Dynamical Recurrent Neural Networks towards Prediction and Modeling of Dynamical Systems

Alex Aussem

Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes (LIMOS)

ISIMA - Université Blaise Pascal, Clermont Ferrand II

Campus des Cézeaux - B.P. 125 63173 AUBIERE Cedex FRANCE

e-mail : alex@sp.isima.fr Tel: +33 4 73405039

Abstract

This paper addresses the use of Dynamical Recurrent Neural Networks (DRNN) for time series prediction and modeling of small dynamical systems. Since the recurrent synapses are represented by Finite Impulse Response (FIR) filters, DRNN are state-based connexionist models in which all hidden units act as state variables of a dynamical system. The model is trained with Temporal Recurrent Backprop, an efficient backward recurrent training procedure with minimal computational burden which benefits from the exponential decay of gradient reversely in time. The gradient decay is first illustrated on intensive experiments on the standard sunspot series. The ability of the model to internally encode useful information on the underlying process is then illustrated by several experiments on well known chaotic processes. Parsimonious DRNN models are able to find an appropriate internal representation of various chaotic processes from the observation of a subset of the state variables.

Keywords - Recurrent neural networks, gradient descent, temporal processing, chaotic maps.

1 Introduction

Much of the progress in artificial neural networks over the past years can be ascribed to the development of the back-propagation algorithm [24, 30] an elegant method for training feed-forward networks at low cost. Since then, a considerable number of extension have been proposed for adapting various real- and/or discrete-time neural network architectures, including back-propagation through time [24, 30], recurrent back-propagation [21], real-time recurrent learning [31], temporal back-propagation [28], time-dependent recurrent back-propagation [25], to cite just a few. It is relatively staightforward to propose a powerful connexionist architecture, but it is much more arduous to understand its intrinsic mechanisms, potential and limitations.

In this paper, a very intuitive and straightforward enhancement is achieved trough embedding Finitie Impulse Response filters within a fully recurrent neural structure. By virtue of the recurrency, the DRNN has a trainable internal memory and its behavior is described by a system of nonlinear difference equations, whose internal variables are the hidden unit activations. Also, there is no need anymore to present the memory spatially at the input of a network, DRNN able to retain useful information from the past thanks to the internal memory it has been endowed with.

As a rule of thumb, the more general the architecture, the more tricky and computationally intensive the training. In other words, the attractive features displayed by the model are usually to the detriment of the ease of the training. Fortunately, the number of time slices needed for the backpropagation in time through the DRNN adjoint network is tailored by the exponential decay rate of the norm of the gradient [2, 1]. This result is paramount: the exponential vanishing rate of the gradient allows the expansion backwards in time of the *adjoint* to be limited while still following as precisely as desired the true gradient trajectory. This yields a local and recursive procedure, called Temporal Recurrent Back-Propagation (TRBP), that considerably reduces the amount of storage and computational resources required by a forward propagation algorithm.

In the experiments presented in this paper, our primary interest is not to assess the predictive ability of DRNN models against other forecasting techniques but instead to: i) illustrate the theoretical analysis regarding the error gradient propagation in the adjoint, especially when only the current time series value, x_t, is directly supplied to the network for the prediciton of x_{t+1}, and ii) force the DRNN to encode as much information of the past as possible to infer some form of embedding dimension regarding the underlying process.

Therefore, intensive simulations are first carried out on the classical sunspot series to confirm and illustrate the exponential decay of the gradient reversely in time. In order to demonstrate the modeling capabilities of DRNN, networks of various dimensions are then trained on time series generated by chaotic processes. The ability to infer the unknown internal variable of the process is clearly ilustrated. When iterated thousands of time steps into the future, the signal continues to accurately represent the system dynamics and exibit no noise-like corrruption. While the neural network outputs diverge from the true sequence - an unavoidable consequence of the chaotic nature of time series -, the chaotic attractor reconstructed from the closed-loop DRNN is almost identical to the true attractor. Experiments are carried out on the Makey-Glass equations, the Hénon map, the Lorenz equqtions and the Ikeda map. In all cases, the DRNN was able to find an appropriate representation to internally encode useful information.

The paper is organized as follows. In Section 2, we briefly review the neural based techniques that have been employed so far as nonlinear extensions of traditional linear mlodels with a view to ascertaining the problems for which they are best suited. Section 3 presents the specific architecture of the DRNN model. Temporal recurrent backprop is briefly discussed in Section 4 and the DRNN aplications on various chatic processes are carried out in Section 5.

2 Nonlinear Modeling

The problem of designing a neural model of an unknown process based on observed data, and without any physical insight in the underlying dynamics, has attracted much attention during the past years [16, 19, 23, 26]. For the time being, most of the literature is concerned with black-box modeling usually performed using input-output models. This paragraph provides an understanding of of their limitations. We show that state-space models such as DRNN constitute a broader class of nonlinear dynamical models capable of remedying these limitations.

2.1 Optimal Predictors

We briefly review the cornerstone of black-box modeling in order to highlight the role of neural networks in approximating the optimal predictors of nonlinear dynamical systems [8]. One of the class of models which appeared to offer plausible description of a wide range of different types of time series is the nonlinear extension of the *Autoregressive Moving Average* model termed NARMA(p,q) in the literature. If x_t denotes the value of the series recorded at time point T, we may write the ARMA(p,q) process in the form,

$$x_t = f(x_{t-1}, \ldots, x_{t-p}, e_{t-1}, \ldots, e_{t-q}) + e_t \tag{1}$$

where $f()$ is an unknown function. It is assumed that $\mathrm{E}[e_t \mid x_{t-1}, x_{t-2}, \ldots] = 0$. This condition is satisfied for instance when e_t are zero mean, independent and identically distributed and are independent of the past x_t's. Without loss of generality, it is assumed that e_t has zero mean and finite variance σ^2.

The theory of prediction focuses on optimum prediction of dynamical systems in the minimum mean square sense. When the conditional mean existsis, it is given by

$$\hat{x}_t = \mathrm{E}[f(x_{t-1}, \ldots, x_{t-p}, e_{t-1}, \ldots, e_{t-q}) \mid x_{t-1}, x_{t-2}, \ldots, x_1] \tag{2}$$

Suppose that the NARMA is invertible in the sense that there exists a function $g()$ such that

$$x_t = g(x_{t-1}, x_{t-2}, \ldots) + e_t \tag{3}$$

then given the infinite past observations, one can in principle use the above equation to compute the e_{t-j} in (2) as a function the past observations $x_{t-1}, x_{t-2}, \ldots$ such that (2) becomes

$$\hat{x}_t = f(x_{t-1}, \ldots, x_{t-p}, e_{t-1}, \ldots, e_{t-q}) \tag{4}$$

where the e_{t-j} are specified by $x_t - g(x_{t-1}, x_{t-2}, \ldots)$. The predictor has a mean-squared error σ^2. Since in practice one has only access to a finite observation record, e_{t-j} cannot be computed exactly; it seems reasonable, as for the linear ARMA process, to approximate the conditional mean predictor (4) by the following formula

$$\hat{x}_t = f(x_t, \ldots, x_{t-p}, \hat{e}_{t-1}, \ldots, \hat{e}_{t-q}) \tag{5}$$

where $\hat{e}_{t-i} = x_{t-i} - \hat{x}_{t-i}$. The optimal predictor is thus given by 5 provided that the effect of arbitrary initial conditions will die away depending on the unknown function f.

Such a general NARMA scheme provides a valuable framework for modeling a wide range of different types of nonlinear time series structures of the type (1). They have an advantage over NAR models in much the same way that linear ARMA models have advantages over autoregressive models for some types of series. However, the length of the delay line, $p + q + 1$, of shift register must be chosen in advance to accommodate the longest possible sequence. Moreover, in contradistiction to the linear case, the is no simple equivalence between nonlinear input-ouput and state-space models. Therefore, if input-output modeling is unsatisfactory, a natural step is to resort to state-space models.

2.2 State Space Paradigm

Whereas it is always possible to rewrite a nonlinear input-output model in a state-space representation, conversely, an input-output model equivalent to a given state-space model might not exit [15], and if it does, it is surely of higher order [16]. Conciser the following deterministic state-space model:

$$\left\{ \begin{array}{lcl} x_{t+1} & = & f(x_t, u_t) \\ y_t & = & g(x_t) \end{array} \right. \tag{6}$$

where u_t is a scalar external input, y_t is the scalar output, and x_t is the n-state vector of the model at time t. Under fairly general conditions on the *observability* of the system, an equivalent input-output number of past model does exist, and is given by:

$$y_t = h(y_{t-1}, \ldots, y_{t-r}, u_{t-1}, \ldots, u_{k-r}) \tag{7}$$

with $n \leq r \leq 2n+1$ [16]. Therefore, state-space models are likely to have lower order and require a smaller number of past inputs, and hopefully a smaller number of parameters. This is of great importance when a limited number of data is available. State-space models can be used as black-box models with additional outputs dedicated to modeling the state variables of the process. This gives the neural predictor more flexibility while still taking advantage of the feed-forward structure.

Consider the stochastic state-space model given by

$$\left\{ \begin{array}{lcl} x_{t+1} & = & f(x_t, u_t, \epsilon_t^1) \\ y_t & = & g(x_t, \epsilon_t^2) \end{array} \right. \tag{8}$$

where ϵ_t^1 and ϵ_t^2 are sequences of zero mean i.i.d. random indepedent vectors. If $f()$ and $g()$ are known, the extended Kalman predictor gives a *sub-optimal* solution using a linearization of the model around the current state estimate. The one step ahead predictions are computed from the following recursion:

$$\left\{ \begin{array}{lcl} \hat{x}_{t+1} & = & f(\hat{x}_t, u_t, 0) + K(t, \hat{x}_t) \cdot (y_t - g(\hat{x}_t, 0)) \\ \hat{y}_{t+1} & = & g(\hat{x}_{t+1}, 0) \end{array} \right. \tag{9}$$

where $K(t, \hat{x}_t)$ is the time-varying and state-dependent extended Kalman gain, which is computed using the linearizations of f and g. In practice however, these functions are partially or completely unknown, there is thus no reason to stick to this structure. Such time-invariant associated predictor can be put in more general form:

$$\left\{ \begin{array}{rcl} \hat{x}_{t+1} & = & \mathcal{N}_1(\hat{x}_t, y_t) \\ \hat{y}_{t+1} & = & \mathcal{N}_2(\hat{x}_{t+1}) \end{array} \right. \tag{10}$$

where $\mathcal{N}_j, j = 1, 2$ may be implemented by feed-forward networks for instance. This state-space based model can be modeled by two cascaded feed-forward networks. However, a feed-forward network in which the output is fed back to the input, is a special case of the somewhat more general class of fully interconnected networks [2, 8]. For such networks, there is no need to present the $\hat{e}_{t-j}$ and the state variables $x_{t-j}, j > 1$, to the input since specific non trainable recurrent links from a hidden layer towards itself results in the the same functional input-output mapping. The snag is that the increase in architectural complexity is usually to the detriment of the ease of the training. We show in the next sections that the training effort for the DRNN can be maintained within acceptable limits thanks to the exponential decay of the gradient backpropagated in time.

3 The DRNN Model

Dynamical recurrent neural networks [2] are a class of state-based neural networks involving explicit inclusion of delays in the neural system equations. In such a system, multiplicative interactions of scalars are substituted by temporal convolution operations necessitating a time-dependent weight matrix [13, 14, 28]. A general formulation of delayed networks in discrete time is a coupled differential equations of the form:

$$\begin{array}{rcl} \mathbf{T}\dfrac{d\mathbf{v}_k(t)}{dt} & = & -\mathbf{v}_k(t) + g(\mathbf{u}_k(t)) + \mathbf{i}_k \\ \mathbf{u}_k(t) & = & \displaystyle\sum_{d=1}^{D} \mathbf{W}_k^{d^T} \mathbf{v}_{k-d} + \mathbf{W}_k^{0^T} \mathbf{v}_k(t) \end{array} \tag{11}$$

where N is the number of units, $\mathbf{v}_k(t)$ and $\mathbf{u}_k(t)$ are the N-dimensional vector activity and internal inputs of the units at time t, whose values are determined at each step k by the relaxation of the system (11) of coupled differential equations. $\mathbf{i}_k$ denotes the external input vector of dimension N and $\mathbf{T}$ is a diagonal matrix of positive time constant.

