University of Potsdam
Institute of Physics
Nonlinear Dynamics Group

Jan Philipp Dietrich

Phase Space Reconstruction using the frequency domain

a generalization of actual methods

Diploma Thesis
July 2008

Mentors:
Prof. Dr. Frank Spahn – University of Potsdam
Prof. Dr. Bernd Blasius – University of Oldenburg (ICBM)

Contents

1. Introduction

1.1. What is it about?

Analysis of nonlinear systems is often connected to significant problems. Many analysing methods do not work because they are based on benefits provided by the superposition principle (which only holds for linear systems). Without superposition it is not possible to chop the whole system into smaller parts. Therefore one complex problem has to be solved instead of solving a few easier ones.

Keeping this part in mind it is easy to understand why many analysing methods fail for nonlinear systems. But how can a nonlinear system be analysed?

One possible solution will be discussed in detail on the following pages: Phase Space Reconstruction (PSR) or also called Attractor Reconstruction.

PSR is a method to reconstruct the whole, multidimensional dynamics of a system out of only one component. So only one dimension is needed to be able to reconstruct the dynamics of a system with many dimensions. This affords fantastic applications for different problems.

It can be used for the analysis of real data, where only one or a few components of a dynamical system can be measured. One example is the investigation of measle epidemics as done by B. Blasius et al. [3]. It is easy to measure the number of infected persons, but the number of sucebtibles can not be provided. This component which is important for the dynamics can be obtained by PSR.

Another application is the classification of attractors. To catch the behaviour of a system it is necessary to distinguish between periodic and chaotic cases. The analysis of the power spectrum can give some hints (as used e.g. by Fenstermacher et al. for analysing a Taylor vortex flow [6]) but for more accurate results again the whole dynamics are needed. After reconstruction the classification can be done by calculation of the fractal dimension ([29], [5]) and identification of the Lyapunov exponents ([35], [14]). An example for this application can be found in "Observation of a strange attractor" by Roux et al. [24].

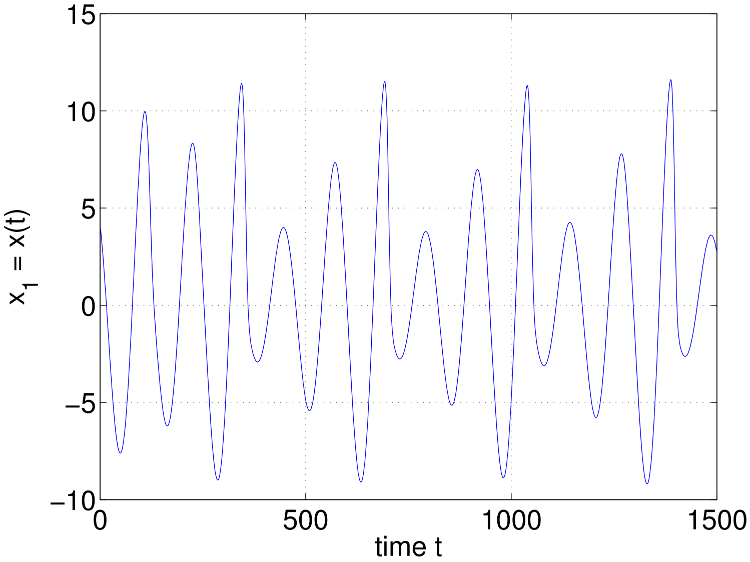


Figure 1.1.: X component of a Rössler-attractor

PSR is also useful to get a general expression of the behaviour of a system. Whereas a time series is often hard to interpret the whole dynamics are often a good hint for what is happening exactly. An elementary example is the Rössler system designed by O. E. Rössler [21].

Looking only at the x-component of this system (fig. 1.1) some periodicity is noticeable but the varying amplitude is hard to explain. Taking a look at the reconstructed attractor (fig. 1.2) helps to comprehend this phenomen much better. The further knowledge enhances the ability of making predictions about the system.

There are already good methods for doing PSR, but a general structure which unites all of them is still missing. So the question for this work is: Is there a general structure of all functions/methods that can be used for Phase Space Reconstruction?

The idea is to compare all these methods in the frequency domain because it seems to be more natural for this process: Taking a look back to the roots of PSR which were done for the detection of strange attractors one will find that the first approaches for getting information about the kind of attractor was to take a look at the frequency distribution in the power spectrum (e.g. done in [9], [8], [6] or

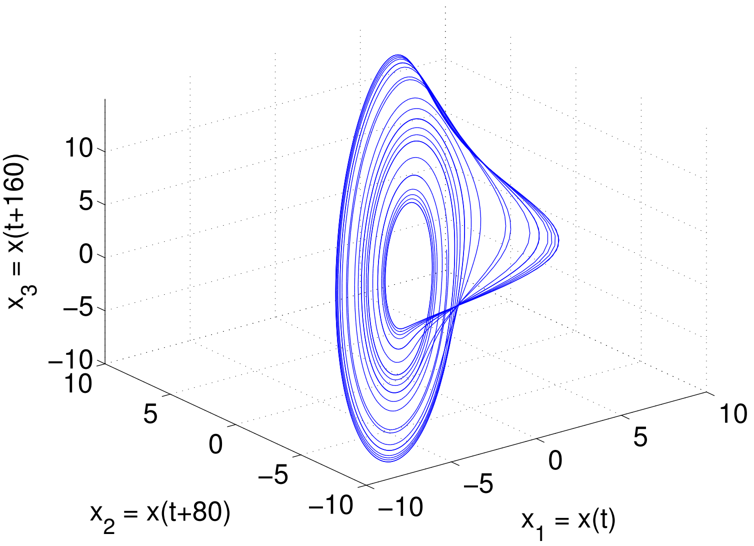


Figure 1.2.: Phase Space Reconstruction of a Rössler-attractor

[27]). At this point the power spectrum could deliver already some information about the attractor. But Takens doubted in his article "Detecting strange attractors in turbulence" [30] that the power spectrum contains enough information for reconstruction. Instead he invented the time-delay method which uses the time series directly.

His assumption of missing information in the power spectrum was probably right. But still the use of frequencies for reconstruction looks natural using the association that a frequency is some kind of rotation around some center. With this interpretation a single frequency becomes a two-dimensional cycle and a set of frequencies something more-dimensional. Hence it gets more clearly why one can reconstruct the whole dynamics using only one dimension as input: To get trajectories without gaps and bends it is necessary that the dynamics contained in every dimension are the same. That means that the frequencies of each dimension need to be represented also in all other dimensions. Hence the ansatz in this diploma thesis is to use the complex frequency domain which is achieved by using the fourier transformation for reconstruction. It seems that this can provide a more intuive imaginitation how the reconstruction process works compared using the time-series itself. And in opposite to the powerspectrum one has no loss of information.

Presenting this work is done in the following way: The remaining part of chapter 1 provides some basic information which is necessary to understand the following chapters. It includes basic methods and definitions (1.2), dynamical systems (1.3), and common methods for reconstruction (1.4). Chapter 2 takes a look at the frequency domain. Classical reconstruction methods are transformed into it (2.2) and a more general description of reconstruction methods is developed (2.3). Accordingly these new results are justified in chapter 3 and applied to some choosen examples in chapter 4. Chapter 5 provides a selection of further results concerning this topic as e.g. some thoughts about the universality (5.1) and plausibility (5.2) of this reconstruction method. But also other useful additional results obtained during this investigation are presented. At the end chapter 6 gives an overview about all results and a preview for the steps that should be done next.

Appendix A contains further data concerning the numerical verification done in chapter 3. Appendix B offers the sourcecode of the matlabfunction used for the reconstructions in the frequency domain.

1.2. Basic methods and definitions

1.2.1. Dynamical System

A dynamical system is a system which is described by a given time evolution for each point. That means after choosing a starting position in the ambient space the dynamical system will describe the evolution in time of this state. For a dynamical system this expansion is described by a deterministic formula, so that the time evolution is well defined for each state without being influenced by stochastic factors. That means knowing the exact starting position in the phase space and knowing the dynamical equations allow to predict the system state for any time exactly. There are two main types of dynamical systems: Systems continuous in time which are described by differential equations (eq. 1.1) and systems discrete in time which are described by maps (eq. 1.2).

$$\dot{\vec{x}}(t) = \vec{g}(\vec{x}) \tag{1.1}$$

$$\vec{x}_{n+1} = \vec{g}(\vec{x}_n) \tag{1.2}$$

Both cases can be handled by PSR. In the case of PSR the equations of the dynamical system are not known normally (otherwise there is no need for reconstruction). Instead PSR is working with so called "time series": The measured values of a variable of the system at different times under fixed external conditions (eq. 1.3).

$$\text{time series } = (x(t_1), x(t_2), x(t_3), ...) \tag{1.3}$$

Examples for such a time series are the temperature measured every hour at one location, or the counted number of sun spots measured each day. Mathematically this time series can be described as a projection of the evolution of a point in phase space onto a lower dimensional subspace.

1.2.2. Phase Space

To handle a dynamical system the phase space is an useful construction. It is the space spanned by all variables which are defining the state of an object in the system. So one can say it is the space in which the dynamical system is living. And it is, as the name "phase space reconstruction" already suggests, the space that is reconstructed by PSR. The time evolution of a state is represented by trajectories in phase space which are constructed by drawing each state of the time evolution into the phase space. These trajectories are giving information about the behaviour of the system and are showing in which direction a point in the system evolutes. Trajectories never intersect each other in phase space because the evolution at every point is well defined (otherwise the evolution of a point at an intersection would not be clear). This is an important characteristic to distinguish between a correct and a wrong reconstruction of phase space.

1.2.3. Topological Equivalence

To understand what an embedding is one has to understand what topological equivalence means: Two objects are topological equivalent if there is a deformation which transforms one object to the other without destroying the surface of it. That means that the object can be squeezed, distorted, pulled and even cut if it is at the end sticked together at the same place. But it is not allowed to cut connections without resticking it or to agglutinate parts where no connection was before.

As an example a ball is topological equivalent to a desk but not to a doughnut. Whereas a doughnut is topological equivalent to a cup. The reason is that cup and doughnut contain exactly one hole whereas ball and desk have no hole in their structure.

Another important point to keep in mind for the following considerations is the option to cut an object and to reagglutinate it without changing its topology. Taking only simple objects into account this opportunity might not be relevant but dealing with more complex structures this is important to remind. It explains e.g. why point reflected objects are topological equivalent to their original object. Without cutting and reagglutination it is mostly not possible to transform an object to its point reflected version. This remark will become important later on.

1.2.4. Embedding and diffeomorphism

Having the description of topological equivalence it is easy to understand what an embedding is: An embedding is a representation of a system in phase space which is topological equivalent to its original representation. That means that all topological properties of the original system are also contained in the embedded system. This relation can also be described more mathematically: An embedding of an object A is a diffeomorphic map of A into another phase space. That means that the map is bijective (one-to-one and onto), differentiable and its inverted map is also differentiable. Hence every point in an embedding can be identified explicit with a point of the original system and the embedding is smooth and has no corners or gaps.

1.2.5. Frequency domain and power spectrum

Instead of saving values for each time t there is also another useful way of storage: The frequency domain. In this case the data is presented for each frequency ω instead for each time t. This presentation makes it easy to identify important frequencies in the data or to filter explicit kinds of noise.

To transform data from time to frequency domain the Fourier Transformation can be used. One has primary to distinguish between four cases: The data is either discrete or continuous and either finite or infinite. For the following work especially two cases are relevant: The case of continuous data of infinite length presents the optimal situation and therefore represents the basis on which the following considerations will rely theoretically. In this case the Fourier transformation is done by integration:

$$\tilde{x}(\omega) = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{\infty} x(t) e^{-i\omega t} \mathrm{dt} \tag{1.4}$$

Practically one has to deal with discrete and finite data. This is the situation for which all considerations will be applied to. In the discrete situation the integral is substituted by a limited sum and the available frequencies become discrete ($\omega_k = 2\pi k/N$ for $k = 0, ..., N-1$):

$$\tilde{x}(\omega_k) = \frac{1}{N} \sum_{t=0}^{N-1} x(t) e^{-i\omega_k t} \tag{1.5}$$

Hence adverse the idealized continuous and infinite situation one has to deal with some sources of error: Due to the discretization information is lost if the sampling rate f_s is too low. As shown in the Nyquist-Shannon sampling theorem the sampling rate must be higher then two times the highest frequency contained in the spectrum f_{max}: $f_s > 2f_{max}$ ($\omega = 2\pi f$). Another problem is that some transformations as e.g. derivation are only defined for continuous data. Coevally the finite length of time series generates the problem that transformation rules, that are valid for the infinite case become more complex and therefore impracticable. A more detailed discussion about this errors can be found in chapter 2 when Fourier Transformation is applied directly for reconstruction.

Besides these differences it has to be noted that the results of both equations are complex: The result contains information about the amplitude of each frequency and also about its phase.

For analysing only the frequencies of a time-series the power spectrum $P(\omega)$ is used. The powerspectrum of a process is defined as the squared modulus of the continuos Fourier transform $\tilde{x}(\omega)$ [15].

$$P(\omega) = |\tilde{x}(\omega)|^2 \tag{1.6}$$

One gets a real valued representation of all amplitudes. Before invention of Phase Space Reconstruction this result was used to detect chaotic behaviour in a time series.

In the case of discrete time-series we have to estimate the power-spectrum. One often used estimate is the so called periodogram which is just the squared modulus of the discrete Fourier transform. But this approximation has to be handled carefully. There are better methods for estimation the power spectrum (e.g. the modified periodogram method by P. Welch [32]). Further discussions about this problematic can be found in "Nonlinear Time Series Analysis" by Holger Kantz and Thomas Schreiber [15].

1.2.6. Attractor

An attractor describes the long-time behaviour of a system. Starting somewhere within the phase space a trajectory will reach the attractor after some time. Hence one can say that an attractor is the stable part of a system. This is meant in the sence that this part will be reached by trajectories at any time while other parts will be emptied. For an undisturbed system only the attractor will remain after some time and normally only the attractor itself or trajectories similar to the attractor will be observed. This gives the opportunity to understand a dynamical system only by taking a look at the attractor. There are several different definitions of an attractor. In general one can say that it is an area A within the phase space to which all (or some - depending on the used definition) parts of the dynamical system will flow to. A typical definition of an attractor A for a discrete dynamic g(x) as done by R.F. Williams (1968) [34] is:

A is an attractor of the dynamic g(x) if,

1. $g(A) \subset A$. Points within A cannot escape anymore.

2. A has a neighborhood U so that $f(U) \subset U$ and $\bigcap\limits_{i>0} g^i(U) = A$. Points within U are attracted to A.
3. There is no $A' \subset A$ for which 1. and 2. is fulfilled. A is the smallest set for which 1. and 2. holds, every point in A is important.

In other words: Once reached an attractor it cannot be left anymore (1), it attracts points in its environment (2) and every point of the attractor is needed to fulfil the first two requrirements - the whole attractor is really used (3).

A good overview concerning all these different definitions (including the given one) and a less restrictive definition can be found in "On the Concept of Attractor" by John Milnor (1985) [18].

1.2.7. Lyapunov Exponents

As shown the attractor contains important information about the dynamics of a system. Hence it is worthwhile to study some characteristics of it. One question is: What will happen with points started closely together? Will they come closer to each other, will the distance remain nearly constant or will they spread?

To answer it one can take a starting point x_0 with some small disturbance δ_0 and observe the exponential growth-rate λ of the disturbance:

$$x_0 + \delta_0 \rightarrow x_n + \delta_n \tag{1.7}$$

In the case of an one-dimensional discrete system with map $g(x)$ one assumes the following exponential evolution of the disturbance δ:

$$\delta_n = \delta_0 e^{\lambda n} \quad \Rightarrow \quad \lambda = \frac{1}{n} \ln \left(\frac{\delta_n}{\delta_0} \right) \tag{1.8}$$

With $\delta_n = g^n(x_0 + \delta_0) - g^n(x_0)$ and $g^n(x_0) =$ n-times application of g on x_0 one gets

$$\lambda = \frac{1}{n} \ln \left(\frac{g^n(x_0 + \delta_0) - g^n(x_0)}{\delta_0} \right) \quad \Rightarrow \quad \lambda = \frac{1}{n} \ln \left(\frac{d(g^n)(x_0)}{dx} \right) \tag{1.9}$$

The Lyapunov Exponent is now defined as the upper limit for $n \to \infty$ of this expression and can be calculated for each dimension of a system separately.

$$\lambda(x_0) = \lim_{n\to\infty} \sup \frac{1}{n} \ln \left(\frac{d(g^n)(x_0)}{dx} \right) \tag{1.10}$$

The classically expected result for dissipative systems are negative Lyapunov exponents. The reason is that dissipation reduces the volume in phase space due to the energy loss. This indicates that trajectories should come closer together and that differences in the starting-position should be erased. Furthermore solely this characteristic makes it possible to get the same results of an experiment for each execution. Otherwise a small change in starting conditions (which happens generally repeating an experiment) would cause completely different results. Hence many error reduction methods are based on this reproducibility.

Nevertheless every attractor of a continuous dynamic which is bounded and not a fixed point contains at least one vanishing Lyapunov exponent in direction of the trajectory [11]. A negative Lyapunov exponent along the trajectory means that points on the trajectory are pressed together. Hence the speed of the trajectory would be reduced constantly. Such a behaviour has to end up in a fixed point. For a positive exponent in this direction the trajectory would speed up constantly, but within a bounded region this is not possible. The reason is that the dynamic $\dot{\vec{x}}(t) = \vec{g}(\vec{x})$ does not permit a raising speed upto infinity for finite values of $\vec{x}$ because of the continuity of $\vec{g}(\vec{x})$.

At least there is also the possibility of positive Lyapunov exponents. In this cases one observes an interesting phenomen: Chaos. Having a positive Lyapunov exponent means that a small disturbance will raise exponentially. A cluster of points starting closely together will be spreaded fast over the whole attractor. Hence the starting position become more and more unimportant while the influence of disturbances raises. Long time predictions become impossible.

Keeping these different behaviours in mind one can now start to categorize differnt attractor types. The kind of attractor can be described by its Lyapunov exponents. A limit cycle in three dimensional space has e.g. (0,-,-) exponents (one vanishing exponent and two negative ones), that means that in two directions everything flows together while the distance along the trajectory remains constant. A torus has because of its structure two vanishing and one negative exponent (0,0,-) and so on.

1.2.8. Chaos and "Strange Attractors"

One reason why the study of attractors became so popular is the existence of so called "Strange attractors": Attractors containing chaotic dynamics [1]. In our everyday life "chaos" means that something is in a way unsorted, mixed and without any structure. It describes things which are hard to handle because of this features. Surprisingly in math the word "chaos" is used to describe deterministic systems: Systems which are completely systematic because they have rules which desribe exactly how the system evolutes in time. Hence it is the precise opposite of the everyday life definition where every kind of rule is missing. The reason for the use of the word "chaos" therefore is not its definition but its behaviour: these completely deterministic systems become unpredictable! This fact is very surprising because it looks strange that there should be a system for which one knows exactly its time evolution but is not able to make any predictions.

If one knows exactly how to calculate the evolution of a state why should it be impossible to make predictions? It seems as one would know everything about the system what one needs to know, but in fact this assumption is wrong: The problem can be understood well by using Lyapunov Exponents: In a strange attractor one has at least one positive Lyapunov Exponent. This means that points starting closely together will spread in time. But these systems also contain dissipation. Hence there is another trend pushing the trajectories back together. This process can be described as streching and folding. It is the typical characteristic of chaotic systems: The trajectories are streched first, so that points closely together flow apart and they are folded so that they do not leave the compact attractor. This leads to a giant dependency on disturbances which outshines the starting position completely. So the problem of prediction in chaotic systems is not the process of time-evolution - this is well defined. The problem is the limited knowledge about the starting conditions. Hence a prediction is only limited because the actual state cannot be determined perfectly.

[1] In literature one finds different definitions of "strange attractors". I use the definition that an attractor is strange if the underlying dynamic is chaotic. Other definitions requesting a fractal dimension for a strange attractor. Using this alternative definition one can find strange attractors which are not chaotic and chaotic attractors which are not strange [10]. However this discussion is not essential for the following considerations. Hence, to simplify the situation I will just use the phrase "strange attractors" for attractors of chaotic systems. Nevertheless it is important to keep this differences in mind.

Reading through literature there are many different definitions of chaos. I will use the relatively broad definition offered by S.H. Strogatz [29]:

> Chaos is aperiodic long-term behavior in a deterministic system that exhibits sensitive dependence on initial conditions.

Related to the description of chaos above only the formulation "aperiodic long-term behavior" is new: It is only a more mathematical description of its unpredictable behavior.

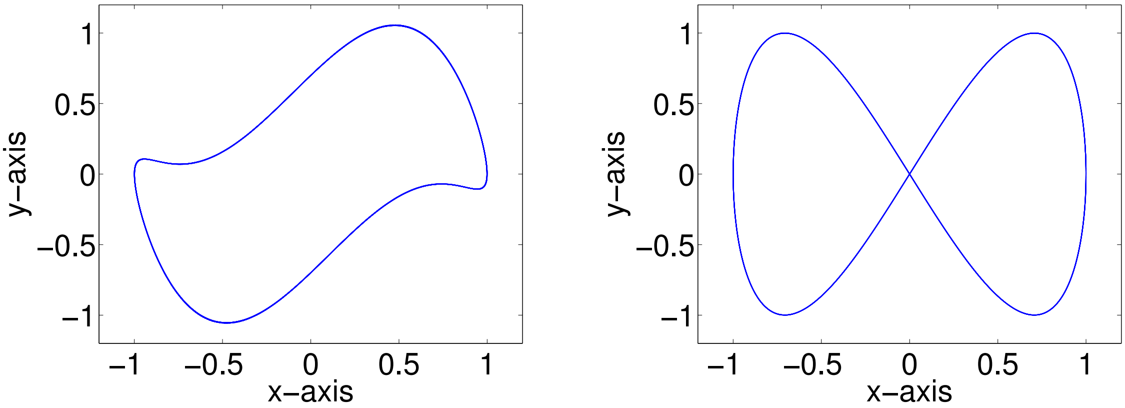
Another interesting point about chaos is that for continuous systems it can only exist with more than two dimensions. The reason can be understand easily looking at the Lyapunov Exponents: A continuous and chaotic system needs at least one positive exponent for stretching, at least one negative component for folding and one vanishing component along the trajectory. Hence a continuous chaotic system must have at least three dimensions (+,0,-). In discrete systems there is not necessarily a vanishing Lyapunov exponent. Therefore this rule cannot be applied on discrete systems.

1.2.9. Whitneys embedding theorem

The previous parts showed the use of attractors and explained that knowing the attractor means knowing the dynamical system. Hence the next step is to take a closer look to the subject of this work itself: Attractor Reconstruction. Before one can reconstruct a system some thoughts about the dimension one needs are necessary. How many dimensions do one need that the attractor fits in it? Let us assume a randomly chosen map of an attractor in another phase space. Will this map be an embedding? An interesting answer is delivered by Whitneys embedding theorem [33]. Using a for attractor reconstruction optimized formulation done by Sauer et. al [26] it says:

> Let A be a compact smooth manifold of dimension d contained in $\mathbb{R}^k$. Then Almost every smooth map $\mathbb{R}^k \to \mathbb{R}^{2d+1}$ is an embedding of A.

In other words: Given the collection of all maps of A into $\mathbb{R}^{2d+1}$ the probability to choose an embedding is 1. By choosing an embedding dimension bigger than $2d$ one can be relatively sure to get an embedding.



(a) good embedding (b) failed embedding

Figure 1.3.: 2-dimensional representations of a Limit Cycle

To understand this statement better let us assume a limit cycle attractor (a closed circuit). Its dimension is $d = 1$ so one should use a 3-dimensional space for reconstruction. But one knows that it is possible to present a limit cycle also in 2 dimensions (fig. 1.3(a)). Why should one use 3 dimensions? The problem in 2 dimensions are situations for which the reconstruction fails. Fig. 1.3(b) is no proper embedding because of its intersection. Looking at the situation of an embedding in 3 dimensions as required by Whitneys embedding theorem (fig. 1.4) one can see that an intersection is also possible. The crucial difference between both cases is its behaviour for small changes within the mapping: Imagine a small change in the mapping of fig. 1.3(b). The position of the intersection probably would change a bit but the intersection would still remain. Imagine the situation in 3 dimensions nearly every small change in the map would remove an intersection. The reason is the higher degree of freedom. And that is exactly the point: Whitneys theorem delivers the dimension in which the probabilty for an intersection falls down to zero.

