Practical implementation of nonlinear time series methods: The TISEAN package

Rainer Hegger and Holger Kantz[a]
Max Planck Institute for Physics of Complex Systems, Nöthnitzer Str. 38, D-01187 Dresden, Germany

Thomas Schreiber
Department of Physics, University of Wuppertal, D-42097 Wuppertal, Germany

We describe the implementation of methods of nonlinear time series analysis which are based on the paradigm of deterministic chaos. A variety of algorithms for data representation, prediction, noise reduction, dimension and Lyapunov estimation, and nonlinearity testing are discussed with particular emphasis on issues of implementation and choice of parameters. Computer programs that implement the resulting strategies are publicly available as the TISEAN software package. The use of each algorithm will be illustrated with a typical application. As to the theoretical background, we will essentially give pointers to the literature. © *1999 American Institute of Physics.* [S1054-1500(99)00802-2]

Nonlinear time series analysis is becoming a more and more reliable tool for the study of complicated dynamics from measurements. The concept of low-dimensional chaos has proven to be fruitful in the understanding of many complex phenomena despite the fact that very few natural systems have actually been found to be low-dimensional deterministic in the sense of the theory. In order to evaluate the long term usefulness of the nonlinear time series approach as inspired by chaos theory, it will be important that the corresponding methods become more widely accessible. This paper, while not a proper review on nonlinear time series analysis, tries to make a contribution to this process by describing the actual implementation of the algorithms, and their proper usage. Most of the methods require the choice of certain parameters for each specific time series application. We will try to give guidance in this respect. The scope and selection of topics in this article, as well as the implementational choices that have been made, correspond to the contents of the software package TISEAN which is publicly available from http://www.mpipks-dresden.mpg.de/~tisean. In fact, this paper can be seen as an extended manual for the TISEAN programs. It fills the gap between the technical documentation and the existing literature, providing the necessary entry points for a more thorough study of the theoretical background.

I. INTRODUCTION

Deterministic chaos as a fundamental concept is by now well established and described in a rich literature. The mere fact that simple deterministic systems generically exhibit complicated temporal behavior in the presence of nonlinearity has influenced thinking and intuition in many fields. However, it has been questioned whether the relevance of chaos for the understanding of the time evolving world goes beyond that of a purely philosophical paradigm. Accordingly, major research efforts are dedicated to two related questions. The first question is if chaos theory can be used to gain a better understanding and interpretation of observed complex dynamical behavior. The second is if chaos theory can give an advantage in predicting or controlling such a time evolution. Time evolution as a system property can be measured by recording the time series. Thus, nonlinear time series methods will be the key to the answers of the above questions. This paper is intended to encourage the explorative use of such methods by a section of the scientific community which is not limited to chaos theorists. A range of algorithms has been made available in the form of computer programs by the TISEAN project.[1] Since this is fairly new territory, unguided use of the algorithms bears considerable risk of wrong interpretation and unintelligible or spurious results. In the present paper, the essential ideas behind the algorithms are summarized and pointers to the existing literature are given. To avoid excessive redundancy with the text book[2] and the recent review,[3] the derivation of the methods will be kept to a minimum. On the other hand, the choices that have been made in the implementation of the programs are discussed more thoroughly, even if this may seem quite technical at times. We will also point to possible alternatives to the TISEAN implementation.

Let us at this point mention a number of general references on the subject of nonlinear dynamics. At an introductory level, the book by Kaplan and Glass[4] is aimed at an interdisciplinary audience and provides a good intuitive understanding of the fundamentals of dynamics. The theoretical framework is thoroughly described by Ott,[5] but also in the older books by Bergé *et al.*[6] and by Schuster.[7] More advanced material is contained in the work by Katok and Hasselblatt.[8] A collection of research articles compiled by Ott *et al.*[9] covers some of the more applied aspects of chaos, like synchronization, control, and time series analysis.

[a]Electronic mail: kantz@mpipks-dresden.mpg.de

Nonlinear time series analysis based on this theoretical paradigm is described in two recent monographs; one by Abarbanel[10] and one by Kantz and Schreiber.[2] While the former volume usually *assumes* chaoticity, the latter book puts some emphasis on practical applications to time series that are not manifestly found, nor simply assumed to be, deterministic chaotic. This is the rationale we will also adopt in the present paper. A number of older articles can be seen as reviews, including Grassberger *et al.*,[11] Abarbanel *et al.*,[12] as well as Kugiumtzis *et al.*[13,14] The application of the nonlinear time series analysis to real world measurements, where determinism is unlikely to be present in a stronger sense, is reviewed in Schreiber.[3] Apart from these works, a number of conference proceedings volumes are devoted to a chaotic time series, including Refs. 15–19.

A. Philosophy of the TISEAN implementation

A number of different people have been credited for the saying that every complicated question has a simple answer which is wrong. Analyzing a time series with a nonlinear approach is definitely a complicated problem. Simple answers have been repeatedly offered in the literature, quoting numerical values for attractor dimensions for any conceivable system. The present implementation reflects our skepticism against such simple answers which are the inevitable result of using black box algorithms. Thus, for example, none of the ''dimension'' programs will actually print a number which can be quoted as the estimated attractor dimension. Instead, the correlation sum is computed and basic tools are provided for its interpretation. It is up to the scientist who does the analysis to put these results into their proper context and to infer what information she or he may find useful and plausible. We should stress that this is not simply a question of error bars. Error bars do not tell about systematic errors and neither do they tell if the underlying assumptions are justified.

The TISEAN project has emerged from work of the involved research groups over several years. Some of the programs are in fact based on the code published in Ref. 2. Nevertheless, we still like to see it as a starting point rather than a conclusive step. First of all, nonlinear time series analysis is still a rapidly evolving field, in particular with respect to applications. This implies that the selection of topics in this article and the selection of algorithms implemented in TISEAN are highly biased towards what we know now and found useful so far. But even the well established concepts like dimension estimation and noise reduction leave considerable room for alternatives to the present implementation. Sometimes this resulted in two or more concurring and almost redundant programs entering the package. We have deliberately not eliminated these redundancies since the user may benefit from having a choice. In any case it is healthy to know that for most of the algorithms the final word has not been spoken yet—nor is ever to be.

While the TISEAN package does contain a number of tools for *linear* time series analysis (spectrum, autocorrelations, histograms, etc.), these are only suitable for a quick inspection of the data. Spectral or even ARMA estimation are industries in themselves and we refer the reader—and the user of TISEAN—to the existing literature and available statistics software for an optimal, up-to-date implementations of these important methods.

Some users will miss a convenient graphical interface to the programs. We felt that at this point the extra implementational effort would not be justified by the expected additional functionality of the package. Work is in progress, however, to provide interfaces to higher level mathematics (or statistics) software.

B. General computational issues

The natural basis to formulate nonlinear time series algorithms from chaos theory is a multi-dimensional phase space, rather than the time or the frequency domain. It will be essential for the global dynamics in this phase space to be nonlinear in order to fulfill the constraints of nontriviality and boundedness. Only in particular cases this nonlinear structure will be easily representable by a global nonlinear function. Instead, all properties will be expressed in terms of local quantities, often by suitable global averages. All local information will be gained from neighborhood relations of various kinds from time series elements. Thus, a recurrent computational issue will be that of defining local neighborhoods in phase space. Finding neighbors in multidimensional space is a common problem of computational geometry. Multidimensional tree structures are widely used and have attractive theoretical properties. Finding all neighbors in a set of N vectors takes $O(\log N)$ operations, thus the total operation count is $O(N \log N)$. A fast alternative that is particularly efficient for relatively low-dimensional structures embedded in multidimensional spaces is given by box-assisted neighbor search methods which can push the operation count down to $O(N)$ under certain assumptions. Both approaches are reviewed in Ref. 20 with particular emphasis on time series applications. In the TISEAN project, a fast neighbor search is done using a box-assisted approach, as described in Ref. 2.

No matter in what space dimension we are working, we will define *candidates* for nearest neighbors in two dimensions using a grid of evenly spaced boxes. With a grid of spacing ϵ, all neighbors of a vector $\mathbf{x}$ closer than epsilon must be located in the adjacent boxes. But not all points in the adjacent boxes are neighbors, they may be up to 2ϵ away in two dimensions and arbitrarily far in higher dimensions. The neighbors search is thus a two stage process. First, the box-assisted data base has to be filled and then for each point a list of neighbors can be requested. There are a few instances where it is advisable to abandon the fast neighbor search strategy. One example is the program noise that does nonlinear noise filtering in a data stream. It is supposed to start filtering soon after the first points have been recorded. Thus the neighbor data base cannot be constructed in the beginning. Another exception is if quite short (<500 points, say), high-dimensional data are processed. Then the overhead for the neighbor search should be avoided and instead an optimized straight $O(N^2)$ method be used, like it is done in c2naive.

For portability, all programs expect time series data in

column format represented by ASCII numbers. The column to be processed can be specified on the command line. Although somewhat wasteful for storing data, ASCII numbers can be produced and read by most other software. All parameters can be set by adding options to the command, which, in many programs, just replace the default values. Obviously, relying on default settings is particularly dangerous in such a subtle field. Since almost all routines can read from standard input and write to standard output, programs can be part of as pipelines. For example, they can be called filters from inside graphics software or other software tools which are able to execute shell commands. Also, data conversion or compression can be done ''on the fly'' this way. The reader here realizes that we are speaking of UNIX or LINUX platforms which seems to be the most appropriate environment. It is, however, expected that most of the programs will be ported to other environments in the near future.

For those readers familiar with the programs published in Ref. 2 we should mention that these form the basis for a number of those TISEAN programs written in FORTRAN. The C programs, even if they do similar things, are fairly independent implementations. All C and C++ programs now use dynamic allocation of storage, for example.

II. PHASE SPACE REPRESENTATION

Deterministic dynamical systems describe the time evolution of a system in some phase space $\Gamma \subset \mathbf{R}$. They can be expressed, for example, by ordinary differential equations,

$$\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t)), \tag{1}$$

or in discrete time $t = n\,\Delta t$ by maps of the form

$$\mathbf{x}_{n+1} = \mathbf{f}(\mathbf{x}_n). \tag{2}$$

A time series can then be thought of as a sequence of observations $\{s_n = s(\mathbf{x}_n)\}$ performed with some measurement function $s(\cdot)$. Since the (usually scalar) sequence $\{s_n\}$ in itself does not properly represent the (multi-dimensional) phase space of the dynamical system, one has to employ some technique to unfold the multi-dimensional structure using the available data.

A. Delay coordinates

The most important phase space reconstruction technique is the *method of delays*. Vectors in a new space, the embedding space, are formed from time delayed values of the scalar measurements:

$$\mathbf{s}_n = (s_{n-(m-1)\tau}, s_{n-(m-2)\tau}, \ldots, s_n). \tag{3}$$

The number m of elements is called the *embedding dimension*, the time τ is generally referred to as the *delay* or *lag*. Celebrated embedding theorems by Takens[21] and by Sauer *et al.*[22] state that if the sequence $\{s_n\}$ does indeed consist of scalar measurements of the state of a dynamical system, then under certain genericity assumptions, the time delay embedding provides a one-to-one image of the original set $\{\mathbf{x}\}$, provided m is large enough.

Time delay embeddings are used in almost all methods described in this paper. The implementation is straightforward and does not require further explanation. If N scalar
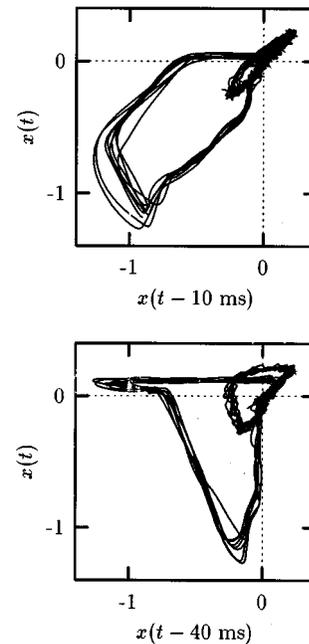

FIG. 1. Time delay representation of a human magneto-cardiogram. In the upper panel, a short delay time of 10 ms is used to resolve the fast waveform corresponding to the contraction of the ventricle. In the lower panel, the slower recovery phase of the ventricle (small loop) is better resolved due to the use of a slightly longer delay of 40 ms. Such a plot can be conveniently be produced by a graphic tool such as gnuplot without generating extra data files.

measurements are available, the number of embedding vectors is only $N - (m-1)\tau$. This has to be kept in mind for the correct normalization of averaged quantities. There is a large literature on the ''optimal'' choice of the embedding parameters m and τ. It turns out, however, that what constitutes the optimal choice largely depends on the application. We will therefore discuss the choice of embedding parameters occasionally together with other algorithms below.

A stand-alone version of the delay procedure (delay, embed) is an important tool for the visual inspection of data, even though visualization is restricted to two dimensions, or at most two-dimensional projections of three-dimensional renderings. A good unfolding already in two dimensions may give some guidance about a good choice of the delay time for higher-dimensional embeddings. As an example let us show two different two-dimensional delay coordinate representations of a human magneto-cardiogram (Fig. 1). Note that we do neither assume nor claim that the magneto- (or electro-) cardiogram is deterministic or even chaotic. Although in the particular case of cardiac recordings the use of time delay embeddings can be motivated theoretically,[23] here we only want to use the embedding technique as a visualization tool.

B. Embedding parameters

A reasonable choice of the delay gains importance through the fact that we always have to deal with a finite amount of noisy data. Both noise and finiteness will prevent us from having access to infinitesimal length scales, so that

the structure we want to exploit should persist up to the largest possible length scales. Depending on the type of structure we want to explore we have to choose a suitable time delay. Most obviously, delay unity for highly sampled flow data will yield delay vectors which are all concentrated around the diagonal in the embedding space and thus all structure perpendicular to the diagonal is almost invisible. In Ref. 24 the terms *redundancy* and *irrelevance* were used to characterize the problem: Small delays yield strongly correlated vector elements, large delays lead to vectors whose components are (almost) uncorrelated and the data are thus (seemingly) randomly distributed in the embedding space. Quite a number of papers have been published on the proper choice of the delay and embedding dimension. We have argued repeatedly[11,2,3] that an ''optimal'' embedding can—if at all—only be defined relative to a specific purpose for which the embedding is used. Nevertheless, some quantitative tools are available to guide the choice.

The usual autocorrelation function (`autocor`, `corr`) and the time delayed mutual information (`mutual`), as well as visual inspection of delay representations with various lags provide important information about reasonable delay times while the false neighbors statistic (`false_nearest`) can give guidance about the proper embedding dimension. Again, ''optimal'' parameters cannot be thus established except in the context of a specific application.

