Measure for Turbulence: Preliminary Ideas

Franz Bozsak

MAE

fbozsak@ucdavis.edu

March 20, 2009

Winter Quarter, 2009

Final Project

PHY 256

UC Davis

Abstract

Starting from the fact that turbulence is a chaotic and random process the hypothesis is suggested that an information theoretic measure for turbulence can exist and that the entropy rate h_μ is an appropriate measure. In order to provide first evidence a 9D Lorenz system is investigated with means of a designed Matlab software toolbox. The necessary background leading to the hypothesis and for the presented analysis is provided. The achieved results corroborate the hypothesis. Future investigation methods are proposed.

Contents

List of Figures

1 Introduction

Turbulence is one of the biggest unsolved physical sciences and engineering problems of our time. Although studied for decades the mechanisms are still barely understood and we are far away from a closed theory. Hence, today's engineers and scientist have to model the phenomenon rather than simulate it. Although today's models provide in some cases fair results, to get there they need a lot of adjustment, which relies on a huge amount of experience and experimental testing. But predicting a system with an (in theory) infinite number of degrees of freedom is very close to impossible and demands a huge amount of computational effort and the restriction to a rather small simulated area, as the Direct-Numerical-Simulation (DNS) approach shows.

The leap from modeling to simulating, in other words, from qualified guessing to predicting, would revolutionize many areas of science, e. g. aeronautical, automotive and space engineering, atmospheric sciences, astrophysics, to name just a few. It could also affect our daily lives, e. g. from tomorrow's means of transportation up to answering the simple question: Shall I bring an umbrella along?

The following report shall introduce some new ideas of how turbulence could be approached. It will focus on the idea of prediction from an information theoretic point of view. Therefore, a method that explains how to get from the governing equations of a well studied flow phenomenon, the Rayleigh-Bénard Convection, to a possible measure of its degree of turbulent behavior, the (information) entropy rate of an attractor, will be presented. Also, a set of Matlab tools was developed that realizes this method and some first results will be shown. Finally, subsequent work will be outlined.

2 From Flow to Entropy Rate

2.1 Rayleigh-Bénard Convection: Governing Equations

The investigations throughout this paper will be restricted to the Rayleigh-Bénard Convection (RBC), which describes the flow of a fluid confined by a hot lower boundary and a cold upper boundary (Fig. 2.1). There are two very straight forward reasons for this constraint: Firstly, while the RBC is a rather simple model flow, it still describes a flow phenomenon that is very common in nature. Secondly, it is (for obvious reasons) the most excessively, one could say, almost solely studied flow when it comes to the methods we are going to present. But by no means should this “restriction” be regarded as a limitation as the methods presented, given the correct modifications, can be applied to any flow phenomenon. Without going into the details of their derivation [8] the governing equations of this flow in

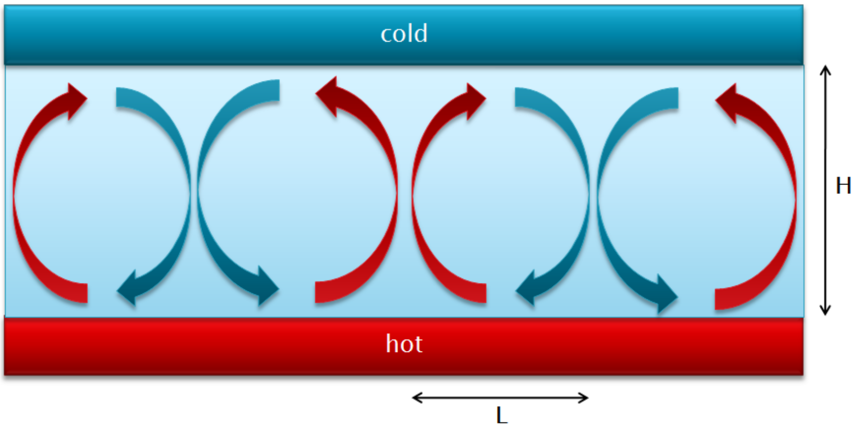

Figure 2.1: Scheme of Rayleigh-Bénard Convection

non-dimensional form are the momentum balance:

$$\frac{1}{\mathrm{Pr}}\left\{-\frac{\partial}{\partial t}\Delta\psi+\frac{\partial}{\partial z}\left(u\frac{\partial u}{\partial x}+w\frac{\partial u}{\partial z}\right)-\frac{\partial}{\partial x}\left(u\frac{\partial w}{\partial x}+w\frac{\partial w}{\partial z}\right)\right\}=-\Delta^2\psi-\frac{\partial\theta}{\partial x} \tag{2.1}$$

and the energy balance:

$$\frac{\partial\theta}{\partial t}+u\frac{\partial\theta}{\partial x}+w\frac{\partial\theta}{\partial z}=\Delta\theta+Ra\cdot w \tag{2.2}$$

They describe the evolution in space and time of two-dimensional deviations (disturbances) from the equilibrium state of the velocity Ψ and temperature Θ field of the flow. The use of stream function Ψ to describe the velocity field guaranties mass balance automatically as it is defined by

$$u=-\frac{\partial\Psi}{\partial z}\quad\text{and}\quad w=\frac{\partial\Psi}{\partial x} \tag{2.3}$$

There are two important non-dimensional parameters in the two equations: The first one is the Prandtl number:

$$\sigma=\frac{\nu}{\kappa} \tag{2.4}$$

The Prandtl number, usually reffered to as Pr, is a property constant of the fluid medium and describes the ratio of viscous over thermal effects. The most common value for the Prandtl number in our context is $\sigma = 10$, which is equivalent to water at a temperature of about 5 °C. The second parameter is the Rayleigh number:

$$Ra=\frac{\alpha g H^3\Delta T}{\kappa\nu} \tag{2.5}$$

This will be the varied parameter of our system. It represents the ratio of buoyancy over thermal effects. As most of the quantities in (2.5) are constant (acceleration of gravity g, coefficient of volume expansion α, coefficient of thermal diffusivity κ) $Ra \propto \Delta T$ and thus, can be regarded as the heating rate of our confined system. The Rayleigh number significantly influences the behavior of the flow: the higher the temperature gradient between the upper and lower boundary the more sensitive the fluid reacts to external disturbances. Up to a critical Rayleigh number Ra_c all disturbances are damped out by the flow and it remains stable. This critical Rayleigh number can be expressed in terms of the aspect ratio $a = \frac{H}{L}$

of our system as

$$Ra_c = \frac{\pi^4(1+a^2)^3}{a^2} \tag{2.6}$$

Once we raise the temperature gradient above this value the flow becomes unstable, the disturbances begin to grow and the well known convection rolls appear. If we keep on increasing the heating rate the flow becomes more and more tumultuous until – passing through the transient phase – it finally enters the turbulent regime and the flow becomes fully chaotic.

2.2 Reducing the Degrees of Freedom

2.2.1 The Galerkin Method

As mentioned before a major problem, when investigating flow phenomena, is the theoretically infinite number of degrees of freedom. In the 1960s the pioneering work of two men, namely Barry Saltzman and Edward N. Lorenz, tried to cope with this problem and introduced a procedure that dramatically reduced the degrees of freedom to a countable amount. It can be seen from experiments that the flow and temperature field of the occurring flow in RBC have periodic behavior. Thus, Saltzman expanded the stream function and the temperature field into a double Fourier expansion:

$$\psi(x,z,t) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \psi(m,n)(t) \exp\left(2\pi h i\left(\frac{m}{L}x + \frac{n}{2H}z\right)\right) \tag{2.7}$$

$$\theta(x,z,t) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \theta(m,n)(t) \exp\left(2\pi h i\left(\frac{m}{L}x + \frac{n}{2H}z\right)\right) \tag{2.8}$$

These are split in their real and imaginary parts: $\psi(m,n) = \psi_1(m,n) - i\psi_2(m,n)$ and $\theta(m,n) = \theta_1(m,n) - i\theta_2(m,n)$. The second step is then to choose a finite number of modes that approximate these expansions, but still contain the main features of the flow. We will talk about this in greater detail in the next chapter. Now these approximations are plugged into (2.1) and (2.2) and a Galerkin method is applied, which exploits the Kronecker delta property of the basis function space of the Fourier series. By multiplying each of the governing equations with each of the chosen modes and then integrating over the domain of the flow the fourth order system of partial differential equations is turned into a system of first order ordinary differential equations for the Fourier coefficients ψ_1, ψ_2, θ_1 and θ_2. These are now the degrees of freedom of our system:

$$\dot{X}_i = f\left(\vec{X}_i\right) \tag{2.9}$$

This system can easily be solved numerically, e. g. with a Runge-Kutta method. In a last step the resulting system is transformed to a final form by introducing a dimensionless time $t = \pi^2 \left(1 + a^2\right) \tau$ and the normalized heating rate r which is defined as $r = \frac{Ra}{Ra_c}$.

2.2.2 Selecting the right Modes

Picking a finite number of modes out of the infinite number of possibilities offered by the Fourier expansion is the true art behind the procedure of effectively reducing the degrees of freedom of the flow. What sounds like a bit of magic at first is founded on a rather small research field of its own, with some of the most elaborate results being very recent [5] [6] [7]. As a rule Saltzman, as well as Lorenz, excluded all ψ_2 and θ_1 modes. Moreover, both did not consider shear flows and thus, ψ_1 modes with $m = 0$ were also excluded.

Since Edward N. Lorenz picked “his” distinct three modes ($X\left(t\right) = \psi_1\left(1,1\right)$, $Y\left(t\right) = \theta_2\left(1,1\right)$, $Z\left(t\right) = \theta_2\left(0,2\right)$) which lead to the celebrated Lorenz system, a lot of authors tried to expand the minimal set of modes to study its effect on the emerging dynamical system. Many of these attempts were carried out without any justification of the choice of modes. But this trial and error method leads only in very few (lucky) cases to a dynamical system worth investigating, hereby not even speaking of the correctness of the underlying physics (Fig. 2.2). Both pioneers already mentioned the importance of energy conservation without going into greater detail.