Following approach can be also helpful for comprehension: Given two d_1- and d_2-dimensional objects N and M placed in a d_{space}-dimensional space. Then the expected dimension d_{over} of the overlap of both objects will be the sum of the object-dimensions subtracted by the space dimension (compare the limit cycle example):

$$d_{over} = d_1 + d_2 - d_{space} \tag{1.11}$$

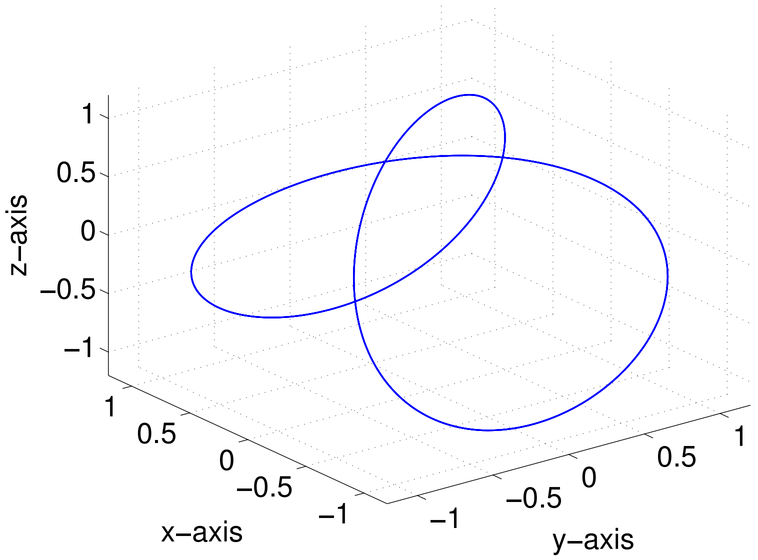


Figure 1.4.: 3-dimensional embedding of a Limit Cycle

For the situation of a self-intersection both objects become the same with dimension $d_1 = d_2 = d$.

$$d_{over} = 2 \cdot d - d_{space} \quad \Rightarrow \quad d_{space} = 2 \cdot d - d_{over} \tag{1.12}$$

For an embedding there must not be any overlap. Hence the dimension of the overlap d_{over} must be at least -1 or smaller:

$$d_{space} \geq 2 \cdot d + 1 \tag{1.13}$$

And finally one receives the equation provided by Whitneys embedding theorem.

Remark: Whitneys embedding theorem is powerful. It delivers important information about the optimal dimension for reconstruction. Nevertheless it does not say that nearly every map can be used for Phase Space Reconstruction. The reason is that possible maps for reconstruction only depend on one component of the original attractor per definition. This means that the group of all possible reconstruction functions is only a vanishing small subset of all possible maps. Because of this strong restriction the probability to get an embedding could be smaller than 1, it could possibly also be 0. Also it is not clear if this additional restriction allows to change the mapping in a way that a false embedding become a true one. Hence it is important to keep in mind that Whitneys embedding theorem says something about the optimal dimension for reconstruction but it does not say anything about the probability that a reconstruction is an embedding!

1.2.10. False Nearest Neighbours

Whitneys embedding theorem delivers information about the optimal embedding dimension for a known attractor. But for measurement data the underlying system normally is not known. Therefore it is necessary to have additional tools for estimation of the embedding dimension. One possible solution is the so called "False-Nearest-Neighbours"-method. It bases on the idea that in the case of a too low dimensional embedding points will become direct neighbours which normally lie far away of each other. At the same time it is not possible to disperse real neighbours by using a too high embedding. To find the optimal embedding dimension d one has just to increment the embedding dimension starting with $d = 1$ and to test which points are neighbours in this actual embedding. Increasing the dimension this number of neighbours will decrease: The false nearest neighbours will be reduced step by step. The optimal embedding dimension will be reached when the number of neighbours does not decrease anymore. This happens when only true nearest neighbours remain. Hence the attractor is embedded correctly.

Imagine a simple limit cycle. Using a too high embedding the attractor will still remain as limit cycle. Direct neighbours will remain as direct neighbours. Using only a one-dimensional embedding the attractor is projected on one line. One sides of the limit cycle will be folded on the other side. Therefore some points that were before on the opposite will now become neighbours.

This method is not used on the following pages. For analysing the reconstruction itself it is more useful to work with systems that are fully known. Anyhow it is important to be familiar with this method when applying it to real measurement data. A more detailed description concerning the optimal embedding dimension and a presentation of alternative methods can be found e.g. in [15] or [29].

1.2.11. Fractal dimension

Another important property of chaotic attractors is its fractal structure: Mostly they do not have an integral dimension. To understand this phenomena is it necessary to know how the dimension can be measured. Actually there are several measures for it. The best measure to understand its functionality is probably the box dimension:

The box dimension takes a look at the scaling of an object. For that purpose the object is covered with boxes of the same edge length ϵ. The relation between number of needed boxes N and edge length ϵ delivers the dimension D of the object:

$$N(\epsilon) \propto \epsilon^{-D} \tag{1.14}$$

Choosing ϵ big means that the resolution for analysing the attractor is low. Hence the definition of the Boxdimension uses the limit of $\epsilon \to 0$ to make sure that every detail of the attractor is included in the result. After some transformations one gets the final formula:

$$D_{Box} = -\lim_{\epsilon \to 0} \frac{\ln(N(\epsilon))}{\ln(\epsilon)} \tag{1.15}$$

Another measure for the dimension is the so called information dimension D_I. It bases on the same idea but works with probabilities. This definition delivers slightly different results (in general one can say $D_I \leq D_{Box}$) but can also be used to describe fractal objects:

$$D_I = -\lim_{\epsilon \to 0} \frac{\sum_{i=1}^{N(\epsilon)} P_i(\epsilon) \log(P_i(\epsilon))}{\log(\epsilon)} \quad \text{with} \quad \sum_{i=1}^{N(\epsilon)} P_i(\epsilon) = 1 \tag{1.16}$$

$P_i(\epsilon)$ is the probability to find a particle in the i-th space-area (the probability to find the particle in the whole space is 1).

Applying this concept on real time series also the numerical aspect becomes relevant. A limited set of points has dimension 0 by definition. Hence for a discrete dataset the dimension of the underlying attractor has to be approximated. One of the numerical fastest ways is to calculate the correlation dimension d_2. For this dimension first the correlation integral has to be calculated:

$$C(\epsilon) = \lim_{n\to\infty} \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{i-w} H(\epsilon - |s_j - s_i|) \tag{1.17}$$

N is the number of summands and n is the number of used points. s_i stands for the i-th point of the used data in phase space. H is the Heaviside-function which has value 1 for inputs greater 0 and value 0 for all other inputs and w is the Theiler window which prevents counting of temporally correlated pairs (otherwise the calculated dimension will be too small [31]). Hence the correlation integral counts all pairs of points which have a distance smaller than ϵ excluding pairs of points which were recorded approximately at the same time. Increasing ϵ the number of pairs with a distance less than ϵ will raise. The exponent of this raising is the correlation dimension d_2.

$$C(\epsilon) \propto \epsilon^{d_2} \quad \Rightarrow \quad d_2 = \lim_{\epsilon,\epsilon'\to 0} \frac{\ln\left(\frac{C(\epsilon)}{C(\epsilon'}\right)}{\ln\left(\frac{\epsilon}{\epsilon'}\right)} \tag{1.18}$$

Numerically d_2 will be derived by averaging d_2 for different ϵ or by fitting a straight line in a log-log-plot of $C(\epsilon)$ over ϵ.

For the following work it is not important to get to know about other measures of fractal dimension and differences between them, hence I will only provide this basic information. Important to know is that the fractal dimension should only depend on the measure that is used and the attractor that is measured but not on its embedding. Hence the fractal dimension is such as the Lyapunov-exponents a good measure to test if an embedding was successful. For an correct embedding one should recieve the correct fractal dimension.

1.3. Some dynamical systems

For testing the functionality of reconstruction methods dynamical systems are needed. PSR is mostly applied to real measurement data. By contrast for testing purposes it is more helpful to use systems which are already known exactly. At the same time most interesting systems are strange attractors. Hence I will use four different chaotic systems for presenting and comparing different reconstruction methods. The choice was done arbitrarily so any other attractor could also be used for the following parts.

1.3.1. Lorenz Attractor

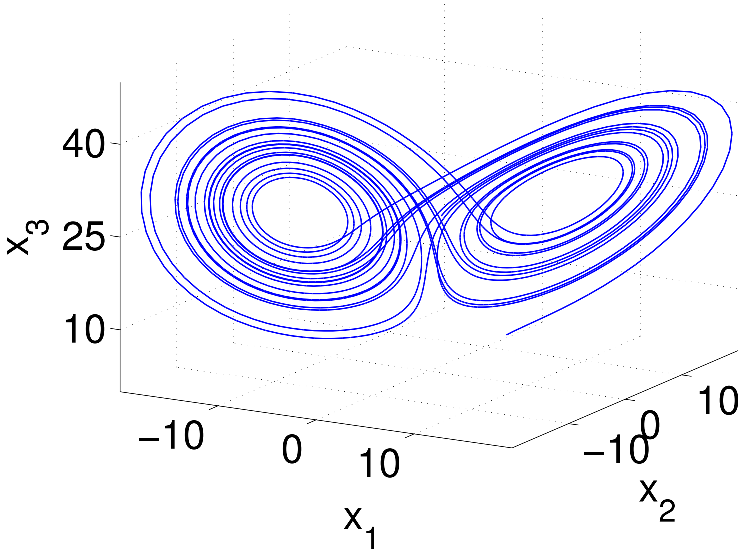


Figure 1.5.: Lorenz attractor

One of the first presented deterministic systems with chaotic behavior was a simple model for cellular convection invented by E. Lorenz [17]. Because of its low dimensionality and its - for a chaotic system - clear structure it is one of the most used examples for strange attractors. It is described by the following equations:

$$\dot{x}_1 = \sigma(x_2 - x_1) \tag{1.19}$$

$$\dot{x}_2 = -x_1x_3 + rx_1 - x_2 \tag{1.20}$$

$$\dot{x}_3 = x_1x_2 - bx_3 \tag{1.21}$$

It becomes chaotic for certain parameters. One possible choice to produce chaotic behaviour which is used for all following examples is $\sigma = 10$, $r = 28$ and $b = 8/3$

(fig. 1.5). For calculations a dataset was used with starting coordinates $x_1 = x_2 = x_3 = 5$, timestep $\Delta t = 0.01$ and length of 100000 points.

1.3.2. Rössler Attractor

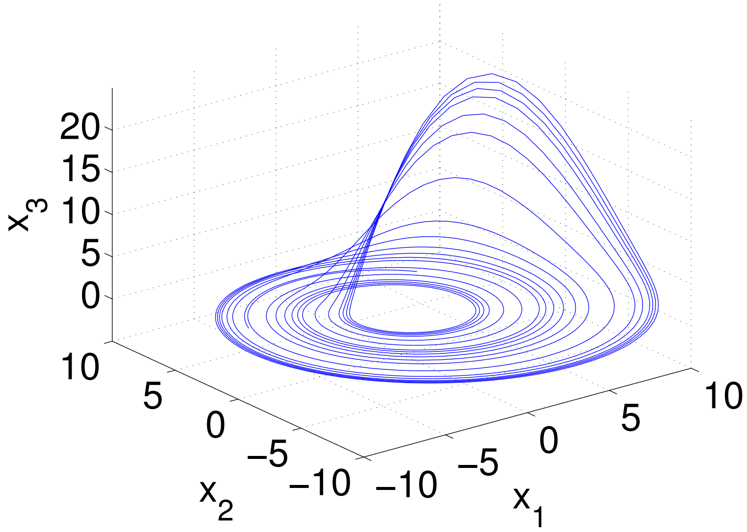


Figure 1.6.: Rössler attractor

The Lorenz attractor is simple for a chaotic system. However it is still too complex for some analysing methods. Therefore O.E. Rössler built a "model of a model" which should described the basic characteristics of the Lorenz system [21]. Instead of two spirals this system only contains one (fig. 1.6). For rebuilding the Lorenz system with the used parameters above Rössler got:

$$\dot{x}_1 = -(x_2 + x_3) \tag{1.22}$$

$$\dot{x}_2 = x_1 + 0.2x_2 \tag{1.23}$$

$$\dot{x}_3 = 0.2 + x_3(x_1 - 5.7) \tag{1.24}$$

Another interesting property of the Rössler system is its mixture of chaotic and periodic elements. Taking a look at x_1 or x_2 one will find functions with well defined periodicities combined with chaotic amplitudes.

For my work I used a dataset with 100000 points, starting from $x_1 = x_2 = 4$, $x_3 = 0$ with timestep $\Delta t = 0.005$.

1.3.3. Hénon Map

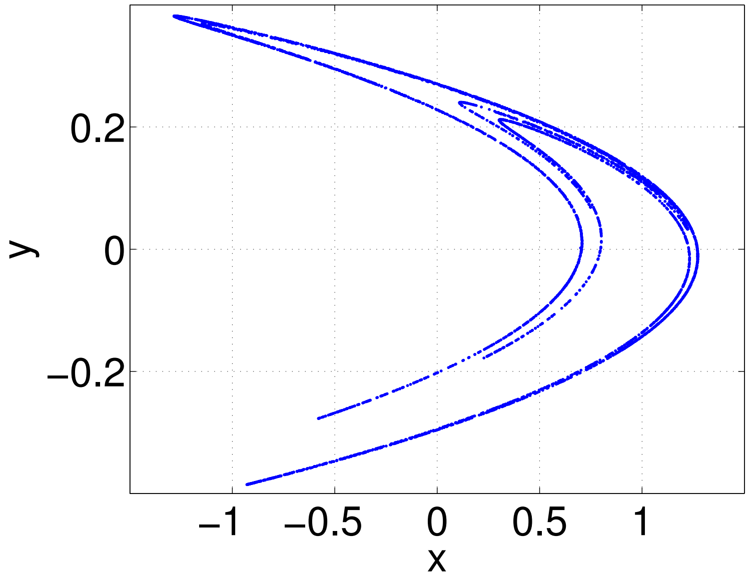


Figure 1.7.: Hénon attractor

Another ansatz to build up a simple chaotic system was done by Hénon [13]. The goal was to create a system which has similar properties compared to a Lorenz system but with simpler equations. This should reduce the amount of calculations needed for a simulation. Therefore he tried to rebuild the Poincare map of a strange attractor instead of rebuilding the whole trajectories. The result was a two-dimensional discrete map with chaotic behaviour.

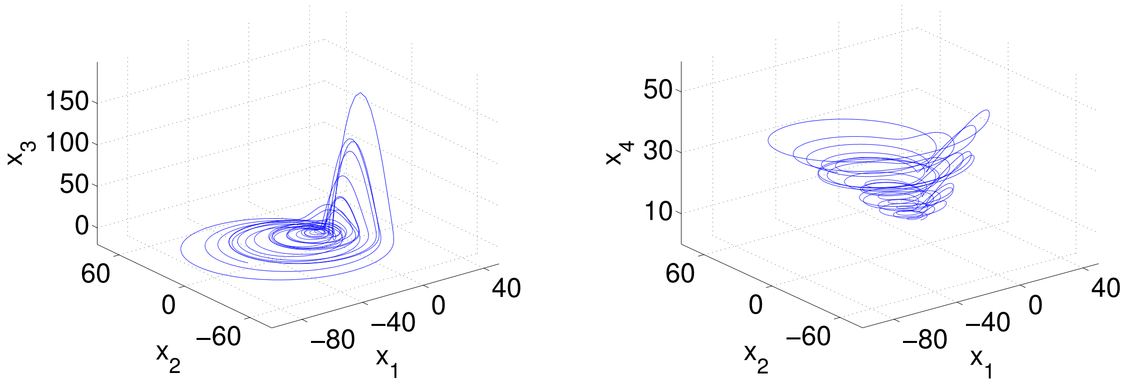
$$x_{n+1} = 1 + y_n - a x_n^2 \tag{1.25}$$

$$y_{n+1} = b x_n \tag{1.26}$$

For $a = 1.4$ and $b = 0.3$ one gets a chaotic behaviour of this system. It is a good example to test the embedding of discrete systems and will be used in this way on the following pages.

The used dataset has 100000 points and starts from $x_1 = 0.7$, $x_2 = 0.002$.

1.3.4. Hyperrössler Attractor



(a) x_1,x_2 and x_3

(b) x_1,x_2 and x_4

Figure 1.8.: Hyperrössler attractor

Another class of chaotic systems are so called "hyperchaotic systems". In contradiction to chaotic objects these systems contain at least two positive Lyapunov Exponents. Hence hyperchaos is more chaotic and higher dimensional than chaotic systems. I will use a 4-dimensional hyperchaotic system invented by Rössler ([23], [22]) which is closely connected to the Rössler system introduced before.

$$\dot{x}_1 = -x_2 - x_3 \tag{1.27}$$

$$\dot{x}_2 = x_1 + 0.25x_2 + x_4 \tag{1.28}$$

$$\dot{x}_3 = 3 + x_1 x_3 \tag{1.29}$$

$$\dot{x}_4 = -0.5x_3 + 0.05x_4 \tag{1.30}$$

It is a good example for testing reconstructions with a more complex system. For that purpose I use a time series with 150000 points starting from $x_1 = -10$, $x_2 = -6$, $x_3 = 0$, $x_4 = 10$ with a time step $\Delta t = 0.001$.

1.4. Common methods for attractor reconstruction

1.4.1. Time Delay

Time delay was one of the first published methods for reconstruction (see [30] [20]) and is today the most frequently used one. One reason is probably that Floris Takens could present very early a mathematical proof for its functionality. The other reason is its good usability: The additional coordinates are built using the original time series shifted in time with a delay τ. So the reconstructed system $v(t)$ has the structure:

$$v(t) = (x(t), x(t+\tau), x(t+2\tau), ...) \tag{1.31}$$

This is easy to handle for real measurement data because the reconstruction itself does not need any complex calculations. Instead the time series can be used directly. It also has the benefit that every dimension shows the same noise level.

Analysing discrete systems a reconstruction with $\tau = 1$ can also be interpreted very well because it follows the structure of discrete equations ($x_{n+1} = f(x_n)...$). Hence one can say that it is some kind of natural reconstruction for these systems.

A disadvantage is the introduction of the parameter τ which has to be determined first. Takens embedding theorem says that nearly every τ gives a correct embedding but there are qualitative differences between these choices. Therefore it is necessary to calculate a proper delay τ first. At the same time the possibilities of optimizing the reconstruction is limited due to its definition. To get better results multiple time delays were introduced which make the reconstruction even more complex.

$$v(t) = (x(t), x(t+\tau_1), x(t+2\tau_2), ...) \tag{1.32}$$

Nevertheless most papers published about attractor reconstruction are working only with time delay reconstruction and many papers are offering more efficient ways to calculate the correct embedding dimension and embedding delays.

1.4.2. Derivation

Another early published but not so well known method is a reconstruction using the derivatives of the original function (see [20]).

$$v(t) = (x(t), \frac{dx(t)}{dt}, \frac{d^2x(t)}{dt^2}, ...) \tag{1.33}$$

Whereas most dynamical systems can be reconstructed very well using time delay coordinates there are also many cases for which the derivation method has some advantages. There are more numerical calculations necessary to get the new coordinates. In return this method needs only the reconstruction-dimension as input, no more values are introduced. This reconstruction gives often better results if the dynamical system bases on differential equations. In this cases the reconstruction has a similar structure as the original system and it is often possible to get a proper reconstruction in a lower dimension compared to time delay.

Hence time delay delivers correct reconstructions for nearly every system but often it is possible to get the same results in a lower dimension using other methods. The latter bears big advantages because a lower dimension also means better opportunities for working with the system.

Because of its structural similarity to differential equations the reconstructed dimensions are often easier to identify with physical properties of a system and therefore the reconstruction is often better to explain physically compared to a reconstruction with time delay.

On the other hand the derivation method has the disadvantage that the coordinates have different noise levels because of their different dependency on frequencies (high frequencies are amplified through derivation).

1.4.3. Integration

Reconstruction with integration is the opposite method of derivation.

$$v(t) = (x(t), \int\limits_0^t x(t')dt', \int\limits_0^t \left(\int\limits_0^{t'} x(t'')dt'' \right) dt', ...) \tag{1.34}$$

It has similar pros and cons but does not work for real data as good as derivation. The reason is its high dependency on low frequent noise. This leads to the problem that it only gives useful results in a combination with some moving average noise reduction or a similar technique. Some further discussions and explanations concerning integration reconstruction were done by Blasius et al. [3] and Gilmore [7].

1.4.4. Hilbert-Transformation

Hilbert Transformation is a method which is used to calculate the imaginary part of a signal. For this reason it is often applied in electronics to determine the complex impedance of an electrical system. This is its main application but it can also be used for reconstruction of other systems. Its behaviour can be described best taking a look at the frequency domain. There it is a rotation around the origin. Positive frequencies are rotated $+\pi/2$, negative ones $-\pi/2$.

$$v(t) = (x(t), IFFT\left(i \cdot sign(\omega) \cdot FFT\left(x(t)\right)\right)) \tag{1.35}$$

(FFT = Fast Fourier Transform, IFFT = Inverse Fast Fourier Transform)

The method needs some numerical calculations to get the additional dimension but is easy to use because of the absence of selectable parameters. It also does not change the noise level and conserves the most important properties of the original time series. The big disadvantage is its limitation to only one additional reconstruction dimension. So only a two-dimensional embedding is possible. This disqualifies the method for most situations. Anyhow it will help to understand the following ideas and will also be useful within the new methods presented in this work.

2. Look at the frequency-domain

2.1. Motivation

As we have seen above there are a few different methods for achieving the same goal. At the same time all these reconstructions have a quite different appearance in the time domain. Therefore the questions are:

- Why are all these methods producing similar results? (they are all offering proper embeddings)
- Are there more reconstructions that can be used?
- What are the requirements for a map that it can be used as a reconstruction?

To answer this questions one needs to find a representation for which all these methods get a similar shape. The following pages will show that the frequency space is offering such a structure. There are primary two reasons that can motivate the use of the frequency space:

1. Using a time-series it is actually hard to imagine "where" information about the other dimensions is stored. Reconstruction appears in a way magical. Working in the frequency space the reconstruction process becomes more evident: Instead of time one handles with frequencies and they are closely connected with motion on a circle. Therefore at least one single frequency has already some kind of two-dimensional interpretation. Using more frequencies a more-dimensional interpretation becomes more obvious.

   For understanding of this motivation let us take the picture of a frequency as a motion on a circle: Intuition will predict that every frequency contained in one dimension should also be contained in all other dimensions. Intuitively the only exception of this should be the non-generic case of a dimension that stands orthogonal on the flow.

2. Before Phase-Space-Reconstruction was invented the power spectrum of a function was used to determine chaotic behaviours. On its basis it was easier to detect signs for chaotic characteristics. But the power spectrum has the big disadvantage that the phase-information is neglected. Therefore it cannot be used for reconstruction and Takens began to use the time-series directly. But still it remains reasonable to work with frequencies for reconstruction. Hence the solution is to work with the fourier transform which uses a frequency-representation without neglecting the phases.

Most likely this argumentations are easy to challenge but one has to keep in mind that these are only motivations. They should be understood as indications for the following pages.

2.2. Common methods

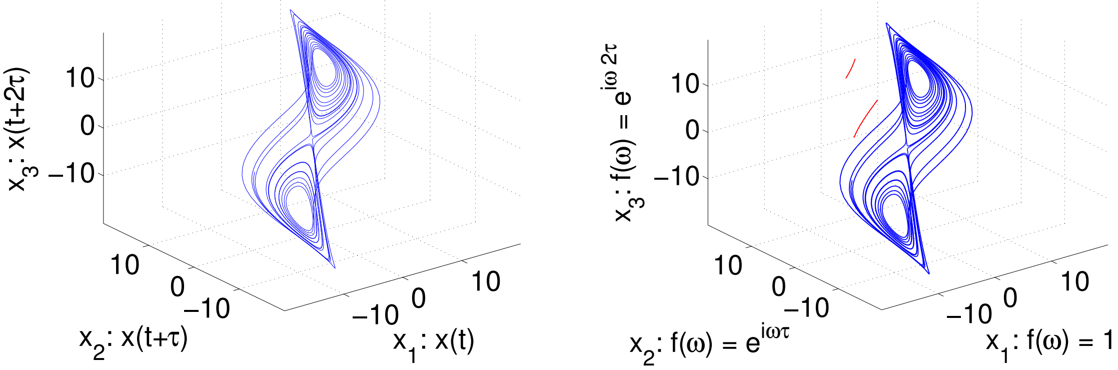
2.2.1. Time Delay

Using the Fourier-transformation a time delay $n\tau$ in the time domain becomes a multiplication with the exponential function $e^{i\omega n\tau}$ in the frequency domain.

$$x_{n+1}(t) = x(t + n\tau) \quad \Rightarrow \quad \underline{\underline{\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot e^{i\omega n\tau}}} \tag{2.1}$$

It has to be remarked that this representation is valid for the case of infinite time series. For real data one has to deal with finite data. Also the numerical ways of getting the additional coordinates are completely different. For the classical method one has only to shift the data. In the new case one has to transform the data to the frequency domain, multiply it with a frequency-depending factor and have to transform it back to the time domain. So it is necessary to compare both realizations to test the assumption that both ways producing the same result.

