Mean of absolute value of difference of phase difference	1.3041	1.1687
Mean of absolute value of second difference of phase difference	2.3258	2.1626

Using continous signals

	First set –poorly synchronized	Second set – mostly synchronized
Mean of absolute value of phase difference	5.7874	4.6019
Mean of absolute value of difference of phase difference	0.4614	0.3622
Mean of absolute value of second difference of phase difference	0.5332	0.4809

The second set has to have the value associated to the average of absolute value of phase difference smaller than the first set, as it is more synchronized. In contrast, when using the spikes (0-1) representation the results are reversed.

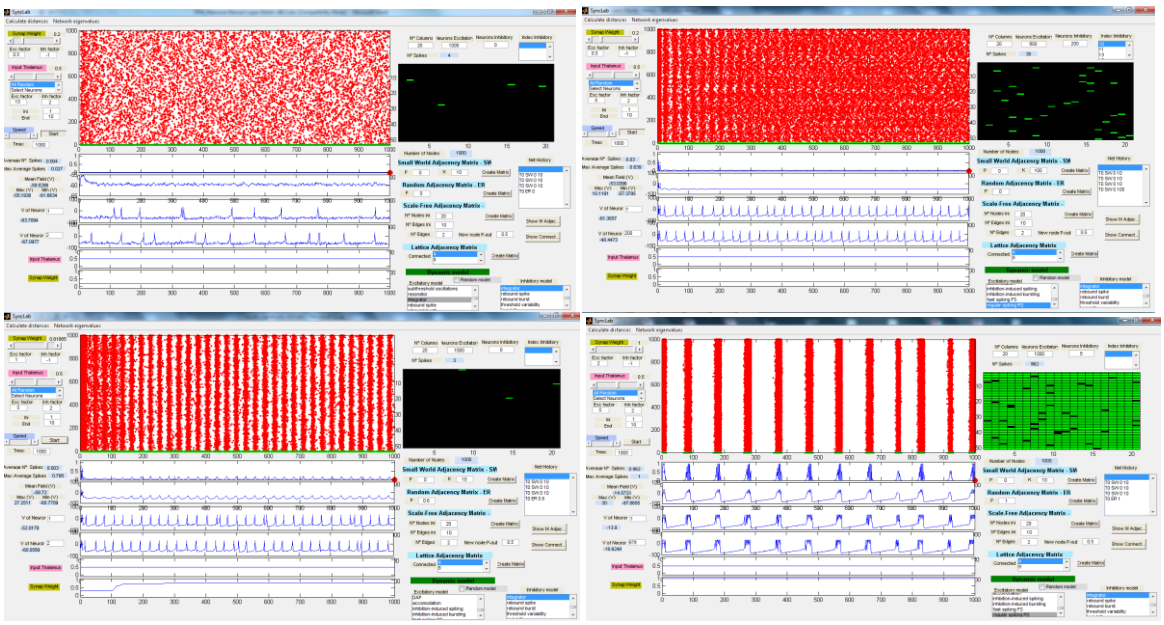
8.1.6 Shifted recurrence diagram results

This section explores the capabilities of the shifted recurrence diagram (SRD) method to detect synchronization in a signals ensemble.

Several pre-built signals have been used:

- **Regular Synchronous (RS):** These signals have been produced with the Matlab toolbox starting with a non-connected set of 1000 neurons with a random input signal for each of the neurons to produce a random signals ensemble. Then it has been increased (incrementally) the coupling weight and degree of connectivity of the network, producing three additional signals ensembles with: weak, medium and strong synchronization.

  In the next figure are presented the four ensembles used for RS synchronization, from left to right and up to bottom: random (no synchronization), weak, medium and strong synchronization ensembles.



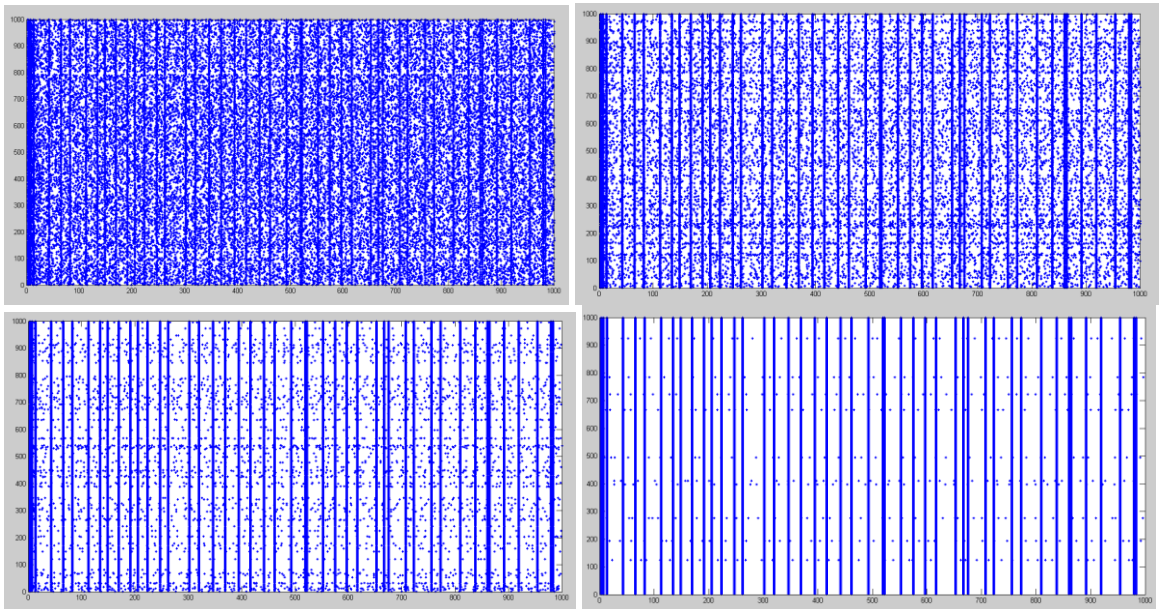
  In all cases the signals present a periodic or quasi-periodic recurrence activity (except the first which is not periodic). The signal coming from each neuron is different to all other signals but the global activity forms a periodic dynamic.

- **Irregular Synchronous (IS):** These signals have been produced starting from the same random ensemble of 1000 neurons used in the RS case. From this ensemble, we select a neuron signal and start to substitute a percentage of other neurons signals with the chosen one. In this way, it has been produced several ensembles with a different percentage of repetition of the one signal chosen for substitution. There are ensembles with 30%, 70%, 90%, 99% and 100% as percentage of

identical signals in the ensemble. For example, the 100% repetition case is an ensemble with all neurons signals been identical.

The repetition of equal signals in the ensemble is done in a random manner, meaning that the identical signals occupy random positions in the ensemble matrix rows of signals.

In the next figure are presented several ensembles (from left to right and up to bottom) for 30%, 70%, 90% and 99% repetition percentage.



The signals have been produced using a specifically developed Matlab code; they have not been generated by the Matlab toolbox as part of a dynamical system simulation.

The above signals have been used to identify their synchronization level using the shifted recurrence diagram (SRD) method (section 4.2.11.2) and the Matlab Distanceapp.m function accessible through the SyncLab Toolbox.

In figure 8.1.6-1 are presented the values obtained after applying SRD to the above signals. We use two different distance measures (correlation and jaccard). In the figure are presented the similarity distances measures applied to the rows of the activity matrix, that is, position (i,j) of the similarity matrix is the distance between row i and j of the activity matrix which is the distance between the activity of neuron i and j for all simulation time.

The indices presented in the figure are:

- Mean similarity values: Average value of all values in the similarity matrix.
- Index entropy of distance pdf (threshold): Entropy of the probability density function for the probability of having a certain similarity value over the threshold. The random variable range is the similarity value and the probability is the probability of having a certain similarity value. The values below the threshold are considered zero. In the figure are presented the indices for two threshold values: zero and the mean value of the similarity values.
- Index entropy of shifted distance pdf (threshold): Entropy of the probability density function for the probability of having a significant similarity value at a

certain time shift. A significant value is a value that is greater than the established threshold. The random variable range is the time shift and the probability is the probability of having a significant similarity value for that shift. In the figure are presented the indices for two threshold values: zero and the mean value of the similarity values.

- Index entropy weighted shifted of distance pdf (threshold): Entropy of the probability density function for the probability of having a significant similarity value at a certain time shift. A significant value is a value that is greater than the established threshold. The random variable range is the time shift and the probability is the probability of having a significant similarity value for that shift.
  The difference with the previous case is the way we compute the significant value, in the previous case we add one each time a significant value is found for any allowable shift in the matrix, while in this case we add the similarity value itself.
  In the figure are presented the indices for two threshold values: zero and the mean value of the similarity values.

Below each block of index values is shown a small function depict which makes easy to see the evolution of values. The background color indicates if the evolution of the indices is in accordance with the evolution of synchronization level, a green color indicates that the index is working properly in accordance with synchronization increase.

	Similarity distance (between neurons)													
	Correlation							Jaccard						
IS signals	Mean similarity values	Index Entropy of distance pdf (threshold=0)	Index Entropy of shifted distance pdf (threshold=0)	Index Entropy of weighted shifted distance pdf (threshold 0)	Index Entropy of distance pdf (threshold=mean)	Index Entropy of shifted distance pdf (threshold=mean)	Index Entropy of weighted shifted distance pdf (threshold=mean)	Mean similarity values	Index Entropy of distance pdf (threshold=0)	Index Entropy of shifted distance pdf (threshold=0)	Index Entropy of weighted shifted distance pdf (threshold 0)	Index Entropy of distance pdf (threshold=mean)	Index Entropy of shifted distance pdf (threshold=mean)	Index Entropy of weighted shifted distance pdf (threshold=mean)
1000x1000 Random	0.13281	0.17173	0.00032289	0.00033296	0.5133	0.00048097	0.00048857	0.027147	0.44203	0.00033854	0.00038402	0.64257	0.00052207	0.00054937
1000x1000 same signal 30% (random permutation horizontally)	0.2103	0.40335	0.00032289	0.00046816	0.31131	0.0012943	0.0013046	0.11522	0.57707	0.00033248	0.00091873	0.93324	0.0013598	0.0013682
1000x1000 same signal 70% (random permutation horizontally)	0.55266	0.55236	0.00032289	0.00034799	0.84953	0.00036489	0.00036489	0.50445	0.65116	0.00032593	0.00035991	0.84953	0.00036489	0.00036489
1000x1000 same signal 99% (random permutation horizontally)	0.98249	0.97056	0.0003229	0.00032298	0.97879	0.00032302	0.00032302	0.9809	0.97239	0.0003229	0.000323	0.97879	0.00032302	0.00032302
1000x1000 Random	0.13281	0.17173	0.00032289	0.00033296	0.5133	0.00048097	0.00048857	0.027147	0.44203	0.00033854	0.00038402	0.64257	0.00052207	0.00054937
1000x1000 weak sync (vertical)	0.18486	0.19053	0.00032289	0.00033378	0.53439	0.00050601	0.00050987	0.020636	0.58768	0.00037903	0.00045708	0.67484	0.00045464	0.00051944
1000x1000 medium sync (vertical)	0.22329	0.16864	0.00032289	0.00033348	0.51298	0.00046	0.00046378	0.066226	0.38365	0.00032492	0.000356	0.56838	0.00047408	0.0004874
1000x1000 strong sync (vertical)	0.88081	0.1737	0.00032289	0.00032296	0.67712	0.00043218	0.00043135	0.5717	0.17434	0.00032289	0.00032451	0.55223	0.00045428	0.00045253
RS signals														

Fig 8.1.6-1 SRD indices between neurons activity for all time.

In Fig 8.1.6-2 are presented a similar data set but this time the distances are taken between columns of the activity matrix, that is, position (i,j) of the similarity matrix is the distance between column i and j of the activity matrix which is the distance between the activity of all neuron at time i and all neurons at time j.

In figure 8.1.6-1 the meaningful results correspond to the IS signals because these signals have a row repeating pattern (but not periodic). For these signals only the mean value and the index entropy of distance pdf provide correct results.

In figure 8.1.6-2 the meaningful results correspond to the RS signals because these signals have a column repeating pattern (and this time the pattern is periodic o quasi periodic). For these signals only the index entropy of distance pdf and the index entropy of weighted distance pdf provide correct results, and only for the jaccard distance. For these later indices the results are correct independently of the threshold used.

