Prediction in chaotic nonlinear systems: Methods for time series with broadband Fourier spectra

Henry D. I. Abarbanel
Institute for Nonlinear Science, University of California, San Diego, La Jolla, California 92093-0402; Marine Physical Laboratory, Scripps Institution of Oceanography, La Jolla, California 92093-0701; and Department of Physics, University of California, San Diego, La Jolla, California 92093-0319

Reggie Brown and James B. Kadtke
Institute for Nonlinear Science, University of California, San Diego, La Jolla, California 92093-0402
(Received 22 May 1989; revised manuscript received 9 August 1989)

We consider the problem of prediction and system identification for time series having broadband power spectra that arise from the intrinsic nonlinear dynamics of the system. We view the motion of the system in a reconstructed phase space that captures the attractor (usually strange) on which the system evolves and give a procedure for constructing parametrized maps that evolve points in the phase space into the future. The predictor of future points in the phase space is a combination of operation on past points by the map and its iterates. Thus the map is regarded as a dynamical system and not just a fit to the data. The invariants of the dynamical system, the Lyapunov exponents and optimum moments of the invariant density on the attractor, are used as constraints on the choice of mapping parameters. The parameter values are chosen through a constrained least-squares optimization procedure, constrained by the values of these invariants. We give a detailed discussion of methods to extract the Lyapunov exponents and optimum moments from data and show how to equate them to the values for the parametric map in the constrained optimization. We also discuss the motivation and methods we utilize for choosing the form of our parametric maps. Their form has a strong similarity to the work in statistics on kernel density estimation, but the goals and techniques differ in detail. Our methodology is applied to "data" from the Hénon map and the Lorenz system of differential equations and shown to be feasible. We find that the parameter values that minimize the least-squares criterion do not, in general, reproduce the invariants of the dynamical system. The maps that do reproduce the values of the invariants are not optimum in the least-squares sense, yet still are excellent predictors. We discuss several technical and general problems associated with prediction and system identification on strange attractors. In particular, we consider the matter of the evolution of points that are off the attractor (where few or no data are available), onto the attractor where long-term motion takes place. We find that we are able to realize maps that give a least-squares approximation to the data with rms variation over the attractor of 0.5% or less and still reproduce the dynamical invariants to 5% or better. The dynamical invariants are the classifiers of the dynamical system producing the broadband time series in the first place, so this quality of the maps is essential in representing the correct dynamics.

I. INTRODUCTION

A. General remarks

Analysis of time series from dynamical systems is an important issue in many different fields of engineering and science. The most common tool for this analysis is the Fourier (or other similar) transform of the data $x(n)$ to discover sharp lines in its power spectrum. Spectral identification lies at the heart of much of the work on *linear* systems to which time series analysis is applied.[1,2] When one encounters a broadband power spectrum, the common assumption is that it represents *extrinsic noise* and not characteristics of the signal.

It has become increasingly clear in recent years that nonlinear systems exhibiting deterministic chaos will generate a time series whose power spectrum is broadband. Generically, dissipative nonlinear chaotic systems evolve nonperiodically on a strange attractor that lives in a phase space of finite (and often small) dimension. *Noise* does not evolve on a strange attractor and will occupy an arbitrarily large number of dimensions. Hence to model nonlinear chaotic systems as noise is certainly incorrect. For these systems the source of the broadband spectrum is the *intrinsic* chaotic dynamics that underlies the time series.

Our focus in this work is on signals with a substantial broadband power spectrum which, since external noise is absent or very small, represents the nonperiodic behavior of a dynamical system whose orbits lie on a strange attractor. The idea, now rather well established, that such an object can have a small fractal dimension (and still govern the long time evolution of a system with far more numerous degrees of freedom than represented by the dimension of the attractor) is really the starting point of our work.[3,4]

It is very important that though $x(n)$ may be a long, quiet data set it is likely to have a very broad power spectrum. Indeed, if the signal one is studying has a power spectrum with substantial strong lines, one is well advised to recognize the implied sinusoids as the underlying linear degrees of freedom and avoid altogether the labor we propose here.

It has been shown that in nonlinear systems that exhibit deterministic chaos one can determine from the observation of a *single* dynamical variable the geometric structure of the *many* variable dynamics that produced the measured signal.[5–10] The method that has developed for the construction of the phase space in which the dynamics dwells is called *phase-space reconstruction.* The result of this reconstruction is an embedding space of d dimensions (d is an integer) in which one may observe the attractor. One can view the evolution in the reconstructed phase space of the many dimensional dynamics in a quantitative fashion in the time domain.

In this article we describe both in outline and implementation a program for extracting from the observations of this single broadband temporal signal quantitative predictions for the evolution of initial conditions differing from the observed data points. We assume that once transients are gone the evolution of the system is on a strange attractor with dimension d_A, where d_A is generally fractional. If the evolution of the system is on such an attractor, then the d-dimensional embedding space enclosing the attractor should be sufficiently larger than d_A that all the geometric information about the attractor is exposed in the embedding space. Mañé and Takens's[6,7] formal result requires $d > 2d_A + 1$ to assure one of a faithful representation of the d_A-dimensional attractor as seen in the d-dimensional embedding space, but often, in practice, $d > d_A$ will do. The method of phase-space reconstruction seeks to construct from the $x(n)$'s d-dimensional vectors which, when embedded in $\mathbb{R}^d$ describes the full dynamical evolution of the system. Section II is devoted to the issue of identifying the correct value of d from the data set.

For the moment suppose we have found d by one means or another. We imagine measuring a single scalar variable x at discrete time points $x(n)$ for $n=1,2,\ldots,N_D$. (Observation of several dynamical variables from the system is even better, and serves to provide confirmation of the information on the deductions from observations of any single variable.) We can construct d-dimensional vectors $\mathbf{y}(n)$ in the embedding space by

$$\mathbf{y}(n)=(x(n),x(n+\tau_1),x(n+\tau_2),\ldots,x(n+\tau_{d-1}))\ ,$$

for some set of time lags $\tau_1,\tau_2,\ldots,\tau_{d-1}$. The set of $\mathbf{y}(n)$'s, of which we have $N=N_D-d$, capture the evolution of the nonlinear system under observation as it moves through the d-dimensional phase space. Familiar phase-space coordinates are the time derivatives $x(n),dx(n)/dt,d^2x(n)/dt^2,\ldots$, evaluated at discrete times. The data on $x(n)$ are acquired only at discrete times and establishing the values of these derivatives is certain to be inaccurate. The time lagged $x(n)$'s, which are the coordinate elements of the $\mathbf{y}(n)$'s, are nonlinear combinations of the local time derivatives and are fully acceptable substitutes for the usual phase-space coordinates. This has been emphasized by Eckmann and Ruelle.[10]

With the $\mathbf{y}(n)$'s and the embedding space in hand, we ask here the ambitious question of how we can use the series of $\mathbf{y}(n)$'s to predict $\mathbf{y}(N+1),\mathbf{y}(N+2)$, etc. Equivalently, we can ask what is the evolution, under the same dynamical system that produced the $\mathbf{y}(n)$'s, of a point $\mathbf{y}$, that is on the attractor but not in the original data set. We will have answered this question when given a data set $\mathbf{y}(1),\mathbf{y}(2),\ldots,\mathbf{y}(N)$, we have identified a "reliable" map $\mathbf{F}$ from $\mathbb{R}^d$ to itself parametrized by $\mathbf{a}=(a_1,a_2,\ldots,a_P)$ which takes us from $\mathbf{y}(n)$ to $\mathbf{y}(n+1)$,

$$\mathbf{y}(n+1)=\mathbf{F}(\mathbf{y}(n),\mathbf{a})\ .$$

If we can establish a reliable $\mathbf{F}(\mathbf{y},\mathbf{a})$, then the evolution of a point $\mathbf{y}$ in $\mathbb{R}^d$ that is not a member of the measured data set would be $\mathbf{y}\rightarrow\mathbf{y}_1=\mathbf{F}(\mathbf{y},\mathbf{a})$, $\mathbf{y}_1\rightarrow\mathbf{y}_2=\mathbf{F}(\mathbf{y}_1,\mathbf{a})=\mathbf{F}(\mathbf{F}(\mathbf{y},\mathbf{a}),\mathbf{a})=\mathbf{F}^2(\mathbf{y},\mathbf{a})$, etc.

Our first view of the data $\mathbf{y}(1),\mathbf{y}(2),\ldots,\mathbf{y}(N)$ is that it can be thought of as a pair of columns of vectors in $\mathbb{R}^d$

$$\begin{array}{ll}\mathbf{y}(1) & \mathbf{y}(2)\\ \mathbf{y}(2) & \mathbf{y}(3)\\ \vdots & \vdots\\ \mathbf{y}(n) & \mathbf{y}(n+1)\\ \vdots & \vdots\\ \mathbf{y}(N-1) & \mathbf{y}(N)\ ,\end{array}$$

and our function $\mathbf{F}(\mathbf{y},\mathbf{a})$ comes from parametrically "fitting" the right-hand column of $\mathbf{y}(n+1)$ resulting from the left-hand column of $\mathbf{y}(n)$. Fitting the data then suggests making a least-squares estimation of $\mathbf{a}$ so that the *cost function*

$$\hat{C}(\mathbf{a})=\sum_{n=1}^{N-1}\left[\sum_{m=1}^{d}[y_m(n+1)-F_m(\mathbf{y}(n),\mathbf{a})]^2\right]$$

is minimized. Our approach differs from previous work in detailed tactics and in our imposition of important geometrical structure as constraints on the minimization of the cost function. The articles we have greatly relied on for guidance and initial impetus in our research are those by Farmer and Sidorovitch[11] (we refer to this paper as FS in the following), Lapedes and Farber,[12] and Crutchfield and McNamara.[13]

Our main point, simply stated, is that we are not just making a fit to data with a set of functions $\mathbf{F}(\mathbf{y},\mathbf{a})$. Rather, these functions evaluated along the orbit are to be related to each other in the manner of a dynamical system. This leads to a rather different view of the fitting functions than the one usually taken in trying to match data to observations. It means that the function $\mathbf{F}(\mathbf{y},\mathbf{a})$ evalu-

ated on the data vector $\mathbf{y}(n)$ is required to do more than reproduce $\mathbf{y}(n+1)$ as accurately as possible. $\mathbf{F}(\mathbf{y},\mathbf{a})$ must also be a function which when iterated will reproduce $\mathbf{y}(n+2)$ after two applications to $\mathbf{y}(n)$ and $\mathbf{y}(n+3)$ after three, etc. The notion of $\mathbf{F}(\mathbf{y},\mathbf{a})$ as a dynamical system also leads to modifications of the cost function. The cost function should reflect the fact that iterations of $\mathbf{F}(\mathbf{y},\mathbf{a})$ also yield points on the orbit. Furthermore, under our approach geometrical properties of the dynamical system given by $\mathbf{F}(\mathbf{y},\mathbf{a})$ are used to determine the success of the fit. It is not just the function's ability to reproduce in a least-squares sense the observed data that is important. The data contain invariant information that is essential for a full description of the geometrical structure of the attractor that it evolves on. Our key observation in this article is that, in general, least-squares fitting alone does not produce a map that captures the invariant characteristics of the attractor described by the data $\mathbf{y}(n)$, $n=1,\ldots,N$. One must calculate from the data as many of these invariant quantities as possible and then impose them as constraints on the fit. In this way we emphasize the fact that one is creating a dynamics and not just a fit to data. The product of our minimization of the constrained cost function is a mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$ of $\mathbb{R}^d$ to itself which is not only *reliable* in that it reproduces the given data set by having a small cost function, but is also *representational* in that it has the same geometric invariants as the underlying dynamical system. The methods for identifying those invariants and utilizing them as classifiers for the dynamical system is a matter of some importance in itself.

The invariants are properties of the function $\mathbf{F}(\mathbf{y},\mathbf{a})$ viewed as a dynamical system which maps $\mathbb{R}^d$ to itself. We will concentrate on two kinds of invariants. One kind of invariant, the Lyapunov characteristic exponents $\lambda_1,\lambda_2,\ldots,\lambda_d$, describes the expansion or contraction of phase-space volumes under the iteration of $\mathbf{F}(\mathbf{y},\mathbf{a})$.[10–17] Lyapunov exponents are invariant under smooth changes of coordinate and are independent of the initial conditions of the orbit one follows on the attractor. The second kind of invariant is the density of points on the attractor $\rho(\mathbf{y})$. It captures *global* features of the frequency with which orbits visit various portions of the attractor. The density is a different kind of invariant than the Lyapunov exponents. Its integrals with smooth functions $G(\mathbf{y})$ are unchanged under operation with the mapping function which underlies the dynamics $\mathbf{y}(n)\rightarrow\mathbf{y}(n+1)$,

$$\int d^d y\, \rho(\mathbf{y})G(\mathbf{y})=\int d^d y\, \rho(\mathbf{y})G(\mathbf{F}(\mathbf{y},\mathbf{a}))\ .$$

It too is independent of the initial conditions on the orbits.[10,18,19]

In this paper we find the parameters $\mathbf{a}$ in $\mathbf{F}(\mathbf{y},\mathbf{a})$ by minimizing a cost function subject to certain constraints. The constraints are chosen to insure that iterations of the mapping function $\mathbf{F}(\mathbf{y},\mathbf{a})$ give rise to values of dynamical invariants which are the same as those indicated by the experimentally measured data set $\mathbf{y}(n)$. In this way essential geometric information about the particular attractor on which the data live will be built into the parametric mapping. Straightforward least-squares minimization does not accurately reproduce these invariants. Thus one must perform a least-square minimization subject to the constraints that $\mathbf{F}(\mathbf{y},\mathbf{a})$ accurately produce the Lyapunov spectra $\lambda_1,\lambda_2,\ldots,\lambda_d$ and the invariant density $\rho(\mathbf{y})$. This paper is devoted to explaining in detail how one implements the idea just stated.

B. Choosing maps and predictors

Assuming for the moment that we have successfully embedded the data $x(n)$ in $\mathbb{R}^d$ by creating d-dimensional vectors $\mathbf{y}(n)$, $n=1,\ldots,N$. We need to choose a class of parametrized mappings, a cost function to minimize, and a means to impose the constraints on our minimization. The maps must have some way of fitting the data by closely reproducing one data point from the previous one by $\mathbf{y}(n+1)\approx\mathbf{F}(\mathbf{y}(n),\mathbf{a})$. Our maps are required to "look around" at the behavior of the phase-space neighbors of the point $\mathbf{y}(n)$ and predict forward according to how a cluster of phase-space neighbors, regardless of their temporal sequence, are moved forward in time. The idea here is that one may use knowledge of the behavior of local regions of phase space as well as past points on an orbit to determine where a point will be mapped in the temporal future. The maps we choose must then be sensitive to their neighborhood in phase space and must inquire about the fate of any spatial neighbor under the map without concern of its temporal arrival in the neighborhood. The map will then try to take any new point $\mathbf{y}$ and map it forward to some weighted average of its neighbors' forward evolution.

We take our mappings to be of the form

$$\mathbf{F}(\mathbf{y},\mathbf{a})=\sum_{n=1}^{N-1}\mathbf{y}(n+1)g(\mathbf{y},\mathbf{y}(n);\mathbf{a})\ , \tag{1}$$

where $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ is near 1 for $\mathbf{y}=\mathbf{y}(n)$, and vanishes rapidly for nonzero $|\mathbf{y}-\mathbf{y}(n)|$; the vertical bars represent some norm, in our case Euclidean, in $\mathbb{R}^d$. $\mathbf{F}(\mathbf{y}(k),\mathbf{a})$ will then be quite close to $\mathbf{y}(k+1)$.

This type of mapping is strongly suggestive of the form used in the statistical literature under the name of *kernel estimation* or *kernel density estimation.* An explicit recent example that illustrates the similarity is found in Ref. 20. Other useful discussions of this method applied to various problems are to be found in Refs. 21 and 22; our attention was directed to this similarity by Farmer and Sidorowich.[23] We do not claim to have a better method for choosing our function g than those in the literature, but our motivation here does differ from all the citations except Rice.[20] Our constraints on g are also different, but could be modified. For example, the integral of g over $\mathbf{y}$ need not be unity, nor do we require that g be positive. We will return to a discussion of choices for g in our summary in Sec. VI.

Our choice here for $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$—one among many, of course—is this:

$$g(\mathbf{y},\mathbf{y}(n);\mathbf{a})=\frac{\exp[-|\mathbf{y}-\mathbf{y}(n)|^2/\sigma]\left[a_1+a_2\mathbf{y}(n)\cdot(\mathbf{y}-\mathbf{y}(n))+\sum_{k=3}^{P}a_k(|\mathbf{y}-\mathbf{y}(n)|^2/\sigma)^{m_k}\right]}{\sum_{n=1}^{N-1}\exp[-|\mathbf{y}-\mathbf{y}(n)|^2/\sigma]\left[a_1+\sum_{k=3}^{P}a_k(|\mathbf{y}-\mathbf{y}(n)|^2/\sigma)^{m_k}\right]} . \quad (2)$$

The parameter space $\mathbf{a}$ is P dimensional, $\mathbf{a}=(a_1,a_2,\ldots,a_P)$. σ is a fixed parameter that provides a scale we can use to determine which points in the data set are "close" to $\mathbf{y}$. The m_k's are also fixed at various values. We could treat both σ and the m_k's as parameters to be optimized in the same sense as the $\mathbf{a}$'s. However, we choose not to do this in our work; not for any fundamental reasons, but because we wished to explore other issues and wished to keep down the size of the parameter space over which our minimization searches were performed.

The weight function $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ which we use was arrived at after some experimentation. It, as do many other choices, certainly satisfies our general requirements. These requirements include the following:

The function is sensitive to the presence of near "neighbors" in phase space. Only points $\mathbf{y}(n)$ within a distance from $\mathbf{y}$ of order $\sqrt{\sigma}$ make any sizable contribution to $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$.

When $\sigma\to 0$, $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ becomes essentially a Kronecker delta and the point $\mathbf{y}(n)$ is mapped precisely to $\mathbf{y}(n+1)$.

It is easy to differentiate both in $\mathbf{y}$ and in $\mathbf{a}$. These derivatives are important in the minimization of the cost function using our methods, and having explicit expressions for the required derivatives in either of these independent variables makes the optimization routines run much faster.

In the function we have chosen it is easy to retain many parameters all of the same general form, thus as the number of constraints on the optimization of the cost function is increased, the pattern of our searches remains the same.

