

Smoothing Strange Attractors with Splines <sup>1</sup>**Junheng Luo<sup>†</sup> and Dominique Thiébaud<sup>‡</sup>**

**Department of Computer Science  
Smith College  
Northampton, MA 01063**

**Abstract**

A noise-reduction algorithm for time-series of non-linear systems is presented. The algorithm smoothes the attractors in phase space using B-splines, allowing a more accurate measure of their dynamics. The algorithm is tested on numerical and experimental data. It is linear in complexity, and can be applied to embeddings of any dimension.

**Keywords**

Nonlinear systems, chaos, noise-reduction algorithm, B-spline.

---

<sup>1</sup> Research supported by NSF Grant # 9212032.

<sup>†</sup> junluo@microsoft.com

<sup>\*</sup> thiebaut@cs.smith.edu, fax # (413)585 3786

## 1. Introduction

This paper introduces a new algorithm for noise reduction of non-linear dynamical systems. Our approach takes place in phase space. It is based on sampling the time series describing the strange attractor of a given system, and on applying B-splines to the different samples. B-splines are a technique used in geometry to smooth out curves, in effect removing high frequency components from them. The smoothing introduced by the splining, when applied to attractors, reduces the amount noise present in the time series data and allows a more accurate estimation of the dynamics of the strange attractor, such as it lyapunov spectra, or the different dimensions. For example, our experiments show that the relative error made in computing the lyapunov exponent of the Rössler attractor corrupted with only 5% of Gaussian noise can be reduced from 73.21% to 1.73% after splining. Our method does not require searching of the phase space, is linear in the number of values in the time-series, is not restricted in the number of elements of the time series, and works for low-dimensional embeddings as well as higher-dimensional ones.

In the next section we introduce B-spline. Section 3 presents the four steps of our algorithm: embedding, sampling, B-splining, and merging. Its application to numerical and experimental data is performed in Section 4.

## 2. B-splines

Experimental data are always subject to error or noise, and *data-fitting* is often required before dynamics can be analyzed. Data-fitting involves the construction of an approximation to an unknown function based on some finite amount of data on the function [9]. The use of polynomial splines is a standard method for data fitting.

When using a polynomial spline to fit a number of prescribed points  $p_0, p_1, \dots, p_n$ , the interval  $[p_0, p_n]$  is partitioned into smaller sections  $[x_0, x_1], [x_{k-1}, x_k]$  where  $k$  is a nonnegative integer,  $x_0$  is the  $x$ -coordinate of  $p_0$  and  $x_k$  is the  $x$ -coordinate of  $p_n$ . Each section is bridged by a polynomial  $f_k(x)$ . Hence the whole spline curve consists of pieces of different polynomials joined together. The points  $x_0, \dots, x_k$  at which two polynomials join are called *knots*. There are several types of splines: *polynomial, interpolation,*

*approximation*, and *basic splines*. The polynomial spline is piecewise polynomial and continuous over the knots. The interpolation splines passes through all the points, that is, the knots are at the points. Because the effect of noise on an attractor reconstructed from an embedding of a time series affects the actual position of the points of the attractor in the phase space, we chose the approximation spline for our algorithm, and in particular the *basic splines*, also known as *B-splines*.

B-splines are spline functions that form a basis for the construction of curves from a set of control points [7]. The B-splines of any degree  $d$ , where  $d$  is the highest power of  $x$  in a polynomial  $P_x$ , and with knot values  $x_i$  can be defined by the Cox-de Boor recursion formula for general B-splines:

$$b_{i,1}(x) = \begin{cases} 1 & \text{if } x_i \leq x \leq x_{i+1} \\ 0 & \text{elsewhere} \end{cases}$$

$$b_{i,k}(x) = \frac{(x - x_i)b_{i,k-1}(x)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - x)b_{i+1,k-1}(x)}{x_{i+k} - x_{i+1}}$$

where  $i$  identifies the knot to which this particular B-spline is attached, and  $k$  is one plus the degree of the associated polynomial. The knot values are chosen at will.

To obtain an approximating curve of  $n+1$  points attached to the first and last points, we choose the following  $n+k+1$  knot values:

$$\begin{aligned} x_i &= 0 & \text{if } 0 \leq i < k \\ x_i &= i - k + 1 & \text{if } k \leq i \leq n \\ x_i &= n - k + 2 & \text{if } n < i \leq n + k \end{aligned}$$

Then we construct a curve  $C$  as  $C(x) = \sum_{i=0}^n b_{i,k}(x)p_i$  where  $C(x)$  is the value of the curve at a particular knot value  $x$ , and is given by summing the product of each control point and the value of its corresponding basic spline at.