Let the matrix element w_{ij}^d denote the connection strength, or coupling, from the d-th delay link connecting node i to node j and D the maximum connection delay, i.e., the filter order. The weights for the d-th delay connections heading to unit j at iteration k are specified by the vector $\mathbf{w}_k^d(j) = [w_k^d(1,j), \ldots, w_k^d(N,j)]^T$, and the $N \times N$ matrix $\mathbf{W}_k^d = [\mathbf{w}_k^d(1), \mathbf{w}_k^d(2), \ldots, \mathbf{w}_k^d(N)]$ contains all adaptive weights of delay $d \leq D$ at iteration k.

We do not allow for connections towards input units, i.e., $w_{ij}^d = 0$ if j is an input unit, and set $i_k(j) = 0$ if unit j is not an input unit. Output units have target values $\mathbf{d}_k$. In addition, a bias input to each unit is achieved by fixing a particular unit activation to 1.0. For notational convenience, we won't distinguish between input, hidden, output and bias units in the following formulae. The sigmoidal function $g(h) = (1 + e^{-\beta h})^{-1}$ is a common choice for the nonlinear activation function. Consistent with our notations, we note $g(\mathbf{u}_k) = [g(u_k^1), \ldots, g(u_k^n)]^T$.

4 Temporal Recurrent Backprop

The introduction of time delays leads to somewhat more complicated training procedure because the current state of a unit may depend on several events having occurred in the neighboring neurons at different times. Consequently, the gradients with respect to past and present parameters have to be determined for proper parameter adaptation. For purpose of completness, the key results on the training procedure are addressed in this section.

Classically, the goal is to find a procedure to adjust the weight components w_{ij}^d so that a given fixed initial conditions $\mathbf{v}_0$ and a set of input vectors $\{\mathbf{i}_k\}$ taken from the training set result in a set of fixed points, whose components along the output units have a desired set of values $\{\mathbf{d}_k\}$. This will be accomplished by our minimizing a quadratic function $E = \frac{1}{2}\sum_{k=0}^{k_f} \mathbf{e}_k^T \mathbf{e}_k$, which measures the errors $\mathbf{e}_k = \mathbf{d}_k - \mathbf{v}_k$ between the desired and the actual fixed point values along a trajectory or epoch starting at iteration 0 up to k_f. In an epochwise adaptation, a new weight is generated at the end of the trajectory by $\Delta w = -\eta \frac{\partial E}{\partial w}$, where η is the learning rate. The primary goal is to find an expression for the term $\partial E / \partial w$. In order to clarify our notation, let $\mathbf{G}_k$ and $\mathbf{G}_k'$ denote the $N \times N$ diagonal matrixes constructed from $g(u_k(j))$ and $g'(u_k(j))$ respectively, for $j = 1, \ldots, N$. Note that for sigmoid functions, $\mathbf{G}\text{'}_k = \beta \mathbf{G}_k (1 - \mathbf{G}_k)$ (Recall that $g'(h) = \beta g(h)(1 - g(h))$).

Proposition 1 - *If the DRNN model is convergent over the trajectory $0 \leq k \leq k_f$, then the gradient of cost can be computed at time step k_f by*

$$\frac{\partial E}{\partial \mathbf{W}^d} = \sum_{k=d}^{k_f} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T \quad \forall d \leq D \tag{12}$$

where the vector $\boldsymbol{\delta}_k$ *is given by* $\boldsymbol{\delta}_k = \mathbf{G'}_k \mathbf{y}_k$ *and* $\mathbf{y}_k$ *is the solution of the following backward difference equation*

$$\mathbf{y}_k = [\mathbf{I} - \mathbf{W}_k^0 \mathbf{G'}_k]^{-1}[\mathbf{e}_k + \sum_{j=1}^{D} \mathbf{W}_{k+j}^j \mathbf{G'}_{k+j} \mathbf{y}_{k+j}] \tag{13}$$

with the boundary conditions:

$$\begin{aligned} \mathbf{y}_{k_f} &= [\mathbf{I} - \mathbf{G'}_{k_f} \mathbf{W}_{k_f}^0]^{-1} \mathbf{e}_{k_f} \\ \mathbf{y}_j &= \mathbf{0} \ \textit{for} \ j > k_f - D \end{aligned}$$

The demonstration of this proposition has been given elsewhere [2]. It is based on the repeated application of partial derivative chainrules in a dynamical system for the computation of the error gradient. This generalized ∂ rule is essentially a *backsweep* or *backward* recurrent propagation procedure in that it puts recurrent-backprop [21], temporal backprop [28] and backprop-through-time [24, 30] into a single framework.

As BPTT in its general from, the procedure suggested here makes necessary the artifact of unfolding the adjoint network in a reverse way in time. This may be achieved by replicating the units and the connections backward through time until all delays have been removed. However, because the storage and memory requirements grow proportionally with the pattern index, the procedure becomes swiftly inhibitory from an implementation standpoint when long trajectories are to be considered. This is also the major flaw of BPTT. Fortunately, the cost due to the duplication of the hardware, here, can be maintained within acceptable limits, thanks to the fast decay of the gradient through the adjoint network.

4.1 Vanishing Gradient

Due to the dynamical behavior of the DRNN, the initial conditions of the system at time 0, namely $\mathbf{v}_0$ and $\boldsymbol{W}_0$, along with the sequence of input patterns $\mathbf{i}_k$ completely determine the trajectory of the system. Therefore any change in a weight value at some time will influence the subsequent behavior of the system. In order to prevent the systematic expansion of the error-propagation network and to ensure the stability of the parameter adaptation procedure, a fundamental result is stated below.

Define $\mathbf{y}_k^{k+n}$ as the solution of Eq. 13 at time k when all $\mathbf{e}_{k+j}$ are set to $\mathbf{0}$ except for $j = n$. Given the linearity of Eq. 13, we may write

$$\mathbf{y}_k = \sum_{n=0}^{k_f - k} \mathbf{y}_k^{k+n} \tag{14}$$

Now define

$$\boldsymbol{\delta}_k^{k+n} = \mathbf{G'}_k \mathbf{y}_k^{k+n} \tag{15}$$

We may now expand the delta-rule (12) by incorporating the terms $\mathbf{y}_k^{k+n}$ and $\boldsymbol{\delta}_k^{k+n}$

$$\Delta \mathbf{W}^d = \eta \sum_{k=d}^{k_f} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T = \eta \sum_{n=d}^{k_f} \sum_{k=d}^{n} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T \tag{16}$$

Typically, all the individual weight adjustment resulting from the duplication of the connections in the unfolded adjoint network must be recombined to compute the overall change. This is achieved by summing the individual gradient components over the overall unfolding interval. The question we are now faced with is whether the sum is diverging or remains bounded as $k_f \to \infty$. Answering this question in the general framework is a formidable task since the weight evolution is strongly dependent on the dataset itself. In practice the weights may blow up, if no regulizer is used, only to be limited by their saturation limits. In fact, it seems that no general answers can be formulated. But as we are focusing on a samplewise algorithm, what we are concerned with is not the *global* stability of the weight updates over the overall trajectory, but instead the stability at *each* time step of the recursive backward propagation through the adjoint network. Therefore, we will say that the backpropagation is *stable* at each time step whenever:

$$\lim_{n \to \infty} \| \sum_{k=d}^{n} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T \| < \infty \tag{17}$$

With this in mind, a key result can be stated as follows:

Proposition 2 - If *the following conditions hold* $\forall d = 1, \ldots, D$, $\forall k = 0, \ldots, k_f$

$$||\mathbf{W}_k^0|| + D \cdot ||\mathbf{W}_{k+d}^d|| < \frac{4}{\beta} \tag{18}$$

then $\exists M > 0$ *such that*

$$||\mathbf{y}_{k_f-n}^{k_f}|| < M \left[D \cdot \max_{k,d} \left(\frac{\beta||\mathbf{W}_{k+d}^d||}{4 - \beta||\mathbf{W}_k^0||} \right) \right]^{\frac{n}{D}} \tag{19}$$

holds for $n = 0, 1, \ldots, k_f$.

This technical result is a recent extension of the theoretical developments presented in [2]. The proposition says, in essence, that the terms $\mathbf{y}_{k-n}^k$ are bounded above by a series decaying at exponential rate with respect to n given some necessary conditions on the weights which, by the way, are also sufficient conditions for the stability of the weight update procedure for the DRNN. More specifically, provided the weights are kept fixed over the trajectory, we obtain two explicit sufficient conditions:

$$\begin{cases} D \cdot max_{i,d} \sum_j |w_{ij}^d| + max_i \sum_j |w_{ij}^0| < \frac{4}{\beta} \\ D \cdot max_{j,d} \sum_i |w_{ij}^d| + max_j \sum_i |w_{ij}^0| < \frac{4}{\beta} \end{cases} \tag{20}$$

Note that these upper bounds are in nice agreement with the results found in the literature for recurrent networks that are special cases of the DRNN framework [10, 12, 31]. Interestingly, the existence of $[\mathbf{I} - \mathbf{G'}_k \mathbf{W}_k^0]^{-1}$, and thereby the *stability* of the fixed points, are also guaranteed. It suffices to set $D = 0$, in (18). This yields $||\mathbf{W}_k^0|| \cdot ||\mathbf{G'}_k|| < 1$ (since $||\mathbf{G'}_k|| < \beta/4$) which is a sufficient condition for the existence of $[\mathbf{I} - \mathbf{G'}_k \mathbf{W}_k^0]^{-1}$.

Under stable conditions, the above theoretical decay rate may be used to determine the number of time steps the gradient has to be back-propagated to ensure a proper adaptation of the network parameters, i.e. to ensure that the weight changes computed at each iteration follow the true gradient as precisely as desired.

In practice, however, the decay rate observed in the simulations is somewhat smaller than the theoretical one but this is beyond the scope of this paper to delve deeper into these issues. In the experiments presented in this paper, the number, s, of time slices needed for the backpropagation through the adjoint network is tailored by the observed decay rate of the norm of the gradient. But instead of inferring M and approximating the observed decay rate by linear regression, we use an implementation-friendly criterion which avoids any computational burden. More specifically, the gradient propagation backward in time is stopped when the norm of the gradient satisfies

$$\frac{||\mathbf{y}_{k-s+j}^{k}||}{||\mathbf{y}_{k}^{k}||} \leq \epsilon \quad \text{for} \quad j = 1, \ldots, D \tag{21}$$

where ϵ is an arbitrary predefined value, equal to 10^{-2} in our implementation. While s varies with k, the cost due to the duplication of the hardware is maintained within acceptable limits. For further details on the procedure, the reader is directed to [2] as well as references therein.

Vanishing gradient leads to an efficient parallel implementation requiring storage of order DN to keep track of the network previous states and DN^2 to store the filter parameters. The time required at each backward pass to compute the elementary weight update scales as $O(DN^2)$ so that $O(D^2N^2)$ operations are needed at each time step to adjust the network parameters.

It is essential to note, however, that this procedure essentially *gives up* on the the long time lag problem by optimally cuting off the gradient, thus reducing the computational effort involved. Therefore, it should be clearly understood that TRBP is surely to fail if the time gap between an event and the occurrence of the corresponding error signal spans a length of time significantly greater than the maximum delay order present in the DRNN structure. In this case, one has to resort to more sophisticated architectures or data processing techniques [4]

5 Experimental Results

In this section, we report on extensive investigations of the performance of the DRNN as an anticipative model for the behavior of the well known time series, namely the standard sunspots and several series generated by chaotic processes. Our main goals were not to assess the predictive ability of DRNN models against other forecasting techniques but instead to

- confirm and illustrate by simulation the theoretical analysis regarding the error gradient propagation in the adjoint, especially when no past information $(x_{t-1}, x_{t-2}, \ldots)$ is directly supplied to the network for the prediciton of x_{t+1}
- force the dynamical network to encode as much information of the past as possible to infer some form of embedding dimension regarding the underlying process.

Comparisons are made on the basis of the Normalized Mean Squared Error (NMSE), a standard measure of fit given by:

$$\mathrm{NMSE} = \frac{1}{\sigma^2 N} \sum_{k=1}^{N} (x(k) - \hat{x}(k))^2 \tag{22}$$

where $x(k)$ is the true value of the sequence, $\hat{x}(k)$ is the prediction, and σ^2 is the variance of the true sequence over the prediction duration N. A value of the NMSE $= 1$ thus corresponds to predicting the unconditional mean.

