

The construction of smooth models using irregular embeddings determined by a Gamma test analysis

Alban P.M. Tsui, Antonia J. Jones
Department of Computer Science
University of Wales, Cardiff
PO Box 916
Cardiff CF2 3XF
Wales, UK

Ana Guedes de Oliveira
Centre for Nonlinear Dynamics & its Applications
University College London
London WC1E 6BT
England, UK

Abstract

One of the key problems in forming a smooth model from input-output data is the determination of which input variables are relevant in predicting a given output. In this paper we show how the Gamma test can be used to select that combination of input variables which can best be employed to form a smooth model of an output. For time series prediction this amounts to the selection of an appropriate *irregular* embedding. We give some simple zero noise examples of time series analysis and illustrate how using these techniques a binary message encoded into a chaotic carrier can be retrieved without knowledge of the dynamics used to generate the carrier.

Provided the underlying dynamics are such as to produce a smooth embedding model with bounded partial derivatives, the sampling distribution is dense in input space, and any associated distribution of measurement error has the first few moments bounded, so that the typical pre-requisite conditions of the Gamma test are satisfied, we conclude that the Gamma test is an effective tool in the determination of irregular time series embeddings.

These techniques can also be useful in practical applications which involve filtering seismic data to detect anomalous events.

Keywords: Chaotic dynamics, modelling, predication, irregular embeddings, Gamma test, seismic anomalies.

1 Introduction

The Gamma test was first briefly reported in [1, 2] and later used and discussed in [3, 4]. Given a set of input-output data composed of real variables the Gamma test allows us to efficiently calculate directly from the data an estimate, the *Gamma* statistic, of the best mean-squared error on an output that can be attained by a smooth model. As the number of data samples becomes large the Gamma statistic converges with probability one to the noise variance of the data modulo any smooth model (we state this more precisely in the next section).

This information is of itself extremely useful. For example, when building a neural network it tells us, without the necessity of a separate check using a validation set, when to cease training. However, the utility of the Gamma test goes well beyond merely giving an estimate for the mean squared error. By examining the Gamma statistic for different selections of input variables we choose that selection which minimises the expected mean squared error. It is this application that the present paper is designed to illustrate using time series data.

In order to model time series data, we need to construct the model by choosing the past values, up to some number m (often called the embedding dimension) to form the inputs of the model. The output is then the current value of the time series. Thus an *embedding* of a time series is a selection of past values which are used to predict the current value via a model constructed from the data. A *regular* embedding takes all past values up to m . The formal basis for this approach for dynamical system modelling using regular embeddings was first studied by Takens [5]. An *irregular* embedding chooses some subset of the m past values, and there are $2^m - 1$ possible irregular embeddings once m is chosen. It was suggested in Judd [6] that irregular embeddings may often provide a better model. In fact irregular embeddings were already used to advantage in modelling sunspot data in [2]. Thus we have known for some while that the choice of a suitable irregular embedding can be critical for model based time series prediction.

In this paper we illustrate the utility of the Gamma test in selecting irregular embeddings for time series data. The idea can easily be extended to the situation where the inputs are selected from multiple time series and the goal is to form a good predictive model for some (possibly distinct) time series.

The Gamma test is non-parametric, the result will apply regardless of the particular technique used to construct a model. We consider here two modelling techniques, local linear regression and feedforward neural networks trained using the BFGS algorithm [7].

Local linear regression proceeds by locating a specified number of near neighbours of the query point and then constructing a locally linear model to predict the output associated with the query. In the context of the Gamma test this is very easy to do because we have already constructed a kd-tree describing the near-neighbour relationships. Although this will not produce a globally *smooth* model, under reasonable conditions the resulting function will converge to a smooth model as the number of data points increases.

Local linear regression models are fast to construct and quite fast to execute a query. Local linear regression models can also be easily updated as new training data becomes available, which is not the case with neural networks (where a prolonged extra period of training, or starting training all over again, may be required to modify the model on the basis of new data). Usually local linear regression is extremely accurate in parts of the input space where the training data density is high. However, local

linear regression will not generalise well to parts of the input space for which training data is sparse.

There has been considerable recent interest in using chaotic synchronization to secure communications (see for example [8, 9, 10, 11, 12] and references therein). The principle used is that if we have two identical chaotic systems, one in the transmitter, and one in the receiver, we can use one of the variables of the transmitter system as a carrier for the message. Then at the receiver end, using synchronization, we can decode the message.

Our final examples illustrate how a digital message embedded in a chaotic carrier can be detected and extracted by comparing the received signal with the model predicted value, where the model is constructed only using the transmitted data.

The method proposed here is similar to the idea in [13] that local predictive models can be used to detect unexpected variations in the signal, except that here rather than use only past values (and reference a prediction model) we buffer a few past and ‘future’ values before using a model to estimate the carrier without an encoded value. The novel aspect of our modelling technique lies in the use of the Gamma test which enables us to quickly determine a suitable *irregular* embedding to construct a predictor model and also to determine how many sample points are required to give a given Mean Squared Error.

The Gamma test software suite used throughout these experiments is a general purpose data analysis tool specifically developed for modelling and predicting non-linear systems.¹

2 Constructing a model

2.1 The Gamma test

The modelling of a time series signal is just a special case of the general nonlinear modelling problem in which we are given a set of observations of the form $(\mathbf{x}(i), y(i))$ ($1 \leq i \leq M$), where $\mathbf{x}(i) \in \mathbb{R}^m$, and without loss of generality, $y(i) \in \mathbb{R}$. If $y(i)$ is a vector we treat each component separately and at very little extra computational cost obtain an individual estimate of the Gamma Test result for each output. We seek to construct a function f such that $\mathbf{y} = f(\mathbf{x}) + E$ where E is minimized in some sense.

We focus on the case where samples are generated by a continuous function $f : C \subset \mathbb{R}^m \rightarrow \mathbb{R}$ and

$$\mathbf{y} = f(x_1, \dots, x_m) + r \tag{1}$$

where C is closed bounded and r represents an indeterminable part, which may be due to real noise or might be due to lack of functional determination in the posited input/output relationship i.e. an element of ‘one \rightarrow many-ness’ present in the data. Given samples (\mathbf{x}, y) generated by (1), in which the underlying

¹*winGamma*TM licensed by the University of Wales, Cardiff.

continuous function f is unknown, we cannot hope to estimate the mean μ of r , since a non-zero mean will create a bias which could just as easily be incorporated into the data model by considering f to be replaced by $f + \mu$. We therefore assume in what follows that $\mu = 0$.