## 2.1. B-spline Program in Pseudocode

### Input:

```
n --- number of points
k --- degree of basic splines to be used + 1
multiplier --- (controls number of knots to be used)
```

### Steps:

```
for (knot value = 0; knot value <= n-k+1; knot value +=1/multiplier) do
begin
    compute  $C(\text{knot value}) = \sum_{i=0}^n b_{i,k}(\text{knot value})p_i$ 
end
```

## 3. The algorithm

The input of the algorithm consists of a scalar time series representing the variation of one of the system's parameters as a function of time. This time series is used to reconstruct the attractor of the system in phase space using delay-time coordinates [1]. The time series is filtered through several steps, in such a way that the attractor of the system, embedded in a  $D$ -dimensional space, gets "smoothed" out. Figure 1 illustrates the process.

Insert Figure 1  
here

### 3.1. Embedding

The first step of the process is to create a  $D$ -dimensional embedding of the time series  $V_z(t_i)$ , using a delay  $\tau$  to separate consecutive coordinates of a given point  $x(t_s) = (V_z(t_s), V_z(t_s + \tau), \dots, V_z(t_s + \tau(D-1)))$  and, optionally, a delay  $\delta$  separating the first coordinate of successive points. This embedding is illustrated in Cell 1 of Figure 1. This results in a series of points, such that  $x(t_s)$ , which we abbreviate to  $x_s$ , has for coordinates  $(V_z(t_s), \dots, V_z(t_s + \tau(D-1)))$  while  $x_{s+1}$  has for coordinates  $(V_z(t_s + \delta), \dots, V_z(t_s + \delta + \tau(D-1)))$ .

### 3.2. Sampling

In the sampling step, the points in  $D$ -space are sampled to form  $S$  different curves, as illustrated in Cells 2 and 3 of Figure 1, where the selected points of a sample are shown as

circles. The sampling algorithm simply scans the series of points and assigns points to each of the separate sample curves, in succession. Assuming that the sample curves are referred to as  $C_0, C_1, \dots, C_S$ , Point  $x_0$  is assigned to  $C_0$ ,  $x_1$  to  $C_1$ , ...  $x_{s-1}$  to  $C_{s-1}$ ,  $x_s$  to  $C_0$ ,  $x_{s+1}$ , to  $C_1$ , and so on, until all points are assigned to a sample curve.

### 3.3. Generating B-Splines

In the smoothing step, the sample curves  $C_0, C_1, \dots, C_s$ , are individually processed and for each one a B-spline is generated. The B-spline associated with a sample curve  $C_i$  contains  $Pm$  points, where  $m$  is the *multiplier* associated with the splining process. Cells 4 and 5 of Figure 1 show the two B-splines resulting for the two samples. In Cell 4 the B-spline is shown as a dotted line, while in Cell 5 the B-spline is shown with a dashed line. The smoothness of the B-splines is directly influenced by the multiplier. The higher the value of  $m$ , the smoother the spline.

### 3.4. Merging

The final step takes the  $S$  B-spline curves and merges them into one. Figure 2 illustrates this process for the case  $S=2$ , selected here for simplicity. First the two curves are *aligned*, that is the first point of the first curve (using squares as markers) is compared to the first few points of the second curve (marked with full circles). The circled-point that is the closest to the first square is then paired up with it, and all successive points from the two curves are paired up in succession, until no new pairs can be found. Each pair now contains a point from the first curve and a point from the second curve. Our process does not guaranty that the overall average distance between all paired-points is the smallest, but it is simpler and yield good results.

Once the pairs are formed, the middle point, or *center of gravity*, of each pair is computed. This center of gravity is computed assuming that all the points have equal weight. The curve with hollow circles in Figure 2 shows the resulting curve.

We found that this sample-and-merge process is necessary, and that using only one B-spline for the whole time series resulted in poor estimation of the dynamics of the system.

Insert Figure 2  
here

One interesting note here, is that the method of flow approximation presented by Enge et al. [2], is reminiscent in style of the process of sampling the time series, and merging them, without computing the B-splines.