5.1 The sunspot series

The sunspot series was the first time series ever studied with autoregressive models [6], and thus has served as a benchmark in the statistical literature [6, 22, 27] as well as in the connectionist literature [29]. Consistent with these appraisals, we use range-normalized yearly averages of the sunspot data tabulated from 1720 to 1979. The single-step prediction error was monitored on the withheld 59 sunspot values ranging from 1921 to 1979, while the remaining data is used for training.

The framework pursued here attempts to force the DRNN to encode the sequence internally. In this spirit, we experiment with a generic $1 \times N \times 1$ architecture in which the *single input* x_{t-1} is instantly propagated to the N hidden units fully connected with D-th order connections; i.e. no delays are used in the input-to-hiddden connections. Clearly, Given the lack of spatial information, the DRNN is forced to encode *internally* some information of the past to infer the future value. The single output unit is linear.

Networks of various dimensions were taught to provide the best one-step ahead estimate $\hat{x}_t$ given $\hat{x}_{t-1}$. A weight elimination approach [29] was adopted and η was varied in the course of training to speed up training as discussed in [2]. The smallest acceptable solution we arrived at in terms of number of parameters ($N^2(D+1)$ weights) is the solution $N = 2, D = 5$. This solution exhibits the best trade-off between the increasing bias and the decreasing variance.

For purposes of illustration, the one-step-ahead prediction NMSE measures are shown in Table 1 for the 2-neuron DRNN solution driven by the prior sunspot number and two standard benchmark models for the sunspots, namely the TAR model [27] and the multilayer perceptron [29]. While the DRNN model seems to offer no dramatic advantage over a multilayer perceptron in terms of performance, the number of parameters has been significantly reduced from 43 to 30 even if the TAR model [27] has still a lower number of parameters.

Model	training set	1921-1955	1956-1979	nb parameters
Carbon copy	0.289	0.427	0.966	-
TAR model	0.097	0.097	0.280	18
MLP $12 \times 3 \times 1$	0.082	0.086	0.350	43
DRNN $N = 2, D = 5$	0.105	0.091	0.273	30

Table 1: Single-step prediction performance assessment with Carbon copy prediction (i.e. $\hat{x}_t = x_{t-1}$), threshold autoregressive model (TAR) model and a multi-layer perceptron (MLP). The number of adjustable parameters is also shown.

5.1.1 Back-Propagated Errors

Figure 1 illustrates the behavior of the terms $||\mathbf{y}^n_{n-j}||_\infty$, averaged over the overall training set for the different network configurations. They are plotted on a log scale as a function of the retrograde time step j in the unfolded adjoint network. The curves are in nice agreement with our analytical findings. Typically, the error terms drop off linearly from the first time lag onwards, mostly oscillating slightly along a line through $-\infty$. It should be pointed out that estimates of the error gradient decay rates can be measured directly from the plots. In all cases, a quasi monotonic decrease is observed indicating that the stopping criteria of the unfolding process is adequate. With a residual error of $\epsilon = 10^{-2}$, it is seen that the average number of retrograde steps the adjoint had to be unfolded is 15, that is 3 times the filter order D.

A histogram of the average number of time-layers in the adjoint network for the $N = 2, D = 5$ network is plotted in Fig. 2. The curve ranging from 15 to 45 is centered around 30, that is, 6 times the connection order. It is readily seen that a bookkeeping of the 45 past states of the network, that is $9 \times D$, is sufficient for a proper back-propagation to be performed. This is consistent with the decay rate of the network shown in Fig. 1b. It must be emphasized that "best" model $N = 2, D = 5$ in terms of bias/variance ratio, has also the slowest gradient decay rate among the 30 networks we experimented with. For purposes of visualization, the histogram of the average variation of the number of time-layers in the adjoint between successive iterations is plotted Fig. 3. As seen, the variation is mostly confined in the interval $[-D, +D]$,

5.1.2 Underlying Law Characterization

In the context of real world time series where only a limited number of recorded observations are available, frequently the small amount of data excludes a confident characterization of the underlying law governing the series. It is therefore instructive to see whether neural networks in general can help in characterizing some essential properties underlying the data. With this objective, Weigend et al. varied the input delay vector length of a standard MLP and observed the resulting performance variation [29]. They estimated the *dimension of the recursion* around 12.

The experimental results obtained with fully-recurrent DRNN allow us to further the identification of the underlying system. The optimal solution, $N = 2, D = 5$, we arrived at - in terms of complexity/performance rate - have certainly essential elements of the optimal system of equations of which we assumed the existence, suggesting that the dimension of the system of differential equations is around 2. In other words, there *might* exist a system of 2 second-order differential equations that approximate with reasonable certainty the behavior of the sunspot series.

6 Experiments with Chaotic Processes

In the examples that follow, the DRNN models of various dimensions are trained on time series generated by chaotic processes. The task is to achieve accurate single-step predictions, the network being fed with a *subset* of the state variables. The set-up is chosen such as to facillitate comparisons with previous work [7, 28], in particular with FIR networks which are *feedforward* versions of DRNN.

6.1 Mackey-Glass

For this first example, consider the Mackey-Glass equations:

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-\Delta)}{1+x(t-\Delta)^{10}} \tag{23}$$

A 4-*th* order Runge-Kutta technique was used to simulate the chaotic series associated with the values $\Delta = 17$, $\Delta = 30$; the same parameters were used to allow consistent comparisons with prior work [7, 28] (i.e. $x(t) = 0.9$ for $0 \leq t \leq \Delta$ and a sampling rate of $f = 6$, Fig. 7).

	NMSE	
model	$\Delta = 17$	$\Delta = 30$
Linear	-0.57	-0.49
Poly	-1.95	-1.40
Rational	-1.14	-1.33
loc(1)	-1.48	-1.24
loc(2)	-1.89	-0.42
Radial BF	-1.97	-1.60
MLP	-2.00	-1.50
FIR Network	-2.31	-1.79
DRNN	-2.33	-2.12

Table 2: Comparison of log-normalized single-step prediction errors (NMSE) for the Mackey-Glass equations.

The configuration of the DRNN model is $1 \times 7 \times 1$ and the filter orders is $4:2:0$ where 4,2 and 0 denote respectively the maximum delay of the recurrent connections linking: the input neuron to the 7 hidden neurons, the 7 hidden neurons themselves (recursively), the hidden neurons to the output neuron.

The network was trained on the first 500 points. The selection of this configuration is based on the grounds of the same number of parameters with the (feedforward) FIR network, of dimension $1 \times 15 \times 1$ nodes and $8:2$ in taps, used by Wan [28]. The DRNN and the FIR networks have respectively 197 and 196 ajustable weights. Yet, both networks have distinct time-delay vector size at their input, the FIR network is fed with 9 delayed values whereas the DRNN is fed with only 5 values.

The resulting log-normalized single-step errors for the subsequent 1500 points are shown in Table 2 along with the results of the FIR network used in [28] and other regresion methods [7]. A noticeable improvement is observed for the DRNN with $\Delta = 30$. A more challenging problem is the iterated prediction. The DRNN is compared to the FIR: both networks receive no input past the point 500. Fig. 7 shows a plot of the log of the *average* normalized MSE. The linear slopes confirm the iterated prediction exponential divergence from the true values. It is interesting to note that the DRNN exhibits a steeper slope than the FIR network, indicating in prediction worse on the long term, and better in the short term. This is not surprising though, and results from a different delay vector size at the input. The less information presented at the input, the more perturbated is the input/output mapping. From the 6-th iteration onwards, the DRNN operates in closed loop whereas the FIR network is still fed with the true past values.

6.2 Hénon Map

In this second example, consider the bi-dimensional system:

$$\begin{aligned} x(k+1) &= 1.0 - 1.4x^2(k) + 0.3y(k) \\ y(k+1) &= x(k) \end{aligned} \tag{24}$$

The iterated time series is illustrated in Fig. 9. The phase-spot, shown in Fig. 15, reveals a remarkable structure called a *strange attractor* of dimension $D = 1.26$ [18]. It is the region in which the trajectory of states is confined to. Increased magnification of the attractor would reveal ever finer detail in a fractal like geometry.

A DRNN of dimension $1 \times 10 \times 1$ with filter orders $3 : 1 : 0$ was trained on 5000 data. Single step NMSE are shown in Table 3. Again, comparisons are shown with a FIR network of dimension $1 \times 12 \times 12 \times 1$ with filter orders $3 : 1 : 1$, trained on the same number of data. The DRNN slightly outperforms the FIR with fewer parameters.

model	NMSE
linear model	0.874
RN FIR	0.0017
DRNN	0.0012

Table 3: Log-normalized single-step mean squared prediction errors (NMSE) for the Hénon map.

As can be seen in Fig. 10, the DRNN prediction are remarquably accurate starting out, but then diverge after 10 time steps. Similar remarks apply to the outputs of FIR network. Again, this divergence is an avoidable consequence of the nature of chaos. When the network is iterated into the future, however, the reconstructed attractor is fairly similar to the original, see Fig. 16.

The iterated network signal accurately represent the system dynamics and exhibit no noise like corruption. The attractor is easily captured.

6.3 Lorenz equations

A Lorenz system [17, 18] is described by the solution of the three simultaneous differential equations:

$$\begin{aligned}
\frac{dx}{dt} &= -\sigma x + \sigma y \\
\frac{dy}{dt} &= -xz + rx - y \\
\frac{dz}{dt} &= xy - bz
\end{aligned} \tag{25}$$

A projection of the trajectory in the x-z plane for the parameters values $\sigma = 10, r = 28, b = 8/3$ is shown in Fig. 17. A DRNN model of dimension $1 \times 10 \times 2$ nodes with filtre taps $30 : 1 : 0$ was trained on 4000 observations of only the x state sampled at period of 0.05 seconds. Comparisons are made with a FIR network of dimensions $1 \times 12 \times 12 \times 2$ nodes with $2 : 5 : 5$ taps. In both cases, the network had 2 outputs corresponding to estimates of both the x and z states. The output of the network was a prediction of both x and z.

	NMSE	
model	x	z
linear model	0.036	0.090
FIR	0.0070	0.0095
DRNN	0.0055	0.0078

Table 4: Log-normalized single-step mean squared prediction errors (NMSE) for the Lorenz equations.

The result for the Log-normalized MSE for single step prediction during the subsequent 1500 points is given in Table 4. In Fig. 18 the iterated network prediction $\hat{x}(t)$ vs. $\hat{z}(t)$ is shown. Both networks were able to capture the underlying dynamics. Fig. 19 shows the plot in log scale of the number of back-propagations recorded during training. Again, linearity is patent.

6.4 Ikeda Map

For the last example, we consider a mapping in the complex plane corresponding to the plane-wave interactivity in an optical ring laser [11]:

$$z(k+1) = a + bz(k)\exp(i[\phi - c/(1 + |z(k)|^2)]) \tag{26}$$

with $a = 1.0, b = 0.9, c = 6.0$ et $\phi = 0.4$. The real and imaginery parts of the *Ikeda* series are plotted in Fig. 11 and the phase-plot of $\Re[z]$ vs. $\Im[z]$ is shown in Fig. 20. Again we formulate a state-estimation approach in which only the real part is fed to the network. Output correrspond to both real and imaginary estimates making the prediction task far more challenging.

	NMSE	
model	x	z
Linear model	0.640	0.715
FIR	0.0080	0.0150
DRNN	0.0063	0.0134

Table 5: Log-normalized single-step prediction errors (NMSE) for the Ikeda Map.

A DRNN of dimensions $1 \times 15 \times 2$ with filter orders $5:1:0$ is trained on 10,000 points. The single step NMSE achieved on the test set is shown in Table 5. The network was then iterated closed loop thousands of time steps to form its corresponding phase-plot shown in Fig. 21. Again, it is clear from the plot that the network has captured much of the underlying dynamics.

The gradient decay is illustrated in Fig. 22. Given a precision of 10^{-2} for the gradient, 20 retro-propagations on the average suffice before stopping occurs.

7 Conclusion

In this paper, we have shown that dynamical recurrent neural networks are valuable for modeling general nonlinear dynamical systems. Modeling connections as autoregressive filters in a fully-recurrent configuration is particularly well suited for temporal tasks insofar as it endows the model with a memory of its previous states.

An efficient adaptation procedure, called temporal recurrent back-propagation, was used to adapt DRNN models. The algorithm, briefly presented, exhibits minimal computational and storage requirements.