A technique which allows one to estimate $Var[r]$, on the hypothesis of an underlying continuous or smooth model f , is of considerable practical utility in applications such as control or time series modelling. For example, the implication of being able to estimate $Var[r]$ in neural network modelling is that then one does not need to train the network in order to predict the best possible performance with reasonable accuracy.

The Gamma test works by exploiting the continuity of the unknown function f . Consider what happens when two data points $\mathbf{x}(i)$, $\mathbf{x}(j)$ are close together. Then since f is smooth we should expect that $f(\mathbf{x}(i))$ and $f(\mathbf{x}(j))$ are also close together. If they are not then it can only be because of the addition of noise.

The Gamma test (or near neighbour technique) is based on the statistic

$$\gamma = \frac{1}{2M} \sum_{i=1}^M (y'(i) - y(i))^2 \quad (2)$$

where $y'(i)$ is the y value which corresponds to the first near-neighbour of $\mathbf{x}(i)$.

Given data samples $(\mathbf{x}(i), y(i))$, where $\mathbf{x}(i) = (x_1(i), \dots, x_m(i))$, $1 \leq i \leq M$, let $N[i, p]$ be the list of (equidistant) p th nearest neighbours to $\mathbf{x}(i)$. We write

$$\delta(p) = \frac{1}{M} \sum_{i=1}^M \frac{1}{L(N[i, p])} \sum_{j \in N[i, p]} |\mathbf{x}(j) - \mathbf{x}(i)|^2 = \frac{1}{M} \sum_{i=1}^M |\mathbf{x}(N[i, p]) - \mathbf{x}(i)|^2 \quad (3)$$

where $L(N[i, p])$ is the length of the list $N[i, p]$. Thus $\delta(p)$ is the mean square distance to the p th nearest neighbour. We also write

$$\gamma(p) = \frac{1}{2M} \sum_{i=1}^M \frac{1}{L(N[i, p])} \sum_{j \in N[i, p]} (y(j) - y(i))^2 \quad (4)$$

where the y observations are subject to statistical noise assumed independent of \mathbf{x} and having bounded variance.

If f has bounded partial derivatives over C , and the r distribution has mean zero one and the first four moments bounded, then one can show [14] that

$$\gamma \approx Var[r] + A(p)\delta + o(\delta) \quad \text{as } M \rightarrow \infty \quad (5)$$

where the convergence is in probability and $0 < A(p) \leq \frac{1}{2} \max |\nabla f|^2$. From which it follows that $\lim \gamma = Var[r]$ (in probability) as $\delta \rightarrow 0$, $M \rightarrow \infty$.

The Gamma test computes the mean-squared p th nearest neighbour distances $\delta(p)$ ($1 \leq p \leq p_{max}$, typically $p_{max} \approx 10$) and the corresponding $\gamma(p)$.² Next the $(\delta(p), \gamma(p))$ regression line is computed and the vertical intercept, $\bar{\Gamma}$ is returned as the gamma value. The regression line slope $A \approx \frac{1}{4} \mathbf{E}|\nabla f|^2$, where \mathbf{E} denotes expectation with respect to the sampling distribution, is also returned as this often provides a useful guide as to the complexity of the model f . Effectively $\bar{\Gamma}$ is the limit $\lim \gamma$ as $\delta \rightarrow 0$, which in theory is $Var[r]$.

Two main preconditions must be satisfied for the Gamma test result to be meaningful. These are

- The input data vector sampling must be such that as the number of data points M increases the distance between $\mathbf{x}(i)$ and its nearest neighbour $\mathbf{x}(j)$ ($1 \leq j \neq i \leq M$) must tend to zero. In particular there should be no isolated points in input space.
- The model assumes that the noise distribution r on each output y is independent of the associated input \mathbf{x} . In particular the noise variance is constant for a given output. If the noise is not homogeneous over the input space this is not necessarily fatal in a practical application: the Gamma test will return an estimate for the average noise variance and can still provide useful information when selecting relevant inputs.

A consequence of the first condition is

- If the input data contains categorical variables the results are liable to be unreliable. In its present form the Gamma test is only designed to deal with continuous inputs.

If the number of near neighbours p_{max} is bounded (typically $p_{max} \approx 10$) a single run of the Gamma test has a time complexity of $O(M \log M)$ and depending on the input dimension m , executes in a few seconds for data sets of a reasonable size (i.e. $M \leq 1000$).

One of the key questions we need to answer in a practical situation is how much data do we need to get an accurate estimate of gamma and to subsequently build a model which can predict with this Mean Squared Error.

To answer this question we run the Gamma test using increasing M and then plot a graph of $\bar{\Gamma}$ against M . Typically what will happen is that for small M the graph will have considerable variability but as M increases the graph will stabilize to an asymptote which reflects the true value of the noise variance. When the graph has stabilized there is nothing much to be gained by using a larger M .

For the kinds of smooth dynamic systems considered here, when there is no added noise and no binary signal encoded, a good predictive model should have $Var[r]$ close to zero.

²The original version in [1, 2] used smoothed versions of $\delta(p)$ and $\gamma(p)$ which rolled off the significance of more distant near neighbours. Later experience showed that this complication was largely unnecessary and the more recent version of the software used here is implemented as described above.

2.2 Model identification.

We define a *mask* over the input variables by selecting ‘1’ if the variable is to be included in the model and ‘0’ otherwise. E.g. For three possible inputs the mask 101 says we should include the first and last variable but not the second. For a time series the ordering in the mask is most recent sample last.

To find the best selection of inputs we need to find the best (irregular) embedding mask, i.e. the mask with least $|\bar{\Gamma}|$. Unfortunately there are $2^m - 1$ such masks (the mask with all zeros can obviously be excluded) and so to run the Gamma test on every possibility might be rather time consuming. In general the software is sufficiently fast to do a ‘full embedding’ on up to 20 inputs (it depends on the input dimension m and the number of data points M of course). Nevertheless, for up to 20 variables we can probably afford to search through all possible masks. For $m > 20$ we shall need to employ one of a number of possible heuristic search techniques provided by the software.

2.3 Model building.

Given the time series data we first use the Gamma test software to determine an appropriate embedding and the required number of points M in the training set to give a model which predicts with the mean squared error estimated by the Gamma value.