1. Mutual information

The time delayed mutual information was suggested by Fraser and Swinney[25] as a tool to determine a reasonable delay: Unlike the autocorrelation function, the mutual information also takes into account nonlinear correlations. One has to compute

$$S = -\sum_{ij} p_{ij}(\tau)\ln\frac{p_{ij}(\tau)}{p_i p_j}, \qquad (4)$$

where for some partition on the real numbers p_i is the probability to find a time series value in the i-th interval, and $p_{ij}(\tau)$ is the joint probability that an observation falls into the i-th interval and the observation time τ later falls into the j-th. In theory this expression has no systematic dependence on the size of the partition elements and can be quite easily computed. There exist good arguments that if the time delayed mutual information exhibits a marked minimum at a certain value of τ, then this is a good candidate for a reasonable time delay. However, these arguments have to be modified when the embedding dimension exceeds two. Moreover, as will become transparent in the following sections, not all applications work optimally with the same delay. Our routine `mutual` uses Eq. (4), where the number of boxes of identical size and the maximal delay time has to be supplied. The adaptive algorithm used in Ref. 25 is more data intensive. Since we are not really interested in absolute values of the mutual information here but rather in the first minimum, the minimal implementation given here seems to be sufficient. The related generalized mutual information of order two can be defined using the correlation sum concept (Sec. VII, Refs. 26, 27). An estimation of the correlation entropy is explained in Sec. VII A.
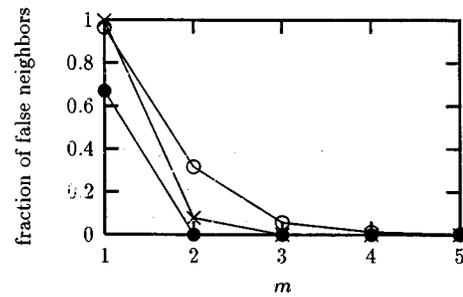

FIG. 2. The fraction of false nearest neighbors as a function of the embedding dimension for noise free Lorenz (crosses) and Hénon (filled circles) time series, as well as a Hénon time series (open circles) corrupted by 10% of noise.

2. False nearest neighbors

A method to determine the minimal sufficient embedding dimension m was proposed by Kennel *et al.*[28] It is called the *false nearest neighbor* method. The idea is quite intuitive. Suppose the minimal embedding dimension for a given time series $\{s_i\}$ is m_0. This means that in a m_0-dimensional delay space the reconstructed attractor is a one-to-one image of the attractor in the original phase space. Especially, the topological properties are preserved. Thus the neighbors of a given point are mapped onto neighbors in the delay space. Due to the assumed smoothness of the dynamics, neighborhoods of the points are mapped onto neighborhoods again. Of course the shape and the diameter of the neighborhoods is changed according to the Lyapunov exponents. But suppose now you embed in an m-dimensional space with $m < m_0$. Due to this projection the topological structure is no longer preserved. Points are projected into neighborhoods of other points to which they would not belong in higher dimensions. These points are called *false neighbors*. If now the dynamics is applied, these false neighbors are not typically mapped into the image of the neighborhood, but somewhere else, so that the average ''diameter'' becomes quite large.

The idea of the algorithm `false_nearest` is the following. For each point $\vec{s}_i$ in the time series look for its nearest neighbor $\vec{s}_j$ in an m-dimensional space. Calculate the distance $\|\vec{s}_i - \vec{s}_j\|$. Iterate both points and compute

$$R_i = \frac{|s_{i+1} - s_{j+1}|}{\|\vec{s}_i - \vec{s}_j\|}. \qquad (5)$$

If R_i exceeds a given heuristic threshold R_t, this point is marked as having a false nearest neighbor.[28] The criterion that the embedding dimension is high enough is that the fraction of points for which $R_i > R_t$ is zero, or at least sufficiently small. Two examples are shown in Fig. 2. One is for the Lorenz system (crosses), one for the Hénon system (filled circles), and one for a Hénon time series corrupted by 10% of Gaussian white noise (open circles). One clearly sees that, as expected, $m = 2$ is sufficient for the Hénon and $m = 3$ for the Lorenz system, whereas the signature is less clear in the noisy case.

The introduction of the false nearest neighbors concept and other ad hoc instruments was partly a reaction to the

finding that many results obtained for the genuine invariants, like the correlation dimension, has been spurious due to caveats of the estimation procedure. In the latter case, serial correlations and small sample fluctuations can easily be mistaken for nonlinear determinism. It turns out, however, that the ad hoc quantities basically suffer from the same problems—which can be cured by the same precautions. The implementation `false_nearest` therefore allows us to specify a minimal temporal separation of valid neighbors.

Other software for the analysis of false nearest neighbors is available in source form from Kennel.[29] Or, if you prefer to pay for a license, from Ref. 30.

C. Principal components

It has been shown in Ref. 22 that the embedding technique can be generalized to a wide class of smooth transformations applied to a time delay embedding. In particular, if we introduce time delay coordinates $\{\mathbf{s}_n\}$, then almost every linear transformation of sufficient rank again leads to an embedding. A specific choice of linear transformation is known as *principal component analysis, singular value decomposition, empirical orthogonal functions, Karhunen–Loéve decomposition*, and probably a few other names. The technique is fairly widely used, for example, to reduce multivariate data to a few major modes. There is a large literature, including textbooks like that by Jolliffe.[31] In the context of nonlinear signal processing, the technique has been advocated among others by Broomhead and King.[32]

The idea is to introduce a new set of orthonormal basis vectors in embedding space such that projections onto a given number of these directions preserve the maximal fraction of the variance of the original vectors. In other words, the error in making the projection is minimized for a given number of directions. Solving this minimization problem[31] leads to an eigenvalue problem. The desired *principal directions* can be obtained as the eigenvectors of the symmetric autocovariance matrix that correspond to the largest eigenvalues. The alternative and formally equivalent approach via the trajectory matrix is used in Ref. 32. The latter is numerically more stable but involves the singular value decomposition of an $N \times m$ matrix for N data points embedded in m dimensions, which can easily exceed computational resources for time series of even moderate length.[33]

In almost all the algorithms described below, simple time delay embeddings can be substituted by principal components. In the TISEAN project (routines `svd`, `pc`), principal components are only provided as a stand-alone visualization tool and for linear filtering,[34] see Sec. II E below. In any case, one first has to choose an initial time delay embedding and then a number of principal components to be kept. For the purpose of visualization, the latter is immediately restricted to two or at most three. In order to take advantage of the noise averaging effect of the principal component scheme, it is advisable to choose a much shorter delay than one would for an ordinary time delay embedding, while at the same time increasing the embedding dimension. Experimentation is recommended. Figure 3 shows the contributions
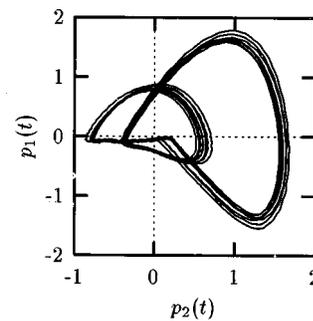

FIG. 3. The phase space representation of a human magneto-cardiogram using the two largest principal components. An initial embedding was chosen in $m=20$ dimensions with a delay of $\tau=7$ ms. The two components cover 70% of the variance of the initial embedding vectors.

of the first two principal components to the magneto-cardiogram shown in Fig. 1.

D. Poincaré sections

Highly sampled data representing the continuous time of a differential equation are called *flow* data. They are characterized by the fact that errors in the direction tangent to the trajectory do neither shrink nor increase exponentially (so called marginally stable direction) and thus possess one Lyapunov exponent which is zero, since any perturbation in this direction can be compensated by a simple shift of the time. Since in many data analysis tasks this direction is of low interest, one might wish to eliminate it. The theoretical concept to do so is called the Poincaré section. After having chosen an $(m-1)$-dimensional hyperplane in the m-dimensional (embedding) space, one creates a compressed time series of only the intersections of the time continuous trajectory with this hyperplane *in a predefined orientation*. These data are then vector valued discrete time *map like* data. One can consider the projection of these $(m-1)$-dimensional vectors onto the real numbers as another measurement function (e.g., by recording the value of s_n when $\mathbf{s}_n$ passes the Poincaré surface), so that one can create a new scalar time series if desirable. The program `poincare` constructs a sequence of vectors from a scalar flow-like data set, if one specifies the hyperplane, the orientation, and the embedding parameters. The intersections of the discretely sampled trajectory with the Poincaré plane are computed by a third order interpolation (see Fig. 4).

The placement of the Poincaré surface is of high relevance for the usefulness of the result. An optimal surface maximizes the number of intersections, i.e., minimizes the time intervals between them, if at the same time the attractor remains connected. One avoids the trials and errors related to that if one defines a surface by the zero crossing of the temporal derivative of the signal, which is synonymous with collecting all maxima or all minima, respectively. This is done by `extrema`. However, this method suffers more from noise, since for small time derivatives (i.e., close to the extrema) additional extrema can be produced by perturbations. Another aspect for the choice of the surface of section is that one should try to maximize the variance of the data inside the section, since their absolute noise level is independent of
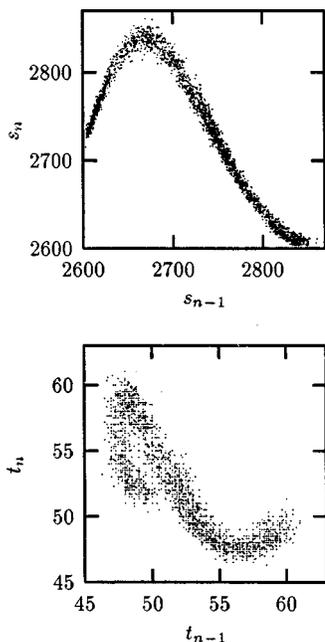
FIG. 4. A Poincaré surface of section using `extrema`: A two-dimensional delay plot of the sequence of maxima (top) and of the time intervals between successive maxima (bottom). Without employing the option `-t` *time*, where *time* is the number of time steps after the last extremum during which no further extrema are searched for (here: 3), one finds some fake extrema due to noise showing up close to the diagonal of the delay representation. Data: Time series of the output power of a CO_2 laser (Ref. 35).
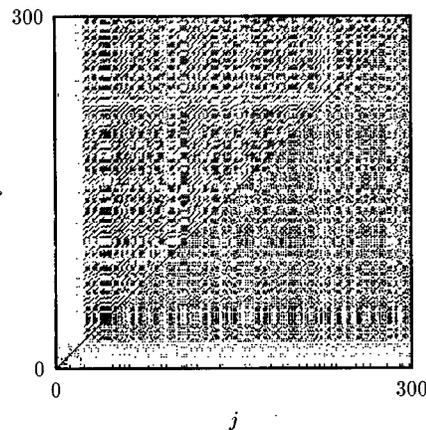

FIG. 5. The recurrence plot for a Poincaré section data from a vibrating string experiment (Ref. 39). Above the diagonal an embedding in two dimensions was used while below the diagonal, scalar time series values were compared. In both cases the lighter shaded region at the beginning of the recording indicates that these data are dynamically distinct from the rest. In this particular case this was due to adjustments in the measurement apparatus.

the section. One last remark: Time intervals between intersections are phase space observables as well[36] and the embedding theorems are thus valid. For a time series with pronounced spikes, one often likes to study the sequence of interspike time intervals, e.g., in cardiology the RR-intervals. If these time intervals are constructed in a way to yield time intervals of a Poincaré map, they are suited to reflect the deterministic structure (if any). For complications see Ref. 36.

For a periodically driven nonautonomous system the best surface of section is usually given by a fixed phase of the driving term, which is also called a *stroboscopic view*. Here again the selection of the phase should be guided by the variance of the signal inside the section.

E. SVD filters

There are at least two reasons to apply a SVD filter to time series data: Either, if one is working with flow data, one can implicitly determine the optimal time delay, or, when deriving a stroboscopic map from synchronously sampled data of a periodically driven system, one might use the redundancy to optimize the signal to noise ratio.

In both applications the mathematics is the same: One constructs the covariance matrix of all data vectors (e.g., in an m-dimensional time delay embedding space),

$$C_{ij} = \langle s_{n-m+i} s_{n-m+j} \rangle - \langle s_{n-m+i} \rangle \langle s_{n-m+j} \rangle, \qquad (6)$$

and computes its singular vectors. Then one projects onto the m-dimensional vectors corresponding to the q largest singular values. To work with flow data, q should be at least the

correct embedding dimension, and m considerably larger (e.g., $m = 2q$ or larger). The result is a vector valued time series, and in Ref. 22 the relation of these components to temporal derivatives on the one hand and to Fourier components on the other hand were discussed. If, in the nonautonomous case, one wants to compress flow data to map data, $q = 1$. In this case, the redundancy of the flow is implicitly used for noise reduction of the map data. The routine `svd` can be used for both purposes.

III. VISUALIZATION, NONSTATIONARITY

A. Recurrence plots

Recurrence plots are a useful tool to identify structure in a data set in a time resolved way qualitatively. This can be intermittency (which one detects also by direct inspection), the temporary vicinity of a chaotic trajectory to an unstable periodic orbit, or nonstationarity. They were introduced in Ref. 37 and investigated in much detail in Ref. 38, where you find many hints on how to interpret the results. Our routine `recurr` simply scans the time series and marks each pair of time indices (i,j) with a black dot, whose corresponding pair of delay vectors has distance $\leq \epsilon$. Thus in the (i,j)-plane, black dots indicate closeness. In an ergodic situation, the dots should cover the plane uniformly on average, whereas nonstationarity expresses itself by an overall tendency of the dots to be close to the diagonal. Of course, a return to a dynamical situation the system was in before becomes evident by a black region far away from the diagonal. In Fig. 5, a recurrence plot is used to detect transient behavior at the beginning of a longer recording.

For the purpose of stationary testing, the recurrence plot is not particularly sensitive to the choice of embedding. The contrast of the resulting images can be selected by the distance ϵ and the percentage of dots that should be actually plotted. Various software involving the color rendering and quantification of recurrence plots is offered in DOS executable form by Webber.[40] The interpretation of the often in-
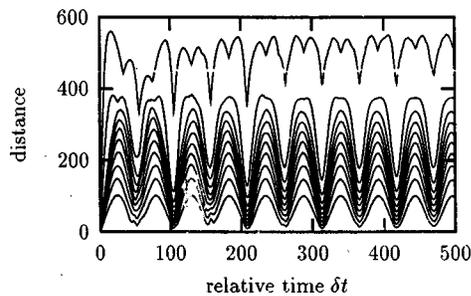
FIG. 6. A space–time separation plot of the CO_2 laser data. Shown are lines of constant probability density of a point to be an ϵ-neighbor of the current point if its temporal distance is δt. Probability densities are 1/10 to 1 with increments of 1/10 from bottom to top. Clear correlations are visible.

triguing patterns beyond the detection and study of nonstationarity is still an open question. For suggestions for the study of nonstationary signals see Ref. 3 and references given there.

B. Space–time separation plot

While the recurrence plot shows absolute times, the space–time separation plot introduced by Provenzale *et al.*[41] integrates along parallels to the diagonal and thus only shows relative times. One usually draws lines of constant probability per time unit of a point to be an ϵ-neighbor of the current point, when its time distance is δt. This helps identifying temporal correlations inside the time series and is relevant to estimate a reasonable delay time, and, more importantly, the Theiler-window w in dimension and Lyapunov-analysis (see Sec. VII). Said in different words, it shows how large the temporal distance between points should be so that we can assume that they form independent samples according to the invariant measure. The corresponding routine of the TISEAN package is `stp`; see Fig. 6.

IV. NONLINEAR PREDICTION

To think about predictability in time series data is worthwhile even if one is not interested in forecasts at all. Predictability is one way in which correlations between data express themselves. These can be linear correlations, nonlinear correlations, or even deterministic constraints. Questions related to those relevant for predictions will reappear with noise reduction and in surrogate data tests, but also for the computation of Lyapunov exponents from data. Prediction is discussed in most of the general nonlinear time series references, in particular, a nice collection of articles can be found in Ref. 17.