	Similarity distance (between time)-using Transposed Matrix													
	Correlation							Jaccard						
IS signals	Mean similarity values	Index Entropy of distance pdf (threshold=0)	Index Entropy of shifted distance pdf (threshold=0)	Index Entropy of weighted shifted distance pdf (threshold 0)	Index Entropy of distance pdf (threshold=mean)	Index Entropy of shifted distance pdf (threshold=mean)	Index Entropy of weighted shifted distance pdf (threshold=mean)	Mean similarity values	Index Entropy of distance pdf (threshold=0)	Index Entropy of shifted distance pdf (threshold=0)	Index Entropy of weighted shifted distance pdf (threshold 0)	Index Entropy of distance pdf (threshold=mean)	Index Entropy of shifted distance pdf (threshold=mean)	Index Entropy of weighted shifted distance pdf (threshold=mean)
1000x1000 Random	0.35074	0.48804	0.0003229	0.00046454	0.67084	0.0043557	0.0049913	0.02191	0.40413	0.0015627	0.002049	0.56293	0.0026816	0.0027401
1000x1000 same signal 30% (random permutation horizontally)	0.35958	0.23425	0.00032289	0.00048186	0.52696	0.0014333	0.0018861	0.021969	0.59985	0.0017317	0.0026612	0.72795	0.0027068	0.0036823
1000x1000 same signal 70% (random permutation horizontally)	0.42782	0.14708	0.0003229	0.00043489	0.48421	0.00088942	0.0011585	0.021949	0.68412	0.0020113	0.002938	0.68848	0.0020914	0.0029698
1000x1000 same signal 99% (random permutation horizontally)	0.50184	0.041684	0.0003229	0.00042168	0.44707	0.00066824	0.00082091	0.014291	0.97573	0.0092514	0.012137	0.97573	0.01067	0.012141
1000x1000 Random	0.35074	0.48804	0.0003229	0.00046454	0.67084	0.0043557	0.0049913	0.02191	0.40413	0.0015627	0.002049	0.56293	0.0026816	0.0027401
1000x1000 weak sync (vertical)	0.27055	0.35347	0.0003229	0.0022189	0.60151	0.040218	0.044138	0.016231	0.59773	0.015214	0.02383	0.63362	0.016784	0.025563
1000x1000 medium sync (vertical)	0.4911	0.48342	0.0003229	0.00039724	0.67535	0.0037975	0.0047038	0.0090607	0.77146	0.0095747	0.025264	0.77146	0.013282	0.027419
1000x1000 strong sync (vertical)	0.50506	0.47884	0.00032289	0.00047537	0.68261	0.0062888	0.01015	0.0074572	0.93009	0.086787	0.12565	0.93009	0.092418	0.12617

RS signals

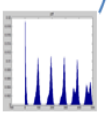


Fig 8.1.6-2 SRD indices between activities of all neurons at different times.

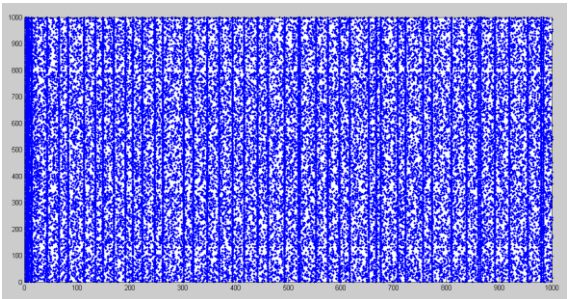
From the above data it can be finally concluded:

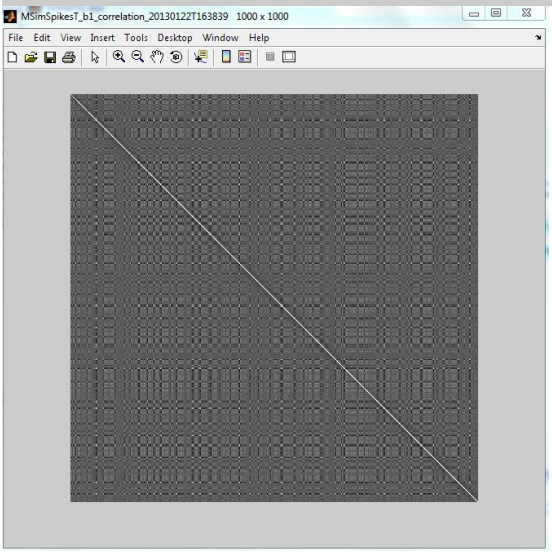
- To detect synchronization, with or without periodicity, the index entropy of distance pdf with the jaccard distance is the best method.
- To detect periodic synchronization the index entropy of weighted distance pdf with the jaccard distance provide correct results, nevertheless the values are very small and the sensitivity seems not to be very good (a great change in synchronization provides an small increase in value). So the interest of the SRD methods can be better directed to identify the synchronization state and to compare synchronization signatures using the distribution of probabilities that the method provides.

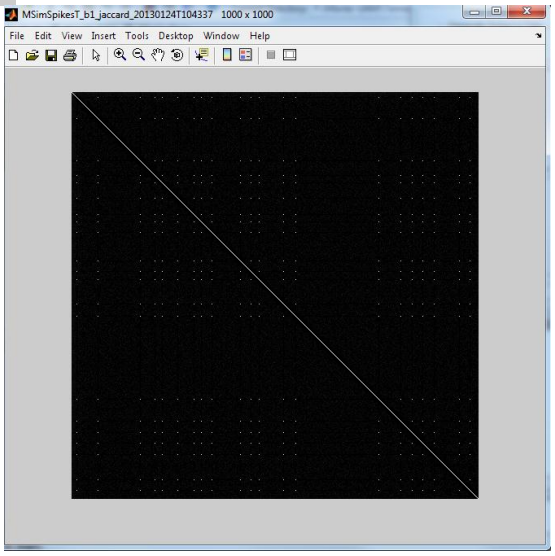
For completeness purposes, in the following diagrams are presented the correlation and jaccard similarity matrices for the RS and IS signals used (the more white the value in the matrix the greater similarity between the corresponding elements):

IS Signals

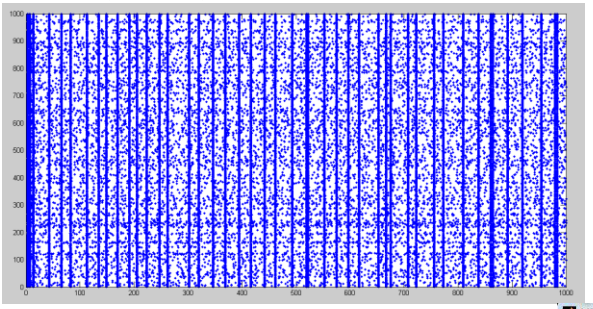
30% equal signals (rows)

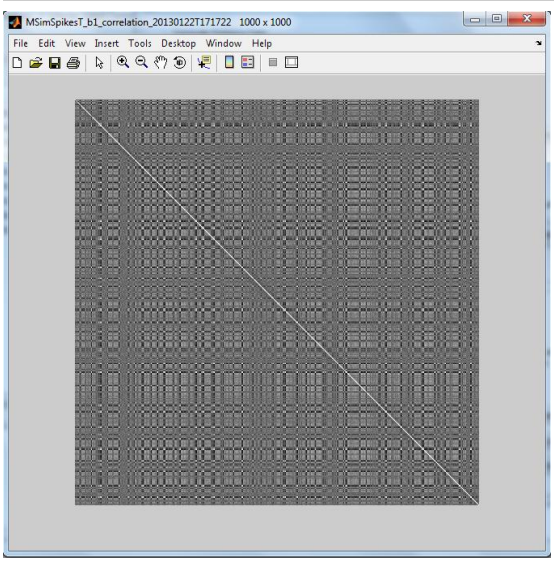


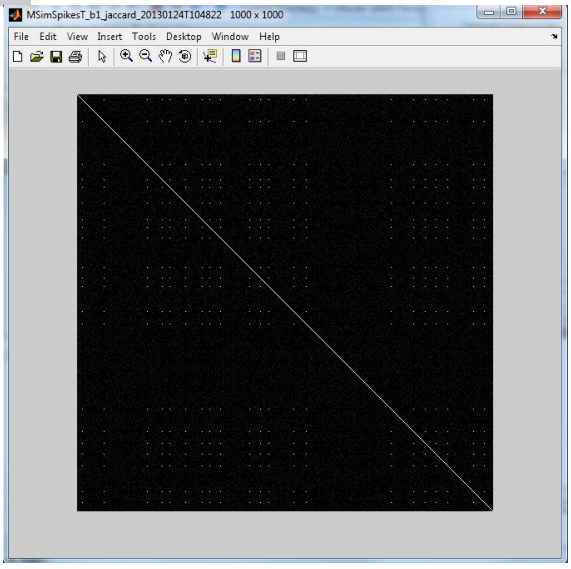




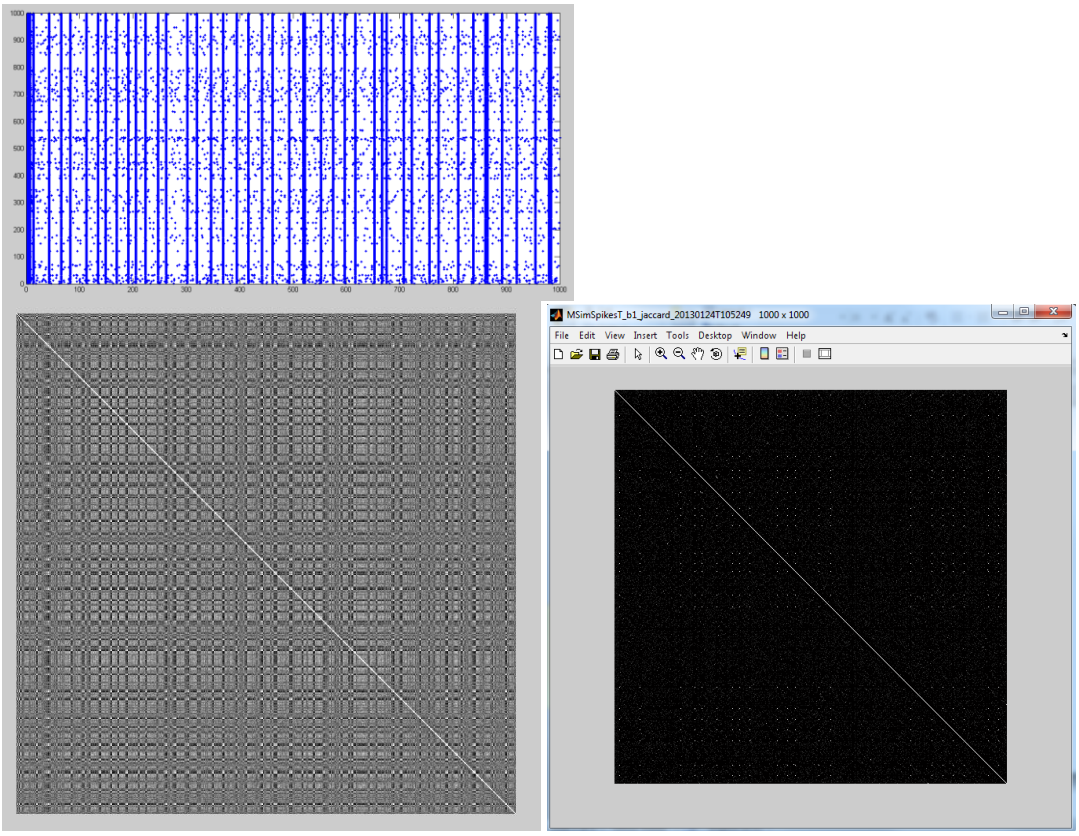
70% equal signals (rows)



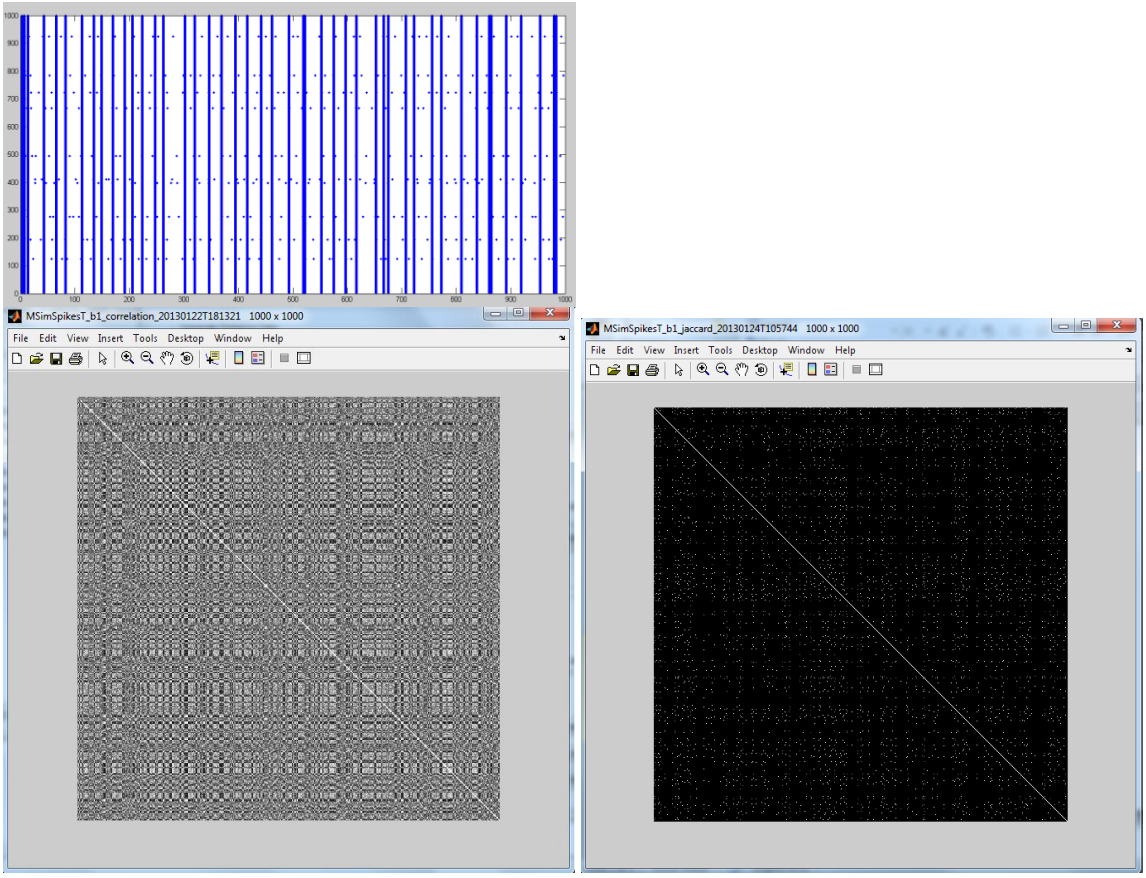




90% equal signals (rows)