The first modelling technique used here is *local linear regression*. Since the Gamma test works by constructing a kd-tree [15, 16] from the input (training) data this data structure can be used to rapidly find the p_{max} nearest neighbours of any query point. We then build a simple linear regression model based on these neighbours and use this model to make a prediction for the query point. The choice of p_{max} in local linear regression is not necessarily related to the choice in the Gamma test. The optimal value of p_{max} for local linear regression depends on the embedding dimension and on the amount of noise present. If the data is noisy a larger value of p_{max} will be required. Of course we are assuming that there is enough data available that the local sampling density is sufficient to allow a good locally linear model to be constructed. If the system is such that the surface in embedding space has regions of very high local curvature then the method may otherwise fail.

Alternatively, we can construct feedforward neural network as our detection of a digital signal on a chaotic carrier illustrates.

2.4 Choosing the right embedding: an example using Hénon maps with different delays

In this section we will show that the Gamma test applied to example times series (where we know in advance the right embedding), will give us the correct answers.

To demonstrate this point we will first describe *Hénon-type* chaotic maps with different *delays*. The general form of such a map is

$$x_n = a + bx_{n-d_1}^2 + cx_{n-d_2} \quad (6)$$

where a, b, c are chosen constants to produce chaos, and $d_1, d_2 \geq 1$ are different delays. The maximum of d_1 and d_2 , $d = \text{Max}(d_1, d_2)$ will give us the dimension of the map, and the first d initial conditions are randomly initialized.

If we consider the time series produced by equation (6), and we are trying to find the best embedding (i.e. the embedding with $\bar{\Gamma}$ closest to zero) using, say N inputs (where $N \geq d$), it seems obvious that the delay d_1 and d_2 have to appear in the “right” embedding. For example, if $d_1 = 2$ and $d_2 = 4$, with $N = 6$, then, using previous notation, 001010 is an embedding using the 2^{nd} and 4^{th} input out of the 6, so d_1 and d_2 appear in this embedding.

Let us consider now several examples. First consider a map such as (6) with delays 5 and 6, i.e.,

$$x_n = a + bx_{n-5}^2 + cx_{n-6} \quad (7)$$

With $a = 1.4, b = 1$ and $c = 0.05$ the map (7) produces a chaotic attractor shown in Figure 1. The map is a 6-dimensional chaotic map, with Lyapunov exponents shown in Table 1. We can see that there are three positive Lyapunov exponents, so the map is actually hyperchaotic³.

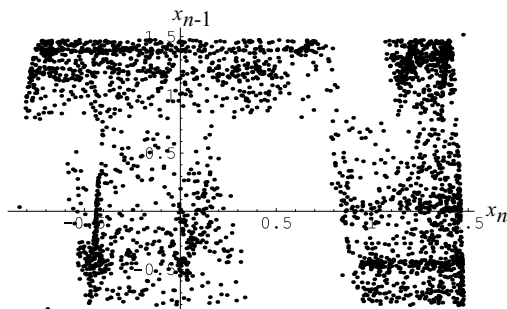


Figure 1: The chaotic attractor produced by the *Hénon-type* map with delays 5 and 6

Lyapunov Exponents
0.0385218
0.0136457
0.0000927729
-0.0454073
-0.142591
-0.807471

Table 1: Lyapunov Exponents of the delayed Hénon map (7).

Now consider the time series produced by (7). Before we do a **full embedding** (i.e. do a Gamma test with every possible mask, given a particular set of possible inputs), we apply an **increasing embedding**.

³At least one positive Lyapunov is required for chaos. If two or more Lyapunov exponents are positive the system is called hyperchaotic.

This algorithm starts with the mask obtained by taking only the rightmost input (in the case of a time series this is the most recent) and obtains a Gamma value for this mask. It progressively increases the number of bits set in the mask working from right to left performing a Gamma test for each new mask.

Figure 2 we can see that 5 seems to be a good value for the number of inputs (the underlying embedding dimension), since the gamma value seems to be stable. However, closer inspection of the actual numbers suggests that 8 is a better choice. After the increasing embedding, we run a M -test to see how much data we are likely to need. We can see the results in Figure 3, and for $M = 3000$ the Gamma value seems to have stabilized.

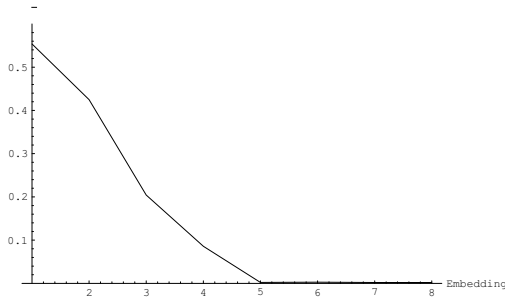


Figure 2: The results of an increasing embedding for the delayed Hénon map (7).

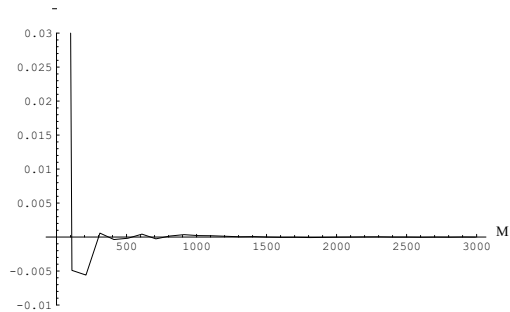


Figure 3: Results of the M-test for the time series produced by the delayed Hénon map (7).

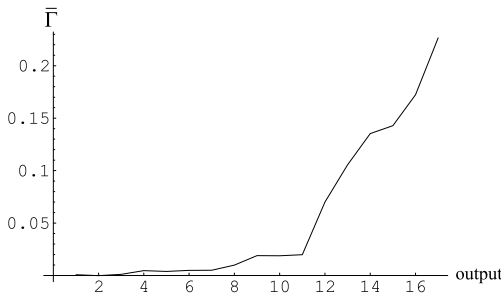


Figure 4: The Gamma value increases as we try to predict outputs further away.

Embedding	$\bar{\Gamma}$
00110001	-9.388×10^{-6}
00110000	1.6255×10^{-5}
01110001	1.6786×10^{-5}

Table 2: Top three results of a full embedding on the map (7) in the ascending order of $|\bar{\Gamma}|$.

We apply a full embedding, on 3000 data points of a time series produced by (7) using 8 inputs and one output, and order them according to the best gamma value (i.e., $\bar{\Gamma}$ closest to zero). The top three results can be seen in Table 2, and we can see that the expected delays appear in all of the top 3 results (5th and 6th counting from the right).

Also, an interesting test is to calculate the Gamma value for outputs increasingly further forward in time, that is, using a full mask we do a Gamma test for 20 outputs. The results can be seen in Figure 4, and since the map is chaotic it is not surprising that the Gamma value rapidly increases.