A number of experiments on the sunspots and on deterministic chaotic processes were carried out to confirm the exponential gradient decay backwards in time and to illustrate the ability of DRNN models to infer an internal representation of the nonlinear processes from the observation of a subset of the state variables.

References

[1] A. Aussem, F. Murtagh and M. Sarazin, "Dynamical recurrent neural networks and pattern recognition methods for time series prediction: application to seeing and temperature forecasting in the context of ESO's VLT Astronomical Weather Station," *Vistas in Astronomy* **38**, pp. 357-374, 1995.

[2] A. Aussem, F. Murtagh and M. Sarazin, "Dynamical recurrent neural networks - towards environmental time series prediction," *International Journal on Neural Systems*, Vol. 6, No. 2, pp. 145-170, 1995.

[3] A. Aussem, F. Murtagh and M. Sarazin, "Fuzzy astronomical seeing nowcasts with a dynamical and recurrent connectionist network," *Neurocomputing*, **12**, No. 4, pp. 359-373, 1996.

[4] A. Aussem et F. Murtagh, "Combining Neural Network Forecasts on Wavelet-Transformed Time Series," *Connection Science* **9**, No. 1, pp.113-121, 1997.

[5] A. Aussem, "Nonlinear modeling of chaotic processes with dynamical recurrent neural networks," *Proceedings of the Int. Conf. on Neural Networks and Their Applications NEURAP'98*, Marseille, France, pp. 425-433, March 1998.

[6] G.E. Box and G.M. Jenkins, *Time series analysis: forecasting and control*, Holden-Day, San Francisco, 1976.

[7] M. Casdagli, "Nonlinear prediction of chaotic time series," *Physica D* **35**, pp. 335-356, 1989.

[8] J.T. Connor, R.D. Martin, an L.E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE Trans. on Neural Networks* **5**, pp. 240-254, 1994.

[9] J.L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, pp. 179-211, 1990.

[10] P. Frasconi, M. Gori, et G. Soda, "Local feedback multilayered networks," *Neural Computation* **4**, pp. 120-130, 1992.

[11] S. Hammel, C. Jones et J.V. Moloney, "Global dynamical systems, and bifurcations of vector fields," *J. Opt. Soc. Am.* **B-2**, pp. 552, 1985.

[12] S. Hochreiter and J. Schmidhuber, "Long short term memory," *Neural Computation* **9**, pp. 1735-1780, 1997.

[13] K. Lang and G. Hinton, "Time-Delay Neural Network architecture for speech recognition," Tech. rep. CMU-CS-88-152, Carnegie-Mellon University, Pittsburgh, 1988.

[14] R. Leighton and B. Conrad, "The autoregressive backpropagation algorithm," *Proceedings of the International Joint Conference on Neural Networks*, 1991.

[15] I.J. Leontaritis, S. Billings, "Input-output parametetric models for non-linear systems," *Int. J. Control* **41**, pp. 303-344, 1985.

[16] A.U. Levin, K.S. Narendra, "Identification using feedforward networks," *Neural Computation* **7**, pp.349-357, 1995

[17] E.N. Lorenz, "Deterministic nonperiodic flow," *J. Atmos. Sci.* **20**, pp. 130-141, 1963.

[18] E. Ott, *Chaos in Dynamical Systems.* Cambridge University Press, 1993.

[19] K.S. Narendra et K. Parthasarathy, "Gradient methods for the optimization of linear control systems," *IEEE Trans. on Neural Networks*, pp. 252-262, 1991.

[20] S.W. Piché, "Steepest descent algorithms for neural network controllers and filters," *IEEE Trans. on Neural Networks*, vol. 5, pp. 198-212, 1994.

[21] F.J. Pineda, "Generalization of back-propagation to recurrent neural networks," *Physical Review Letters* **59**, pp. 2229-2232, 1987.

[22] M.B. Priestley, *Spectral Analysis and Time Series*, Academic Press, 1981.

[23] I. Rivals and L. Personnaz, "Black-box modeling with state-space neural networks," in *Neural Adaptive Control Technology,* I.R. Zbokowski and K.J. Hunt Eds., World Scientific, 1995.

[24] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, D.E. Rumelhart and J.L. McClelland, Eds., Cambridge, MA: MIT Press, Bradfords Books, vol. 1, pp. 318-362, 1986.

[25] J. Schmidhuber, “A fixed size storage O(n^3) time complexity learning algorithm for fully recurrent continually running networks,” *Neural Computation* **4**, pp. 243-248, 1992.

[26] B. Srinivasan, U.R. Prasad and N.J. Rao, “Back propagation through adjoints for the identification of nonlinear dynamic systems using recurrent neural networks,” *IEEE Trans. on Neural Networks*, vol. 5, pp. 213-228, 1994.

[27] H. Tong, *Non Linear Time Series*, Clarendon Press Oxford, 1990.

[28] E.A. Wan, *Finite Impusle Response Neural Networks with Applications in Time Series Prediction.* Ph.D. Thesis, Stanford University, November, 1993.

[29] A.S. Weigend, D.E. Rumelhart and B.A. Huberman, “Predicting the future: a connectionist approach,” *International Journal of Neural Systems*, vol 1, pp. 195-220, 1990.

[30] P. Werbos, *Beyond regression: new tools for prediction and analysis in behavioral sciences*, Ph.D. thesis, Harvard University, Cambridge, MA, August 1974.

[31] R.J. Williams and D. Zipser, “Experimental analysis of the real-time recurrent learning algorithm,” *Connection Science*, vol. 1, pp. 87-111, 1989.

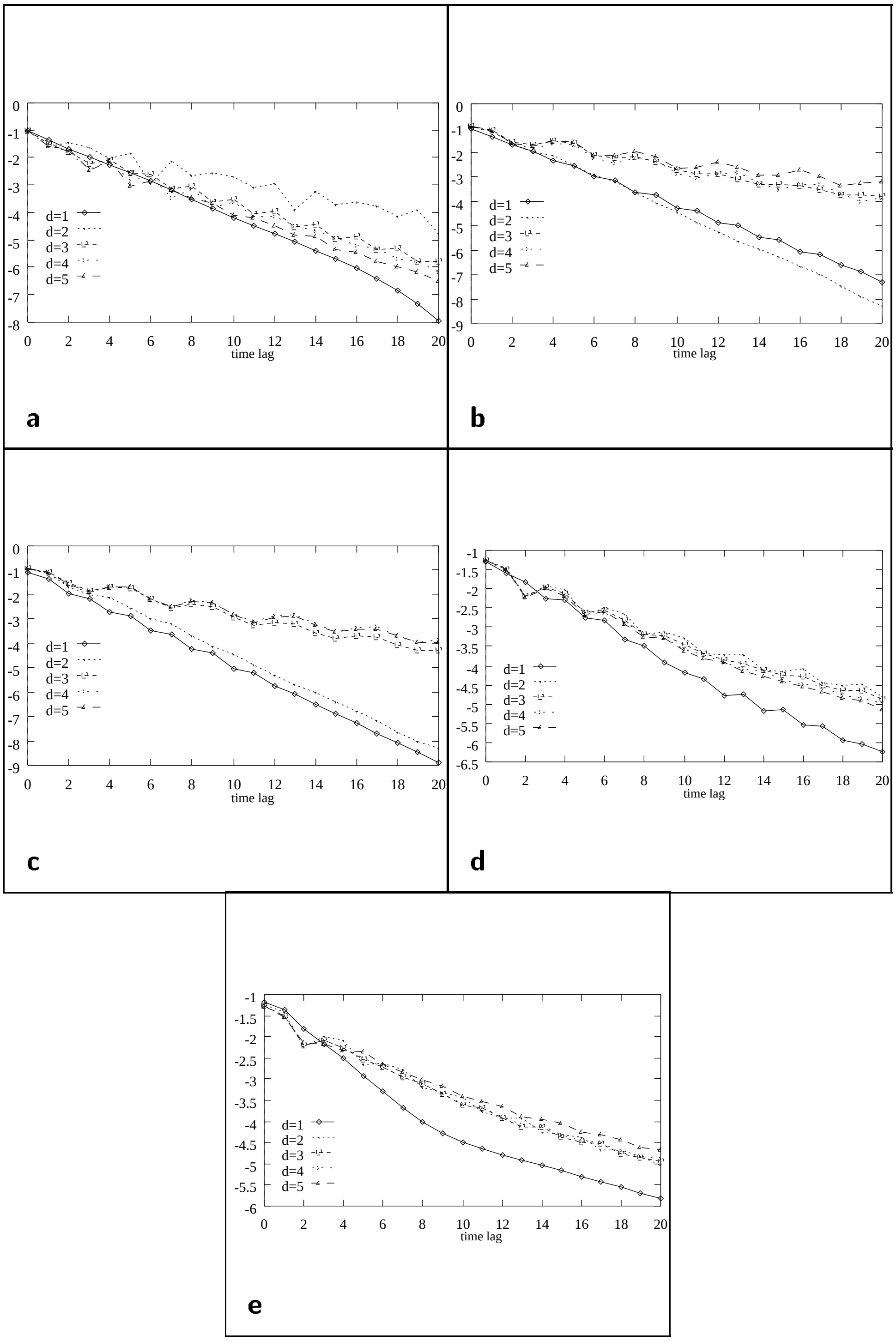

Figure 1: Sunspots : $\|\mathbf{y}_{n-j}^{n}\|_{\infty}$ are averaged over the overall training set and plotted against j for the different network configurations of the type $1 \times N \times 1$ with FIR connection order $D = 1, \ldots, 5$. (a) $N = 1$, (b) $N = 2$, (c) $N = 3$, (d) $N = 4$, (e) $N = 5$.

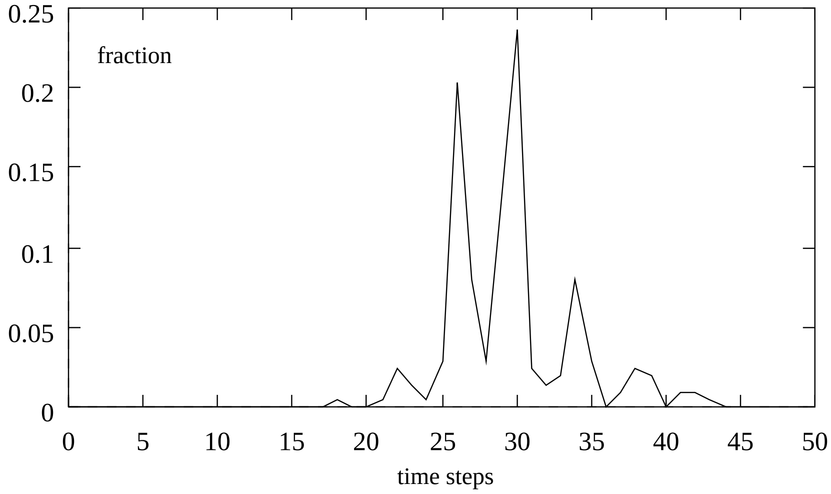

Figure 2: Histogram of the average number of time-layers required by the TRBP training algorithm in the adjoint network.

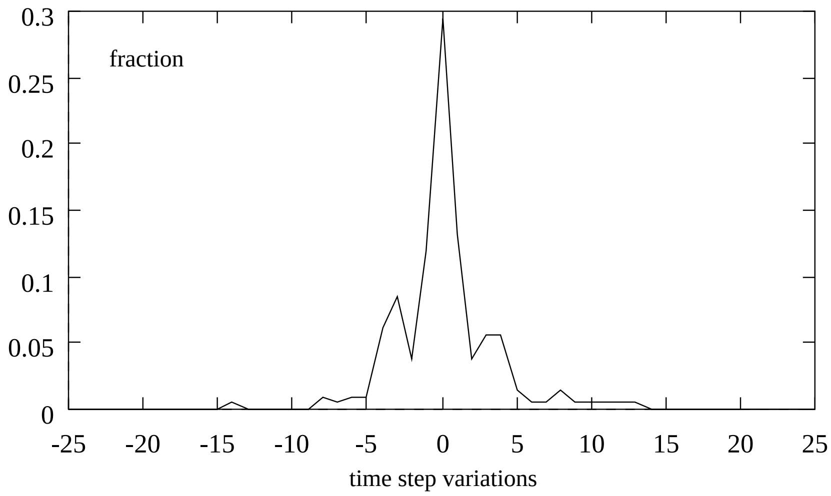

Figure 3: Histogram of the average variation of the number of time-layers in the adjoint between successive iterations.