A. Model validation

Before entering the methods, we have to discuss how to assess the results. The most obvious quantity for the quantification of predictability is the average forecast error, i.e., the root of the mean squared (rms) deviation of the individual prediction from the actual future value. If it is computed on those values which were also used to construct the model (or to perform the predictions), it is called the *in-sample error*. It is always advisable to save some data for an out-of-sample

test. If the out-of-sample error is considerably larger than the in-sample error, data are either nonstationary or one has overfitted the data, i.e., the fit extracted structure from random fluctuations. A model with less parameters will then serve better. In cases where the data base is poor, one can apply *complete cross-validation* or *take-one-out statistics*, i.e., one constructs as many models as one performs forecasts, and in each case ignores the point one wants to predict. By construction, this method is realized in the local approaches, but not in the global ones.

The most significant, but least quantitative way of model validation is to iterate the model and to compare this synthetic time series to the experimental data. One starts form an observed delay vector as an initial condition and performs a forecast. Its outcome is combined with all but the last components of the initial vector to a new delay vector, and the next forecast is performed. After $n>m$ iterations, the n-th delay vector contains only values generated by the model and no observations any more. In terms of an n-step prediction, the outcome will be terribly bad, since due to the sensitive dependence on initial conditions even an ideal model will create a diverging trajectory due to inaccuracies in the measurement of the initial condition. However, for the model to be reasonable, the resulting attractor should be as similar to the observed data as possible (e.g., in a delay plot), although it is not easy to define the similarity quantitatively.

B. Simple nonlinear prediction

Conventional linear prediction schemes average over all locations in phase space when they extract the correlations they exploit for predictability. Tong[42] promoted an extension that fits different linear models if the current state is below or above a given threshold (TAR, **T**hreshold **A**utoregressive **M**odel). If we expect more than a slight nonlinear component to be present, it is preferable to make the approximation as local in phase space as possible. There have been many similar suggestions in the literature on how to exploit a local structure; see, e.g., Refs. 43–46. The simplest approach is to make the approximation local but only keep the zeroth order, that is, approximate the dynamics locally by a constant. In the TISEAN package we include such a robust and simple method: In a delay embedding space, all neighbors of $\mathbf{s}_n$ are sought, if we want to predict the measurements at time $n+k$. The forecast is then simply

$$\hat{s}_{n+k}=\frac{1}{|\mathcal{U}_n|}\sum_{\mathbf{s}_j\in\mathcal{U}_n}s_{j+k}, \tag{7}$$

i.e., the average over the ''futures'' of the neighbors. The average forecast errors obtained with the routine `zeroth` (`predict` would give similar results) for the laser output data used in Fig. 4 as a function of the number k of steps ahead the predictions are made is shown in Fig. 7. One can also iterate the predictions by using the time series as a data base.

Apart from the embedding parameters, all that has to be specified for zeroth order predictions is the size of the neighborhoods. Since the diffusive motion below the noise level cannot be predicted anyway, it makes sense to select neigh-
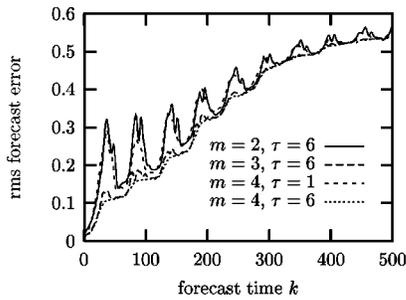
FIG. 7. Predictions k time steps ahead (no iterated predictions) using the program zeroth. Top curve: embedding dimension two is insufficient, since these flow data fill a $(2+\epsilon)$-dimensional attractor. Second from top: Although embedding dimension four should in theory be a good embedding, $\tau=1$ suppresses structure perpendicular to the diagonal so that the predictions are as bad as in $m=2$! Lower curves: $m=3$ and 4 with a delay of about 4–8 time units serve well.
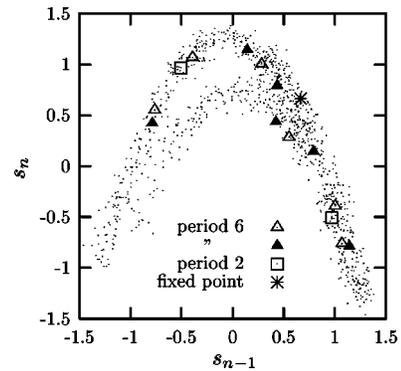

FIG. 8. Orbits of period six, or a sub-period thereof, of the Hénon map, determined from noisy data. The Hénon attractor does not have a period three orbit.

borhoods which are at least as large as the noise level, maybe two or three times larger. For a fairly clean time series, this guideline may result in neighborhoods with very few points. Therefore zeroth also permits us to specify the minimal number of neighbors on which to base the predictions.

A relevant modification of this method is to extend the neighborhood $\mathcal{U}$ to infinity, but to introduce a distance dependent weight,

$$\hat{s}_{n+k} = \frac{\sum_{j \neq n} s_{j+k} w(|\mathbf{s}_n - \mathbf{s}_j|)}{\sum_{j \neq n} w(|\mathbf{s}_n - \mathbf{s}_j|)}, \tag{8}$$

where w is called the kernel. For $w(z) = \Theta(\epsilon - z)$ where Θ is the Heaviside step function, we return to Eq. (7).

C. Finding unstable periodic orbits

As an application of a simple nonlinear phase space prediction, let us discuss a method to locate unstable periodic orbits embedded in a chaotic attractor. This is not the place to review the existing methods to solve this problem, some references include.[47–50] The TISEAN package contains a routine that implements the requirement that for a period p orbit $\{\tilde{\mathbf{s}}_n, n=1,\ldots,p\}$ of a dynamical system like Eq. (2) acting on delay vectors,

$$\tilde{\mathbf{s}}_{n+1} = \mathbf{f}(\tilde{\mathbf{s}}_n), \quad n=1,\ldots,p, \quad \tilde{\mathbf{s}}_{p+1} \equiv \tilde{\mathbf{s}}_1. \tag{9}$$

With unit delay, the p delay vectors contain p different scalar entries, and Eq. (9) defines a root of a system of p nonlinear equations in p dimensions. Multidimensional root finding is not a simple problem. The standard Newton method has to be augmented by special tricks in order to converge globally. Some such tricks, in particular means to select different solutions of Eq. (9), are implemented in Ref. 50. Similar to the problems encountered in nonlinear noise reduction, solving Eq. (9) *exactly* is particularly problematic since $\mathbf{f}(\cdot)$ is unknown and must be estimated from the data. In Ref. 49, approximate solutions are found by performing just one iteration of the Newton method for each available time series point. We prefer to look for a *least squares* solution by minimizing

$$\sum_{n=1}^{p} \|\tilde{\mathbf{s}}_{n+1} - \mathbf{f}(\tilde{\mathbf{s}}_n)\|^2, \quad \tilde{\mathbf{s}}_{p+1} \equiv \tilde{\mathbf{s}}_1 \tag{10}$$

instead. The routine upo uses a standard Levenberg–Marquardt algorithm to minimize (10). For this it is necessary that $\mathbf{f}(\cdot)$ is smooth. Therefore we cannot use the simple nonlinear predictor based on locally constant approximations and we have to use a smooth kernel version, Eq. (8), instead. With $w(z) = \exp(-z^2/2h^2)$, the kernel bandwidth h determines the degree of smoothness of $\mathbf{f}(\cdot)$. Trying to start the minimization with all available time series segments will produce a number of false minima, depending on the value of h. These have to be distinguished from the true solutions by inspection. On the other hand, we can reach solutions of Eq. (9) which are not closely visited in the time series at all, an important advantage over close return methods.[47]

It should be noted that, depending on h, we may always find good minima of (8), even if no solution of Eq. (9), or not even a truly deterministic dynamics, exists. Thus the finding of unstable periodic orbits in itself is not a strong indicator of determinism. We may, however, use the cycle locations or stabilities as a discriminating statistics in a test for nonlinearity; see Sec. VIII. While the orbits themselves are found quite easily, it is surprisingly difficult to obtain reliable estimates of their stability in the presence of noise. In upo, a small perturbation is iterated along the orbit and the unstable eigenvalue is determined by the rate of its separation from the periodic orbit.

The user of upo has to specify the embedding dimension, the period (which may also be smaller) and the kernel bandwidth. For efficiency, one may choose to skip trials with very similar points. Orbits are counted as distinct only when they differ by a specified amount. The routine finds the orbits, their expanding eigenvalue, and possible sub-periods. Figure 8 shows the determination of all period six orbits from 1000 iterates of the Hénon map, contaminated by 10% Gaussian white noise.

D. Locally linear prediction

If there is a good reason to assume that the relation $s_{n+1} = f(\mathbf{s}_n)$ is fulfilled by the experimental data in good approximation (say, within 5%) for some unknown f and that
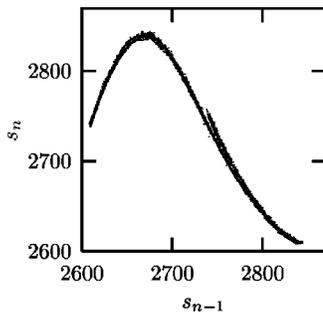
FIG. 9. A time delay representation of 5000 iterations of the local linear predictor `nstep` in two dimensions, starting from the last delay vector of Fig. 4.

FIG. 10. The Casdagli test for nonlinearity: The rms prediction error of local linear models as a function of the neighborhood size ϵ. Lower curve: The CO_2 laser data. These data are obviously highly deterministic in $m = 4$ dimensions and with lag $\tau = 6$. Central curve: The breath rate data shown in Fig. 12 with $m = 4$ and $\tau = 1$. Determinism is weaker (presumably due to a much higher noise level), but still the nonlinear structure is dominant. Upper curve: Numerically generated data of an AR(5) process, a linearly correlated random process ($m = 5$, $\tau = 1$).

f is smooth, predictions can be improved by fitting local linear models. They can be considered as the local Taylor expansion of the unknown f, and are easily determined by minimizing

$$\sigma^2 = \sum_{\mathbf{s}_j \in \mathcal{U}_n} (s_{j+1} - \mathbf{a}_n \mathbf{s}_j - b_n)^2, \tag{11}$$

with respect to $\mathbf{a}_n$ and b_n, where $\mathcal{U}_n$ is the ϵ-neighborhood of $\mathbf{s}_n$, excluding $\mathbf{s}_n$ itself, as before. Then, the prediction is $\hat{s}_{n+1} = \mathbf{a}_n \mathbf{s}_n + b_n$. The minimization problem can be solved through a set of coupled linear equations, a standard linear algebra problem. This scheme is implemented in `onestep`. For moderate noise levels and time series lengths this can give a reasonable improvement over `zeroth` and `predict`. Moreover, as discussed in Sec. VI, these linear maps are needed for the computation of the Lyapunov spectrum. A locally linear approximation was introduced in Refs. 45, 46. We should note that the straight least squares solution of Eq. (11) is not always optimal and a number of strategies are available to regularize the problem if the matrix becomes nearly singular and to remove the bias due to the errors in the ''independent'' variables. These strategies have in common that any possible improvement is bought with a considerable complication of the procedure, requiring subtle parameter adjustments. We refer the reader to Refs. 51, 52 for advanced material.

In Fig. 9 we show iterated predictions of the Poincaré map data from the CO_2 laser (Fig. 4) in a delay representation (using `nstep` in two dimensions). The resulting data do not only have the correct marginal distribution and power spectrum, but also form a perfect skeleton of the original noisy attractor. There are of course artifacts due to noise and the roughness of this approach, but there are good reasons to assume that the line-like substructure reflects fractality of the unperturbed system.

Casdagli[53] suggested the use of local linear models as a test for nonlinearity: He computed the average forecast error as a function of the neighborhood size on which the fit for $\mathbf{a}_n$ and b_n is performed. If the optimum occurs at large neighborhood sizes, the data are (in this embedding space) best described by a linear stochastic process, whereas an optimum at rather small sizes supports the idea of the existence of a nonlinear almost deterministic equation of motion. This protocol is implemented in the routine `ll-ar`; see Fig. 10.
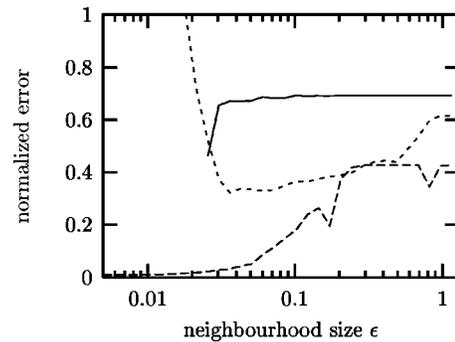
E. Global function fits

The local linear fits are very flexible, but can go wrong on parts of the phase space where the points do not span the available space dimensions and where the inverse of the matrix involved in the solution of the minimization does not exist. Moreover, very often a large set of different linear maps is unsatisfying. Therefore many authors suggested fitting global nonlinear functions to the data, i.e., to solve

$$\sigma^2 = \sum_n (s_{n+1} - f_p(\mathbf{s}_n))^2, \tag{12}$$

where f_p is now a nonlinear function in closed form with parameters p, with respect to which the minimization is done. Polynomials, radial basis functions, neural nets, orthogonal polynomials, and many other approaches have been used for this purpose. The results depend on how far the chosen ansatz f_p is suited to model the unknown nonlinear function, and on how well the data are deterministic at all. We included the routines `rbf` and `polynom` in the TISEAN package, where f_p is modeled by radial basis functions[54,55] and polynomials,[56] respectively. The advantage of these two models is that the parameters p occur linearly in the function f and can thus be determined by simple linear algebra, and the solution is unique. Both features are lost for models where the parameters enter nonlinearly.

In order to make global nonlinear predictions, one has to supply the embedding dimension and time delay as usual. Further, for `polynom` the order of the polynomial has to be given. The program returns the coefficients of the model. In `rbf` one has to specify the number of basis functions to be distributed on the data. The width of the radial basis functions (Lorentzians in our program) is another parameter, but since the minimization is so fast, the program runs many trial values and returns parameters for the best. Figure 11 shows the result of a fit to the CO_2 laser time series (Fig. 4) with radial basis functions.

If global models are desired in order to infer the structure and properties of the underlying system, they should be tested by iterating them. The prediction errors, although
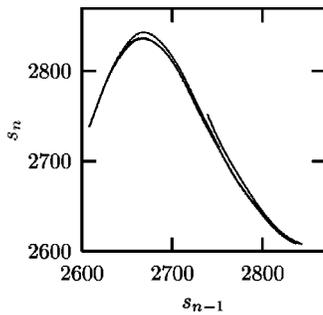
FIG. 11. Attractor obtained by iterating the model that has been obtained by a fit with 40 radial basis functions in two dimensions to the time series shown in Fig. 4. Compare also Fig. 9.

small in size, could be systematic and thus repel the iterated trajectory from the range where the original data are located. It can be useful to study a dependence of the size or the sign of the prediction errors on the position in the embedding space, since systematic errors can be reduced by a different model. Global models are attractive because they yield closed expressions for the full dynamics. One must not forget, however, that these models describe the observed process only in regions of the space which have been visited by the data. Outside this area, the shape of the model depends exclusively on the chosen ansatz. In particular, polynomials diverge outside the range of the data and hence can be unstable under iteration.