### 3.6. Algorithmic Complexity.

Clearly the embedding and sampling phases have  $O(n)$  complexity, where  $n$  is the number of data points in the original time series. Computing the B-spline curves requires  $O(n(k+1)m)$ , or  $O(nkm)$  steps, where  $k$  is the degree of the spline plus one, and  $m$  the multiplier. The merging process is sequential in the number of points to merge, and its complexity is  $O(nm)$ . The overall complexity is therefore controlled by the splining process of  $O(nkm)$ . Since  $k$  and  $m$  are constants linked only to the generation of polynomials in the B-splines, and do not depend on  $n$ , the algorithm is essentially linear in the number of data points in the time series. Table 1 summarizes all the parameters of interest controlling the algorithm.

Parameter	Description
$n$	Total number of points
$k$	Degree of spline polynomial plus one
$m$	Multiplier. Controls how many more points are used in the B-spline
$S$	Number of B-spline curves

Table 1: Parameters controlling noise-reduction algorithm.

## 4. Application. The lyapunov exponent

To test our algorithm we use Wolf's lyapunov exponent-computation algorithm and apply it to three attractors, two numerical: the Duffing attractor [1], and the Rössler attractor [8], and one experimental: the time series of Data Set A, from the Santa Fe competition [12]. For the first two we add various amounts of Gaussian noise and apply our noise reduction algorithm, comparing the lyapunov exponent of the noisy and smoothed attractor to the theoretical value.

#### 4.1. The Duffing attractor

Figure 3 shows the Duffing attractors (parameters  $\delta=0.1$ ,  $\gamma=12$ , and  $\omega=1$ ,  $n=4096$  points), with 5%, 10% and 15% of Gaussian noise on the left, along with their smoothed version on the right hand side. A 5% additive noise is characterized by its maximum amplitude reaching 5% of the maximum range of variation of the parameter of interest. In all the cases presented here, the value of the number of B-spline curves is 8, while  $m$ , the multiplier of number of points in the B-splines is equal to 4. We found that  $k=8$ ,  $m=4$  provided the most accurate reconstructions in terms of the precision obtained in computing the Lyapunov exponent.

Insert Figure 3 here

Table 2 shows the results of computing the Lyapunov exponent using our adaptive version of Wolf's algorithm [10] for the different attractors of Figure 3. We see that the B-spline algorithm maintains the relative error on the computation of the Lyapunov exponent to approximately 5% for all levels of noise.

Noise level	Attractor with additive noise		Smoothed attractor	
	Lyapunov exponent $\lambda$	relative error	Lyapunov exponent $\lambda$	relative error
0%			0.136667	-5.7%
5%	0.179893	24.01%	0.153359	5.7%
10%	0.175005	20.64%	0.137794	-5.0%
15%	0.165416	14.03%	0.135663	-6.4%

Table 2. Comparison of Lyapunov exponents obtained using Wolf's method for Duffing attractors. The 0% noise entry is shown here for the purpose of providing an evaluation of the accuracy of our implementation of Wolf's algorithm. For reference we use  $\lambda=0.145064$  as the theoretical first Lyapunov exponent of the Duffing attractor.

## 4.2. The Rössler attractor

Figure 4 shows the same situation, but this time with the Rössler attractor (4096 points).

Table 3 presents the different values for  $\lambda$  along with the relative error incurred in its computation.

Noise level	Raw (unfiltered) attractor		Filtered attractor	
	lyapunov exponent $\lambda$	relative error	lyapunov exponent $\lambda$	relative error
0%			0.123246	-5.74%
5%	0.227694	74.21%	0.135097	3.36%
10%	0.219965	68.30%	0.140490	7.48%
15%	0.204769	56.67%	0.145978	11.68%

*Table 3: Comparison of lyapunov exponents obtained for the raw and filtered Rössler attractors. The 0% noise entry provides an evaluation of the accuracy of our implementation of Wolf's algorithm. For reference, we use  $\lambda = 0.130701$  as the theoretical first lyapunov exponent of the Rössler attractor.*

Insert Figure 4  
here

Here the smoothing has an even more dramatic effect on the reduction of the relative error in the computation of the lyapunov exponent.

## 4.3. Data set A, from the Santa Fe competition.

Data Set A from the Santa Fe competition [2] represents 10,000 measurements of a Far-Infrared-Laser in a chaotic state. Detailed information on Data Set A can be found elsewhere [4]. Because this data set has been widely analyzed and tested [4,5,6], best values for the dimensions, delays and other parameters have been determined, and allow a direct application of our algorithm, further enabling its future comparison to other noise-reduction schemes. Table 4 summarizes the parameters characterizing Data Set A.