The essential function which senses neighbors, namely the exponential, can easily be replaced by other choices, such as those in Table 3.1 of Silverman's monograph.[21] The general form of our arguments goes through then without modification.

By virtue of the term involving a_2 in the numerator, this form of $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ allowed us to satisfy constraints set by the Lyapunov exponents with numerical stability and accuracy. The denominator serves as an approximate counter for the number of neighbors of the point $\mathbf{y}$, so the numerator works less to produce the required average for the forward prediction of the point $\mathbf{y}$. The presence of the denominator assured us of numerical ease in making the parameters in the map $\mathbf{F}(\mathbf{y},\mathbf{a})$ meet our requirement of producing an average over neighborhood points in projecting forward in time any phase-space point. This made the numerical algorithms we use much more efficient and accurate.

The choice of cost function is also rather much up to us. Since we are to think of $\mathbf{F}(\mathbf{y},\mathbf{a})$ as a dynamical system evolving points $\mathbf{y}(n)$ into new points $\mathbf{y}(n+1)$, we should consider asking the map to reproduce accurately from $\mathbf{y}(n)$ not only the "next" point $\mathbf{y}(n+1)$ but, via iteration, a sequence of points $\mathbf{y}(n+1),\mathbf{y}(n+2),\mathbf{y}(n+3),\ldots,\mathbf{y}(n+L)$ up to some L beyond which we simply do not trust the accuracy of our algorithm $\mathbf{F}$ or of the machines we use to compute the future $\mathbf{y}$'s.

This suggests the *predictor* for future points to be a linear combination of iterated powers of the map $\mathbf{F}(\mathbf{y},\mathbf{a})$,

$$\mathbf{y}(m+1)=\sum_{k=1}^{L}X_k\mathbf{F}^k(\mathbf{y}(m-k+1),\mathbf{a}) , \quad (3)$$

where $\mathbf{F}^k$ is the kth iterate of $\mathbf{F}$ as described above. If $\mathbf{F}(\mathbf{y},\mathbf{a})$ were the exact mapping, then each term in the sum over k would be $X_k\mathbf{y}(m+1)$. Thus we require

$$\sum_{k=1}^{L}X_k=1 .$$

The $\mathbf{X}$'s weight the various iterates of $\mathbf{F}$ and are used to determine which iterates of $\mathbf{F}$ we believe are the most accurate. Typically, one would require $X_j\geq X_{j+1}$ to indicate that the lower iterates of $\mathbf{F}$ are believed to be more accurate than the higher iterates. This predictor is a natural generalization to the nonlinear problem of the common linear predictor

$$\mathbf{y}(m+1)=\sum_{k=1}^{L}X_k\mathbf{y}(m-k+1) ,$$

with the clear differences associated with the iterative nature of the map $\mathbf{F}(\mathbf{y},\mathbf{a})$.

This predictor [Eq. (3)] combines both past temporal information from times $m-k+1$; $k=1,2,\ldots,L$ and information from all the phase-space neighbors of the orbit points $\mathbf{y}(m-k+1)$ because of the structure of $\mathbf{F}(\mathbf{y},\mathbf{a})$. The combination of spatial and temporal information provides a significant "lever arm" which permits Eq. (3) to quite accurately make forecasts about the forward evolution of points $\mathbf{y}$ in $\mathbb{R}^d$. By utilizing the phase-space information in $\mathbf{F}(\mathbf{y},\mathbf{a})$ at each temporal step we efficiently tap properties of the full data set.

The cost function associated with this predictor is

$$C(\mathbf{X},\mathbf{a})=\frac{\sum_{n=L}^{N-1}\left[|\mathbf{y}(n+1)-\sum_{k=1}^{L}X_k\mathbf{F}^k(\mathbf{y}(n-k+1),\mathbf{a})|^2\right]}{\sum_{n=1}^{N}|\mathbf{y}(n)\cdot\mathbf{y}(n)|^2} . \quad (4)$$

This kind of cost function will automatically contain information on the Lyapunov exponents which themselves are expressions of the dynamics as iterations of the map.

Some information on the invariant density function on the attractor is also contained in this improved cost function.[24]

Another major consideration to us is the great difference in the coordinate scale of various attractors. The numerator of the cost function [Eq. (4)] is the residual of the mapped function summed over the entire trajectory, and hence gives a measure of the sum of the absolute errors over all the mapped points. Since the absolute error is obviously dependent on the macroscale of the attractor, it is more informative to rescale the final cost function value in some manner which reflects error. In our samples, the scale of the attractor of the Hénon attractor is on the order of unity, while that of the Lorenz attractor is of order 100. Hence some form of rescaling of the cost function became desirable in order to have a relative measure of comparison between two systems with different macroscale. We chose a normalization in the following straightforward manner: we simple summed the magnitudes of the position vectors of all the points on the attractor, and retained this value as a constant. Absolute values for the cost function after normalization by the denominator in Eq. (4) give a more sensible relative measure of the error of our prediction function $\mathbf{F}(\mathbf{y},\mathbf{a})$.

We note that FS suggest forecasting the evolution of a point $\mathbf{y}$ by looking around at the neighbors of $\mathbf{y}$ among the data set $\mathbf{y}(n)$ and observing where these neighbors go under one iteration of the underlying map taking the $\mathbf{y}(n)$ to $\mathbf{y}(n+1)$. They determine the future of the new point $\mathbf{y}$ by an interpolation involving the future of its neighbors. Our mapping function Eqs. (1) and (2) does precisely this as indicated. All points in the data set are given some weight in the future of $\mathbf{y}$, but if $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ falls rapidly for large $|\mathbf{y}-\mathbf{y}(n)|$, as we shall always choose, only members of the data set $\mathbf{y}(n)$ near $\mathbf{y}$, i.e., the neighbors, play much of a role in its future. Our $\mathbf{F}(\mathbf{y},\mathbf{a})$ in that sense is an analytic formulation of the FS idea. More or less weight can be given to the near neighbors by different choices for the function $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$. The Gaussian we work with could be replaced by a Lorentzian or other choices which weight neighbors more.

C. Invariants

With a map and a cost function, Eqs. (1), (2), and (4), we are ready for the constraints. Section III is devoted to a discussion of Lyapunov exponents. In it we first turn to the extraction of the Lyapunov exponents $\lambda_1,\lambda_2,\ldots,\lambda_d$ from the data $\mathbf{y}(1),\mathbf{y}(2),\ldots,\mathbf{y}(N)$. We do not add anything but our own experience to that of many workers who have explored the calculation of λ_i from data. We attempt to convey to the reader an overview of the available methods for determining Lyapunov spectra and a sense of their reliability. Therefore that portion of Sec. III may be skipped by persons with experience. We include it here since determining the λ_i's is an essential step in our plan for determining $\mathbf{F}(\mathbf{y},\mathbf{a})$ and we have chosen to comment on how we have done it rather than refer the reader to the literature. Of course, we do that too. That established, we discuss how to determine these numbers in terms of the $\mathbf{F}(\mathbf{y},\mathbf{a})$. Equating the numerical values for λ_i from the data to their expression in terms of parameters $\mathbf{a}$ in $\mathbf{F}(\mathbf{y},\mathbf{a})$ will constitute our first set of constraints on the minimization of $C(\mathbf{X},\mathbf{a})$.

Section IV contains our discussion of the invariant distribution of points on the attractor. In principle, this quantity, which we called $\rho(\mathbf{y})$, contains an infinite amount of information on the dynamics. A finite data set $\mathbf{y}(n)$ restricts the resolution we have of this information. We have chosen to express this finite amount of information in terms of the projection of $\rho(\mathbf{y})$ on a set of dual basis functions which are a complete set in $\mathbb{R}^d$. Keeping a finite number of these functions is equivalent to a finite resolution view of the complex structure of $\rho(\mathbf{y})$.[19]

One of our contributions in this work is a scheme for choosing the dual basis functions "tuned" to the structure of $\rho(\mathbf{y})$. This allows us to represent our finite resolution of $\rho(\mathbf{y})$ by a small number of terms in an expansion in the *optimal basis functions*.[25–27] By projecting the $\rho(\mathbf{y})$ determined from the data onto these basis functions, we can determine the coefficients of the expansion of $\rho(\mathbf{y})$ in this basis. Similarly, we can project the $\rho(\mathbf{y})$ determined from the map $\mathbf{F}(\mathbf{y},\mathbf{a})$ onto these basis functions and determine the expansion coefficients of the map. Equating the coefficients one determines from the data to the ones determined from the map constitutes our final constraints on the minimization of $C(\mathbf{X},\mathbf{a})$. Furthermore, we show how the components of $\rho(\mathbf{y})$, in this basis, are the elements of the eigenvalue unity eigenvector of a finite-dimensional matrix constructed from $\mathbf{F}(\mathbf{y},\mathbf{a})$ and the dual basis functions.

In Sec. V we describe our implementation of the constrained minimization program[28] for two model systems: the Hénon map of the plane to itself and (2) the Lorenz system. In each case we numerically generate a data set of $x(n)$'s. We then discuss in some detail our experience in establishing the dimension of the space in which the dynamics is embedded. We also discuss the calculation of Lyapunov exponents, and aspects of the invariant distribution on the attractor from the $\mathbf{y}(n)$'s. Finally, we carry out the constrained minimization of the cost function and indicate how well our parametrized mappings are able to perform in predicting orbits other than those in the given data set.

In this paper we are attempting to describe a method of analyzing *experimental* data. For such a situation we do not know *a priori* the correct embedding dimension, the correct Lyapunov exponents, or the underlying dynamical system that can be used to generate the correct invariant distribution. Yet we have used data sets generated by a dynamical system that we know. We have used known systems for two reasons. The first is that it provides a simple way to obtain large, noise free, data sets. Second, it provides a way of measuring how well existing techniques are able to determine the embedding dimension and the Lyapunov exponents. In order to simulate experimental systems we treat the data set as having come to us from an unknown source. Thus we do not use any of the known properties of either the Hénon or the Lorenz system.

An issue of some importance we do not address in this paper is that of extrinsic *noise* which could contaminate

our signal $x(n)$. This is not a dismissal of this important issue but an attempt to separate out the matters of efficiency and utility of our plan for prediction on strange attractors from issues concerning the practical degradation of our procedures by external noise. An equally important issue is the quantity of data available. The determination of Lyapunov exponents is very difficult for short data sets. As we have stated above the resolution of $\rho(\mathbf{y})$ is determined by the number of data points available. As the dimension of the phase space increases, the amount of data necessary for accurate prediction increases dramatically. We will return to the implications of noisy and/or short data sets for our prediction procedure in later work. For now we assume that we are given essentially noise-free, arbitrarily long time series $x(n)$.

It is our expectation that our experiences with the two systems listed above will give us the ability, in many instances, to construct models in the form of our parametrized mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$ which allow prediction and control of the underlying nonlinear dynamical system producing an observed signal $x(n)$. The details of the $\mathbf{F}(\mathbf{y},\mathbf{a})$ for a specific application should reflect the known features of the physical or other phenomena giving the signal. It seems too bold, if at all possible, to suggest any general rules for choosing forms for $\mathbf{F}(\mathbf{y},\mathbf{a})$. This is sure to be a rich area for experimentation and our own choice will be motivated by considerations we shall defend in a later section and slightly alluded to above.

The matter of noise will be addressed in a future paper. Our methods for dealing with noise follow those outlined by Fuller[29] and seem similar to the ideas of Sidorwich.[30]

II. CHOICE OF AN EMBEDDING SPACE

In this section we illustrate how one can determine the phase-space embedding dimension d from the scalar time series $x(n)$, $n=1,\ldots,N_D$. We assume that the data set is long enough that we need not be concerned with statistical issues about the numerical accuracy of the quantities we consider below. We also assume extrinsic noise is absent from the $x(n)$'s when we receive them. Matters of short and/or noisy data sets, while critical in all applications, are addressed only peripherally in this paper.

Following the work of Packard *et al.*[5] and Mañé and Takens[6,7] and the developmental work of numerous others we seek a set of lagged variables $x(n),x(n+\tau_1)$, $x(n+\tau_2),\ldots,x(n+\tau_{d-1})$ which act as the coordinates in a d-dimensional space in which the dynamics producing the $x(n)$'s is fully captured or embedded.

The choice of lags τ_a is not a well agreed upon matter.[31] The issue is the accuracy and efficiency with which the d-dimensional vectors that result from a particular choice of τ_a's represents the phase space in which the attractor resides. If the underlying system were a differential equation and a scalar variable $x(t)$ were measured at discrete times $x(n)=x(t_0+n\Delta t)$, then we are by the choice of lagged variables trying to find a discrete replacement for the usual phase-space coordinates $x(t),dx/dt,\ldots,d^{d-1}x/dt^{d-1}$. Mañé and Taken's results indicate that, in principle, any choice of lags τ_a will do. We adopt the practice of choosing a single lag τ and making all other lags multiples of τ. The question of what is the best way to choose τ is still open. In a heuristic sense, if τ is too small, then the coordinate at $x(n+\tau)$ and $x(n+2\tau)$ represent almost the same information. Similarly, if τ is too large, then $x(n+\tau)$ and $x(n+2\tau)$ represent distinct uncorrelated descriptions of the embedding space.

For reasons of consistency and ease in calculating Lyapunov exponents (cf. Sec. III) we adopt the following practice. We take the original scalar measurements and calculate its autocorrelation function

$$\frac{1}{T}\int_0^T x(t+\tau)x(t)dt\ .$$

We then choose τ to be approximately $\frac{1}{10}$ to $\frac{1}{20}$ the time associated with the first local minimum of the autocorrelation function. We find that this system, although somewhat arbitrary, works well in practice and provides a simple and systematic way of determining τ. We set τ to unity and thereby establish a time scale for the problem. The data $x(n)$, $n=1,\ldots,N_D$ thus become measurements of the scalar variable separated by a constant time step τ.

We then form d vectors

$$\mathbf{y}(n)=(x(n),x(n+1),\ldots,x(n+d-1)) \tag{5}$$

for $n=1,2,\ldots,\ N=N_D-d$ in a space $\mathbb{R}^d$ capturing the geometric structure of the attractor on which the orbits $x(n)$ lie. To establish d we need some characteristic of the attractor that becomes unchanging as d becomes large enough, thus indicating that the attractor can be embedded in $\mathbb{R}^d$. The usual Hausdorff or other *dimensions* of the attractor are such characteristic quantities. Numerical calculations of the Hausdorff dimension $d_A(N,d)$ of an attractor may depend on the finite length of the data set N and/or the embedding dimension d. For N large enough d_A will become independent of d when the attractor is properly embedded in $\mathbb{R}^d$. Operationally one increases d until d_A remains constant and identifies the minimum d where d_A "saturates" as the embedding dimension.

In fact, we, along with numerous others, do not recommend the computation of d_A, however geometrically appealing it may be, because it is too demanding of computer time. We suggest, and we use, the properties of the correlation function $D(r)$, proposed by Takens[32] and by Grassberger and Procaccia,[33] which is much easier to compute.[34] In terms of the data vectors $\mathbf{y}(n)$ this is defined to be

$$D(r,N,d)=\frac{2}{N(N-1)}\sum_{i=1}^{N}\sum_{j=1}^{N}\Theta(r-|\mathbf{y}(j)-\mathbf{y}(i)|)\ ,$$
$$i\neq j \tag{6}$$

where $\Theta(x)$ is the Heaviside function $\Theta(x>0)=1$ and $\Theta(x<0)=0$. The vertical bars represent some measure of distance in $\mathbb{R}^d$—we use the Euclidean norm, but that is only a convenient choice. This correlation function counts the points of the attractor within a distance r of each other. Thus it possesses much of the same geometri-

cal content as the Hausdorff or other invariant dimension attributes.

For N large enough the behavior of $D(r,N,d)$ for small r becomes independent of N. As one would expect from scaling arguments about fractals, as well as observationally, $D(r,N,d)$ is seen to take the form

$$D(r,N,d)=\Phi(r,d)r^{\nu(d)}$$

for small r and large N.[35]

We will identify as the embedding dimension that value of d where the structure in $D(r,N,d)$ becomes independent of d. In this regime it is sufficient that $D(r,N,d)$, becomes independent of d over a range of r near $r\to 0$, and large N [$r=0$ in a finite data set always gives strictly zero for $D(r,N,d)$ and is an uninteresting limit].

To illustrate the use of the correlation function as an embedding dimension discriminant we have calculated $D(r,N,d)$ for very long time series taken from the two examples we will be working with in this paper: (i) the Hénon map of the plane to itself,[36]

$$\begin{aligned} x_1(n+1)&=1.0-ax_1(n)^2+x_2(n)\ , \\ x_2(n+1)&=bx_1(n)\ , \end{aligned} \tag{7}$$

with conventional parameter values $a=1.4$ and $b=0.3$, and (ii) the Lorenz system of three autonomous differential equations[37]

$$\begin{aligned} \frac{dx_1(t)}{dt}&=\sigma(x_2(t)-x_1(t))\ , \\ \frac{dx_2(t)}{dt}&=-x_1(t)x_3(t)+rx_1(t)-x_2(t)\ , \\ \frac{dx_3(t)}{dt}&=x_1(t)x_2(t)-bx_3(t)\ , \end{aligned} \tag{8}$$

with parameter values $\sigma=16$, $b=4$, and $r=45.92$.

For the Hénon map we took an initial condition lying in its basin of attraction and iterated the map 4550 times. The first 50 iterates were discarded as representing transient behavior, while the last 4500 points of $x_1(n)$ and $x_2(n)$ were then used to make d vectors

$$\mathbf{y}_i(n)=(x_i(n),x_i(n+1),\ldots,x_i(n+d-1))$$

for $i=1$ or 2. For $d=1,2,\ldots,5$ $D(r,N,d)$ was computed using an efficient code developed by Theiler.[34] For $\mathbf{y}_1(n)$ data these $D(r,N,d)$ are plotted in Fig. 1. A similar plot was generated for $\mathbf{y}_2(n)$, but is not shown. Because of the simplicity of the connection $x_2(n+1)=bx_1(n)$ in the Hénon map, these two views of $D(r,N,d)$ are really redundant. However, in the spirit of treating each data series as having originally come to us from a source whose underlying dynamics is unknown we performed both calculations.

While a cautious and careful observer might say the embedding dimension for the $\mathbf{y}_1(n)$ data would be $d=3$, we feel safe in concluding from these figures that $d=2$. Computations with N greater than 4500 support this conclusion.