It was left to Treve and Manley [11] in 1981 to carry out the first systematic analysis of the mentioned energy conservation criterion. Their work resulted in a set of rules for choosing only modes that do not violate energy conservation. Moreover, they classified different mode expansions by their order, namely the number of chosen modes (thus the classical Lorenz system would be of order $k = 3$). Eventually they analyzed the capability of the expansions to approximate some properties of the underlying flow, like the asymptotic and time-dependent behavior in estimating the Nusselt number for different initial values. The

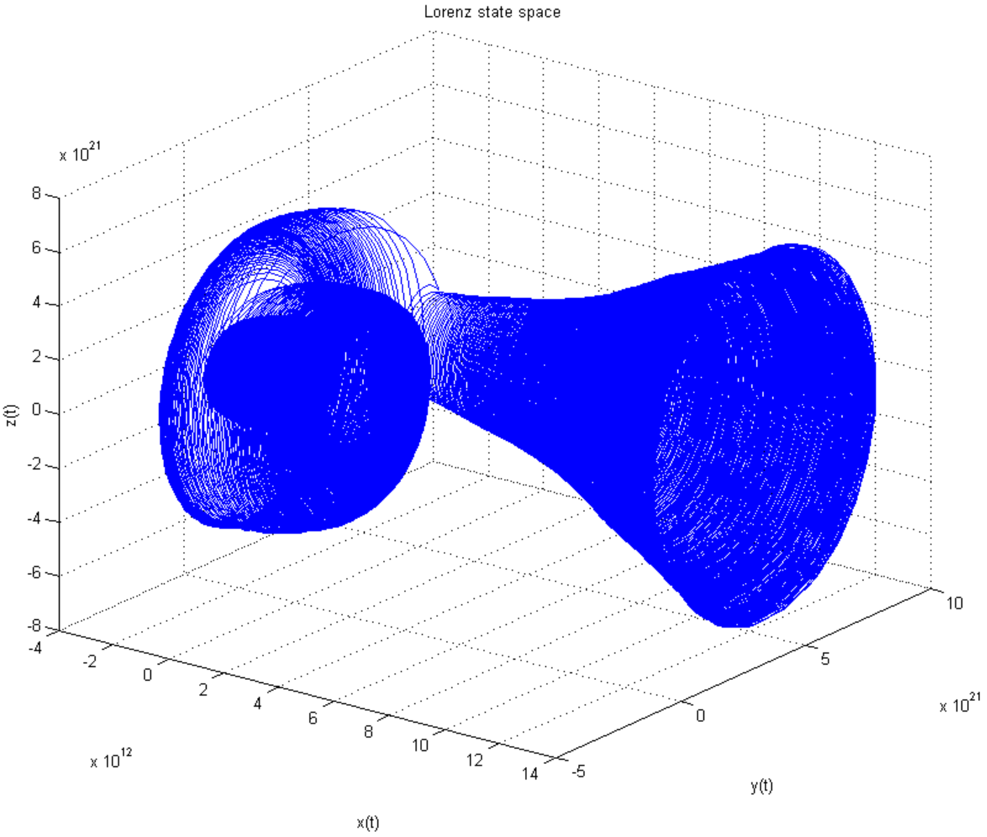

Figure 2.2: Phase portrait of an unphysical dynamical system

main result of Treve's and Manley's research was that for different regimes of the heating rate r a different number of modes is needed in order to get reliable results. The authors point out that the very famous Lorenz phase portrait for $r = 28$ does not correctly reflect the underlying physics. For r of this magnitude their method implies the need of far more than 100 modes.

Throughout the 80s and 90s more and more Low Order Models were derived, some of which following the set of rules, introduced by Treve and Manley, some of which ignoring them. But none of them further investigated the underlying physics of the selection procedure. In the beginning 2000s a group around Alexander Gluhovsky [4] made new efforts to bring more structure into the arbitrary selection chaos. The group's research identified the Volterra gyrostat as an essential building block for deriving new Lorenz like low order models (LOM) which ensures energy conservation, as well as, other conservation laws, e. g. vorticity conservation in the case of flows with shear. This revolutionary result implies that the selection process of physically correct modes does not only follow a distinct set of rules but as a result emerges in a structure that can be reinterpreted as a mechanical model.

Very recently, a series of three consecutive papers was published by Roy and Musielak [5] [6] [7]. This is the so far most detailed research on mode selection. The dynamical system studied in this paper is taken from the first paper of this series. First of all the authors refine and expand the rules for selecting appropriate modes: In order to be a physically correct LOM the set of selected modes must

- (Rule 1) conserve energy in the dissipationless limit
- (Rule 2) have bounded solutions.

Starting from these criteria the papers investigate three different methods of adding Fourier modes to the expansion: vertical mode truncation, which corresponds to keeping m fixed and increasing n, horizontal mode truncation, which is the other way round and the combination of both. For all methods the system with the minimal number of dimensions (modes) that complies with the above rules is derived. It is worth mentioning that the original Lorenz system is always a subset of these systems. Furthermore, some basic properties of the dynamical systems are explored and the effects of the different methods on the behavior of the system on its route to chaos over varying r are compared. The paper also discusses the structure and role of mode coupling and the significance of the Z modes. These are those temperature modes with $m = 0$.

In the following those results with significance to the paper at hand shall be summarized:

The minimal system with vertical mode truncation has nine dimensions, the one with horizontal five and in the case of mixed modes eight dimensions are needed. The first and the second system have the same routes to chaos, namely via chaotic transients, this is the same route as the original Lorenz system. The 8D system having a route to chaos via quasi-periodicity is caused by the higher number of coupled modes. Finally, a dependence of the onset of chaos, i. e. for which value of $r = r_t$ the system enters chaos, on the number of Z modes is observed: the more Z modes the dynamical system has the higher the value of the critical r. The range of r_t is from 24.75 for the 3D Lorenz system to 41.54 for the 9D system.

2.3 Quantifying Dynamical Systems

We have reached the point where we converted the governing equations of the convective flow, with an infinite number of degrees of freedom (dof), into a dynamical system of odes with a (low) finite number of dofs, conserving the most important properties of the initial system. The same way a flow can be quantified by its properties like velocity, temperature and pressure, a dynamical system can be described by the properties of its stable and unstable manifolds. A key quantity is its Lyapunov Characteristic Exponent Spectrum (LCES) [10]. By just looking at the algebraic signs of LCES one can immediately classify the attractors of the system. Also, the values of the LCES can be used to determine further properties, like the dissipation rate of the system

$$D = \sum_i \lambda_i \tag{2.10}$$

the (conjectured) Kaplan-Yorke dimension of its attractor

$$d = j + \frac{\sum_{i=1}^{j} \lambda_i}{|\lambda_{j+1}|} \tag{2.11}$$

and the inherent randomness of the attractor quantified by the entropy rate of the attractor

$$h_\mu = \sum_{\lambda_i > 0} \lambda_i \tag{2.12}$$

Formally speaking, in the thermodynamic sense, the entropy describes the number of microstates a thermodynamic system can be in. More loosely, one can refer to the entropy as the degree of disorder a system is in. In a similar way the information theoretic entropy rate, in this case calculated by the positive Lyapunov exponents of the dynamical system, describes the irreducible rate of randomness produced by its source [9], i. e. loosely speaking it measures the rate of uncertainty produced by the attractor. Thus, the attractor is stable and its entropy is zero, as the behavior is fully predictable: eventually all trajectories are drawn towards the attractor. In this case all of its Lyapunov exponents are either zero or negative. But as soon as the attractor becomes strange or chaotic and the outcome is no longer predictable, even sensitively depending on the initial conditions, the entropy rate becomes a positive number, as at least one of its Lyapunov exponents is positive.

The analogies to the behavior of fluid flows are evident: Once can identify the three different flow regimes: the stable laminar flow, the unstable transient phase and the fully chaotic and (according to the current state of knowledge) random turbulent regime. With this in mind the (information) entropy rate of an attractor could very well serve as a measure and predictor for turbulence! And in addition to just characterizing a flow as laminar or turbulent it would even be possible to quantify the degree of turbulence of a flow.

3 Dynamical Systems Toolbox

So far we have gathered the necessary background to identify a possible candidate for measuring turbulence. The following chapters will try to find more evidence for this correlation and outline some ideas that could support this assumption. As part of this project a Matlab toolbox was developed that supports all of the above mentioned steps, hereafter referred to as “Dynamical Systems Toolbox” (DST): The source code of each of the tools can be found in the appendix.

The first tool “modeval”, which still is in an early beta stadium, has the purpose to assist the derivation of the odes of the dynamical system. After entering the desired dimension and modes of the system, it was supposed to automatically derive the equations of the dynamic system and put them into a for Matlab readable form. Unfortunately, the programming reached the maximum capabilities of Matlab and thus what remains is a program that only derives the equations in their untransformed form. The current state is intended for a system in 6D but extending the program to an arbitrary number of dimensions is possible. The tool was verified by deriving the original Lorenz 3D system.

Thus, after using “modeval” some manual work is necessary to first transform the equations and then convert them into a function handle file for Matlab. The second tool “tangentspace” then calculates the tangent space of the dynamical system. This is needed for the calculation of the Lyapunov exponents.

The third tool “Lorenz9analysis” is the core piece of the toolbox. It covers the whole analysis and calculation part. It is intended to for use with an arbitrary number of dimensions, but some minor modifications are still necessary in order to adjust it to the desired number of dimensions. The current state is intended for a 9D system.

In a first step the tool “Loren9sim” simulates the dynamical system using the implemented Runge-Kutta method (ode45) and displays the phase portraits and time series of the dynamical system. It also calculates Poincaré Return Maps of the Z modes of the system.