V. NONLINEAR NOISE REDUCTION

Filtering of signals from nonlinear systems requires the use of special methods since the usual spectral or other linear filters may interact unfavorably with the nonlinear structure. Irregular signals from nonlinear sources exhibit genuine broad band spectra and there is no justification to identify any continuous component in the spectrum as noise. Nonlinear noise reduction does not rely on frequency information in order to define the distinction between signal and noise. Instead, structure in the reconstructed phase space will be exploited. General serial dependencies among the measurements $\{s_n\}$ will cause the delay vectors $\{\mathbf{s}_n\}$ to fill the available m-dimensional embedding space in an inhomogeneous way. Linearly correlated Gaussian random variables will for example be distributed according to an anisotropic multivariate Gaussian distribution. Linear geometric filtering in phase space seeks to identify the principal directions of this distribution and project onto them; see Sec. II E. Nonlinear noise reduction takes into account that nonlinear signals will form curved structures in delay space. In particular, noisy *deterministic* signals form smeared-out lower-dimensional manifolds. Nonlinear phase space filtering seeks to identify such structures and project onto them in order to reduce noise.

There is a rich literature on nonlinear noise reduction methods. Two articles of review character are available; one by Kostelich and Schreiber,[57] and one by Davies.[58] We refer the reader to these articles for further references and for the discussion of approaches not described in the present article. Here we want to concentrate on two approaches that repre-

sent the geometric structure in phase space by a local approximation. The first and simplest does so to constant order; the more sophisticated uses local linear subspaces plus curvature corrections.

A. Simple nonlinear noise reduction

The simplest nonlinear noise reduction algorithm we know of replaces the central coordinate of each embedding vector by the local average of this coordinate. This amounts to a locally constant approximation of the dynamics and is based on the assumption that the dynamics is continuous. The algorithm is described in Ref. 59, a similar approach is proposed in Ref. 43. In an unstable, for example chaotic, system, it is essential not to replace the first and last coordinates of the embedding vectors by local averages. Due to the instability, initial errors in these coordinates are magnified instead of being averaged out.

This noise reduction scheme is implemented quite easily. First an embedding has to be chosen. Except for extremely oversampled data, it is advantageous to choose a short time delay. The program `lazy` always uses unit delay. The embedding dimension m should be chosen somewhat higher than that required by the embedding theorems. Then for each embedding vector $\{\mathbf{s}_n\}$, a neighborhood $\mathcal{U}_\epsilon^{(n)}$ is formed in phase space containing all points $\{\mathbf{s}_{n'}\}$ such that $\|\mathbf{s}_n - \mathbf{s}_{n'}\| < \epsilon$. The radius of the neighborhoods ϵ should be taken large enough in order to cover the noise extent, but still smaller than a typical curvature radius. These conditions cannot always be fulfilled simultaneously, in which case one has to repeat the process with several choices and carefully evaluate the results. If the noise level is substantially smaller than the typical radius of curvature, neighborhoods of radius about 2–3 times the noise level gave the best results with artificial data. For each embedding vector $\mathbf{s}_n = (s_{n-(m-1)}, \ldots, s_n)$ (the delay time has been set to unity), a corrected middle coordinate $\hat{s}_{n-m/2}$ is computed by averaging over the neighborhood $\mathcal{U}_\epsilon^{(n)}$:

$$\hat{s}_{n-m/2} = \frac{1}{|\mathcal{U}_\epsilon^{(n)}|} \sum_{\mathbf{s}_{n'} \in \mathcal{U}_\epsilon^{(n)}} s_{n'-m/2}. \tag{13}$$

After one complete sweep through the time series, all measurements s_n are replaced by the corrected values $\hat{s}_n$. Of course, for the first and last $(m-1)/2$ points (if m is odd), no correction is available. The average correction can be taken as a new neighborhood radius for the next iteration. Note that the neighborhood of each point at least contains the point itself. If that is the only member, the average, Eq. (13), is simply the uncorrected measurement and no change is made. Thus one can safely perform multiple iterations with decreasing values of ϵ until no further change is made.

Let us illustrate the use of this scheme with an example, a recording of the air flow through the nose of a human as an indicator of breath activity. (The data is part of data set B of the Santa Fe time series contest held in 1991/92;[17] see Rigney *et al.*[60] for a description.) The result of simple nonlinear noise reduction is shown in Fig. 12.
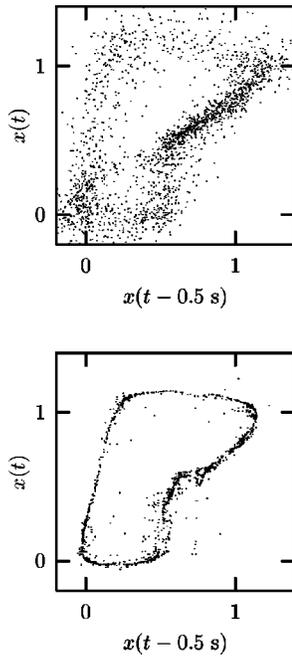
FIG. 12. The simple nonlinear noise reduction of human breath rate data. Three iterations have been carried out, staring with neighborhoods of size 0.4 units. Embeddings in 7 dimensions at unit delay have been used. Arguably, the resulting series (lower panel) is less noisy. However, in Sec. VIII we will show evidence that the noise is not just additive and independent of the signal.

B. Locally projective nonlinear noise reduction

A more sophisticated method makes use of the hypotheses that the measured data is composed of the output of a low-dimensional dynamical system and of random or high-dimensional noise. This means that in an arbitrarily high-dimensional embedding space the deterministic part of the data would lie on a low-dimensional manifold, while the effect of the noise is to spread the data off this manifold. If we suppose that the amplitude of the noise is sufficiently small, we can expect to find the data distributed closely around this manifold. The idea of the projective nonlinear noise reduction scheme is to identify the manifold and to project the data onto it. The strategies described here go back to Ref. 61. A realistic case study is detailed in Ref. 62.

Suppose the dynamical system, Eq. (1) or Eq. (2), form a q-dimensional manifold $\mathcal{M}$ containing the trajectory. According to the embedding theorems, there exists a one-to-one image of the attractor in the embedding space, if the embedding dimension is sufficiently high. Thus, if the measured time series were not corrupted with noise, all the embedding vectors $\mathbf{s}_n$ would lie inside another manifold $\tilde{\mathcal{M}}$ in the embedding space. Due to the noise, this condition is no longer fulfilled. The idea of the locally projective noise reduction scheme is that for each $\mathbf{s}_n$ there exists a correction $\mathbf{\Theta}_n$, with $\|\mathbf{\Theta}_n\|$ small, in such a way that $\mathbf{s}_n - \mathbf{\Theta}_n \in \tilde{\mathcal{M}}$ and that $\mathbf{\Theta}_n$ is orthogonal on $\tilde{\mathcal{M}}$. Of course, a projection to the manifold can only be a reasonable concept if the vectors are embedded in spaces which are higher dimensional than the manifold $\tilde{\mathcal{M}}$. Thus we have to over-embed in m-dimensional spaces with $m > q$.

The notion of orthogonality depends on the metric used. Intuitively one would think of using the Euclidean metric. But this is not necessarily the best choice. The reason is that we are working with delay vectors which contain temporal information. Thus, even if the middle parts of two delay vectors are close, the late parts could be far away from each other due to the influence of the positive Lyapunov exponents, while the first parts could diverge due the negative ones. Hence it is usually desirable to correct only the center part of delay vectors and leave the outer parts mostly unchanged, since their divergence is not only a consequence of the noise, but also of the dynamics itself. It turns out that for most applications it is sufficient to fix just the first and the last component of the delay vectors and correct the rest. This can be expressed in terms of a metric tensor $\mathbf{P}$ which we define to be[61]

$$\mathbf{P}_{ij} = \begin{cases} 1: & i = j \ \text{ and } \ 1 < i, j < m, \\ 0: & \text{elsewhere}, \end{cases} \tag{14}$$

where m is the dimension of the ''over-embedded'' delay vectors.

Thus we have to solve the minimization problem,

$$\sum_i (\mathbf{\Theta}_i \mathbf{P}^{-1} \mathbf{\Theta}_i) \overset{!}{=} \min, \tag{15}$$

with the constraints

$$\mathbf{a}_n^i (\mathbf{s}_n - \mathbf{\Theta}_n) + b_n^i = 0, \quad \text{for } i = q+1, \ldots, m \tag{16}$$

and

$$\mathbf{a}_n^i \mathbf{P} \mathbf{a}_n^j = \delta_{ij}, \tag{17}$$

where the $\mathbf{a}_n^i$ are the normal vectors of $\tilde{\mathcal{M}}$ at the point $\mathbf{s}_n - \mathbf{\Theta}_n$.

These ideas are realized in the programs ghkss, project, and noise in TISEAN. While the first two work as *a posteriori* filters on complete data sets, the last one can be used in a data stream. This means that it is possible to do the corrections online, while the data is coming in (for more details see Sec. V C). All three algorithms mentioned above correct for curvature effects. This is done by either post-processing the corrections for the delay vectors (ghkss) or by preprocessing the centers of mass of the local neighborhoods (project).

The idea used in the ghkss program is the following. Suppose the manifold were strictly linear. Then, provided the noise is white, the corrections in the vicinity of a point on the manifold would point in all directions with the same probability. Thus, if we added all the corrections $\mathbf{\Theta}$ we expect them to sum to zero (or $\langle \mathbf{\Theta} \rangle = \mathbf{O}$). On the other hand, if the manifold is curved, we expect that there is a trend towards the center of curvature ($\langle \mathbf{\Theta} \rangle = \mathbf{\Theta}_{av}$). Thus, to correct for this trend each correction $\mathbf{\Theta}$ is replaced by $\mathbf{\Theta} - \mathbf{\Theta}_{av}$.

A different strategy is used in the program project. The projections are done in a local coordinate system which is defined by the condition that the average of the vectors in the neighborhood is zero. Or, in other words, the origin of the coordinate systems is the center of mass $\langle \mathbf{s}_n \rangle_{\mathcal{U}}$ of the neighborhood $\mathcal{U}$. This center of mass has a bias towards the

center of the curvature.[2] Hence, a projection would not lie on the tangent at the manifold, but on a secant. Now we can compute the center of mass of these points in the neighborhood of $\mathbf{s}_n$. Let us call it $\langle\langle\mathbf{s}_n\rangle\rangle_{\mathcal{U}}$. Under fairly mild assumptions this point has twice the distance from the manifold than $\langle\mathbf{s}_n\rangle_{\mathcal{U}}$. To correct for the bias the origin of the local coordinate system is set to the point: $\langle\langle\mathbf{s}_n\rangle\rangle_{\mathcal{U}}-2\langle\mathbf{s}_n\rangle_{\mathcal{U}}$.

The implementation and use of locally projective noise reduction as realized in `project` and `ghkss` is described in detail in Refs. 61, 62. Let us recall here the most important parameters that have to be set individually for each time series. The embedding parameters are usually chosen quite differently from other applications since considerable over-embedding may lead to better noise averaging. Thus, the delay time is preferably set to unity and the embedding dimension is chosen to provide embedding windows of reasonable lengths. Only for highly oversampled data (like the magneto-cardiogram, Fig. 15, at about 1000 samples per cycle), larger delays are necessary so that a substantial fraction of a cycle can be covered without the need to work in prohibitively high-dimensional spaces. Next, one has to decide how many dimensions q to leave for the manifold supposedly containing the attractor. The answer partly depends on the purpose of the experiment. Rather brisk projections can be optimal in the sense of the lowest residual deviation from the true signal. Low rms error can, however, coexist with systematic distortions of the attractor structure. Thus for a subsequent dimension calculation, a more conservative choice would be in order. Remember, however, that points are only moved *towards* but not *onto* the local linear subspace and too low a value of q does not do as much harm as may be thought.

The noise amplitude to be removed can be selected to some degree by the choice of the neighborhood size. In fact, nonlinear projective filtering can be seen independently of the dynamical systems background as filtering by amplitude rather than by frequency or shape. To allow for a clear separation of noise and signal directions locally, neighborhoods should be at least as large as the supposed noise level, rather larger. This of course competes with curvature effects. For small initial noise levels, it is recommended to also specify a minimal number of neighbors in order to permit stable linearizations. Finally, we should remark that in successful cases most of the filtering is done within the first one to three iterations. Going further is potentially dangerous since further corrections may lead mainly to distortion. One should watch the rms correction in each iteration and stop as soon as it does not decrease substantially any more.

As an example for nonlinear noise reduction we treat the data obtained from a NMR laser experiment.[63] Enlargements of two-dimensional delay representations of the data are shown in Fig. 13. The upper panel shows the raw experimental data which contains about 1.1% of noise. The lower panel was produced by applying three iterations of the noise reduction scheme. The embedding dimension was $m=7$, the vectors were projected down to two dimensions. The size of the local neighborhoods were chosen such that at least 50 neighbors were found. One clearly sees that the fractal structure of the attractor is resolved fairly well.
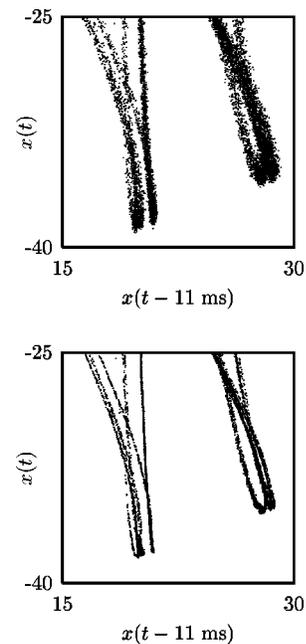

FIG. 13. A two-dimensional representation of the NMR Laser data (top) and the result of the `ghkss` algorithm (bottom) after three iterations.

The main assumption for this algorithm to work is that the data is well approximated by a low-dimensional manifold. If this is not the case it is unpredictable what results are created by the algorithm. In the absence of a real manifold, the algorithm must pick statistical fluctuations and spuriously interprets them as structure. Figure 14 shows a result of the `ghkss` program for pure Gaussian noise. The upper panel shows a delay representation of the original data, the lower shows the outcome of applying the algorithm for 10 iterations. The structure created is purely artificial and has nothing to do with structures in the original data. This means that
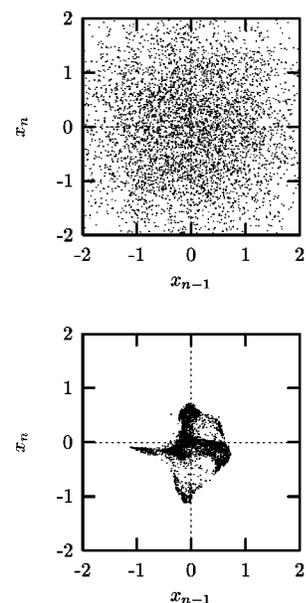

FIG. 14. A two-dimensional representation of a pure Gaussian process (top) and the outcome of the `ghkss` algorithm (bottom) after 10 iterations. Projections from $m=7$ down to two dimensions were performed.
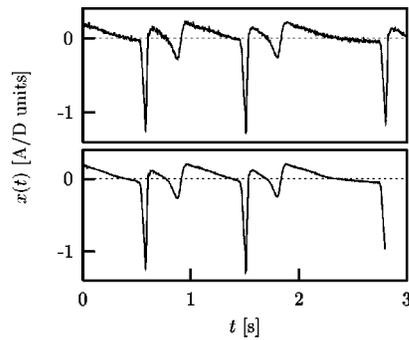
FIG. 15. The real time nonlinear projective filtering of a magneto-cardiogram time series. The top panel shows the unfiltered data. Bottom: Two iterations were done using projections from $m=10$ down to $q=2$ dimensions (delay 0.01 s). Neighborhoods were limited to a radius of 0.1 units (0.05 in the second iteration) and to maximally 200 points. Neighbors were only sought up to 5 s back in time. Thus the first 5 s of data are not filtered optimally and are not shown here. Since the output of each iteration leaps behind its input by one delay window, the last 0.2 s cannot be processed given the data in the upper panel.

if one wants to apply one of the algorithms, one has to carefully study the results. If the assumptions underlying the algorithms are not fulfilled in principle anything can happen. One should note, however, that the performance of the program itself indicates such spurious behavior. For data which is indeed well approximated by a lower-dimensional manifold, the average corrections applied should rapidly decrease with each successful iteration. This was the case with the NMR laser data and, in fact, the correction was so small after three iteration that we stopped the procedure. For the white noise data, the correction only decreased at a rate that corresponds to a general shrinking of the point set, indicating a lack of convergence towards a genuine low-dimensional manifold. Below, we will give an example where an approximating manifold is present without pure determinism. In that case, projecting onto the manifold does reduce noise in a reasonable way. See Ref. 64 for material on the dangers of geometric filtering.