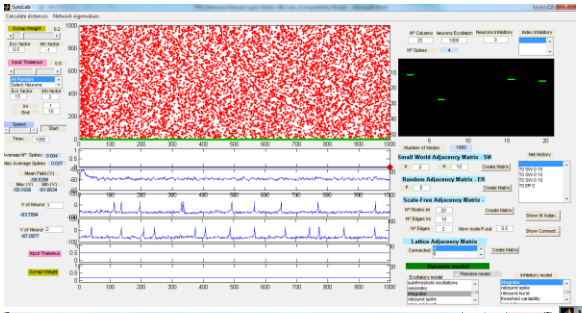


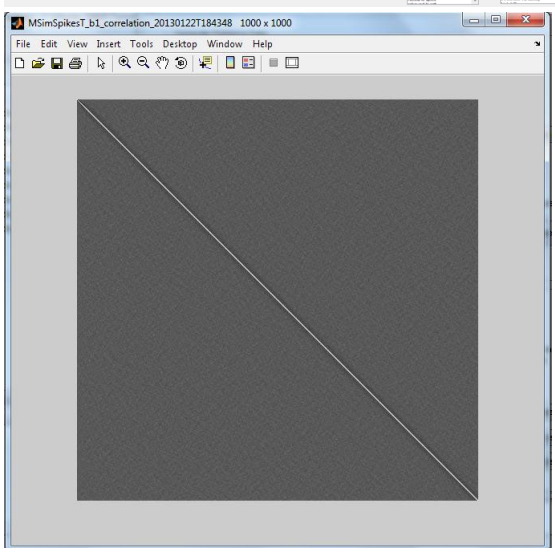
99% equal signals (rows)

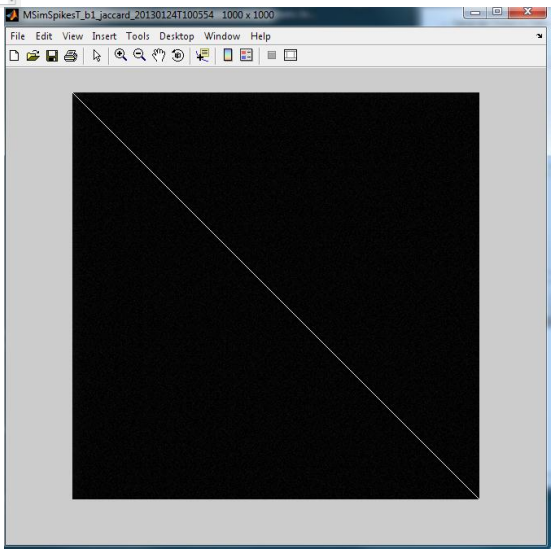


RS Signals

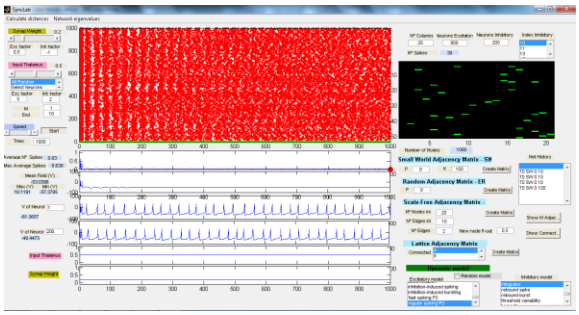
Random signals

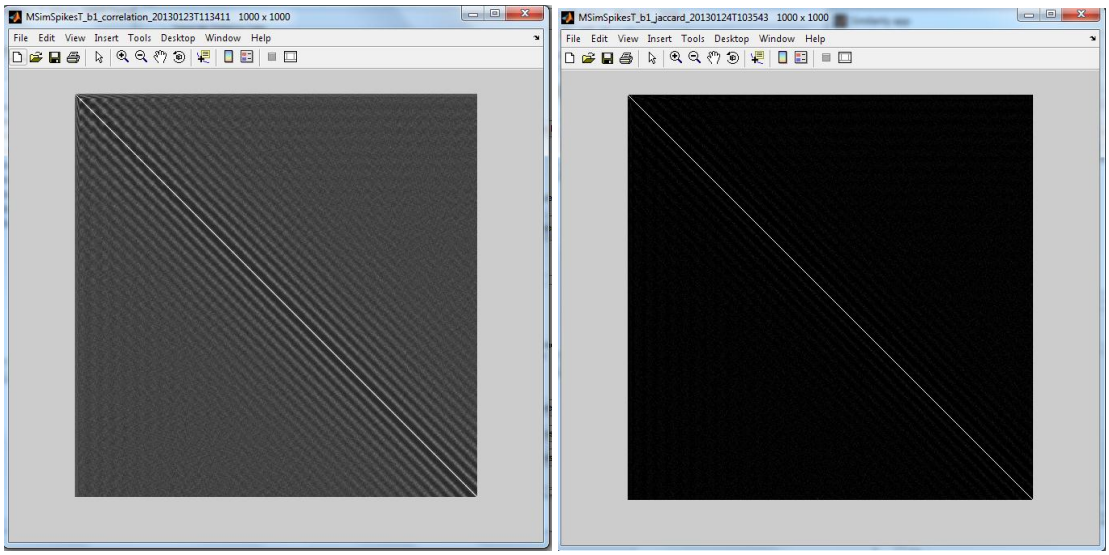




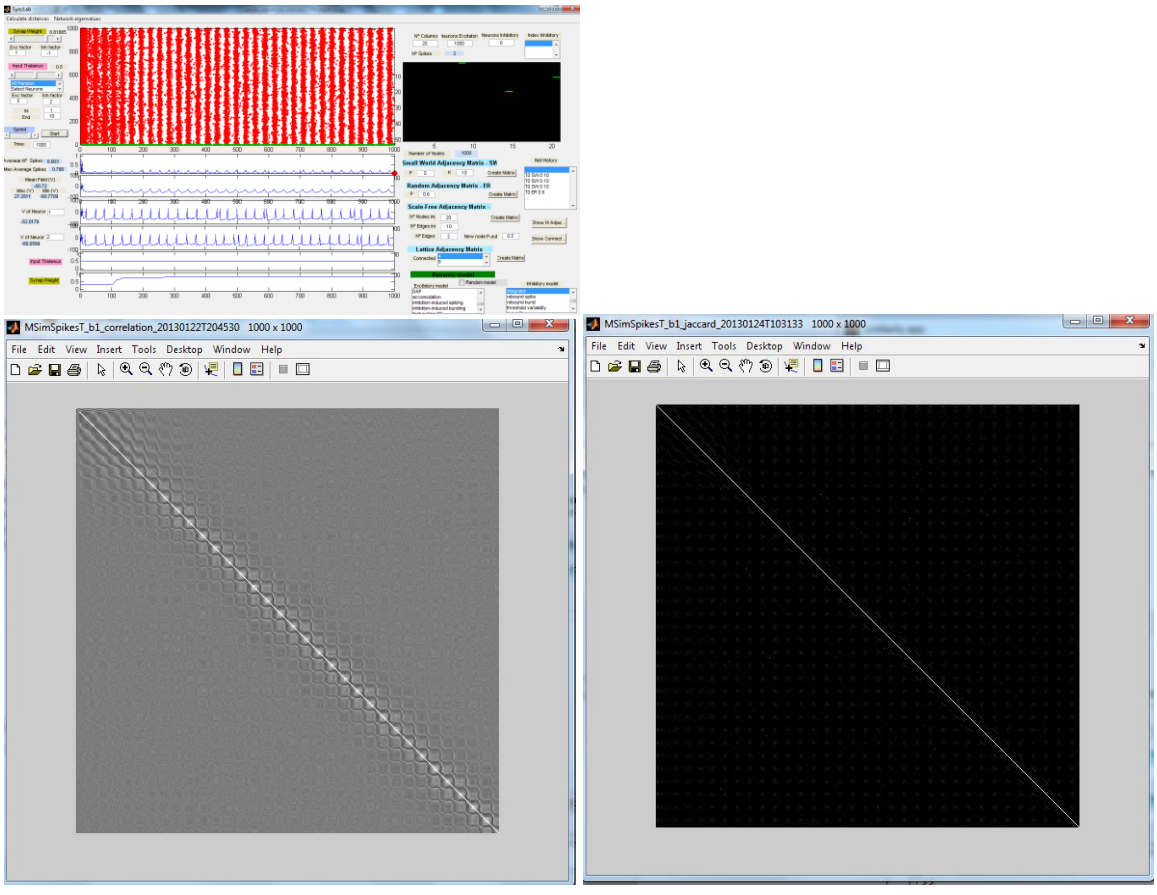


Weak synchronization (columns)

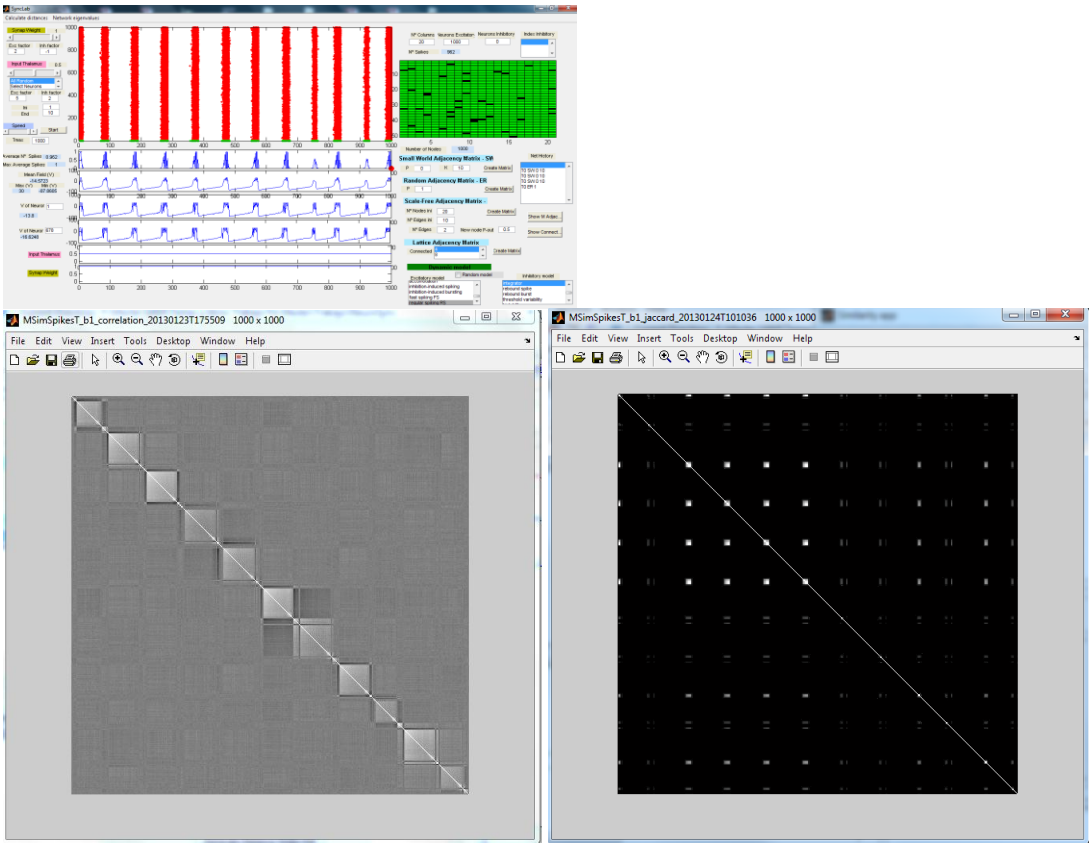




Medium synchronization (columns)



Strong synchronization (columns)



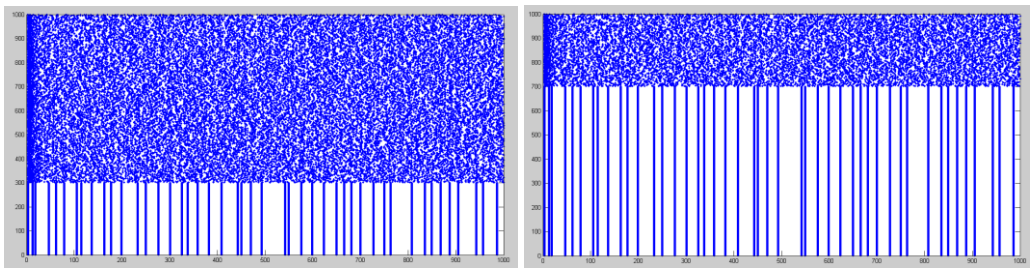
From the above figures it can be observed the great difference between the patterns formed in the similarity matrices for RS and IS signals.