The second step “Lorenz9lyap” calculates the Lyapunov exponents of the dynamical system using the Gram-Schmidt-Renormalization algorithm (GSRA) [1]. To verify the correct implementation the tool was tested for two cases: firstly, with a linear dynamical system, where the Lyapunov exponents can be calculated as the eigenvalues of

$$\frac{1}{2}\left(Jacobian + Jacobian^{T}\right) \tag{3.1}$$

The Lyapunov exponents could be recovered with an accuracy of less than 0.2%. The second test was the successful detection of the zero Lyapunov exponent in a two dimensional dynamic system with a limit cycle.

Fig. 3.1 shows that the three leading Lyapunov exponents calculated in [5] can be recovered by the toolbox. In this case the Lyapunov exponent was calculated with the natural logarithm instead of the logarithm to the base two.

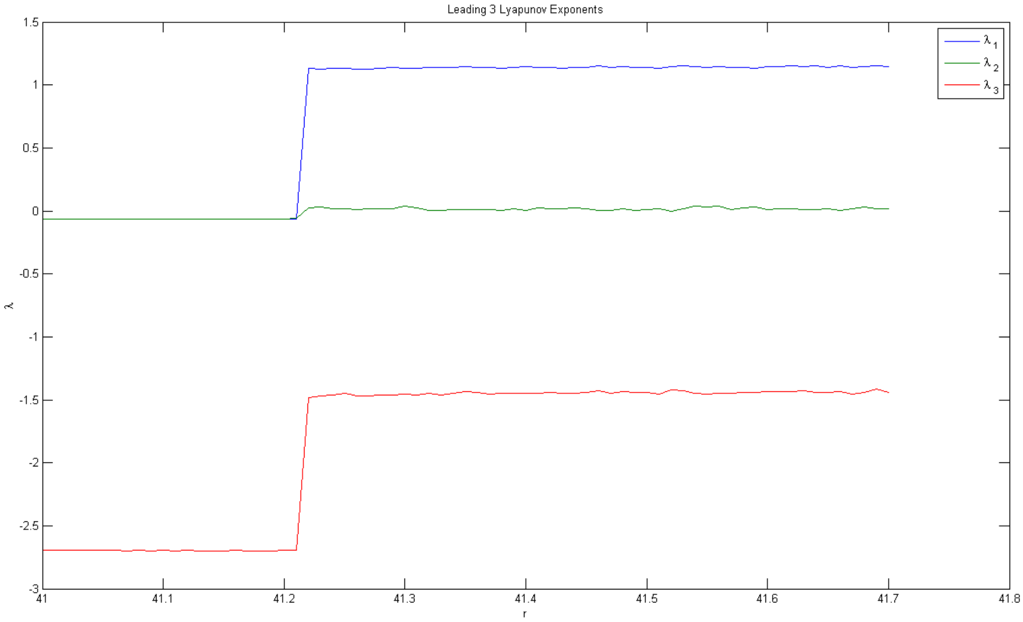

Figure 3.1: Test calculation for verifying the implemented algorithm

The third step, called “Lorenz9lyan”, finally calculates the dissipation rate of the system, the dimension of the attractor and its entropy rate.

In a final (optional) step “fieldsym” the resulting velocity and temperature field are simulated.

4 Results and Conclusions

4.1 9D Lorenz System

To bring forward some evidence towards the project's leading assumption the 9D system, derived in [5] by the vertical truncation method, was selected to be explored by the DST. Two main reasons lead to the selection of this system: as mentioned in [5] the vertical truncation method should be the preferred one, as the fluid motion is dominant in the vertical direction. Secondly, this system has the largest number of Z modes. The results of all three papers indicate that especially the Z modes influence the onset of chaos in the dynamical system and thus, might store some valuable amount of information about the turbulent behavior of the flow. Further analysis of these particular modes is therefore interesting. These are the equations of the studied system:

$$\frac{dX}{dt} = -\sigma X + \sigma Y \tag{4.1}$$

$$\frac{dY}{dt} = -XZ + rX - Y + ZX_2 - 2X_2 Z_1 \tag{4.2}$$

$$\frac{dZ}{dt} = XY - bZX_2 Y - XY_2 \tag{4.3}$$

$$\frac{dX_1}{dt} = -c_1 \sigma X_1 + \frac{\sigma}{c_1} Y_1 \tag{4.4}$$

$$\frac{dY_1}{dt} = -2X_1 Z_1 + rX_1 - c_1 Y_1 \tag{4.5}$$

$$\frac{dZ_1}{dt} = 2XY_2 + 2YX_2 + 2X_1 Y_1 - 4bZ_1 \tag{4.6}$$

$$\frac{dX_2}{dt} = -\sigma c_2 X_2 + \frac{\sigma}{c_2} Y_2 \tag{4.7}$$

$$\frac{dY_2}{dt} = XZ + rX_2 - c_2Y_2 - 2XZ_1 - 3X_2Z_2 \tag{4.8}$$

$$\frac{dZ_2}{dt} = 3X_2Y_2 - 9bZ_2 \tag{4.9}$$

First the resulting phase portraits, in the Lorenz state space X, Y, Z (Fig. 4.1 and Fig. 4.2) and the X_2, Y_2, Z_2 space (Fig. 4.3 and Fig. 4.4), and the time series (Fig. 4.5 and Fig. 4.6) of each of the amplitudes are displayed for some characteristic values of the heating rate r. The X_1, Y_1, Z_1 is not presented as the result is meaningless.