C. Nonlinear noise reduction in a data stream

In Ref. 65, a number of modifications of the above procedure have been discussed which enable the use of nonlinear projective filtering in a data stream. In this case, only points in the past are available for the formation of neighborhoods. Therefore the neighbor search strategy has to be modified. Since the algorithm is described in detail in Ref. 65, we only give an example of its use here. Figure 15 shows the result of nonlinear noise reduction on a magneto-cardiogram (see Figs. 1 and 3) with the program `noise`. The same program has also been used successfully for the extraction of the fetal ECG.[66]

VI. LYAPUNOV EXPONENTS

Chaos arises from the exponential growth of infinitesimal perturbations, together with global folding mechanisms to guarantee boundedness of the solutions. This exponential instability is characterized by the spectrum of Lyapunov exponents.[67] If one assumes a local decomposition of the

phase space into directions with different stretching or contraction rates, then the spectrum of exponents is the proper average of these local rates over the whole invariant set, and thus consists of as many exponents as there are space directions. The most prominent problem in time series analysis is that the physical phase space is unknown, and that instead the spectrum is computed in some embedding space. Thus the number of exponents depends on the reconstruction, and might be larger than in the physical phase space. Such additional exponents are called *spurious*, and there are several suggestions to either avoid them[68] or to identify them. Moreover, it is plausible that only as many exponents can be determined from a time series as are entering the Kaplan Yorke formula (see below). To give a simple example: Consider motion of a high-dimensional system on a stable limit cycle. The data cannot contain any information about the stability of this orbit against perturbations, as long as they are exactly on the limit cycle. For transients, the situation can be different, but then data are not distributed according to an invariant measure and the numerical values are thus difficult to interpret. Apart from these difficulties, there is one relevant positive feature: Lyapunov exponents are invariant under smooth transformations and are thus independent of the measurement function or the embedding procedure. They carry a dimension of an inverse time and have to be normalized to the sampling interval.

A. The maximal exponent

The maximal Lyapunov exponent can be determined without the explicit construction of a model for the time series. A reliable characterization requires that the independence of embedding parameters and the exponential law for the growth of distances are checked[69,70] explicitly. Consider the representation of the time series data as a trajectory in the embedding space, and assume that you observe a very close return $\mathbf{s}_{n'}$ to a previously visited point $\mathbf{s}_n$. Then one can consider the distance $\Delta_0 = \mathbf{s}_n - \mathbf{s}_{n'}$ as a small perturbation, which should grow exponentially in time. Its future can be read from the time series $\Delta_l = \mathbf{s}_{n+l} - \mathbf{s}_{n'+l}$. If one finds that $|\Delta_l| \approx \Delta_0 e^{\lambda l}$ then λ is (with probability one) the maximal Lyapunov exponent. In practice, there will be fluctuations because of many effects, which are discussed in detail in Ref. 69. Based on this understanding, one can derive a robust consistent and unbiased estimator for the maximal Lyapunov exponent. One computes

$$S(\epsilon, m, t) = \left\langle \ln\left(\frac{1}{|\mathcal{U}_n|} \sum_{\mathbf{s}_{n'} \in \mathcal{U}_n} |s_{n+t} - s_{n'+t}| \right) \right\rangle_n . \quad (18)$$

If $S(\epsilon, m, t)$ exhibits a linear increase with identical slope for all m larger than some m_0 and for a reasonable range of ϵ, then this slope can be taken as an estimate of the maximal exponent λ_1.

The formula is implemented in the routines `lyap_k` and `lyapunov` in a straightforward way. (The program `lyap_r` implements the very similar algorithm of Ref. 70, where only the closest neighbor is followed for each reference point. Also, the Euclidean norm is used.) Apart from parameters characterizing the embedding, the initial neigh-
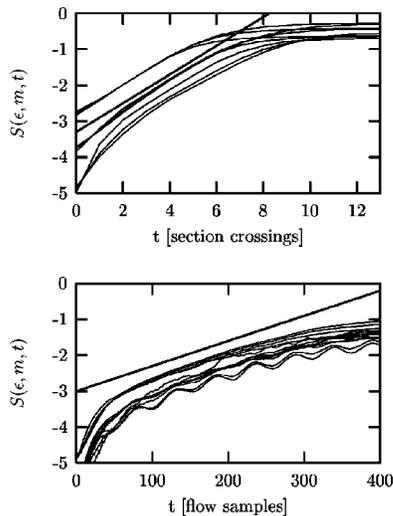
FIG. 17. The breath rate data (cf. Fig. 12) exhibit no linear increase, reflecting the lack of exponential divergence of nearby trajectories.

FIG. 16. Estimating the maximal Lyapunov exponent of the CO_2 laser data. The top panel shows results for the Poincaré map data, where the average time interval T_{av} is 52.2 samples of the flow, and the straight line indicates $\lambda = 0.38$. For comparison: The iteration of the radial basis function model of Fig. 11 yields $\lambda = 0.35$. Bottom panel: Lyapunov exponents determined directly from the flow data. The straight line has slope $\lambda = 0.007$. In a good approximation, $\lambda_{map} = \lambda_{flow} T_{av}$. Here, the time window w to suppress correlated neighbors has been set to 1000, and the delay time was 6 units.

borhood size ϵ is of relevance: The smaller ϵ, the large the linear range of S, if there is one. Obviously, noise and the finite number of data points limit ϵ from below. The default values of lyap_k are rather reasonable for map-like data. It is not always necessary to extend the average in Eq. (18) over the whole available data, reasonable averages can be obtained already with a few hundred reference points s_n. If some of the reference points have very few neighbors, the corresponding inner sum in Eq. (18) is dominated by fluctuations. Therefore one may choose to exclude those reference points which have less than, say, ten neighbors. However, discretion has to be applied with this parameter since it may introduce a bias against sparsely populated regions. This could in theory affect the estimated exponents due to multifractality. Like other quantities, Lyapunov estimates may be affected by serial correlations between reference points and neighbors. Therefore, a minimum time for $|n - n'|$ can and should be specified here as well. See also Sec. VII.

Let us discuss a few typical outcomes. The data underlying the top panel of Fig. 16 are the values of the maxima of the CO_2 laser data. Since this laser exhibits low-dimensional chaos with a reasonable noise level, we observe a clear linear increase in this semi-logarithmic plot, reflecting the exponential divergence of nearby trajectories. The exponent is $\lambda \approx 0.38$ per iteration (map data!), or, when introducing the average time interval, 0.007 per μs. In the bottom panel we show the result for the same system, but now computed on the original flow-like data with a sampling rate of 1 MHz. As an additional structure, an initial steep increase and regular oscillations are visible. The initial increase is due to nonnormality and effects of alignment of distances towards the locally most unstable direction, and the oscillations are an effect of the locally different velocities and thus different densities. Both effects can be much more dramatic in less
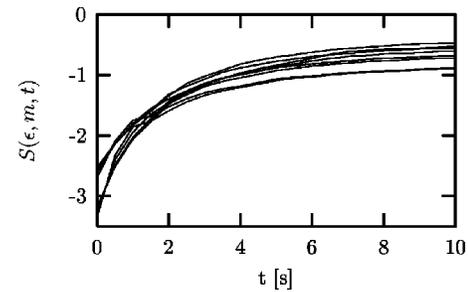
favorable cases, but as long as the regular oscillations possess a linearly increasing average, this can be taken as the estimate of the Lyapunov exponent. Normalizing by the sampling rate, we again find $\lambda \approx 0.007$ per μs, but it is obvious that the linearity is less pronounced than for the map-like data. Finally, we show in Fig. 17 an example of a negative result: We study the human breath rate data used before. No linear part exists, and one cannot draw any reasonable conclusion. It is worth considering the figure on a doubly logarithmic scale in order to detect a power law behavior, which, with power 1/2, could be present for a diffusive growth of distances. In this particular example, there is no convincing power law either.

B. The Lyapunov spectrum

The computation of the full Lyapunov spectrum requires considerably more effort than just the maximal exponent. An essential ingredient is some estimate of the local Jacobians, i.e., of the linearized dynamics, which rules the growth of infinitesimal perturbations. One either finds it from direct fits of local linear models of the type $s_{n+1} = \mathbf{a}_n \mathbf{s}_n + b_n$, such that the first row of the Jacobian is the vector $\mathbf{a}_n$, and $(\mathbf{J})_{ij} = \delta_{i-1,j}$ for $i = 2,...,m$, where m is the embedding dimension. The $\mathbf{a}_n$ is given by the least squares minimization $\sigma^2 = \Sigma_l (s_{l+1} - \mathbf{a}_n \mathbf{s}_l - b_n)^2$ where $\{\mathbf{s}_l\}$ is the set of neighbors of $\mathbf{s}_n$.[45,71] Or one constructs a global nonlinear model and computes its local Jacobians by taking derivatives. In both cases, one multiplies the Jacobians one by one, following the trajectory, to as many different vectors $\mathbf{u}_k$ in tangent space as one wants to compute Lyapunov exponents. Every few steps, one applies a Gram–Schmidt orthonormalization procedure to the set of $\mathbf{u}_k$, and accumulates the logarithms of their rescaling factors. Their average, in the order of the Gram–Schmidt procedure, give the Lyapunov exponents in descending order. The routine lyap_spec uses this method, which goes back to Refs. 71 and 45, employing local linear fits. Apart from the problem of spurious exponents, this method contains some other pitfalls: It *assumes* that there exist well defined Jacobians, and does not test for their relevance. In particular, when attractors are thin in the embedding space, some (or all) of the local Jacobians might be estimated very badly. Then the whole product can suffer from these bad estimates and the exponents are correspondingly wrong. Thus the global nonlinear approach can be superior, if a modeling has been successful; see Sec. IV.

TABLE I. Lyapunov exponents of the NMR laser data, determined with a three-dimensional embedding. The algorithms described in Sec. VI A give $\lambda_1 = 0.3 \pm 0.02$ for the largest exponent.

Method		λ_1	λ_2	λ_3
Local linear	$k = 20$	0.32	−0.40	−1.13
Local linear	$k = 40$	0.30	−0.51	−1.21
Local linear	$k = 160$	0.28	−0.68	−1.31
Radial basis functions		0.27	−0.64	−1.31
Polynomial		0.27	−0.64	−1.15

In Table I we show the exponents of the stroboscopic NMR laser data in a three-dimensional embedding as a function of the neighborhood size. Using global nonlinear models, we find the numbers given in the last two rows. More material is discussed in Ref. 2. The spread of values in the table for this rather clean data set reflects the difficulty of estimating Lyapunov spectra from time series, which has to be done with great care. In particular, when the algorithm is blindly applied to data from a random process, it cannot internally check for the consistency of the assumption of an underlying dynamical system. Therefore a Lyapunov spectrum is computed which now is completely meaningless.

The computation of the first part of the Lyapunov spectrum allows for some interesting cross-checks. It was conjectured,[72] and is found to be correct in most physical situations, that the Lyapunov spectrum and the fractal dimension of an attractor are closely related. If the expanding and least contracting directions in space are continuously filled and only one partial dimension is fractal, then one can ask for the dimensionality of a (fractal) volume such that it is invariant, i.e., such that the sum of the corresponding Lyapunov exponents vanishes, where the last one is weighted with the noninteger part of the dimension:

$$D_{KY} = k + \frac{\sum_{i=1}^{k} \lambda_i}{|\lambda_{k+1}|}, \tag{19}$$

where k is the maximum integer such that the sum of the k largest exponents is still non-negative. D_{KY} is conjectured to coincide with the information dimension.

The Pesin identity is valid under the same assumptions and allows us to compute the KS-entropy:

$$h_{KS} = \sum_{i=1}^{m} \Theta(\lambda_i) \lambda_i. \tag{20}$$

VII. DIMENSIONS AND ENTROPIES

Solutions of dissipative dynamical systems cannot fill a volume of the phase space, since dissipation is synonymous with a contraction of volume elements under the action of the equations of motion. Instead, trajectories are confined to lower-dimensional subsets which have measure zero in the phase space. These subsets can be extremely complicated, and frequently they possess a fractal structure, which means that they are in a nontrivial way self-similar. Generalized dimensions are one class of quantities to characterize this fractality. The *Hausdorff dimension* is, from the mathematical point of view, the most natural concept to characterize

fractal sets,[67] whereas the *information dimension* takes into account the relative visitation frequencies and is therefore more attractive for physical systems. Finally, for the characterization of measured data, other similar concepts, like the *correlation dimension*, are more useful. One general remark is highly relevant in order to understand the limitations of any numerical approach: dimensions characterize a set or an invariant measure whose support is the set, whereas any data set contains only a finite number of points representing the set or the measure. By definition, the dimension of a finite set of points is zero. When we determine the dimension of an attractor numerically, we extrapolate from finite length scales, where the statistics we apply is insensitive to the finiteness of the number of data, to the infinitesimal scales, where the concept of dimensions is defined. This extrapolation can fail for many reasons which will be partly discussed below. Dimensions are invariant under smooth transformations and thus again computable in time delay embedding spaces.

Entropies are an information theoretical concept to characterize the amount of information needed to predict the next measurement with a certain precision. The most popular one is the Kolmogorov–Sinai entropy. We will discuss here only the correlation entropy, which can be computed in a much more robust way. The occurrence of entropies in a section on dimensions has to do with the fact that they can be determined both by the same statistical tool.