Parameter	Value
Embedding dimension	$\geq 5$ [6]
Embedding delay $\tau$	2 [6,4] or 3 [4]
Correlation dimension $D$	2.06 [4]

Table 4: Some important parameters relating to Data Set A. (Numbers in brackets correspond to references where the values are reported.)

Because the embedding dimension for Data Set A's attractor is five or larger, we cannot show it, but we can still apply our algorithm. While B-splines are defined for two dimensions only, we can easily extend their application to an attractor in 5-space by simply applying our algorithm to two dimensions at a time. Coordinates for Dimensions 1 and 2 are treated first, as if they represented a 2-dimensional attractor. Then Dimensions 3 and 4, and finally Dimensions 4 and 5. When combining them back together to get a smoother 5-dimensional attractor, the second set of numbers for Dimension 4 is discarded.

Insert Figure 5  
here

Figure 5 shows the value of the first lyapunov exponent obtained for the attractor of Data Set A, and for its smoothed version, as the embedding dimension is increased from 3 to 8. The computation of the lyapunov exponent on the smoothed attractor becomes more stable once Dimension 5 is reached, and averages 0.01655. This value is smaller than that computed by Kantz [5], and might be attributed to the fairly small  $S_{max}$  value we chose to control point-replacement in Wolf's algorithm. Where Kantz uses absolute values for  $S_{max}$ , we use limits that are relative to the maximum range of variation of the coordinates, specifically 2% of the maximum variation of the coordinates of the points.

## 5. Conclusion

We presented an algorithm that reduces noise in strange attractors by smoothing them using B-splines. This algorithm is linear in terms of the number of samples in the time series and only a few parameters control its behavior. We applied the noise-reduction algorithm to numerical and experimental attractors and showed that the original dynamics, measured in the form of the lyapunov exponent, are conserved. The algorithm is not

limited to 2- or 3-D attractors, and adapts readily to higher dimensions. Furthermore, the algorithm is not sensitive to the length of the data series, and works for short data series as well as for long ones.

In the case of the experimental algorithm, the smoothing introduced by the algorithm yields a smaller range of variation of the lyapunov exponent over increasing embedding dimensions.

## References

- [1] G. Duffing, *Erzwungene Schwingungen Bei veränderlichen Eigenfrequenzen und ihre technische Bedeutung*, (Vieweg, Brunswick, 1918).
- [2] N. Enge, Th. Buzug, and G. Pfister *Physics Letters A*, 175 (1993) 178--186.
- [3] Horsthemke Lefever. *Noise-induced Transitions*. (Springer-Verlag, Berlin, Germany, 1984).
- [4] U. Hübner, C-O. Weiss, N. Abraham, and D. Tang, in *Time Series Prediction: Forecasting the Future and Understanding the Past.*, Wigend and Gershenfeld Eds., (Addison-Wesley, Reading, MA, 1993).
- [5] H. Kantz, in *Time Series Prediction: Forecasting the Future and Understanding the Past.*, Wigend and Gershenfeld Eds., (Addison-Wesley, Reading, MA, 1993).
- [6] F. Pineda, and J. Sommerer, in *Time Series Prediction: Forecasting the Future and Understanding the Past.*, Wigend and Gershenfeld Eds., (Addison-Wesley, Reading, MA, 1993).
- [7] C. Pokorny, and Curtis F. Gerald, *Computer Graphics: The Principles Behind the Art and the Science* (Franklin, Beedle and Associates, Irvine, California, 1989).
- [8] O. Rössler, *Physics Letters* 57A (1976) 397--398
- [9] L. Schumaker, *Spline Functions Basic Theory*. (John Wiley and sons, New York, New York, 1981).
- [10] L. Swinney, J. A. Vasano, A. Wolf, and J. B. Swift, *Physica*, 16D (1985) 285--317.
- [11] Takens, in *Lecture Notes in Mathematics*, Vol. 898, (D. A. Rand and L-S. Young, eds., Springer, Berlin, 1980).
- [12] Wigend and Gershenfeld, Eds., *Time Series Prediction: Forecasting the Future and Understanding the Past*, (Addison-Wesley, Reading, MA, 1993).