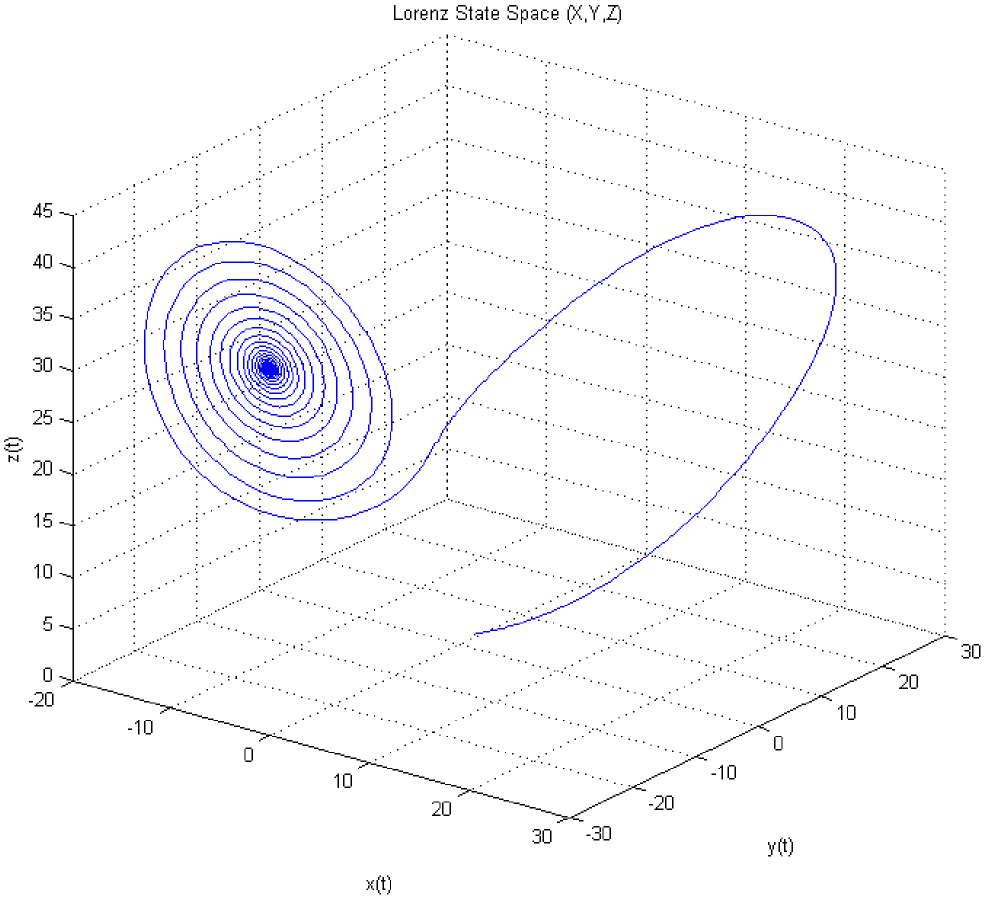

Figure 4.1: Phase portrait in XYZ for $r = 28$

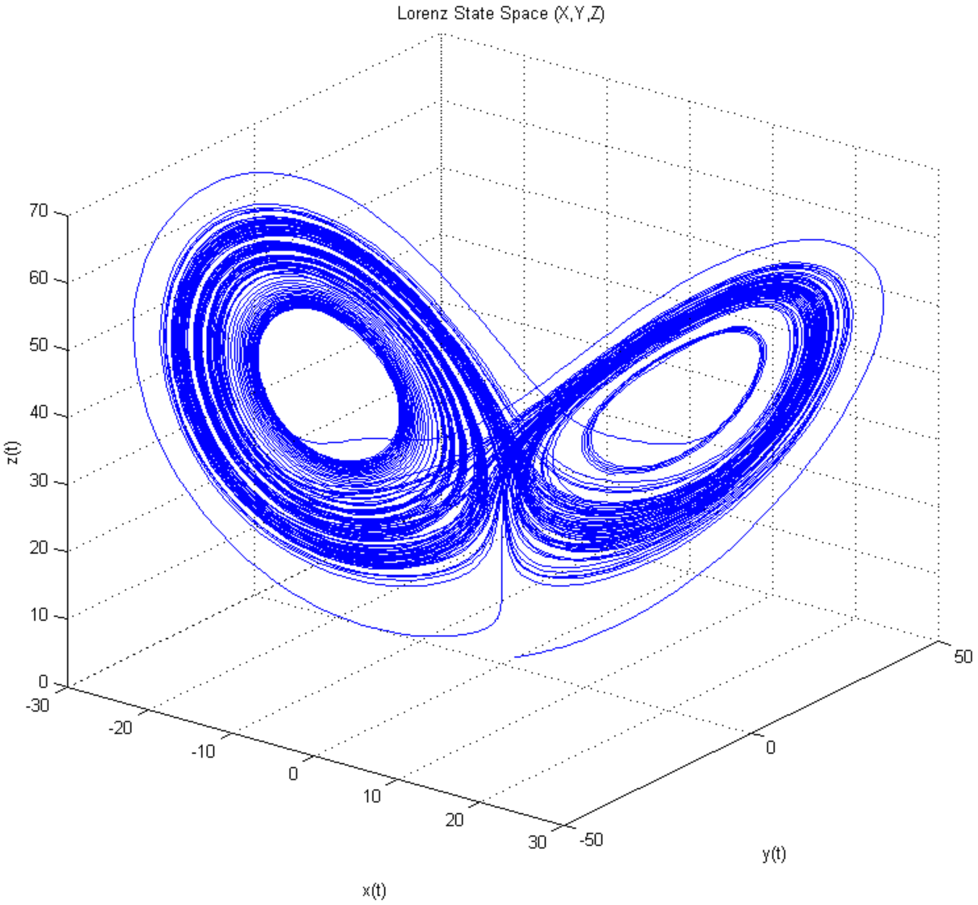

Figure 4.2: Phase portrait in XYZ for $r = 42$

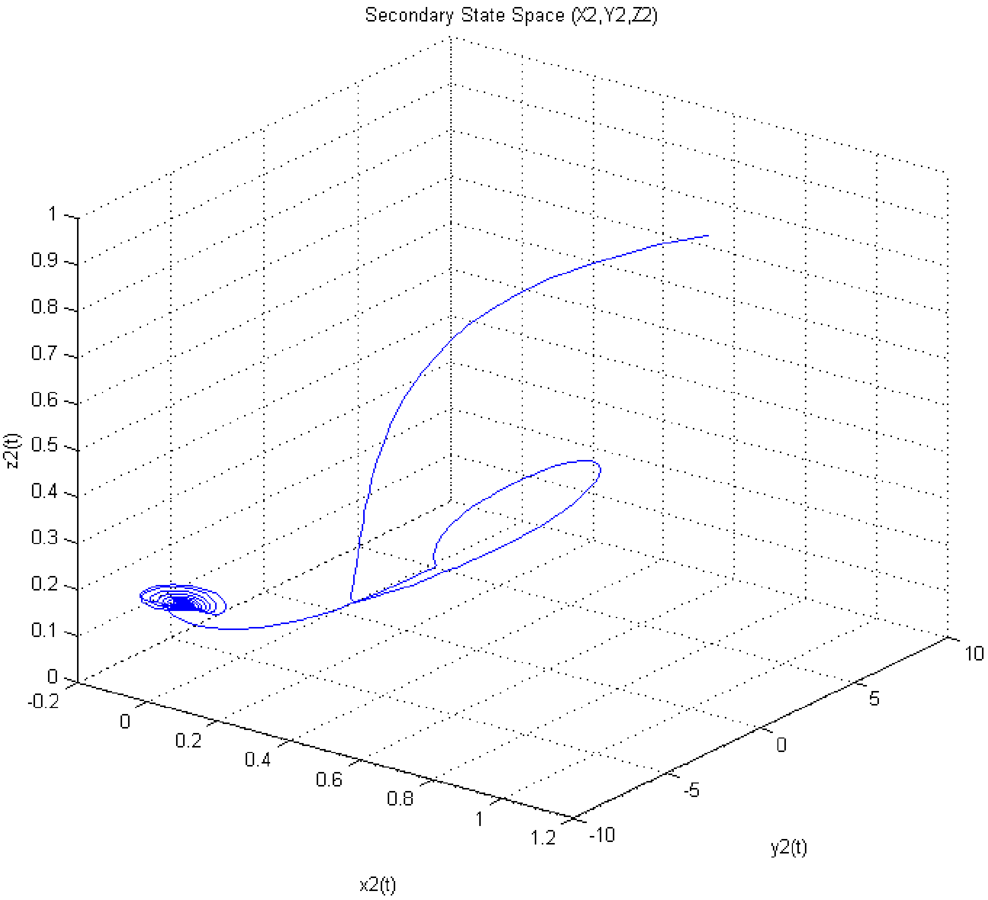

Figure 4.3: Phase portrait in $X_2Y_2Z_2$ for $r = 28$

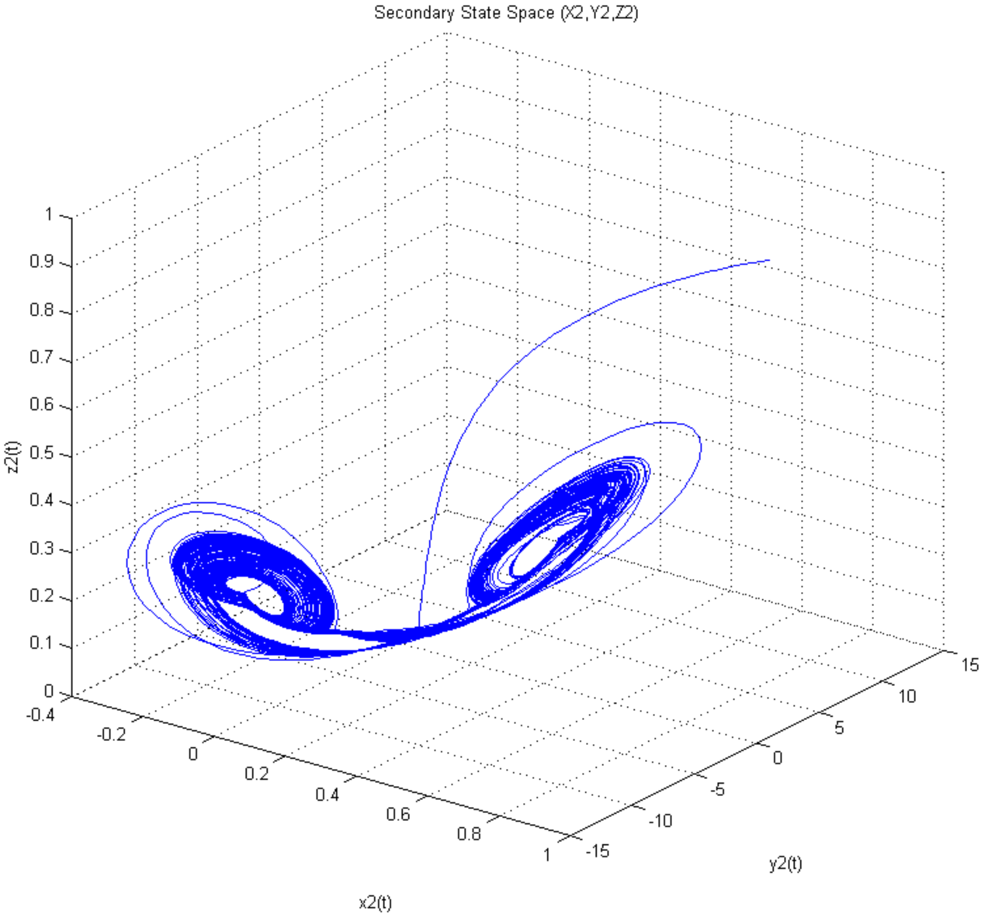

Figure 4.4: Phase portrait in $X_2Y_2Z_2$ for r=42

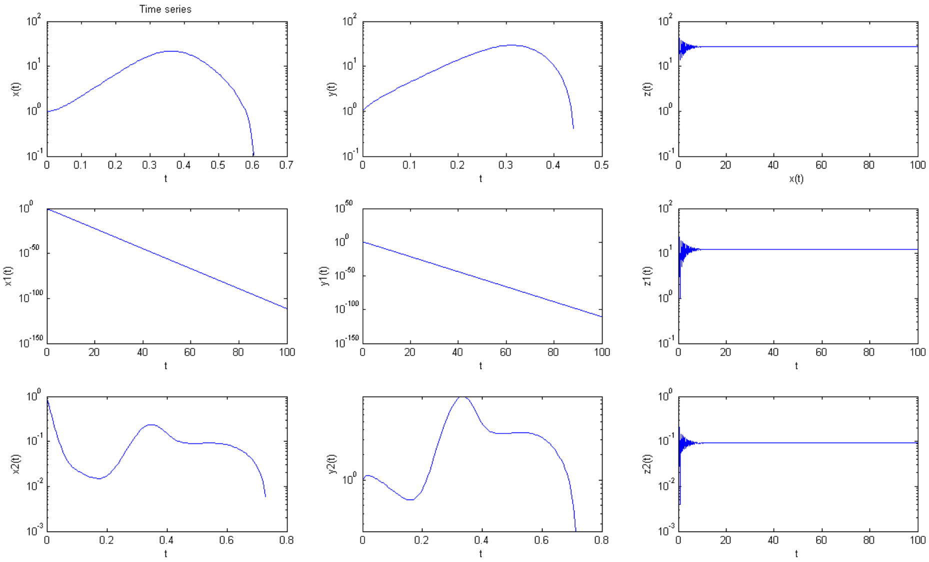

Figure 4.5: Time series of the amplitudes for $r = 28$

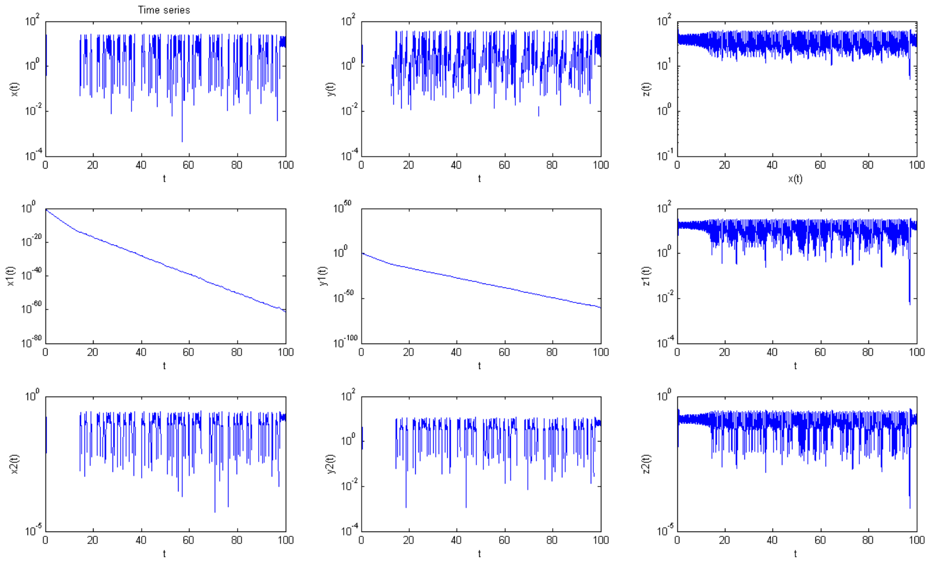

Figure 4.6: Time series of the amplitudes for $r = 42$

4.2 Lyapunov Exponents and Information Theoretic Quantities

Next the Lyapunov exponents for different values of r and the resulting quantities dissipation rate, dimension of the attractor and entropy of the attractor are presented.