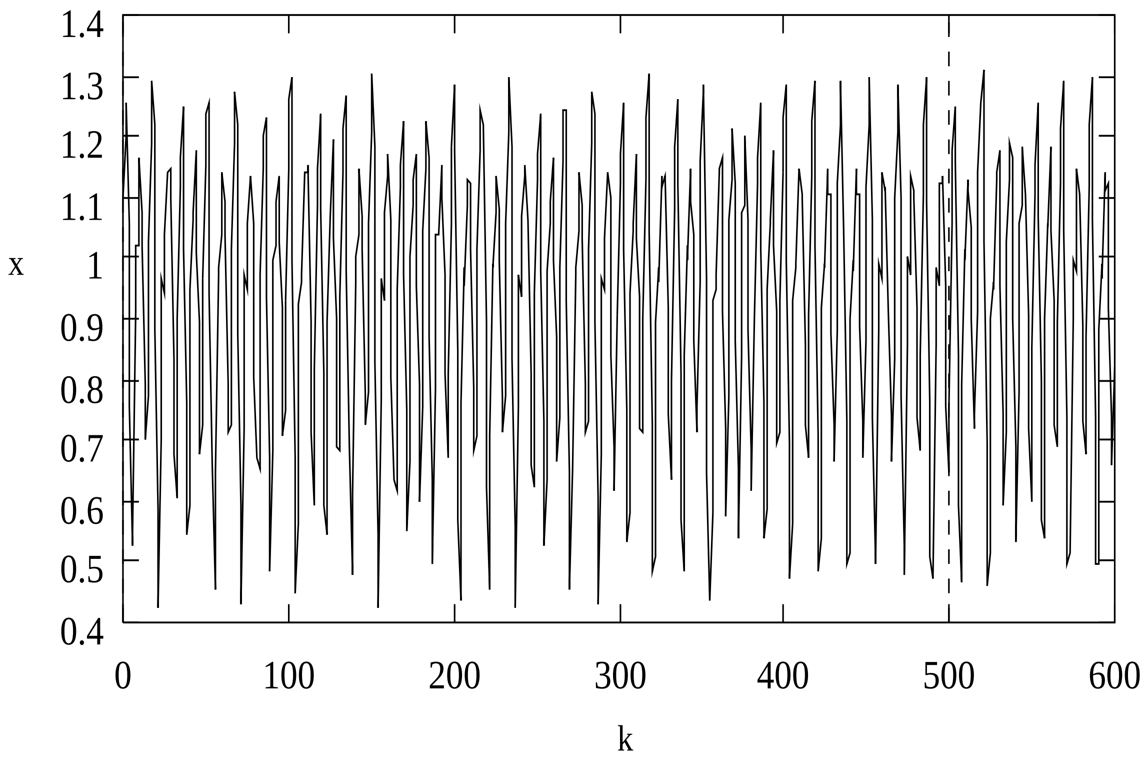

Figure 4: Mackey Glass series with $\Delta = 17$.

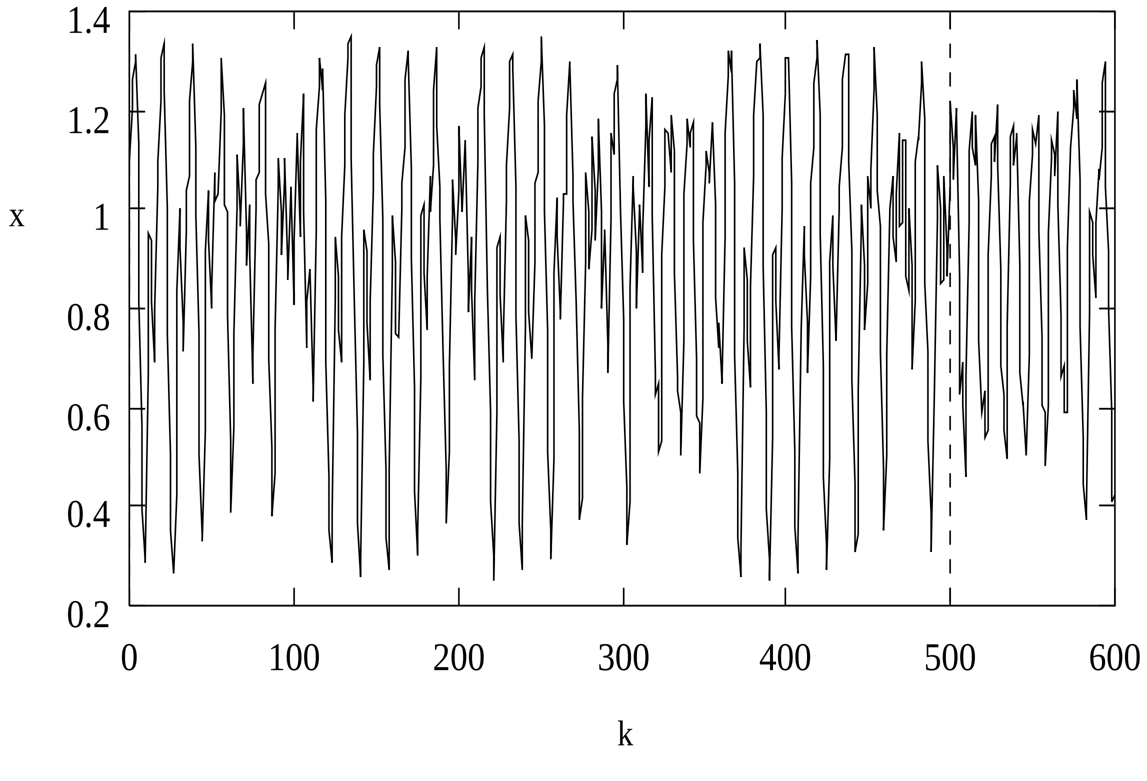

Figure 5: Mackey Glass series with $\Delta = 30$.

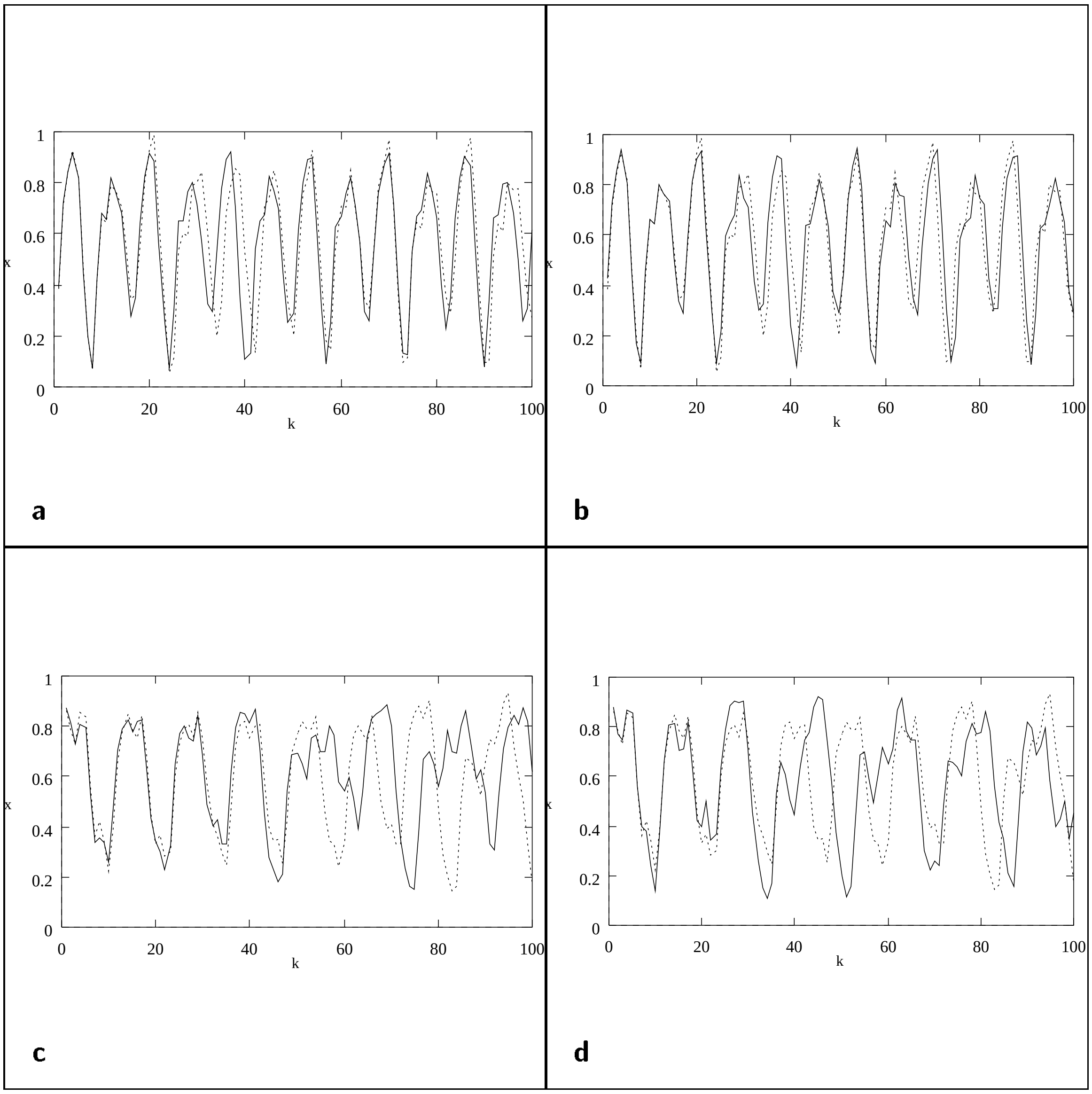

Figure 6: Iterated predictions for Mackey-Glass series: a) FIR network, $\Delta = 17$, b) DRNN, $\Delta = 17$, c) FIR network , $\Delta = 30$, d) DRNN, $\Delta = 30$. In plain line, the prediction; in dotted line, the actual series.

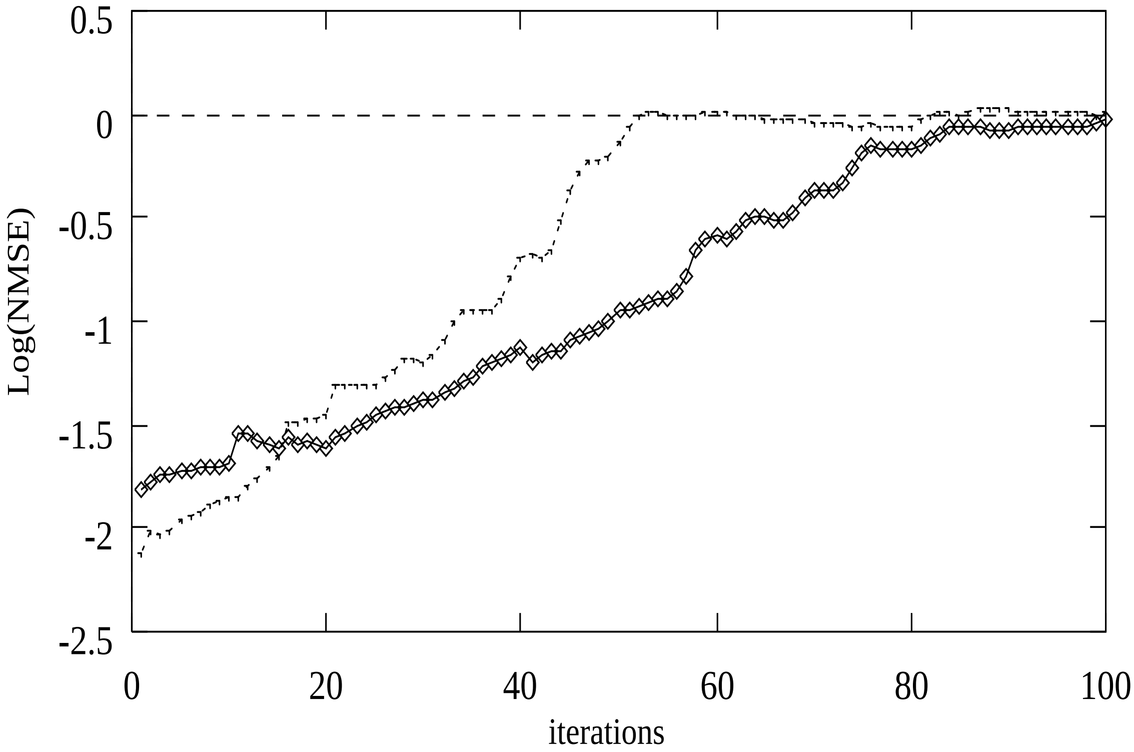

Figure 7: Mackey-Glass: iterated log average normalized mean square errors. Dashed curve is the DRNN output, the ficker curve is the FIR Network output.