## List of Figures:

Figure 1. The phases of the algorithm. In Step 1 the time series is imbedded in a D-dimensional space (D is selected here as 2 for the purpose of illustration). In Steps 2 and 3 the points forming the attractor are sampled to create  $n$  sample curves (here  $n$  is 2). In Steps 4 and 5 the sample curves are smoothed using B-splines (dotted and dashed lines). Finally, the resulting splines are merged in Step 6 to create the smoothed attractor (solid line).

Figure 2. Two B-spline curves are merged. a) Two B-spline curves are selected. b) The curves are first “aligned,” a process in which points from the two curves are paired up. The middle point of each pair is then computed, and the resulting “average” curve is generated (c). The two original B-splines have been merged.

Figure 3. Result of noise-reduction algorithm on the Duffing attractor artificially corrupted with 5%, 10% and 15% Gaussian noise

Figure 4. Result of noise-reduction algorithm on the Rössler attractor artificially corrupted with 5%, 10% and 15% Gaussian noise.

Figure 5. Lyapunov exponent of the Santa Fe Data Set A, as a function of the embedding dimension.

**List of Tables:**

Table 1: Parameters controlling noise-reduction algorithm.

Table 2: Comparison of lyapunov exponents obtained using Wolf's method for Duffing attractors. The 0% noise entry is shown here for the purpose of providing an evaluation of the accuracy of our implementation of Wolf's algorithm. For reference we use  $\lambda=0.145064$  as the theoretical first lyapunov exponent of the Duffing attractor.

Table 3: Comparison of lyapunov exponents obtained for the raw and filtered Rössler attractors. The 0% noise entry provides an evaluation of the accuracy of our implementation of Wolf's algorithm. For reference, we use  $\lambda=0.130701$  as the theoretical first lyapunov exponent of the Rössler attractor.

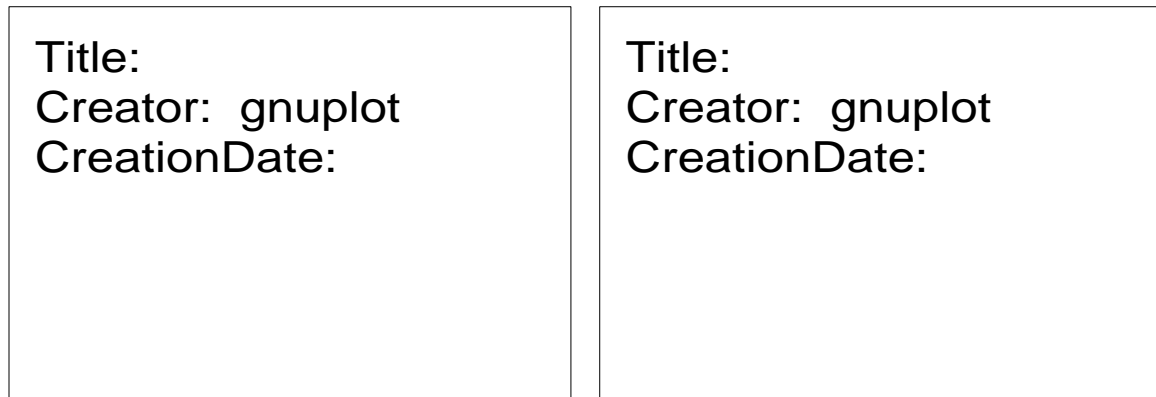
Table 4: Some important parameters relating to Data Set A.  
(Numbers in brackets correspond to references where the values are reported.)