Some remarks have to be made about the quality of the plots: The plots of the Lyapunov exponent and the resulting quantities imply a smoothness of the curves, which is not in agreement with the true behavior. The reason for this is the rather low r resolution of the plots. All computations were carried out on an Intel Core 2 Duo 2.54 GHz processor, with which the necessary accuracy for generating high quality plots could not be reached within a reasonable amount of time. Some of the more accurate plots presented in this paper already had a computation time of over 12 hours.

In Fig. 4.7 the transition from the stable region to the transient (unstable) region can be observed where the leading Lyapunov exponent touches the x-axis, and thus indicating a bifurcation of the formerly stable origin.

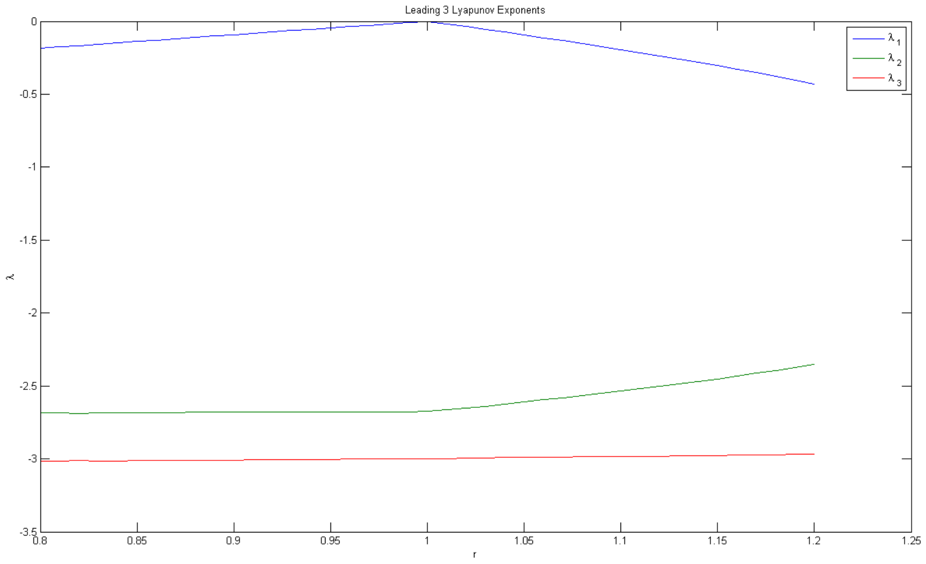

Figure 4.7: Leading 3 Lyapunov exponents for $0.8 < r < 1.2$, resolution: $\Delta r = 0.1$

Fig. 4.10 confirms the behavior of h_μ discussed in chapter 2.3. h_μ reflects the onset of chaos at $r = 41.54$. Right after this onset the entropy rate, at first, rapidly increases to $1.672\frac{bits}{second}$ and then grows a lot slower as r is further increased. From the perspective of the flow this can be explained with the fact that the flow does not abruptly change from an unstable to a turbulent behavior. While very steep region reflects the last part of the transition phase where the flow is not yet fully turbulent, the sharp bend at $r = 42$ (Fig. 4.11) indicates the beginning of the fully turbulent regime.

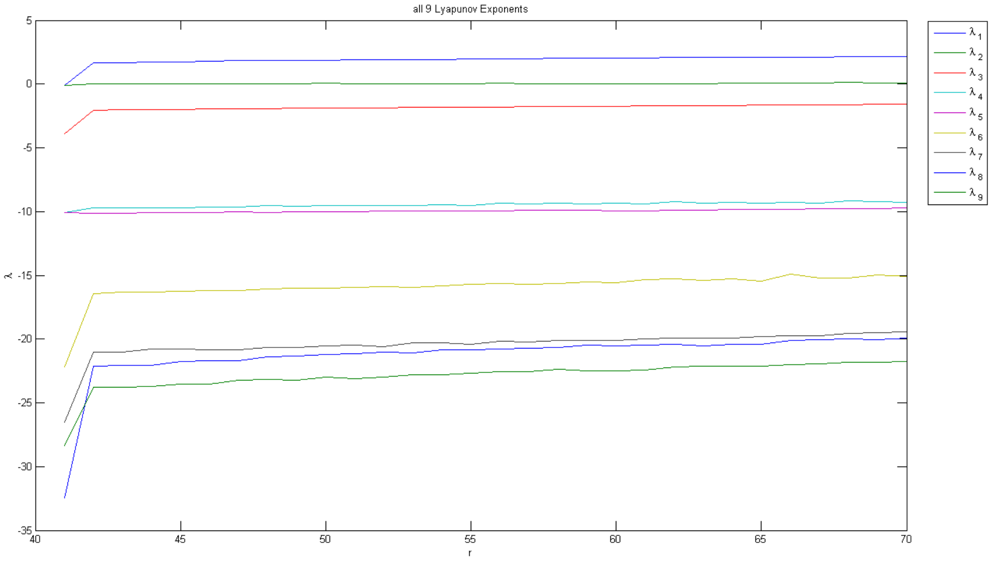

Figure 4.8: Evolution of the LCE spectrum for $42 < r < 70$, resolution: $\Delta r = 1$

For the explanation of the second, flat part of the curve some terminology from information theory is needed that was not introduced in this paper. For an explanation the reader is referred to [2]. Assuming it is possible to identify structures in a turbulent flow that can be interpreted as causal states and thus, the dynamics of turbulence can be reduced to the transition from one causal state of an ϵ-machine to another, this behavior implies that with increasing r more of these structures develop and the flow constantly produces more uncertainty about the state it is in.

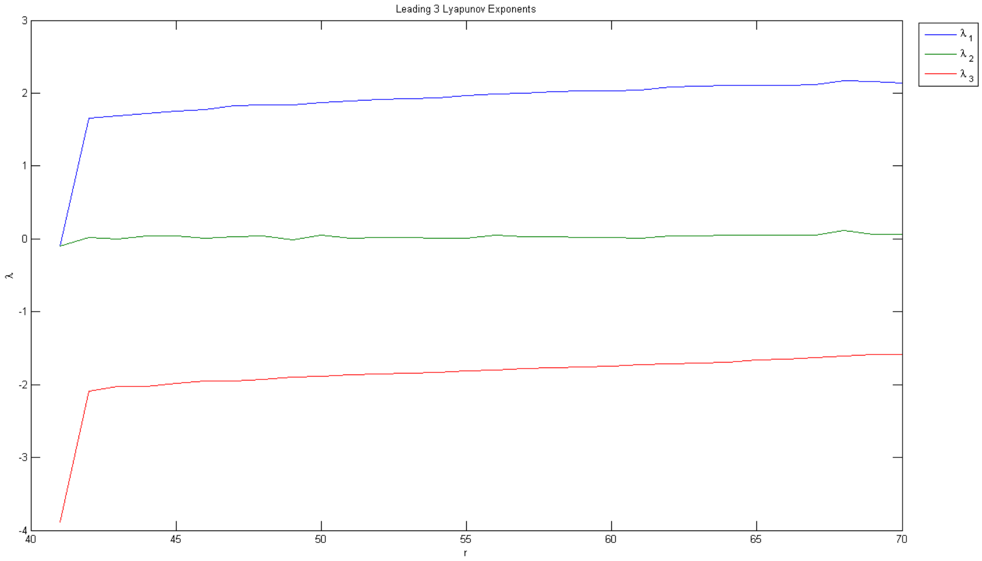

Figure 4.9: Evolution of the 3 leading Lyapunov exponents for $42 < r < 70$, resolution: $\Delta r = 1$

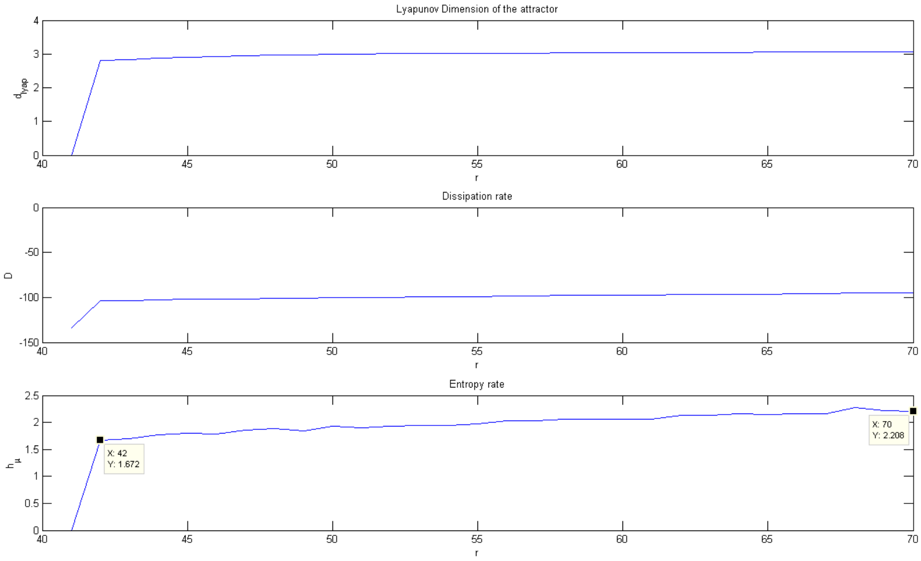

Figure 4.10: Dissipation rate, dimension and Entropy rate for $42 < r < 70$, resolution: $\Delta r = 1$