Figure 2.1 shows the example of a Lorenz attractor [17] reconstructed with both realizations. As expected both ways provide the same results with only two small



(a) classical

(b) emulated in the frequency domain

Figure 2.1.: Time-Delay reconstruction of a lorenz system using the x-axis: Comparison between the classical method shifting the data (2.1(a)) and the emulated version in frequency-space using a multiplication with $\exp(i\omega n\tau)$ (2.1(b)).

exceptions (marked red, fig. 2.1(b)). These wrong trajectories are caused by boundary effects due to the finiteness of the time series. Equation 2.3 shows the difference of the finite calculation. Because of the limited dataset one gets some additional terms which are rising with time-shift τ:

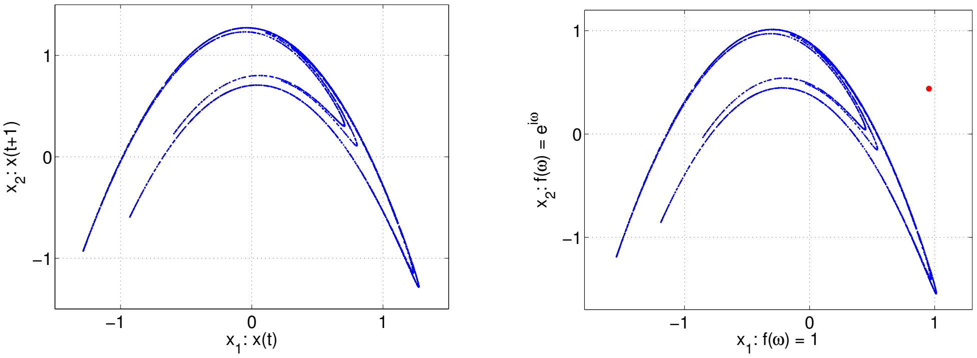
$$\begin{aligned}\sum_{t=0}^{n} x(t+\tau)e^{-i\omega t} &= \sum_{t=\tau}^{n+\tau} x(t)e^{-i\omega(t-\tau)} \qquad (2.2)\\ &= \sum_{t=0}^{n} x(t)e^{-i\omega(t-\tau)} + \underbrace{\sum_{t=n+1}^{n+\tau} x(t)e^{-i\omega(t-\tau)} - \sum_{t=0}^{\tau-1} x(t)e^{-i\omega(t-\tau)}}_{\text{difference to continuous case}}\end{aligned}$$

One could use now the exact formula but this would gain the complexity of further calculations. Instead it is more practicable to keep this problem in mind and deal with it.

In case of time delay both lines have the length τ and can be easily removed just cutting the ends of the reconstructed datasets (in the classical case we also have to cut the ends of our datasets because of the shifting). Finally after cutting one receives two reconstructed datasets with the same length and values. So exactly the same reconstruction is received with the continuous formula using some corrections at the end.

One remark has to be made at this stage: The result above is only valid if no information is lost due to the Fourier Transformation. This induces that the sampling rate fulfilled the Nyquist-Shannon sampling condition!

A similar result is found looking at the example of a discrete Hénon map (fig. 2.2).



(a) classical

(b) emulation in frequency space

Figure 2.2.: Time-Delay reconstruction of a discrete hénon map using the x-axis: Comparison between classical method (2.2(a)) and emulation in frequency space (2.2(b)). The used delay is $\tau = 1$.

The classical method produces nearly the same result compared to the reconstruction in fourier space. Also both reconstructions are similar to the original system. As in the example of Lorenz system the reconstruction delivers one wrong point (only one point because of $\tau = 1$) marked red in fig. 2.2(b).

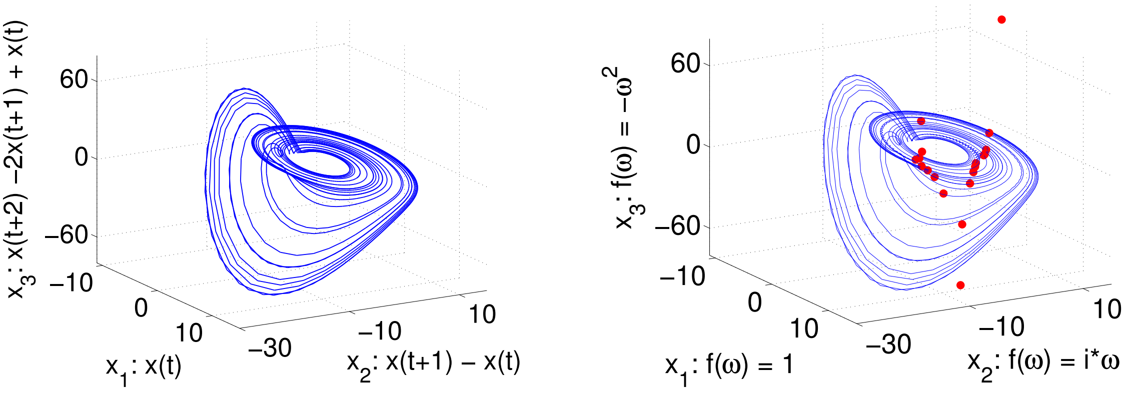
Another visible but not fundamental difference is the absolute position of the attractor. This is caused by the syntax I used for reconstruction in frequency space: Using time-delay the component for $\omega = 0$ has normally to be multiplied with $\exp(i\tau \cdot 0) = 1$. My explicit realization in frequency space instead sets this component to zero. The reason for this choice is to prevent singularities for $\omega = 0$ which are e.g. produced by integration reconstruction in chapter 2.2.3. Actually this constant shift does not infect the quality of an embedding. Hence it does not matter if this part is removed or not.

2.2.2. Derivation

Derivation of a function in time domain becomes a multiplication with the function $f_1(\omega) = i\omega$ in frequency domain. Hence the transformation gets the structure:

$$x_{n+1}(t) = \frac{d^n x(t)}{dt^n} \quad \Rightarrow \quad \underline{\underline{\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot (i\omega)^n}} \tag{2.3}$$

Fig.2.3 compares the classical derivation reconstruction of a Rössler system [21] with its emulation in frequency space.



(a) classical

(b) emulated in frequency space

Figure 2.3.: derivation reconstruction of a Rössler attractor using the x-axis: Comparison between classical realization (2.3(a)) and its idealized counterpart in frequency space (2.3(a)).

Also for derivation reconstruction one receives similar results for both realizations. But compared to the results of time-delay the differences are bigger: In contrast to time-delay the disturbances are not primary produced due to the finiteness of the used time series. More problematic becomes the discretization of data: For discrete data it is not possible to calculate its derivation in the time domain. Instead the difference between two following points is used as approximation. Coevally the counterpart of an exact derivation was used in frequency space for reconstruction. In case of continuous data the result would be the same but for discrete data the operations differ. Nevertheless the results are still comparable. The disturbances in the alternative reconstruction are only situated at beginning and end of the time series. Therefore it is easy to remove these parts.

Alternatively it is also possible to use the exact counterpart of the used deviation instead of the derivation (eq. 2.4).

$$x_2(t) = \frac{x(t+dt)-x(t)}{dt} \quad \Rightarrow \quad \underline{\underline{\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot \left(\frac{e^{i\omega dt}-1}{dt}\right)^n}} \tag{2.4}$$

Fig. 2.4 shows the result applying this accurate emulation of the used reconstruction. The difference to the classical case (fig. 2.3(a)) are two wrong points caused again by boundary effects due to a delay of $\tau = 1$ (beyond the plotting range of the figure).

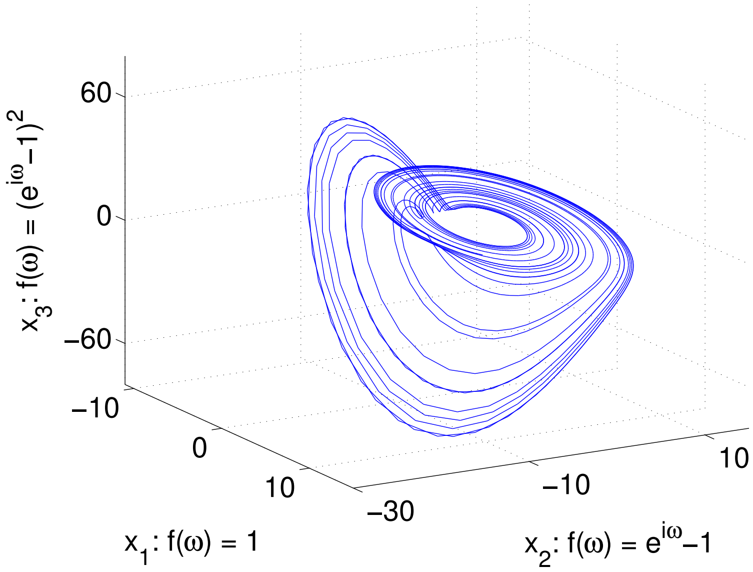


Figure 2.4.: alternative derivation reconstruction of a Rösslersystem using the x-axis: exact emulation of the classical reconstruction reconstruction (2.3(a)).

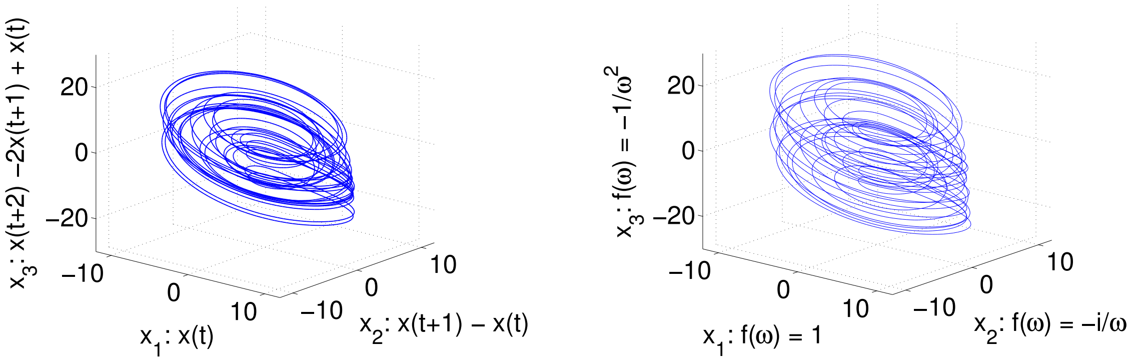
Because of its easier handling only equation 2.3 will be used for the following examples. But there is no problem to use also the exact transformed equation. Both methods can be applied in the same way and the choice only depends on the requirements the experimentalist has for reconstruction.

2.2.3. Integration

Without taking notice of the mentioned problems of integration reconstruction (chapter 1.4.3) the integration reconstruction becomes a division with $i\omega$ in frequency space. As the counterpart of derivation reconstruction this result gets clear immediately.

$$x_2(t) = \int\limits_0^t x(t')dt' \quad \Rightarrow \quad \underline{\underline{\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot \frac{1}{(i\omega)^n}}} \tag{2.5}$$

As explained before this result normally cannot be applied directly on a time series because of its magnification of low frequent disturbances. However the aim at this point is to understand the reconstruction structurally and not to find the practically most valuable method. This can be reached best by comparing first the idealised and most simplified reconstruction version before taking a look at its practical issues. Hence 2.5 gives a comparison between the skeletal structures of classical integration reconstruction and its emulation in frequency space.



(a) classical

(b) emulated in frequency space

Figure 2.5.: reconstruction with integration of a Rössler attractor using the x-component: Comparison of the standard method (2.5(a)) with its emulated counterpart (2.5(a)) and without additional filtering.

As one can see the result does not embed the dynamics correctly. But at the same time it is noticeable that both methods deliver similar results. Analogue to derivation reconstruction the emulated result differs more of its prototype as e.g. for time-delay. And as for derivation reconstruction this error is caused by the discretization of data: Because of this condition there is no real integration in the time domain. Instead all points are summed up. This difference leads to the differences in the reconstruction. It has also to be remarked that there is a small discrepancy between the equation shown above and the used equation in frequency space: As already denoted in chapter 2.2.1 I added a small change to the sourcecode that was used for emulation: The component for a vanishing frequency is set to zero. This becomes here important because otherwise one would get a singularity for $\omega = 0$. Coevally this fact is closely connected to the problems using integration reconstruction: Also frequencies close to zero are amplified too much. In the presentet case this does not affect the first reconstructed component but leads to a divergency in the second reconstructed dimension because of its quadratical dependency on ω.

Classically this problem is solved by applying a moving average filter to the data. Working in the frequency space this can be done much easier just by introducing a small offset. Figure 2.6 shows the result using this modification.

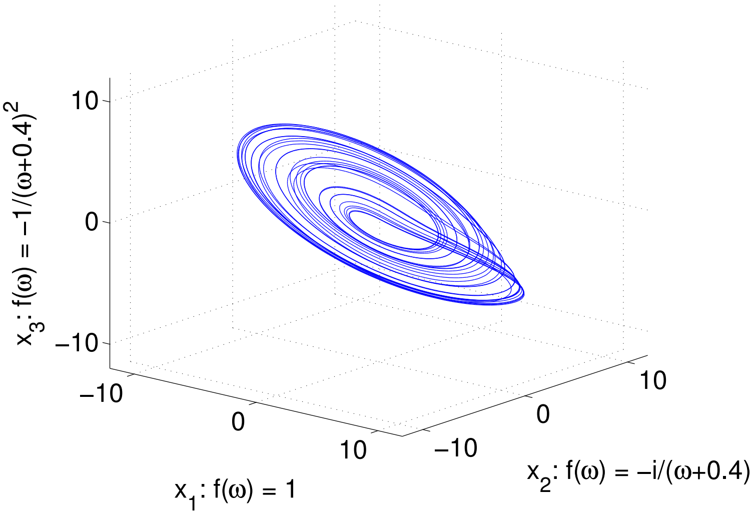


Figure 2.6.: optimized integration reconstruction of a Rössler system: Lowfrequent disturbances are neglected by introduction of a small offset.

The emerged embedding is not the best but all topological properties are mapped correctly. Also it is easy to identify the Rösslersystem. Hence this reconstruction gives a first example for the advantages of reconstruction in the frequency domain: It is easier to modify reconstructions and to optimize them for a given dataset.

2.2.4. Hilbert-Transformation

In the case of the Hilbert transformation no comparison is necessary: The reconstruction is already done in the frequency domain. At the same time the structure in the frequency domain is the simplest one of all presented reconstructions:

$$\underline{\underline{\tilde{x}_2(\omega) = \tilde{x}(\omega) \cdot i \cdot sign(\omega)}} \tag{2.6}$$

Worth mentioning is its strong affinity to reconstruction with derivation or integration. The only difference is that derivation and integration are working with the frequency as a weighting function instead of only using its sign. This fortifies the relevance of Hilbert-Transformation as a reconstruction that is reduced to the essential.

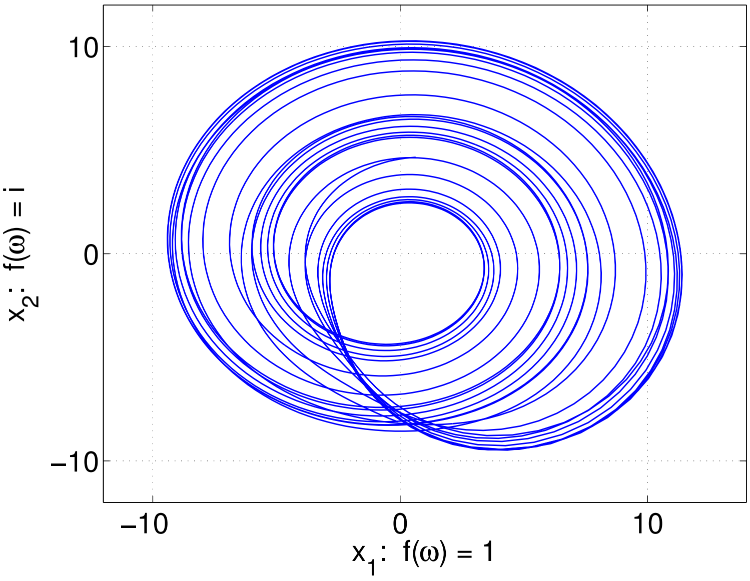


Figure 2.7.: Hilbert Reconstruction of a Rössler system using the x-component

2.3. Generalization

2.3.1. Overview

I showed that it is possible to emulate all presented reconstructions in frequency space. Considering the mentioned limitations the results of classical and emulated technics delivering nearly the same results. Hence in the following the emulations will be used as proper replacements for the classical methods. The next step is to compare them with each other to find similarities. For this purpose table 2.1 provides an overview about the presented methods.

As already known there are not many similarities between these methods looking at the time domain: It is nearly impossible to identify a general structure for all reconstructions. Taking a look at the frequency space one will find a much more

method	time domain	frequency domain
Time Delay	$x_{n+1}(t) = x(t + n\tau)$	$\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot e^{i\omega n\tau}$
Derivation	$x_2(t) = \dot{x}(t), \quad x_3(t) = \ddot{x}(t),...$	$\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot (i\omega)^n$
Integration	$x_2(t) = \int\limits_0^t x(t')dt',...$	$\tilde{x}_{n+1}(\omega) = \tilde{x}(\omega) \cdot \frac{1}{(i\omega)^n}$
Hilbert	$x_2(t) = IFFT\,(i \cdot sign(\omega) \cdot \tilde{x}(\omega))$	$\tilde{x}_2(\omega) = \tilde{x}(\omega) \cdot i \cdot sign(\omega)$

Table 2.1.: Comparison of most common reconstruction methods displayed in time and frequency domain.

structured situation: All reconstructions can be described as a product between original time-series and a frequency-depending function $f(\omega)$.

$$\boxed{\boxed{\tilde{x}_n(\omega) = \tilde{x}(\omega) \cdot f_n(\omega)}} \tag{2.7}$$

That means that one needs only the original time series $x(\omega)$ and n different reconstruction functions $f_n(\omega)$ (one function per embedding dimension) to reconstruct the attractor. Hence every reconstructed dimension provides the frequency structure of the original time series with some variations in phase and amplitude. This clear structure of reconstruction leads to the questions of the beginning: Are there more possible reconstruction functions? Is there perhaps a whole class of functions which can be used for reconstruction? What are the requirements for $f(\omega)$ to be a proper reconstruction function?

2.3.2. Requirements on the reconstruction functions $f_n(\omega)$

To get information about the properties of the reconstruction functions $f_n(\omega)$ it is necessary to think about the requirements for a good reconstruction. One important requirement is that all reconstructed dimensions must be real-valued. Because we are working with complex values in the frequency domain a real-valued output leads to restrictions for the reconstruction function:

$$
\begin{aligned}
x(t) = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{+\infty} (a(\omega) + ib(\omega))\, e^{-i\omega t} d\omega \qquad x(t), a(\omega), b(\omega) \in \mathbb{R} \\
\Rightarrow \forall t \in \mathbb{R}: \quad 0 = \int\limits_{-\infty}^{+\infty} b(\omega)\cos(\omega t) - a(\omega)\sin(\omega t) d\omega \\
\Rightarrow \quad \underline{\underline{a(\omega) = a(-\omega)}} \quad \wedge \quad \underline{\underline{b(\omega) = -b(-\omega)}} \qquad (2.8)
\end{aligned}
$$

This means that the Fourier transformed of a real valued time series has the property: $\tilde{x}(\omega) = \overline{\tilde{x}(-\omega)}$. The function itself and its complex conjugated of the negative frequency needs to be the same.

In the case of a reconstructed dimension the Fourier transformed is a multiplication of the reconstruction function and the Fourier transform of the original time series $f_n(\omega)\tilde{x}(\omega)$:

$$f_n(\omega)\tilde{x}(\omega) \overset{!}{=} \overline{f_n(-\omega)\tilde{x}(-\omega)} \quad \wedge \quad \tilde{x}(\omega) = \overline{\tilde{x}(-\omega)} \qquad (2.9)$$

Hence one property of the reconstruction functions $f_n(\omega)$ is:

$$\boxed{f_n(\omega) = \overline{f_n(-\omega)}} \qquad (2.10)$$

Another requirement is the linear independency of all reconstruction functions:

$$\boxed{f_1(\omega)...f_n(\omega) \text{ linearly independent}} \qquad (2.11)$$

This is necessary to prevent same reconstruction results for different reconstruction dimensions. In phase space all components needs to be linearly independent. Otherwise the embedding will be flat and could also be presented in less dimensions. Hence in this case at least one dimension must be redundant. Because of the linearity of Fourier transformation this requirement can directly be adopted to frequency space.

Besides these two obvious requirements there must be even more. In the following part I will present a third requirement in a heuristic way. An accurate description of this part would be very useful and should be aimed at future investigations.

Using the limitations above $f_n(\omega)$ can still be nearly everything. So it would be also possible to use e.g. $f_n(\omega) = random$ (fig. 2.8) or to use functions which are discontinuous at every point. As shown in fig. 2.8 for the random case this obviously does not work. In fact this would mean that one could produce every result one wants to have just by choosing the reconstruction functions. It would be possible to eliminate the original signal nearly completely. This cannot deliver proper embeddings!

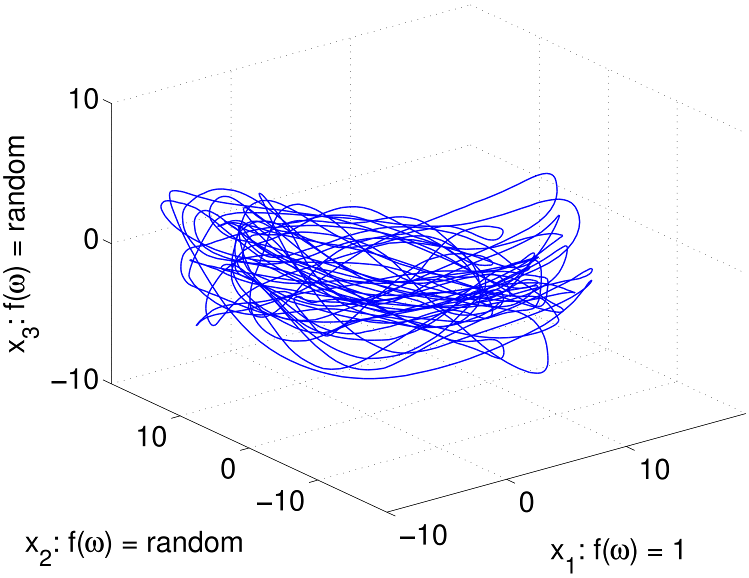


Figure 2.8.: Failed reconstruction of a Rössler system using random numbers as reconstruction functions

For an accurate reconstruction it is necessary to conserve the basic elements of the original system. The ideal reconstruction changes the accent within the time series but it does not add or remove information. The reconstruction functions themselves should be neutral. Mathematical imprecise this means that the functions $f_n(\omega)$ have to be smooth. And in fact all classical functions are continuous and differentiable in ω except the point $\omega = 0$.

So a good but perhaps not necessary postulation should be the continuity of $f_n(\omega)$.

$$\boxed{f_n(\omega) \text{ continuous}} \tag{2.12}$$

Also a stronger postulation could be necessary. A continuous function could also be able to "overwrite" the frequency information of the original system if the reconstruction function moves very fast in the order of magnitude of the original data. The reconstruction can also be understood as some kind of filtering [1] (effectively filtering has the same structure in frequency domain as the propagated reconstruction in equation 2.7).

Filtering means that some parts are highlighted while some other parts are attenuated. As more loading is done as more information of the original signal is lost. Without filtering we have the case $f(\omega) = 1$ and we do not get any reconstruction. At the same time too aggressive filtering as e.g. $f(\omega) = random$ removes the original information completely. Hence a good reconstruction needs to find a good balance between both extremes. Under this point of view it could be better to postulate slowly varying functions $f_n(\omega)$ (eq. 2.13) instead of only claim continuity (eq. 2.12).

$$\boxed{f_n(\omega) \text{ slowly varying}} \tag{2.13}$$

2.3.3. Syntax declaration

Before taking a look at some examples it is important to fix the used syntax for all reconstruction functions $f_n(\omega)$. Chapter 2.3.2 showed that there is an explicit relationship between $f(\omega)$ and its negative counterpart $f(-\omega)$ (equation 2.10). That means that one only needs to define the reconstruction functions for positive frequencies $\omega \geq 0$. Then the negative part is defined automatically.

So far all shown reconstruction functions were already fulfilling equation 2.10 in the presented structure. Anyhow there are many situations were the full function for the whole frequency-spectrum is more complex as the same function defined only for positive frequencies. For example the Hilbert transformation has $f(\omega) = i \cdot sign(\omega)$. Describing the behaviour only for positive frequencies one can write $f(\omega) = i$.

[1] Here the notion "Filter" is used in the physical meaning, not in the mathematical. Mathematically a filter discriminates more: Objects are either removed or left unaltered. For reconstruction is this behaviour too strong: information of the system is changed significantly. In contrast a physical filter only modifies the phases and amplitudes of different frequencies. Therefore it changes the signal quite less.

Another example is an integration with a small offset a. The reconstruction-function for the full space (2.14, left) is much more complex compared to the function only written for positive frequencies (2.14, right):

$$f(\omega) = \frac{1}{i \cdot sign(\omega)(|\omega| + a)} \quad \Leftrightarrow \quad f(\omega) = \frac{1}{i(\omega + a)} \tag{2.14}$$

A simpler structure allows to capture the kind of reconstruction more easily. At the same time no information is lost. For example in the last case the relationship to unmodified integration is much better to detect in the right declaration. From my point of view the higher complexity of the left formula is only confusing. Also under knowledge of the symmetry in frequency space the left equation can be derived easily of the right one if needed. Therefore in the following examples the reconstruction function will be only defined for positive frequencies!

> So keep in mind: $f(\omega)$ is defined only for $\omega \geq 0$! The behaviour for negative frequencies can be derived using equation 2.10.