Title: ALGO.EPS  
Creator: Freelance Plus 3.01  
CreationDate: 12/28/1995

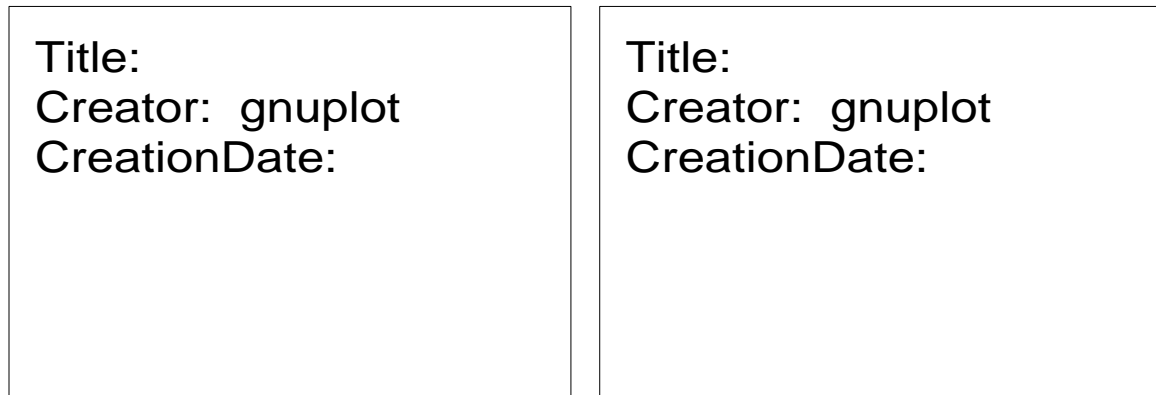
*Figure 1. The phases of the algorithm. In Step 1 the time series is imbedded in a  $D$ -dimensional space ( $D$  is selected here as 2 for the purpose of illustration). In Steps 2 and 3 the points forming the attractor are sampled to create sample curves (herein is 2). In Steps 4 and 5 the sample curves are smoothed using B-splines (dotted and dashed lines). Finally, the resulting splines are merged in Step 6 to create the smoothed attractor (solid line).*

Title: MERGE.EPS  
Creator: Freelance Plus 3.01  
CreationDate: 12/28/1995

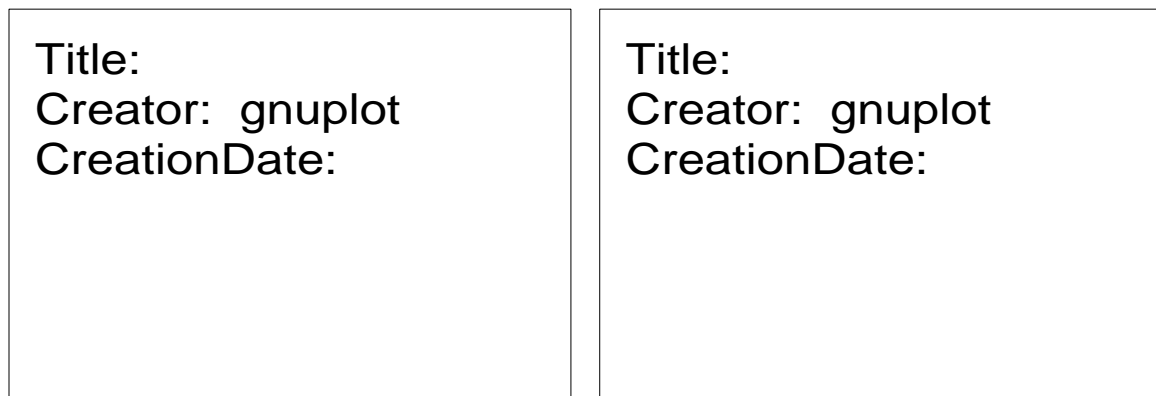
*Figure 2. Two B-spline curves are merged. a) Two B-spline curves are selected. b) The curves are first “aligned,” a process in which points from the two curves are paired up. The middle point of each pair is then computed, and the resulting “average” curve is generated (c). The two original B-splines have been merged.*