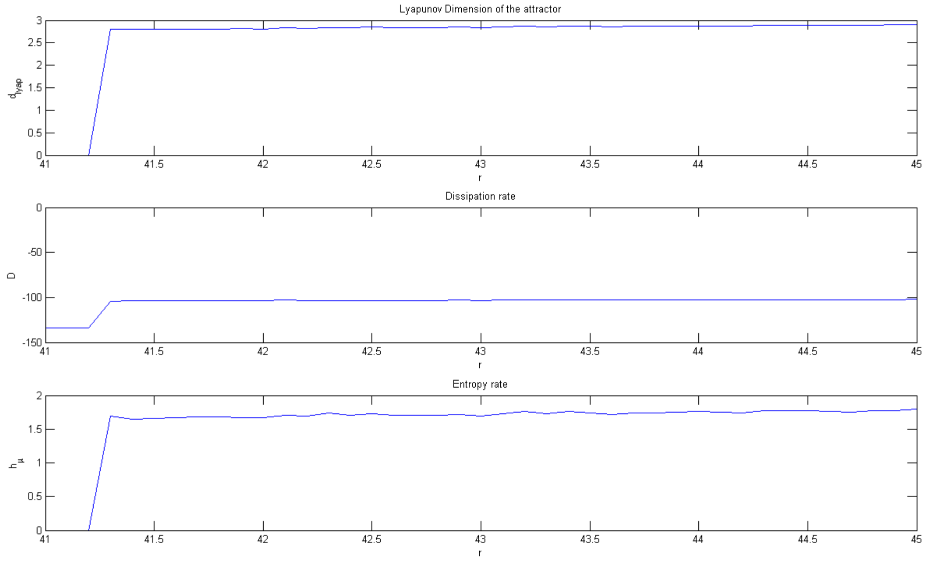

Figure 4.11: Dissipation rate, dimension and Entropy rate for $41 < r < 45$, resolution: $\Delta r = 0.1$

4.3 Poincaré Return Maps

Lastly, without further discussion, the calculated Z mode Poincaré Return Maps (again Z Fig. 4.12 and Z_2 Fig. 4.13 only) for $r = 42$ are presented.

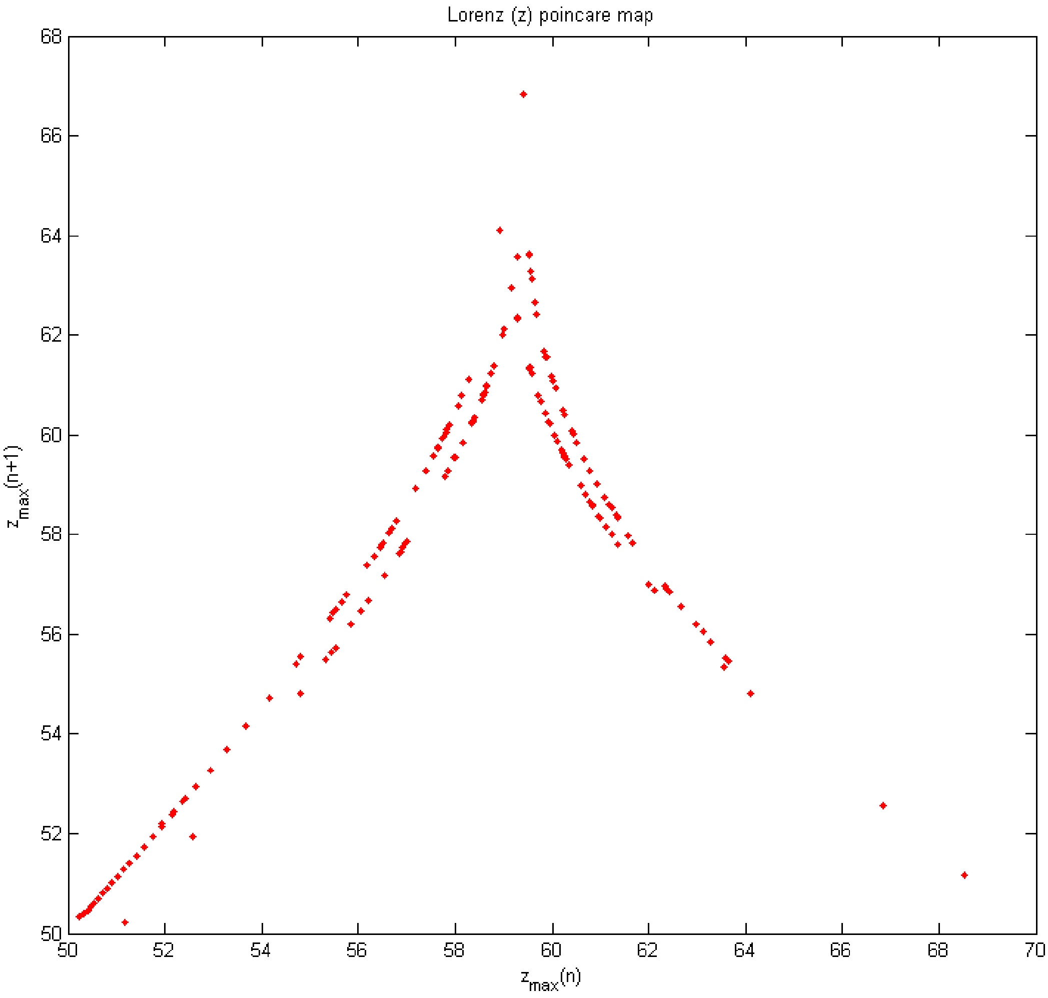

Figure 4.12: Z Poincaré Return Map for $r = 42$

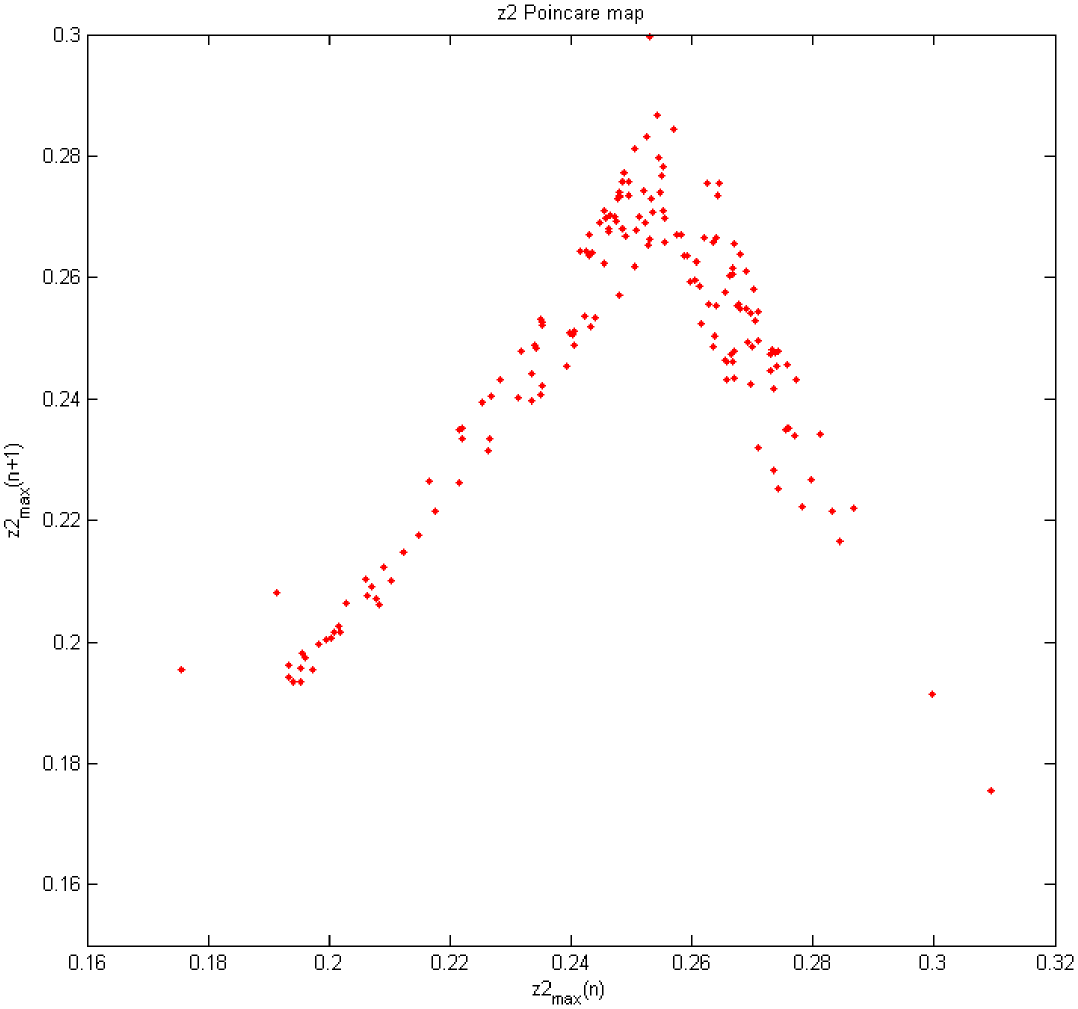

Figure 4.13: Z_2 Poincaré Return Map for $r = 42$

5 Further Work

The next necessary step in order to find further proof for the stated assumption, that the information theoretic entropy rate h_μ is an appropriate measure for the turbulent behavior of a flow, is to verify that the predicted value of r for the onset of chaos indeed corresponds to a Rayleigh number where the flow becomes turbulent. This can be done by comparing the results achieved by the presented method to numerical and experimental data, e. g. the studies carried out by Busse et al. [3]. The data provided in [7] suggests that the marginal value for r does not tend to infinity for an increasing number of Z modes, but converges. Combined with the results of Treve and Manley [11] it seems necessary to derive systems with a higher number of modes. Further investigations should find an optimal number of modes that best describe the flow at the onset of turbulence. A reasonable trade-off has to be made between computational effort and accuracy of the approximation. If this first step results in a positive answer, the next step should be the analysis of the Z modes, as these seem to contain a fair amount of information about the turbulent behavior [7]. It should be possible to conduct an ϵ-machine reconstruction from the Poincaré Maps presented in this paper. These ϵ-machines could then be used to retrieve causal states (in all dimensions) from the flow. If this procedure was successful the consequences for the understanding of turbulent flows are unpredictable.