A. Correlation dimension

Roughly speaking, the idea behind certain quantifiers of dimensions is that the weight $p(\epsilon)$ of a typical ϵ-ball covering part of the invariant set scales with its diameter like $p(\epsilon) \approx \epsilon^D$, where the value for D depends also on the precise way one defines the weight. Using the square of the probability p_i to find a point of the set inside the ball, the dimension is called the correlation dimension D_2, which is computed most efficiently by the correlation sum:[73]

$$C(m, \epsilon) = \frac{1}{N_{\text{pairs}}} \sum_{j=m}^{N} \sum_{k < j-w} \Theta(\epsilon - |\mathbf{s}_j - \mathbf{s}_k|), \tag{21}$$

where $\mathbf{s}_i$ are m-dimensional delay vectors, $N_{\text{pairs}} = (N - m - w)(N - m - w + 1)/2$ the number of pairs of points covered by the sums, Θ is the Heaviside step function, and w will be discussed below. On sufficiently small length scales and when the embedding dimension m exceeds the correlation-dimension of the attractor,[74]

$$C(m, \epsilon) \propto \epsilon^{D_2}. \tag{22}$$

Since one does not know the correlation-dimension before doing this computation, one checks for convergence of the estimated values of D_2 in m.

The literature on the correct and spurious estimation of the correlation dimension is huge and this is certainly not the place to repeat all the arguments. The relevant caveats and misconceptions are reviewed, for example, in Refs. 75, 11, 76, 2. The most prominent precaution is to exclude temporally correlated points from the pair counting by the so called Theiler window w.[75] In order to become a consistent estimator of the correlation *integral* (from which the dimension is

derived) the correlation *sum* should cover a random sample of points drawn independently according to the invariant measure on the attractor. Successive elements of a time series are not usually independent. In particular, for highly sampled flow data subsequent delay vectors are highly correlated. Theiler suggested to remove this spurious effect by simply ignoring all pairs of points in Eq. (21) whose time indices differ by less than w, where w should be chosen generously. With $O(N^2)$ pairs available, the loss of $O(wN)$ pairs is not dramatic as long as $w \ll N$. At the very least, pairs with $j=k$ have to be excluded,[77] since otherwise the strong bias to $D_2=0$, the mathematically correct value for a finite set of points, reduces the scaling range drastically. Choosing w, the first zero of the auto-correlation function, sometimes even the decay time of the auto-correlation function, are not large enough since they reflect only overall linear correlations.[75,76] The space–time-separation plot (Sec. III B) provides a good means of determining a sufficient value for w, as discussed, for example in Ref. 41, 2. In some cases, notably processes with inverse power law spectra, inspection requires w to be of the order of the length of the time series. This indicates that the data does not sample an invariant attractor sufficiently and the estimation of invariants like D_2 or Lyapunov exponents should be abandoned.

Parameters in the routines d2, c2, and c2naive are as usual the embedding parameters m and τ, the time delay, and the embedding dimension, as well as the Theiler window.

Fast implementation of the correlation sum have been proposed by several authors. At small length scales, the computation of pairs can be done in $O(N \log N)$ or even $O(N)$ time rather than $O(N^2)$ without losing any of the precious pairs; see Ref. 20. However, for intermediate size data sets we also need the correlation sum at intermediate length scales where neighbor searching becomes expensive. Many authors have tried to limit the use of computational resources by restricting one of the sums in Eq. (21) to a fraction of the available points. By this practice, however, one loses valuable statistics at the small length scales where points are so scarce anyway that all pairs are needed for stable results. In Ref. 62, both approaches were combined for the first time by using a fast neighbor search for $\epsilon < \epsilon_0$ and restricting the sum for $\epsilon \geq \epsilon_0$. The TISEAN implementations c2 and d2 go one step further and select the range for the sums individually for each length scale to be processed. This turns out to give a major improvement in speed. The user can specify a desired number of pairs which seems large enough for a stable estimation of $C(\epsilon)$, typically 1000 pairs will suffice. Then the sums are extended to a range which guarantees that number of pairs, or, if this cannot be achieved, to the whole time series. At the largest length scales, this range may be rather small and the user may choose to give a minimal number of reference points to ensure a representative average. Still, using the program c2 the whole computation may thus at large scales be concentrated on the first part of the time series, which seems fair for stationary, noninterment data (nonstationary or strongly intermittent data is usually unsuitable for correlation dimension estimation anyway). The program d2 is safer with this respect. Rather than restricting the range of the sums, only a randomly selected subset is used. The ran-
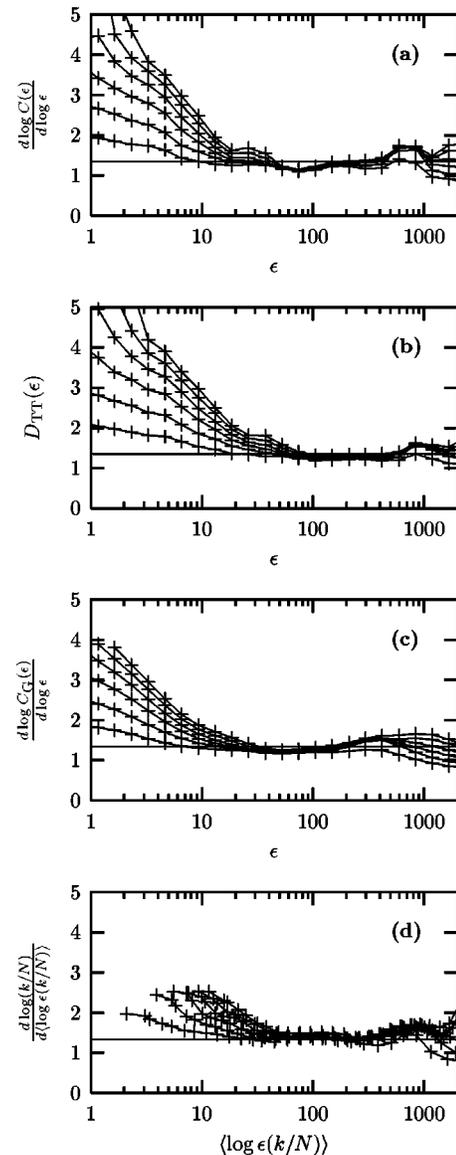

FIG. 18. The dimension estimation for the (noise filtered) NMR laser data. Embedding dimensions 2 to 7 are shown. From above: (a) slopes are determined by straight line fits to the log–log plot of the correlation sum, Eq. (21). (b) The Takes–Theiler-estimator of the same slope. (c) Slopes are obtained by straight line fits to the Gaussian kernel correlation sum, Eq. (25). (d) Instead of the correlation dimension, it has been attempted to estimate the information dimension.

domization, however, requires a more sophisticated program structure in order to avoid an overhead in computation time.

1. Takens–Theiler estimator

Convergence to a finite correlation dimension can be checked by plotting scale-dependent ''effective dimensions'' versus length scale for various embeddings. The easiest way to proceed is to compute (numerically) the derivative of $\log C(m,\epsilon)$ with respect to $\log \epsilon$, for example, by fitting straight lines to the log–log plot of $C(\epsilon)$. In Fig. 18(a) we see the output of the routine c2 acting on data from the NMR laser, processed by c2d in order to obtain local slopes. By default, straight lines are fitted over one octave in ϵ; larger ranges give smoother results. We can see that on the

large scales, self-similarity is broken due to the finite extension of the attractor, and on small but yet statistically significant scales we see the embedding dimension instead of a saturated, m-independent value. This is the effect of noise, which is infinite dimensional, and thus fills a volume in every embedding space. Only on the intermediate scales we see the desired *plateau* where the results are in good approximation independent of m and ϵ. The region where scaling is *established*, not just the range selected for straight line fitting, is called the *scaling range*.

Since the statistical fluctuations in plots like Fig. 18(a) show characteristic (anti-)correlations, it has been suggested[78,79] to apply a maximum likelihood estimator to obtain optimal values for D_2. The Takens–Theiler-estimator reads as

$$D_{TT}(\epsilon) = \frac{C(\epsilon)}{\int_0^\epsilon \frac{C(\epsilon')}{\epsilon'} d\epsilon'}, \tag{23}$$

and can be obtained by processing the output of c2 by c2t. Since $C(\epsilon)$ is available only at discrete values $\{\epsilon_i, i = 0,\ldots,I\}$, we interpolate it by a pure power law [or, equivalently, the log–log plot by straight lines: $\log C(\epsilon) = a_i \log \epsilon + b_i$] in between these. The resulting integrals can be solved trivially and summed:

$$\int_0^\epsilon \frac{C(\epsilon')}{\epsilon'} d\epsilon' = \sum_{i=1}^I e^{b_i} \int_{\epsilon_{i-1}}^{\epsilon_i} (\epsilon')^{a_i-1} d\epsilon'$$

$$= \sum_{i=1}^I \frac{e^{b_i}}{a_i} (\epsilon_i^{a_i} - \epsilon_{i-1}^{a_i}). \tag{24}$$

Plotting D_{TT} vs ϵ [Fig. 18(b)] is an interesting alternative to the usual local slopes plot, Fig. 18(a). It is tempting to use such an ''estimator of dimension'' as a black box to provide a number one might quote as a dimension. This would imply the unjustified assumption that all deviations from exact scaling behavior is due to the statistical fluctuations. Instead, one still has to verify the existence of a scaling regime. Only then, $D_{TT}(\epsilon)$ evaluated at the upper end of the scaling range is a reasonable dimension estimator.

2. Gaussian kernel correlation integral

The correlation sum, Eq. (21), can be regarded as an average density of points where the local density is obtained by a kernel estimator with a step kernel $\Theta(\epsilon - r)$. A natural modification for small point sets is to replace the sharp step kernel by a smooth kernel function of *bandwidth* ϵ. A particularly attractive case that has been studied in the literature[80] is given by the Gaussian kernel, that is, $\Theta(\epsilon - r)$ is replaced by $e^{-r^2/4\epsilon^2}$. The resulting Gaussian kernel correlation sum $C_G(\epsilon)$ has the same scaling properties as the usual $C(\epsilon)$. It has been observed in Ref. 3 that $C_G(\epsilon)$ can be obtained from $C(\epsilon)$ via

$$C_G(\epsilon) = \frac{1}{2\epsilon^2} \int_0^\infty d\tilde{\epsilon}\, e^{-\tilde{\epsilon}^2/4\epsilon^2} \tilde{\epsilon} C(\tilde{\epsilon}), \tag{25}$$

without having to repeat the whole computation. If $C(\epsilon)$ is given at discrete values of ϵ, the integrals in Eq. (25) can be

carried out numerically by interpolating $C(\epsilon)$ with pure power laws. This is done in c2g which uses a 15 point Gauss–Kronrod rule for the numerical integration.

B. Information dimension

Another way of attaching weight to ϵ-balls, which is more natural, is the probability p_i itself. The resulting scaling exponent is called the information dimension D_1. Since the Kaplan–Yorke dimension of Sec. VI is an approximation of D_1, the computation of D_1 through scaling properties is a relevant cross-check for highly deterministic data. D_1 can be computed from a modified correlation sum, where, however, unpleasant systematic errors occur. The *fixed mass* approach[81] circumvents these problems, so that, including finite sample corrections,[77] a rather robust estimator exists. Instead of counting the number of points in a ball one asks here for the diameter ϵ which a ball must have to contain a certain number k of points when a time series of length N is given. Its scaling with k and N yields the dimension in the limit of small length scales by

$$D_1(m) = \lim_{k/N \to 0} \frac{d \log k/N}{d\langle \log \epsilon(k/N) \rangle}. \tag{26}$$

The routine c1 computes the (geometric) mean length scale $\exp\langle \log \epsilon(k/N) \rangle$ for which k neighbors are found in N data points, as a function of k/N. Unlike the correlation sum, finite sample corrections are necessary if k is small.[77] Essentially, the log of k has to be replaced by the digamma function $\Psi(k)$. The resulting expression is implemented in c1. Given m and τ, the routine varies k and N such that the largest reasonable range of k/N is covered with moderate computational effort. This means that for $1/N \leqslant k/N \leqslant K/N$ (default: $K = 100$), all N available points are searched for neighbors and k is varied. For $K/N < k/N \leqslant 1$, $k = K$ is kept fixed and N is decreased. The result for the NMR laser data is shown in Fig. 18(d), where a nice scaling with $D_1 \approx 1.35$ can be discerned. For comparability, the logarithmic derivative of k/N is plotted versus $\exp\langle \log \epsilon(k,N) \rangle$ and not *vice versa*, although k/N is the independent variable. One easily detects again the violations of scaling discussed before: Cutoff on the large scales, noise on small scales, fluctuations on even smaller scales, and a scaling range in between. In this example, D_1 is close to D_2, and multifractality cannot be established positively.

C. Entropy estimates

The correlation dimension characterizes the ϵ dependence of the correlation sum inside the scaling range. It is natural to ask what we can learn from its m-dependence, once m is larger than D_0. The number of ϵ-neighbors of a delay vector is an estimate of the local probability density, and, in fact, it is a kind of joint probability: All m-components of the neighbor have to be similar to those of the actual vector simultaneously. Thus when increasing m,

joint probabilities covering larger time spans get involved. The scaling of these joint probabilities is related to the correlation entropy h_2, such that

$$C(m, \epsilon) \approx \epsilon^{D_2} e^{-mh_2}. \tag{27}$$

As for the scaling in ϵ, also the dependence on m is valid only asymptotically for large m, which one will not reach due to the lack of data points. So one will study $h_2(m)$ vs m and try to extrapolate to large m. The correlation entropy is a lower bound of the Kolmogorov Sinai entropy, which in turn can be estimated by the sum of the positive Lyapunov exponents. The program d2 produces as output the estimates of h_2 directly, from the other correlation sum programs it has to be extracted by post-processing the output.

The entropies of first and second order can be derived from the output of c1 and c2, respectively. An alternate means of obtaining these and the other generalized entropies is by a box counting approach. Let p_i be the probability to find the system state in box i, then the order q entropy is defined by the limit of small box size and large m of

$$\sum_i p_i^q \approx e^{-mh_q}. \tag{28}$$

To evaluate $\Sigma_i p_i^q$ over a fine mesh of boxes in $m \gg 1$ dimensions, economical use of memory is necessary: A simple histogram would take $(1/\epsilon)^m$ storage. Therefore the program boxcount implements the mesh of boxes as a tree with $(1/\epsilon)$-fold branching points. The tree is worked through recursively so that at each instance at most one complete branch exists in storage. The current version does not implement finite sample corrections to Eq. (28).

VIII. TESTING FOR NONLINEARITY

Most of the methods and quantities discussed so far are most appropriate in cases where the data show strong and consistent nonlinear deterministic signatures. As soon as more than a small or at most moderate amount of additive noise is present, scaling behavior will be broken and predictability will be limited. Thus we have explored the opposite extreme, nonlinear and fully deterministic, rather than the classical linear stochastic processes. The bulk of real world time series falls in neither of these limiting categories because they reflect nonlinear responses and effectively stochastic components at the same time. Little can be done for many of these cases with current methods. Often it will be advisable to take advantage of the well founded machinery of spectral methods and venture into nonlinear territory only if encouraged by positive evidence. This section is about methods to establish statistical evidence for nonlinearity beyond a simple rescaling in a time series.

A. The concept of surrogate data

The degree of nonlinearity can be measured in several ways. But how much nonlinear predictability, say, is necessary to exclude more trivial explanations? All quantifiers of nonlinearity show fluctuations but the distributions, or error bars if you wish, are not available analytically. It is therefore necessary to use Monte Carlo techniques to assess the sig-

nificance of the results. One important method in this context is the method of surrogate data.[82] A null hypothesis is formulated, for example, that the data has been created by a stationary Gaussian linear process, and then it is attempted to reject this hypothesis by comparing results for the data to appropriate realizations of the null hypothesis. Since the null assumption is not a simple one but leaves room for free parameters, the Monte Carlo sample has to take these into account. One approach is to construct *constrained realizations* of the null hypothesis. The idea is that the free parameters left by the null are reflected by specific properties of the data. For example, the unknown coefficients of an autoregressive process are reflected in the autocorrelation function. Constrained realizations are obtained by randomizing the data subject to the constraint that an appropriate set of parameters remains fixed. For example, random data with a given periodogram can be made by assuming random phases and taking the inverse Fourier transform of the given periodogram. Random data with the same distribution as a given data set can be generated by permuting the data randomly without replacement. Asking for a given spectrum and a given distribution at the same time poses already a much more difficult question.