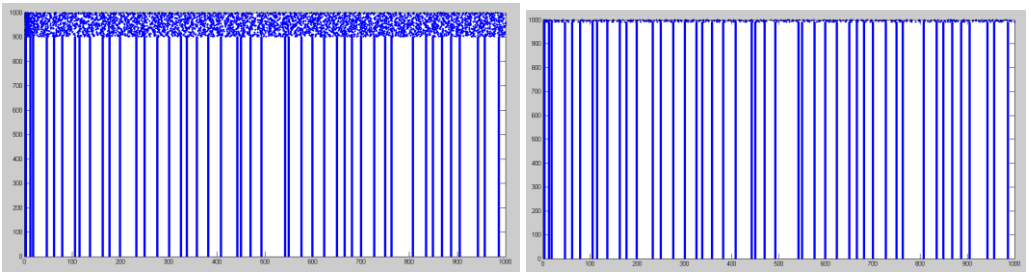
8.1.7 Phase index multivariate results

This section explores the capabilities of the phase index multivariate method (section 4.2.3.2.6) to detect synchronization in a signals ensemble.

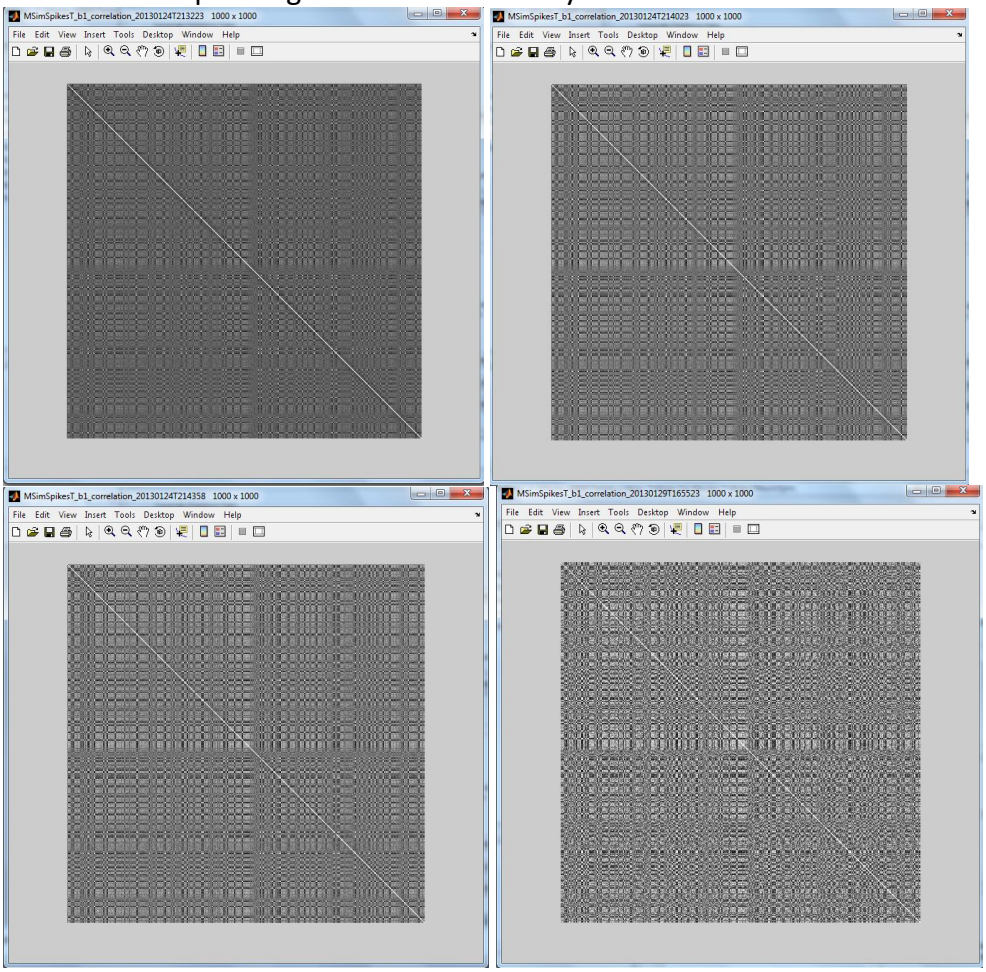
The signals used to test the method are the same pre-built signals of previous section plus an additional set of signals which is described below:

- **Irregular Synchronous (IS)- cluster:** Similar to IS signals but with equal signals placed in contiguous positions.
  In the next figure are presented several ensembles (from left to right and up to bottom) for 30%, 70%, 90% and 99% repetition percentage of the IS signals in cluster.





And the corresponding correlation similarity matrices:



In figure 8.1.7-1 are presented the values obtained after applying the phase index to the IS, RS and IS-cluster signals.
The indices presented in the figure are:

- Phase index-entropy: Average value of an entropy function of time, where the function at time t gives the entropy of the distribution of phase values of all signals at time t. The entropy function is normalized to have values between 0-1.
- Phase index-KL: Identical to previous one but the function is the KL divergence function between the distribution of phase values and the uniform distribution.

The figure presents results also for the **transposed** matrix, which provides the corresponding indices:

- Phase index-entropy (for transposed matrix): Average value of an entropy function of the neuron index, where the function at neuron k gives the entropy of the

distribution of phase values for all time at neuron k. That is, the distribution is for phase values for change along the neurons for all time. That means that in this case the relative position of neurons is important. The entropy function is normalized to have values between 0-1.

- Phase index-KL (for transposed matrix): Identical to previous one but the function is the KL divergence function between the distribution of phase values and the uniform distribution.

Below each block of index values is shown a small function depict which makes easy to see the evolution of values. The background color indicates if the evolution of the indices is in accordance with the evolution of synchronization level, a green color indicates that the index is working properly in accordance with synchronization increase

From the data presented in figure 8.1.7-1 it can be observed that the index behaves well except for the transposed RS signals. Nevertheless, this is not a problem as the phase change along neurons in this case has no special meaning.

SIGNAL - Angle

IS signals

	Phase index	
	Phase index multivariate-KL	Phase index multivariate-Entropy
1000x1000 Random	1.9098	0.28745
1000x1000 same signal 30% (random permutation horizontally)	2.6203	0.39439
1000x1000 same signal 70% (random permutation horizontally)	4.4814	0.67452
1000x1000 same signal 90% (random permutation horizontally)	5.7814	0.87018
1000x1000 same signal 99% (random permutation horizontally)	6.5402	0.98441
1000x1000 same signal 100% (random permutation horizontally)	6.6439	1

RS signals

1000x1000 Random	1.9098	0.28745
1000x1000 weak sync (vertical)	2.8285	0.42574
1000x1000 medium sync (vertical)	3.1184	0.46936
1000x1000 strong sync (vertical)	4.5423	0.68369

1000x1000 Random	1.9098	0.28745
1000x1000 same signal 30%-cluster	2.6192	0.39423
1000x1000 same signal 70% -cluster	4.4787	0.67411
1000x1000 same signal 90% -cluster	5.7805	0.87006
1000x1000 same signal 99%-cluster	6.5396	0.98431

IS signals forming clusters of contiguous signals

Transposed

	Phase index	
	Phase index multivariate-KL	Phase index multivariate-Entropy
1000x1000 Random	1.7462	0.26283
1000x1000 same signal 30% (random	1.9191	0.28886
1000x1000 same signal 70% (random	2.7488	0.41373
1000x1000 same signal 90% (random	3.8958	0.58637
1000x1000 same signal 99% (random	5.1219	0.77092
1000x1000 same signal 100%	6.3204	0.95131

1000x1000 Random	1.7462	0.26283
1000x1000 weak	2.5101	0.37781
1000x1000 medium	2.4851	0.37404
1000x1000 strong	3.6096	0.5433

1000x1000 Random	1.7462	0.26283
1000x1000 same signal 30%-cluster	2.3484	0.35347
1000x1000 same signal 70% -cluster	3.5352	0.53211
1000x1000 same signal 90% -cluster	4.4985	0.67709
1000x1000 same signal 99%-cluster	5.176	0.77906

Fig 8.1.7-1 Phase indices for different signals

In figure 8.1.7-2 are presented the values obtained after applying a similar index to the same signals, but in this case the index is applied over the signal itself and not their

phases. It can be observed that this index behaves well for the signal but not for the transposed one.
The definition of this additional index is:

- Real value index-entropy: Average value of an entropy function of time, where the function at time t gives the entropy of the distribution of signal values of all signals at time t. The entropy function is normalized to have values between 0-1.
- Real value index-KL: Identical to previous one but the function is the KL divergence function between the distribution of values and the uniform distribution.

SIGNAL - Real part

IS signals

	Real value index	
	Real value index multivariate-KL	Real value index multivariate-Entropy
1000x1000 Random	1.2365	0.18612
1000x1000 same signal 30% (random permutation horizontally)	2.1002	0.31611
1000x1000 same signal 70% (random permutation horizontally)	4.2356	0.63752
1000x1000 same signal 90% (random permutation horizontally)	5.7	0.85794
1000x1000 same signal 99% (random permutation horizontally)	6.5352	0.98365
1000x1000 same signal 100% (random permutation horizontally)	6.6439	1
1000x1000 Random	1.2365	0.18612
1000x1000 weak sync (vertical)	2.0215	0.30427
1000x1000 medium sync (vertical)	2.0664	0.31102
1000x1000 strong sync (vertical)	3.79	0.57045

RS signals

Transposed

		Real value index	
		Real value index multivariate-KL	Real value index multivariate-Entropy
1000x1000 Random	1.2263	0.18457
1000x1000 same signal 30% (random	1.2345	0.18581
1000x1000 same signal 70% (random	1.1978	0.18028
1000x1000 same signal 90% (random	1.197	0.18016
1000x1000 same signal 99% (random	1.1653	0.1754
1000x1000 same signal 100%	1.1312	0.17027
1000x1000 Random	1.2263	0.18457
1000x1000 weak	2.2795	0.3431
1000x1000 medium	1.7329	0.26082
1000x1000 strong	1.1443	0.17224

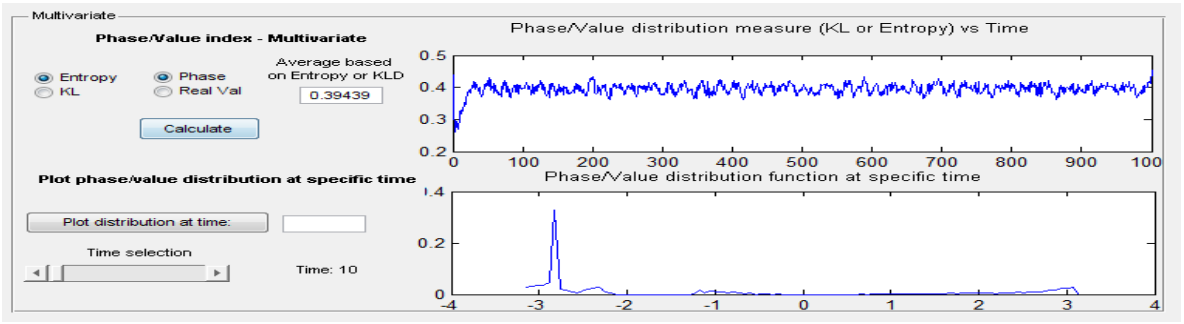
Fig 8.1.7-2 Indices for different signals, using the same technique used for phase index but with just the real part of the signal (the signal itself)

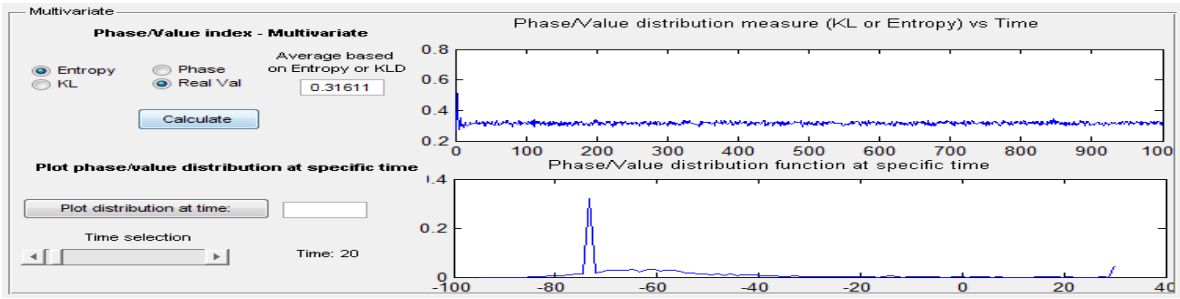
Using both the (1) real value and (2) phase indices for (a) the signals matrix and (b) its transposed is possible to characterize the signal type and its level of synchronization.