2.3.4. Some examples

Using this new reconstruction method one gets a whole bunch of possible reconstruction functions. One interesting option is to change well known reconstructions slightly. Instead of $f(\omega) = i\omega$ (derivation) one can use $f(\omega) = \omega$ or $f(\omega) = i\sqrt{\omega}$. Both changes can help to produce a reconstruction similar to derivation reconstruction but with less dependency on high-frequent parts.

In the case of a 4-dimensional reconstruction of a signal one gets with derivation reconstruction terms up to the order of ω^3:

$$f_1(\omega) = 1 \quad f_2(\omega) = i\omega \quad f_3(\omega) = -\omega^2 \quad f_4(\omega) = -i\omega^3 \tag{2.15}$$

This means that this reconstruction is already useless for experimental data with low amount of noise. As a solution following reconstructions could be used:

$$f_1(\omega) = 1 \quad f_2(\omega) = i \quad f_3(\omega) = \omega \quad f_4(\omega) = i\omega \tag{2.16}$$

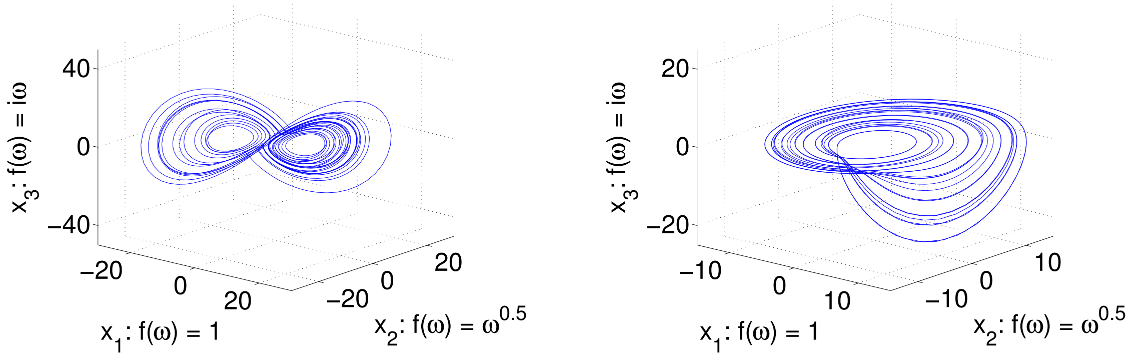
This reconstruction can be interpreted as a standard derivation reconstruction which is stretched by doubling each reconstruction dimension with a Hilbert transformation. Another choice would be:

$$f_1(\omega) = 1 \quad f_2(\omega) = i\sqrt{\omega} \quad f_3(\omega) = -\omega \quad f_4(\omega) = -i\omega^{3/2} \tag{2.17}$$

In this case the classical raise in orders of ω for each step is still conserved. Instead the raising order itself is decreased from 1 to $1/2$. Both cases reduce the highest order in ω strongly. Which method should be used depends on which properties should be conserved. Should the reconstruction use only integral orders of magnitude (eq. 2.16)? Or is it more important that for every dimension the order of magnitude in ω is increased (eq. 2.17)? Furthermore it is still possible to create own reconstructions and to make stronger reductions.

Another idea is to use functions with a low gradient for high frequencies as e.g. $\arctan(\omega)$ or $\log(\omega + 1)$. So one could use a derivation reconstruction and just replace ω with $\log(\omega + 1)$.

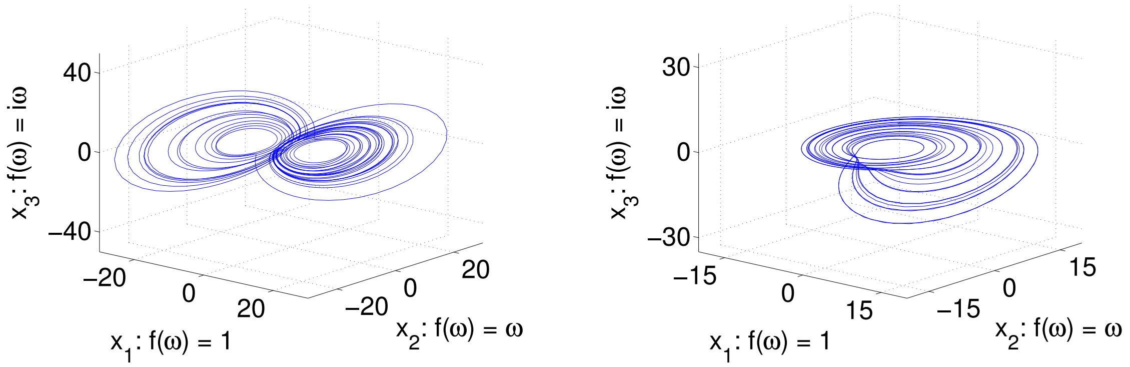
To show that all these choices delivering useful results I will present some plots of these customized reconstructions applied to Lorenz- and Rössler-systems:



(a) Lorenz attractor

(b) Rössler atractor

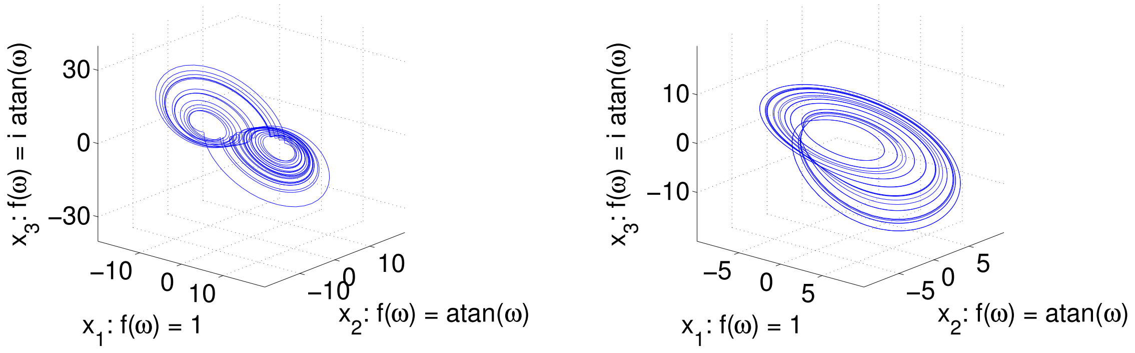
Figure 2.9.: Customized reconstructions using a modified derivation reconstruction with weakened frequency dependence



(a) Lorenz attractor

(b) Rössler atractor

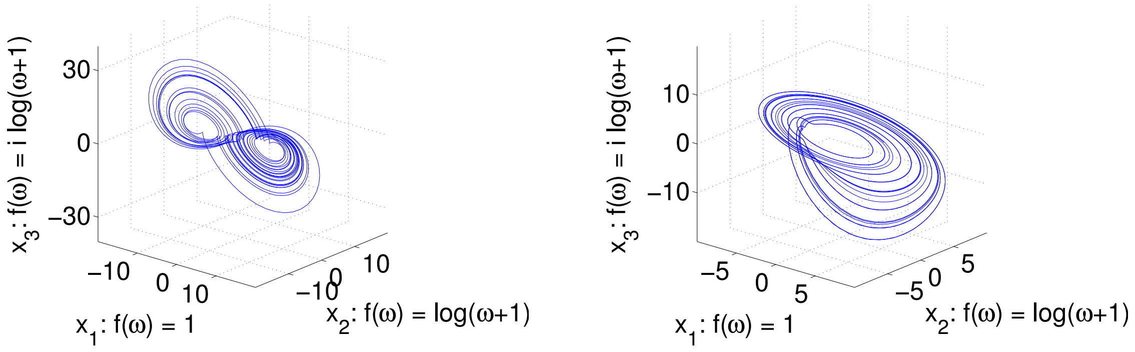
Figure 2.10.: Another alternative reconstruction using a modified derivation reconstruction with weakened frequency dependence



(a) Lorenz attractor

(b) Rössler attractor

Figure 2.11.: Reconstruction under use of the arcus-tangens-function



(a) Lorenz attractor

(b) Rössler attractor

Figure 2.12.: Reconstruction with a shifted logarithmic function

Also the d_2-dimension calculated for these alternative reconstructions delivers a good agreement with data derived of the original dynamical system:

method	d_2 Lorenz	d_2 Rössler
original system	1.99 ± 0.07	1.81 ± 0.05
derivation alternative 1	2.02 ± 0.09	1.83 ± 0.06
derivation alternative 2	2.01 ± 0.09	1.86 ± 0.07
atan reconstruction	1.95 ± 0.07	1.79 ± 0.07
log reconstruction	2.01 ± 0.07	1.81 ± 0.05

Table 2.2.: calculated d_2-dimension for original data and the reconstructed versions shown above

For calculations the program d2.exe of the TISEAN package [12] was used (r=1, R=10, t=1000). Unfortunalety the results for the used Rössler-system do not agree with the known dimension which has a value above 2. But the output for the reconstructed systems is consistent with the output for the original system. The mistake is probably caused by the high samplingrate of the attractor. Even under use of a wide Theiler-window this can lead to a lowering of the estimated dimension because the attractor appears at many parts 1-dimensional. To check if the embedding was succesful this fact does not play any role. The calculated d_2-dimension of the original system is also obtained by analysing the reconstructed systems. Hence the reconstructions have not distorted the result.

3. Numerical Verification

3.1. Method

So far most results were only checked by taking a look at the graph. For systems with simple and low dimensional structures this method works very well. In these cases it is mostly easy to recognize if an embedding was successful or not.

Nevertheless it is also necessary to test the results numerically. This helps to verify the results received discovering the graph and gives the opportunity to investigate more complex systems.

The most common numerical methods for analysing data of dynamical systems are the determination of Lyapunov exponents and fractal dimension. Both properties only depend on the dynamics and therefore on the topology of the system but not on its embedding. Hence the results should be the same for every embedding under use of the same time series.

To calculate it I used the function lyap_spec of the Nonlinear data analysis tool TISEAN [12]. This function computes the whole Lyapunov spectrum of a dynamic by estimating the local jacobian matrix at each point. Using the equations of a dynamical system one can obtain the jacobian matrix by calculating the time evolution of a group of points lying dense together. By using measurement data one normally does not have a dense group of starting points. Instead one has to work with the time evolution of only one point. To get that group of points the algorithm takes a starting point and searches for all points of this time series lying in a small ball of radius ϵ around this point. These points are used then as starting group. A more detailed description of the algorithm can be found in the work of Sano and Sawada [25] and Eckmann, Kamphorst, Ruelle and Ciliberto [4] on which the function lyap_spec bases. Besides the Lyapunov exponents in each direction one also get the Kaplan-Yorke-Dimension D_{KY} which is directly derived from the Lyapunov exponents as shown in eq. 3.1.

$$D_{KY} = k + \frac{\sum\limits_{i=1}^{k} \lambda_i}{|\lambda_{k+1}|} \tag{3.1}$$

Where λ_i are the Lyapunov exponents in order of its value beginning with the highest one and k is the highest integer for which the sum over all exponents from 1 to k remains non-negative. It is conjectured in [16] that this result should be equal to the information dimension D_I of the system.

For verification I used 5 time series: x-,y- and z-component of a Lorenz-system (to investigate differences within the same dynamical system) and the first components of Rössler- and Hyperrössler-systems (to obtain differences between different dynamical systems). In the next step all time series were reconstructed with a set of representative reconstruction functions (table 3.1). For every possible combination of three (in case of Hyperrössler four) different reconstruction functions the reconstruction were performed and the Lyapunov spectrum and Kaplan-York-Dimension were calculated. Because of the additional dimension of a Hyperrössler system a smaller selection of reconstruction functions was used for it. Otherwise the calculation time for the Lyapunov exponents would have been to long. Each original time series had 100000 points. To reduce calculation time and to remove errors caused by the limited length of the used time series only the points 10001 - 30001 were used.

3.2. Results

Before presenting the results the following remark has to be given: Theoretically the results for Lyapunov exponents and Kaplan-Yorke-Dimension should be completely independent of the choice of embedding. It should only deliver wrong results for failed embeddings. In reality the used methods are not as robust as wanted. Because they are working with some assumptions they are very sensitive to the kind of embedding. Hence it is necessary for analysing measurement data to

no.	$f(\omega)$	Part of small selection?
1	1	yes
2	i	yes
3	ω	yes
4	$i \cdot \omega$	yes
5	ω^2	yes
6	$i \cdot \omega^2$	no
7	$\sqrt{\omega}$	yes
8	$i \cdot \sqrt{\omega}$	no
9	$\arctan(\omega)$	yes
10	$i \cdot \arctan(\omega)$	no
11	$\log(\omega + 1)$	yes
12	$i \cdot \log(\omega + 1)$	no
13	$\exp(i\omega)$	yes
14	$i \cdot \exp(i\omega)$	no
15	$\exp(2i\omega)$	yes
16	$i \cdot \exp(2i\omega)$	no
17	$1/(\omega + 0.1)$	yes
18	$i/(\omega + 0.1)$	yes
19	$1/(\omega^2 + 0.1)$	yes
20	$i/(\omega^2 + 0.1)$	no
21	$\omega/(\omega + 1)$	no
22	$i \cdot \omega/(\omega + 1)$	no
23	$\omega^2/(\omega^2 + 1)$	no
24	$i \cdot \omega^2/(\omega^2 + 1)$	no

Table 3.1.: Set of test functions for numerical verification

choose the parameters of these functions very carefully under explicit knowlegde of the actual situation.

On the other hand adopted as a tool for comparing different embeddings it is a direct requirement that the calculation parameters kept the same for all embeddings. Otherwise it would not be possible to make a comparison. That means that this numerical measurements have to be treated in a rough way they are not built for. This leads to results that are not as good as probably expected. But it is actually the only way to get data for a numerical verification.

Nevertheless the findings are still useful. The mentioned problem produces only trouble for finding a proper measure to assess its results because it is not possible to distinguish between errors produced by the embedding and errors produced by the numerical calculation. To handle this problem I will use different measures to take a look at it with different views. Hence one will receive many different aspects of the result.

3.2.1. Results using the lyap_spec output directly

Probably the most evident way to analyse the results is to compare them under use of its relative and absolute errors calculated by lyap_spec itself with results directly derived out of the equations of the original system (table 3.4). For this purpose I checked for how many reconstructions the calculated Lyapunov exponents were (under adherence of its relative and absolute measurement errors) identically to the results of the original system (table 3.2).

used data	L_1 rel.	L_1 abs.	L_2 rel.	L_2 abs.	L_3 rel.	L_3 abs.
Rössler X	38,44%	69,61%	48,81%	71,34%	11,31%	14,87%
Lorenz X	43,28%	83,55%	60,18%	90,46%	10,23%	22,38%
Lorenz Y	35,72%	85,13%	53,11%	92%	8,55%	14,77%
Lorenz Z	16,36%	53,18%	79,55%	98,18%	9,09%	35,45%

Table 3.2.: Percentage of successful calculations of Lyapunov exponent under adherence of its relative (rel.) and absolute (abs.) measurement errors. For calculations the attractor was reconstructed with all combinations of reconstruction functions mentioned in table 3.1). Exception: Lorenz Z. This dataset was reconstructed with a smaller but similar set of reconstruction functions shown in table A.1

Using the absolute average error one gets a good agreement for the first two Lyapunov exponents L_1 and L_2, whereas the third Lyapunov exponent L_3 is calculated wrong in the majority of cases. This error in L_3 is due to the more problematic numerical estimation of negative Lyapunov-exponents und therefore an error produced by the Lyapunov-estimation and not by the reconstruction. The results using the relative average errors is more catchy. In this case around every second calculation was successful. Hence this result does not support the assumption that the reconstructions were successful but it also does not disprove it because the high amount of wrong results could also caused in the numerical estimation process itself. The whole result is also qualified by the used average errors. Table 3.3 shows the average of these averaged relative and absolute errors calculated by lyap_spec for different time series.

Comparing these errors with the reference Lyapunov exponents (table 3.4) one finds that most errors have a higher order than the reference value. Hence the results

used data	L_1 rel.	L_1 abs.	L_2 rel.	L_2 abs.	L_3 rel.	L_3 abs.
Rössler X	0.4146	2.1310	0.4153	1.8103	0.1630	0.5515
Lorenz X	0.0627	0.8195	0.0729	1.2808	0.0570	0.5629
Lorenz Y	0.1779	2.5903	0.1080	2.4563	0.1419	4.2703
Lorenz Z	0.0311	2.4752	0.0445	3.5781	0.0106	0.5398

Table 3.3.: Average of all with lyap_spec estimated average relative (rel.) and absolute (abs.) errors for the calculated Lyapunov exponents L_1, L_2 and L_3 for different used time series.

should not be overrated because such a high average error allows a whole bunch of results to be counted as a success.

	L_1	L_2	L_3	D_{KY}	ref.
Rössler	0.069 ± 0.003	-0.0002 ± 0.0002	-4.978 ± 0.002	2.01	[28],[25]
Lorenz	1.37	0.00	-22.37	2.06	[25]

Table 3.4.: Under use of the original equations numerical derived Lyapunov exponents and Kaplan-Yorke-dimensions. This values are used as references for all comparisons.

Another problem is that these errors are varying a lot between different calculations for different embeddings. That implies that a successful calculation of Lyapunov exponent could either mean that the result was real close to the reference value but it could also mean that the average error was just big enough so that the result was counted as success. At the same an unsuccessful calculation could mean a result far away from the reference value or just a too low estimated error. Hence the results have to be handled with care and more investigations are needed to get a more significant statement.

3.2.2. Chaotic behaviour mapped correctly?

Besides the quantitative results it is interesting to take a look at the qualitative results. One question is: Is the chaotic behaviour of our test-systems displayed correctly in the results. As explained in chapter 1.2.8 a 3D chaotic system has one positive, one vanishing and one negative lyapunov exponent (+,0,-). Table 3.5 shows the percentage of positive outputs for each lyapunov-exponent. Hence the perfect result for a 3-dimensional, chaotic system should have 100% positive values for L_1, 50% for L_2 and 0% for L_3. In the case of a hyperchaotic 4-dimensional system the result should be [100% | 100% | 50% | 0%].

	L_1	L_2	L_3	L_4
Rössler X	95,95%	22,23%	0%	-
Lorenz X	99,85%	18,18%	0%	-
Lorenz Y	99,9%	15,56%	0%	-
Lorenz Z	100%	23,18%	0%	-
Hyperrössler X_1	99,3%	85,31%	20%	0%

Table 3.5.: Percentage of positive valued Lyapunov-exponents L_1 to L_4 for different systems using the reconstruction-functions listed in table 3.1

Table 3.5 reproduces this behaviour adequately. For the vanishing exponent one finds some tendency to negative exponents but the relevant positive exponent can be found for at least every reconstruction. That means that the chaotic behaviour is visible in nearly every case. This can be seen as a good verification of the initial thesis.

3.2.3. Kaplan-Yorke-Dimension D_{KY}

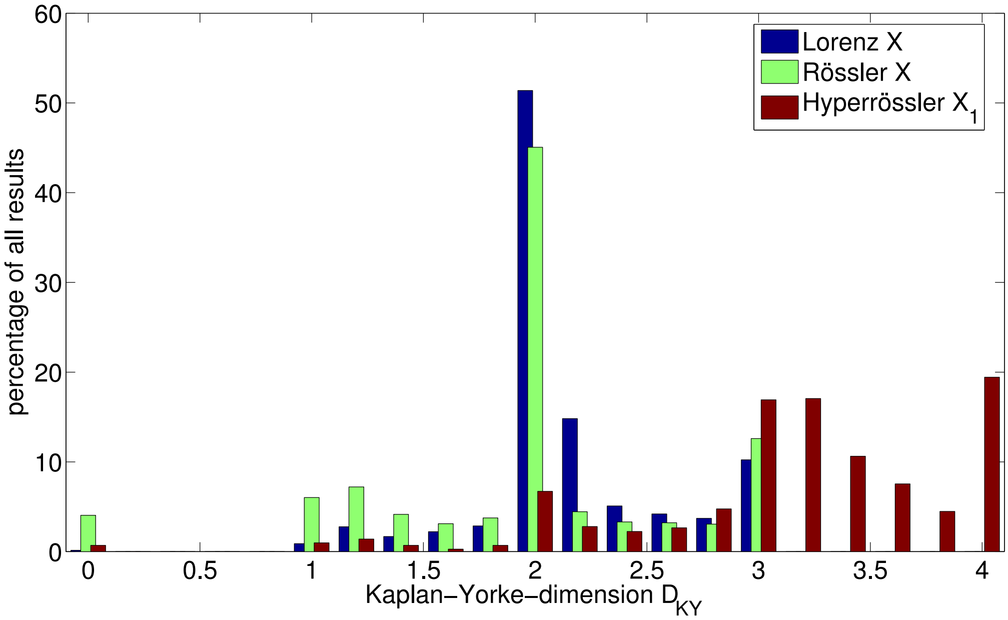


Figure 3.1.: Histogram showing the distribution of calculated Kaplan-Yorke-Dimensions D_{KY} for all reconstructions applied to the first components of Lorenz-, Rössler- and Hyperrössler-system.

To analyse the results of the Kaplan-Yorke-Dimension the histogram seemed to be the most useful way, because it delivers a graphical subsumption of all results. Figure 3.1 shows the distribution of the calculated Kaplan-Yorke-dimension for Lorenz-, Rössler- and Hyperrössler-system. As one can see one recieves a broad spectrum from $D_{KY} = 0$ to 4. Hence there are many combinations of reconstruction functions which deliver wrong results for the Kaplan-Yorke-Dimension. This can be explained either by a failed embedding process or by a calculation error within the dimension estimation. Special cases are the calculated dimensions 0 and 3 for Lorenz- and Rössler-system and 0 and 4 for the Hyperrössler-system. These values are the boundaries of the applied reconstructions. Hence a calculated dimension $D_{KY} = 0$ means that the calculation routine could not find any attractor whereas the upper limit means that the data points seemed to be equally distributed within the whole space.

Neglecting these obvious errors one receives another important result: The maxima of all three time series agree with the known dimensions of the used attractors. Rössler- and Lorenz-system have their maxima at $D_{KY} = 2$ with some tendency to higher values which corresponds to $D_{KY} = 2.01$ for the Rössler-system and $D_{KY} = 2.06$ for the Lorenz-system (see table 3.4). Even the higher dimension of the Lorenz-attractor is expressed in the graph: The results for the Lorenz-system showing a smaller decrease from 2 to the next higher step as the Rössler-system does. Therefore the Lorenz-system is tending a bit more to higher values than the Rössler-system. The Hyperrössler-system has its maximum somewhere between $D_{KY} = 3.0$ and $D_{KY} = 3.3$. I could not find any reference values in this case but the result also seems to be coherent.

Furthermore one can see that Rössler- and Hyperrössler-system have a broader spectrum compared to the Lorenz-system. Probably this can be explained by the qualities of the used time series. Chapter 2.3.4 showed already that the sampling rate of the Rössler-attractor was chosen a bit too high whereas the Lorenz attractor seemed to have an optimal sampling rate. At the same time the Hyperrössler-system probably had not enough data points because the number of necessary data points for good calculations is raising significantly with the dimension of the system. Anyhow all three time series showing their maxima in the histogram at expected values.

Another interesting ansatz is to take reconstructions using different components of the same system as input data. For this purpose figure 3.2 shows the distribution

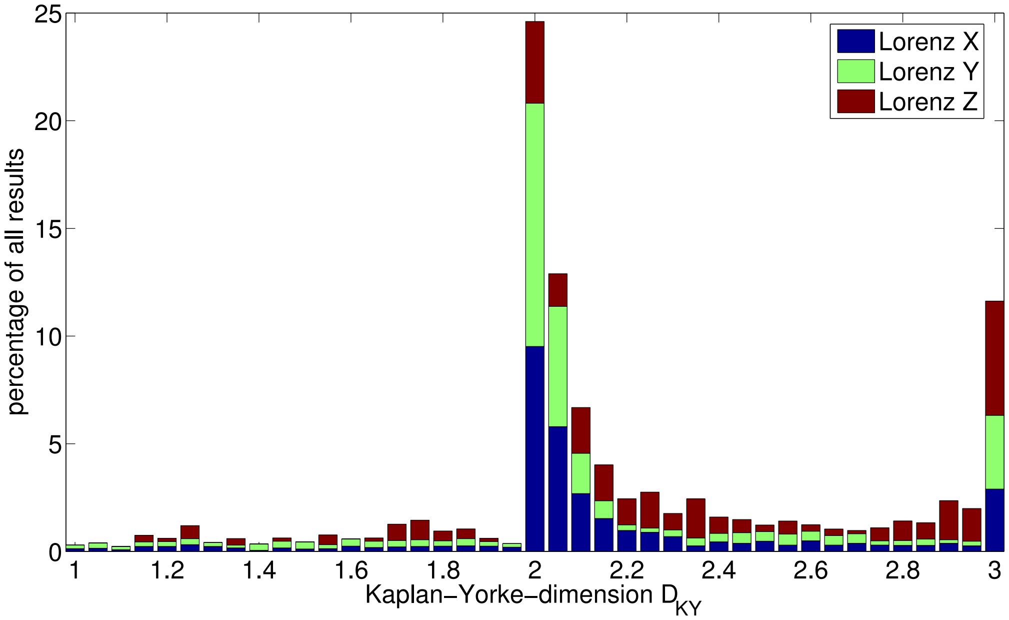


Figure 3.2.: Histogram showing the distribution of calculated Kaplan-Yorke-Dimensions D_{KY} for all reconstructions of all three components (x,y,z) of a Lorenz-attractor.

for all three components of a Lorenz system. Again the maximum can be found somewhere above $D_{KY} = 2$, but there are some differences concerning the used input component. Whereas x and y are showing nearly the same results the z component is producing a much broader spectrum. This can be explained by the fact that a reconstruction under use of the z component of a Lorenz system is problematic. All classical methods having strong problems to obtain a succesful embedding using the z component. Hence it make sense that also the dimension estimation fails more often. A more detailed discussion of the problems produced by the z component can be found in chapter 5.3.