Further, if we take the $x_1(n)$ data and plot the two-

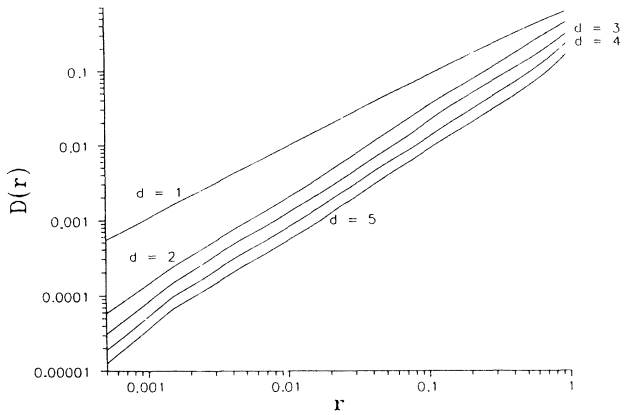

FIG. 1. $D(R)$ vs r for the Hénon map. $\mathbf{y}_1(n)=(x_1(n), x_1(n+1))$ for 4500 points.

vectors $\mathbf{y}_1(n)=(x_1(n),x_1(n+1))$, we reconstruct the figure seen in Fig. 2. This is, as should not be surprising in this simple example, a rotated form of the Hénon attractor. The usual display of the Hénon attractor is obtained by plotting $(x_1(n),x_2(n))$ for our data. Since $x_1(n)$ is $(1/b)x_2(n+1)$, the coincidence of these plots is certainly not remarkable. Our goal in presenting this kind of detail is as a guide to what one might expect in more complicated examples rather than as revelations about the Hénon map.

Next we turn to the Lorenz equations. Once again we chose initial conditions in the basin of attraction and solved Eqs. (8) with a straightforward fourth-order Runge-Kutta ordinary differential equation (ODE) solver with a fixed time step. After discarding the first 50 time steps as transients, we recorded x_1, x_2, and x_3 for $N=4500$ corresponding to many natural cycles of the orbit around the attractor. From each of the three data sets we formed the d vectors as in Eq. (5) and with them evaluated the correlation function $D(r,N,d)$ for $d=1,2,\ldots,5$. The $D(r,N,d)$'s for $\mathbf{y}_1(n)$ data are shown

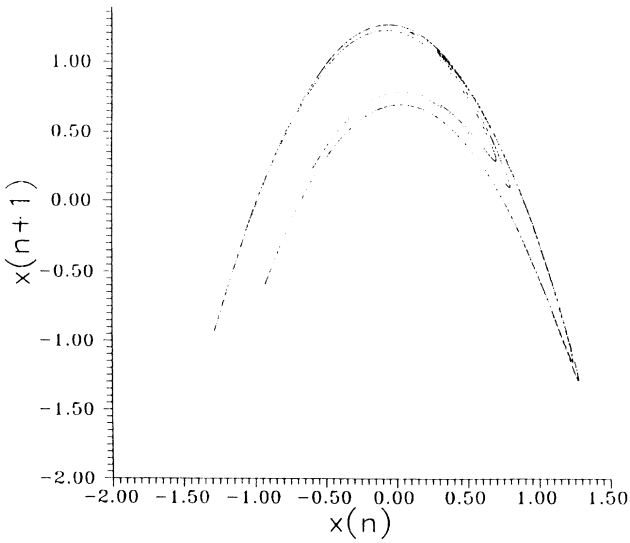

FIG. 2. Hénon attractor $x_1(n)$ plotted against $x_1(n+1)$.

in Fig. 3. An embedding dimension of $d=3$ is fairly clearly a safe choice for these data. A bolder choice would have been $d=2$. Since it is known that the Hausdorff dimension of the Lorenz attractor is just above 2 in this regime of parameter space, this would have been a convincing, although incorrect, choice. The message here is that choosing d too large entails extra subsequent computation, but no loss of information on the attractor. It is probably safer to live with a d one dimension too large as a general matter of care. We thus choose $d=3$. The results of the $\mathbf{y}_2$ and $\mathbf{y}_3$ data are not shown. As with the Hénon example the results of $\mathbf{y}_2$ and $\mathbf{y}_3$ are similar to those of $\mathbf{y}_1$. The fact that the $\mathbf{y}_1$, $\mathbf{y}_2$, and $\mathbf{y}_3$ vectors for the Lorenz data ($\mathbf{y}_1$ and $\mathbf{y}_2$ for Hénon) yields similar results is to be expected since all three measurements evolve on the same attractor.

Next we plot the points $\mathbf{y}_1(n)=(x_1(n),x_1(n+1),x_1(n+2))$ in the three-dimensional embedding space. These are shown in Fig. 4 as a projection on a plane with normal vector $\hat{\mathbf{n}}=(\cos\theta)\hat{\mathbf{x}}_1(n)+(\sin\theta)\hat{\mathbf{x}}_1(n+1)+0\hat{\mathbf{x}}_1(n+2)$ for $\theta=1.31$. We note the similarity between Fig. 4 and the well-known shape of the Lorenz attractor. Thus the method of *phase-space embedding* reliably reproduces the Lorenz attractor. For the two examples we have used the reconstructed attractor is similar in appearance to the attractor generated by the "true" underlying equations of motion. In general, the reconstructed attractor will not have this visual similarity. However, the reconstructed attractor will contain all of the important invariant information as the true attractor. The difference in visual shapes is the result of a nonlinear change of variables between the true dynamical variables and the reconstructed variables.

We close this section with a summary note reminding the reader that our use of the correlation integral Eq. (6) has been to establish an embedding dimension d in which to view the system attractor described by our time series $x(n)$. We chose $D(r,N,d)$ because it is familiar, easy to compute, and has a clear geometrical meaning. For us it

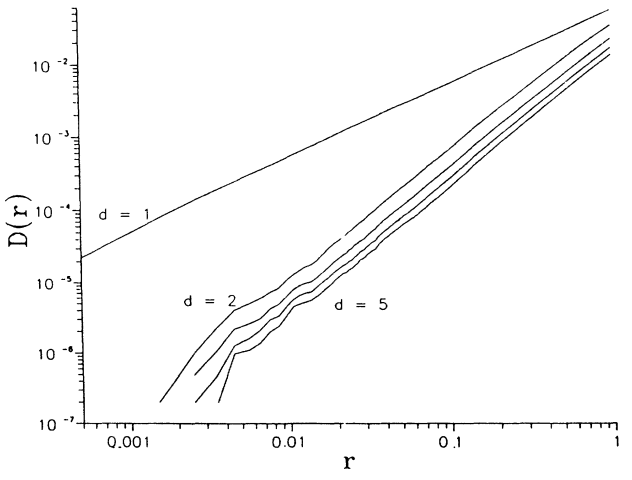

FIG. 3. $D(r)$ vs r for the Lorenz equations, $\mathbf{y}_1(n)$ for 4500 points and embedding dimensions $d=1,\ldots 5$. For this case $r=45.92$, $b=4.0$, and $\sigma=16$.

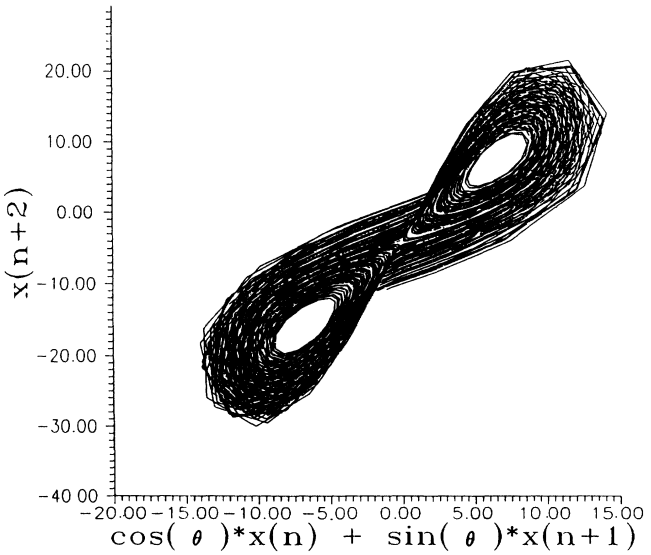

FIG. 4. Lorenz attractor created from $x_1(n)$ data. The parameter values are $r=45.92$, $b=4.0$, and $\sigma=16$, while the projection angle is $\theta=1.31$.

is a diagnostic tool. While the details of the small r behavior $D(r,d)\sim r^{\nu}\Phi(r)$ contains important information about the dynamics, we do not focus on that here. Indeed, we are quite happy to accept other diagnostic tools in its place.

III. LYAPUNOV CHARACTERISTIC EXPONENTS—FROM DATA AND FROM THE MAP

In this section we discuss how one determines the Lyapunov exponents that govern a dynamical system. First we discuss how to extract them from an experimental data set and then from our mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$. By choosing the parameters $\mathbf{a}$ in such a way that $\mathbf{F}(\mathbf{y},\mathbf{a})$ yields the same Lyapunov exponents as the experimental data set, we are forcing a constraint on $\mathbf{F}(\mathbf{y},\mathbf{a})$ that is not explicitly required by minimizing the cost function given by Eq. (4). This local constraint should improve our ability to predict the short-term (and possibly long-term) evolution of points that are not in the data set, but near the attractor. Certainly points outside the basin of attraction of the attractor we have observed in the original data set will not evolve according to our $\mathbf{F}(\mathbf{y},\mathbf{a})$.

Rather than writing our own computer program, and thereby become embroiled in the controversy of what is the best way to determine Lyapunov spectra from an experimental time series, we have chosen to use methods that have already been proposed by two different research groups. By comparing the results of both methods we hope to improve our confidence in the spectra given by each of them separately. The first method we shall report on was developed by Eckmann *et al.*[38] The second method was developed by Wolf *et al.*[39] Finally, we will show how we calculated the Lyapunov spectra from our mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$.

The choice of an appropriate data set for use in either the Eckmann *et al.* or the Wolf *et al.* method is some-

thing that cannot be overstressed. As stated in Sec. II the time lag τ between successive measurements of the dynamical variable must be appropriately chosen, if one wants optimal results.

A. Eckmann-Kamphorst-Ruelle-Ciliberto method

For the Eckmann *et al.* method the FORTRAN source code we used when performing our numerical experiments on the dynamical systems denoted in Sec. II was provided by the authors of Ref. 38. It assumes that the input is a string of positive integer data whose sampling rate is τ. [The temporary conversion of the data set $x(n)$, $n=1,\ldots,N_D$ to positive integers for the sake of the Lyapunov calculation should not be a difficult matter.] The code reads the data set and embeds it in a d-dimensional space in the manner specified in Sec. II. The result is a set of $N=N_D$ data vectors $\mathbf{y}(n)=(x(n),x(n+1),\ldots,x(n+d-1))$ where we have normalized τ to unity [cf. Eq. (5)]. It then chooses an initial $\mathbf{y}(n)$ and finds all neighbors of $\mathbf{y}(n)$ within a radius r. These points, as well as their forward images, are used to construct a linear mapping $\mathbf{T}$ from time n to time $n+1$. The Lyapunov exponents are related to the eigenvalues of the successive iterates of the map $\mathbf{T}$. For a detailed discussion of the algorithm we direct the reader to Ref. 38.

The Eckmann *et al.* method assumes that the embedding dimension d is related to the number of Lyapunov exponents via the rule $d=(d_m-1)M+1$, where d_m is the number of Lyapunov exponents and M is a positive integer. By allowing d_m and M to range over various values a wide range of embedding dimensions is used. We remark that the reader will recall that in Sec. II we established a method for determining the minimum embedding dimension d. The data vectors $\mathbf{y}(n)$ are assumed to live on some attractor that occupies some portion of $\mathbb{R}^d$. It is a numerically difficult exercise to calculate Lyapunov exponents from data. Thus it is necessary to examine a wide range of possible embedding dimensions d. It is our experience that the calculated values of the exponents converge onto their correct values as d is increased above the number specified by methods in Sec. II. We report numerical experiments for d_m in the range between 2 and 9 for $M=1,2$. (We remark that $M=1$ recovers $d=d_m$, while $M=2$ is slightly below the Takens and Mañé limit.[6,7]) In all of our tests we iterated the tangent map $\mathbf{T}$ 2000 times before evaluating the Lyapunov exponent.

To get a feel for the proper densities of points on the reconstructed attractor, it is useful to use diagnostics such as, say, a histogram of the number of neighbors falling within a range around the smallest nearest-neighbor distance on the attractor. If the density of points on an attractor is quite inhomogeneous, much higher mean point densities are often necessary to insure that most points have at least a few nearby neighbors. Often a useful diagnostic is simply to plot out the reconstructed attractor, and visually obtain an intuitive feel for how homogeneous the point density is. As a general rule of thumb (inspired by Wolf *et al.*), we find empirically that the minimum number of points required for the prediction algorithm to go as something like 20^d, where d is the dimension of the embedding space, although this is probably an overestimate when d is 4 or more.

The first dynamical system for which we present results is the Hénon map of the plane given by Eqs. (7). We used a data set with $N=10\,000$ entries. The results are shown in Fig. 5 and Table I. As one can see, the numerical experiments accurately predict the accepted value of the positive Lyapunov exponent $\lambda_1=0.418$. Although for the $M=1$ case the computer code produced a reasonably accurate prediction of the negative Lyapunov exponent, the code, in principle, will not yield accurate values of the negative or zero Lyapunov exponents. This fact is born out in the $M=2$ case (which is not shown). Furthermore, we know of no method that will produce negative Lyapunov exponents from an experimental data set. Since we are unable to reliably determine the negative Lyapunov exponents from the data, we will not constrain $\mathbf{F}(\mathbf{y},\mathbf{a})$ to reproduce the negative values of the spectra.

It should not be surprising that we are unable to determine the negative Lyapunov spectra using our data sets. We have assumed that the data describe motion *on* an attractor. The negative Lyapunov exponents indicate how points in the phase space that are *off* the attractor get onto the attractor. The portion of the data set that might reveal how points off the attractor get to the attractor is the initial transient. This transient is typically very short (sometimes as few as 10 time steps τ) and is usually discarded or otherwise unavailable.

A related issue to be addressed is that the code produces d_m exponents regardless of the actual number of Lyapunov exponents that govern the dynamics of the

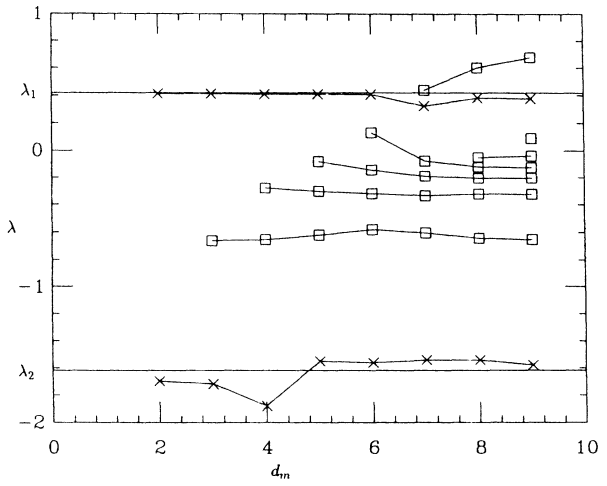

FIG. 5. The results of calculating Lyapunov exponents by the Eckmann *et al.* method for Hénon data. The horizontal axis is d_m, the assumed dimension of the dynamical system that produced the data set. Thus the method will produce d_m Lyapunov exponents. The vertical axis contains the numerical values calculated for the d_m different λ's. The two horizontal lines are the known correct values for $\lambda_1=0.418$ and $\lambda=-1.62$. The method relates d_m to the embedding dimension d via $d=(d_m-1)M+1$. This figure shows results for $M=1$. Spurious exponents are labeled with squares while dynamical exponents are labeled with X's.

TABLE I. Lyapunov exponents for the Hénon attractor $M=1$, and the number of data points is 10 000.

d_m	λ				
2	$\lambda_1=0.412$	$\lambda_2=-1.70$			
3	$\lambda_1=0.412$	$\lambda_2=-0.662$	$\lambda_3=-1.72$		
4	$\lambda_1=0.408$	$\lambda_2=-0.281$	$\lambda_3=-0.655$	$\lambda_4=-1.88$	
5	$\lambda_1=0.408$	$\lambda_2=-0.0824$	$\lambda_3=0.305$	$\lambda_4=-0.622$	$\lambda_5=-1.55$
6	$\lambda_1=0.407$	$\lambda_2=0.128$	$\lambda_3=-0.144$	$\lambda_4=-0.321$	$\lambda_5=-0.581$
	$\lambda_6=-1.56$				
7	$\lambda_1=0.437$	$\lambda_2=0.323$	$\lambda_3=-0.0767$	$\lambda_4=-0.190$	$\lambda_5=-0.335$
	$\lambda_6=-0.604$	$\lambda_7=-1.54$			
8	$\lambda_1=0.602$	$\lambda_2=0.382$	$\lambda_3=-0.0509$	$\lambda_4=-0.118$	$\lambda_5=-0.203$
	$\lambda_6=-0.332$	$\lambda_7=-0.642$	$\lambda_8=-1.54$		
9	$\lambda_1=0.677$	$\lambda_2=0.377$	$\lambda_3=0.0896$	$\lambda_4=-0.0390$	$\lambda_5=-0.124$
	$\lambda_6=-0.203$	$\lambda_7=-0.324$	$\lambda_8=-0.652$	$\lambda_9=-1.58$	
Accepted values of λ		$\lambda_1=0.418$		$\lambda_2=-1.62$	

physical system in question. However, it is relatively easy to determine which of the d_m exponents govern the dynamics of the system and which are spurious. We assume that one has successfully determined the minimum embedding dimension of the attractor by the method we presented in Sec. II (or any other reliable method at the reader's disposal). Examination of Fig. 5 indicates that most of the spurious exponents are negative. These negative exponents are necessary to contract the d-dimensional phase space onto the attractor whose dimension is $d_A < d$. The one positive spurious exponent appears at $d_m=7$ for the $M=1$ case. We know from Fig. 1 that the dynamics of the Hénon attractor can be embedded in two dimensions. Hence we conclude that an exponent that exist only for $d_m \geq 7$ must be spurious. The origin of this spurious positive exponent is discussed by Eckmann *et al.*[38] It is believed it will stabilize at a value of $2\lambda_1$.