B. Iterative Fourier transform method

Very few real time series which are suspected to show nonlinearity follow a Gaussian single time distribution. Non-Gaussianity is the simplest kind of nonlinear signature but it may have a trivial reason: The data may have been distorted in the measurement process. Thus a possible null hypothesis would be that there is a stationary Gaussian linear stochastic process that generates a sequence $\{x_n\}$, but the actual observations are $s_n = s(x_n)$ where $s(\cdot)$ is a monotonic function. Constrained realizations of this null hypothesis would require the generation of random sequences with the same power spectrum (fully specifying the linear process) and the same single time distribution (specifying the effect of the measurement function) as the observed data. The **A**mplitude **A**djusted **F**ourier **T**ransform (AAFT) method proposed in Ref. 82 attempts to invert the measurement function $s(\cdot)$ by rescaling the data to a Gaussian distribution. Then the Fourier phases are randomized and the rescaling is inverted. As discussed in Ref. 83, this procedure is biased towards a flatter spectrum since the inverse of $s(\cdot)$ is not available exactly. In the same reference, a scheme is introduced that removes this bias by iteratively adjusting the spectrum and the distribution of the surrogates. Alternatively, the surrogates are rescaled to the exact values taken by the data and then the Fourier transform is brought to the exact amplitudes obtained from the data. The discrepancy between both steps either converges to zero with the number of iterations or to a finite inaccuracy which decreases with the length of the time series. The program surrogates performs iterations until no further improvement can be made. The last two stages are returned, one having the exact Fourier amplitudes and one taking on the same values as the data. For not too exotic data these two versions should be almost identical. The relative discrepancy is also printed.
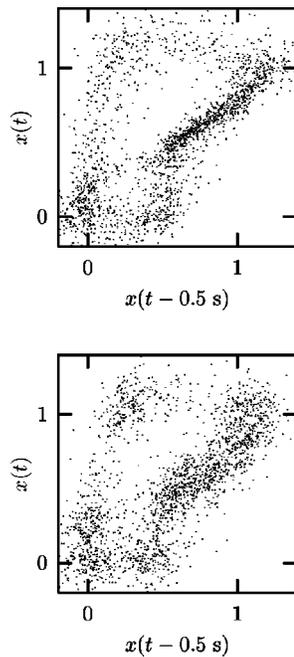
FIG. 19. Upper: The human breath rate data from Fig. 12. Lower: the noise component extracted by the noise reduction scheme has been randomized in order to destroy correlations with the signal. The result appears slightly but significantly less structured than the original.

FIG. 20. Upper trace: Data from a stationary Gaussian linear stochastic process ($x_n = 0.7 x_{n-1} + \eta_n$) measured by $s(x_n) = x_n^3$. Samples 200–220 are an artifact. With the Fourier based scheme (middle trace) the artifact results in an increased number of spikes in the surrogates and reduced predictability. In the lower trace, the artifact has been preserved along with the distribution of values and lags 1,...,25 of the autocorrelation function.

In Fig. 19 we used this procedure to assess the hypothesis that the noise reduction on the breath data reported in Fig. 12 removed an additive noise component which was independent of the signal. If the hypothesis were true, we could equally well add back on the noise sequence or a randomized version of it which lacks any correlations to the signal. In the upper panel of Fig. 19 we show the original data. In the lower panel we took the noise reduced version (cf. Fig. 12, bottom) and added a surrogate of the supposed noise sequence. The result is similar but still significantly different from the original to make the additivity assumption implausible.

Fourier based randomization schemes suffer from some caveats due to the the inherent assumption that the data constitutes one period of a periodic signal, which is not what we really expect. The possible artifacts are discussed, for example, in Ref. 84 and can, in summary, lead to spurious rejection of the null hypothesis. One precaution that should be taken when using surrogates is to make sure that the beginning and the end of the data approximately match in value and phase. Then, the periodicity assumption is not too far wrong and harmless. Usually, this amounts to the loss of a few points of the series. One should note, however, that the routine may truncate the data by a few points itself in order to be able to perform a *fast* Fourier transform which requires the number of points to be factorizable by small prime factors.

C. General constrained randomization

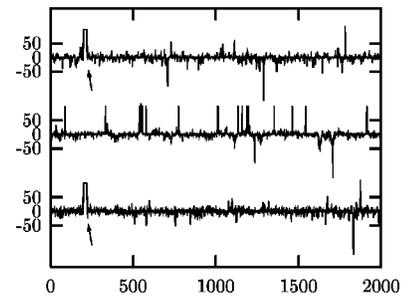In Ref. 85 a general method has been proposed to create random data which fulfill specified constraints. With this method, the artifacts and remaining imprecision of the Fourier based randomization schemes can be avoided by specifying the autocorrelation function rather than the Fourier transform. The former does not assume periodic continuation. Maybe more importantly, the restriction to a rather narrow null hypothesis can be relaxed since, in principle, arbitrary statistical observables can be imposed on the surrogates. A desired property of the data has to be formulated in terms of a cost function which assumes an absolute minimum when the property is fulfilled. States arbitrarily close to this minimal cost can be reached by the method of simulated annealing. The cost function is minimized among all possible permutations of the data. See Ref. 85 for a description of the approach.

The TISEAN package contains the building blocks for a library of surrogate data routines implementing user specified cost functions. Currently, only the autocorrelation function with and without periodic continuation have been implemented. Further, a template is given from which the user may derive her/his own routines. A module is provided that drives the simulated annealing process through an exponential cooling scheme. The user may replace this module by other scheme of her/his choice. A module that performs random pair permutations is given which allows us to exclude a list of points from the permutation scheme. More sophisticated permutation schemes can be substituted if desired. Most importantly, the cost function has to be given as another module. The autocorrelation modules use $\max_{\tau=1}^{\tau_{max}} |C(\tau) - C(\tau)_{data}| / \tau$, where $C(\tau)$ is the autocorrelation function with or without periodic continuation.

In Fig. 20 we show an example fulfilling the null hypothesis of a rescaled stationary Gaussian linear stochastic process which has been contaminated by an artifact at samples 200–220. The Fourier based schemes are unable to implement the artifact part of the null hypothesis. They spread the structure given by the artifact evenly over the whole time span, resulting in more spikes and less predictability. In fact, the null hypothesis of a stationary rescaled Gaussian linear stochastic process can be rejected at the 95% level of significance using nonlinear prediction errors. The artifact would spuriously be mistaken for nonlinearity. With the program randomize_auto_exp_random, we can

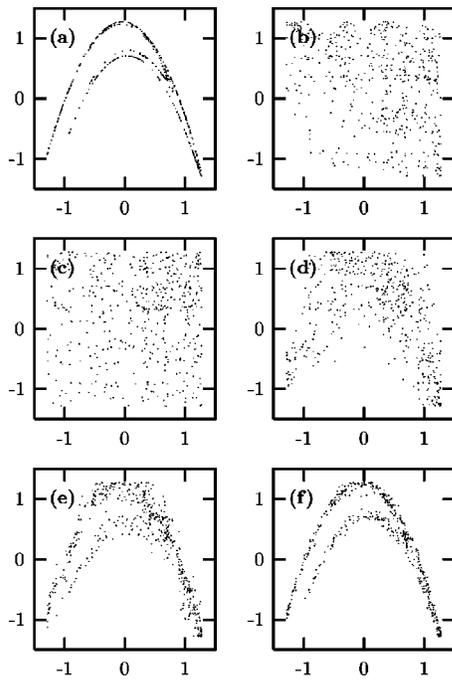FIG. 21. Randomization of 500 points generated by the the Hénon map. (a) Original data; (b) the same autocorrelations and distribution; (c)–(f) different stages of annealing with a cost function C involving three- and four-point correlations. (c) A random shuffle, $C = 2400$; (d) $C = 150$; (e) $C = 15$; (f) $C = 0.002$. See the text.

exclude the artifact from the randomization scheme and obtain a correct test.

As an example of a more exotic cost function, let us show the randomization of 500 iterates of the Hénon map, Fig. 21(a). Panel (b) shows the output of `surrogates` having the same spectrum and distribution. Starting from a random permutation (c), the cost function,

$$
\begin{aligned}
C = & \langle x_{n-1}x_n \rangle + \langle x_{n-2}x_n \rangle + \langle x_{n-1}^2 x_n \rangle + \langle x_{n-1}x_n^2 \rangle \\
& + \langle x_{n-2}^2 x_n \rangle + \langle x_{n-2}x_{n-1}x_n \rangle + \langle x_{n-1}^2 x_n^2 \rangle + \langle x_{n-1}x_n^3 \rangle \\
& + \langle x_{n-1}^3 x_n \rangle,
\end{aligned}
\tag{29}
$$

is minimized (`randomize_generic_exp_random`). It involves are all the higher order autocorrelations which would be needed for a least squares fit with the ansatz $x_n = c - a x_{n-1}^2 + b x_{n-2}$ and in this sense fully specifies the quadratic structure of the data. The random shuffle yields $C = 2400$, panels (c)–(f) correspond to $C = 150, 15, 0.002$, respectively.

Since the annealing process can be very CPU time consuming, it is important to provide an efficient code for the cost function. Specifying $\tau_{\max}$ lags for N data points requires $O(N\tau_{\max})$ multiplications for the calculation of the cost function. An update after a pair has been exchanged, however, can be obtained with $O(\tau_{\max})$ multiplications. Often, the full sum or supremum can be truncated since after the first terms it is clear that a large increase of the cost is unavoidable. The driving Metropolis algorithm provides the current maximal permissible cost for that purpose.

The computation time required to reach the desired accuracy depends on the choice and implementation of the cost

function but also critically on the annealing schedule. There is a vast literature on simulated annealing which cannot be reviewed here. Experimentation with cooling schemes should keep in mind the basic concept of simulated annealing. At each stage, the system—here the surrogate to be created—is kept at a certain ''temperature.'' Like in thermodynamics, the temperature determines how likely fluctuations around the mean energy—here the value of the cost function C—are. At temperature T, a deviation of size ΔC occurs with the Boltzmann probability $\propto \exp(-\Delta C/T)$. In a Metropolis simulation, this is achieved by accepting *all* downhill changes ($\Delta C < 0$), but also uphill changes with probability $\exp(-\Delta C/T)$. Here the changes are permutations of two randomly selected data items. The present implementation offers an exponential cooling scheme, that is, the temperature is lowered by a fixed factor whenever one of two conditions is fulfilled: Either a specified number of changes has been *tried*, or a specified number of changes has been *accepted*. Both these numbers and the cooling factor can be chosen by the user. If the state is cooled too fast it gets stuck, or ''freezes'' in a false minimum. When this happens, the system must be ''melted'' again and cooling is taken up at a slower rate. This can be done automatically until a goal accuracy is reached. It is, however, difficult to predict how many steps it will take. The detailed behavior of the scheme is still subject to ongoing research and in all but the simplest cases, experimentation by the user will be necessary. To facilitate the supervision of the cooling, the current state is written to a file whenever a substantial improvement has been made. Further, the verbosity of the diagnostic output can be selected.

D. Measuring weak nonlinearity

When testing for nonlinearity, we would like to use quantifiers that are optimized for the weak nonlinearity limit, which is not what most time series methods of chaos theory have been designed for. The simple nonlinear prediction scheme (Sec. IV B) has proven quite useful in this context. If used as a comparative statistic, it should be noted that sometimes seemingly inadequate embeddings or neighborhood sizes may lead to rather big errors which have, however, small fluctuations. The tradeoff between bias and variance may be different from the situation where predictions are desired *per se*. The same rationale applies to quantities derived from the correlation sum. Neither the small scale limit, genuine scaling, or the Theiler correction, are formally necessary in a comparative test. However, any temptation to interpret the results in terms like ''complexity'' or ''dimensionality'' should be resisted, even though ''complexity'' does not seem to have an agreed-upon meaning anyway. Apart from average prediction errors, we have found the stabilities of short periodic orbits (see Sec. IV C) useful for the detection of nonlinearity in surrogate data tests. As an alternative to the phase space based methods, more traditional measures of nonlinearity derived from higher order autocorrelation functions (Ref. 86, routine `autocor3`) may also be considered. If a time-reversal asymmetry is present, its statistical confirmation (routine `timerev`) is a very powerful

detector of nonlinearity.[87] Some measures of weak nonlinearity are compared systematically in Ref. 88.

IX. CONCLUSION AND PERSPECTIVES

The TISEAN project makes available a number of algorithms of nonlinear time series analysis to people interested in applications of the dynamical systems approach. To make proper use of these algorithms, it is not essential to have written the programs from scratch, an effort we intend to spare the user by making TISEAN public. Indispensable, however, is a good knowledge of what the programs do, and why they do what they do. The latter requires a thorough background in the nonlinear time series approach which cannot be provided by this paper, but rather by textbooks like in Refs. 10, 2, reviews,[11,12,3] and the original literature.[9] Here, we have concentrated on the actual implementation as it is realized in TISEAN and on examples of the concrete use of the programs.

Let us finish the discussion by giving some perspectives on future work. So far, the TISEAN project has concentrated on the most common situation of a single time series. While for multiple measurements of a similar nature most programs can be modified with moderate effort, a general framework for heterogeneous multivariate recordings (say, blood pressure and heart beat) has not been established so far in a nonlinear context. Nevertheless, we feel that concepts like generalized synchrony, coherence, or information flow are well worth pursuing and at some point should become available to a wider community, including applied research.

Initial experience with nonlinear time series methods indicates that some of the concepts may prove useful enough in the future to become part of the established time series tool box. For this to happen, availability of the algorithms and reliable information on their use will be essential. The publication of a substantial collection of research level programs through the TISEAN project may be seen as one step in that direction. However, the potential user will still need considerable experience in order to make the right decisions—about the suitability of a particular method for a specific time series, about the selection of parameters, about the interpretation of the results. To some extent, these decisions could be guided by software that evaluates the data situation and the results automatically. Previous experience with black box dimension or Lyapunov estimators has not been encouraging, but for some specific problems, ''optimal'' answers can, in principle, be defined and computed automatically, once the optimality criterion is formulated. For example, the prediction programs could be encapsulated in a framework that automatically evaluates the performance for a range of embedding parameters, etc. Of course, quantitative assessment of the results is not always easy to implement and depends on the purpose of the study. As another example, it seems realistic to define ''optimal'' Poincaré surfaces of section and to find the optimal solutions numerically.

Like in most of the time series literature, the issue of stationarity has entered the discussion only as something the lack of which has to be detected in order to avoid spurious results. Taking this point seriously amounts to rejecting a substantial fraction of time series problems, including the most prominent examples, that is, most data from finance, meteorology, and biology. It is quite clear that the mere rejection of these challenging problems is not satisfactory and we will have to develop tools to actually analyze, understand, and predict nonstationary data. Some suggestions have been made for the detection of fluctuating control parameters.[89–92] Most of these can be seen as continuous versions of the classification problem, another application which is not properly represented in TISEAN yet.