Summarized the results of the Kaplan-Yorke-Dimension showed that not all used reconstructions were producing correct results. Hence it seems that not all used reconstructions were creating successful embeddings. Probably a part of the wrong results were produced by failed embeddings and another part by errors within the dimension estimation. Nevertheless there is also a big part which was delivering correct results. Therefore one can say that many of the investigated reconstructions were successful.

4. Applications

4.1. Noisy time series

As already mentioned in chapter 2.3.2 the reconstruction can also be interpreted as a filtering. Using classical methods this identity is not obvious. Analysing real measurement data with these methods normally two steps are necessary: First the data has to be filtered and after that it can be reconstructed. Using the results of the former chapters this means that a two step filtering is applied to the original data. One filter is used to reduce the noise and the other is used to get a proper reconstruction. But because both filters have different aims it could happen that the second one neutralizes the effect of the first one. Also two different filters are destroying more information of the original data than necessary. Using a reconstruction in the frequency domain these two steps become one. The number of calculations needed and the amount of data lost due to the reconstruction is reduced. Furthermore it offers the opportunity to produce a custom reconstruction-filtering for each special problem.

4.1.1. High frequent noise

The following example shows a Rössler-attractor pointwise combined with Gaussian noise ($\sigma^2 = 1, \mu = 0$). It presents the case of a system which measurement is disturbed by highfrequent noise.

Fig.4.1 displays the original noisy system. The Rössler-attractor is reconizable because of its characteristic structure but the topological characteristics are hidden by the noise. Especially the characteristic switching of the trajectories from outer to inner parts is not visible. It is possible to suppose an already known system behind this structure but it would be impossible to analyse a completely new system.

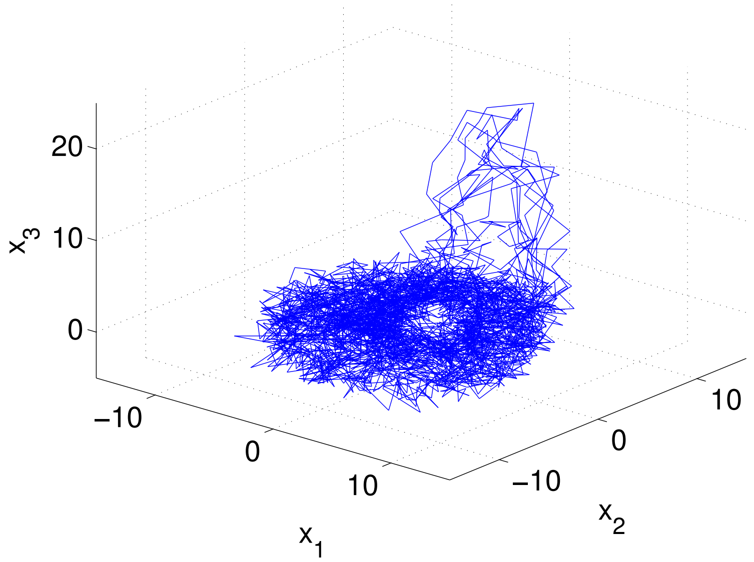


Figure 4.1.: Original Rössler-attractor overlayed with highfrequent gaussian noise ($\sigma^2 = 1, \mu = 0$)

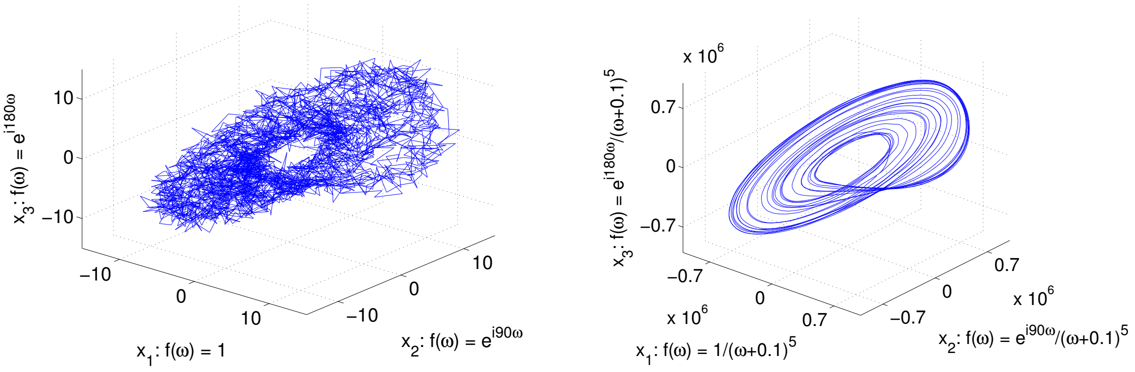
Fig.4.2 compares the case of a classical reconstruction using the time-delay method and an optimized reconstruction using the reconstruction functions $f(\omega)$ shown in equation 4.1. As expected time delay produces a proper reconstruction of the noisy dataset without rebuilding the topological properties of the original system. Using the optimized reconstruction functions (eq. 4.1) the result is a perfect image of the original Rössler system.

$$f_1(\omega) = \frac{1}{(\omega + 0.1)^5} \qquad f_2(\omega) = \frac{e^{i90\omega}}{(\omega + 0.1)^5} \qquad f_3(\omega) = \frac{e^{i180\omega}}{(\omega + 0.1)^5} \tag{4.1}$$

In this example the reconstruction functions are some kind of mixture between time-delay method and a high-frequent-noise-filter. There are many other possible choices for $f_i(\omega)$ that should lead to similar results.

4.1.2. Random Walk

Another useful example for noise is the Random Walk. Because of its dependency on its former values (shown in eq. 4.2) is it a good model for noise with a strong lowfrequent part.



(a) time-delay (FFT)

(b) optimized reconstruction

Figure 4.2.: Reconstruction of a noisy Rössler-system. Comparison between classical delay method (4.2(a)) and a customized reconstruction including noise-reduction (4.2(b)).

$$n(t+1) = n(t) + random(t) \tag{4.2}$$

For this example again the Rössler attractor was used and combined with a random walk. It demonstrates the case of real measurement data containing noise of various frequencies with focus on low frequent parts.

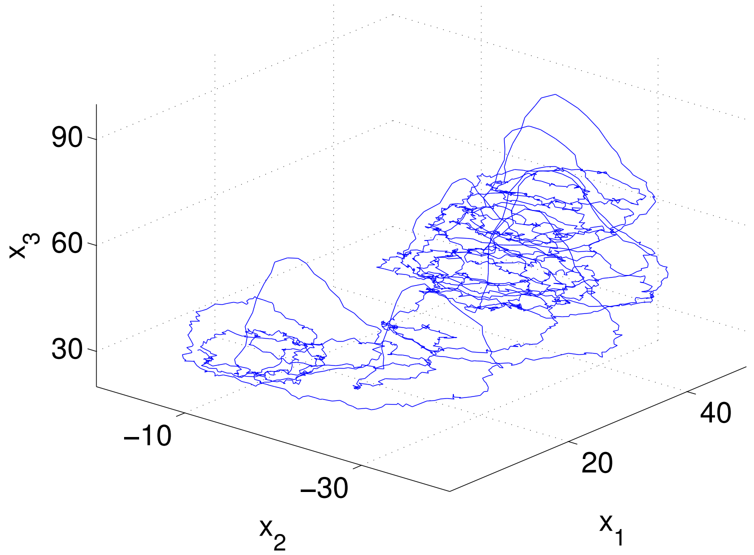


Figure 4.3.: Rössler attractor combined with random walk

Fig.4.3 shows the original system combined with a random walk. Even for the case that one expect a Rössler attractor as underlying dynamics it is hard to recognize it in the graph. Hence without noise reduction nearly no further investigations are possible. Also all classical methods would not deliver any further information without using another filtering before. Even reconstruction with derivation which damps lowfrequent parts would not lead to better results because every dimension would get another power of ω in $f(\omega)$. Every dimension would have another noiselevel and the reconstruction would be strongly inhomogeneous.

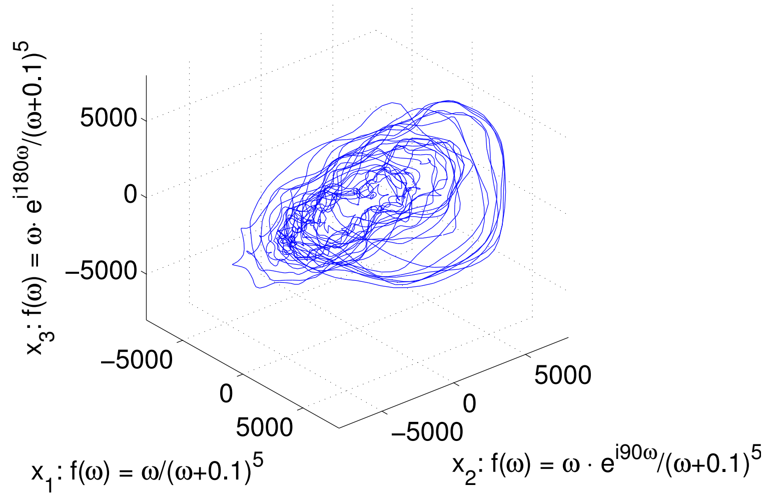


Figure 4.4.: Customized reconstruction of a Rössler system interfered with a random walk. Noise is reduced due to special choice of reconstruction-functions.

Fig.4.4 shows the result applying similar reconstruction functions $f_i(\omega)$ as used in the first example (chapter 4.1.1). To take into account the lowfrequent parts the numerator of each function is expanded with ω. The used reconstruction functions are shown in eq.4.3.

$$f_1(\omega) = \frac{\omega}{(\omega+0.1)^5} \qquad f_2(\omega) = \frac{\omega \cdot e^{i90\omega}}{(\omega+0.1)^5} \qquad f_3(\omega) = \frac{\omega \cdot e^{i180\omega}}{(\omega+0.1)^5} \tag{4.3}$$

Compared to the example with high frequent noise the result is less accurate but it still shows the basic features of the underlying attractor. Hence the Rössler system can be recognized. One should keep in mind that this choice of reconstruction-functions is only an example. There are probably reconstruction functions which are providing much better results for this case.

5. Further results

5.1. Some thoughts about universality

As shown all presented methods do fit in the propagated structure, but does that mean that all possible reconstructions have to do that? Obviously not. One example which does not meet with this structure was developed by Gilmore for symmetric systems [7]. He used the reconstruction:

$$x(t) \quad ; \quad y(t) = \frac{dx(t)}{dt} \quad ; \quad z(t) = y(t)\frac{dy(t)}{dt} \tag{5.1}$$

Whereas the first two components are equivalent to a derivation reconstruction the third term uses a multiplication of the classical derivation term $\frac{dy(t)}{dt}$ and the second component $y(t)$. Applying this set of functions to the Lorenz attractor (fig. 5.1) one recieves a successful embedding which displays all topological properties correctly. Especially the reinjection of trajectories coming from one ring into the other one is displayed correctly without any autocrossing.

Converting its z-variable into the frequency domain one gets a convolution of two components:

$$\tilde{z}(\omega) = (i\omega\tilde{x}(\omega)) * \left(-\omega^2\tilde{x}(\omega)\right) \tag{5.2}$$

Hence there is no possibility to express this reconstruction with a reconstruction-function $f(\omega)$. This means that this representation does not contain all possible reconstructions. But taking a closer look at this example one gets another interesting result: Let us assume that one can produce a new and independent reconstruction variable by multiplying two different reconstructions. So it should be also possible to use the following reconstruction:

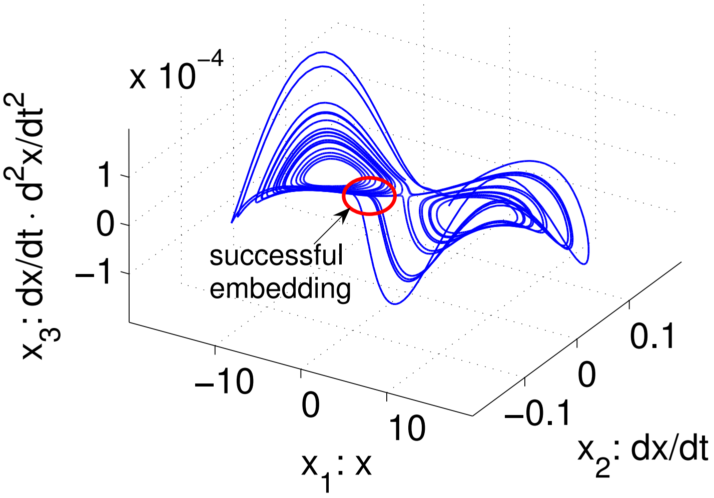


Figure 5.1.: Successful embedding of a Lorenzattractor using equation 5.1. The reinjection of trajectories (marked red) happens orthogonal to the ring-plane. Therefore no autocrossing occurs.

$$x(t) \quad ; \quad y(t) = \frac{dx(t)}{dt} \quad ; \quad z(t) = x(t)\frac{dx(t)}{dt} \tag{5.3}$$

This reconstruction has the same structure as equation 5.1 but uses a multiplication of $x(t)$ and its derivative as z-component instead of of $y(t)$ and its derivative. In fact this reconstruction does not work (fig. 5.2). It looks similar but produces only a two-dimensional reconstruction which is lying twisted in three-dimensional space. This leads to autocrossing at the reinjection places and other parts of the system. The quality of this embedding is equivalent to one using only the first two components, therefore the third component does not produce any new information. The reason is probably that $x(t)$ and $\frac{dx(t)}{dt}$ were already used as reconstructions (first and second component). In the first example (eq. 5.1) $y(t)$ was also already used but the derivative of $y(t)$ was new. Hence one does not get a new variable by multiplying two already used ones. It looks like at least one part has to be completely new and has to follow the structure $f(\omega)\tilde{x}(\omega)$ to produce a proper reconstruction.

So I guess that the propagated structure does not describe all possible reconstructions but it contains some kind of basic reconstruction elements which are necessary for proper embeddings. This is only an assumption because a mathematical proof is missing but I could not find any example so far which disagrees with this

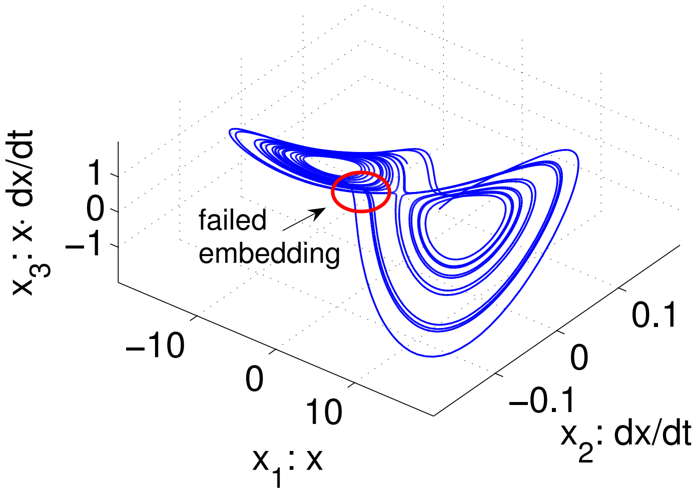


Figure 5.2.: Failed embedding of a Lorenzattractor using equation 5.3. The reinjected trajectories (marked red) lying in the ring-plane and crossing the trajectories in the plane. Hence one does not get a proper embedding.

statement.

5.2. Some thoughts about a proof and plausibility

One actual problem for reconstruction in the frequency domain is that the proof for time-delay cannot be expanded easily for the case of general reconstruction functions in the frequency domain. Takens time-delay proof bases on the idea that a time series can be written as $(h(x), h(g(x)), h(g^2(x)), ...)$. Here means $h(x)$ the measurement-function which projects the whole dynamic x on the measured dimension, $g(x)$ is a function mapping the original state to the next state of the dynamical system. Using this description a time-series can be understood as a series of different representations of x, hence x can be rebuilt combining enough of these different representations.

Therefore the proof bases on the idea to reconstruct the phase-space pointwise. In opposite to this ansatz a reconstruction in frequency space happens at once for the whole system. It is not possible to describe this kind of reconstruction point by point. Hence the proof has to look completely different.

So I could not find any proof for that reconstruction but there are some points that make the idea plausible: First of all some thoughts about the general structure. Only to use a function that can change phase and amplitude of a frequency but which cannot shift frequency peaks is very restrictively. It means that the general structure in frequency space has to be the same for each component of a dynamical system. That looks arguable because in the time-domain different components often have significantly different behaviour. Hence the idea is to look at original dynamical systems and compare its components in the frequency domain. Are all components of a dynamical system such similar in frequency space?

The first example is the Rössler attractor. Here especially the z-component behaves different to x and y (fig. 5.3). But looking at the frequency space (fig. 5.4) one gets the situation that all three components are showing exactly the same peaks.

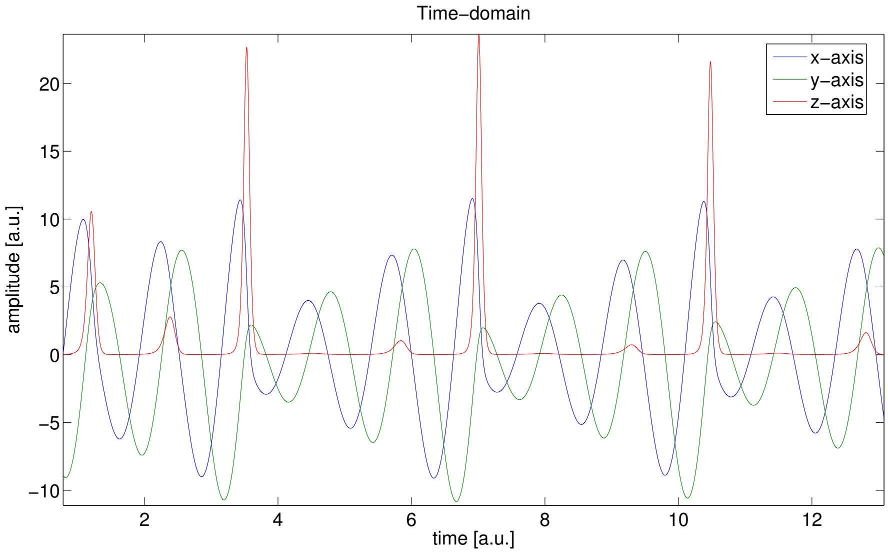


Figure 5.3.: Components of a Rössler attractor in time-domain (detail). X and Y exhibit similar characteristics whereas Z behaves considerably different.

Unfortunately not every example delivers such a favoured structure, e.g. looking at the Lorenz attractor one finds a z-component that differs completely from x and y in its frequencies (fig. 5.5).

This conflict can be solved by taking again a look at Gilmores reconstruction for symmetric systems [7] mentioned in chapter 5.1: The original Lorenz attractor has a symmetry relative to a line parallel to the z-axis (fig. 5.6(a)) whereas classical embedding delivers a system with a point as symmetric center (fig. 5.6(b)).

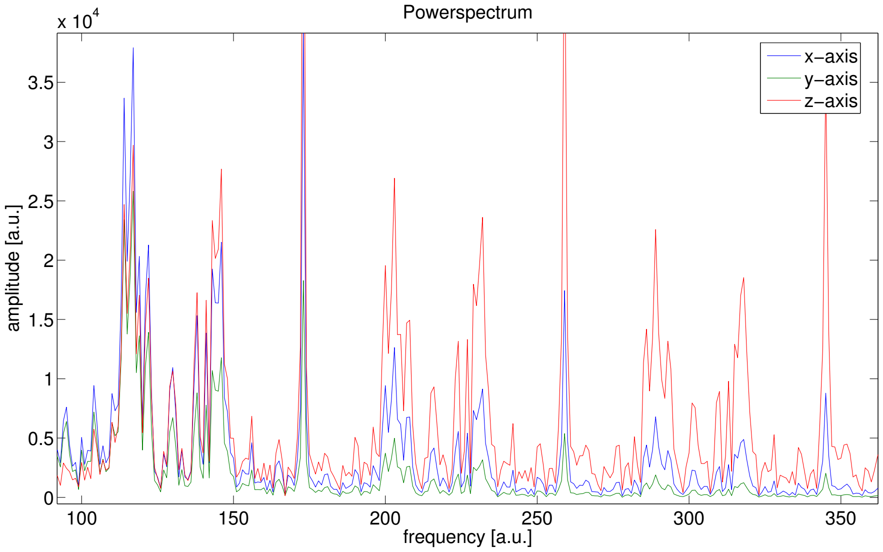


Figure 5.4.: Powerspectrum of a Rössler attractor (detail). In opposite to the presentation in time-space all components behave similar.

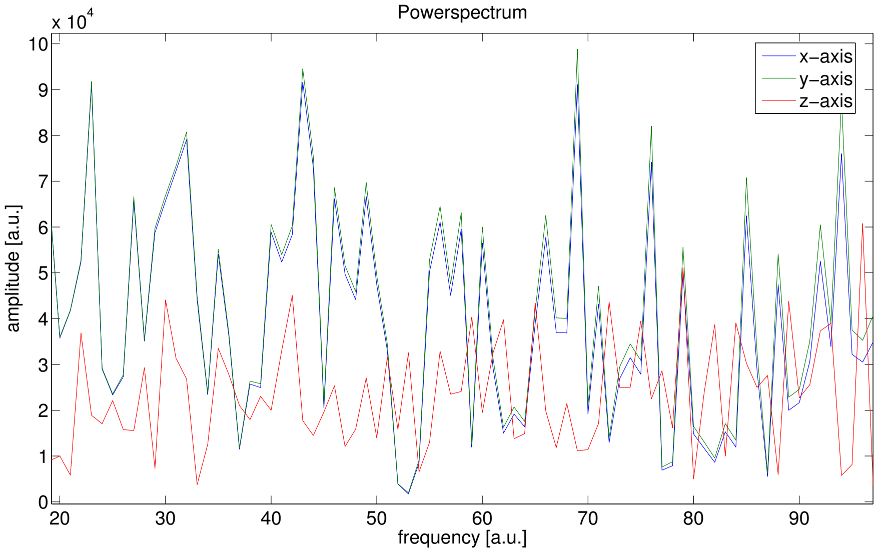
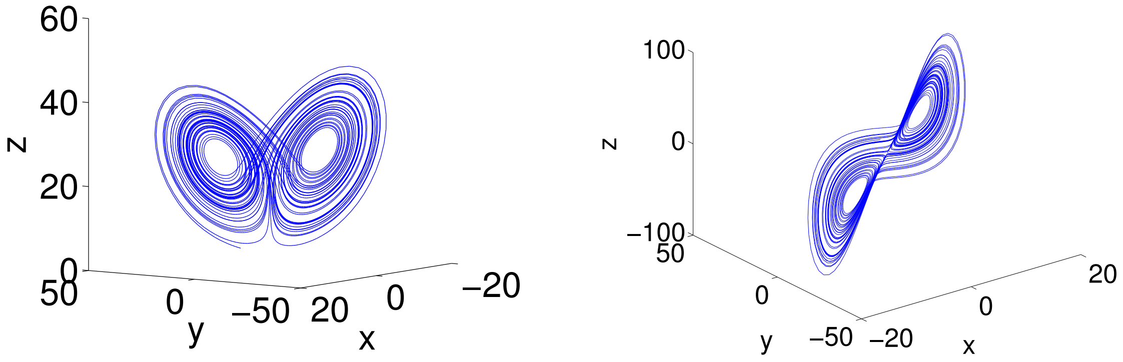


Figure 5.5.: Powerspectrum of a Lorenz attractor (detail). Z diverge in its behaviour from X and Y and exhibits different peaks.



(a) original

(b) symmetry of z-component modified

Figure 5.6.: Original Lorenz attractor unchanged (5.6(a)) and with modified z-axis $z(t) = z_{\text{old}}(t) \cdot y(t)$ to obtain another symmetry (5.6(b))

This difference can be understood easily by taking into account that a symmetry-axis along the z-axis means that x and y have a symmetry that differs from the symmetry z contains. At the same time a classical reconstruction delivers the same symmetric properties for every component. This leads to a dynamic which is symmetric to a point instead to a line.

Gilmore used a reconstruction which inserts that information about the symmetry artificial to get a picture that is closer to the original system. But this change does not affect the topological properties of the system.

One can use this idea to compare the original Lorenz attractor with its reconstructions. For this test I multiplied the z-component of the original system with its y-component to get a new z-axis (fig. 5.6 compares the original system with this new version). This new system derived from the original Lorenz attractor is now symmetric to the center and therefore has the same symmetric properties as all reconstructions of the Lorenz system which do not insert any changes in the symmetry.