For completeness purposes in the following figures are presented the entropy functions obtained for several signals ensembles and their transposed (first it is given the phase index followed by the real value index):

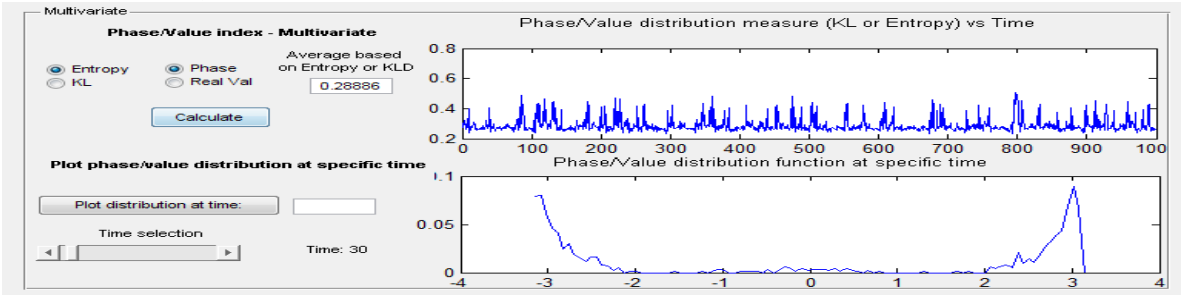
IS Signals

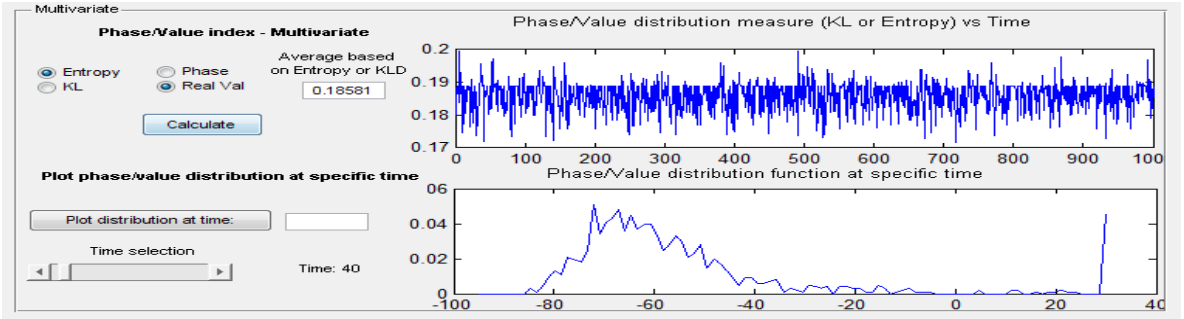
30% equal signals (rows)



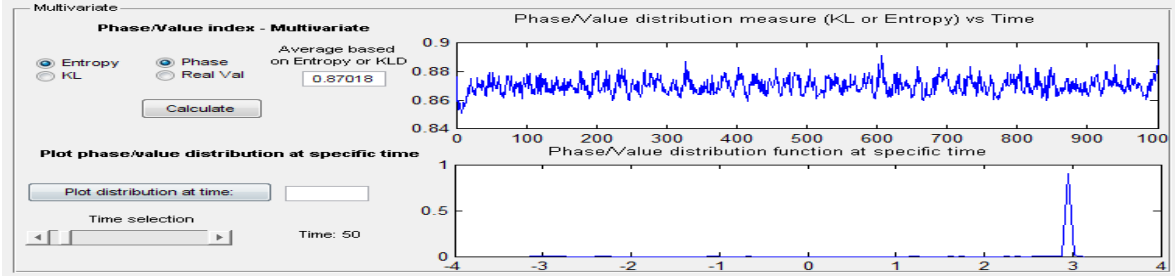


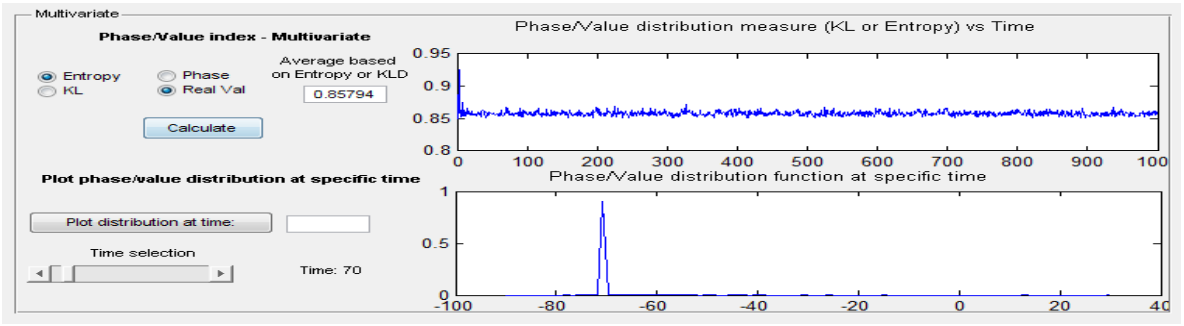
30% equal signals (rows) – transposed



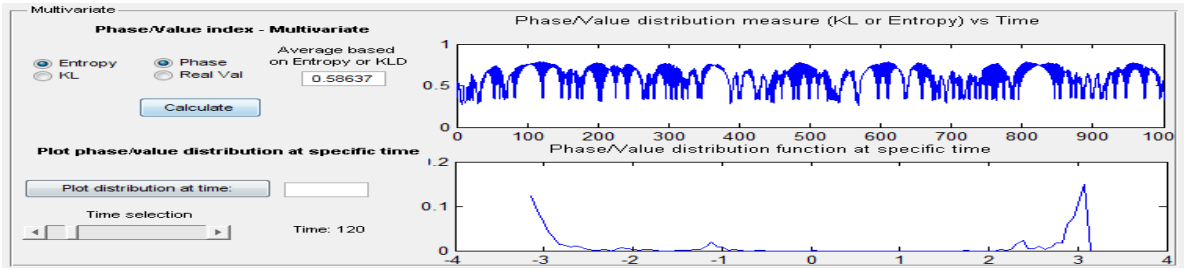


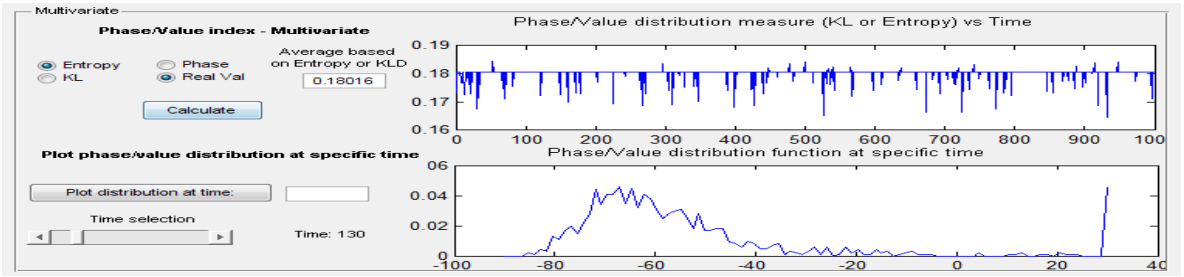
90% equal signals (rows)





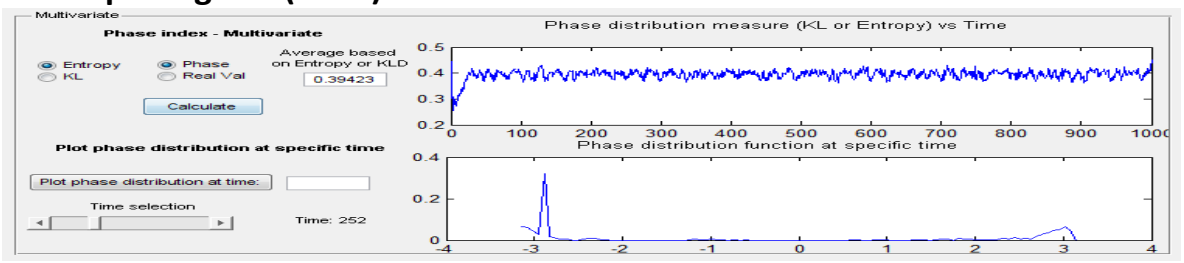
90% equal signals (rows) - transposed

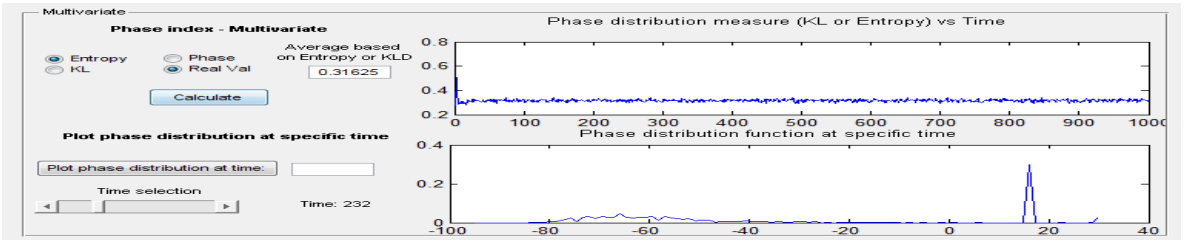




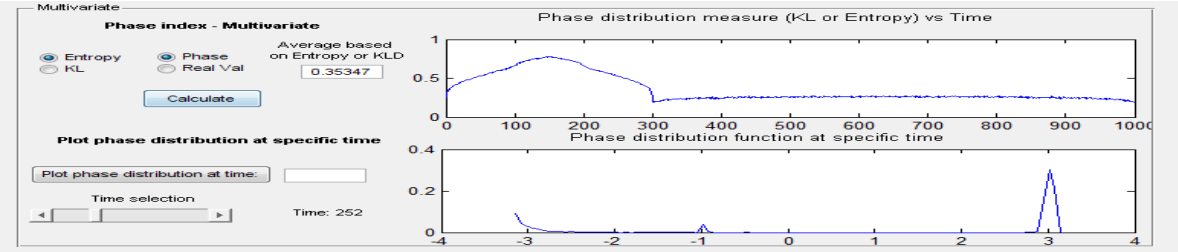
IS-cluster Signals

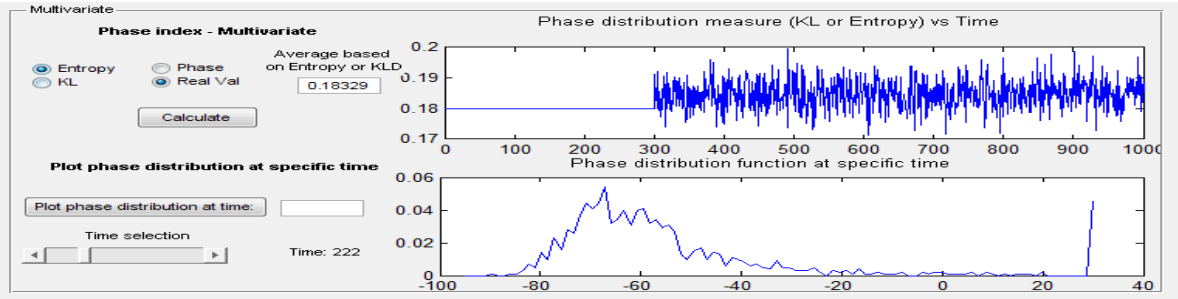
30% equal signals (rows)-cluster



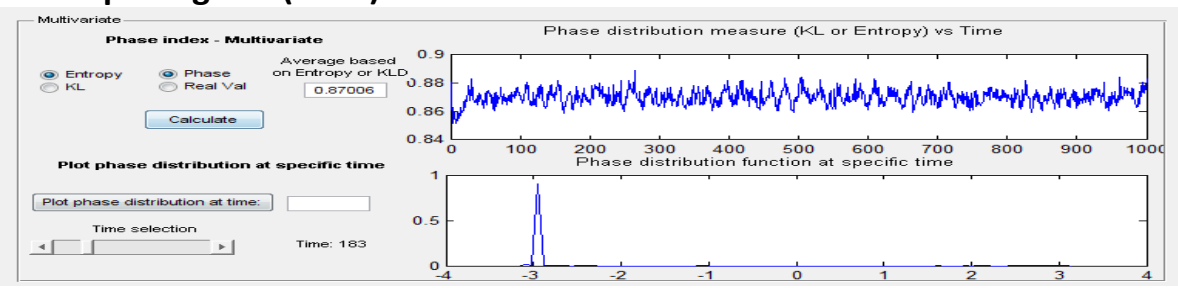


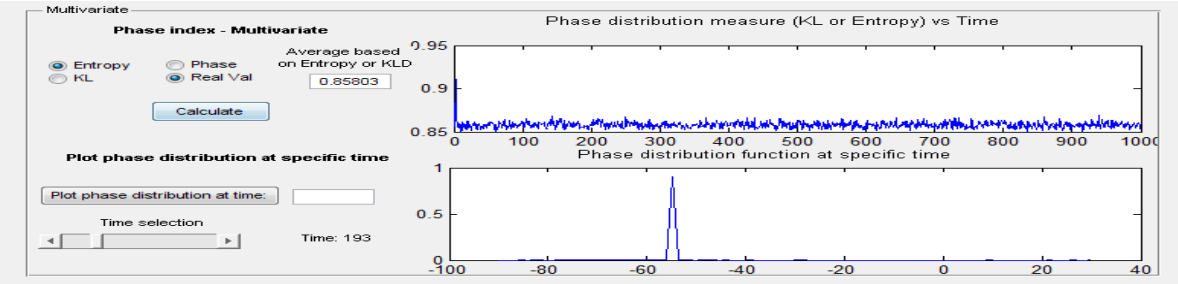
30% equal signals (rows) –cluster– transposed