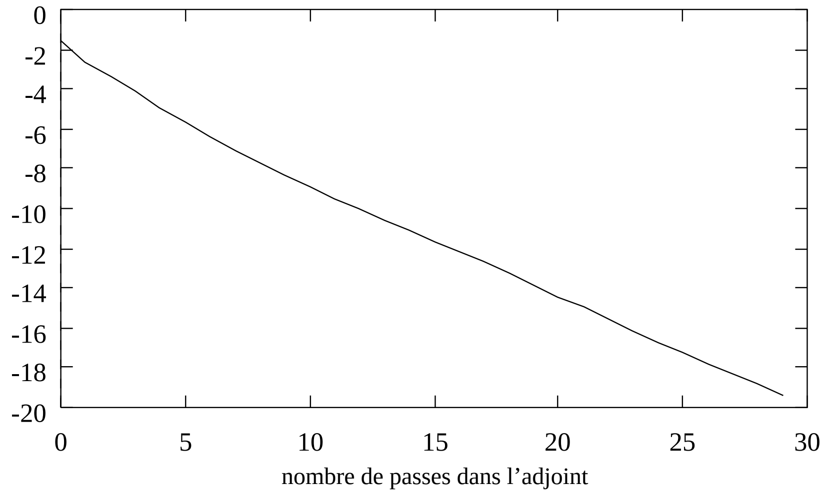

Figure 8: Mackey-Glass: illustration of the gradient decay in log scale $< \mathrm{Log}_{10}(\|\mathrm{y}^n_{n-j}\|_\infty >$ versus the number of back-propagations j in the unfolded adjoint.

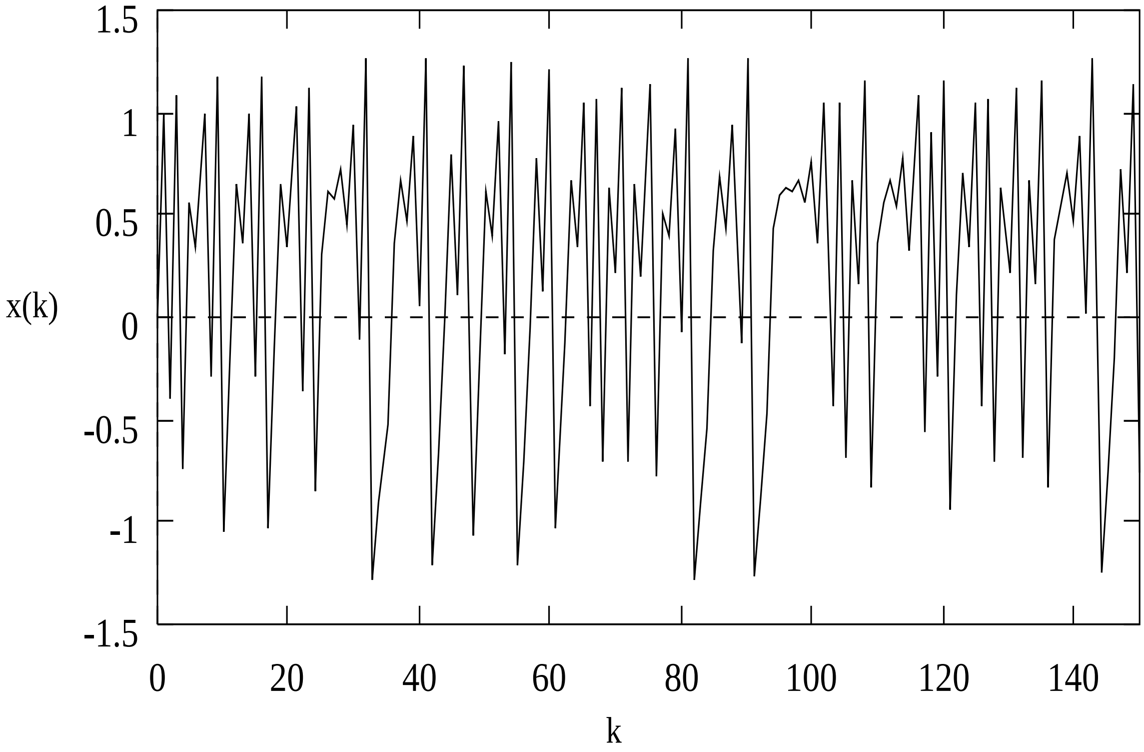

Figure 9: Hénon series.

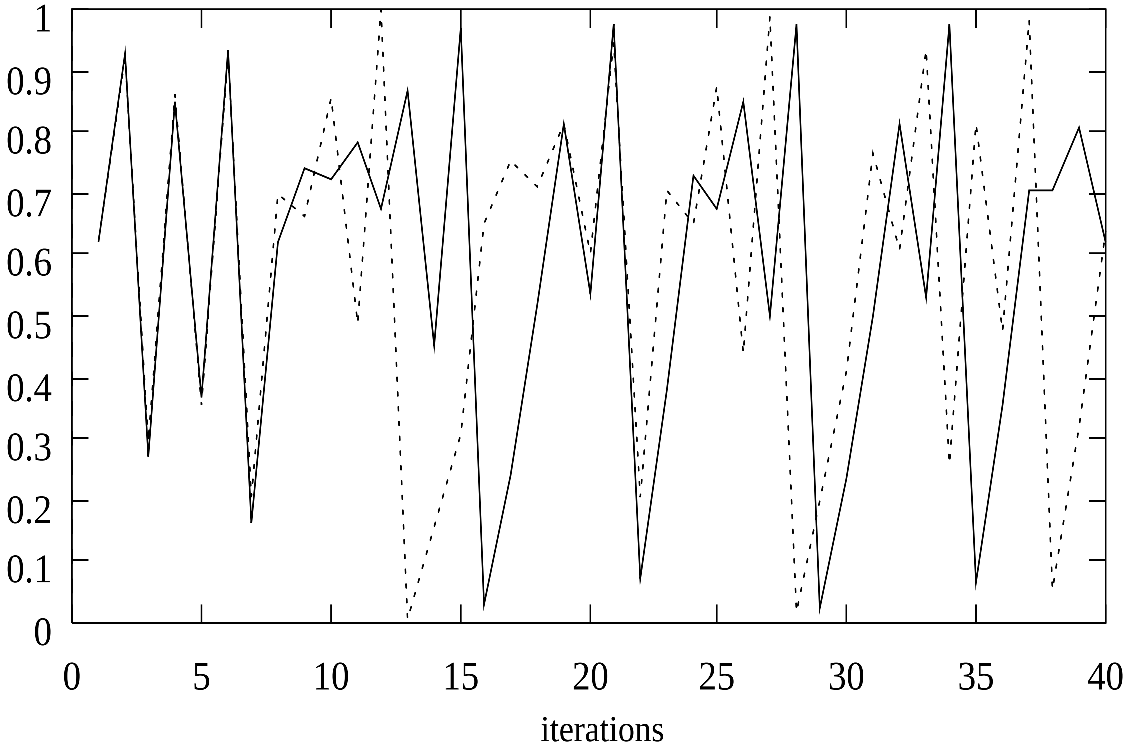

Figure 10: Iterated predictions for the Hénon map.

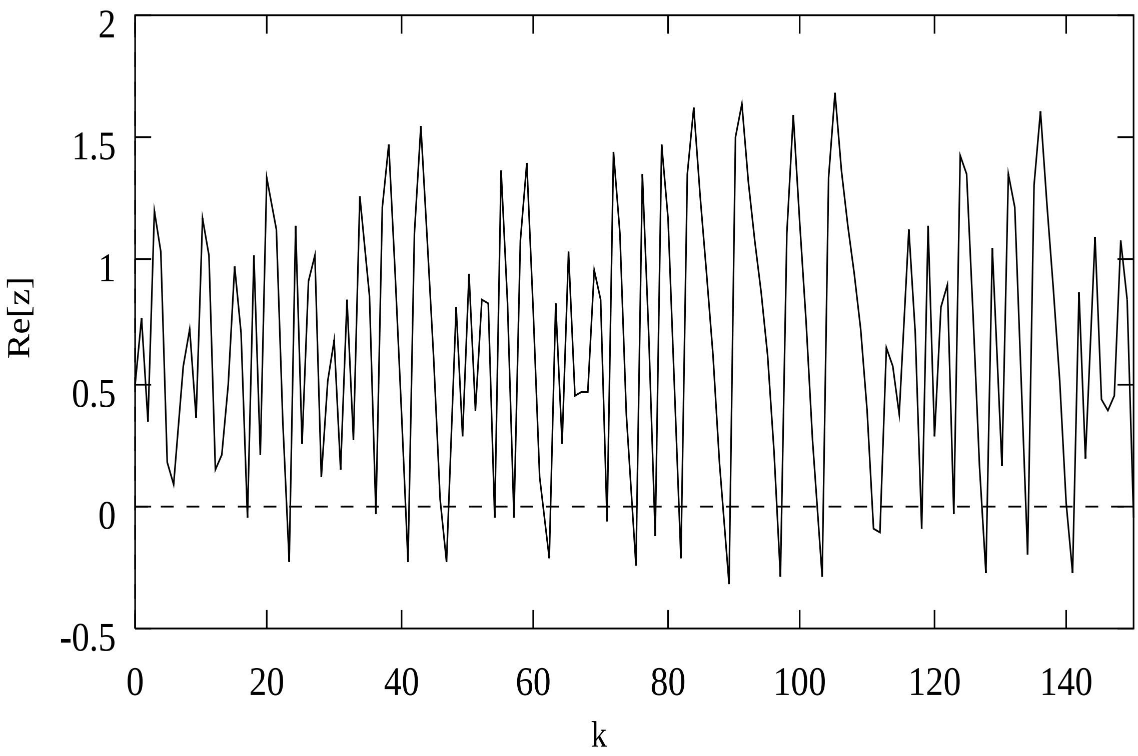

Figure 11: Imaginary series corresponding to the Ikeda Map.

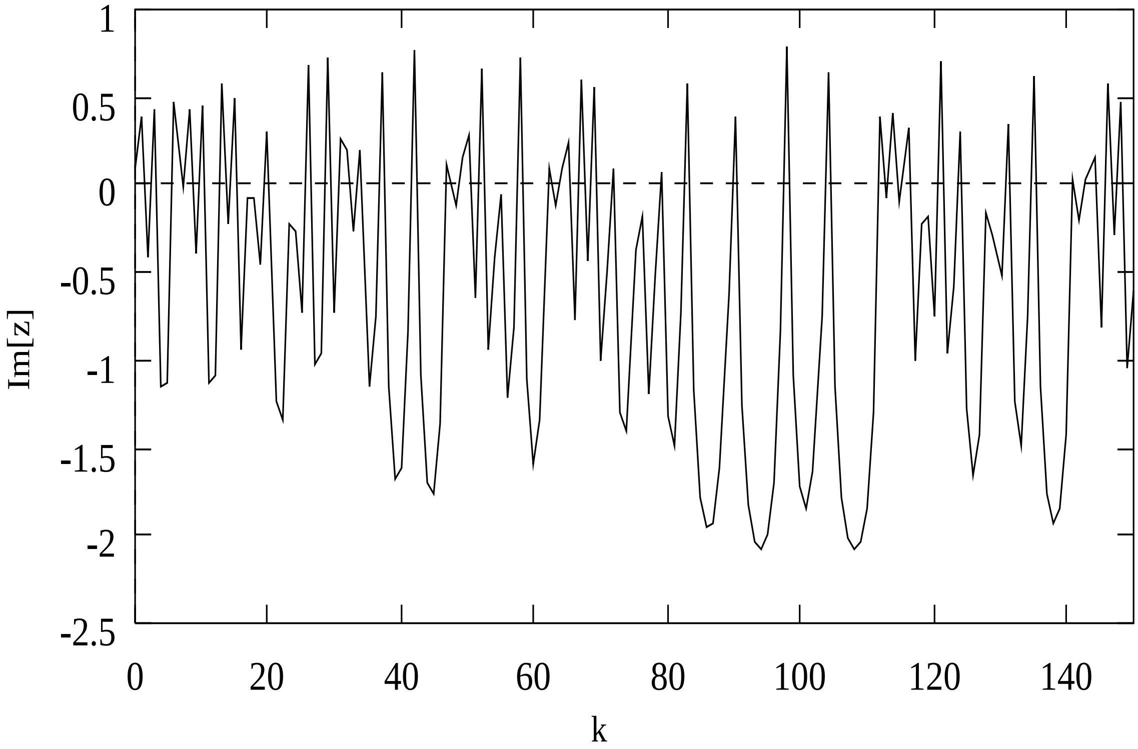

Figure 12: Real series corresponding to the Ikeda Map.