Fig.5.7 shows the result in the frequency domain. Now also the modified z-component has the same peaks in its powerspectrum compared to x and y. Hence it supports the assumption that every dynamical system features at least one representation for which all components have same peaks in their powerspectra. This makes it more plausibel that the propagated class of reconstructions is working correctly.

Another argument are the classical reconstruction functions themselves: Besides

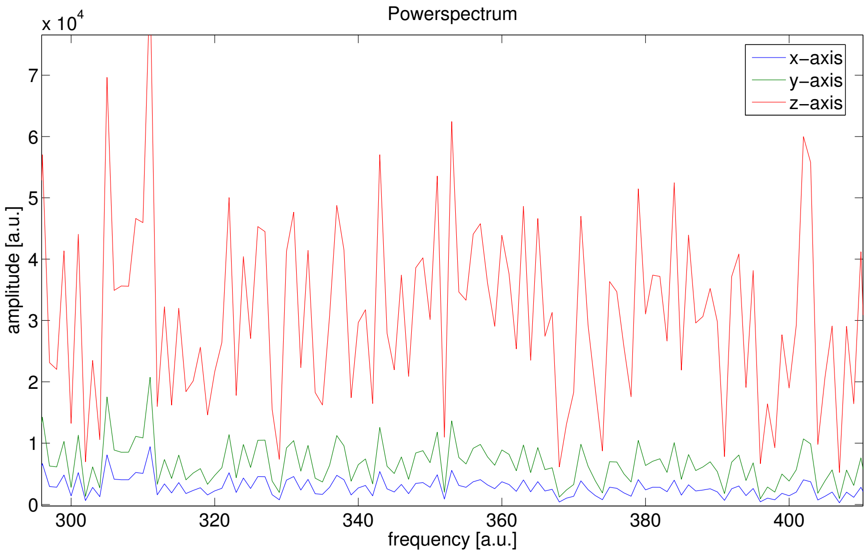


Figure 5.7.: Powerspectrum of a Lorenz attractor with $z = z_{\text{old}} \cdot y$ (detail). Now also Z exhibits the same characteristic peaks as X and Y.

the fact that all this methods fit in this propagated structure there is another interesting point. It is proven that time delay delivers a correct embedding in nearly every case (probability not to get an embedding is equal to zero [26]). This means by changing the direction of this proof that at least nearly every system has a representation for which all components have the same peaks in their porwerspectra. Under this circumstances it make sense to assume that a reconstruction that is only smoothly changing phases and amplitudes works.

5.3. Symmetries

It seems that symmetries are a good tool to understand the being of a reconstruction. Especially the difference in symmetric properties between original and reconstructed system shown in chapter 5.2 might give interesting hints. It looks like a reconstruction in frequency space is not able to modify the symmetry information of any component. Instead the symmetries of the used time series are conserved.

As a product one finds only reconstructions with the same symmetric behaviour for each component as e.g. $(x, y, z) \rightarrow (-x, -y, -z)$ for a reconstruction of a lorenz attractor using its x oder y-component as source. The assumption that reconstruc-

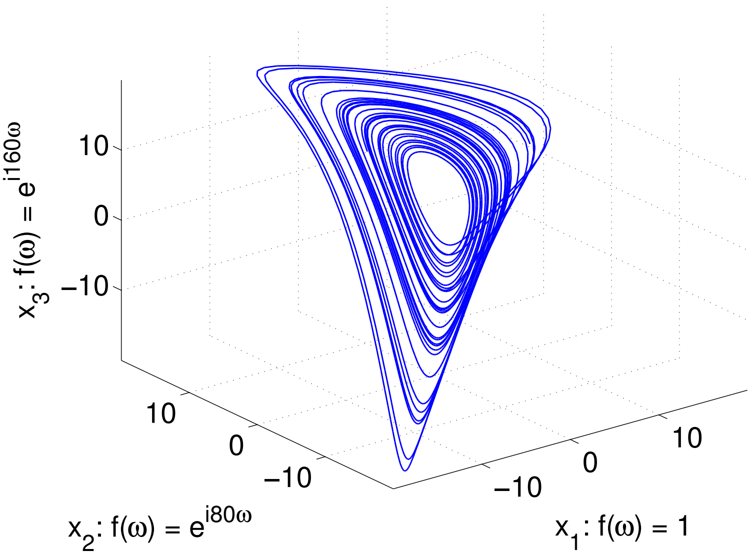


Figure 5.8.: Time-delay reconstruction using the z-component of a Lorenz system. Due to the symmetric properties of the z-component both characteristic rings are folded together.

tion cannot change the symmetries could also be a good explanation for the result of a reconstruction using the z-component of a Lorenz system (fig. 5.8): In this case one gets an attractor wich looks a bit like a broadened limit cycle. Instead of that two characteristic loops of a lorenz system the result contains only one loop. Under assumption of symmetry-conservation this is easy to explain: The original system has a symmetry of the structure $(x, y, z) \rightarrow (-x, -y, z)$. Taking a look at the z-components that means that booth loops lying exactly above each other. Using this dataset for reconstruction also the reconstructed x and y-component will have this property because of symmetry-conservation. This leads to a reconstruction where both loops become one. Hence it can be understood very well why a reconstruction using the z-component delivers such an unusual result.

Another interesting point is to investigate the consequences of a symmetry change. One could expect that a symmetry change does not change the power spectrum strongly. Especially looking at the example of a sinus, which has a point-symmetry to the origin, and cosine, which is axis-symmetric to the origin, one knows that in powerspectrum the difference is only a shift in phase. Hence one could expect that especially that kind of transformations we use for reconstruction (smooth change of phase and amplitude) could be intpertreted as changes in symmetry. In fact this assumption must be completely wrong!

5.4. DFT related implementation errors

As already mentioned all transformations are derived from the continuous Fourier Transformation and then used for the case of Discrete Fourier Transformation (DFT). This lead to some problems and errors. Most of them were already discussed but one problem concerning the implementation is still missing:

Using a time series containing $2n$ data points with $n \in \mathbb{N}$ one gets also $2n$ data points in Fourier space. But an even number of datapoints means that the dataset for $\omega = 0$ cannot be the centre of all datapoints and the symmetry one normally expects for the Fourier space is ruined. Instead of datapoints for $(-\omega_{max}...0...\omega_{max})$ one gets $((-\omega max + \Delta\omega)...0...\omega_{max})$. Hence ω_{max} has no negative counterpart.

In the discrete case this fact does not produce any problems because the DFT is projecting the time series on the unit circle and the maximal frequency is $\omega = \pi$. Because of the symmetry at the circle a rotation with π is equivalent to a rotation with $-\pi$. Hence the lonely data point presents $\omega = \pi$ and $\omega = -\pi$ at the same time. That means also that for real-valued time series this data point is always real-valued $(x(\omega) = \bar{x}(-\omega))$.

Applying methods of the continuous Fourier Transformation for such a DFT-transformed time series leads to errors because in the continuous case only the value for $\omega = 0$ has to be real-valued to obtain a real-valued time series. Hence most methods derived from the continuous case do produce complex values for the last data point of the series. Because of the missing symmetry in the last data point this produces a complex time series after applying the inverse DFT.

To solve this problem I restricted the incoming dataset to have an odd length. This restores the required symmetry in frequency space and thereby avoids the mentioned error.

5.5. Optimal embedding

One of the biggest advantages delivered by this generalization of the reconstruction process simultaneously produces one of the biggest problems of it: An endless amount of possible reconstructions. Even for Time-delay itself existing many different methods and suggestions to calculate the optimal time-delays to get the best embedding. Taking the problem now to this new, more generalized description it gets even more complicated.

There are two different approaches two deal with this topic: The passive way measuring the quality of an embedding and the active way providing optimal functions for embedding. The passive method has the advantage that the user maintains the freedom of choice for the embedding functions. On the other hand this freedom has the disadvantage that this measure is only giving some information after the embedding was realized. The active method allows to perform an embedding without thinking about the reconstruction functions. At the same time this prohibits to implement special requirements into the embedding process.

Anyhow I made some approaches for both ways but they did not produce satisfying results:

5.5.1. Measure for an optimal embedding

The primary ansatz for finding a good measure was to take a look at the powerspectrum. As requirements for a good embedding I found that all reconstructed dimensions must be linearly independent and that the reconstruction functions should be slowly varying compared to the time series (chapter 2.3.2). Chapter 5.2 shows that a slowly varying reconstruction function produces a signal which still containes the same peaks as the original dataset. Using a reconstruction function that is varying too fast can remove or produce peaks in the data. This led to the idea that a good embedding should conserve peaks in the powerspectrum. But using only a measure for the peak conservation cannot be enough because in this case the reconstruction function $f(\omega) = 1$ would be the best choice. The other requirement one has to deal with is the linearly independency of all components. So a good embedding should conserve peaks AND should produce components that are as indepent as possible.

For this purpose I invented the following measure $m(a, b)$:

$$m(a, b) = \text{powerdiffcorr}(a, b) + 1 - \text{abs}(\text{corr}(a, b)) \tag{5.4}$$

a and b are the power spectra of two components that should be compared. "abs" is the absolute value function and "corr" the correlation function. The function "powerdiffcorr" calculates the similarity between two time series. The aim was to compare only the positions of peaks and not to consider its amplitudes. For this purpose only the sign of the derivative of the powerspectrum was used:

$$\text{powerdiffcorr}(a,b) = \text{corr}\left(\text{sign}(\frac{da}{d\omega}), \text{sign}(\frac{db}{d\omega})\right) \tag{5.5}$$

So two signals with exactly the same peaks produce $\text{powerdiffcorr}(a,b) = 1$ as result and two exactly opposite signals result in $\text{powerdiffcorr}(a,b) = -1$. Applied to the measure of embedding quality m an optimal embedding should have $\text{powerdiffcorr}(a,b) = 1$ and $\text{abs}(\text{corr}(a,b)) = 0$. So the best combination of two components should result in $m = 2$ whereas every other combination produces a result $m < 2$. Hence the best combination should have the highest m-value. For the embedding with more than two dimensions one could just sum up the results of m for each possible combination and again the highest value should present the best embedding.

Unfortunately in pratice this measure does not deliver good results. In fact it is a good algorithm to determin wrong embeddings. For example applied on the 24 reconstruction functions used for the numerical verification in chapter 3 (table 3.1) and a Rössler system a failed embedding was choosen as optimal case (the system was blown up in one direction with some highfrequent disturbances). Also in most other tests especially failed embeddings were evaluated extremely positive. Mostly the choosen embedding had the characteristic that the system was blown up in some directions which was happening because of foldings within the system or because of high-frequent disturbances. So obviously the made assumptions were wrong.

5.5.2. Constructing an optimal embedding

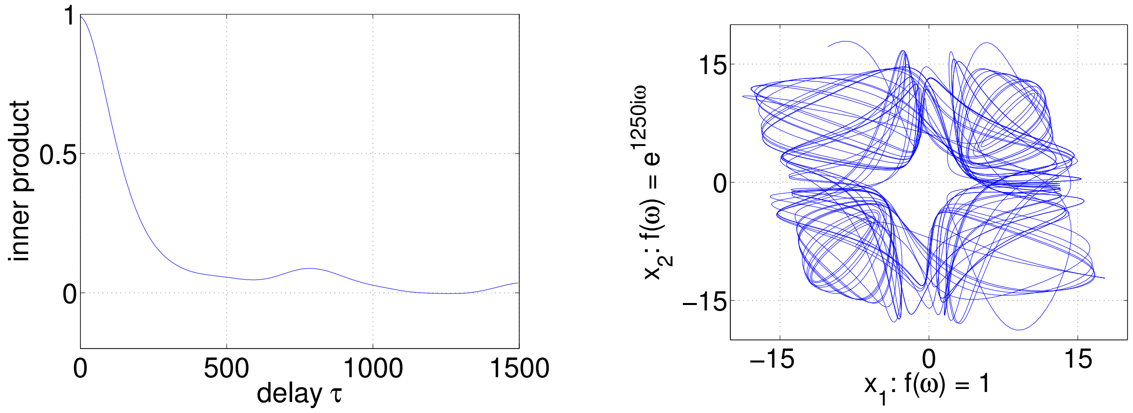
For the construction of an optimal embedding the initial idea was to use orthogonal reconstruction functions. This makes sense because it is the classical way to span a vector space. But which inner product should be used? Because one handles with reconstruction functions in frequency space also the scalar product should be defined in this space. So the first ansatz was to use the integral over all frequencies:

$$S_1(f_1(\omega), f_2(\omega)) = \int_{-\pi}^{\pi} \overline{f_1(\omega)} \cdot f_2(\omega) d\omega \tag{5.6}$$

This had the big disadvantage that most dynamical systems do not have frequencies over the full spectrum. Often only a quarter of the frequency spectrum shows significant signals whereas the rest is vanishing. So it does not play any role how the reconstruction function behave in parts where the power spectrum $p(\omega)$ vanishes. For this reason I introduced the square root of the power spectrum as a weighting function to take into account the influence of the time series itself. As a second step also the integration limits of the scalar product could be reduced. Under use of the symmetry between positive and negative frequencies it is enough to integrate from 0 to π:

$$S_2(f_1(\omega), f_2(\omega)) = \int_{0}^{\pi} \sqrt{p(\omega)} \left(\overline{f_1(\omega)} \cdot f_2(\omega) + f_1(\omega) \cdot \overline{f_2(\omega)} \right) d\omega \tag{5.7}$$

One first result using the scalar product S_2 is that a Hilbert transformation delivers an optimal two-dimensional embedding: Independently of the used time series the integral will vanish. This effect is also observed for every combination of one only real-valued and one only imaginary-valued function. In most other cases the underlying time series plays a role in the choice of the optimal embedding. This also agrees with time-delay, where the choice of delay depend on the time series. Unfortunately this inner product is not able to predict the best delay for a time-delay embedding. Calculating the scalar product between $f_1(\omega) = 1$ and $f_2(\omega) = exp(i\omega\tau)$ one gets the case of orthogonal functions for a delay much higher than the delay normally used. Figure 5.9(a) shows the relation of delay τ and calculated inner product for a Lorenz attractor. The optimal embedding delay for the first component is lying somewhere between 50 and 200. In this area one finds a linear decreasing of the inner product. The relevant area of a vanishing scalar product is reached for delays greater 1000. Using such a big delay does not offer a proper embedding as figure 5.9(b) shows. Hence the choosen inner product does not deliver a measure for an optimal embedding. Possibly another scalar product could provide better results.



(a) Inner product of $f_1(\omega) = 1$ and $f_2(\omega) = exp(i\omega\tau)$ for different delays τ. Orthogonality is reached for $\tau \approx 1250$

(b) Time-delay embedding using a delay of $\tau = 1250$. The graph shows strong foldings. The underlying attractor cannot be identified.

Figure 5.9.: Embedding of a Lorenz-attractor using orthogonal time-delay functions

5.5.3. Comment

As shown both methods (active and passive) do not work so far. Nevertheless I expect that there are solutions for both cases that should work. My approach for a measure bases on many different assumptions, hence it is not such a big surprise that it does not work. In opposite the ansatz using orthogonal functions seems more fundamental to me and I expect that there should be a solution similar to the presented approach.

Anyhow compared to the situation for time-delay the problematic becomes much more complex but in my opinion also less important. For time-delay it is essential to find a good choice for τ. Using a too small delay both components remain too similar. Using a too big delay the signal is varying too much and the the original data mutates significantly: As a result one receives foldings in the phase space. This problematic is caused in the special structure of time delay where τ is some kind of speed-parameter for the mutation of the original dataset. For most other reconstruction functions as e.g. derivation this problematic does not play any role because they have a more stable structure in frequency space. They do not modify the time series too much but also not too less. This behaviour leads to the situation that using ordinary functions the reconstruction mostly will work. This means that it is still an important aim to find measures and methods for an optimal embedding but it is not as important as one could expect. A good embedding under use of generic functions is more common than a bad one. Hence it is more problematic to mess up an embedding than to achieve it.

5.6. Z-transform

So far all considerations based on Fourier transformation. This special choice was done because it offers simple representations of all used reconstructions and it is easy to compute by using the Fast-Fourier-Transformation (FFT). However it is also possible to use the more general Z-transformation. In some cases the representations in z-space become more complex but the basic process remain unchanged.

$$\tilde{x}(\omega_k) = \frac{1}{N}\sum_{t=0}^{N-1} x(t)e^{-i\omega_k t} \quad \Rightarrow \quad \hat{x}(z) = \frac{1}{N}\sum_{t=0}^{N-1} x(t)z^{-t} \tag{5.8}$$

Using the Z-transformation $\hat{x}(z)$ the term $e^{i\omega_k}$ from the Fourier transformation $\tilde{x}(\omega_k)$ is replaced by a generalized complex number z (eq 5.8). Hence the frequency depending reconstruction functions $f_n(\omega)$ have to be replaced by z-depending reconstruction functions $f_n(z)$. Everything else is done analog to the limited case of the Fourier-transform.

5.7. Filter theory

An interesting ansatz is to interpret a reconstruction as some kind of filtering. Thus it is possible to apply results of the filter theory on the reconstruction:

The counterpart of the reconstruction function $f(\omega)$ in filter theory is called "transfer function". The transfer is the fourier transform of the used filter in time domain. Filter are divided in different classes. The most important separations are between causal and non causal filters and between finite-impulse-response-filter (FIR) and infinite-impulse-response-filter (IIR).

Causal filters have the property that their result only rely on the past of a signal and not on its future. Hence they can be used in real time. Examples are the reconstruction functions for time-delay, derivation and integration. In opposite non causal filter using also the future. Hence they can only be applied on a full time series. Hilbert transformation is an example for this behaviour.

Applied on reconstruction theory this distinction does not play such a big role. Nevertheless it highlights one fundamental difference between classical and new approach: In the classical case the reconstruction is applied pointwise on the input. Hence it is self-evident to use causal filters. This is more common because this allows to transform a reconstruction in real-time. Anyhow it is also possible to use non-causal filters, but this enforces a more complex reconstruction process. Working in frequency space the reconstruction is done for the whole system at once. Hence it does not play any role whether the reconstruction function is causal or not. Summarized one can say that both kinds of filters (causal and non-causal) can be applied in both reconstruction approaches. But only for the classical case the difference between causal and non-causal reconstruction becomes apparent. The reason is that the classical approach has to make a difference in appliance whereas the reconstruction process in frequency space is not affected by the filter-causality.

More relevant to reconstruction theory becomes the distinction between FIR and IIR filters. FIR filter producing -as mentioned by its name- an output that has a finite length. This is because their output only depends on a limited range of points around the actual position. Hence the output has to vanish if all inputs within the range of the filter vanish. In literature one finds often a causal representation of this behaviour for which only the N-th last point affect the actual output but the definition can be easily extended for the non-causal case:

$$y(n) = \sum_{i=-N}^{N} a_i x(n-i) \tag{5.9}$$

This structure is a convolution of the original time series $x(n)$ and the used filter. Therefore one receives a multiplication between time series and filter applying a Z-transformation:

$$Y(z) = \sum_{i=-N}^{N} a_i z^{-i} X(z) \quad \Rightarrow \quad f(z) = \sum_{i=-N}^{N} a_i z^{-i} \tag{5.10}$$

Equation 5.10 shows the structure for a FIR reconstruction function in z-space. It has to be remarked that this representation already uses the discretization in time-domain whereas many representations of reconstruction functions shown before intentionally neglect this fact. Hence with this representation it is not obvious that $f(\omega) = i\omega$ is a FIR-filter. It becomes obvious taking again a look at the discrete case of this derivation: $f(\omega) = 1 - e^{-i\omega}$. In z-space one gets $f(z) = z^0 - z^{-1}$.

For the case of IIR-filter the crucial difference is its recursive depence on the filter output. Every new output depends on at least one other output of the IIR-filter.

$$y(n) = \sum_{i=-N}^{N} a_i x(n-i) - \sum_{j=1}^{M} b_j y(n-j) \quad \Rightarrow \quad \sum_{j=0}^{M} b_j y(n-j) = \sum_{i=-N}^{N} a_i x(n-i) \tag{5.11}$$

Equation 5.11 shows this structure of an IIR filter. Using $b_0 = 1$ one can express it as a relation between a limited number of outputs and inputs of the filter. Again this representation can be extended to the more general case which also includes the future of the filter and not only its past. One gets:

$$\sum_{j=-M}^{M} b_j z^{-j} Y(z) = \sum_{i=-N}^{N} a_i z^{-i} X(z) \quad \Rightarrow \quad f(z) = \frac{\sum_{i=-N}^{N} a_i z^{-i}}{\sum_{j=-M}^{M} b_j z^{-j}} \tag{5.12}$$

Applying the z-transformation one receives a reconstruction function similar to FIR but divided by another sum. Hence every reconstruction function with a z-depending denominator is an IIR-filter whereas a constant denominator agrees with a FIR-filter.

The next step is to take a closer look at differences in the output generated by these two kinds of filters: Because of the recursive relation of IIR-filters every input can affect the actual output. Hence it is theoretically possible that the output does not vanish even if the last non-vanishing signal occurred a long time before. So a FIR can only see a limited part of the whole time series whereas an IIR filter can see the whole series. For reconstruction this difference becomes important because it generates in the case of IIR-filters some serious problems: Because of its unlimited memory it can build up effects within the signal. This allows IIR-filter to add own dynamics to the system. In the worst case the dynamical system can get a whole new dimension produced by the filter. This explains e.g. the wrong embedding for the unfiltered integration reconstruction (fig.2.5 in chapter 2.2.3). There one can see that the whole system is inflated. But at the same time the reconstruction with an applied offset (fig.2.6, same chapter) shows that applying an IIR filter does not produce necessarily a new dimension. Anyhow FIR filter does not have the opportunity to paste additional dynamics into the system. Hence it is recommended to use FIR functions for reconstruction not to run the risk of changing the dynamics. If there is a strong reason to use an IIR filter instead of FIR this is still possible but one has to notice that this could lead to errors.

A good explanation and a more detailed discussion of this dependence was done by Badii et al. [2] [1] and Mitschke et al. [19]. All these publications assume that the time series is filtered first and then reconstructed using time-delay. Nevertheless all results can also be applied describing the reconstruction itself as filtering.

6. Conclusion and Outlook

As shown Phase Space Reconstruction is a powerful tool for analysing nonlinear data. It can be used for calculating Lyapunov-exponents and detecting chaos. It helps to understand complex dynamics and their behaviour. And it can reproduce datasets which were not measured **(chapter 1)**.

There are many different methods which produce correct reconstructions as time-delay, Hilbert-transformation, derivation and integration **(chapter 1.4)**. The most used one is time-delay but all methods have special properties which are useful in different situations. Hence every reconstruction method has some situations where it is the best choice.

Looking at all these different methods the questions were: Why can all these different looking methods be used for the same purpose? Is there any connection between all these functions? The answer is found in the frequency domain **(chapter 2)**: Performing a Fourier transformation all these methods getting a similar shape:

$$\tilde{x}_n(\omega) = f_n(\omega)\tilde{x}(\omega) \tag{6.1}$$

Every presented reconstruction method can be described as a multiplication in the frequency domain with a frequency-depending reconstruction function $f_n(\omega)$. This structure is also known as a filter. From this point of view every reconstructed dimension can be seen as a filtered version of the measured time series. It contains the original data but applies just a new focus: Some parts are amplified and other parts are reduced.

Continuing this investigation I showed that obviously not every function $f(\omega)$ can be used for reconstruction (e.g. $f(\omega) = random$ does not work). Hence there have to be some restrictions. I found these **(chapter 2.3.2)**:

- **real-valued output**

  To obtain a real-valued output in time-domain the reconstruction functions need to have this symmetry. Otherwise a real-valued time series together with a reconstruction function would produce a complex output in phase space:

$$f_n(\omega) = \overline{f_n(-\omega)} \tag{6.2}$$

- **linear independency**

  In phase space all components need to be linearly independent. Otherwise some dimensions are redundant and could be neglected. The embedding would have a lower embedding dimension than the number of components denotes. Because of the linearity of Fourier transformation this requirement leads directly to the linear independency of reconstruction functions.

$$f_1(\omega)...f_n(\omega) \text{ linearly independent} \tag{6.3}$$

- **smoothness**

  The reconstruction functions must be smooth in a way. Otherwise the reconstruction functions would mutate the original dataset too strong and too much information of the original system would get lost. So far an explicit mathematical formulation is missing. Probably it is useful to postulate continuous and slowly varying reconstruction functions. $f_n(\omega)$.

$$f_n(\omega) \text{ smooth} \tag{6.4}$$

Under consideration of these restrictions one got now a whole bunch of new possible reconstructions. This gives the opportunity to build reconstructions which are fitting best for the special situation it is made for. So it is now possible to reduce noise within the reconstruction process itself **(chapter 4)** instead of doing that in two steps (which can also interfere with each other in a negative way). And one can set now the focus on chosen parts of the frequency spectrum or use some advantages of already known reconstructions while suppressing other negative properties of it. For example is is possible to create a reconstruction with a similar structure as derivation but without amplifying high-frequent parts in such a strong way **(chapter 2.3.4)**.