We have averaged the calculated values of λ_1 for the $M=1$ case over the range $d_m=2$–6. We discarded the values of λ_1 for $d_m \geq 7$ since they have obviously been altered by the spurious Lyapunov exponent generated at $d_m=7$. We find that the average value is $\bar{\lambda}_1=0.409$, which differs from the accepted value of 0.418 by only 2%. For the $M=2$ case we found similar results. After averaging we find that $\bar{\lambda}_1=0.420$. In conclusion, we state that by comparing the $M=1$ and 2 cases we feel that the code successfully determined the positive Lyapunov exponent associated with the Hénon attractor.

We now turn our attention to the second dynamical system we wish to analyze, the Lorenz system of ODE's given by Eqs. (8). The data set used for our numerical experiments consisted of $N=20\,000$ entries and was generated by integrating Eqs. (8) forward in time using a simple fourth-order Runge-Kutta routine with a fixed time step. We chose to record the $x_1(t)$ variable, although either the $x_2(t)$ or $x_3(t)$ variable would do as well. Figure 6 is a graph of the autocorrelation function. The first minimum is at $n \sim 12$ where n is the number of Runge-Kutta time steps of length 0.03. The time associated with this first minimum is approximately $t_c \sim 0.36$. We use a sampling rate $\tau=0.03$, which is approximately $\frac{1}{12}$ of the autocorrelation time. Thus we use every Runge-Kutta data point as our experimental data set. We allowed d_m to range between 2 and 9 for $M=1$ and 2. The results of our numerical experiments are shown in Fig. 7 and Table II.

For all cases $M=1$ and 2, we are able to accurately determine both the positive and the zero Lyapunov exponent. The accepted value of λ_1 is 1.50. The average of the calculated values of λ_1 for $d_m \geq 5$ in the $M=1$ case is $\bar{\lambda}_1=1.45$, which is an error of only 3%. As with the Hénon example, we found better results for the $M=2$ case.

The question of a zero Lyapunov exponent requires special consideration. Any dynamical system that is represented by an ODE will contain a zero Lyapunov exponent. As can be seen from Fig. 7 and Table II, one of the Lyapunov exponents calculated from the experimental data set is very small (as much as two orders of magnitude) compared to λ_1. We also notice that this exponent is very stable and very persistent. It exists for $M=1$ and 2 over the entire range of d_m. Given this behavior and

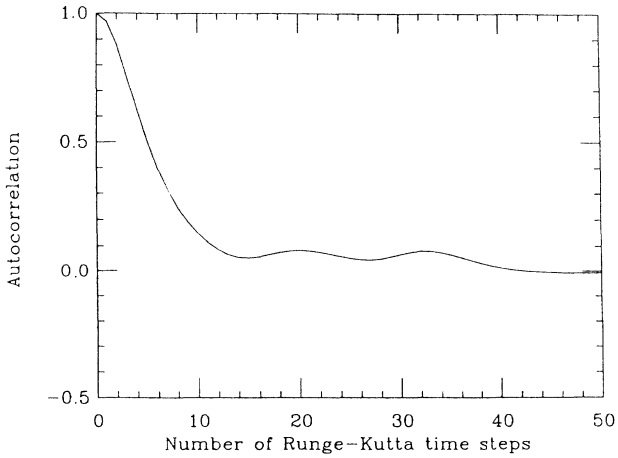

FIG. 6. Autocorrelation function for $x_1(t)$, the Lorenz system from 20 000 data points.

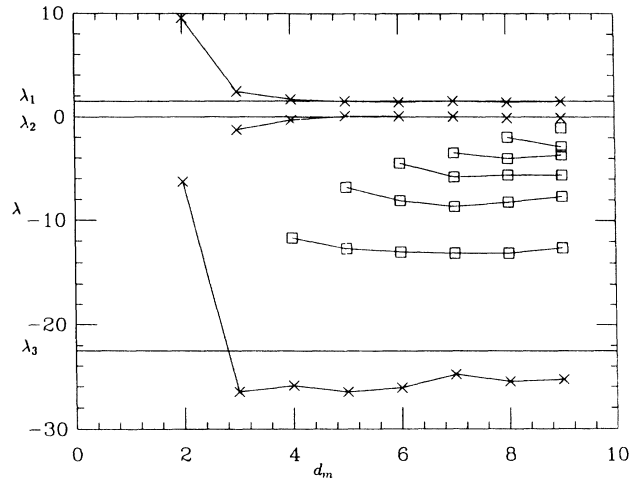

FIG. 7. The results of applying the Eckmann *et al.* method of calculating Lyapunov exponents to a Lorenz data set. The results shown are for $M=1$. For this dynamical system there are three dynamical exponents at $\lambda_1=1.50$, $\lambda_2=0.0$, and $\lambda_3=-22.5$.

the prevalence of ODE's as dynamical systems, we feel confident in predicting a zero Lyapunov exponent.

Of course, we have the luxury here of *knowing* that our data set came from an ODE. This type of knowledge concerning the origin of a data set is typically unavailable. Thus we must use our best judgment and live with the fact that we cannot know for *certain* whether a Lyapunov exponent generated by the Eckmann *et al.* method should be interpreted as zero or just very small. Our recommendation is that one compare the suspected zero exponent to the smallest nonspurious positive Lyapunov exponent generated by the code. If the suspected exponent is as persistent, as stable, and more than a factor of 25 smaller than the smallest positive exponent, we recommend that the suspected exponent be assigned the value zero.

B. Wolf-Swift-Swinney-Vastano method

A second technique that we have investigated to determine Lyapunov exponents from time series is due to Wolf, Swift, Swinney, and Vastano (WSSV).[39] This paper presents two algorithms, one for determining the full Lyapunov spectrum from a known set of dynamical equations, and one for determining only the largest positive exponent if one has available only a time series from the dynamical system. Since the paper includes the source codes for the two algorithms, we copied and used them directly. The WSSV code for time series analysis can only determine the largest positive exponent. We have up to now chosen to use only one Lyapunov exponent as a constraint to the nonlinear fitting method, and so this program proves sufficient for our needs. Given the current difficulty of accurately determining other exponents from a time series of data, we restrict the constraints to one Lyapunov exponent. In addition to these considerations, the WSSV code is exceptionally easy to use, and requires relatively minimal amounts of data.

The WSSV code for time series works in a manner somewhat similar to other techniques which attempt to approximate in some way the local tangent space about a fiducial orbit. In this case, the code initially constructs the time-delay reconstructed coordinates for the system in the usual manner, taking the parameters for the reconstruction as input to the program. The calculation of the Lyapunov exponent then begins by finding the nearest neighbor in the reconstructed phase space to the first point of the orbit, where "nearest" is measured using the usual Euclidean metric. Once this point is found, the magnitude of the difference vector between the two points is recorded. The algorithm then proceeds by evolving the fiducial point along its trajectory, and the neighboring point along its trajectory, a given number of steps of the time series. The magnitude of the final separation between the two points is then determined, and the contribution to the Lyapunov exponent is then simply given as the logarithm of the final separation divided by the initial separation, divided by the time interval of evo-

TABLE II. Lyapunov exponents for the Lorenz attractor $M=1$, and the number of data points is 20 000.

d_m	λ				
2	$\lambda_1=9.54$	$\lambda_2=-6.30$			
3	$\lambda_1=2.42$	$\lambda_2=-1.27$	$\lambda_3=-26.5$		
4	$\lambda_1=1.68$	$\lambda_2=-0.308$	$\lambda_3=-11.7$	$\lambda_4=-25.9$	
5	$\lambda_1=1.47$	$\lambda_2=0.0619$	$\lambda_3=-6.84$	$\lambda_4=-12.7$	$\lambda_5=-26.5$
6	$\lambda_1=1.40$	$\lambda_2=0.0471$	$\lambda_3=-4.50$	$\lambda_4=-8.12$	$\lambda_5=-13.0$
	$\lambda_6=-26.1$				
7	$\lambda_1=1.50$	$\lambda_2=0.0141$	$\lambda_3=-3.49$	$\lambda_4=-5.81$	$\lambda_5=-8.69$
	$\lambda_6=-13.1$	$\lambda_7=-24.8$			
8	$\lambda_1=1.40$	$\lambda_2=-0.105$	$\lambda_3=-1.96$	$\lambda_4=-4.02$	$\lambda_5=-5.62$
	$\lambda_6=-8.24$	$\lambda_7=-13.1$	$\lambda_8=-25.5$		
9	$\lambda_1=1.48$	$\lambda_2=-0.109$	$\lambda_3=-1.06$	$\lambda_4=-2.88$	$\lambda_5=-3.68$
	$\lambda_6=-5.63$	$\lambda_7=-7.70$	$\lambda_8=-12.6$	$\lambda_9=-25.3$	
	Accepted values of λ		$\lambda_1=1.50$	$\lambda_2=0.00$	$\lambda_3=-22.5$

lution. These contributions are then averaged over the length of the time series.

This simple scheme works to provide the largest Lyapunov exponent because, given arbitrary initial conditions for the two neighbors and an appropriately long evolution time, the exponential growth due to the largest positive exponent dominates the overall behavior of the difference vectors, so that to good accuracy the net change in the magnitude of the two vectors reflects almost solely this rate of growth. Note that a technical problem exists here, in that the Lyapunov exponent is defined only in terms of the linearized equations of motion about the fiducial trajectory, and the exponential divergence of neighboring trajectories can rapidly drive them out of the linear regime. In the WSSV code, this problem is addressed in a straightforward manner; i.e., when the distance between the neighbors becomes too large, the algorithm abandons this point and searches for a new neighbor. A suitable new neighbor is chosen on the basis of two criteria: first, the point must again be as close to the fiducial trajectory point as possible, and second, the orientation of the abandoned difference vector must be preserved as nearly as possible. The process of choosing a new neighbor using these criteria is thus approximately equivalent to rescaling the difference vector to a much smaller size. In practical terms there is a trade-off between choosing points which are very close to the fiducial point and points whose difference vectors lie nearly along the ray defined by the abandoned difference vector. This problem is handled internally in the code by a multistep search algorithm. Once a suitable new neighbor is determined, the new difference vector is then evolved until it too becomes too large, and then the process is repeated.

Because this numerical procedure is relatively straightforward, there are actually few variables necessary as input to the algorithm, and hence the program is much easier to use than other Lyapunov exponent algorithms. There are seven basic variables which must be set to perform an analysis of a data set, most of which are determined when one calculates the embedding dimension as in Sec. II. The first four variables, which are related to the time-delay reconstruction, are the number of points in the data set (N), the embedding dimension d, reconstruction time delay τ_d, and the sampling rate for the data T_s. The first of these variables N is usually fixed when an experimental time series is being analyzed, although some criterion for the minimum number of points necessary for a good estimate of the Lyapunov exponent can be given. Wolf, Swift, Swinney, and Vastano give a general rule for the minimum number of data points as at least 10^d, although this value can depend on the topology of the attractor and the relative magnitudes of the Lyapunov exponents. Our experience has shown that at least twice this number of points is usually necessary for two significant figures of accuracy, and greater accuracy can require much longer time series. It should be noted that in terms of the algorithm, longer time series are required not just to improve the convergence by providing more contributions to the Lyapunov value; longer time series also provide a higher density of points on the attractor and hence there are more nearby neighbors to choose from when replacements are necessary, making this process more accurate.

The embedding dimension parameter d is the dimension of the time-delay reconstructed vectors $\mathbf{y}(n)$, and is determined as in Sec. II. As discussed there, the dimension of the embedding space must be sufficiently large to ensure that none of the dynamical information about the attractor is lost; however, needlessly large values of the embedding dimension results in greatly increased computation time for the Lyapunov calculation and also increased sensitivity to noise. For the example systems that we have investigated using this method, we have chosen the embedding dimension to be the next highest integer dimension to the (known) fractal dimension, although for experimental data, where one is not sure of the fractal dimension, one may often feel safer to choose a larger value.

The second variable that is necessary for the time-delay reconstruction in the program is the actual time delay value τ_d. This variable, as discussed in Sec. II, gives the time separation of the components of the d vectors in terms of the number of iterates of the time series, and can be thought of as being chosen to make the d components as "orthogonal" as possible. For dynamical time series derived from a mapping, as for the Hénon system, this value can be chosen to be 1, since each iterate generally represents one entire "orbit" on the attractor of the flow that the mapping is derived from. For continuous phase-space flows, as for the Lorenz system, one can often use the rule of thumb given by $d\tau_d = 1$, where d is the embedding dimension and τ_d is here given as the fraction of the orbital period on the attractor, which must then be expressed in time series steps. Another more sophisticated method is to take τ_d as roughly the first zero of the autocorrelation function for the time series. The choice of method for determining the time delay is not crucial, however, since the reconstructed dynamics is generally not strongly dependent on the exact value as long as it is within a reasonable range of the correct value.

The fourth variable T_s is the time between successive measurements in the time series, or rather the inverse of the sampling rate. This value is not actually a variable, but rather an additional piece of information that must be supplied with any time series, and is used in the algorithm to rescale the Lyapunov exponents by setting the time scale for the rate of divergence of the trajectories. Although one may have no control over the sampling rate for an arbitrary set, for systems where one does have control this parameter is an important issue, and can greatly affect the quality of data. Many of the aspects of problems that can arise are from improper sampling rates are discussed by Mayer-Kress.[40]

Two of the input variables to the algorithm are concerned with setting length scales for the reconstructed dynamics. The parameter $S_{\max}$ controls the maximum distance that the algorithm will look for neighbors when it attempts replacement. Since we take a rough value for the limit of the validity of the linear approximation to be about 1% of the macroscale of the attractor, the value of

S_{max} should be taken at somewhat less than this value. Of course, making this parameter smaller will increase accuracy; however, the density of points on the attractor determines how small it can be, and making S_{max} too small also has the unfortunate effect of greatly increasing the computation time of the algorithm. It is instructive to do some experimentation with the effect of this variable when analyzing a data set, however, we have found that the 1% rule is usually a good guess. The second scale variable is S_{min}, which sets the minimum distance that the algorithm will look for neighbors during replacement. The purpose of this parameter is to reduce the effects of very small levels of noise by eliminating the choice of neighbors which are so close that they are within the scale of distance that the noise defines. Since we deal with "clean" data sets throughout the discussion in this paper, the value of S_{min} was set quite low. For actual experimental data corrupted by noise, a good deal of experimentation with this variable is probably necessary, as it is difficult to estimate the effective scale that the noise will appear on. It should be noted that this parameter is only effective at reducing the effects of very small magnitudes of noise, as we have found that S_{min} can usually be not much larger than about 1% of S_{max}, or the algorithm has difficulty finding sufficient numbers of neighbors within the linear regime for replacement.

The last input parameter to the algorithm is T_E, which gives the evolution time (in time series steps) that a given pair of neighbors are allowed to evolve before replacement. The value of this variable can greatly effect the accuracy of the calculation of the Lyapunov exponent, for a number of reasons. If the evolution time is too short, the difference vector between the two neighboring trajectories may not have sufficient time to evolve with the dynamics on the attractor, and the frequent replacement process can introduce considerable inaccuracies. If the evolution time is too long, the neighboring points can often evolve to distances which are greater than the linearized regime, and so these contributions are also inaccurate. Additionally, for attractors which may have a multilobed structure, such as the Lorenz attractor, enormous errors can be introduced if the evolution time is sufficiently long to allow two neighboring points to eventually evolve along the two separate lobes.

To choose T_E for a time series produced by a map, one or two iterations of the map is usually a good value, as was the case for the Hénon system. For a flow, some experimentation must be done. A good general rule is that the evolution time for a flow should be on the order of $\frac{1}{2}$ to $1\frac{1}{2}$ orbital periods on the attractor, although this again can depend on the magnitude of the Lyapunov exponents. When one only has a time series to work with, an orbital period for the system can be determined by taking a power spectrum of the time series and picking the dominant feature, if any. Once a rough estimate of what the evolution time should be is obtained, it is strongly advised to calculate the Lyapunov value for a range of evolution times around the rough value. The computed values of the Lyapunov exponent versus the evolution time will usually remain flat for some range of the evolution times, and a value within this stable range is usually an accurate choice.

Using the above general guidelines, the Wolf code was used to determine the Lyapunov exponents of two sample systems for which the exponents are already known: the Hénon map and the Lorenz system. In both cases, all of the parameters could be chosen ahead of time with good confidence, with the exception of the evolution time (T_E). For this parameter, a series of runs with differing T_E values were done, as a check of the stability of the Lyapunov value with different evolution times, and to demonstrate how this may be done with other parameters for which good guesses are not available *a priori*.

For the Hénon map (whose dimension is known to be 1.26), we chose $d=2$, and $N=2000$, although about 1000 ($=30^d$) would probably suffice. Since the system is defined by a mapping, we choose $\tau_d=1$ (this is verified using the autocorrelation calculation, Fig. 8), and likewise $t_c=1$. Since the largest scale of the map is about 4, we chose S_{max} to be 0.25 to 0.05. Also, since the data are generated numerically, the only noise is from machine error, so we chose S_{min} to be a conservative 10^{-5}. Note that some experimentation was conducted with these values, but that the result of the calculation showed λ_1 was not greatly affected for parameter values within reasonable limits of the ones given, although the run times could be considerably affected for S_{max} too small. For the remaining parameter T_E we present a graph of the value of the largest exponent λ versus the value of T_E (Fig. 9). Note that there is a plateau in the value of λ at about 0.624, for values of the parameter T_E out to about 5, after which it drops off sharply. Note that even though the characteristic time for his map is 1, we see that λ_1 is stable to a reasonably large variation in T_E. The value we obtain for λ_1 is within about 3% of the value quoted by Wolf *et al.*

For the second example, the Lorenz system, a data set was generated by integrating the dynamical equations with a Runge-Kutta integrator, using a time step [$=1$ (sampling rate)] of about 0.03 sec. Since the characteris-

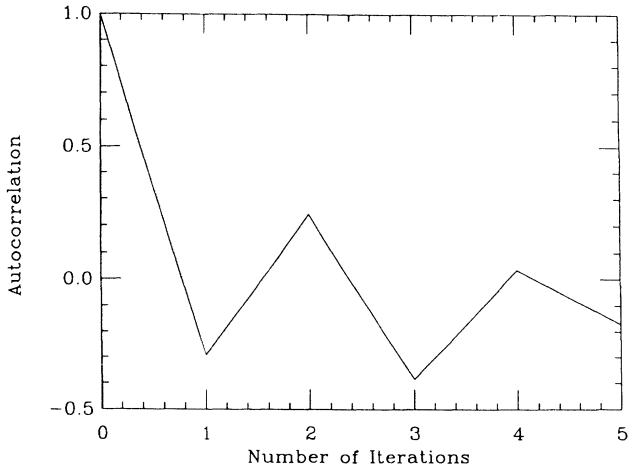

FIG. 8. Autocorrelation function for $x_1(n)$ in the Hénon system from 2000 data points.