Let us consider now delays of 2 and 5 and the map given by the equation

$$x_n = 1.4 - x_{n-2}^2 - 0.25x_{n-5} \quad (8)$$

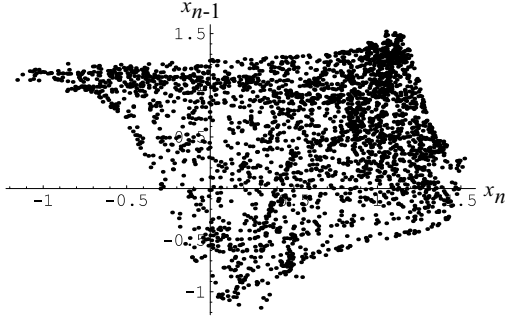


Figure 5: The chaotic attractor produced by a delayed Hénon map with delays 2 and 5.

Lyapunov Exponents
0.0901945
0.0150675
-0.0915081
-0.284305
-0.72071

Table 3: Lyapunov Exponents of the delayed Hénon map (8).

This map produces a chaotic attractor shown in Figure 5 and has 2 positive Lyapunov exponents shown in Table 3.

Again we applied an increasing embedding and an M -test, and the results can be seen in Figure 6 and Figure 7, respectively. In this example, looking at the results produced by these graphs, we choose 6 to be the number of inputs, and again choose 3000 data points.

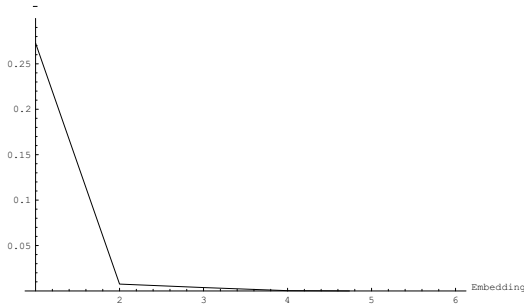


Figure 6: The results of an increasing embedding for the delayed Hénon map (8).



Figure 7: Results of the M -test for the time series produced by the delayed Hénon map (8).

Embedding	$\bar{\Gamma}$
010010	3.6648×10^{-6}
010111	9.594×10^{-6}
111010	-1.338×10^{-5}

Table 4: Top three results of a full embedding on the delayed Hénon map (8) in ascending order of $|\bar{\Gamma}|$.

So we now apply a full embedding, on 3000 data points of a time series produced by (8) using this time 6 inputs and one output, and again order them according to the best gamma value. The top 3 results can be seen in Table 4, and again we can see that the right delays appear in all top 3 results.

These examples illustrate the Gamma test indeed “picks” the right embeddings. The next section applies the Gamma test to decode messages masked in chaotic carriers.

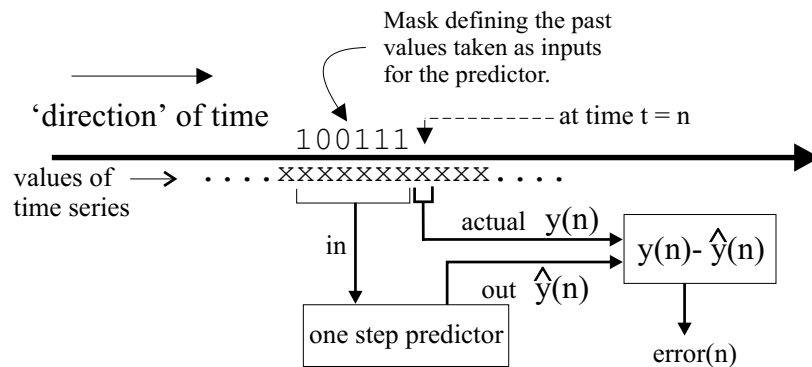


Figure 8: General detecting/decoding scheme for hidden binary messages in chaotic carrier.

3 Masking and modulation using chaos: different schemes

There are many different methods in the recent literature designed to secure communications based on synchronization of chaos. They vary in the way that synchronization is achieved and on how the hidden message in the chaotic carrier is “decoded”. But for someone ‘eavesdropping’ on the communication channel, what is important is not how the message is retrieved in the receiver or what synchronization method is used, but how to detect the message hidden in the chaotic carrier.

The basic idea is that the binary encodings will appear as noise on the smooth model. We then compare the one-step ahead predicted value \hat{y} , generated by the model, with the actually received value of the time series y , to detect and decode any unusually large errors which correspond to the hidden signal in the carrier. Thus the smaller the amplitude of the encoded binary signal the more accurate the predictive model will need to be. The general detecting/decoding scheme is summarized in Figure 8.

There are two main ways to encode a message: a message signal can be sent by modulating a system bifurcation parameter (*modulation*), or it can be added directly to the chaotic carrier (*masking*). We can see this in Figure 9 and Figure 10 respectively. In these figures $\dot{\mathbf{x}} = f(\mathbf{x}, p_0)$ is a set of n differential equations that define a chaotic system, where $\mathbf{x} = (x_1, \dots, x_n)$, and p_0 is one of the system’s parameters.

We will now describe some of the chaotic communication schemes that use masking to hide messages. We will focus on the ways to conceal the message and not on the synchronization methods used to retrieve it. We will also concentrate only on encoding binary messages as a square wave function. Decoding messages encoded by modulating a system parameter is undoubtedly more difficult. The examples below are designed to illustrate the utility of the Gamma test rather than as ‘state-of-the-art’ decoding *per se*.

3.1 Message masked by adding it to a chaotic carrier

In [17] Oppenheim *et. al* proposed a scheme to mask messages using a chaotic carrier and used the synchronization proposed by Pecora and Carroll [18] to recover the message at the receiver.

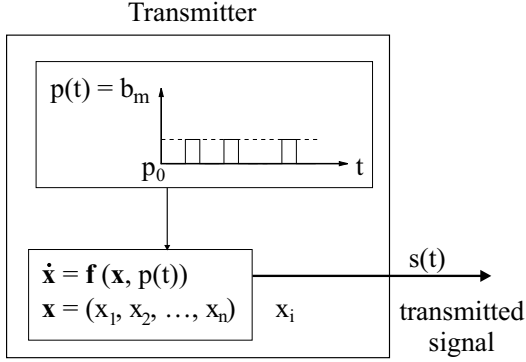


Figure 9: Hiding a binary message by modulating one of the parameters.