Because I was not able to find a mathematical proof for that method so far I tried to justify it heuristically **(chapter 3)**. This was only achieved in a limited way. The idea was to calculate properties of a system which are theoretically invariant concerning the used embedding. Practically all calculation methods for time series are only approximations. Hence they were not independent of the used embedding. Normally this effect is neglected by choosing several calculation parameters under adherence of the used embedding. In contrast it does not make sense to change the calculation parameters for different embeddings if one want to compare them. Hence the calculations had to be treated in an unusual way and therefore the results were not very clear. In our special case the Lyapunov-spectrum was calculated for different dynamical systems reconstructed with every possible combination of 24 different reconstruction functions. The estimated Lyapunov-exponents were differing in many cases of the exact value. At the same time the chaotic behaviour of the used systems was reflected correctly in nearly every case.

The analysis of the Kaplan-Yorke-Dimension **(chapter 3.2.3)** was giving some kind of subsumption: It showed that many calculations were providing wrong results but that at the same time the maxima of all calculated dimensions in the histogram were giving the correct values. Hence it seemed that not all reconstructions were delivering correct embeddings but many of them.

Anyhow only a mathematical proof seems to be able to clarify this relation. For this reason I gave some thoughts to the characteristics of a reconstruction **(chapter 5)**. Interesting results were that there are reconstructions that do not fit in the propagated structure but that probably all of them are basing on it **(chapter 5.1)**.

Concerning the reconstruction process there is an important difference between classical methods and their counterpart in frequency space: The classical reconstruction happens pointwise whereas the new methods reconstruct the whole dynamic at once. This difference will probably play an important role for a proof **(chapter 5.2)**.

Another interesting point is that the propagated structure is strongly limited. It only allows to change phase and amplitude in frequency space smoothly. That means that the power spectra of all dimensions will have its peaks at the same positions. To investigate this fact I took a look at some original dynamical systems to check if they also have the same peaks in frequency space in all dimensions. The result was that this is not true. But it seems that every component can be transformed to a structure where the peaks are lying at the same positions using a symmetry transformation. I suppose that the symmetry properties of the original dataset are conserved under reconstruction transformation (or in other words: the symmetry properties of the full dynamical system are lost). This would also explain why the reconstruction of a Lorenz system always has a different symmetry compared to the original data **(chapter 5.3)**.

Having an endless number of possible reconstruction functions it is essential to know which functions should be used. For this purpose it is important to think about an optimal embedding **(chapter 5.5)**. I tested two different approaches to find an optimal embedding: Defining a measure for an optimal embedding and building up an orthogonal system of reconstruction functions. Unfortunately both approaches were not very successful so far. But it is still an important topic and should be investigated further.

Also interesting and already mentioned is the description of reconstruction as filtering. This allows to apply the whole filter theory on reconstruction theory **(chapter 5.7)**.

One of my biggest problems concerning this topic was its wide range. The possible fields of investigation are endless. One big part of it is the problem to find the optimal embedding. Using only the classical methods the number of possible reconstruction functions was strongly limited. Anyhow there is an uncountable amount of publications dealing only with the problem to find the optimal embedding functions. Now in frequency space the number of possible functions is endless. Hence also the problem of finding the optimal embedding gets much more complex.

In my opinion this generalization of reconstruction methods has a great potential for future applications. Classical methods working well in most cases but they are not as flexible as the generalized method is. Especially when it is not clear which reconstruction delivers the best result reconstruction in frequency space becomes interesting: One can test different reconstructions much faster because the source code can remain the same. Also one has more opportunities. This is not such a big advantage because everything can be done also classically but one can handle everything faster. For most applications the more complex numerical calculations should not play any role because normally the choice of reconstruction costs more time than the reconstruction itself. Hence for most applications this new method should be a bit better than the classical ones because of its better usability. More visible becomes the potential of this new method applying it in very special situations where explicit requirements are assessed for the reconstruction. In this cases it is necessary to have a method which can be fully customized.

Beyond this great opportunities there are also some problematic points that have to be marked: Without a mathematical proof it is critically to apply this method and to interpret its results. One can only assume that one got a proper embedding and there are strong hints that are supporting this assumption but one cannot be sure. Even with time-delay one has the problem that an embedding could also fail but this has another quality because there it is known that the probability for this case is vanishing.

Another problem of the missing proof is that there must be some requirements on the reconstruction functions $f_n(\omega)$ we do not know at the moment. One point is the exact mathematical formulation for "smooth" one has to use. At the same time there are some combinations of reconstruction functions which are fulfilling all requests made in chapter 2.3.2 but do not produce a proper reconstruction. One example is the Hilbert-transformation which works for a Rössler system but which does not work for a Lorenz attractor. The reconstruction function $f(\omega) = i$ seems to produce in this case some errors: The attractor is dispersing under this reconstruction. Hence there must be more rules which are not clear at the moment. Probably a solution can be found applying results of the filter theory, but at the moment this further requirements are not known. Hence it is still problematic to be sure to have a proper embedding.

Summed up this generalized method has a great potential but it still needs some investigations to be usable in an optimal way. The next steps that should be most important are:

- **Finding a mathematical proof:** This is probably the most important step. A proof would set this method on a strong base. At the same time it would help to understand the process much better and one should obtain the requirements on $f_n(\omega)$ which are actually missing.

- **Detecting optimal embedding functions:** For optimal application one needs strong mechanisms to detect the optimal embedding. Otherwise it is problematic to make a choice out of this endless set of functions. Actually it is still possible to orientate oneself at the classical methods and most generic functions deliver proper results but to use the full potential of the generalized method an optimal-embedding-detection is essential.

Furthermore there are many other topics that could be interesting. Some of them are already mentioned under further results (chapter 5). The interpretation of reconstruction as some kind of filtering could produce interesting results but also other topics have potential for great findings.

Acknowledgments

I thank Bernd Blasius, Frank Spahn and his whole science group for support. Special thanks go also to André Bergner, Eckehard Olbrich and Udo Schwarz for useful discussions.

A. Numerical verification - data

	$f(\omega)$	L_1 rel.	L_1 abs.	L_2 rel.	L_2 abs.	L_3 rel.	L_3 abs.
1	1	35,97%	55,34%	66,8%	85,38%	20,16%	24,9%
2	i	12,65%	37,94%	59,68%	94,47%	3,56%	7,11%
3	ω	7,91%	65,61%	56,92%	81,42%	1,98%	4,74%
4	$i \cdot \omega$	60,08%	77,86%	24,11%	40,71%	0,4%	1,58%
5	ω^2	55,73%	78,26%	29,25%	52,96%	0,4%	1,58%
6	$i \cdot \omega^2$	17,79%	60,47%	77,47%	98,02%	1,58%	3,95%
7	$\sqrt{\omega}$	50,59%	75,49%	56,92%	79,05%	15,81%	24,11%
8	$i \cdot \sqrt{\omega}$	27,27%	75,49%	21,34%	49,8%	0%	0%
9	$\arctan(\omega)$	52,57%	79,45%	54,94%	77,08%	23,32%	28,85%
10	$i \cdot \arctan(\omega)$	39,13%	66,01%	45,45%	70,36%	12,25%	13,83%
11	$\log(\omega+1)$	50,59%	79,84%	57,71%	74,7%	22,13%	27,27%
12	$i \cdot \log(\omega+1)$	21,74%	59,29%	27,67%	53,75%	0,4%	1,58%
13	$\exp(i\omega)$	45,06%	67,59%	50,99%	74,7%	18,58%	23,72%
14	$i \cdot \exp(i\omega)$	30,04%	73,12%	49,01%	79,05%	1,98%	3,16%
15	$\exp(2i\omega)$	37,55%	61,26%	52,96%	72,73%	17,39%	22,13%
16	$i \cdot \exp(2i\omega)$	24,11%	61,26%	41,9%	62,85%	2,37%	4,74%
17	$1/(\omega+0.1)$	48,22%	81,03%	49,8%	69,57%	15,42%	20,55%
18	$i/(\omega+0.1)$	40,71%	72,73%	58,89%	80,63%	17,39%	23,72%
19	$1/(\omega^2+0.1)$	47,04%	77,08%	41,5%	65,22%	15,81%	21,34%
20	$i/(\omega^2+0.1)$	43,08%	75,89%	55,34%	77,08%	16,6%	20,95%
21	$\omega/(\omega+1)$	54,55%	80,24%	58,5%	73,91%	21,34%	26,48%
22	$i \cdot \omega/(\omega+1)$	36,36%	64,82%	42,29%	62,45%	11,86%	13,44%
23	$\omega^2/(\omega^2+1)$	47,43%	78,26%	53,36%	71,94%	19,76%	24,11%
24	$i \cdot \omega^2/(\omega^2+1)$	36,36%	66,4%	38,74%	64,43%	11,07%	13,04%

Table A.1.: (Rössler X) Table shows the percentage of reconstructions for which lyap_spec calculated Lyapunovexponents (L_1,L_2,L_3) which agree with results derived directly from the equations. As source was used the x-component of a Rössler-attractor. As accuracy was used the average relative error (columns marked with "rel.") and average absolute error ("abs.") estimated by lyap_spec itself.

	$f(\omega)$	L_1 rel.	L_1 abs.	L_2 rel.	L_2 abs.	L_3 rel.	L_3 abs.
1	1	39,13%	68,38%	68,38%	88,54%	15,42%	30,43%
2	i	29,64%	72,73%	78,26%	98,02%	4,74%	25,69%
3	ω	33,6%	81,03%	69,57%	95,65%	9,88%	30,04%
4	$i \cdot \omega$	56,92%	81,03%	24,9%	65,61%	0,4%	1,58%
5	ω^2	67,19%	85,77%	33,6%	79,45%	0%	1,58%
6	$i \cdot \omega^2$	6,72%	60,47%	71,94%	100%	0,4%	3,95%
7	$\sqrt{\omega}$	47,83%	87,75%	62,85%	91,7%	18,58%	34,78%
8	$i \cdot \sqrt{\omega}$	20,95%	84,19%	32,81%	86,96%	0%	0%
9	$\arctan(\omega)$	50,2%	83,79%	66,4%	91,3%	20,55%	35,18%
10	$i \cdot \arctan(\omega)$	34,78%	84,58%	56,92%	91,3%	1,19%	14,62%
11	$\log(\omega + 1)$	44,66%	82,61%	64,03%	90,51%	16,2%	35,18%
12	$i \cdot \log(\omega + 1)$	22,92%	78,26%	45,85%	83,79%	0%	0,4%
13	$\exp(i\omega)$	41,11%	86,56%	57,31%	91,3%	12,25%	22,13%
14	$i \cdot \exp(i\omega)$	32,81%	79,05%	58,89%	94,86%	3,56%	19,76%
15	$\exp(2i\omega)$	35,18%	83%	58,5%	92,09%	7,11%	24,11%
16	$i \cdot \exp(2i\omega)$	29,64%	75,89%	60,08%	94,47%	5,93%	25,3%
17	$1/(\omega + 0.1)$	64,43%	94,86%	72,33%	90,91%	19,76%	32,41%
18	$i/(\omega + 0.1)$	73,52%	97,63%	76,68%	95,26%	23,72%	46,25%
19	$1/(\omega^2 + 0.1)$	80,24%	97,63%	79,45%	93,68%	23,32%	47,43%
20	$i/(\omega^2 + 0.1)$	67,19%	97,23%	69,96%	93,28%	14,23%	34,78%
21	$\omega/(\omega + 1)$	48,22%	87,75%	62,45%	89,33%	20,55%	27,67%
22	$i \cdot \omega/(\omega + 1)$	31,62%	82,61%	52,96%	89,72%	1,19%	2,37%
23	$\omega^2/(\omega^2 + 1)$	47,83%	84,98%	64,43%	92,09%	22,53%	32,81%
24	$i \cdot \omega^2/(\omega^2 + 1)$	32,41%	87,35%	55,73%	91,3%	3,95%	8,7%

Table A.2.: (Lorenz X) Table shows the percentage of reconstructions for which lyap_spec calculated Lyapunovexponents (L_1,L_2,L_3) which agree with results derived directly from the equations. As source was used the x-component of a Lorenz-attractor. As accuracy was used the average relative error (columns marked with "rel.") and average absolute error ("abs.") estimated by lyap_spec itself.

	$f(\omega)$	L_1 rel.	L_1 abs.	L_2 rel.	L_2 abs.	L_3 rel.	L_3 abs.
1	1	28,46%	60,47%	60,87%	90,51%	16,6%	22,92%
2	i	20,16%	65,22%	60,87%	97,23%	0,79%	2,77%
3	ω	28,46%	83,4%	69,96%	97,23%	13,44%	26,09%
4	$i \cdot \omega$	64,82%	82,61%	27,67%	70,36%	0,4%	1,58%
5	ω^2	49,8%	85,38%	34,39%	84,58%	1,58%	1,58%
6	$i \cdot \omega^2$	9,88%	60,47%	84,19%	99,6%	1,98%	3,95%
7	$\sqrt{\omega}$	40,71%	86,17%	61,66%	90,51%	15,81%	26,88%
8	$i \cdot \sqrt{\omega}$	21,74%	91,7%	22,13%	93,28%	0%	0%
9	$\arctan(\omega)$	40,32%	86,56%	61,66%	92,09%	16,6%	27,67%
10	$i \cdot \arctan(\omega)$	26,88%	84,58%	43,48%	90,91%	0%	3,56%
11	$\log(\omega + 1)$	35,97%	86,96%	59,68%	92,09%	14,62%	24,9%
12	$i \cdot \log(\omega + 1)$	23,32%	84,58%	37,15%	90,51%	0%	0,79%
13	$\exp(i\omega)$	36,36%	90,51%	49,41%	93,68%	9,88%	19,37%
14	$i \cdot \exp(i\omega)$	26,88%	83%	46,64%	93,68%	0,79%	1,58%
15	$\exp(2i\omega)$	22,92%	86,96%	52,96%	96,84%	2,77%	20,95%
16	$i \cdot \exp(2i\omega)$	24,9%	84,98%	50,99%	96,05%	1,98%	7,91%
17	$1/(\omega + 0.1)$	54,55%	96,44%	59,68%	91,3%	17%	26,48%
18	$i/(\omega + 0.1)$	56,52%	96,84%	64,03%	94,07%	20,95%	27,27%
19	$1/(\omega^2 + 0.1)$	63,64%	97,23%	67,98%	94,86%	20,55%	27,67%
20	$i/(\omega^2 + 0.1)$	57,71%	96,84%	58,89%	94,86%	15,02%	25,69%
21	$\omega/(\omega + 1)$	39,92%	89,72%	52,57%	91,3%	17%	25,3%
22	$i \cdot \omega/(\omega + 1)$	20,55%	86,17%	44,66%	88,14%	0%	1,19%
23	$\omega^2/(\omega^2 + 1)$	39,92%	90,12%	57,31%	92,89%	17,39%	26,09%
24	$i \cdot \omega^2/(\omega^2 + 1)$	22,92%	86,17%	45,85%	91,3%	0%	2,37%

Table A.3.: (Lorenz Y) Table shows the percentage of reconstructions for which lyap_spec calculated Lyapunovexponents (L_1,L_2,L_3) which agree with results derived directly from the equations. As source was used the y-component of a Lorenz-attractor. As accuracy was used the average relative error (columns marked with "rel.") and average absolute error ("abs.") estimated by lyap_spec itself.

	$f(\omega)$	L_1 rel.	L_1 abs.	L_2 rel.	L_2 abs.	L_3 rel.	L_3 abs.
1	1	10,91%	49,09%	80%	94,55%	18,18%	49,09%
2	i	1,82%	20%	81,82%	96,36%	0%	27,27%
3	ω	10,91%	58,18%	83,64%	98,18%	12,73%	49,09%
4	$i \cdot \omega$	34,55%	52,73%	47,27%	94,55%	0%	7,27%
5	$-\omega^2$	23,64%	74,55%	67,27%	98,18%	3,64%	16,36%
6	$i \cdot \omega^2$	9,09%	30,91%	83,64%	100%	3,64%	18,18%
7	$\exp(i \cdot 10\omega)$	14,55%	54,55%	85,45%	100%	12,73%	49,09%
8	$\exp(i \cdot 20\omega)$	12,73%	56,36%	85,45%	100%	10,91%	40%
9	$\exp(i \cdot 100\omega)$	7,27%	50,91%	81,82%	100%	10,91%	38,18%
10	$\exp(i \cdot 200\omega)$	10,91%	52,73%	87,27%	100%	14,55%	56,36%
11	$\log(\omega + 1)$	29,09%	67,27%	83,64%	98,18%	10,91%	36,36%
12	$\arctan(\omega)$	30,91%	70,91%	87,27%	98,18%	10,91%	38,18%

Table A.4.: (Lorenz Z) Table shows the percentage of reconstructions for which lyap_spec calculated Lyapunovexponents (L_1,L_2,L_3) which agree with results derived directly from the equations. As source was used the z-component of a Lorenz-attractor. As accuracy was used the average relative error (columns marked with "rel.") and average absolute error ("abs.") estimated by lyap_spec itself.

B. Software

```
function [Datensatz, F_Datensatz] = fftrec(arg1, arg2, arg3,
                                          arg4, arg5, arg6, arg7);
%
%#################################################################
%function for space time reconstruction. The Fourier transformed
%input data will be multiplied with frequency-depending functions
%to produce additional dimensions. (Version 1.10)
%#################################################################
%
%[Data, Fourier_Data] = fftrec(TIMESERIES, resolution, FUNCTIONS,
%                       plots, plotstyle, {axis1, axis2,...}, dt);
%
%##################### OPTIONAL OUTPUT ###########################
%Data: reconstructed attractor (matrix)
%   1.column = input data
%   2.column = 1.reconstructed dimension
%   3.column = 2.reconstructed dimension
%   ...
%
%Fourier_Data: reconstructed attractor at the frequency domain
%                                                        (matrix)
%   1.column = input data  (frequency space)
%   2.column = 1.reconstructed dimension (frequency space)
%   3.column = 2.reconstructed dimension (frequency space)
%   ...
%
%###################### MANDATORY INPUT ##########################
%timeseries: input data (vector)
%
```

```
%functions: functions to use for reconstruction arranged in a cell
%           array.
%           structure = {@(x) func1(x), @(x) func2(x),...}
%           Functions only have to be defined for positive x
%(negative part has to be the complex conjugated of this function)
%
%##################### OPTIONAL INPUT #############################
%resolution (optional): resolution=n means: every n-th point
%                       of the input data will be used for
%                       reconstruction
%
%dt (optional): difference between two data points of the input
%               data in seconds.
%
%plots (optional): Choose of the plots that should be calculated
%            plots = 'no plot' : no plots
%            plots = 'all plots' : 3D and 2D plots (standard)
%            plots = '2D plot' : only 2D plots
%            plots = '3D plot' : only 3D plots
%
%plotstyle (optional): Input of the plot design
%                      (standard matlab format, e.g. 'k-')
%
%axis (optional): plot-scaling-factor. Graph will be plotted
%                 in the area -axis...axis
%##################################################################


%############################## CHANGELOG #########################
%1.01: -Fixed a bug that prevented the calculation of the negative
%       frequency part
%1.02: -Reduced the calculation time due to initializing the value
%       "Datensatz" with zeros before using.
%1.03: -renamed function to "fftrec"
%      -resorted input parameters for better usabilty
%      -implemented standard values for unchoosen parameters
%1.04: -improved input for optional parameters
```

```
%      -replaced numbers with text commands for parameter "plots"
%1.05: -added part to set the omega=0 component of the frequency
%       spectrum to zero to prevent complex values in the time
%       domain
%1.06: -added 1D-graphs of the reconstructed timeseries
%1.07: -added parameter "frequencyinput" to control the kind of
%       inputvalues for used functions (experimental only)
%1.08: -changed reconstruction calculations: negative frequency
%       part is now calculated directly as complex conjugated
%       of the positive frequency part
%1.10: -restricted datasets to ones with an odd length to prevent
%       errors induced by calculations in the frequency domain
%       and adjusted calculations for that purpose
%      -Inserted a routine for handling with datasets of even
%       length removing the last entry.
%#################################################################

%#################################################################
%Some additional configurations

%Standard values for arguments requested at the execution of
%this function
standard_resolution = 1;
standard_functions = {@(x) 1, @(x) i*x, @(x) x};
standard_plots = 1;
standard_plotstyle = 'k-';
standard_dt = 1;

%Additional information for the output (actually only correlation)
AdditionalInformation = 1;

%Defining wheter functions should get real frequencies
%(frequencyinput = 1 STANDARD) or integers from 1 to maxn/2
%(frequencyinput = 0) as input. The second option is useful if
%there are predefined arrays with numbers instead of functions
%as input
```

```
frequencyinput = 1;

%##################################################################
AutoScale = 1;
Korn = standard_resolution;
funktionen = standard_functions;
plots = standard_plots;
plotstyle = standard_plotstyle;
deltaT = standard_dt;

%#####################READING PARAMETERS###########################

if nargin == 0 %Testing mode without parameters, only for testing
    load roessler2.dat; Datenquelle = roessler2(:,3); end;

if nargin > 0 Datenquelle = arg1; end;

if nargin > 1
    if isnumeric(arg2) Korn = arg2;
        if nargin > 2 funktionen = arg3; end;
    else
        funktionen = arg2;
        if nargin > 2
            if isnumeric(arg3)
                deltaT = arg3;
            else
                if ischar(arg3)
                    switch arg3
                        case ’no plot’   plots = 0;
                        case ’all plots’ plots = 1;
                        case ’2D plot’   plots = 2;
                        case ’3D plot’   plots = 3;
                        case ’1D plot’   plots = 4;
                        otherwise        plotstyle = arg3;
                    end;
                else
                    Achsen = arg3; AutoScale = 0;
                end;
            end;
        end;
    end;
end;

if nargin > 3
    if isnumeric(arg4) deltaT = arg4;
    else
       if ischar(arg4)
           switch arg4
               case 'no plot'    plots = 0;
               case 'all plots'  plots = 1;
               case '2D plot'    plots = 2;
               case '3D plot'    plots = 3;
               case '1D plot'    plots = 4;
               otherwise         plotstyle = arg4;
           end;
       else
            Achsen = arg4; AutoScale = 0;
       end;
    end;
end;

if nargin > 4
    if isnumeric(arg5) deltaT = arg5;
    else
       if ischar(arg5)
           switch arg5
               case 'no plot'     plots = 0;
               case 'all plots'   plots = 1;
               case '2D plot'     plots = 2;
               case '3D plot'     plots = 3;
               case '1D plot'     plots = 4;
               otherwise          plotstyle = arg5;
           end;
       else
```

```
            Achsen = arg5; AutoScale = 0;
       end;
    end;
end;

if nargin > 5
    if isnumeric(arg6) deltaT = arg6;
    else
       if ischar(arg6)
           switch arg6
               case ’no plot’     plots = 0;
               case ’all plots’   plots = 1;
               case ’2D plot’     plots = 2;
               case ’3D plot’     plots = 3;
               case ’1D plot’     plots = 4;
               otherwise          plotstyle = arg6;
           end;
       else
            Achsen = arg6; AutoScale = 0;
       end;
    end;
end;

if nargin > 6
    if isnumeric(arg7) deltaT = arg7;
    else
       if ischar(arg7)
           switch arg7
               case ’no plot’     plots = 0;
               case ’all plots’   plots = 1;
               case ’2D plot’     plots = 2;
               case ’3D plot’     plots = 3;
               case ’1D plot’     plots = 4;
               otherwise          plotstyle = arg7;
           end;
       else
            Achsen = arg7; AutoScale = 0;
```

```
        end;
    end;
end;

%##################################################################
deltaT = deltaT*Korn;
maxdimension = length(funktionen);
maxn = floor(length(Datenquelle)/Korn);

%############DEFINING GRAPHICAL OUTPUT#########################
if (maxdimension == 1) x1d = 1; y1d = 1; end;
if (maxdimension == 2) x2d = 1; y2d = 1; x1d = 1; y1d = 2; end;
if (maxdimension == 3) x3d = 1; y3d = 1; x2d = 2; y2d = 2;
                       x1d = 1; y1d = 3; end;
if (maxdimension == 4) x3d = 2; y3d = 2; x2d = 3; y2d = 2;
                       x1d = 1; y1d = 2; end;
if (maxdimension > 4)  x3d = 3; y3d = 2; x2d = 3; y2d = 2;
                       x1d = 1; y1d = 3; end;

%##################################################################
if (maxn/2 == floor(maxn/2)) maxn = maxn-1;
    'NOTE: Last entry of timeseries removed to recieve an
     symmetric dataset in frequency space!' end;

Datensatz = zeros(maxn,maxdimension+1);
for n1=1:1:maxn; Datensatz(n1,1) = Datenquelle(n1*Korn,1); end;
F_Datensatz = zeros(maxn,maxdimension+1);
F_Datensatz(:,1) = fftshift(fft(Datensatz(:,1)));
w_middle = ceil(maxn/2);

if (frequencyinput == 1)
    for n2 = 1:1:maxdimension;
        for n1 = 1:1:w_middle-1;
            F_Datensatz(w_middle+n1,n2+1)
                     = F_Datensatz(n1+w_middle,1)
                          *funktionen{n2}(2*pi*n1/(deltaT*maxn));
            F_Datensatz(w_middle-n1,n2+1)
```

```
                      = conj(F_Datensatz(n1+w_middle,n2+1));
        end;
        %setting omega = 0 component to zero to prevent
        %complex values in the time domain
        F_Datensatz(w_middle,n2+1) = 0;
    end;
else
    for n2 = 1:1:maxdimension;
        for n1 = 1:1:w_middle-1;
            F_Datensatz(w_middle+n1,n2+1)
                  = F_Datensatz(n1+w_middle,1)*funktionen{n2}(n1);
            F_Datensatz(w_middle-n1,n2+1)
                  = conj(F_Datensatz(n1+w_middle,n2+1));
        end;
        %setting omega = 0 component to zero to prevent
        %complex values in the time domain
        F_Datensatz(w_middle,n2+1) = 0;
    end;
end;

for n2 = 1:1:maxdimension;
    Datensatz(:,n2+1) = ifft(ifftshift(F_Datensatz(:,n2+1)));
end;

n4 = 1; n5 = 1;

if (maxdimension > 2)
    if (plots == 1) | (plots == 3)
        for n1 = 1:1:maxdimension-2;
            for n2 = n1+1:1:maxdimension-1;
                for n3 = n2+1:1:maxdimension;
                    if (n4 == x3d*y3d + 1)
                        n4 = 1;
                        n5 = n5 + 1;
                    end;
                    if (n4 == 1)
                    figure(’Name’, [’3D-plots - part ’ int2str(n5)
                    ’/’ int2str(ceil(factorial(maxdimension)
                    /(6*factorial(maxdimension-3))/(x2d*y2d)))]);
                    end;
                    subplot(y3d,x3d,n4);
                    plot3(real(Datensatz(:,n1+1)),
                          real(Datensatz(:,n2+1)),
                          real(Datensatz(:,n3+1)),
                          plotstyle,
                          ’MarkerSize’,1);
                    if (AutoScale == 0)
                        axis([-Achsen{n1}  Achsen{n1}  -Achsen{n2}
                             Achsen{n2} -Achsen{n3}  Achsen{n3}]);
                    end;
                    if (AdditionalInformation == 1)
                        Korrelation1 = corrcoef(
                                         real(Datensatz(:,n1+1)),
                                         real(Datensatz(:,n2+1)));
                        Korrelation2 = corrcoef(
                                         real(Datensatz(:,n1+1)),
                                         real(Datensatz(:,n3+1)));
                        Korrelation3 = corrcoef(
                                         real(Datensatz(:,n2+1)),
                                         real(Datensatz(:,n3+1)));
                        title([’x’ int2str(n1) ’ - x’ int2str(n2)
                              ’ - x’ int2str(n3) ’ (’
                              num2str(Korrelation1(2,1)) ’, ’
                              num2str(Korrelation2(2,1)) ’, ’
                              num2str(Korrelation3(2,1)) ’)’],
                              ’Color’,’r’);
                    else
                        title([’x’ int2str(n1) ’ - x’ int2str(n2)
                               ’ - x’ int2str(n3)],’Color’,’r’);
                    end;
                    xlabel([’x’ int2str(n1) ’ : ’
                           func2str(funktionen{n1})],’Color’,’b’);
                    ylabel([’x’ int2str(n2) ’ : ’
                           func2str(funktionen{n2})],’Color’,’b’);
```

```
                    zlabel(['x' int2str(n3) ' : '
                           func2str(funktionen{n3})],'Color','b');
                    n4 = n4 + 1;
                end;
            end;
        end;
    end;
end;

n4 = 1; n5 = 1;

if (maxdimension > 1)
    if (plots == 1) | (plots == 2)
        for n1 = 1:1:maxdimension-1;
            for n2 = n1+1:1:maxdimension;
                if (n4 == x2d*y2d + 1)
                    n4 = 1;
                    n5=n5+1;
                end;
                if (n4 == 1)
                figure('Name', ['2D-plots - part ' int2str(n5) '/'
                     int2str(ceil(factorial(maxdimension)
                     /(2*factorial(maxdimension-2))/(x2d*y2d)))]);
                end;
                subplot(y2d,x2d,n4);
                plot(real(Datensatz(:,n1+1))
                    ,real(Datensatz(:,n2+1))
                    ,plotstyle, 'MarkerSize',1);
                if AutoScale == 0 axis([-Achsen{n1}  Achsen{n1}
                                        -Achsen{n2}  Achsen{n2}]);
                end;
                if (AdditionalInformation == 1)
                    Korrelation = corrcoef(
                                         real(Datensatz(:,n1+1))
                                        ,real(Datensatz(:,n2+1)));
                    title(['x' int2str(n1) ' - x' int2str(n2) '
                          (' num2str(Korrelation(1,2)) ')'],
                          'Color','r');
                else
                    title(['x' int2str(n1) ' - x' int2str(n2)],
                          'Color','r');
                end;
                xlabel(['x' int2str(n1) ' : '
                          func2str(funktionen{n1})],'Color','b');
                ylabel(['x' int2str(n2) ' : '
                          func2str(funktionen{n2})],'Color','b');
                n4 = n4 + 1;
            end;
        end;
    end;
end;

n2 = 1; n3 = 1;

if (plots == 1) | (plots == 4)
    for n1 = 1:1:maxdimension;
       if (n2 == x1d*y1d + 1)
           n2 = 1;
           n3 = n3 + 1;
       end;
       if (n2 == 1)
           figure('Name', ['1D-plots - part ' int2str(n3)
                  '/' int2str(ceil(maxdimension/(x1d*y1d)))]);
       end;
       subplot(y1d,x1d,n2);
       plot(Datensatz(:,n1+1), plotstyle,'MarkerSize',1);
       if AutoScale == 0 ylim([-Achsen{n1}  Achsen{n1}]); end;
       xlabel('number of datapoint','Color','b');
       ylabel(['x' int2str(n1) ' : '
               func2str(funktionen{n1})],'Color','b');
       n2 = n2 + 1;
    end;
end;
```