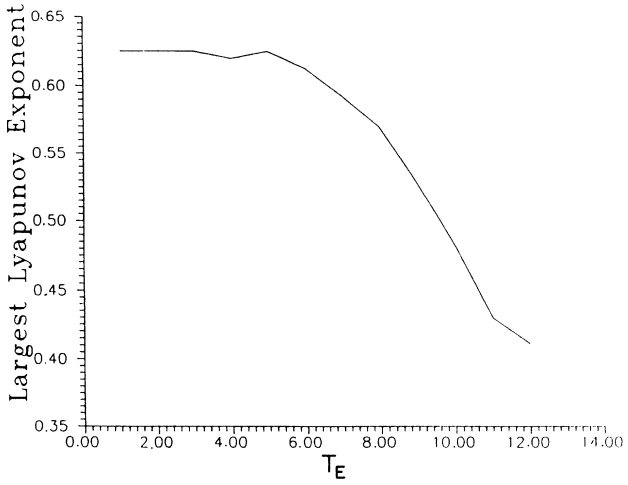

FIG. 9. λ vs T_E for the WSSV method in the Hénon map.

tic time for the Lorenz attractor is about 0.5 sec, this gives about 17 points per orbit on the attractor. The dimension of the attractor is known to be about 2.06, and an embedding dimension of $d=3$ was chosen. The minimum number of data points required, as estimated by our rule of thumb, is about 5000, so we close a set of 10 000 points ($=N$). The autocorrelation calculation (Fig. 6) suggests a value of $t_c=13$, and τ_d is 0.03. Since the maximum scale of the Lorenz attractor is about 40, $S_{\max}$ was chosen to be about 0.4 or 0.5, and $S_{\min}$ was chosen, by the same arguments as for the Hénon system, to be about 10^{-5}. As for the previous example, we calculated the largest Lyapunov exponent for a range of the last parameter T_E and these results are shown in Fig. 10. From the graph, one notes that λ_1 settles into a somewhat flat region by a value of T_E of about 16 or so (one orbital period) and remains roughly so until about 30 (two $\frac{1}{2}$ orbital periods). There is still a considerable variation in the values of λ along this region, which very likely indicates that the convergence is still not very good and a longer data set is necessary. The average value from this

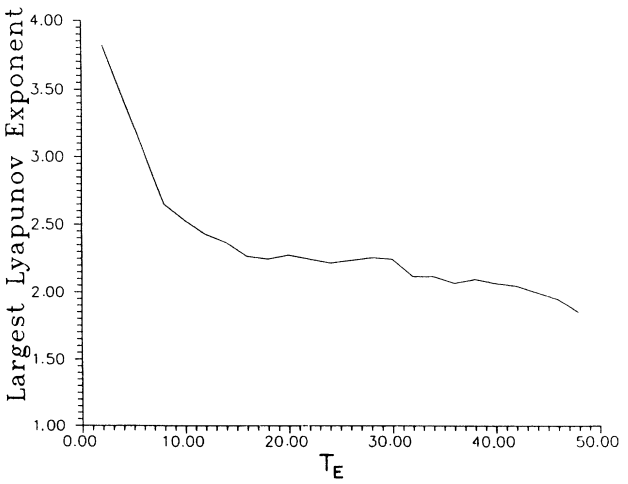

FIG. 10. λ vs T_E for the WSSV method in the Lorenz system.

regime is about 2.22, which is within 2.7% of the value 2.16 quoted by Wolf *et al.*

It is worth noting that, because of the double-lobed structure of the Lorenz attractor, the program can often be "fooled" by choosing two nearby initial orbits which wind up on different lobes of the attractor, thereby giving erroneous contributions to the averaged exponent. In this sense, the Lorenz system is a somewhat difficult case for study using the fixed-time-evolution program, and hence the results can be somewhat less accurate than would be expected.

Calculations of the largest Lyapunov exponent were carried out for other systems beside the two examples above, and in all cases the worst errors were on the order 5%, with most values being about 1–2 % of the expected exponent. We conclude that, at least for the largest positive exponent, the above code is relatively simple to use and provides reliable and reasonably accurate results. Although we have not tested them yet, more elaborate versions of the code promise greater accuracy, as well as the calculation of the rest of the positive Lyapunov exponents. The one drawback of the method is that it does not allow for calculation of the negative exponents of the spectrum, although current research suggests that it may be possible to capture at least the largest negative exponent using time reversal of the data sequence.

Some experimentation was done with calculating the largest Lyapunov exponent for a few other systems, and in all cases the worst errors were on the order of 4–5 %, with most values being within 1–2 % of the expected exponent. We can conclude from these studies that the WSSV code provides a very simple and reasonably accurate way of determining the largest Lyapunov exponent, and does not require the excessive amounts of data that some of the other algorithms seem to need. For applications where only the dominant behavior of the spreading of nearby trajectories is needed, and where it is not necessary to know the remaining Lyapunov exponents, this algorithm can prove very useful.

C. Lyapunov exponents from the map F(y,a)

Whatever method one chooses to use for determining the Lyapunov exponents from the data, and we have examined only two possible methods proposed in the literature, we must now establish a way to express these same quantities in terms of our map $\mathbf{F}(\mathbf{y},\mathbf{a})$. A direct transcription of the methods of Shimada and Nagashima,[15] Benettin, Froeschle, and Scheidecker,[16] or others would lead to a correct prescription, but not one which is easily used in the optimization or fitting we wish to do using the function $\mathbf{F}(\mathbf{y},\mathbf{a})$. The point is that one can achieve better results in this fitting if one has available a useful analytic formula for the derivatives of the constraints with respect to the parameters $\mathbf{a}$. We will choose then a slightly different way of expressing the Lyapunov exponents in terms of the map $\mathbf{F}(\mathbf{y},\mathbf{a})$ than appears in the literature. Ours may be a useful technique in itself.

Lyapunov exponents characterize the way in which neighboring points, small areas, or small volumes near the orbit of interest evolve under the mapping. To find

them one linearizes the mapping $\mathbf{y}(n+1)=\mathbf{F}(\mathbf{y}(n),\mathbf{a})$ around a given orbit $\mathbf{y}(1),\mathbf{y}(2),\ldots,\mathbf{y}(N)$. Small deviations from this orbit, call them $\delta\mathbf{y}(n)$, evolve as

$$\delta\mathbf{y}(n+1)=\mathbf{M}(\mathbf{y}(n))\delta\mathbf{y}(n) ,$$

where

$$[M(\mathbf{y})]_{kl}=\frac{\partial}{\partial y_l}F_k(\mathbf{y},\mathbf{a})$$

is evaluated along the orbit of interest. The Lyapunov exponents are found from the eigenvalues of the matrix $\mathbf{M}^K(\mathbf{y}(1))$

$$\mathbf{M}^K(\mathbf{y}(1))=\mathbf{M}(\mathbf{y}(K))\mathbf{M}(\mathbf{y}(K-1))\cdots\mathbf{M}(\mathbf{y}(1)) ,$$

which has information about the orbit generated by $\mathbf{F}(\mathbf{y},\mathbf{a})$ beginning at $\mathbf{y}(1)$. Indeed, calling the Lyapunov exponents λ_i, $i=1,2,\ldots,d$ the eigenvalues of $\mathbf{M}^K(\mathbf{y}(1))$ are $\exp(\tau K\lambda_i)$ in the limit as $K\rightarrow\infty$. The τ in this expression is the same one we set to unity in Sec. II. For the Hénon system (a map) τ is 1, while for the Lorenz system (an ODE) $\tau=0.03$ (cf. the Eckmann method in this section).

The familiar method of finding these λ_i's (Refs. 10, 15, and 16) is to apply the matrix $\mathbf{M}^K$ to an arbitrary vector $\mathbf{w}$. Then forming

$$\frac{1}{\tau K}\ln\left[\frac{\|\mathbf{M}^K\mathbf{w}\|}{\|\mathbf{w}\|}\right] \tag{9}$$

yields the largest exponent λ_1 for large K. To find the next largest exponent λ_2 one applies $\mathbf{M}^K$ to the elements of an outer product $\mathbf{w}^1\times\mathbf{w}^2$, and calculates the logarithm of the norm of this vector for large K. This gives the sum of λ_1 and λ_2. Continuing in this fashion, the full Lyapunov spectrum may, in principle, be extracted.

While the expression of the λ_i's as logarithms of the norms of various vectors to which $\mathbf{M}^K$ has been applied is correct, it presents serious problems in evaluating the derivatives with respect to the parameters $\mathbf{a}$ of the mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$ from which $\mathbf{M}$ is formed. So we take a slightly different approach.

We note that the trace of the matrix $\mathbf{M}^K$ contains the information on Lyapunov exponents we desire. Our first observation is that the expression

$$\mathrm{tr}(\mathbf{M}^K)=\sum_{m=1}^{d}\exp(\tau K\lambda_m)$$

allows us to find the largest exponent λ_1 by

$$\lambda_1=\frac{1}{\tau K}\ln[\mathrm{tr}(\mathbf{M}^K)] \tag{10}$$

in the formal limit that $K\rightarrow\infty$. This expression is much more conducive to differentiation with respect to the parameters $\mathbf{a}$ (recall that $\mathbf{M}$ is a function of $\mathbf{a}$) since we have to deal with the logarithm of a simple scalar, the trace of $\mathbf{M}^K$, rather than the norm of a vector $\|\mathbf{M}^K\mathbf{w}\|$ as in Eq. (9).

One can find an expression for the next exponent λ_2 by observing that the combination

$$[\mathrm{tr}(\mathbf{M}^K)]^2-\mathrm{tr}(\mathbf{M}^{2K}) ,$$

where

$$\mathbf{M}^{2K}(\mathbf{y}(1))=\mathbf{M}(\mathbf{y}(2K))\times\mathbf{M}(\mathbf{y}(2K-1))\cdots\mathbf{M}(\mathbf{y}(2))\mathbf{M}(\mathbf{y}(1))$$

behaves as $\exp[\tau K(\lambda_1+\lambda_2)]$ for large K. So we can find the sum of $\lambda_1+\lambda_2$ by

$$\lambda_1+\lambda_2=\frac{1}{\tau K}\ln\{[\mathrm{tr}(\mathbf{M}^K)]^2-\mathrm{tr}(\mathbf{M}^{2K})\}$$

for large K. It is straightforward to construct expressions for the sum of exponents up to order m by recognizing the terms in the above logarithms as those of an expansion of the trace of the m^{th} power the matrix $(\mathbf{M}^K)_{ij}-\mathrm{tr}(M^K)\delta_{ij}$.

In any case, our procedure is now clear. Use whatever means available to evaluate the λ_i's from the data. Then form the indicated logarithms of combinations of traces of the matrices $\mathbf{M}^K$, $\mathbf{M}^{2K}$, etc. as computed from the parametrized mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$. Equating the λ_i's evaluated from the data to the expressions for the λ_i's in terms of $\mathbf{F}(\mathbf{y},\mathbf{a})$ gives us a set of d constraints. We impose these constraints on our choice of the parameters $\mathbf{a}$ in conjunction with the minimization of our cost function.

Our actual practice restricts attention to the largest Lyapunov exponent λ_1 since that is the only one we know how to reliably extract from data. Thus only the trace of $\mathbf{M}^K$ is needed in our constraints. It seems to us a matter of some importance to devise accurate methods to determine the full spectrum of Lyapunov exponents from data. They would be useful in the program we are engaged in, and they act as classifiers for nonlinear dynamical systems with broadband power spectra. In the case of broadband spectra, sharp lines are not available for classifying and one must turn to the kind of dynamical invariant we have here.

IV. INVARIANT DISTRIBUTIONS ON THE ATTRACTOR

The frequency with which orbits $\mathbf{y}(n)$ visit regions of the phase space $\mathbb{R}^d$ defines an invariant distribution function, $\rho(\mathbf{y})$, which is formally defined for the mapping $\mathbf{y}(n+1)=\mathbf{F}(\mathbf{y}(n))$ as

$$\rho(\mathbf{y})=\lim_{N\rightarrow\infty}\frac{1}{N}\sum_{k=1}^{N}\delta^d(\mathbf{y}-\mathbf{F}^k(\mathbf{y}(1)))=\lim_{N\rightarrow\infty}\rho_N(\mathbf{y}) . \tag{11}$$

In a similar fashion, the invariant distribution for a numerical data set $\mathbf{y}(n)$, $n=1,\ldots,N$ is given by

$$\rho(\mathbf{y})=\lim_{N\rightarrow\infty}\frac{1}{N}\sum_{k=1}^{N}\delta^d(\mathbf{y}-\mathbf{y}(k)) . \tag{12}$$

Eckmann and Ruelle[10] discuss the features of $\rho(\mathbf{y})$ at some length. In particular, they address the question of the dependence of $\rho(\mathbf{y})$ on the initial point $\mathbf{y}(1)$. They state that any two initial points in the basin of attraction will lead to the same $\rho(\mathbf{y})$. In this sense $\rho(\mathbf{y})$ is invariant. For a dynamical system with two attractors it is possible for their basins of attraction to be intertwined in a complicated way. Any uncertainty in the initial point $\mathbf{y}(1)$

due to noise, machine round off, etc., may effect our ability to say with certainty the attractor to which a particular $\mathbf{y}(1)$ will go. Also, in the absence of noise there may be particular $\mathbf{y}(1)$'s (often, but not exclusively, on a set of measure zero in $\mathbb{R}^d$) that lead to nongeneric orbits. An example of this type of nongeneric behavior would be an unstable fixed point or periodic orbit in the presence of a strange attractor. In any event we will assume that noise levels are low and the only nongeneric orbits are unstable and of measure zero in the phase space. In this case, once a particular $\mathbf{y}(1)$ has moved beyond its transient stage the frequency with which it visits various portions of the attractor is, by definition, $\rho(\mathbf{y})$.

The complete invariant density $\rho(\mathbf{y})$ has too much information in it for our purposes. (We could not constrain a cost function to reproduce every point on the invariant density without an inordinate amount of work.) Any finite sequence of N points has a finite resolution on the attractor. That resolution is approximately N^{-1/d_A}, which is the order of the mean distance of N points on a d_A-dimensional set. Furthermore, we will never actually resolve the detailed δ-function resolution implied by Eqs. (11) and (12).

To handle this matter of finite resolution we introduce a complete orthonormal set of functions $\psi_\mu(\mathbf{y})$ defined on $\mathbb{R}^d$ which can serve as a basis set. We then expand $\rho(\mathbf{y})$ in terms of this basis

$$\rho(\mathbf{y})=\sum_{\mu=1}^{G} B_\mu \psi_\mu(\mathbf{y}) \ . \tag{13}$$

Truncating this expansion at some finite order ($\mu=G$) provides a finite-resolution representation corresponding to whatever information we have on $\rho(\mathbf{y})$. The coefficients B_μ will be the invariants of the dynamical process which characterize $\rho(\mathbf{y})$ within a given basis $\psi_\mu(\mathbf{y})$. After our discussion of how to select the $\psi_\mu(\mathbf{y})$'s we will establish how one extracts B_μ's both from the data vectors $\mathbf{y}(n)$ and from the parametrized map $\mathbf{F}(\mathbf{y},\mathbf{a})$. Equating the B_μ's from the data to those from the map will be our final constraints on the cost function $C(\mathbf{X},\mathbf{a})$.

While any complete orthonormal set of functions $\psi_\mu(\mathbf{y})$ would do to determine our B_μ's, some are more appealing than others. For example, Fourier series formed by taking

$$\psi_m(\mathbf{y})=e^{i\mathbf{m}\cdot\mathbf{y}}, \quad \mathbf{m}=(m_1,m_2,\ldots,m_\alpha)$$

are formally fine. However, since the attractor is occupying only a small portion of $\mathbb{R}^d$, most of the work performed by the Fourier representation of $\rho(\mathbf{y})$ will be expended in making $\rho(\mathbf{y})$ vanish off the attractor. What we seek are orthonormal functions *concentrated on the attractor,* so all the work in the expansion of $\rho(\mathbf{y})$ is expended exhibiting structure where the attractor is located. This would also result in the need for many fewer B_μ than required for Fourier series or other familiar choices for $\psi_\mu(\mathbf{y})$.

An *optimal* choice using information in the data set is constructed as follows.[26,27] Take the total data set $\mathbf{y}(n)$, $n=1,2,\ldots,$ and divide it into two portions. The first portion (of length N) will be treated as the data we are trying to model. The second portion of the data set (of length N') will be used to construct orthonormal functions. These orthonormal functions will be the $\psi_\mu(\mathbf{y})$'s that we will use in our expansion of $\rho(\mathbf{y})$, shown in Eq. (13). To explicitly construct these functions we further divide the second portion of the data into G groups of length L ($N'=LG$). Each group is a sample of the invariant attractor. If L is large enough, each sample is a significant look at $\rho(\mathbf{y})$. Treat each of the G data sets as an independent sample of $\rho(\mathbf{y})$ and form the invariant distribution for the αth sample

$$\rho_\alpha(\mathbf{y})=\frac{1}{L}\sum_{k=1}^{L}\delta^d(y-\mathbf{y}(k,\alpha)) \ , \tag{14}$$

with $\alpha=1,2,\ldots,G$. The data point $\mathbf{y}(k,\alpha)$ is the kth member of the αth sample. Of course, the mean density of the G samples is just the total invariant density of the data set of length N',

$$\rho(\mathbf{y})=\frac{1}{G}\sum_{\alpha=1}^{G}\rho_\alpha(y) \ . \tag{15}$$

From the G samples $\rho_\alpha(\mathbf{y})$ we form the following *phase-space correlation function*

$$R(\mathbf{z},\mathbf{w})=\frac{1}{G}\sum_{\alpha=1}^{G}\rho_\alpha(\mathbf{z})\rho_\alpha(\mathbf{w}) \ . \tag{16}$$

It can be shown[22,26] that the normalized eigenfunctions of this correlations function are the optimal eigenfunctions for expansion of functions localized on the attractor. Optimal means that these eigenfunctions provide the best representation in a least-squares sense of the information in $\rho(\mathbf{y})$ when expressed as a finite series in an eigenbasis. The label α is to be treated as a sampling index from a set of independent looks at the data each of which is to be thought of as selected from a uniform statistical distribution of invariant densities. The various averages over α then appear quite natural.