Appendix

Sourcecode

Lorenz9analysis.m

```
function Lorenz9_analysis(begin,vid,rmin,rmax,step)
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% This tool analyzes the 9D Lorenz system in the range of            %
% r_min < r < r_max                                                  %
% Input variables:                                                   %
% begin     -    # of step to begin with                             %
% vid       -    animate flow and temperature field? (0 - No,1 - Yes)%
% rmin      -    minimum r                                           %
% rmax      -    maximum r                                           %
% step      -    stepsize between r                                  %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% clear head                                                         %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
clc
close all

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% global declaration                                                 %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

dim = 9;

if begin == 1
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% The first step simulates the System                                %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% declaration                                                        %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% initial condition
    x0 = [1 1 1 1 1 1 1 1 1];

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% simulation call (initial_condition, r, t_end)                      %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
    for r = rmin:2:rmax
        [tv, vv] = Lorenz9_sim(x0,r,100);
    end

 end

if begin <= 2
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% The second step calculates the lyapunov exponents                  %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% declaration                                                        %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    i = 0;
    nrs = (rmax-rmin)/step+1;
    lyap = zeros(dim,nrs);

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% lyapunov call (Dimension, # Iterations, r, Trans. Phase, t_step)   %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
  disp('Running Lyapunov calculations')
    for r=rmin:step:rmax
        i = i+1;
        perc = i/nrs*100;
        fprintf('Currently at %3.1f %% of calculation \n',perc)
        tic
        lyap(:,i) = Lorenz9_lyap(dim,1000,r,50,1);
        toc
    end

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% visualization                                                      %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    r=rmin:step:rmax;

%     figure;
%     plot(r,lyap);
%     title('all 9 Lyapunov Exponents');
%     xlabel('r');
%     ylabel('\lambda');
%     legend('\lambda _1','\lambda _2','\lambda _3', ...
      '\lambda _4', '\lambda _5', '\lambda _6', ...
%         '\lambda _7', '\lambda _8', '\lambda _9');

    figure;
    plot(r,lyap(1:3,:));
    title('Leading 3 Lyapunov Exponents');
    xlabel('r');
    ylabel('\lambda');
    legend('\lambda _1','\lambda _2','\lambda _3');

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% The third step analyzes the lyapunov exponents                     %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% analysis call (Lyapunov Matrix, Dimension)                         %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    [diss, dlyap h_mu] = Lorenz9_lyan(lyap,dim);

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% visualization                                                      %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    figure;
    subplot(3,1,1);
    plot(r,dlyap);
    title('Lyapunov Dimension of the attractor');
    xlabel('r');
    ylabel('d_{lyap}');

    subplot(3,1,2);
    plot(r,diss);
    title('Dissipation rate');
    xlabel('r');
    ylabel('D');

    subplot(3,1,3);
    plot(r,h_mu);
    title('Entropy rate');
    xlabel('r');
    ylabel('h_{\mu}');
end

if vid == 1
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Simulation of Flow and Temperature Field (v) for rmax              %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
         field_sim(tv,vv,dim)
end
```

Lorenz9sim.m

```
function [t, v] = Lorenz9_sim(x0,r,t_end)
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Numerical integration and visualization of the Lorenz system in 9D %
% Input variables:                                                   %
% x0           -              initial condition                      %
% r            -              Rayleigh parameter                     %
% t_transient  -              Length of Integration                  %
% By Franz Bozsak                                                    %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% declaration                                                        %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% Model parameters
sigma  = 10;
b      = 8/3;
a      = 4/b-1;          % this is a^2
c1     = (4+a)/(1+a);
c2     = (9+a)/(1+a);

i = 1;

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Integration using a Runge-Kutta Method                             %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

[t, v] = ode45('lorenz9ode', t_end, [x0 x0], [], sigma, r, b,c1,c2);

% Split out the trajectory matrix into x,y,z,x1,y1,z1,x2,y2,z2 vector variables
x = v(:, 1);
y = v(:, 2);
z = v(:, 3);
x1 = v(:, 4);
y1 = v(:, 5);
z1 = v(:, 6);
x2 = v(:, 7);
y2 = v(:, 8);
z2 = v(:, 9);
```

```
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Visualization                                                      %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% Plot trajectory in 3x3D state-space
figure;
plot3(x, y, z, 'b-');
title('Lorenz State Space (X,Y,Z)');
xlabel('x(t)');
ylabel('y(t)');
zlabel('z(t)');
grid on;
rotate3d on;
view([37,26]);

% figure;
% plot3(x1, y1, z1, 'b-');
% title('Secondary State Space (X1,Y1,Z1)');
% xlabel('x1(t)');
% ylabel('y1(t)');
% zlabel('z1(t)');
% grid on;
% rotate3d on;
% view([37,26]);

figure;
plot3(x2, y2, z2, 'b-');
title('Secondary State Space (X2,Y2,Z2)');
xlabel('x2(t)');
ylabel('y2(t)');
zlabel('z2(t)');
grid on;
rotate3d on;
view([37,26]);

% Display time series plots
figure;
subplot(3,3,1);
semilogy(t, x, 'b-');
title('Time series');
xlabel('t');
ylabel('x(t)');
```

```
%
subplot(3,3,2);
semilogy(t, y, 'b-');
xlabel('t');
ylabel('y(t)');
%
subplot(3,3,3);
semilogy(t, z, 'b-');
xlabel('x(t)');
ylabel('z(t)');
%
subplot(3,3,4);
semilogy(t, x1, 'b-');
xlabel('t');
ylabel('x1(t)');
%
subplot(3,3,5);
semilogy(t, y1, 'b-');
xlabel('t');
ylabel('y1(t)');
%
subplot(3,3,6);
semilogy(t, z1, 'b-');
xlabel('t');
ylabel('z1(t)');
%
subplot(3,3,7);
semilogy(t, x2, 'b-');
xlabel('t');
ylabel('x2(t)');
%
subplot(3,3,8);
semilogy(t, y2, 'b-');
xlabel('t');
ylabel('y2(t)');
%
subplot(3,3,9);
semilogy(t, z2, 'b-');
xlabel('t');
ylabel('z2(t)');

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Poincare Maps                                                      %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% Now find next maxima to create next-maximum map of z(t)
max_points = find((z(2:end-1) > z(3:end)) & (z(2:end-1) > z(1:end-2))) + 1;
z_max      = z(max_points);
t_max      = t(max_points);

% Plot Poincare map
figure;
plot(z_max(1:end-1), z_max(2:end), ’r.’);
title(’Lorenz (z) poincare map’);
xlabel(’z_{max}(n)’);
ylabel(’z_{max}(n+1)’);

% Now find next maxima to create next-maximum map of z2(t)
max2_points = find((z2(2:end-1) > z2(3:end)) & (z2(2:end-1) > z2(1:end-2))) + 1;
z2_max      = z2(max_points);
t2_max      = t(max_points);

% Plot Poincare map
figure;
plot(z2_max(1:end-1), z2_max(2:end), ’r.’);
title(’z2 Poincare map’);
xlabel(’z2_{max}(n)’);
ylabel(’z2_{max}(n+1)’);
```

Lorenz9lyap.m

```
function lyap = Lorenz9_lyap(dim,numit,r,t_transient,t_step)
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% This tool calculates the Lyapunov exponent of the Lorenz 9D        %
% Input variables:                                                   %
% dim          -              Dimension of the system                %
% numit        -              Number of Iterations                   %
% r            -              Rayleigh parameter                     %
% t_transient  -              Length of Transient Phase              %
% t_step       -              Length of each Iteration               %
% By Franz Bozsak                                                    %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% declaration                                                        %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% create identity matrix
evec  = eye(dim);
lamda = zeros(dim);
% lyap  = zeros(dim);

% Model parameters
sigma  = 10;
b      = 8/3;
a      = 4/b-1;          % this is a^2
c1     = (4+a)/(1+a);
c2     = (9+a)/(1+a);

% Initial conditions
x0     = [1 1 1 1 1 1 1 1 1];

% declare options for the integration
options = odeset('MaxOrder',4);

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Transient phase                                                    %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

for i=1:dim
    % Integrate the system using Runge-Kutta
    [tt, v] = ode45('lorenz9ode', t_transient, [x0 evec(i,:)], ...
              options, sigma, r, b, c1, c2);
    % new direction vectors after transient phase
    evec(i,:) = v(length(v),(dim+1):(2*dim));
end

for i=1:dim
    if i == 1
        abv = sqrt(evec(i,:)*evec(i,:)');
        evec(i,:) = evec(i,:)/abv;
    else
        for k=1:i-1
            d=evec(i,:)*evec(k,:)';
            evec(i,:) = evec(i,:)-d*evec(k,:);
        end
        abv = sqrt(evec(i,:)*evec(i,:)');
        evec(i,:)= evec(i,:)/abv;
    end
end

% new x0 after transient phase
x0 = v(length(v),1:dim);

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Determine the LCE spectrum                                         %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

for j=1:numit
    for i=1:dim
        % Integrate the system using Runge-Kutta
        [tr, v] = ode45('lorenz9ode', t_step, [x0 evec(i,:)], ...
                  options, sigma, r, b, c1, c2);
        evec(i,:) = v(length(v),(dim+1):(2*dim));
    end

    for i=1:dim
        if i == 1
            abv = sqrt(evec(i,:)*evec(i,:)');
            evec(i,:) = evec(i,:)/abv;
            lamda(i) = lamda(i) + log(abv);
        else
            for k=1:i-1
```

```
                d=evec(i,:)*evec(k,:)';
                evec(i,:) = evec(i,:)-d*evec(k,:);
            end
            abv = sqrt(evec(i,:)*evec(i,:)');
            evec(i,:)= evec(i,:)/abv;
            lamda(i) = lamda(i) + log(abv);
        end
    end
    x0 = v(length(v),1:dim);
end

% average lamda

for i=1:dim
    lyap(i) = lamda(i)/(numit*t_step);
end
```