Publishing software, or reviews and textbooks for that matter, in a field evolving as rapidly as nonlinear time series analysis will always have the character of a snapshot of the state at a given time. Having the options either to wait until the field has saturated sufficiently or to risk that programs, or statements made, will become obsolete soon, we chose the second option. We hope that we can thus contribute to the further evolution of the field.

[1] The TISEAN software package is publicly available at http://www.mpipks-dresden.mpg.de/~tisean. The distribution includes an online documentation system.

[2] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis* (Cambridge University Press, Cambridge, 1997).

[3] T. Schreiber, ''Interdisciplinary application of nonlinear time series methods,'' Phys. Rep. **308**, 1 (1998).

[4] D. Kaplan and L. Glass, *Understanding Nonlinear Dynamics* (Springer-Verlag, New York, 1995).

[5] E. Ott, *Chaos in Dynamical Systems* (Cambridge University Press, Cambridge, 1993).

[6] P. Bergé, Y. Pomeau, and C. Vidal, *Order Within Chaos: Towards a Deterministic Approach to Turbulence* (Wiley, New York, 1986).

[7] H.-G. Schuster, *Deterministic Chaos: An Introduction* (Physik-Verlag, Weinheim, 1988).

[8] A. Katok and B. Hasselblatt *Introduction to the Modern Theory of Dynamical Systems* (Cambridge University Press, Cambridge, 1996).

[9] E. Ott, T. Sauer, and J. A. Yorke, *Coping with Chaos* (Wiley, New York, 1994).

[10] H. D. I. Abarbanel, *Analysis of Observed Chaotic Data* (Springer-Verlag, New York, 1996).

[11] P. Grassberger, T. Schreiber, and C. Schaffrath, ''Non-linear time sequence analysis,'' Int. J. Bifurcation Chaos Appl. Sci. Eng. **1**, 521 (1991).

[12] H. D. I. Abarbanel, R. Brown, J. J. Sidorowich, and L. Sh. Tsimring, ''The analysis of observed chaotic data in physical systems,'' Rev. Mod. Phys. **65**, 1331 (1993).

[13] D. Kugiumtzis, B. Lillekjendlie, and N. Christophersen, ''Chaotic time series I,'' Modeling Identification Control **15**, 205 (1994).

[14] D. Kugiumtzis, B. Lillekjendlie, and N. Christophersen, ''Chaotic time series II,'' Modeling Identification Control **15**, 225 (1994).

[15] G. Mayer-Kress, in *Dimensions and Entropies in Chaotic Systems* (Springer-Verlag, Berlin, 1986).

[16] M. Casdagli and S. Eubank, in ''Nonlinear modeling and forecasting,'' *Santa Fe Institute Studies in the Science of Complexity*, Proceedings Vol. XII (Addison-Wesley, Reading, MA, 1992).

[17] A. S. Weigend and N. A. Gershenfeld, in ''Time Series Prediction: Fore-

casting the future and understanding the past,'' *Santa Fe Institute Studies in the Science of Complexity*, Proceedings Vol. XV (Addison-Wesley, Reading, MA, 1993).

[18] J. Bélair, L. Glass, U. an der Heiden, and J. Milton, in *Dynamical Disease* (AIP, Woodbury, NY, 1995).

[19] H. Kantz, J. Kurths, and G. Mayer-Kress, in *Nonlinear Analysis of Physiological Data* (Springer-Verlag, Berlin, 1998).

[20] T. Schreiber, ''Efficient neighbor searching in nonlinear time series analysis,'' Int. J. Bifurcation Chaos Appl. Sci. Eng. **5**, 349 (1995).

[21] F. Takens, *Detecting Strange Attractors in Turbulence*, Lecture Notes in Mathematics Vol. 898 (Springer-Verlag, New York, 1981), Vol. 898.

[22] T. Sauer, J. Yorke, and M. Casdagli, ''Embedology,'' J. Stat. Phys. **65**, 579 (1991).

[23] M. Richter and T. Schreiber, ''Phase space embedding of electrocardiograms,'' Phys. Rev. E **58**, 6392 (1998).

[24] M. Casdagli, S. Eubank, J. D. Farmer, and J. Gibson, ''State space reconstruction in the presence of noise,'' Physica D **51**, 52 (1991).

[25] A. M. Fraser and H. L. Swinney, ''Independent coordinates for strange attractors from mutual information,'' Phys. Rev. A **33**, 1134 (1986).

[26] B. Pompe, ''Measuring statistical dependences in a time series,'' J. Stat. Phys. **73**, 587 (1993).

[27] M. Paluš, ''Testing for nonlinearity using redundancies: Quantitative and qualitative aspects,'' Physica D **80**, 186 (1995).

[28] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, ''Determining embedding dimension for phase-space reconstruction using a geometrical construction,'' Phys. Rev. A **45**, 3403 (1992).

[29] http://hpux.cs.utah.edu/hppd/hpux/Physics/embedding-26.May.93

[30] http://www.zweb.com/apnonlin/

[31] I. T. Jolliffe, *Principal Component Analysis* (Springer-Verlag, New York, 1986).

[32] D. Broomhead and G. P. King, ''Extracting qualitative dynamics from experimental data,'' Physica D **20**, 217 (1986).

[33] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes*, 2nd ed. (Cambridge University Press, Cambridge, 1992).

[34] R. Vautard, P. Yiou, and M. Ghil, ''Singular-spectrum analysis: a toolkit for short, noisy chaotic signals,'' Physica D **58**, 95 (1992).

[35] A. Varone, A. Politi, and M. Ciofini, ''CO_2 laser with feedback,'' Phys. Rev. A **52**, 3176 (1995).

[36] R. Hegger and H. Kantz, ''Embedding of sequences of time intervals,'' Europhys. Lett. **38**, 267 (1997).

[37] J. P. Eckmann, S. Oliffson Kamphorst, and D. Ruelle, ''Recurrence plots of dynamical systems,'' Europhys. Lett. **4**, 973 (1987).

[38] M. Casdagli, ''Recurrence plots revisited,'' Physica D **108**, 206 (1997).

[39] N. B. Tufillaro, P. Wyckoff, R. Brown, T. Schreiber, and T. Molteno, ''Topological time series analysis of a string experiment and its synchronized model,'' Phys. Rev. E **51**, 164 (1995).

[40] http://homepages.luc.edu/~cwebber/

[41] A. Provenzale, L. A. Smith, R. Vio, and G. Murante, ''Distinguishing between low-dimensional dynamics and randomness in measured time series,'' Physica D **58**, 31 (1992).

[42] H. Tong, *Threshold Models in Non-Linear Time Series Analysis*, Lecture Notes in Statistics (Springer-Verlag, New York, 1983), Vol. 21.

[43] A. Pikovsky, ''Discrete-time dynamic noise filtering,'' Sov. J. Commun. Technol. Electron. **31**, 81 (1986).

[44] G. Sugihara and R. May, ''Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series,'' Nature (London) **344**, 734 (1990); reprinted in Ref. 9.

[45] J.-P. Eckmann, S. Oliffson Kamphorst, D. Ruelle, and S. Ciliberto, ''Lyapunov exponents from a time series,'' Phys. Rev. A **34**, 4971 (1986); reprinted in Ref. 9.

[46] J. D. Farmer and J. Sidorowich, ''Predicting chaotic time series,'' Phys. Rev. Lett. **59**, 845 (1987); reprinted in Ref. 9.

[47] D. Auerbach, P. Cvitanović, J.-P. Eckmann, G. Gunaratne, and I. Procaccia, ''Exploring chaotic motion through periodic orbits,'' Phys. Rev. Lett. **58**, 2387 (1987).

[48] O. Biham and W. Wenzel, ''Characterization of unstable periodic orbits in chaotic attractors and repellers,'' Phys. Rev. Lett. **63**, 819 (1989).

[49] P. So, E. Ott, S. J. Schiff, D. T. Kaplan, T. Sauer, and C. Grebogi, ''Detecting unstable periodic orbits in chaotic experimental data,'' Phys. Rev. Lett. **76**, 4705 (1996).

[50] P. Schmelcher and F. K. Diakonos, ''A general approach to the finding of

[unstable periodic orbits in chaotic dynamical systems,'' Phys. Rev. E **57**, 2739 (1998).

[51] D. Kugiumtzis, O. C. Lingjærde, and N. Christophersen, ''Regularized local linear prediction of chaotic time series,'' Physica D **112**, 344 (1998).

[52] L. Jaeger and H. Kantz, ''Unbiased reconstruction of the dynamics underlying a noisy chaotic time series,'' Chaos **6**, 440 (1996).

[53] M. Casdagli, ''Chaos and deterministic versus stochastic nonlinear modeling,'' J. R. Statist. Soc. B **54**, 303 (1991).

[54] D. Broomhead and D. Lowe, ''Multivariable functional interpolation and adaptive networks,'' Complex Syst. **2**, 321 (1988).

[55] L. A. Smith, ''Identification and prediction of low dimensional dynamics,'' Physica D **58**, 50 (1992).

[56] M. Casdagli, ''Nonlinear prediction of chaotic time series,'' Physica D **35**, 335 (1989); reprinted in Ref. 9.

[57] E. J. Kostelich and T. Schreiber, ''Noise reduction in chaotic time series data: A survey of common methods,'' Phys. Rev. E **48**, 1752 (1993).

[58] M. E. Davies, ''Noise reduction schemes for chaotic time series,'' Physica D **79**, 174 (1994).

[59] T. Schreiber, ''Extremely simple nonlinear noise reduction method,'' Phys. Rev. E **47**, 2401 (1993).

[60] D. R. Rigney, A. L. Goldberger, W. Ocasio, Y. Ichimaru, G. B. Moody, and R. Mark, ''Multi-channel physiological data: Description and analysis (Data set B),'' in Ref. 17.

[61] P. Grassberger, R. Hegger, H. Kantz, C. Schaffrath, and T. Schreiber, ''On noise reduction methods for chaotic data,'' Chaos **3**, 127 (1993); reprinted in Ref. 9.

[62] H. Kantz, T. Schreiber, I. Hoffmann, T. Buzug, G. Pfister, L. G. Flepp, J. Simonet, R. Badii, and E. Brun, ''Nonlinear noise reduction: a case study on experimental data,'' Phys. Rev. E **48**, 1529 (1993).

[63] M. Finardi, L. Flepp, J. Parisi, R. Holzner, R. Badii, and E. Brun, ''Topological and metric analysis of heteroclinic crises in laser chaos,'' Phys. Rev. Lett. **68**, 2989 (1992).

[64] A. I. Mees and K. Judd, ''Dangers of geometric filtering,'' Physica D **68**, 427 (1993).

[65] T. Schreiber and M. Richter, ''Nonlinear projective filtering in a data stream,'' Wuppertal preprint, 1998.

[66] M. Richter, T. Schreiber, and D. T. Kaplan, ''Fetal ECG extraction with nonlinear phase space projections,'' IEEE Trans. Biomed. Eng. **45**, 133 (1998).

[67] J.-P. Eckmann and D. Ruelle, ''Ergodic theory of chaos and strange attractors,'' Rev. Mod. Phys. **57**, 617 (1985).

[68] R. Stoop and J. Parisi, ''Calculation of Lyapunov exponents avoiding spurious elements,'' Physica D **50**, 89 (1991).

[69] H. Kantz, ''A robust method to estimate the maximal Lyapunov exponent of a time series,'' Phys. Lett. A **185**, 77 (1994).

[70] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, ''A practical method for calculating largest Lyapunov exponents from small data sets,'' Physica D **65**, 117 (1993).

[71] M. Sano and Y. Sawada, ''Measurement of the Lyapunov spectrum from a chaotic time series,'' Phys. Rev. Lett. **55**, 1082 (1985).

[72] J. Kaplan and J. Yorke, ''Chaotic behavior of multidimensional difference equations,'' in *Functional Differential Equations and Approximation of Fixed Points*, edited by H. O. Peitgen and H. O. Walther (Springer-Verlag, New York, 1987).

[73] P. Grassberger and I. Procaccia, Physica D **9**, 189 (1983).

[74] T. Sauer and J. Yorke, ''How many delay coordinates do you need?,'' Int. J. Bifurcation Chaos Appl. Sci. Eng. **3**, 737 (1993).

[75] J. Theiler, J. Opt. Soc. Am. A **7**, 1055 (1990).

[76] H. Kantz and T. Schreiber, Chaos **5**, 143 (1995); reprinted in Ref. 18.

[77] P. Grassberger, ''Finite sample corrections to entropy and dimension estimates,'' Phys. Lett. A **128**, 369 (1988).

[78] F. Takens, in *Dynamical Systems and Bifurcations*, edited by B. L. J. Braaksma, H. W. Broer, and F. Takens, Lecture Notes in Mathematics (Springer-Verlag, Heidelberg, 1985), Vol. 1125.

[79] J. Theiler, ''Lacunarity in a best estimator of fractal dimension,'' Phys. Lett. A **135**, 195 (1988).

[80] J. M. Ghez and S. Vaienti, ''Integrated wavelets on fractal sets I: The correlation dimension,'' Nonlinearity **5**, 777 (1992).

[81] R. Badii and A. Politi, ''Statistical description of chaotic attractors,'' J. Stat. Phys. **40**, 725 (1985).

[82] J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. D. Farmer, ''Testing for nonlinearity in time series: The method of surrogate data,'' Physica D **58**, 77 (1992); reprinted in Ref. 9.

[83] T. Schreiber and A. Schmitz, ''Improved surrogate data for nonlinearity tests,'' Phys. Rev. Lett. **77**, 635 (1996).

[84] J. Theiler, P. S. Linsay, and D. M. Rubin, ''Detecting nonlinearity in data with long coherence times,'' in Ref. 17.

[85] T. Schreiber, ''Constrained randomization of time series data,'' Phys. Rev. Lett. **80**, 2105 (1998).

[86] T. Subba Rao and M. M. Gabr, *An Introduction to Bispectral Analysis and Bilinear Time Series Models*, Lecture Notes in Statistics (Springer-Verlag, New York, 1984), Vol. 24.

[87] C. Diks, J. C. van Houwelingen, F. Takens, and J. DeGoede, ''Reversibility as a criterion for discriminating time series,'' Phys. Lett. A **201**, 221 (1995).

[88] T. Schreiber and A. Schmitz, ''Discrimination power of measures for nonlinearity in a time series,'' Phys. Rev. E **55**, 5443 (1997).

[89] J. Kadtke, ''Classification of highly noisy signals using global dynamical models,'' Phys. Lett. A **203**, 196 (1995).

[90] R. Manuca and R. Savit, ''Stationarity and nonstationarity in time series analysis,'' Physica D **99**, 134 (1996).

[91] M. C. Casdagli, L. D. Iasemidis, R. S. Savit, R. L. Gilmore, S. Roper, and J. C. Sackellares, ''Non-linearity in invasive EEG recordings from patients with temporal lobe epilepsy,'' Electroencephalogr. Clin. Neurophysiol. **102**, 98 (1997).

[92] T. Schreiber, ''Detecting and analyzing nonstationarity in a time series using nonlinear cross predictions,'' Phys. Rev. Lett. **78**, 843 (1997).