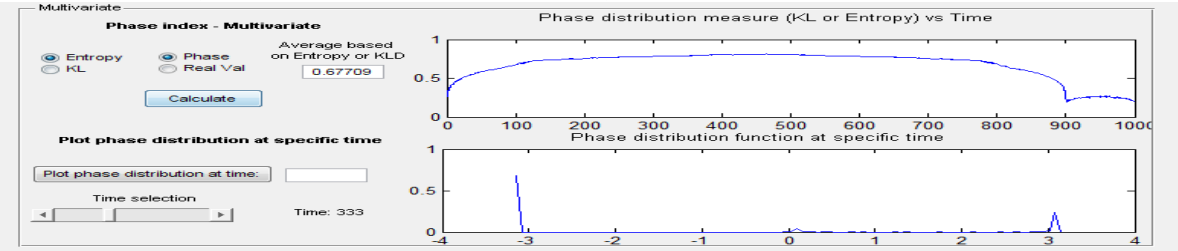


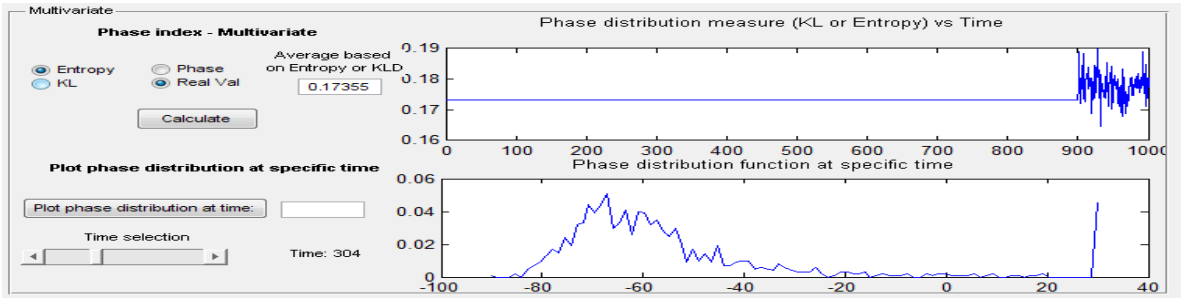
90% equal signals (rows) - cluster





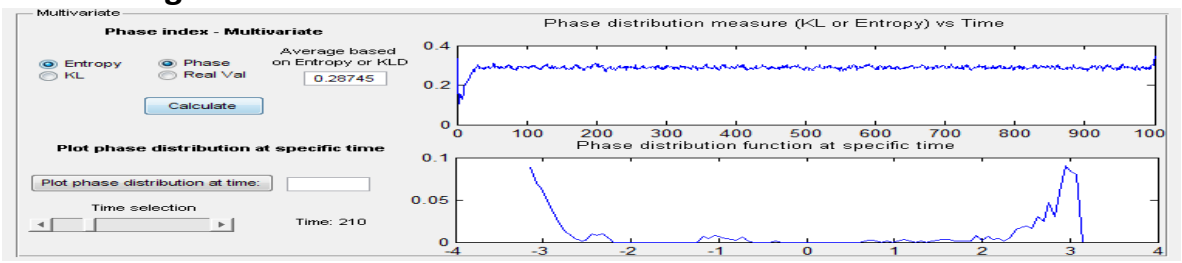
90% equal signals (rows) cluster - transposed