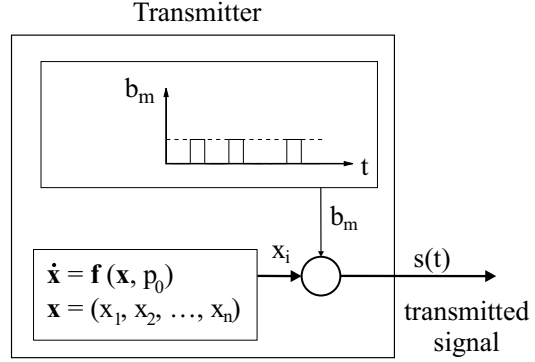


Figure 10: Hiding a binary message by masking it in a chaotic signal.

In [11] Parlitz *et al.* used the Chua circuit to mask a binary message. The Chua system is described by the equations

$$\begin{cases} \dot{x} = \alpha(y - x - f(x)) \\ \dot{y} = x - y + z \\ \dot{z} = -\beta y \end{cases} \quad (9)$$

where $f(x)$ is defined by

$$f(x) = bx + \frac{1}{2}(a - b)(|x + 1| - |x - 1|) \quad (10)$$

and α , β , a , b are constants (we have used here $\alpha = 10$, $\beta = 14.87$, $a = -1.27$ and $b = -0.68$). They used also Pecora and Carroll synchronization but with a more elaborate method to retrieve the message.

Many systems can be used to mask messages as long as synchronization can be achieved to recover them. The use of hyperchaotic systems could make these systems more secure, and although several methods have been proposed to synchronize hyperchaotic systems (for example [19, 20]), it has not so far proved possible to decode messages using these systems. Even so, to check if the security of these communication methods can indeed be improved by the use of hyperchaotic systems, we have masked a binary message in the y variable of the hyperchaotic Rössler system which is defined by the equations:

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + 0.25y + w \\ \dot{z} = b + xz \\ \dot{w} = (-0.5 + p)z + 0.05w \end{cases} \quad (11)$$

where $b = 3$ and $p = 0$ are constants.

The Chua and Rössler systems will be examined in sections 4.1 and 4.2 respectively.

4 Implementation and results

In this section we will use the method described in Section 3 to try to detect messages masked in chaotic systems as described in the previous section. The idea is, when the chaotic carrier is sent using one of

these schemes, a third party captures the signal and attempts to retrieve the binary message hidden in the chaotic carrier.

The chaotic carrier captured was sampled in time steps of 0.01. For each of the experiments we found the best embedding and we used the Gamma-test to determine an appropriate size M for the data set used to train the model to construct simple one-step predictor based on a suitable embedding for the inputs.

4.1 Detecting a message in the Chua system

We will start by trying to recover the binary message 10101011110011110010 shown in Figure 11 masked in the y variable of the Chua system described above (Equation (9)). In Figure 11 a positive square corresponds to a ‘1’ and a negative to a ‘0’. In this experiment the time interval for the square wave was $\Delta t = 100$ sample time steps⁴. The amplitude of the binary message is much smaller than the chaotic carrier and it cannot be spotted when it is masked, as we can see in Figure 12.

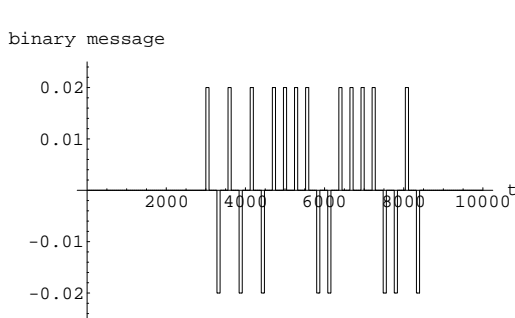


Figure 11: Binary message masked in the chaotic carrier. A positive square corresponds to a ‘1’ and a negative to a ‘0’.

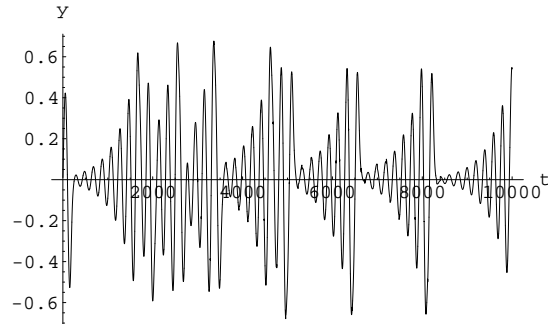


Figure 12: The y variable of the Chua system plus the binary message.

In this experiment, the carrier data was first normalized (with mean 0 and standard deviation of 0.5) before the Gamma test analysis and the model construction. We first use the Gamma test to determine the best embedding and then to determine an appropriate size M for the training data set for the model, we run the Gamma test using increasing M (M -test). In Figure 13 we can see that the Gamma value stabilizes when $M \approx 3000$. This seems very large but we have to take into account that we are sampling a continuous system, so $M = 3000$ (i.e. 3000 sample time steps) is equivalent to $time = 30$ in the original system. In Table 5 we have the details of the test results: the training data contains carrier plus random message. Here the estimate $\bar{\Gamma}$ is close to zero whilst the regression line slope A indicates a fairly simple functional model is required. In Table 5 (as in all these modelling detail tables) the MSError is obtained

⁴ t is used to indicate the sample time step in this section, unless stated otherwise.

from 10,000 samples of carrier plus masking.

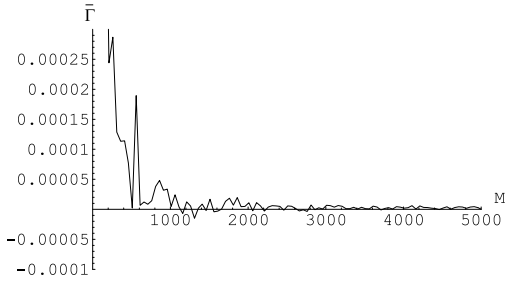


Figure 13: M -test for the Chua system (when a binary message in the form of a square wave was masked)

Embedding	0011110001
M	3000
p_{max}	8
$\bar{\Gamma}$	1.8219×10^{-6}
A	0.29044
MSError	3.3338×10^{-5}

Table 5: Modelling of Chua system with masked binary message in the format of a square wave.

In Figure 14 we see the prediction, the true signal and the errors produced. For each binary signal encoded there are two peaks in the error signal, corresponding to the leading and trailing edge of the square wave, respectively. This is more easily observed in Figure 15, where only part of the error is plotted.