Bibliography

[1] BADII, R. ; BROGGI, G. ; DERIGHETTI, B. ; RAVANI, M. ; CILIBERTO, S. ; POLITI, A. ; RUBIO, M. A.: Dimension increase in filtered chaotic signals. In: *Physical Review Letters* 60 (1988), Nr. 11, S. 979–982. – ISSN 1079–7114

[2] BADII, R. ; POLITI, A.: On the fractal dimension of filtered chaotic signals. In: *Dimensions and entropies in chaotic systems: quantification of complex behavior: proceedings of an international workshop at the Pecos River Ranch, New Mexico, September 11-16, 1985*, 1986. – ISBN 3540162542, S. 67

[3] BLASIUS, B.: *Synchronized fade-outs of epidemic outbreaks in networks of cities. Reconstruction of susceptibles*. – to be published

[4] ECKMANN, J. P. ; KAMPHORST, S. O. ; RUELLE, D. ; CILIBERTO, S.: Liapunov exponents from time series. In: *Physical Review A* 34 (1986), Nr. 6, S. 4971–4979. – ISSN 1094–1622

[5] FALCONER, K.: *Fractal geometry: mathematical foundations and applications*. Wiley & Sons, 1990

[6] FENSTERMACHER, P. R. ; SWINNEY, H. L. ; GOLLUB, J. P.: Dynamical instabilities and the transition to chaotic Taylor vortex flow. In: *Journal of fluid mechanics* 94 (1979), Nr. 01, S. 103–128. – ISSN 0022–1120

[7] GILMORE, R.: Topological analysis of chaotic dynamical systems. In: *Reviews of Modern Physics* 70 (1998), Nr. 4, S. 1455–1529. – ISSN 1539–0756

[8] GOLLUB, J.: The onset of turbulence: convection, surface waves, and oscillators. In: *Systems Far from Equilibrium* (1980), S. 162–180

[9] GOLLUB, J. P. ; SWINNEY, H. L.: Onset of turbulence in a rotating fluid. In: *Physical Review Letters* 35 (1975), Nr. 14, S. 927–930. – ISSN 1079–7114

[10] GREBOGI, C. ; OTT, E. ; PELIKAN, S. ; YORKE, J. A.: Strange attractors that are not chaotic. In: *Physica D: Nonlinear Phenomena* 13 (1984), Nr. 1-2, S. 261–268. – ISSN 0167–2789

[11] HAKEN, H.: At least one Lyapunov exponent vanishes if the trajectory of an attractor does not contain a fixed point. In: *Physics Letters A* 94 (1983), Nr. 2, S. 71–72. – ISSN 0375–9601

[12] HEGGER, R. ; KANTZ, H. ; SCHREIBER, T.: Practical implementation of nonlinear time series methods: The TISEAN package. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 9 (1999), S. 413

[13] HÉNON, M.: A two-dimensional mapping with a strange attractor. In: *Communications in Mathematical Physics* 50 (1976), Nr. 1, S. 69–77. – ISSN 0010–3616

[14] KANTZ, H.: A robust method to estimate the maximal Lyapunov exponent of a time series. In: *Physics Letters A* 185 (1994), Nr. 1, S. 77–87. – ISSN 0375–9601

[15] KANTZ, H. ; SCHREIBER, T.: *Nonlinear time series analysis*. Cambridge Univ Press, 2004. – ISBN 0521529026

[16] KAPLAN, J. ; YORKE, J.: Chaotic behavior of multidimensional difference equations. In: *Functional Differential equations and approximation of fixed points* (1979), S. 204–227

[17] LORENZ, E. N.: Deterministic Nonperiodic Flow. In: *Journal of the Atmospheric Sciences* 20 (1963), S. 130

[18] MILNOR, J.: On the concept of attractor. In: *Communications in Mathematical Physics* 99 (1985), Nr. 2, S. 177–195. – ISSN 0010–3616

[19] MITSCHKE, F. ; MÖLLER, M. ; LANGE, W.: Measuring filtered chaotic signals. In: *Physical Review A* 37 (1988), Nr. 11, S. 4518–4521. – ISSN 1094–1622

[20] PACKARD, N. H. ; CRUTCHFIELD, J. P. ; FARMER, J. D. ; SHAW, R. S.: Geometry from a time series. In: *Physical Review Letters* 45 (1980), Nr. 9, S. 712–716. – ISSN 1079–7114

[21] RÖSSLER, O. E.: An equation for continuous chaos. In: *Physics Letters A* 57 (1976), Nr. 5, S. 397–398. – ISSN 0375–9601

[22] RÖSSLER, O. E.: Chaotic oscillations: an example of hyperchaos. In: *Nonlinear Oscillations in Biology* (1979), S. 141–156

[23] RÖSSLER, O. E.: An equation for hyperchaos. In: *Physics Letters A* 71 (1979), Nr. 2-3, S. 155–157. – ISSN 0375–9601

[24] ROUX, J. C. ; SIMOYI, R. H. ; SWINNEY, H. L.: Observation of a strange attractor. In: *Physica D: Nonlinear Phenomena* 8 (1983), Nr. 1-2, S. 257–266. – ISSN 0167–2789

[25] SANO, M. ; SAWADA, Y.: Measurement of the Lyapunov spectrum from a chaotic time series. In: *Physical Review Letters* 55 (1985), Nr. 10, S. 1082–1085. – ISSN 1079–7114

[26] SAUER, T. ; YORKE, J. A. ; CASDAGLI, M.: Embedology. In: *Journal of Statistical Physics* 65 (1991), Nr. 3, S. 579–616. – ISSN 0022–4715

[27] SHERMAN, J. ; MCLAUGHLIN, J.: Power spectra of nonlinearly coupled waves. In: *Communications in Mathematical Physics* 58 (1978), Nr. 1, S. 9–17. – ISSN 0010–3616

[28] SHIMADA, I. ; NAGASHIMA, T.: A numerical approach to ergodic problem of dissipative dynamical systems. In: *Prog. Theor. Phys* 61 (1979), Nr. 6, S. 1605–1616

[29] STROGATZ, S. H.: *Nonlinear Dynamics and Chaos*. Westview Press, 1994

[30] TAKENS, F.: Detecting strange attractors in turbulence. In: *Lecture Notes in Mathematics* (1981), S. 366–381

[31] THEILER, J.: Estimating fractal dimension. In: *Journal of the Optical Society of America A* 7 (1990), Nr. 6, S. 1055–1073. – ISSN 1084–7529

[32] WELCH, P.: The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. In: *IEEE Transactions on Audio and Electroacoustics* 15 (1967), Nr. 2, S. 70–73. – ISSN 0018–9278

[33] WHITNEY, H.: Differentiable manifolds. In: *Annals of Mathematics* 37 (1936), Nr. 3, S. 645–680. – ISSN 0003–486X

[34] WILLIAMS, R. F.: The zeta function of an attractor. In: *Conference on the Topology of Manifolds (Michigan State University, E. Lansing, Mich., 1967) Prindle, Weber & Schmidt, Boston, Mass*, 1968, S. 155–161

[35] WOLF, A. ; SWIFT, J. B. ; SWINNEY, H. L. ; VASTANO, J. A.: Determining Lyapunov exponents from a time series. In: *Physica D: Nonlinear Phenomena* 16 (1985), Nr. 3, S. 285–317. – ISSN 0167–2789

List of Figures

List of Tables

Deutsche Zusammenfassung

Diese Diplomarbeit behandelt das Thema "Attraktorrekonstruktion". Dabei handelt es sich um eine Technik, die es ermöglicht, aus einer einzelnen Zeitreihe den vollständigen Phasenraum des Systems zu rekonstruieren und somit Rückschlüsse auf topologische Eigenschaften dieses dynamischen Systems zu ziehen **(Kapitel 1)**. Es gibt bereits mehrere Methoden der Attraktorrekonstruktion. Die wohl bekannteste und am weitesten verbreitete Methode ist der "Time-Delay"-Ansatz. Bei dieser Technik wird die Zeitreihe $x(t)$ einfach um τ Punkte verschoben und danach als zusätzliche Dimension verwendet. Der volle Phasenraum hat dann die Struktur:

$$v(t) = (x(t), x(t+\tau), x(t+2\tau), ...)$$

Neben Time-Delay existieren weitere Methoden wie z.B. Rekonstruktion durch Ableitung oder durch Integration **(Kapitel 1.4)**. Alle diese Methoden erzeugen eine korrekte Rekonstruktion des Phasenraums, sehen jedoch in der Anwendung sehr unterschiedlich aus. Ansatz dieser Diplomarbeit war die Idee, dass es eine Verallgemeinerung dieser verschiedenen Methoden geben muss, da sie alle letztendlich demselben Zweck dienen. Die Lösung wurde dabei im Frequenzraum vermutet: Denn obwohl alle Methoden im Ortsraum sehr unterschiedlich aussehen, erhalten sie durch eine Fourier-Transformation ein einheitliches Gesicht: Alle untersuchten Methoden konnten auf folgende Struktur reduziert werden **(Kapitel 2)**:

$$\tilde{x}_n(\omega) = f_n(\omega)\tilde{x}(\omega)$$

Somit ist das Fouriertransformierte der n-ten rekonstruierten Dimension lediglich das Fouriertransformierte der Originalzeitreihe multipliziert mit einer Rekonstruktionsfunktion $f_n(\omega)$. Die nächste Frage war nun: Gibt es neben den klassischen

Methoden also noch weitere Rekonstruktionsvarianten? Welche Art von Rekonstruktionsfunktionen $f_n(\omega)$ sind erlaubt? Auf der Suche nach der Antwort kristallisierten sich vor allem drei Forderungen an die Rekonstruktionsfunktionen heraus **(Kapitel 2.3.2)**:

- **reellwertiges Ergebnis im Ortsraum**

  Durch die Fouriertransformation erhält man im Frequenzraum zunächst eine komplexe Funktion. Gleichzeitig weiß man jedoch, dass die Zeitreihen und somit auch die rekonstruierten Dimensionen reellwertig sein müssen. übersetzt auf die Rekonstruktionsfunktionen bedeutet dies, dass das komplex konjugierte der Funktion für $-\omega$ dem Wert der Funktion für $+\omega$ entsprechen muss:

  $$f_n(\omega) = \overline{f_n(-\omega)}$$

- **lineare Unabhängigkeit**

  Im Ortsraum ist eine bekannte Forderung, dass alle Komponenten linear unabhängig sein müssen. Andernfalls wären Komponenten redundant und der Phasenraum würde nicht vollständig aufgespannt. Aufgrund der Linearität der Fouriertransformation führt dies auch direkt zu einer linearen Unabhängigkeit der Rekonstruktionsfunktionen.

  $$f_1(\omega)...f_n(\omega) \text{ linear unabhängig}$$

- **langsame Veränderlichkeit**

  Ein weiterer wichtiger Punkt, der mathematisch jedoch nicht exakt bestimmt werden konnte, ist die langsame Veränderlichkeit der Rekonstruktionsfunktionen: Zum einen muss die Rekonstruktionsfunktion die originale Zeitreihe in irgendeiner Form verändern, damit das Ergebnis linear unabhängig von der originalen Zeitreihe sein kann, diese Veränderung darf jedoch auch nicht zu stark sein. Stellt man sich z.B. als Rekonstruktionsfunktion eine Aneinanderreihung von Zufallszahlen vor, so wird schnell klar, dass in diesem Fall die Informationen der originalen Zeitreihe vollständig überschrieben werden würden. Deshalb darf sich die Rekonstruktionsfunktion im Vergleich zur originalen Zeitreihe nur sehr wenig verändern, um die Informationen des

Originalsystems zu erhalten. Aus diesem Grund lag die Idee nahe zu fordern, dass die Rekonstruktionsfunktion langsam veränderlich sein muss.

$$f_n(\omega) \text{ langsam veränderlich}$$

Beachtet man nun diese drei Einschränkungen, so bleiben dennoch unendlich viele Möglichkeiten für Rekonstruktionen. Viele dieser Varianten werden von den klassischen Methoden nicht abgedeckt und sind somit neu. Eine interessante Anwendung kann z.B. die Kombination von Rekonstruktion und Filterung sein **(Kapitel 4)**. Des Weiteren sind Situationen denkbar, in welchen es nötig ist, der Rekonstruktion gewisse Eigenschaften aufzudrücken. Dies ist nun durch die Wahlfreiheit der Rekonstruktionsfunktionen möglich **(Kapitel 2.3.4)**.

Leider war es mir während meiner Diplomarbeit nicht möglich, einen mathematischen Beweis dafür zu finden, dass die Rekonstruktion unter der Verwendung des Frequenzraumes eine korrekte Rekonstruktion des Phasenraums ermöglicht. Dies ist zwar naheliegend, jedoch nicht bewiesen. Als Ersatz habe ich daraufhin versucht, die Methode heuristisch zu untermauern. Zu diesem Zweck habe ich einen Blick auf verschiedene Invarianten dynamischer Systeme geworfen **(Kapitel 3)**: Es gibt einige Größen, z.B. Lyapunovexponent oder fraktale Dimension, welche theoretisch nur vom dynamischen System und nicht von seiner Einbettung im Phasenraum abhängen. Somit sollten die Werte dieser Größen für alle Rekonstruktionen übereinstimmen. Um dies zu zeigen, habe ich 24 verschiedene Rekonstruktionsfunktionen gewählt und in allen möglichen Kombinationen zur Rekonstruktion verwendet. Danach wurde für jede Rekonstruktion das Lyapunovepsktrum berechnet und die Ergebnisse verglichen. Unglücklicherweise waren die Ergebnisse jedoch nicht so eindeutig wie erhofft. Dies hängt jedoch vor allem mit der Berechnung des Lyapunovspektrums zusammen: Zeitreihen stellen jedoch immer nur eine Näherung des Originalsystems dar, daher können auch alle Bestimmungen von Invarianten nur Näherungen sein. Dementsprechend sind diese Näherungen in der Praxis auch nicht unabhängig von der Einbettung. Normalerweise wird dieser Effekt durch die Wahl verschiedenster Parameter der Berechnung reduziert, jedoch musste der Vergleich mit einem einzigen Parameterset durchgeführt werden, um die Vergleichbarkeit zu gewährleisten. Diese "unsachgemäße" Anwendung führte somit zu den uneindeutigen Ergebnissen. Aber immerhin ein Ergebnis war relativ eindeutig: Durch nahezu alle Lyapunovspektren wurde korrekt wiedergegegeben, dass es sich um chaotische Systeme in der Untersuchung handelte. Die Analyse der Kaplan-Yorke-Dimension lieferte schließlich eine Art Zusammenfassung dieses Kapitels. Es zeigte sich dass das Spektrum der berechneten Dimensionen im Histogramm relativ breit war und somit viele Ergebnisse von dem korrekten Wert abwichen. Gleichzeitig lagen die Maxima jedoch immer an den Stellen der zu erwartenden Werte. Somit lässt sich vermuten, dass scheinbar nicht alle Einbettung erfolgreich waren (die Fehler könnten alternativ allerdings auch durch Probleme innerhalb der Dimensionsberechnung entstanden sein), dass jedoch ein Großteil der Rekonstruktionen funktioniert haben muss.

Auf der Suche nach einem Beweisansatz ergaben sich viele weitere Teilergebnisse und Ideen **(Kapitel 5)**. So zeigte sich z.B., dass man auch Rekonstruktionsmethoden finden kann, welche nicht dem propagierten Muster entsprechen, dass diese jedoch scheinbar jeweils eine Rekonstruktion als Basis verwenden, welche wiederum in das Muster passt. Die zusätzlichen Teile der Rekonstruktion hatten dabei nur Einfluss auf Symmetrieeigenschaften der Rekonstruktion, halfen bei der eigentlichen Rekonstruktion jedoch nicht mit. Es lässt sich daher vermuten, dass man bei der besprochenen Struktur von einer Art Basis für die Rekonstruktion sprechen könnte **(Kapitel 5.1)**.

Bezüglich eines Beweises war es auch interessant zu überlegen, wie plausibel diese Methode erscheint. Dabei fällt auf, dass die relativ restriktive Form der Rekonstruktionsmethoden zur Folge hat, dass alle Dimensionen eines rekonstruierten Systems zwangsläufig dieselben Peaks im Powerspektrum besitzen müssen und somit die Powerspektren von allen Komponenten sehr ähnlich aussehen müssen. Um zu überprüfen, ob dies plausibel ist, wurden die Powerspektren von Originalsystemen genauer untersucht. Dabei stellte sich heraus, dass nicht zwangsläufig alle Komponenten dieselben Peaks enthalten. Bei den untersuchten Beispielen führten Symmetrieoperationen jedoch zu Formen, in welchen diese Bedingung wiederum erfüllt war **(Kapitel 5.2)**.

Diese Symmetrieüberlegung wiederum könnte erklären, weshalb die Rekonstruktionen des Lorenzsystems immer eine vom Originalsystem abweichende Symmetrie aufweisen. Die Untersuchungen legten die Vermutung nahe, dass bei einer Rekonstruktion für jede Komponente immer die Symmetrie der Originalzeitreihe übernommen wird. Das würde bedeuten, dass die Symmetrie des Originalsystems bei der Rekonstruktion leider verloren geht. Des Weiteren würde es erklären, weshalb die Rekonstruktion des Lorenzssystems aus der Z-Komponente zu merkwürdigen Resultaten führt **(Kapitel 5.3)**.

Ein weiteres wichtiges Thema bezüglich der Rekonstruktion ist die Frage nach der optimalen Einbettung. Zu diesem Zweck wurde nach verschiedenen Methoden zur Beurteilung einer Rekonstruktion gesucht. Dabei habe ich zwei Ansätze verfolgt: Zum einen den passiven Ansatz der Definition eines Maßes für die Einbettungsqualität, zum anderen eine aktive Methode zur direkten Bestimmung der optimalen Einbettung mittels orthogonaler Funktionen. Leider lieferten beide Methoden soweit keine nutzbaren Ergebnisse, jedoch erscheinen die Ansätze an sich für mich sinnvoll, so dass weitere Untersuchungen auf diesem Gebiet sicher Ergebnisse bringen sollten **(Kapitel 5.5)**.

Ein letzter interessanter Punkt ist die Interpretation der Rekonstruktion als Filterung: Schaut man sich die propagierte Struktur im Frequenzraum an, so stellt man fest, dass diese mit der Struktur eines Filters übereinstimmt. Dies bedeutet, dass eine Rekonstruktion nichts anderes ist, als die originale Zeitreihe in gefilterter Form als neue Komponente zu verwenden. Neben der Nützlichkeit dieses Ergebnisses für das Verständnis einer Rekonstruktion liefert es zudem die Möglichkeit, Ergebnisse der Filtertheorie nun direkt auf die Rekonstruktion von Systemen anzuwenden **(Kapitel 5.7)**.

Aus meiner Sicht bringt diese Verallgemeinerung der Rekonstruktion viele Möglichkeiten für zukünftige Anwendungen mit sich. Zum einen erleichtert die Kombination, klassische Methoden gegeneinander abzuwägen, zum anderen sind nun aber auch völlig neue Ansätze möglich. Es ist nun bedeutend leichter, der Rekonstruktion bestimmte Eigenschaften zu geben und sie besser an die Aufgabenstellung anzupassen **(Kapitel 6)**.

Um das gesamte Potential ausnutzen zu können, sind vor allem zwei weitere Schritte wichtig: Zum einen muss ein Beweis der Methode gefunden werden, um sie auf eine solide Basis zu stellen und die Forderungen an die Rekonstruktionsfunktion mathematisch exakter formulieren zu können. Zum anderen sind Methoden zur Wahl der optimalen Einbettung notwendig, um in der Praxis ein System bestmöglich rekonstruieren zu können.