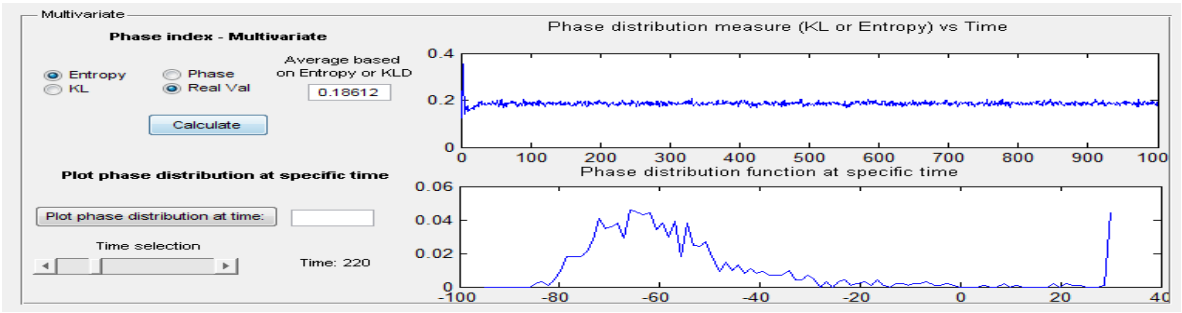




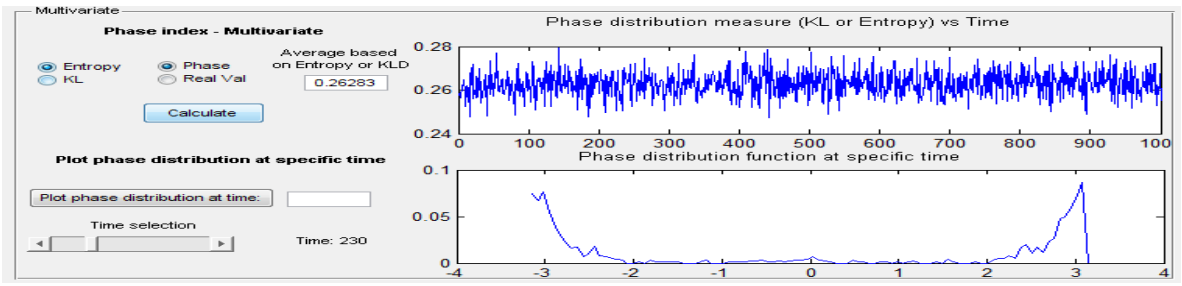
RS Signals

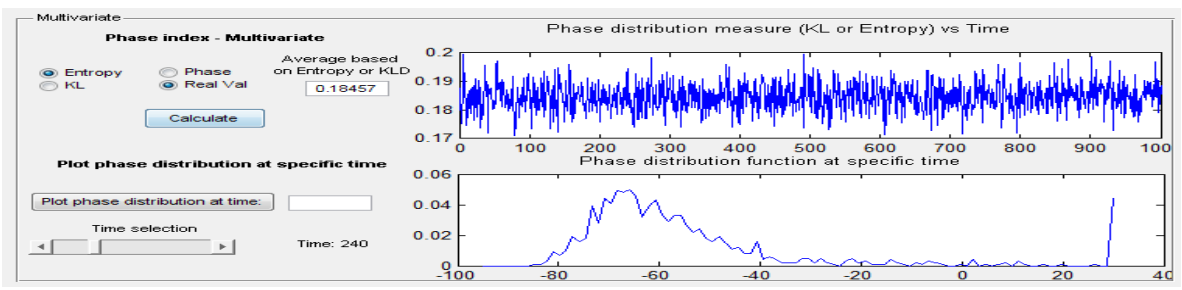
Random signals



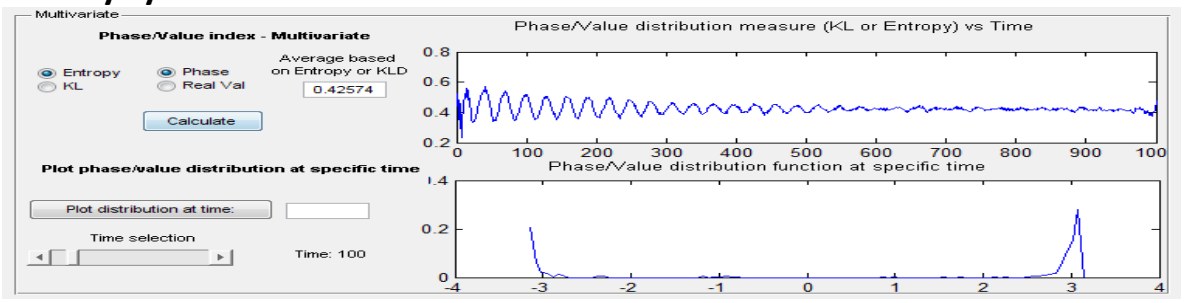


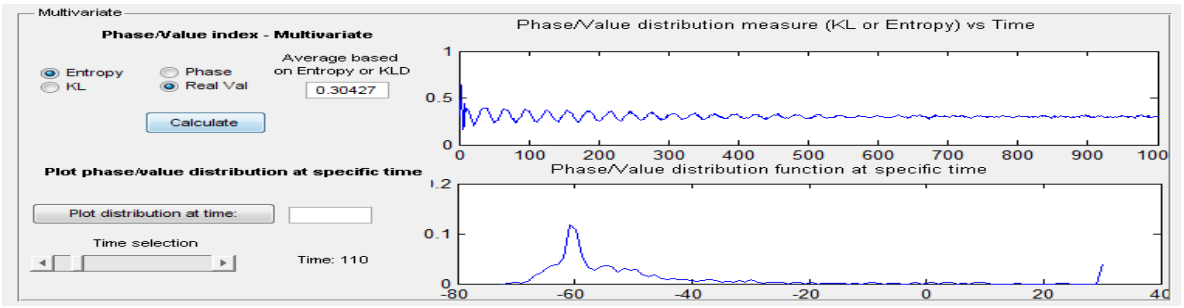
Random signals – transposed



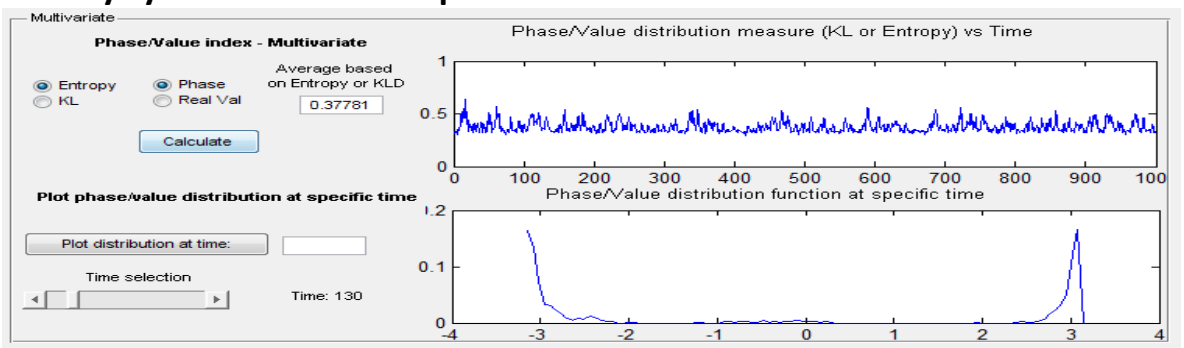


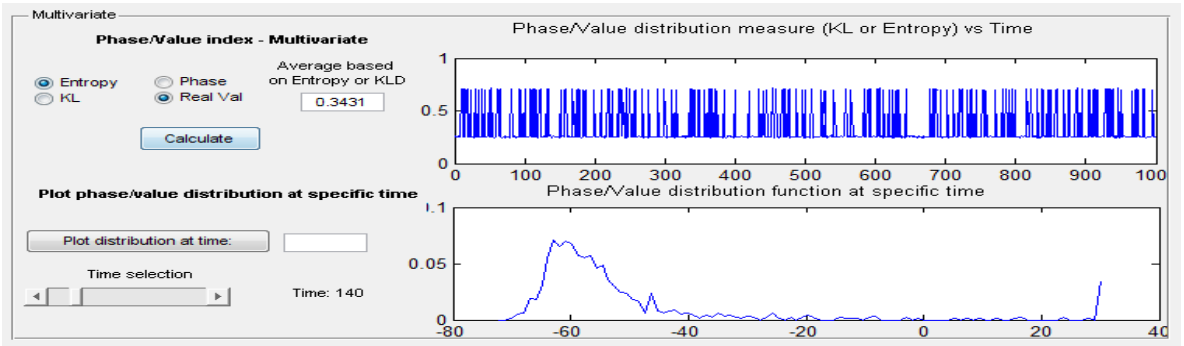
Weakly synchronized



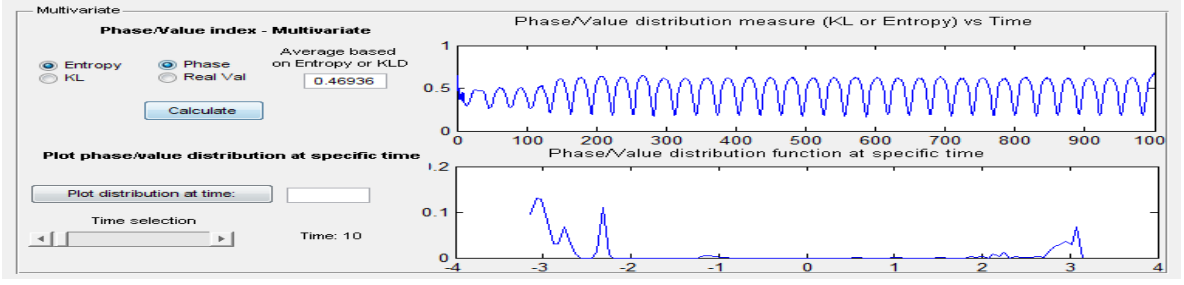


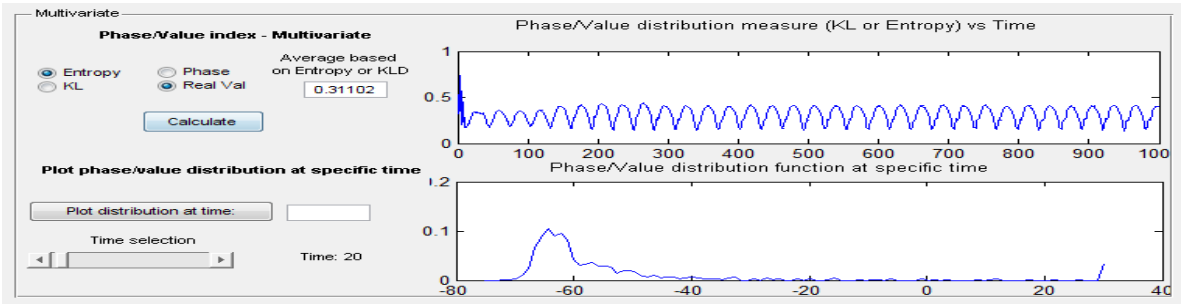
Weakly synchronized - transposed



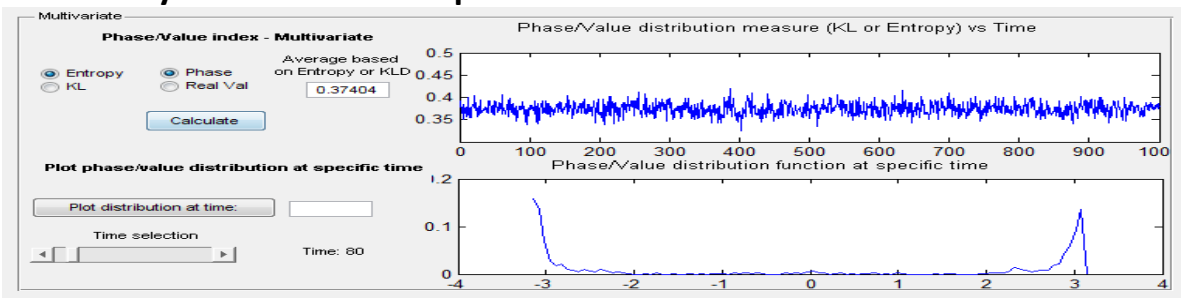


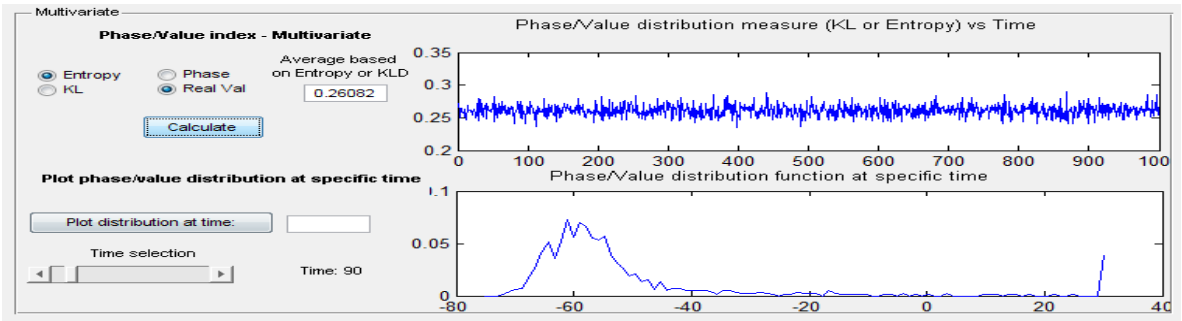
Medium synchronized



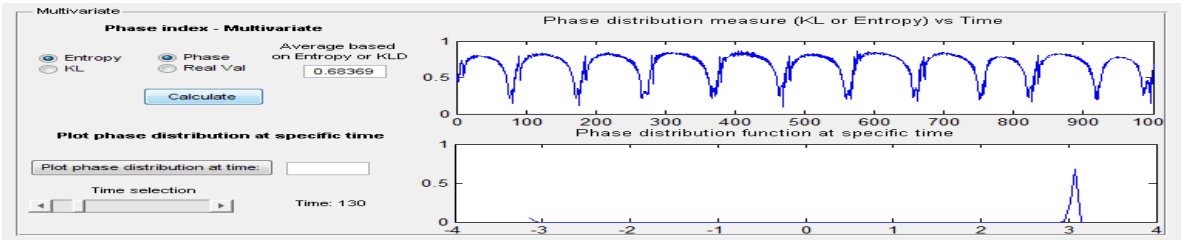


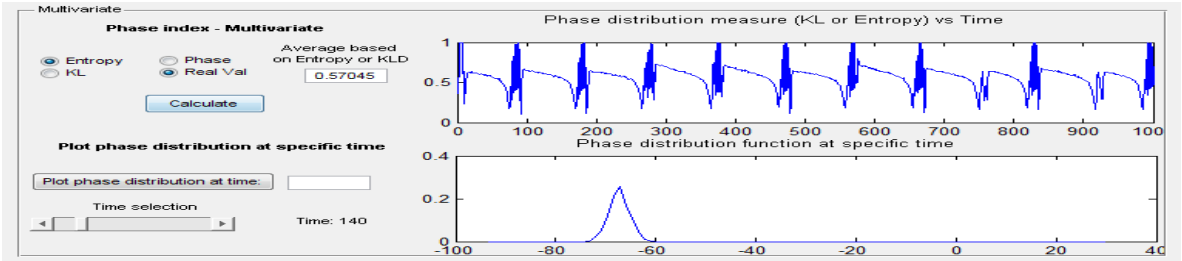
Medium synchronized - transposed



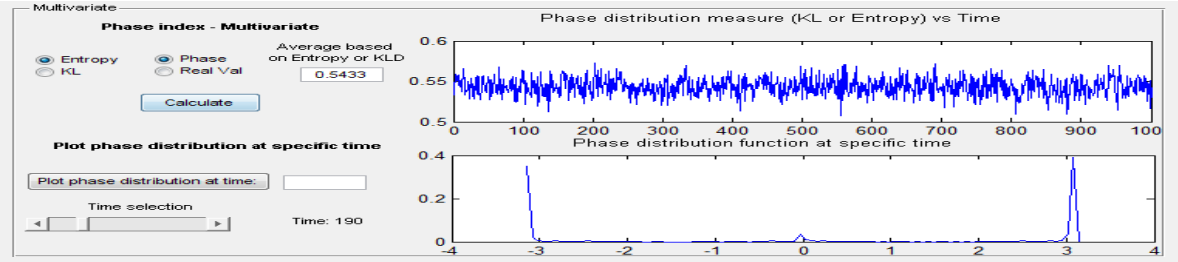


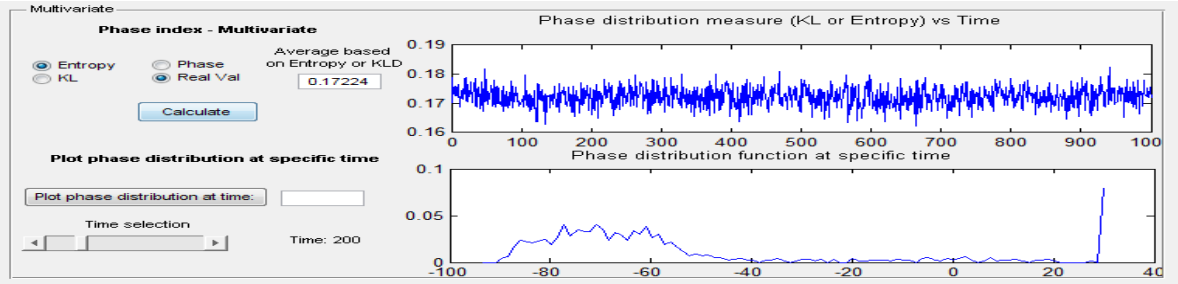
Strongly synchronized





Strongly synchronized –transposed





From the above figures it can be observed that the form of the distribution provide information on synchronization clusters and the existence of periodic and non-periodic synchronization.

9 Conclusions

This work presents the state of the art of detection techniques of synchronization for bivariate and multivariate (ensemble) data.

It also analyzes the relationship between synchronization and network structure of the underlying connectivity matrix.

A Matlab toolbox is provided to carry on experimental tests on synchronization for a neurons ensemble depending on network structure, neurons models and input signals

Finally the work provides two new methods to detect synchronization in a multivariate signals set. The methods are based on phase analysis and recurrence similarity matrices. The phase analysis method provides good results both for regular and irregular synchronous data. The method based on recurrence similarity matrices provides good results only for regular synchronous data and for the jaccard distance measure.

Future work will be focused on improving the phase analysis method, studying how to use the signature given by the phase and real value probability distribution functions obtained from the method, to represent the whole ensemble, and how to compare two ensembles using these signatures.

10 Contributions

As a result of this master thesis a paper will be presented to an international conference with the title: “Detection method for phase synchronization in a population of spiking neurons”

11 References

[1] Synchronization: A Universal Concept in Nonlinear Sciences (Cambridge Nonlinear Science Series). Arkady Pikovsky , Michael Rosenblum , Jürgen Kurths . 2001

[2] A unifying definition of synchronization for dynamical systems. Reggie Brown et al. arXiv:chao-dyn/9811013v2 27 Feb 1999

[3] A Comparative Study of Synchrony Measures for the Early Detection of Alzheimer’s Disease Based on EEG. Justin Dauwels, François Vialatte, Andrzej Cichocki. Elsevier, NeuroImage 49 (2010) 668–693

[4] http://en.wikipedia.org/wiki/Coherence_(statistics)

[5] http://en.wikipedia.org/wiki/Coherence_(signal_processing)

[6] A New Nonlinear Similarity Measure for Multichannel Biological Signals. Jian-Wu Xu , Neural Networks, 2007. IJCNN 2007.

[7] Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives. Jose C. Principe, Springer, 2010

[8] A REVIEW ON MULTIVARIATE MUTUAL INFORMATION. Sunil Srinivasa. University of Notre Dame. Tutorial. EE-80653, Information Theory, Department of Electrical Engineering, University of Notre Dame

[9] Assessing Instantaneous Synchrony of Nonlinear Nonstationary Oscillators in the Brain. Ananda S. Fine, David P. Nicholls, David J. Mogul. J Neurosci Methods. 2010 January 30; 186(1): 42.

[10] Fast algorithm for the metric-space analysis of simultaneous responses of multiple single neurons. Dmitriy Aronov. Journal of Neuroscience Methods 00 (2003) 1_/5. Elsevier.

[11] Frequency flows and the time-frequency dynamics of multivariate phase synchronization in brain signals. David Rudrauf, Abdel Douiri,Christopher Kovach,Jean-Philippe Lachaux,Diego Cosmelli,Mario Chavez,Claude Adam,Bernard Renault,Jacques Martinerie, Michel Le Van Quyen. NeuroImage 31 (2006) 209 – 227. Elsevier.

[12] Metric-space analysis of spike trains: theory, algorithms and application. Jonathan D Victory and Keith P Purpura. Network: Comput. Neural Syst. 8 (1997) 127–164.

[13] State Change Detection Using Multivariate Synchronization Measure from Physiological Signals. Kumiko Oshima, Cristian Carmeli and Martin Hasler. Ecole Polytechnique F´ed´erale de Lausanne (EPFL), School of Computer and Communication Sciences.

[14] Assessment of EEG synchronization based on state-space analysis. Cristian Carmeli, Maria G. Knyazeva, Giorgio M. Innocenti, Oscar De Feo. NeuroImage 25 (2005) 339 – 354

[15] The synchronization of chaotic systems. S. Boccaletti et al. Physics Reports 366 (2002) 1 – 101. Elsevier.
[16] Synchronization in complex networks. A. Arenas, A. Díaz-Guilera, J. Kurths, Y. Moreno, C. Zhou .Physics Reports Volume 469, Issue 3, December 2008, Pages 93–153
[17] COMPLEX NETWORKS: TOPOLOGY, DYNAMICS AND SYNCHRONIZATION. Xiao Fan Wang. International Journal of Bifurcation and Chaos, Vol. 12, No. 5 (2002) 885{916
[18] Recurrence plots for the analysis of complex systems. Marwan et al. Physics Reports 438 (2007) 237 – 329
[19] Estimation of interrelation between chaotic observables. A. Čenys, G. Lasiene, K. Pyragas. Physica D: Nonlinear Phenomena, 1991 ,Volume 52, Issue 2-3, p. 332-337.
[20] Nonlinear multivariate analysis of neurophysiological signals, Ernesto Pereda , Rodrigo Quian Quiroga, Joydeep Bhattacharya. Progress in Neurobiology 77 (2005) 1–37
[21] Nunez, P., Srinivasan, R., 2006. Electric Fields of the Brain: The Neurophysics of EEG. Oxford University Press.
[22] Quian Quiroga, R. et al, 2001, On the performance of diferent synchronization measures in real data: a case study on EEG signals, Chaotic Dynamics, September 2001, 1-32.
[23] Gunduz, A., Principe, J.C., 2009. Correntropy as a novel measure for nonlinearity tests. Signal Process 89 (1), 14–23.
[24] On Speeding Up Computation in Information Theoretic Learning. Sohan Seth and Jose C. Principe. Proceedings of International Joint Conference on Neural Networks, Atlanta, Georgia, USA, June 14-19, 2009
[25] Lachaux, J.P., Rodriguez, E., Martinerie, J., Varela, F.J., 1999. Measuring phase synchrony in brain signals. Hum. Brain Mapp. 8, 194–208.
[26] Analyzing Synchronization Phenomena from Bivariate Data by Means of the Hilbert Transform. Michael Rosenblum Jurgens Kurths. Nonlinear Analysis of physiological Data, Edited by H. Kantz, J. Kurths, and G. Mayer Kress (Springer, Berlin, 1998), pp. 91-99
[27] Detecting synchronization between the signals from multivariate and univariate biological data. Mikhail Prokhorov. Saratov Department of the Institute of Radio Engineering & Electronics of Russian Academy of Sciences.
[28] Corticonics: Neural Circuits of the Cerebral Cortex. Cambridge University Press; First edition (February 22, 1991). M Abeles. (Libro)
[29] http://www.scholarpedia.org/article/Granger_causality
[30] Sauer, T., Yorke, J., Casdagli, M., 2005. Embedology. J. Stat. Phys. 65, 579–616.
[31] Cover, T.M., Thomas, J.A., 1991. Elements of Information Theory. Wiley, New York.
[32] Measuring Information Transfer, Thomas Schreiber, VOLUME 85, NUMBER 2 PHYSICAL REVIEW LETTERS 10 JULY 2000
[33] Presentation on transfer entropy in music analysis: http://users.utu.fi/attenka/TEpresentation081128.pdf
[34] Dauwels, J., Vialatte, F., Rutkowski, T., Cichocki, A., 2008. Measuring neural synchrony by message passing. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (Eds.), Advances in Neural Information Processing Systems, vol. 20. MIT Press, Cambridge, MA, pp. 361–368.
[35] Lachaux, J. et al, 2000, Studying single-trials of phase synchronous activity in the brain, International Journal of Bifurcation and Chaos, Vol. 10, No. 10, 2429-2439.