The requirement that $\psi_\mu(\mathbf{y})$'s be an eigenfunction of $R(\mathbf{z},\mathbf{w})$ leads to

$$\int d^d z \, R(\mathbf{w},\mathbf{z})\psi_\mu(\mathbf{z})=\mu\psi_\mu(\mathbf{w}) \ . \tag{17}$$

The $\psi_\mu(\mathbf{y})$'s are normalized as follows:

$$\int d^d w \, \psi_\mu(\mathbf{w})\psi_{\mu'}(\mathbf{w})=\delta_{\mu\mu'} \ . \tag{18}$$

As the number of samples G becomes infinite, the set of eigenfunctions becomes complete in the usual least-squares sense. If we insert Eq. (16) into Eq. (17), we see that for finite G, $R(\mathbf{w},\mathbf{z})$ becomes a finite sum of separable kernels. It is easily seen that in this case the eigenfunctions $\psi_\mu(\mathbf{y})$ must have the form

$$\psi_\mu(\mathbf{y})=\sum_{\alpha=1}^{G} C_\alpha^\mu \rho_\alpha(\mathbf{y}) \ . \tag{19}$$

The eigenfunctions defined in this fashion are localized near the attractor, just as we wished. This follows directly from Eq. (19) since $\psi_\mu(\mathbf{y})$ is made of the $\rho_\alpha(\mathbf{y})$'s which vanish off the attractor.

Inserting Eqs. (16) and (19) back into Eq. (17) reduces

the eigenvalue equation to a finite matrix problem. The coefficients C_α^μ are the G vectors which are eigenvectors of the $G\times G$ matrix

$$A_{\alpha\beta}=\frac{1}{G}\int d^d z\,\rho_\alpha(\mathbf{z})\rho_\beta(\mathbf{z})\ , \tag{20}$$

i.e.,

$$\sum_{\beta=1}^{G} A_{\alpha\beta}C_\beta^\mu=\mu C_\alpha^\mu\ . \tag{21}$$

We now turn to the normalization condition Eq. (18). Inserting the representation for $\psi_\mu(\mathbf{y})$ given by Eq. (19) into Eq. (18), and using the relationship between the C_α^μ's and $A_{\alpha\beta}$ given by Eqs. (20) and (21), dictates that the vectors C_α^μ obey the following normalization condition:

$$\sum_{\alpha=1}^{G} C_\alpha^\mu C_\alpha^{\mu'}=(\mu G)^{-1}\delta_{\mu\mu'}\ . \tag{22}$$

(Incidentally, this equation also shows that all the eigenvalues μ are positive.)

Formally, the elements of $\rho_\alpha(\mathbf{y})$ are δ functions. Hence, numerically speaking, computation with them is really not possible. We choose to replace $\delta^d(\mathbf{x})$ by

$$\delta^d(\mathbf{x})\to\frac{1}{(\pi\bar\omega)^{d/2}}e^{-|\mathbf{x}|^2\bar\omega}\equiv f_{\bar\omega}(|\mathbf{x}|)\ ,$$

which, when $\bar\omega$ is small, represents only a small loss of resolution in calculating $\rho_\alpha(\mathbf{y})$. $f_{\bar\omega}$ also has the same integral as the δ function it replaces. To this approximation

$$\rho_\alpha(\mathbf{y})=\frac{1}{L}\sum_{k=1}^{L} f_{\bar\omega}(|\mathbf{y}-\mathbf{y}(k,\alpha)|)\ , \tag{23}$$

and Eq. (20) becomes

$$A_{\alpha\beta}=\left[\frac{1}{\pi\bar\omega}\right]^{d/2}\frac{1}{GL^2}\times\sum_{k,j=1}^{L}\exp[-|\mathbf{y}(k,\alpha)-\mathbf{y}(j,\beta)|^2/\bar\omega]\ . \tag{24}$$

We are now in a position to calculate our optimal eigenfunctions $\psi_\mu(\mathbf{y})$ from the G data sets. Use Eq. (24) to numerically calculate the $G\times G$ matrix $A_{\alpha\beta}$. Next calculate the eigenvalues μ and eigenvectors C_α^μ of this matrix, being sure to normalize them according to Eq. (22). We can then form the eigenfunctions $\psi_\mu(\mathbf{y})$ by using the normalized C_α^μ's and the $\rho_\alpha(\mathbf{y})$'s [in the form of Eq. (23)] in Eq. (19).

In Fig. 11 we show $\rho_1(\mathbf{y})$ evaluated for the Hénon attractor from a data set $L=750$ steps in length. These data are displayed on a grid of 75 points in each coordinate direction. The other densities are qualitatively similar in that they are very spikey. However, the exact position and size of the spikes varies from one sample to the next. The $\psi_\mu(\mathbf{y})$'s look like the $\rho_\alpha(\mathbf{y})$'s except that they are allowed to be negative is some regions. This is not surprising since they are composed of the $\rho_\alpha(\mathbf{y})$'s and the weights (given by the C_α^μ's) are not required to be positive [cf. Eq. (19)].

We now have a set of G orthonormal functions $\psi_\mu(\mathbf{y})$ extracted from G samples $\rho_\alpha(\mathbf{y})$ of the invariant distribution. We can use the orthonormality condition, Eq. (18), to project a particular B_μ out of Eq. (13),

$$B_\mu=\int d^d y\,\rho(\mathbf{y})\psi_\mu(\mathbf{y})\ . \tag{25}$$

Incidentally this shows the B_μ are invariants of the mapping since they are integrals of ψ_μ with the density $\rho(\mathbf{y})$ (cf. Sec. I). If we insert Eqs. (12) and (19) into this expression we get

$$\begin{aligned}B_\mu&=\frac{1}{N}\sum_{\alpha=1}^{G}\sum_{k=1}^{N} C_\alpha^\mu\rho_\alpha(\mathbf{y}(k))\\&=\frac{1}{NL}\sum_{k=1}^{N}\left[\sum_{\alpha=1}^{G}\sum_{j=1}^{L}\frac{C_\alpha^\mu}{(\pi\bar\omega)^{d/2}}\times\exp[-|\mathbf{y}(j,\alpha)-\mathbf{y}(k)|^2/\bar\omega]\right]\ ,\end{aligned} \tag{26}$$

where we have used Eq. (23). Equation (26) has been used to numerically calculate the B_μ's from the data.

This should make it clear how one evaluates the B_μ's from the $N'=GL$ data vectors $\mathbf{y}(k,\alpha)$ in $\mathbb{R}^d$. The B_μ's are the G numbers characterizing the invariant density $\rho(\mathbf{y})$ by its projection on the optimum basis vectors $\psi_\mu(\mathbf{y})$. Now we wish to see how to evaluate them from our parametrized mapping $\mathbf{F}(\mathbf{y},\mathbf{a})$. The G equalities between these two evaluations of B_μ form our final constraints on the minimization of the cost function $C(\mathbf{X},\mathbf{a})$.

To determine B_μ from the map $\mathbf{F}(\mathbf{y})$—we suppress the parameters $\mathbf{a}$ for a moment—we return to the definition of the invariant density as expressed by Eqs. (11) and (12). Call A_k the projection on $\psi_\mu(\mathbf{y})$ of each term in the sum in this equation:

$$\begin{aligned}A_k(\mu)&=\int d^d y\,\psi_\mu(\mathbf{y})\delta^d(\mathbf{y}-\mathbf{F}^k(\mathbf{y}(1)))\\&=\psi_\mu(\mathbf{F}^k(\mathbf{y}(1)))\ .\end{aligned} \tag{27}$$

We interpret Eq. (27) as saying that $A_k(\mu)$ is the projection of $\delta^d(\mathbf{y}-\mathbf{F}^k(\mathbf{y}(1)))$ onto the orthonormal eigenfunctions $\psi_\mu(\mathbf{y})$. Using this interpretation we expand the δ function in terms of $\psi_\mu(\mathbf{y})$ to get

$$\delta^d(\mathbf{y}-\mathbf{F}^k(y(1)))=\sum_{\mu=1}^{G} A_k(\mu)\psi_\mu(\mathbf{y})\ .$$

For large N, Eq. (11) can now be written as

$$\rho(\mathbf{y})=\sum_{\mu=1}^{G}\left[\frac{1}{N}\sum_{k=1}^{N} A_k(\mu)\right]\psi_\mu(\mathbf{y})\ .$$

Comparing this equation to Eq. (13) indicates that

$$\begin{aligned}B_\mu&=\frac{1}{N}\sum_{k=1}^{N} A_k(\mu)\\&=\frac{1}{N}\sum_{k=1}^{N}\psi_\mu(\mathbf{F}^k(\mathbf{y}(1)))\ .\end{aligned}$$

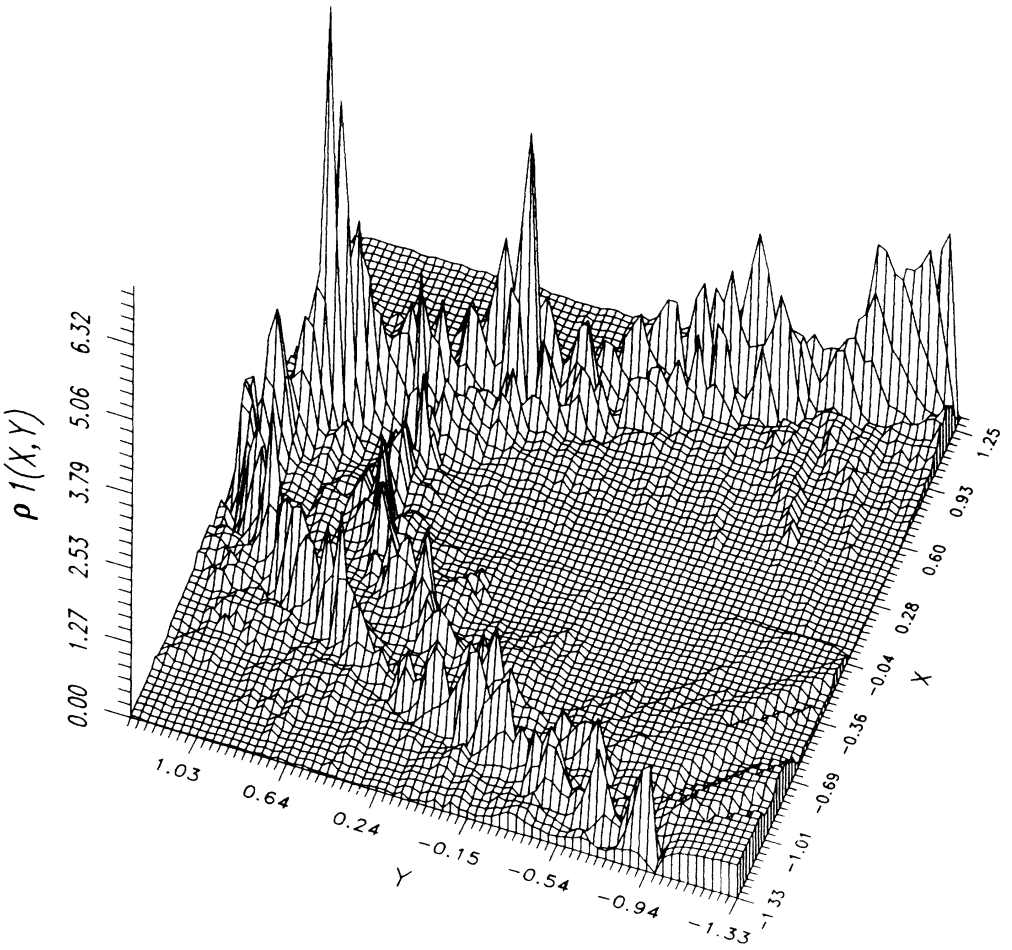

FIG. 11. First density for the Hénon map for $L=750$ and $G=5$.

From Eq. (19) and the definition of $\rho_\alpha(\mathbf{y})$ we rewrite this equation as

$$B_\mu=\frac{1}{NL}\sum_{k=1}^{N}\left[\sum_{j=1}^{L}\sum_{\alpha=1}^{G}C_\alpha^\mu\delta^d(\mathbf{y}(j,\alpha)-\mathbf{F}^k(\mathbf{y}(1)))\right].$$

This expression requires quite high powers of $\mathbf{F}(\mathbf{y},\mathbf{a})$ to be evaluated, and we cannot be confident that such high powers of the parametrized map are numerically accurate. When the map is near the correct or optimum map, then we are accurately reproducing $\mathbf{y}(k+1)$ from $\mathbf{y}(k)$ by a single application of $\mathbf{F}(\mathbf{y},\mathbf{a})$. We utilize this to replace $\mathbf{F}^k(\mathbf{y}(1))$ in the last formula with $\mathbf{F}(\mathbf{y}(k))$. The expression for B_μ's then becomes

$$\begin{aligned}B_\mu&=\frac{1}{NL}\sum_{k=1}^{N}\left[\sum_{\alpha=1}^{G}\sum_{j=1}^{L}C_\alpha^\mu\delta^d(\mathbf{y}(j,\alpha)-\mathbf{F}(\mathbf{y}(k)))\right]\\&=\frac{1}{NL}\sum_{k=1}^{N}\left[\sum_{\alpha=1}^{G}\sum_{j=1}^{L}\frac{C_\alpha^\mu}{(\pi\bar{\omega})^{d/2}}\times\exp(-|\mathbf{y}(j,\alpha)-\mathbf{F}(\mathbf{y}(k))|^2/\bar{\omega})\right],\end{aligned}\tag{28}$$

using our smoothed replacement for the δ functions in the invariant density.

This is the equation we will use to calculate the B_μ's from the map. To implement it we take all N points in the first portion of the data set and iterate them once through the map $\mathbf{F}(\mathbf{y},\mathbf{a})$. When $\mathbf{F}(\mathbf{y},\mathbf{a})$ is near the correct map iterating the data set once will result in points that are still on the attractor. We then evaluate the Gaussian, and numerically sum all the contributions between the iterates of the data set and the points in the G samples.

We close this section with an observation about the B_μ's. By combining the definition Eq. (27) with the identity

$$\delta^d(\mathbf{y}-\mathbf{F}^{k+1}(\mathbf{y}(1)))=\int d^dw\,\delta^d(\mathbf{y}-\mathbf{F}(\mathbf{w}))\delta^d(\mathbf{w}-\mathbf{F}^k(\mathbf{y}(1)))\,,$$

we can derive the recursion relation

$$A_{k+1}(\mu)=\sum_{\mu'}T_{\mu\mu'}A_k(\mu')\,,$$

in which the transition matrix T is given by

$$T_{\mu\mu'}=\int d^d y\ \psi_\mu(\mathbf{y})\psi_{\mu'}(\mathbf{F}(\mathbf{y}))\ . \tag{29}$$

In a matrix notation the recursion relation $A_{k+1}=TA_k$ leads to an expression for

$$\int d^d y\ \rho_L(\mathbf{y})\psi_\mu(\mathbf{y})=\rho_L(\mu)=\frac{1}{L}\sum_{k=1}^{L} A_k(\mu)\ ,$$

which is

$$(1-T)\rho_L(\mu)=\frac{1}{L}(1-T^L)A_1\ .$$

Since $\lim_{L\to\infty}\rho_L(\mu)=\int d^d y\ \rho(\mathbf{y})\psi_\mu(\mathbf{y})=B_\mu$, this shows that the B_μ are the components of the eigenvector of T with eigenvalue unity and further that all other eigenvalues must lie within the unit circle, if the expression, Eq. (11), for $\rho(\mathbf{y})$ converges. By our assumption that the $\rho(\mathbf{y})$ we observe is unique, we infer that the eigenvalue unity of T is nondegenerate. We tried to implement this observation about the B_μ to yield a method for numerically determining them from Eq. (29) (Refs. 18 and 19), but found roundoff error undermined our efforts.

V. OPTIMIZATION OF THE CONSTRAINED COST FUNCTION: PARAMETER DETERMINATION

A. Analysis for the Hénon map

Our first application of the methods described above is to data generated by the Hénon map of the plane to itself. Data were created by iterating the map from some initial condition and discarding the first 50 points of that data set. Two data sets of $x_1(n)$ were created this way. The first had $N'=3750$ points which we divided into five groups of 750 points each. These groups were used to create the densities $\rho_\alpha(\mathbf{y})$, and the phase-space correlation function among groups was used to generate the eigenfunctions. The second data set was then used to select samples of length $N=750$, 1200, and 1752 for our analysis.

We first studied the distribution of Euclidian distances among the two vectors $\mathbf{y}(n)=(x_1(n),x_1(n+1))$, $n=1,2,\ldots,N-1$ formed from the data set. On the natural scale of the attractor, which is order unity, the minimum distance was always order 10^{-5}–10^{-4}. This led us to choose the parameter σ in our maps to be $\sigma=5\times10^{-6}$ so that each data point, at least for $N\geq500$, would have neighbors. We varied σ by a factor of 10 or so with no qualitative differences in our results. A thorough parameter search would vary σ in the constrained minimization of the cost function.

Next we chose to use four parameters $\mathbf{a}$ in our set and took the powers m_3 and m_4 in $\mathbf{F}(\mathbf{y},\mathbf{a})$ to be $m_3=4$, $m_4=5$. We did not further vary these parameters. Our choice of four $\mathbf{a}$'s rested on our knowledge that we would be constraining our cost functions by only the largest Lyapunov exponent λ_1 and the projection B_1 of $\rho(y)$ on the eigenfunction $\psi_1(\mathbf{y})$ with the largest eigenvalue. Four seemed a minimum reasonable number of parameters, and since the work required to search large parameter sets can become significant, we were content with four. In effect, we had two free parameters in $\mathbf{F}(\mathbf{y},\mathbf{a})$ when the values of λ_1 and B_1 were specified.

Our final *a priori* choice was on the values of X_j in the predictor Eq. (3). We took three terms here since we were being quite conservative in how many iterations of the map $\mathbf{F}(\mathbf{y},\mathbf{a})$ we felt we could trust. Then, further reflecting our sense that iterations of $\mathbf{F}(\mathbf{y},\mathbf{a})$ could become unreliable, we chose $X_1=0.8$, $X_2=0.1$, and $X_3=0.1$. Once again the $\mathbf{X}$'s could be parameters which vary in our constrained minimization. We found that varying the $\mathbf{X}$'s by 20% or so did not qualitatively change our results. In the case of the Lorenz attractor study discussed in Sec. V B we report results for $X_1=0.5$, $X_2=0.3$, and $X_3=0.2$, and note that the cost function changes by $\approx20\%$.