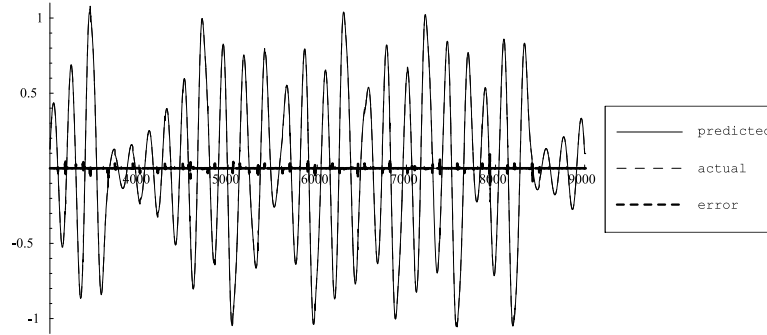


Figure 14: The true signal (normalized), the one-step predictor results (in normalized scale) and the error/difference between them for the Chua system. The differences between the predicted and actual values are so small that the graphs are virtually indistinguishable. A magnified view of part of the error signal is given in Figure 15.

If we examine the errors closely we can detect the binary signal, and we can distinguish two different patterns which seem to correspond to a binary ‘0’ and a ‘1’ respectively. Figures 16 and 17 illustrate the recovery of a binary ‘1’ and a binary ‘0’ respectively in our experiment. We can tell the difference by simply examining the initial direction of the error corresponding to the leading and trailing edge of square wave which encodes the binary bit. These differences in shape are entirely consistent and readily allow successful discrimination between ‘0’ and ‘1’.

In Figure 11, the square wave duration for a binary bit was $\Delta t = 100$. If we reduce this to $\Delta t = 1$ the error graph will show only one spike instead of two. The new modelling details are given in Table 6. Figures 18 and 19 illustrate the two detected distinct patterns that it is still possible to distinguish

between a binary '1' and a binary '0'.

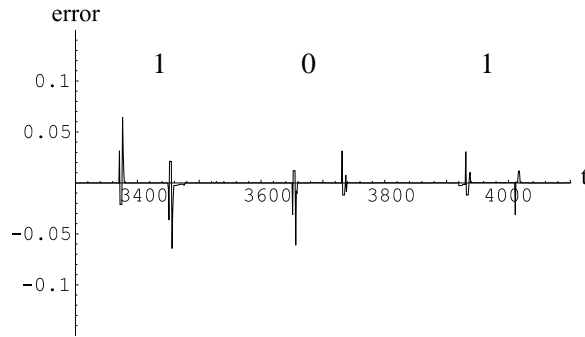


Figure 15: The error between the model and the true Chua System masking the first three bits binary message given in Figure 11.

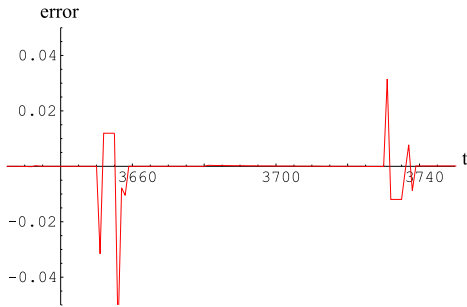


Figure 16: The error between the model and the true Chua system when a '1' was encoded.

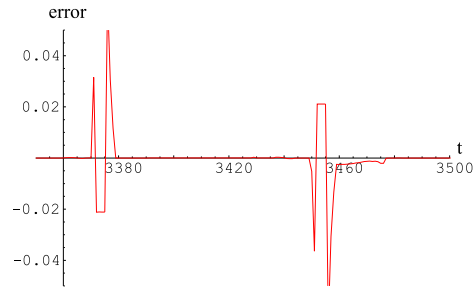


Figure 17: The error between the model and the true Chua system when a '0' was encoded.

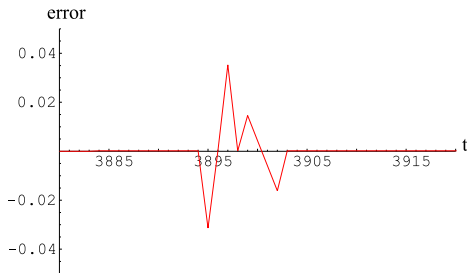


Figure 18: The error between the model and the true Chua system when a '1' was encoded as a "blip" signal.

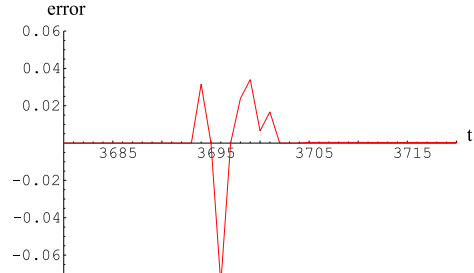


Figure 19: The error between the model and the true Chua system when a '0' was encoded as a "blip" signal.

Thus when we mask a binary message in the Chua system, we can recover the message without much difficulty. The binary '1's and '0's are considered as "noise" by the model, and so by looking at the error between the model and the true signal we can retrieve the message.

Embedding	0001111010
M	2500
p_{max}	8
$\bar{\Gamma}$	1.7716×10^{-7}
A	0.437369
MSError	4.8101×10^{-5}

Table 6: Modelling of Chua system with masked binary message in the format of “blip” signals

4.2 Detecting masked message in a hyperchaotic system

We have also applied the method to the hyperchaotic system described in Equation (11). We masked the binary message shown in Figure 11 in the y variable which is used as the chaotic carrier (see Figure 20). We can see that the amplitude of the binary signal ($= 0.02$) is small in comparison to the hyperchaotic carrier ≈ 30 . The difference between the maximal amplitudes of this system and the Chua system described before is very striking.

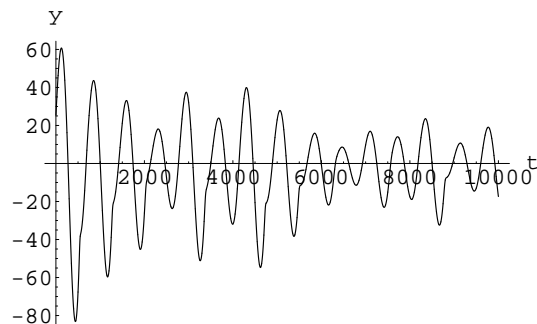


Figure 20: The y variable of the hyperchaotic Rössler system with the added binary message.