*a). Duffing attractor with 5% Gaussian noise.*

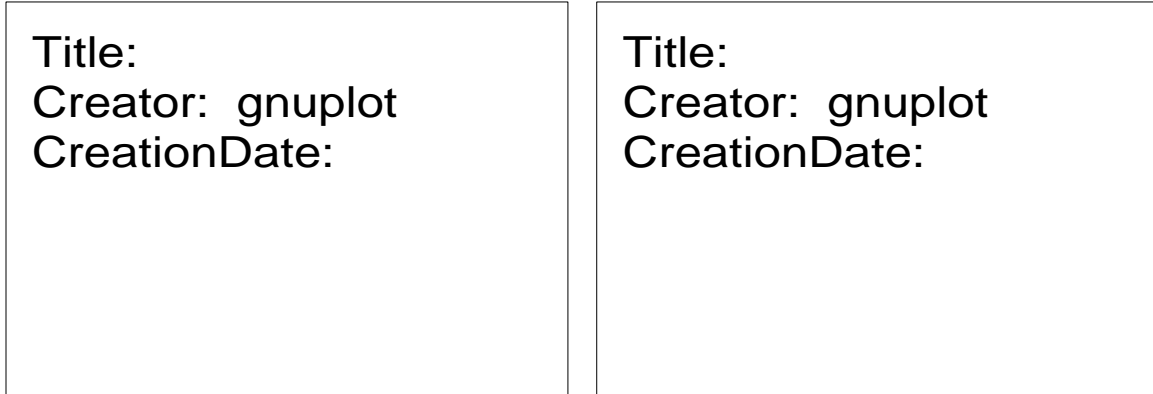


*b). Duffing attractor with 10% Gaussian noise.*

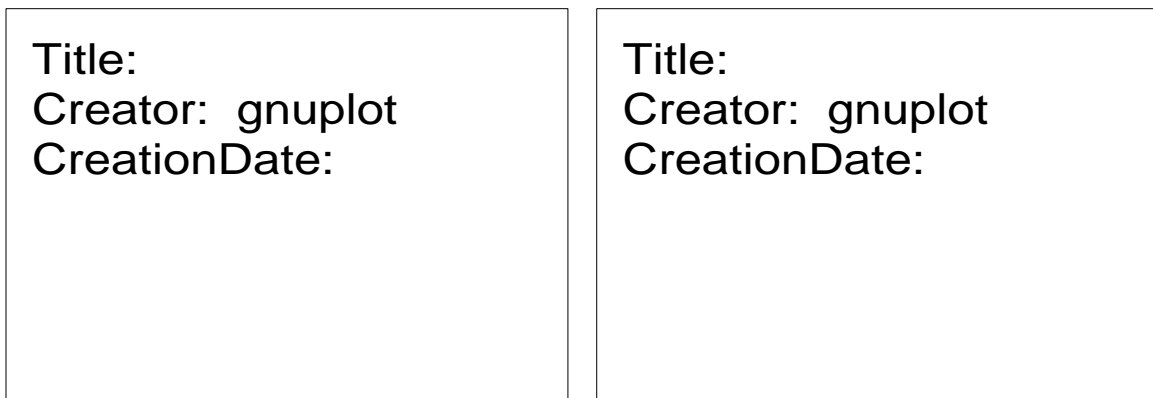


*c). Duffing attractor with 15% Gaussian noise.*

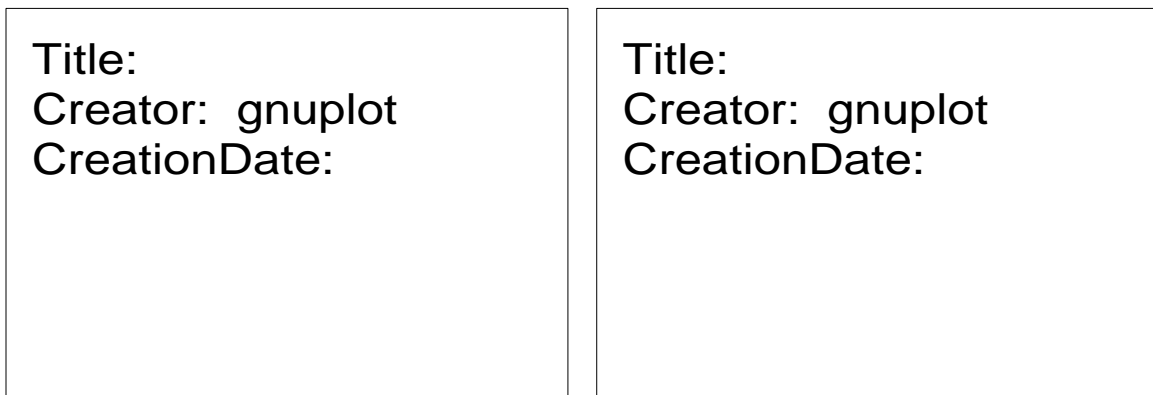
*Figure 3. Result of noise-reduction algorithm on the Duffing attractor artificially corrupted with 5%, 10% and 15% Gaussian noise*



*a). Rössler attractor with 5% Gaussian noise.*



*b). Rössler attractor with 10% Gaussian noise.*



*c). Rössler attractor with 15% Gaussian noise.*

*Figure 4. Result of noise-reduction algorithm on the Rössler attractor artificially corrupted with 5%, 10% and 15% Gaussian noise.*

*Figure 5. Lyapunov exponent of the Santa Fe Data Set A, as a function of the embedding dimension.*