We chose to simply fix the $\mathbf{X}$'s for purposes of this paper. Clearly, the $\mathbf{X}$'s can be varied along with the $\mathbf{a}$'s, σ, and $m_3,\ldots,m_P$, if one wishes. Ours is a first try with the $\mathbf{F}(\mathbf{y},\mathbf{a})$ we have chosen in fitting the data and meeting the invariant constraints. The feasibility of accomplishing this seemed daunting enough when we set out. We expect to include many more parameters in future work in this area.

One additional important matter deserves note before we proceed to the discussion of our numerical results. The maps $\mathbf{F}(\mathbf{y},\mathbf{a})$ as we carry out our search over the parameters $\mathbf{a}$ have very little ability to reliably fit the given data for most $\mathbf{a}$. Only when we arrive near a (constrained) minimum of the cost function can we be very confident that our map is reasonable. Until the map is near the optimum map points in the data set are quite often mapped far off the attractor. For numerical stability in our search algorithms we need a method to identify orbits which are leaving the attractor for nonoptimal values of $\mathbf{a}$ and return them to the neighborhood of the attractor.

Maps of the form we have chosen have the feature that points far off the attractor, as defined by the data set itself, are mapped to $\mathbf{y}=0$. There is no reason to expect the origin of coordinates to lie on an attractor which has $d_A<d$ and is quite sparse in $\mathbb{R}^d$, but we choose to always translate our data set so one of its points is the origin. This changes nothing about the signal processing issues we address in this paper and makes our parameter searches numerically sensible.

With this translation of the origin, an orbit being generated by $\mathbf{F}(\mathbf{y},\mathbf{a})$, when $\mathbf{a}$ is not optimal, which tries to depart significantly from the attractor is sent back to $\mathbf{y}=0$, which is now on the attractor. When $\mathbf{a}$ is near its optimal values, this feature is operationally unimportant because the map is tracking the data very accurately.

Our experience indicates that if one is trying to create global maps $\mathbf{F}(\mathbf{y},\mathbf{a})$, as we are here, some form of "orbit reinjection" will be required to give numerical sense to the whole process of searching parameter space to minimize the cost function. The problem becomes more important as d grows, since the attractor of dimension $d_a<d$ occupies "less and less" of the full volume of the phase space. If one is making "fits" to the data by numerous local or nearly local polynomial maps as in the work of FS, the issue raised here is absent. Global maps

have an economy of parameters and a potential ease of interpretation; local maps appear to have an advantage of calculational speed. We have no overall judgment of a way to choose between these alternatives.

Our results for data from the Hénon map are shown in Table III. The parameter searches were carried out using the FORTRAN package NPSOL.[41] One of its authors, Gill, was kind enough to consult with us extensively on its use and on the interpretation of its output. For each value of the number of points in the data set, namely, $N=750$, 1200, and 1752, we report seven quantities for each of three cases: (1) unconstrained minimization of the cost function; (2) minimization constrained by λ_1 ($\lambda_1^{\text{map}}=0.408$); and (3) minimization constrained by both λ_1 and B_1. In each case we report the value of the cost function normalized by the sum of the squares of the Euclidian lengths over all data vectors, the values of the **a**'s at the minimum cost function, and the deviations $\Delta\lambda_1$ and ΔB_1 from the values of λ_1^{data} and B_1^{data} determined by the data. The allowed tolerances on these deviations are set in NPSOL by the user. We typically required the relative magnitudes of $\Delta\lambda_1$ to λ_1 and the same for B_1 to be in the range 0.5–5 %. This is not a limitation of NPSOL, but it seemed quite accurate enough for our purposes.

A look at Table III reveals a consistent pattern. Unconstrained optimization resulted in a cost function with a rms deviation of our predictor from the data of 0.1% or smaller. Not surprisingly when we track the data so accurately, the value of B_1 comes out quite precise. The value of λ_1^{map} for this best least-squares fit is remarkably bad. Indeed, in our examples this quantity was actually negative, which indicates the absence of chaos for the parametric map.

When the λ_1 constraint is imposed, the parameters **a** change, but we regard their specific values as of incidental interest here. More important is the observation that the rms value of the cost function—the bare measure of the quality of the fit—remains about 0.5% while the Lyapunov exponent is now accurate to about 1% or better. Of course, having moved away from the very best point-to-point least-squares tracking of the data, the accuracy of B_1 degrades to $\approx 10\%$. Finally, imposing both constraints we achieve 0.5% or so in the rms error for the cost function, highly accurate λ_1, and somewhat better B_1 values.

The message of these calculations is that the procedure we outlined in this paper is both feasible and highly accurate. The few scalar numbers, the cost function, λ_1, and B_1 do not tell the whole story. One can take the map with the optimum **a**'s and calculate a new orbit starting from some new phase space point $\mathbf{y}^{\text{new}}(1)$: $y^{\text{new}}(1)$, $y^{\text{new}}(2)$, . . . , and compare the new orbit to that generated by the Hènon map starting with the same initial point. The data so generated look the same when plotted as a sequence of two vectors, but this temporal representation contains very little useful information, so we do not show it.

What is more important is the fact that our predictor

$$\mathbf{y}(m+1)=\sum_{k=1}^{L} X_k \mathbf{F}^k(\mathbf{y}(m-k+1),\mathbf{a}) \tag{30}$$

accurately predicts. We have taken numerous points

TABLE III. Optimization results for Hénon map data. $C(\mathbf{X},\mathbf{a})$ is shown with and without invariant constraints.

$$\mathbf{F}(\mathbf{y},\mathbf{a})=\sum_{j=1}^{N-1}\mathbf{y}(j+1)g(\mathbf{y},\mathbf{y}(j);\mathbf{a}))$$

$$C(\mathbf{X},\mathbf{a})=\frac{\sum_{k=L}^{N-1}|\mathbf{y}(k+1)-\sum_{j=1}^{L}x_j\mathbf{F}^j(\mathbf{y}(k-j+1),\mathbf{a})|^2}{\sum_{n=1}^{N}\mathbf{y}(n)\cdot\mathbf{y}(n)}$$

$X_1=0.8 \quad X_2=0.1 \quad X_3=0.1$

Number of points=750 $\lambda_1^{\text{data}}=0.408$ $B_1^{\text{data}}=3.4739$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$	ΔB_1^{map}
Unconstrained	4.016×10^{-7}	7.5347	1.3289	−0.7041	0.1485	−2.0098	-5.70×10^{-3}
λ_1	4.77×10^{-6}	6.6855	20.6948	−0.1714	0.0956	1.6×10^{-4}	0.223
λ_1, B_1	2.06×10^{-5}	0.3422	1.1169	0.3766	−0.055 86	4.23×10^{-2}	0.140

Number of points=1200 $\lambda_1^{\text{data}}=0.408$ $B_1^{\text{data}}=3.388$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$	ΔB_1^{map}
Unconstrained	3.41×10^{-6}	7.5217	2.9658	−0.3145	0.075 02	−1.676	-9.85×10^{-3}
λ_1	1.1297×10^{-5}	8.4520	26.7454	0.2686	0.011 77	-6.32×10^{-4}	0.214
λ_1, B_1	2.38×10^{-5}	6.6093	19.6341	0.083 62	−0.010 87	-1.1×10^{-4}	0.198

Number of points=1752 $\lambda_1^{\text{data}}=0.408$ $B_1^{\text{data}}=3.369$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$	ΔB_1^{map}
Unconstrained	3.5359×10^{-7}	8.4093	6.0546	−0.1497	0.023 15	−1.211	-5.80×10^{-3}
λ_1	1.7284×10^{-5}	3.1671	9.8576	−0.1120	0.027 43	-6.512×10^{-3}	0.2605
λ_1, B_1	2.54×10^{-5}	5.8314	18.446 12	0.1832	−0.028 18	1.005×10^{-4}	0.2466

from our data set and evolved them forward by use of the predictor. We find we are able to track the actual data to the 1% level, seven to ten steps along the orbit all around the attractor. This means that iterates of our optimum map $\mathbf{F}^k(\mathbf{y},\mathbf{a})$ are accurate to $k \approx 7-10$, far beyond our original safe choice of $k=3$. The implications of this remarkable accuracy for prediction and control of nonlinear chaotic systems are transparent.

B. Prediction for the Lorenz system

We now turn to the application of our methods to the Lorenz system, defined by Eq. (8). These equations were originally motivated by an attempt to model atmospheric phenomenon using only a few degrees-of-freedom dynamical system. It was one of the first systems known to exhibit an attractor of fractal dimension, or a strange attractor, and consequently to connect this with the apparent chaotic motion of the resultant dynamics. The primary concern in modifying our previous techniques for use on the Lorenz system will be (i) the jump to a three-dimensional embedding space, which will require much longer time series to properly fill out the attractor, and (ii) the large difference in the macroscale of the two attractors, which will require the rescaling of some of the variables we have previously defined. We will first, however, give a short review of some of the characteristics of the Lorenz system.

For the parameter values $\sigma=16.0$, $r=45.92$, and $b=4.0$, the Lorenz system possesses a strange attractor which has become one of the classic examples of nonlinear science. The structure consists of two nearly flat lobes connected, roughly at a point and angled somewhat with respect to one another. Hence the local dimension of the attractor is essentially two, however, the minimum embedding space required is three. Note that the motion of the phase-space orbits for the Lorenz systems is continuous, i.e., a flow, as opposed to that of the Hénon system which is a mapping. The discretization of the Lorenz orbits after phase-space reconstruction, and the density of points along an orbit, is therefore due to the choice of a sampling rate in the measurement of the time series of data. This sampling rate therefore can be thought of as setting a time scale in the reconstructed picture of the attractor. In turn, this time scale determines the time-delay values for the method of phase-space reconstruction used in Sec. II, the evolution times for Lyapunov exponent calculations, etc. A discussion of optimal ranges of sampling rates, and the problems which occur when sampling rates are too large or small, is given at some length by Mayer-Kress.[40] In practical applications, of course, one often has no control over the data set one is presented with, although too frequent sampling can often be remedied by simply throwing away data.

To investigate the behavior of our prediction technique on a system with a somewhat larger embedding space, we chose the Lorenz system as a test case with known parameters, as was done with the Hénon system. An "experimental" time series was generated for the Lorenz equations, Eq. (8), using the parameter values listed above, by a Runge-Kutta numerical integration scheme with a fixed time step of 0.03. A data set of the x_1 variable consisting of approximately 20 000 points (after transients) was generated. We used the same time series for all of our numerical runs.

As we have stated above (cf. Sec. II) we chose an embedding dimension of three. Note that in an actual experimental situation, a more cautious choice of four would also be reasonable, although this would have increased our computational requirements by a significant amount. For the choice of the delay time constant τ a number of different choices could be made. Since the delay time reconstruction is rather weakly dependent on this constant, provided one is within certain limits, there is no unique choice for this variable. Our final choice was motivated by the desire to have the reconstructed attractor look most like the original Lorenz attractor. This results in a time delay of two time steps. For an actual case where one would have no *a priori* sense of what the attractor looks like, the methods of Sec. II are, of course, recommended. A feel for the required density of points can also be obtained by calculating the minimum nearest-neighbor distance, and perhaps the frequencies that a range of somewhat larger neighbor distances occur, and comparing this with the "macroscale" of the attractor (i.e., the maximum ranges of the coordinates of an attractor).

The embedding dimension and delay time comprise the two parameters necessary to correctly reconstruct the dynamics of the systems attractor, and hence is the first step in setting up the prediction method. We now turn to the changes necessary in the numerical algorithm when we consider the Lorenz system.

The most significant difference between the prediction models for the Hénon system and for the Lorenz system is that of the size of the time series required for the phase-space reconstruction. Because of the increase in the dimensionality of the embedding space from two to three, the number of phase-space points required to perform our procedure increases dramatically. The reasons for this is clear. Our prediction function $\mathbf{F}(\mathbf{y},\mathbf{a})$ requires that most points have a significant number of nearby neighbors, i.e., points within distances of a few $\sqrt{\sigma}$ values so that a good "mapping" of the local phase space around a particular region is obtained. Additionally, nearby neighbors are important to obtain good numerical approximations to the gradients of the objective and constraint functions. Since the number of points required to yield a given mean nearest-neighbor distance is considerably larger for a volume than for an area, the number of points required to properly fill out the attractor is much greater for a three-dimensional embedding space. In Sec. II we presented general methods for determining the number of data vectors needed for a given embedding dimension d. For our particular analysis of the Lorenz attractor reported here, we found that the minimum number of points that gave reasonable results to be about 6000. For the numerical experiments reported in Table IV we used data sets with 6000 and with 8000 points.

One final change in the numerical parameters for the prediction code is in the number of matrices that are to be multiplied together to obtain the Lyapunov exponent

TABLE IV. Optimization results for Lorenz attractor data. $C(\mathbf{X},\mathbf{a})$ is shown with and without invariant constraints.

$$\mathbf{F}(\mathbf{y},\mathbf{a})=\sum_{j=1}^{N-1}\mathbf{y}(j+1)g(\mathbf{y},\mathbf{y}(j);\mathbf{a}))$$

$$C(\mathbf{X},\mathbf{a})=\frac{\sum_{k=L}^{N-1}|\mathbf{y}(k+1)-\sum_{j=1}^{L}x_j\mathbf{F}^j(\mathbf{y}(k-j+1),\mathbf{a})|^2}{\sum_{n=1}^{N}\mathbf{y}(n)\cdot\mathbf{y}(n)}$$

$X_1=0.8 \quad X_2=0.1 \quad X_3=0.1$
Number of points $=6000 \quad \lambda_1^{\text{data}}=1.51$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$
Unconstrained	$2.516\,72\times10^{-5}$	57.7977	0.087 68	10.2388	−0.043 58	-4.100×10^{-2}
λ_1	$2.516\,72\times10^{-5}$	57.7977	0.090 44	10.2388	−0.043 35	-1.000×10^{-2}

Number of points $=8000 \quad \lambda_1^{\text{data}}=1.51$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$
Unconstrained	$1.870\,51\times10^{-5}$	57.7977	0.085 89	10.2392	−0.036 77	-6.162×10^{-2}
λ_1	$1.870\,51\times10^{-5}$	57.7976	0.093 52	10.2392	−0.037 62	-1.101×10^{-2}

$X_1=0.5 \quad X_2=0.3 \quad X_3=0.2$
Number of points $=6000 \quad \lambda_1^{\text{data}}=1.51$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$
Unconstrained	$3.223\,71\times10^{-5}$	37.1460	0.0201	0.4224	0.00	−1.0706
λ_1	$3.223\,72\times10^{-5}$	37.1459	0.0581	0.4224	0.020 16	-1.000×10^{-2}

Number of points $=8000 \quad \lambda_1^{\text{data}}=1.51$

	$C(\mathbf{X},\mathbf{a})$	a_1	a_2	a_3	a_4	$\Delta\lambda_1^{\text{map}}$
Unconstrained	$2.395\,96\times10^{-5}$	81.1459	0.055 89	2.4222	0.00	−0.8307
λ_1	$2.395\,97\times10^{-5}$	81.1456	−0.1270	2.4226	9.0809×10^{-4}	-9.999×10^{-3}

from the mapping function. Since each iteration of the Hénon map represents a significant evolution of the system, the multiplication of 500 Jacobian matrices for the Lyapunov calculation represents a good average over the phase space, and results in fairly good accuracy of the final value. However, each step of the time series for the Lorenz system represents much less evolution time for the dynamics. It was necessary to experiment with the number of matrices required to give good convergence. It was found that about 1000 matrix products gave a reasonably good convergence to the final value, but was still not excessively computationally intensive.

To complete the formulation of the prediction model for the Lorenz data, it is necessary to pick the exact form of the mapping and cost functions that are to be minimized. We first discuss the choice of the polynomial terms which multiply the exponential in the mapping function. These terms are defined, as for the Hénon analysis, with the intention of giving the exponential form in the mapping function a longer "tail" by adding multiplicative polynomial terms to it. As for the Hénon analysis, we chose to use four polynomial terms in the mapping function, and hence have four variables in the minimization fit. The first coefficient is, of course, the constant term, and the second again multiplies the linear term that expresses some dependence of the mapping function on the Lyapunov exponent. Therefore there remains to be determined the powers of the last two polynomial terms.

In choosing the values of the exponents of the remaining two polynomial terms, we recall that we wish to elongate the tail of the exponential term in the mapping function to make it feel more of the surrounding neighbors. However, we do not wish to make these exponents so large that we increase the scale well beyond that which we set by σ. After some experimentation, we chose $m=3$ and 6 as the two powers for the polynomial terms, although this is by no means the only possible choice.

The second set of parameters of the minimization procedure which need to be chosen are the $\mathbf{X}$'s which appear in the definition of the cost function Eq. (4). These coefficients weight the different iterates of the map $\mathbf{F}(\mathbf{y},\mathbf{a})$ and essentially determine how many iterates forward we wish the map to accurately reproduce the data. For the Hénon analysis, we chose three $\mathbf{X}$'s with values (0.8,0.1,0.1). Our choice indicates a desire to weight the first forward iterate very heavily, while giving the second and third iterates only minimal importance. This set of values was chosen primarily because the Hénon system is a mapping, and each iterate represents a large step in evolution of the original system. On the other hand, the Lorenz system produces a flow in phase space, and the time step we chose for each iterate of the time series represents a rather small amount of forward evolution of the system. Thus we choose to weight some of the multiple iterates of the map more heavily than we did for the Hénon system. We have therefore presented data for the Lorenz system with two different sets of values for these

parameters. In one case we used the original weights of the Hénon system (0.8,0.1,0.1). For the other case, we weighted the multiple iterates more heavily, namely, (0.5,0.3,0.2). Note that we could have easily chosen to take more than two multiple iterates of the system. However, for the sake of simplicity and comparison we chose to use two as for the Hénon system. We also point note that the **X**'s like the **a**'s could be made variables in the minimization search; we will do that in our further work in this matter.

Finally, to determine a value of the parameter σ (which sets a characteristic scale of distance over which the mapping function is influenced by neighbors), it is necessary to experiment with different values by actually doing a number of minimization runs. One can, however, make an *a priori* guess by considering two factors. The largest value that σ can possibly have will certainly be the scale of the linear regime for the system. This is very roughly about 1% of the attractors macroscale, as mentioned previously. Hence σ should be considerably smaller than this value. Additionally, the smallest value that σ can possibly attain is given by the smallest neighbor distance of the data set, and should be at least one to two orders of magnitude larger than this value. Within this range, σ must be chosen with some experimentation. We have found that typically, the value of the $C(\mathbf{X},\mathbf{a})$ at its minima will be relatively large for larger values of σ, and decreases until a threshold in σ is crossed. For values of σ smaller than the threshold value, the minima of $C(\mathbf{X},\mathbf{a})$ becomes a great deal less, sometimes by an order of magnitude or more. We recommend that σ be chosen somewhat smaller than this threshold value, however, not too much smaller as it is still desirable to have as much of the surrounding phase space as possible contribute to the mapping of each orbital point. For our experiments on the Lorenz system we used $\sigma = 1.0 \times 10^{-4}$.