We initially approached the problem as before by using the Gamma test on the normalized time series y data from this system, using the Gamma test to determine a mask for a one-step ahead predictor, and implementing the predictor using local linear regression. Encoded signals could be detected, but there were many signal-like blips on the error graph which made identifying the actual signal hard. Eliminating these by filtering the local flow (thresholding the local principal components) could be effective but proved a somewhat *ad hoc* process.

The following idea proved more effective. We use an alternative modelling strategy by considering a different type of mask using *future* values as well as past values of the time series to predict the current value for our predictor. To implement this in a real system requires that the incoming time series be stored in a buffer until sufficient values are present to accommodate the chosen mask. There would then be a small delay before decoding at the receiver. As before the mask is selected by an embedding search for minimal absolute Gamma values, but this time over embeddings of the form $bbbbbxBBBB$, where b

denotes a binary 1 or 0 and x denotes the value which we are trying to predict. The length of the mask either side of the target value can be determined by the equivalent of an increasing embedding.

In fact, the mask we used for this modelling was 10100 x 11001, where x indicates the current value, and this mask means that we should take the third and fifth past values, and the first, second, and fifth future values for the inputs of the model. This mask was discovered again by the Gamma test (with no normalization of data) and the usual results are shown in Table 7. Using the M -test, we estimated that at least 3000 or more time series values are required to model this system, as shown in Figure 21.

In fact, we should arguably use more data, but if the training data contains encoded binary values, as seems probable for the envisaged application, these effectively act as noise in the training data. For local linear regression, even filtering the local flow, this can pose problems although using a neural network sufficient training can effect some degree of noise cancellation and produce a suitably smooth model. It is worth remarking that, once trained, using a neural network also offers better real-time decoding performance.

We therefore took the minimum 3000 points for our training data and to further enhance our predictor, we implemented a feedforward neural network (5-10-10-1 architecture with output function $f(u) = 1.5(2/(1 + e^{-1.2u}) - 1)$, where u is the usual activation function) as our predictor which was trained by using BFGS optimization technique [7] to a training error of 2.87093×10^{-6} , close to the $\bar{\Gamma} = 2.8761 \times 10^{-6}$ of the data set.

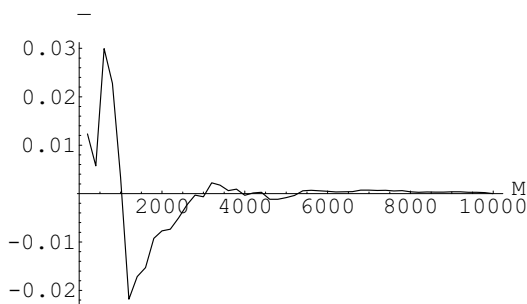


Figure 21: M -test for the Hyperchaotic Rössler system (when the binary message described in Figure 11 is masked).

Embedding	10100x11001
M	3000
$\bar{\Gamma}$	2.8761×10^{-6}
A	0.063725
training MSError	2.8709×10^{-6}
test MSError	2.8102×10^{-6}

Table 7: Modelling the Hyperchaotic Rössler system with masked binary message using feedforward neural network.

In Figure 22 we can see that this technique can retrieve the message as in the example of the Chua system, although for this hyperchaotic system, as one might expect, more modelling effort is required with a *small* signal-to-system amplitude ratio (which is roughly 0.0006 in this case compared with 0.06 for the Chua example). However, if we increase the amplitude of the binary square wave to 0.1 we can easily retrieve it with simple local linear regression technique as before.

As mentioned earlier, even though synchronization of hyperchaotic systems has been achieved, these systems have not (at the time of writing) been applied to secure communications. Therefore we do not yet

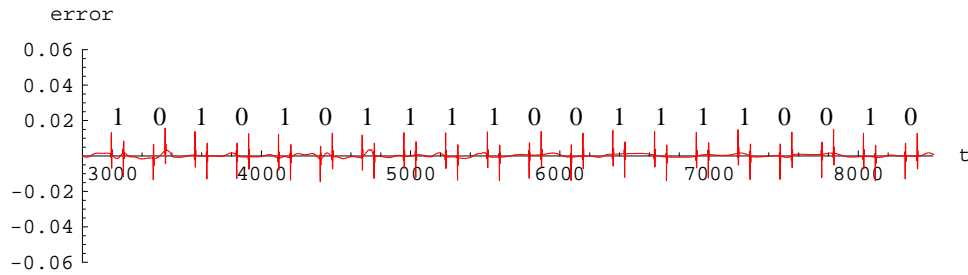


Figure 22: The graph of the error between the predicted value and the true signal of the hyperchaotic Rössler system against the sample time step.

have an estimate for the required amplitude of the binary bit square wave which will allow the recovery of the message when synchronization is achieved under the circumstances of the above experiment. What we can say is that were this to be done with signal-to-carrier ratios above 0.0006 we could be confident of retrieving the signal using these techniques, i.e. without any knowledge of the underlying dynamics and without any necessity to effect synchronization.

5 Conclusions

In this paper we have illustrated with several examples how the Gamma test can be used to determine optimal or near optimal irregular embeddings. We have also given examples how models constructed using these embeddings can be used to detect small amplitude digital signals added to a chaotic carrier. These ideas can also be used to filter seismic data and detect seismic anomalies, such as underground nuclear tests (see [21]). Here the teleseismic signals of interest lie for the most part in the range 0.6 – 3.0 Hz. and distant events can be masked by the microseismic background which overlaps this band. Local modelling with various enhancements can provide a means to overcome these difficulties.

Using the Gamma test analysis we can easily determine a irregular embedding as a basic framework for constructing simple one-step predictors for chaotic time series. We can then use such models, for example to detect hidden binary messages.

Provided the underlying dynamics are such as to produce a smooth embedding model with bounded partial derivatives, the sampling distribution is dense in input space, and the noise distribution has at least the first four moments bounded, so that the typical pre-requisite conditions of the Gamma test are satisfied, we conclude that the Gamma test is an effective tool in the determination of irregular time series embeddings. These conditions are certainly satisfied for the Hénon-type iterative maps, and the Chua and Rössler systems, defined by differential equations, considered here; since the data is generated by an ergodic sampling of input space which wanders across a dense chaotic attractor, and the ‘noise’ is produced by bounded binary modulations imposed upon the chaotic attractor.

With regard to encoding digital signals onto a chaotic carrier, although the use of hyperchaotic systems might increase the difficulty of detection we have shown that even in such a case it is possible to recover an encoded message. However, since the synchronization of hyperchaotic systems has not yet been used successfully for encoding messages we do not know the required amplitude of the binary bit square wave which would allow the recovery of the message when synchronization is achieved. Of course, if this amplitude has to increase in comparison with the masking of ‘normal’ chaotic systems, then the use of hyperchaotic systems would not necessarily increase the level of security.