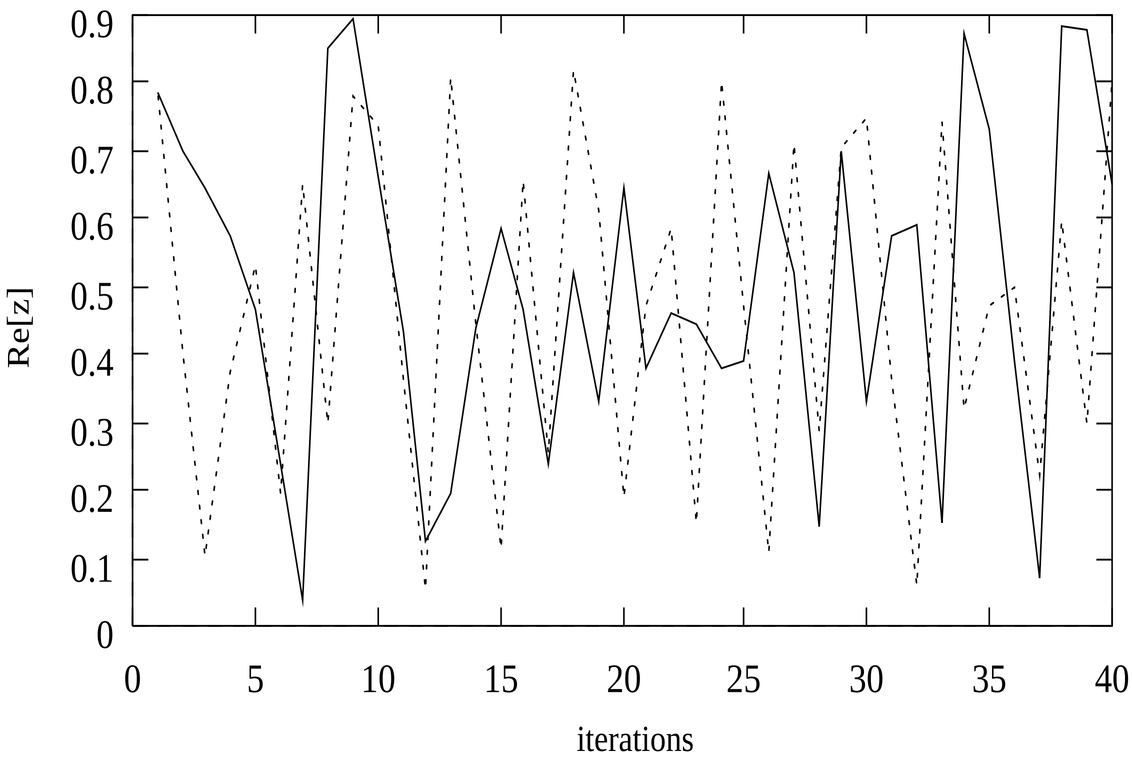

Figure 13: Real iterated series corresponding to the Ikeda Map.

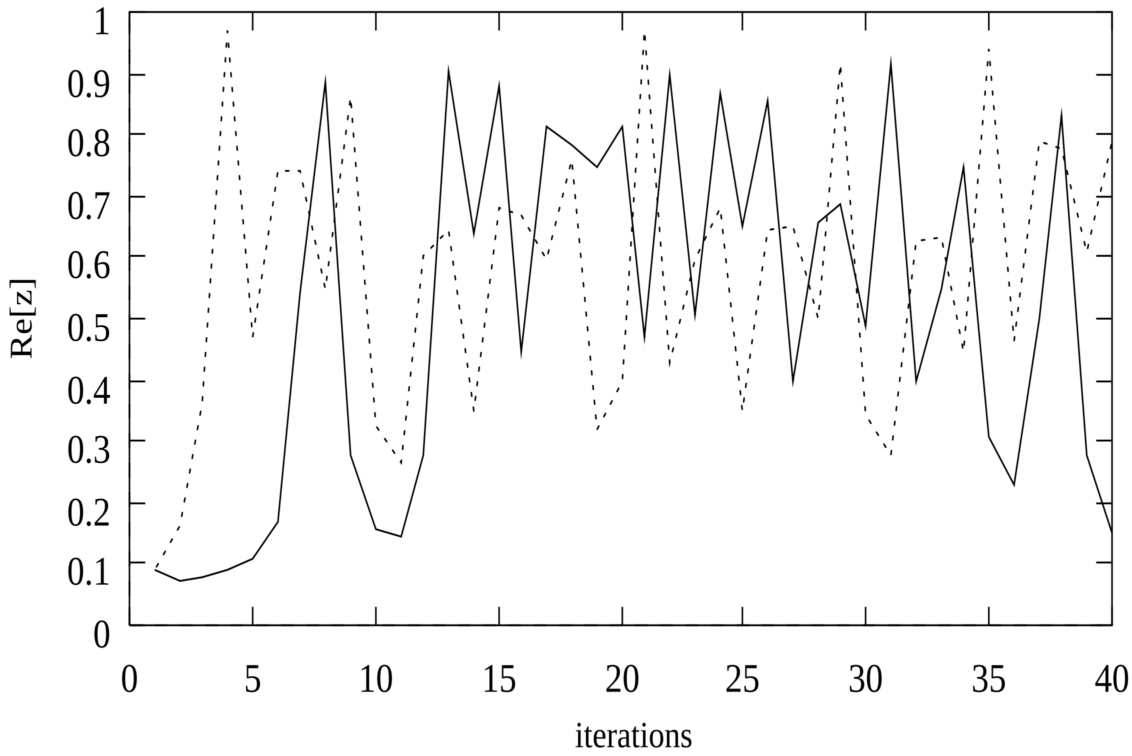

Figure 14: Imaginary iterated series corresponding to the Ikeda Map.

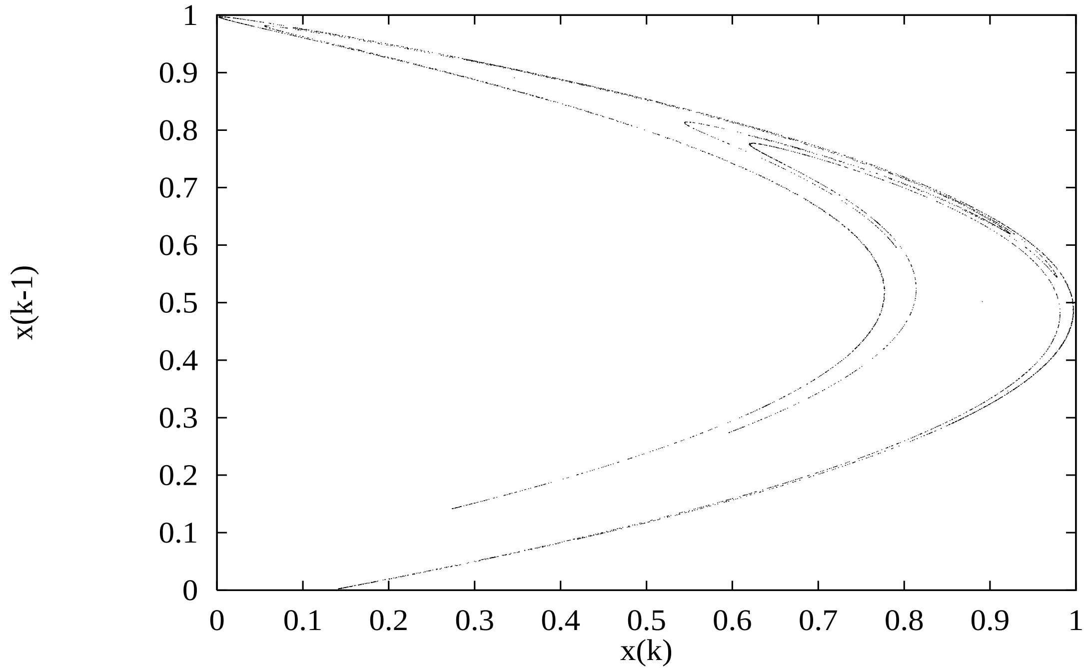

Figure 15: Hénon Attractor: a phase plot of $x(k-1)$ vs. $x(k)$.

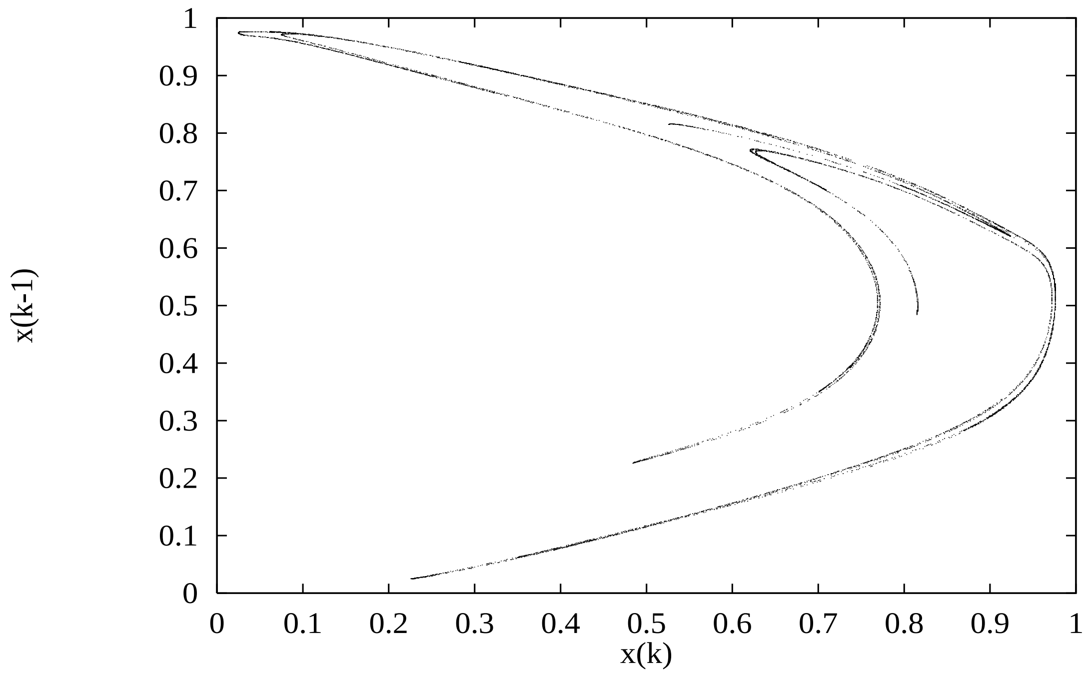

Figure 16: Hénon reconstructed by iterating a DRNN $1 \times 10 \times 1$ model of order $3:1:0$.