Using the parameter values stated above, a search for the minima of Eq. (4) in the parameter space **a** was conducted using the NPSOL (Ref. 41) package. Since there is no general method known for determining the absolute minimum of a function using numerical methods, one generally proceeds by finding the minima after iteration for each of a large number of initial conditions, while attempting to cover a large representation of the phase space. In practice, one will usually find a number of local minima, all of which have "basins of attraction" of varying sizes. After a number of runs, one usually will gain some intuition as to which regions of the parameter space evolve to which local minima. When some confidence is gained that a large region of the parameter space has been investigated, we label the minimum with the lowest cost function value the "absolute" minimum. Of course, generally speaking, one can never be sure that one has bound the actual global minimum.

Using the time series for the Lorenz data and the parameter values we have just described, the NPSOL routine was able to find a number of minima of the cost function $C(\mathbf{X},\mathbf{a})$. There values ranged over as much as two orders of magnitude. The lowest value of the cost function found was in the neighborhood of 1.87×10^{-5}, as indicated in Table IV. In the preliminary analysis there were three minima which had almost this same value. A more detailed analysis, however, found that after many iterations of the search routine two of these minima actually evolved into the third. Using better error tolerances in NPSOL, it was found that this point actually did have a slightly lower minima. It should be noted that even though the three minima had cost functions which agreed very closely, their resulting values for the **a**'s were much different. This is in keeping with our observation that, for a large range of parameter values around these minima, the cost function was very "flat" with respect to the parameters, i.e., $C(X,\mathbf{a})$ varied very little over a large range of **a**'s. This has the unfortunate effect of causing the iteration procedure to proceed very slowly, since the minima were very shallow, and a large number of iterations were required to achieve the optimal solution. One possible conclusion from this is that, if one were interested in a purely least-squares fit of the map to the data, any of the parameter sets in this range were nearly as good as the optimal solution.

After the analysis just described, we performed another changing the **X**'s changed to (0.5,0.3,0.2). These parameter values weight the later iterates of the map more heavily, and correspond to trying to make the map predict farther into the future. We did not impose the B_μ constraints on the Lorenz system, but used this system to explore the variations on the cost function and the quality of our ability to reproduce the largest Lyapunov exponent as we changed the weights X_j in the predictor. The results of these minimization searches are also presented in Table IV; both 6000 and 8000 points on the attractor are used in our example. As can be seen, the cost function for these minima are about $\frac{1}{3}$ higher than for the previous system, and this is to be expected since the later iterates, which must be inherently less accurate, now give a much larger contribution to the cost function. In terms of relative fitting error, however, these minima are still surprisingly low. The final parameter values, although significantly different from the previous system, are still similar enough to give the same general character to the fitting function.

One noticeable difference between the two different values of **X**'s was in the fitting of the map using the Lyapunov constraint. The iteration procedure for the (0.5,0.3,0.2) system went far more quickly than for the (0.8,0.1,0.1) system. This can probably be interpreted in terms of the fact that if later iterates of the map are weighted more heavily, then the parameters result in more sensitivity of the map to the Lyapunov constraint, which usually requires longer evolution times to manifest itself for flows.

VI. SUMMARY AND FUTURE TASKS

In this paper we have given a set of procedures which one may use to process signals $x(n)$, $n = 1,2,\ldots$, having a broadband power spectrum. Using numerically generated data from the Hénon map and from the Lorenz equations we have also demonstrated explicitly the feasibility of our procedures. Processing a signal means that from the time series $x(n)$ we do the following.

Find an integer-dimensional embedding space of time lagged d vectors

$$\mathbf{y}(n)=(x(n),x(n+\tau_1),\ldots,x(n+\tau_{d-1}))\ ,$$

which fully expose the geometric structure of the attractor on which the data evolves. The attractor has dimension d_A which may be fractional. Choosing the integer $d>2d_A+1$ is guaranteed to be sufficient for this purpose, but smaller d may often work.

Find invariants of the evolution $\mathbf{y}(1),\mathbf{y}(2),\ldots,\mathbf{y}(N)$ in $\mathbb{R}^d$—specifically, the Lyapunov exponent spectrum $\lambda_1,\lambda_2,\ldots,\lambda_d$ and selected optimum moments $B_1,B_2,\ldots,B_G$, of the invariant density $\rho(\mathbf{y})$, on the attractor.

Use these vectors $\mathbf{y}(n)$ and invariants to construct a parametrized map of $\mathbb{R}^d$ to itself $\mathbf{y}\rightarrow\mathbf{F}(\mathbf{y},\mathbf{a})$, which minimizes a certain constrained least-squares cost function based on the residual errors of a nonlinear predictor

$$\mathbf{y}(m+1)=\sum_{k=1}^{L}X_k\mathbf{F}^k(\mathbf{y}(m-k+1),\mathbf{a})\ ,$$

involving iterates $\mathbf{F}^k$ of the map.

The output of the signal processing is the map $\mathbf{F}(\mathbf{y},\mathbf{a})$—both its form and the parameters $\mathbf{a}$—and the coefficients X_j in the predictor.

A map $\mathbf{F}(\mathbf{y},\mathbf{a})$ and a predictor which give very small least-squares residuals when evaluated on the data we call *reliable*. We have explicitly demonstrated in this paper that even a reliable $\mathbf{F}(\mathbf{y},\mathbf{a})$ does not necessarily reproduce invariants such as the λ_a and the B_μ discussed by us. The reason is that a least-squares tracking of a data set $\mathbf{y}(n)$ by a map $\mathbf{y}(n+1)\approx\mathbf{F}(\mathbf{y}(n),\mathbf{a})$ does not necessarily provide a good evaluation of the local tangent space mapping $M_{ij}=\partial F_i(\mathbf{y})/\partial y_j$. A map which is reliable and also gives the correct invariants we call *representational*. Our maps are representational because we constrain the least-squares minimization by the invariants. A map which closely tracks data but does not yield the dynamical invariants misses the essential ingredients which classify or identify the dynamical system underlying the data.

Another way to state our constrained optimization procedure is that the cost function to use in determining the map should not be composed only of the square of the residuals in the predictor. It should also contain terms which measure the residuals in matching the invariants determined by the data and the same quantity determined by the maps. NPSOL and other contemporary optimization routines do essentially this by a combination of Lagrange multiplier and quadratic penalty terms added to the least-squares cost function. This point of view suggests that we should not focus on the size of $C(\mathbf{X},\mathbf{a})$ as our goodness of fit criterion but on $C(\mathbf{X},\mathbf{a})+\sum_a(\Delta\lambda_a)^2+\sum_\mu(\Delta B_\mu)^2$. In our Tables III and IV we have reported the values of each of these quantities separately, but the sum as noted should measure the merit of our maps.

In practice, carrying out our signal processing program raises a number of issues of importance in dynamical systems as well as in the present context. The first of these is the determination of the dimension d of the embedding space in which the phase-space reconstruction $x(n)\rightarrow\mathbf{y}(n)$ takes place. We have used the correlation function Eq. (6), but the choice of a dimension at which this stops changing is quite subjective. Establishing an objective criterion would be most useful. Perhaps one of the information theoretic criteria developed in statistics for identifying the number of degrees of freedom in a data set would provide a tool here.[42] An objective criterion for establishing the time delays τ_a would also be desirable.

Methods for determining the Lyapunov spectrum $\lambda_1,\ldots,\lambda_d$ from the data are also quite important. These are classifiers of the dynamical system and a representational map must reproduce them. This is not at all a new issue as should be clear from the discussions in Sec. III. Our own work in this area, which will be reported in detail in a subsequent paper, uses *local* maps of the form of our $\mathbf{F}(\mathbf{y},\mathbf{a})$ and fits the parameters $\mathbf{a}$ and σ to the tangent map at every time step. The local tangent map $M(\mathbf{a}(n))_{ij}$ takes groups of phase-space points in the neighborhood of the orbit point $\mathbf{y}(n)$ into groups around $\mathbf{y}(n+1)$. The dependence of $\mathbf{M}$ on $\mathbf{y}$ is sensitive to the variation of $\mathbf{M}$ over the neighborhood of phase-space points. When one has short data sets and thus sparse neighborhoods, this dependence on $\mathbf{y}$ gives a better approximation to $\mathbf{M}(\mathbf{y})$ than a local constant matrix.[43] The eigenvalues of the product of the local $\mathbf{M}$'s along the orbit yield the λ_a.

As should be clear from our discussion of the structure of the parametrized map $\mathbf{F}(\mathbf{y},\mathbf{a})$, if we remain with our general form (which we do not insist on), then properties of $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ are what we must address. Our choice in this paper has been to use scalar products of $\mathbf{y}$ and $\mathbf{y}(n)$ in forming g. These are insensitive to directional information on the attractor. The structure of neighborhoods of phase-space points near the orbit $\mathbf{y}(n)$ is not isotropic, so much of the information in our data may be used in our present choice of g. Since we want g to provide direction sensitive weights, we might wish to build in some of the local phase-space structure on the attractor. Some of this information is contained in the correlation function among points in the neighborhood of the orbit. If an orbit point $\mathbf{y}(n)$ has N_B neighbors $\mathbf{y}^\beta(n)$ within $\sqrt{\sigma}$, the correlation function is

$$W_{ij}(n)=\frac{1}{N_B}\sum_{\beta=1}^{N_B}[y^\beta(n)-y(n)]_i[y^\beta(n)-y(n)]_i\ .$$

Following a suggestion of Fukunaga[22] we would use the local correlation matrix in our $g(\mathbf{y},\mathbf{y}(n);\mathbf{a})$ by making the replacements

$$|\mathbf{y}-\mathbf{y}(n)|^2\rightarrow\sum_{i,j=1}^{d}[y-y(n)]_iW_{ij}^{-1}(n)[y-y(n)]_j$$

and

$$\mathbf{y}(n)\cdot(\mathbf{y}-\mathbf{y}(n))\rightarrow\sum_{i,j=1}^{d}y(n)_iW_{ij}^{-1}(n)[y-y(n)]_j\ .$$

This now emphasizes directions in phase space along the attractor where the correlation is larger.

In addition to these improvements in our ability to per-

form each element of our signal processing program, the application of methods established here to laboratory and field data would be quite productive. The applications would be both to classification by dynamical invariants of observed broadband signals and to prediction on those signals. Further having a clear idea now of the geometric setting in which the signal processing takes place in time domain, we can begin exploration of these methods to control of nonlinear systems.[44]

Finally, there is the matter of noise, extrinsic noise, which contaminates our broadband signal $x(n)$. Many conventional methods for identifying signals in noise rely on the distinct spectral characteristics of the two. That tool is absent for us, and we must use alternative tactics. We do not have a contribution to this important issue which we have tested out in any quantitative way. A natural framework will be the distinct dynamical characteristics of noise and chaotic motion embodied in differing d_A (finite for a chaotic attractor and filling any dimension for noise), invariant density $\rho(\mathbf{y})$ (structured for chaotic time series and homogeneous for noise), and other similar attributes. We will report on our tested ideas in this matter in future articles.

ACKNOWLEDGMENTS

We are most appreciative for productive conversations with K. Bruckner, M. Freedman, H. Levine, J. Theiler, and B. West about the material covered in this paper. This work was supported in part by U.S. Defense Advanced Research Projects Agency (DARPA), Applied and Computational Mathematics Program, under Contract No. F 49620-87-C-0117 and in part under the DARPA–University Research Initiative (URI), Contract No. N00014-86-K-0758. Some of the computation reported in this paper was carried out at the NASA Ames Research Center's Numerical Aerodynamic Simulation Program under the auspices of the Joint Program in Nonlinear Science between the University of California and NASA-Ames. Additionally, J.B.K. wishes to acknowledge support of the U.S. Air Force Office of Scientific Research Grant No. AFOSR-89-0072.

[1]T. Kailath, *Linear Systems* (Prentice-Hall, Englewood Cliffs, NJ, 1980).

[2]A. Papoulis, *Signal Analysis* (McGraw-Hill, New York, 1977).

[3]Francis C. Moon, *Chaotic Vibrations* (Wiley, New York, 1987).

[4]J. M. T. Thompson, and H. B. Stewart, *Nonlinear Dynamics and Chaos* (Wiley, New York, 1986).

[5]N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, Phys. Rev. Lett. **45**, 712 (1980).

[6]R. Mañé, in *Dynamical Systems and Turbulence,* Vol. 898 of *Lecture Notes in Mathematics,* edited by D. Rand and L. S. Young (Springer, Berlin, 1981), pp. 230–242.

[7]F. Takens, in *Dynamical Systems and Turbulence,* Ref. 6, pp. 366–381.

[8]J.-C. Roux, R. H. Simoyi, and H. L. Swinney, Physica D **8**, 257 (1984).

[9]D. S. Broomhead and G. P. King, Physica D **20**, 217 (1986).

[10]J.-P. Eckmann and D. Ruelle, Rev. Mod. Phys. **57**, 617 (1985).

[11]J. D. Farmer and John J. Sidorowich, Phys. Rev. Lett. **59**, 845 (1987); Center for Nonlinear Studies, Los Alamos National Laboratory Report No. LA-UR-88-901, 1988.

[12]A. S. Lapedes, and R. Farber, Los Alamos National Laboratory Report No. LA-UR87-2662, 1987, and Report No. LA-UR-88-418, 1988.

[13]J. P. Crutchfield and B. S. McNamara, Complex Sys. **1**, 417 (1987).

[14]J. M. Greene and J. S. Kim, Physica D **24**, 213 (1987).

[15]I. Shimada and T. Nagashima, Prog. Theoret. Phys. **61**, 1605 (1979).

[16]G. Benettin, C. Froeschle, and J. P. Scheidecker, Phys. Rev. A **19**, 2454 (1979).

[17]V. I. Oseledec, Tr. Mosk. Mat. Obsc. Moscow Math. Soc. **19**, 17 (1968).

[18]S. Grossmann, and S. Thomae, Z. Naturforsch. **32A**, 1353 (1977).

[19]H. D. I. Abarbanel, and P. E. Latham, Phys. Lett. **89A**, 55 (1982).

[20]J. Rice, in *Computer Science and Statistics: The Interface,* edited by James E. Gentle (North-Holland, Amsterdam, 1983).

[21]B. W. Silverman, *Density Estimation for Statistics and Data Analysis* (Chapman and Hall, London, 1986).

[22]K. Fukunaga, *Introduction to Statistical Pattern Recognition* (Academic, New York, 1972), Chap. 6.

[23]J. D. Farmer and John J. Sidorowich (private communication).

[24]We are grateful to P. H. Diamond, M. H. Freedman, and H. Levine for numerous discussions about the issue of choosing the "cost function."

[25]R. A. Hummel, Comput. Graphics Image Processing **9**, 40 (1979).

[26]A. Rosenfeld and A. C. Kak, *Digital Picture Processing,* 2nd ed. (Academic, New York, 1981).

[27]W. L. Root, Proc. IEEE **75**, 1446 (1987). Root points out that the method, usually called by the names of Karhunen and Loéve, was used as early as 1950 by U. Grenander, Ark. Mat. **1**, 195 (1950); and by D. C. Youla, IRE Trans. Inf. Theory **IT-4**, 171 (1954). Clearly it has a long history.

[28]P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization* (Academic, New York, 1981); R. Fletcher, *Practical Methods of Optimization,* 2nd ed. (Wiley, Chichester, 1987).

[29]Wayne A. Fuller, *Measurement Error Models* (Wiley, New York, 1987). We thank Eric Kostelich for pointing out this work to us. Also, see E. J. Kostelich and J. A. Yorke, Phys. Rev. A **38**, 1649 (1988).

[30]John J. Sidorowich (private communication).

[31]A. M. Fraser and H. L. Swinney, Phys. Rev. **33A**, 1134 (1986); Ph.D. dissertation, University of Texas at Austin, 1988. Also see the article in Physica D **34**, 391 (1989).

[32]F. Takens, in *Proceedings of the Dynamical Systems and Bifurcations Conference, Groningen, 1984* (Springer, Berlin, 1984).

[33]P. Grassberger and I. Procaccia, Phys. Rev. Lett. **50**, 346 (1983); Physica D **9**, 189 (1983).

[34]J. Theiler, Ph.D. dissertation, California Institute of Technology, 1987 (unpublished). We have used Theiler's code for the calculation of $C(r)$ in our examples. We are most appreciative to him for lending us this code and assisting us in its use.

[35]It is usual to extract from the correlation function $D(r)$ the power ν as an estimate of the dimension of the attractor. As

Theiler (Ref. 34) has persuasively argued, this is a problematic exercise since the ν found by $\nu=\lim_{r\to 0}\log D(r)/\log r$ may not even exist. See also the recent report by Theiler, MIT Lincoln Laboratory, 1989 (unpublished). We are thankful to Dr. Theiler for sending us a copy of this paper prior to publication.

[36]M. Hénon, Commun. Math. Phys. **50**, 69 (1976).

[37]E. N. Lorenz, J. Atmos. Sci. **20**, 130 (1963).

[38]J.-P. Eckmann, S. O. Kamphorst, D. Ruelle, and S. Ciliberto, Phys. Rev. **34A**, 4971 (1986).

[39]A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, Physica **16D**, 285 (1985).

[40]Gottfried Mayer-Kress, Los Alamos National Laboratory, Report No. LA-UR-87-1030, 1987 (unpublished).

[41]Philip E. Gill, Walter Murray, Michael A. Saunders, and Margaret H. Wright, *User's Guide for NPSOL (Version 4.0): A FORTRAN Package for Nonlinear Programming,* Systems Optimization Laboratory, Stanford University Technical Report No. SOL 86-2, 1986.

[42]Timothy R. C. Read and Noel A. C. Crease, *Goodness-of-Fit Statistics for Multivariate Data* (Springer-Verlag, New York, 1988), Sec. 8.3.

[43]M. Sano and Y. Sawada, Phys. Rev. Lett. **55**, 1082 (1985).

[44]B. A. Huberman and E. Lumer (unpublished).