[36] Tass, P. et al, 1998, Detection of n:m phase locking from noisy data, Physical review letters, Vol. 81, No. 15, 3291-3294.
[37] MIXED STATE ANALYSIS OF MULTIVARIATE TIME SERIES. MARTIN WIESENFELDT, ULRICH PARLITZ and WERNER LAUTERBORN. International Journal of Bifurcation and Chaos, Vol. 11, No. 8 (2001) 2217-2226
[38] Symbolic Synchronization and the Detection of Global Properties of Coupled Dynamics from Local Information, Sarika Jalan, Jurgen Jost, and Fatihcan M. Atay, arXiv:nlin/0604016v2 [nlin.CD] 29 Nov 2006
[39] Multivariate singular spectrum analysis and the road to phase synchronization Andreas Groth and Michael Ghil, June 27, 2011
[40] Advanced spectral methods for climatic time series, M. Ghil et al, REVIEWS OF GEOPHYSICS, VOL. 40, 1003, 41 PP., 2002
[41] EIGENVALUE DECOMPOSITION AS A GENERALIZED SYNCHRONIZATION CLUSTER ANALYSIS, Allefeld et al, arXiv:0706.3375v3 [physics.data-an] 3 Sep 2008
[42] Detection and characterization of changes of the correlation structure in multivariate time series, Phys Rev E Stat Nonlin Soft Matter Phys. 2005 Apr; Müller M, Baier G, Galka A, Stephani U, Muhle H.
[43] Pattern identification in dynamical systems via symbolic time series analysis. Pattern Recognition 40 (2007) 2897 – 2907. Venkatesh Rajagopalan, Asok Ray, Rohan Samsi, Jeffrey Mayer
[44] Symbolic dynamics of event-related brain potentials, Peter beim Graben, J. Douglas Saddy, and Matthias Schlesewsky, PHYSICAL REVIEW E VOLUME 62, NUMBER 4 OCTOBER 2000
[45] http://www.scholarpedia.org/article/Measures_of_spike_train_synchrony
[46] On the theory and computation of evolutionary distances, Sellers P H, 1974, SIAM J. Appl. Math. 26 787–93
[47] Generalized redundancies for time series analysis, Dean Prichard , James Theiler, PhysicaD84 (1995) 476-493
[48] Time-resolved and time-scale adaptive measures of spike train synchrony, Thomas Kreuz, Daniel Chicharro, Martin Greschner, Ralph G. Andrzejak, Journal of Neuroscience Methods 195 (2011) 92–106
[49] Grun, S., Diesmann, M., Aertsen, A.,. Unitary events in multiple single-neuron spiking activity: I. Detection and significance. Neural Comput. 14, 43–80.
[50] Neural Coding of Spatial Phase in V1 of the Macaque Monkey, Dmitriy Aronov, Daniel S. Reich, Ferenc Mechler, and Jonathan D. Victor, J Neurophysiol 89: 3304–3327, 2003. First published January 29, 2003; 10.1152/jn.00826.2002.
[51] A new multi-neuron spike-train metric, Conor Houghton and Kamal Sen, Neural Comput. 2008 June ; 20(6): 1495–1511. doi:10.1162/neco.2007.10-06-350.
[52] Surrogate data in detecting nonlinearity and phase synchronization, Milan Palus at al., IEEE Engineering in Medicine and Biology Vol. 17 No. 6 (1998) 40-45
[53] Neuronal Assembly Detection and Cell Membership Specification by Principal Component Analysis, Neuronal Assembly Detection and Cell Membership Specification by Principal Component Analysis, Vítor Lopes-dos-Santos, Sergio Conde-Ocazionez, Miguel A. L. Nicolelis, Sidarta T. Ribeiro, and Adriano B. L. , 10.1371/journal.pone.0020996
[54] Eigenvalue Spectra of Random Matrices for Neural Networks, Kanaka Rajan and L. F. Abbott, PRL 97, 188104 (2006) P H Y S I C A L R E V I E W L E T T E R S.

[55] Neural Network Dynamics, Tim P. Vogels, Kanaka Rajan, and L.F. Abbott, Annu. Rev. Neurosci. 2005. 28:357–76

[56] Speed of synchronization in complex networks of neural oscillators: Analytic results based on Random Matrix Theory, Marc Timme, Theo Geisel, and Fred Wolf, Chaos 16, 015108 (2006); doi: 10.1063/1.2150775

[57] http://www.nld.ds.mpg.de/research/projects/theory-of-precise-timing-in-spiking-neural

[58] Network synchronization: Spectral versus statistical properties, Fatihcan M. Atay et al, arXiv:0706.3069v1 [cond-mat.dis-nn] 20 Jun 2007

[59] Synchronization in Complex Networks of Nonlinear Dynamical Systems, Chai Wah WU, 2007 by World Scientific Publishing Co. Pte. Ltd

[60] Spells of Low-Frequency Oscillations and Weather Regimes in the Northern Hemisphere. Plaut, Guy; Vautard, Robert Journal of Atmospheric Sciences, vol. 51, Issue 2, pp.210-236

[61] Multivariate singular spectrum analysis and the road to phase synchronization. Andreas Groth, Michael Ghil, EGU General Assembly 2010, Presentation.

[62] A phase-synchronization and random-matrix based approach to multichannel time-series analysis with application to epilepsy. Ivan Osorio and Ying-Cheng Lai. Chaos 21, 033108 (2011); doi: 10.1063/1.3615642

[63] Topological Speed Limits to Network Synchronization, Marc Timme, Fred Wolf, and Theo Geisel, Phys. Rev. Lett. 92:074101

[64] Diesmann M, Gewaltig M-O, Aertsen A. (1999) Stable propagation of synchronous spiking in cortical neural networks Nature 402, 529-533

[65] Detecting synfire chains in parallel spike data, George L. Gerstein, Elizabeth R. Williams, Markus Diesmannc, Sonja Grün, Chris Trengove, Journal of Neuroscience Methods 206 (2012) 54– 64

[66] Detecting Synfire Chain Activity Using Massively Parallel Spike Train Recording, Sven Schrader,1 Sonja Grun, Markus Diesmann, and George L. Gerstein, J Neurophysiol 100: 2165–2176, 2008.

[67] Detecting spatiotemporal firing patterns among simultaneously recorded single neurons J Neurophysiol 60:909-924, 1988. M. Abeles and G. L. Gerstein

[68] Synchronization in networks of nonlinear dynamical systems coupled via a directed graph, Chai Wah Wu, Nonlinearity 18 (2005) 1057–1064

[69] Matrix Analysis and Applied Linear Algebra, Carl D. Meyer, SIAM (February 15, 2001)

[70] A PCA-based Similarity Measure for Multivariate Time Series, Kiyoung Yang and Cyrus Shahabi, Proceedings of the 2nd ACM international workshop on Multimedia databases, 2004

[71] Real-time pattern isolation and recognition over immersive sensor data streams. C. Shahabi and D. Yan. In the 9th International Conference On Multi-Media Modeling, 2003.

[72] Neural Control Engineering: The Emerging Intersection between Control Theory and Neuroscience, Steven J. Schiff, MIT Press; Computational Neuroscience, 2011

[73] Correlation Based Dynamic Time Warping, Zoltán Bankó, János Abonyi, 8th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics

[74] Lower-Bounding of Dynamic Time Warping Distances for Multivariate Time Series, Toni M. Rath and R. Manmatha_, Multi-Media Indexing and Retrieval Group

[75] Recognition Of Multivariate Temporal Musical Gestures Using N-Dimensional Dynamic Time Warping, Nicholas Gillian at al.

[76] The Synchronized Dynamics of Complex Systems, Volume 6 (Monograph Series on Nonlinear Science and Complexity) , Stefano Boccaletti, Albert C.J. Luo , George Zaslavsky

[77] J.-P. Eckmann, S.O. Kamphorst, D. Ruelle, Recurrence plots of dynamical systems, Europhys. Lett. 5 (1987) 973–977.

[78] Reducing Neuronal Networks to Discrete Dynamics, David Terman, Sungwoo Ahn, Xueying Wang, Winfried Just, Physica D. 2008 March ; 237(3): 324–338.

[79] Network synchronization landscape reveals compensatory structures, quantization, and the positive effect of negative interactions, Takashi Nishikawa and Adilson E. Motter , Published in PNAS 107, 10342-10347 (2010)

[80] Synchronization in large directed networks of coupled phase oscillators ,Juan G. Restrepo et al, CHAOS 16, 015107 2005

[81] Nykamp DQ, “The stability of the asynchronous state as function of largest eigenvalue.” From Math Insight. http://mathinsight.org/stability_asynchronous_state_largest_eigenvalue

[82] “The influence of network structure on correlations.” From Math Insight. http://mathinsight.org/influence_network_structure_correlations

[83] Pernice V, Staude B, Cardanobile S, and Rotter S. How Structure Determines Correlations in Neuronal Networks. *PLoS Comput Biol*, 7:e1002059, 2011.

[84] SYNCHRONIZATION AND GRAPH TOPOLOGY, IGOR BELYKH et al, International Journal of Bifurcation and Chaos, Vol. 15, No. 11 (2005) 3423–3433

[85] On the Synchronization of Networks with Prescribed Degree Distributions. Fatihcan M. Atay et al. arXiv:nlin/0407024 [nlin.AO]

[86] Mathematical Foundations of Neuroscience, G. Bard Ermentrout _ David H. Terman, Springer 2010

[87] Simple Model of Spiking Neurons, Eugene M. Izhikevich, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 14, NO. 6, NOVEMBER 2003

[88] Which Model to Use for Cortical Spiking Neurons?, Eugene M. Izhikevich, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 15, NO. 5, SEPTEMBER 2004.

[89] DYNAMICAL PROCESSES ON COMPLEX NETWORKS, A. Barrat, M. Barthelemy and A. Vespignani , Cambridge University Press, 2008

[90] Wavelet entropy in event-related potentials: a new method shows frequency tuning of EEG-oscillations. Quian Quiroga R, Rosso O, Schürmann M and Basar E. Biological Cybernetics, 84: 291-299; 2001.

[91] Image Registration and Segmentation by Maximizing the Jensen-R´enyi Divergence A. Ben Hamza and Hamid Krim (2003) . LECTURE NOTES IN COMPUTER SCIENCE.

[92] Speech pitch determination based on Hilbert-Huang transform, Hai Huang , Jiaqiang Pan, Signal Processing 86 (2006) 792–803.

[93] The Normalized Compression Distance as a Distance Measure in Entity Identification, Sebastian Klenk et al, ICDM '09 Proceedings of the 9th Industrial Conference on Advances in Data Mining. Applications and Theoretical Aspects

[94] ANALYSIS AND STUDY ON TEXT REPRESENTATION TO IMPROVE THE ACCURACY OF THE NORMALIZED COMPRESSION DISTANCE, Thesis UAM, Ana Granados Fontecha, 2011

[95] R. Cilibrasi and P. Vitanyi. Clustering by Compression. IEEE Transactions on Information Theory, 51(4):1523–1545, 2005

[96] A. Kolmogorov. Three Approaches to the Quantitative Definition of Information. Problems Information Transmission, 1(1):1–7, 1965.

[97] H. Kozima. Text Segmentation Based on Similarity between Words. In Proceedings of the 31st annual meeting on Association for Computational Linguistics, ACL ’93, pages 286–288, 1993.

[98] M. Li, X. Chen, X. Li, B. Ma, and P. Vitanyi. The Similarity Metric.IEEE Transactions on Information Theory, 50(12):3250–3264, 2004.

[99] A compositionality machine realized by a hierarchic architecture of synfire chains Sven Schrader, Markus Diesmann and Abigail Morrison, January 2011, doi: 10.3389/fncom.2010.00154

[100] Phase synchronization: from theory to data analysis, Michael Rosenblum , Arkady Pikovsky, Jurgen Kurths, Carsten Schafer, Peter A. Tass . Handbook of Biological Physics, Volume 4, 2001, Pages 279–321, Neuro-Informatics and Neural Modelling

[101] http://en.wikipedia.org/wiki/Binding_problem

[102] Borisyuk R., Borisyuk G. (1997) Information coding on the basis of synchronization of neural activity. BioSystems, 40, 3-10

[103] A COMPLEX NETWORK PERSPECTIVE OF WORLD STOCK MARKETS: SYNCHRONIZATION AND VOLATILITY XIAO FAN LIU and CHI K. TSE , International Journal of Bifurcation and Chaos, Vol. 22, No. 6 (2012) 1250142

[104] New Developments in Music Information Retrieval, Meinard Muller , Proceedings of the 42nd AES Conference, 2011.

[105] Multi-view Synchronization of Human Actions and Dynamic Scenes, Emilie Dexter et al. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.150.4135.