References

- [1] Aðalbjörn Stefánsson, N. Koncar, and Antonia J. Jones. A note on the gamma test. *Neural Computing and Applications*, 5:131–133, 1997.
- [2] N. Koncar. *Optimization methodologies for direct inverse neurocontrol*. PhD thesis, Imperial College of Science Technology and Medicine, University of London, U.K., 1997.
- [3] Alban P.M. Tsui. *Smooth Data Modelling and Stimulus-Response via Stabilisation of Neural Chaos*. PhD thesis, Imperial College of Science Technology and Medicine, University of London, U.K., 1999.
- [4] Ana Guedes de Oliveira. *Synchronization of Chaos and Applications to Secure Communications*. PhD thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, U.K., 1999.
- [5] F. Takens. Detecting strange attractors in turbulence. In D.A. Rand and L.S. Young, editors, *Dynamical Systems and Turbulence*, volume 898 of *Lecture Notes in Mathematics*, pages 366–381. Springer-Verlag, 1981.
- [6] Kevin Judd and Alistair Mees. Embedding as a modeling problem. *Physica D*, 120:273–286, 1998.
- [7] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition, 1987.
- [8] K. Cuomo and A. Oppenheim. Circuit implementation of synchronized chaos with applications to communications. *Physical Review Letters*, 71(1):65–68, 1993.
- [9] Lj. Kocarev et al. Experimental demonstration of secure communications via chaotic synchronization. *International Journal of Bifurcation and Chaos*, 2(3):709–713, 1992.
- [10] N. Oketani and T. Ushio. Chaotic synchronization of globally coupled maps with an application in communication. *International Journal of Bifurcation and Chaos*, 6(11):2145–2152, 1996.
- [11] U. Parlitz, L.O. Chua, Lj. Kocarev, K.S. Halle, and A. Shang. Transmission of digital signals by chaotic synchronization. *International Journal of Bifurcation and Chaos*, 2(4):973–977, 1992.
- [12] L. Pecora, T. Carroll, G. Johnson, and D. Mar. Fundamentals of synchronization in chaotic systems, concepts and applications. *Chaos*, 7:520–543, 1997.
- [13] K. M. Short. Steps toward unmasking secure communications. *International Journal of Bifurcation and Chaos*, 4(4):959–977, 1994.
- [14] D. Evans. *Data derived estimates of noise using near neighbour asymptotics*. PhD thesis, Department of Computer Science, Cardiff University. Wales U.K., 2001.

- [15] J.L. Bentley. Multidimensional binary search trees used for associative search. *Comm. ACM*, 18:309–517, 1975.
- [16] R.A. Finkel, J.H. Friedman, and J.L. Bentley. An algorithm for finding best matches in logarithmic expected time. *ACM Transaction on Mathematical Software*, 3(3):200–226, 1977.
- [17] A. Oppenheim, G. Wornell, S. Isabelle, and K. Cuome. Signal processing in the context of chaotic signals. In *Proceedings IEEE ICASSP*, pages 117–120, 1992.
- [18] L. Pecora and T. Carroll. Synchronization in chaotic systems. *Physical Review Letters*, 64(8):821–824, 1990.
- [19] M. Brucoli, L. Carnimeo, and G. Grassi. A method for the synchronization of hyperchaotic circuits. *International Journal of Bifurcation and Chaos*, 6(9):1673–1681, 1996.
- [20] J. Peng, E. Ding, M. Ding, and W. Yang. Synchronizing hyperchaos with a scalar transmitted signal. *Physical Review Letters*, 76(6):904–907, 1996.
- [21] K. M. Short. Detection of teleseismic events in seismic sensor data using non-linear dynamic forecasting. *International Journal of Bifurcation and Chaos*, 7(8):1833–1845, 1997.

List of Figures

1	The chaotic attractor produced by the <i>Hénon-type</i> map with delays 5 and 6	7
2	The results of an increasing embedding for the delayed Hénon map (7).	8
3	Results of the M-test for the time series produced by the delayed Hénon map (7).	8
4	The Gamma value increases as we try to predict outputs further away.	8
5	The chaotic attractor produced by a delayed Hénon map with delays 2 and 5.	9
6	The results of an increasing embedding for the delayed Hénon map (8).	9
7	Results of the M-test for the time series produced by the delayed Hénon map (8).	9
8	General detecting/decoding scheme for hidden binary messages in chaotic carrier.	10
9	Hiding a binary message by modulating one of the parameters.	11
10	Hiding a binary message by masking it in a chaotic signal.	11
11	Binary message masked in the chaotic carrier. A positive square corresponds to a ‘1’ and a negative to a ‘0’.	12
12	The y variable of the Chua system plus the binary message.	12
13	M -test for the Chua system (when a binary message in the form of a square wave was masked)	13
14	The true signal (normalized), the one-step predictor results (in normalized scale) and the error/difference between them for the Chua system. The differences between the predicted and actual values are so small that the graphs are virtually indistinguishable. A magnified view of part of the error signal is given in Figure 15.	13
15	The error between the model and the true Chua System masking the first three bits binary message given in Figure 11.	14
16	The error between the model and the true Chua system when a ‘1’ was encoded.	14
17	The error between the model and the true Chua system when a ‘0’ was encoded.	14
18	The error between the model and the true Chua system when a ‘1’ was encoded as a “blip” signal.	14
19	The error between the model and the true Chua system when a ‘0’ was encoded as a “blip” signal.	14
20	The y variable of the hyperchaotic Rössler system with the added binary message.	15

21	<i>M</i> -test for the Hyperchaotic Rössler system (when the binary message described in Figure 11 is masked).	16
22	The graph of the error between the predicted value and the true signal of the hyperchaotic Rössler system against the sample time step.	17

List of Tables

1	Lyapunov Exponents of the delayed Hénon map (7).	7
2	Top three results of a full embedding on the map (7) in the ascending order of $ \bar{\Gamma} $	8
3	Lyapunov Exponents of the delayed Hénon map (8).	9
4	Top three results of a full embedding on the delayed Hénon map (8) in ascending order of $ \bar{\Gamma} $	9
5	Modelling of Chua system with masked binary message in the format of a square wave.	13
6	Modelling of Chua system with masked binary message in the format of “blip” signals	15
7	Modelling the Hyperchaotic Rössler system with masked binary message using feedforward neural network.	16