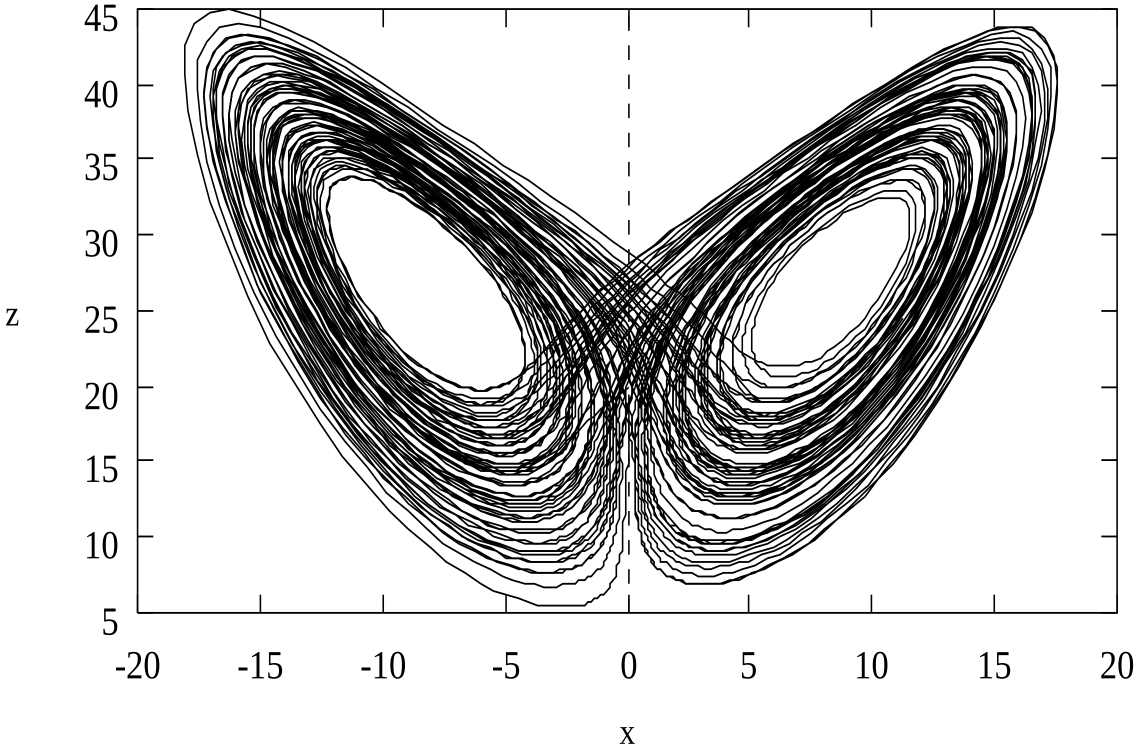

Figure 17: Lorenz attractor: a phase plot of x vs. z

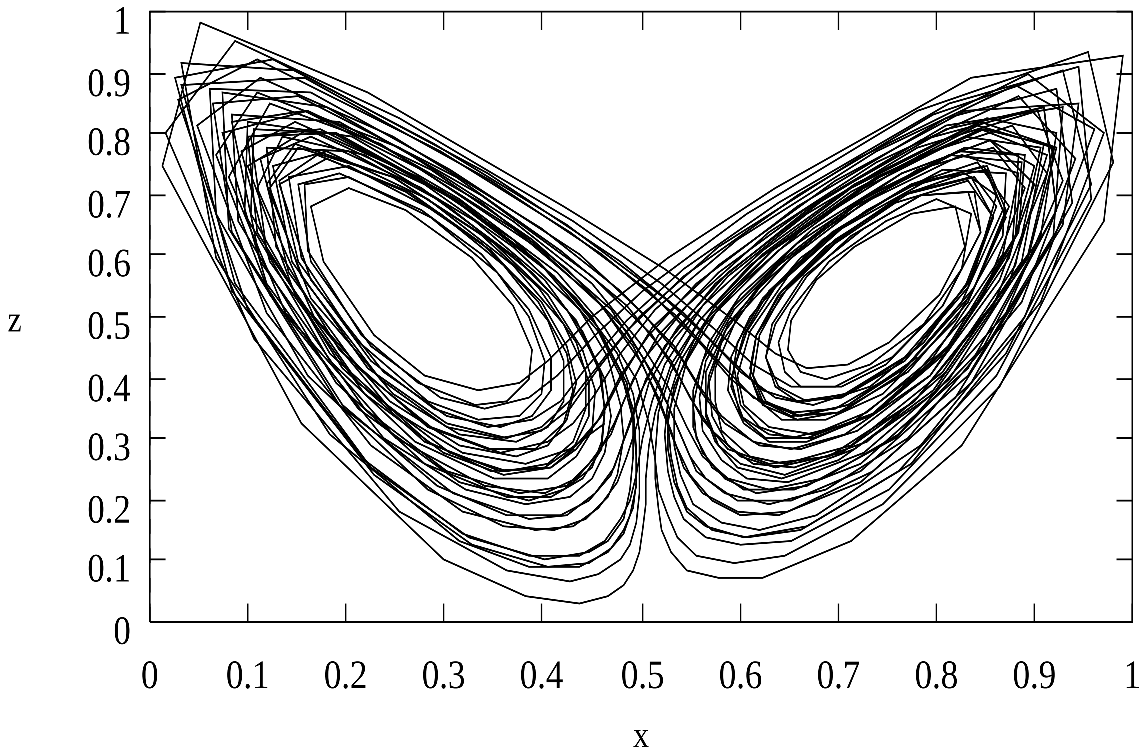

Figure 18: Lorenz attractor reconstructed from the DRNN iterated outputs: a phase plot of x vs. z

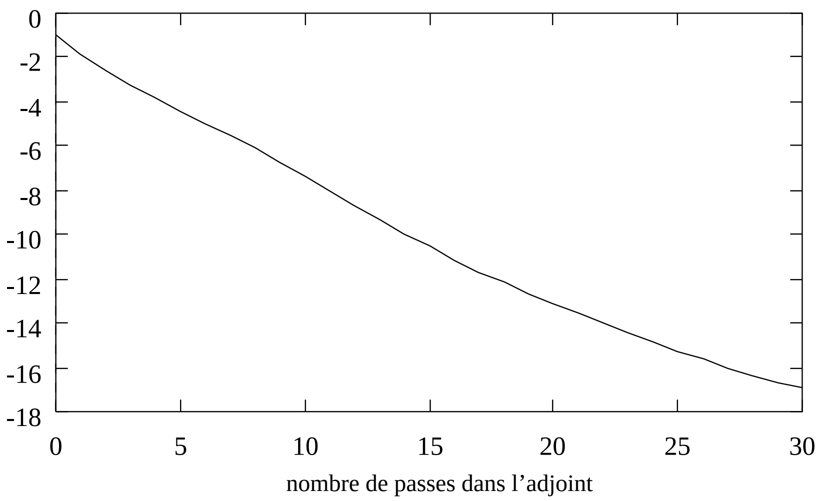

Figure 19: Lorenz: illustration of the gradient decay in log scale $< \mathrm{Log}_{10}(\|\mathrm{y}^n_{n-j}\|_\infty >$ versus the number of back-propagations j in the unfolded adjoint.

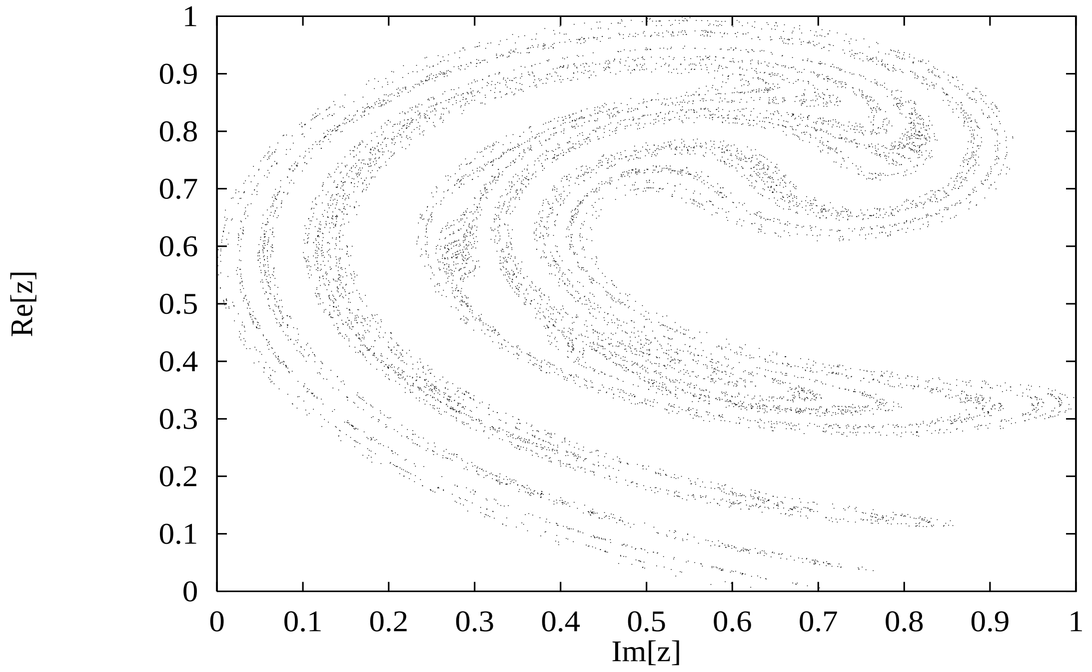

Figure 20: Ikeda attractor: $Re[z]$ vs. $Im[z]$.

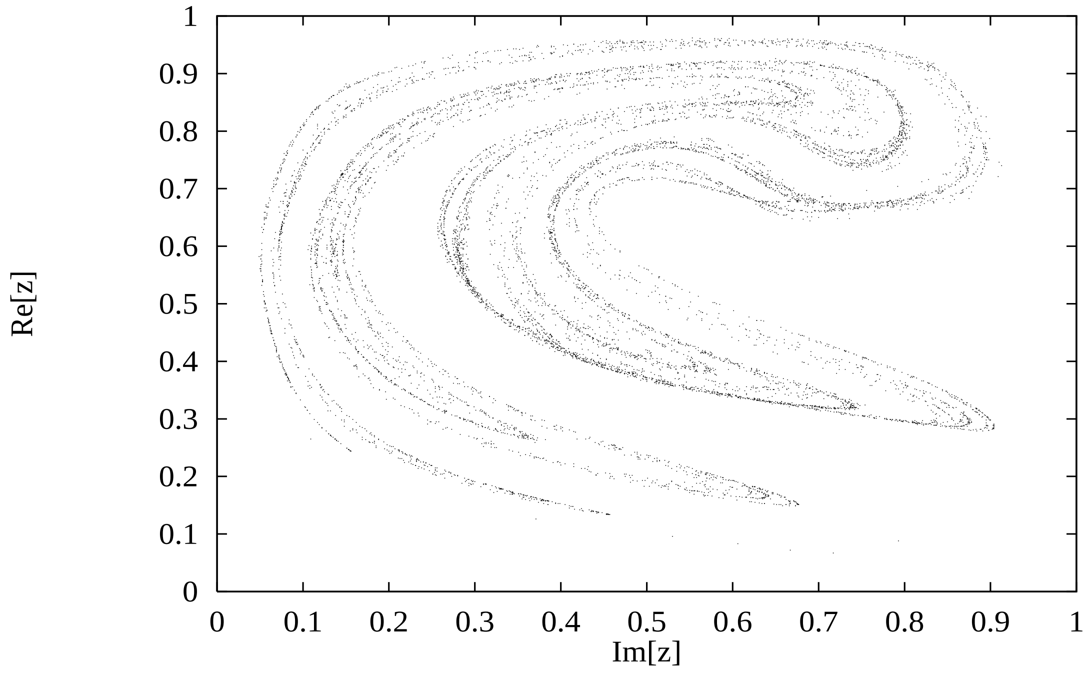

Figure 21: Reconstruction of Ikeda attractor by iterating the DRNN predictions: $Re[z]$ vs. $Im[z]$.

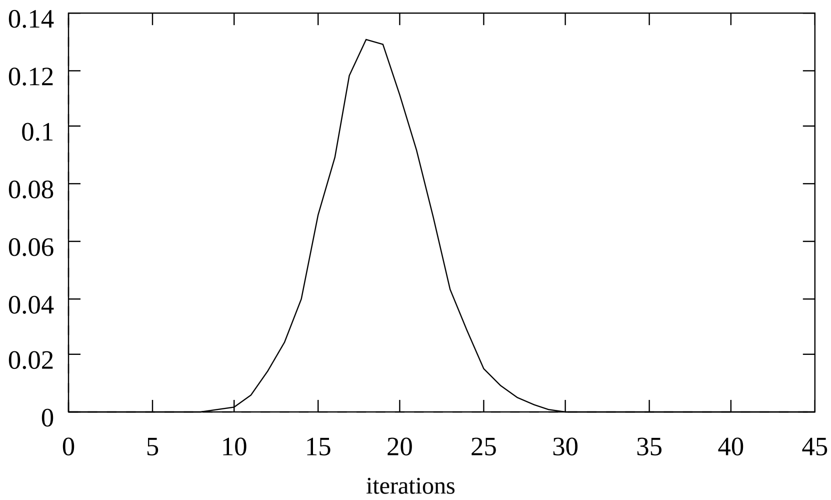

Figure 22: Ikeda: histogram of the average number of back-propagations in the unfolded adjoint.