Lorenz9lyan.m

```
function [diss, dlyap h_mu] = Lorenz9_lyan(lyap,dim)
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% This tool calculates the Lyapunov exponent of the Lorenz 9D        %
% Input variables:                                                   %
% dim          -              Dimension of the system                %
% lyap         -              Lyapunov matrix                        %
% By Franz Bozsak                                                    %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% declaration
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

kmax  = length(lyap(1,:));
dlyap = zeros(kmax);
diss  = zeros(kmax);
h_mu  = zeros(kmax);

for k=1:kmax
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Calculate the dissipation rate                                     %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    diss(k) = sum(lyap(:,k));

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Calculate the entropy rate                                         %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    for j=1:dim
        if lyap(j,k) > 0
            h_mu(k) = h_mu(k) + lyap(j,k);
        end
    end

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% Calculate the lyapunov dimesion of the attractor                   %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %

    if lyap(1,k) < 1
```

```
        dlyap(k) = 0;
    elseif (diss(k) > 0)
        dlyap(k) = dim + diss(k);
    else
        lysum = lyap(1,k);
        i = 2;
        while (lysum+lyap(i,k)) >= 0
            lysum = lysum + lyap(i,k);
            i = i + 1;
        end
        dlyap(k) = (i-1) + lysum/abs(lyap(i,k));
    end
end
```

fieldsim.m

```
function field_sim(t,v,dim)
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% This tool simulates the resluting Flow and Temperature field       %
% Input variables:                                                   %
% v     -       field of amplitudes                                  %
% t     -       corresponding time vector                            %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
syms x z
q = pi/sqrt(2);
Ra = 27*pi^4/4;
m = [1 1 0 1 1 0 1 1 0];
n = [1 1 2 2 2 4 3 3 6];
for i=1:length(t)
    psi = v(i,(1))*sin(m(1)*q*x)*cos(pi*(2*n(1)+1)*z)+...
        v(i,(4))*sin(m(4)*q*x)*cos(pi*(2*n(4)+1)*z)+...
        v(i,(7))*sin(m(7)*q*x)*cos(pi*(2*n(7)+1)*z);
    theta = v(i,2)*cos(m(2)*q*x)*cos(pi*(2*n(2)+1)*z)-1 + ...
        v(i,(3))*cos(m(3)*q*x)*sin(2*pi*n(3)*z)+ ...
        v(i,5)*cos(m(5)*q*x)*cos(pi*(2*n(5)+1)*z)-1 + ...
        v(i,(6))*cos(m(6)*q*x)*sin(2*pi*n(6)*z)+ ...
        v(i,8)*cos(m(8)*q*x)*cos(pi*(2*n(8)+1)*z)-1 + ...
        v(i,(9))*cos(m(9)*q*x)*sin(2*pi*n(9)*z);
    figure(30), clf
    subplot(2,1,1);
    hold on
    ezplot(psi-1, [0 pi/q -1/2 1/2]);
    ezplot(psi-1.5, [0 pi/q -1/2 1/2]);
    ezplot(psi-2, [0 pi/q -1/2 1/2]);
    title('Velocity field');
    xlabel('x');
    ylabel('z');
    hold off

    subplot(2,1,2);
    hold on
    ezplot(theta-1, [0 pi/q -1/2 1/2]);
    ezplot(theta-1.5, [0 pi/q -1/2 1/2]);
    ezplot(theta-2, [0 pi/q -1/2 1/2]);
    title('Temperature field');
    xlabel('x');
    ylabel('z');
    hold off
end
```

lorenz9ode.m

```
function Y = lorenz9ode(t,x,flag,sigma,r,b,c1,c2)
Y = [-sigma*x(1)+sigma*x(2); ...
    -x(1)*x(3)+r*x(1)-x(2)+x(3)*x(7)-2*x(7)*x(6); ...
     x(1)*x(2)-b*x(3)-x(7)*x(2)-x(1)*x(8); ...
    -c1*sigma*x(4)+sigma/c1*x(5); -2*x(4)*x(6)+r*x(4)-c1*x(5); ...
    2*x(1)*x(8)+2*x(2)*x(7)+2*x(4)*x(5)-4*b*x(6); ...
    -sigma*c2*x(7)+sigma/c2*x(8); ...
    x(1)*x(3)+r*x(7)-c2*x(8)-2*x(1)*x(6)-3*x(7)*x(9); ...
    3*x(7)*x(8)-9*b*x(9); ...
    x(11)*sigma - x(10)*sigma; ...
    x(10)*(r-x(3))-2*x(15)*x(7)-x(11)-x(12)*(x(1)-x(7))+x(16)*(x(3)-2*x(6)); ...
    x(11)*(x(1)-x(7))-x(17)*x(1)-x(16)*x(2)-b*x(12)+x(10)*(x(2)-x(8)); ...
    (x(14)*sigma)/c1 - c1*x(13)*sigma; ...
    x(13)*(r - 2*x(6)) - 2*x(15)*x(4) - c1*x(14);
    2*x(17)*x(1) - 4*b*x(15) + 2*x(16)*x(2) + 2*x(14)*x(4) + 2*x(13)*x(5) + ...
    2*x(11)*x(7) + 2*x(10)*x(8); ...
    (x(17)*sigma)/c2 - c2*x(16)*sigma;
    x(12)*x(1) - c2*x(17) - 2*x(15)*x(1) - 3*x(18)*x(7) + x(16)* ...
    (r - 3*x(9)) + x(10)*(x(3) - 2*x(6)); ...
    3*x(17)*x(7) - 9*b*x(18) + 3*x(16)*x(8)];
```

modeval.m

```
function g = modeval()
% This tool derives the odes of the dynamical system
% emerging from the governing equations of the Rayleigh-Benard convection
% written for the 6D case!
% Maple engine required!
clc
clear all
close all
% Declaration of variables

syms psif psi thetaf thetaa thetab x z q t momentum energy
syms u w lap2 lap4 sigma Ra izmom ixmom ixenA izenA ixenB izenB eq
syms psi1 psi2 x Xd y Yd z Zd u Ud v Vd w Wd r
% Declaration of structure

% How many modes per amplitude are to be considered?
num_mod = 2;
% Apart from the Lorenz modes which modes are to be considered?
psi_mod = [1 0; 2 0];
thetaa_mod = [1 0; 2 0];
thetab_mod = [0 1;0 2];
% q = sym('sqrt(1/2)*pi');
psi = sym(zeros(1, num_mod));
thetaa = sym(zeros(1, num_mod));
thetab = sym(zeros(1, num_mod));

for k=1:num_mod
  psi(k) = sym(sprintf('psi%d(t)', k));
  thetaa(k) = sym(sprintf('thetaA%d(t)', k));
  thetab(k) = sym(sprintf('thetaB%d(t)', k));
end

for k=1:3*num_mod
    eq(k) = sym(sprintf('eq%d', k));
end

psif = 0;
thetaf = 0;

% define psi and theta
```

```
for i=1:num_mod
    psif = psif + psi(i)*sin(psi_mod(i,1)*q*x)* ...
         cos(pi*(2*psi_mod(i,2)+1)*z);
    thetaf = thetaf + thetaa(i)*cos(thetaa_mod(i,1)*q*x)* ...
         cos(pi*(2*thetaa_mod(i,2)+1)*z) + ...
         thetab(i)*cos(thetab_mod(i,1)*q*x)*...
         sin(2*pi*thetab_mod(i,2)*z);
end

u = -diff(psif,z);
w = diff(psif,x);
lapp2 = diff(w,x)-diff(u,z);
lapp4 = diff(lapp2,z,2) + diff(lapp2,x,2);
lapt2 = diff(thetaf,x,2) + diff(thetaf,z,2);

% define Momentum Balance and Energy Balance
momentum = 1/sigma*(-diff(lapp2,t)+...
           diff(u*diff(u,x)+w*diff(u,z),z)- ...
           diff(u*diff(w,x)+w*diff(w,z),x))+ ...
           lapp4+diff(thetaf,x);
energy = diff(thetaf,t)+u*diff(thetaf,x)+ ...
         w*diff(thetaf,z)-lapt2 - Ra*diff(psif,x);

% integrate
shift = 0;
for i=1:num_mod
    izmom = int(momentum*sin(psi_mod(i,1)*q*x)* ...
    cos(pi*(2*psi_mod(i,2)+1)*z),z,-1/2,1/2);
    ixmom = int(izmom,x,0,pi/q);
    eq(1+shift) = ixmom;
    izenA = int(energy*cos(thetaa_mod(i,1)*q*x)* ...
    cos(pi*(2*thetaa_mod(i,2)+1)*z),z,-1/2,1/2);
    ixenA = int(izenA,x,0,pi/q);
    eq(2+shift) = ixenA;
    izenB = int(energy*cos(thetab_mod(i,1)*q*x)* ...
    sin(2*pi*thetab_mod(i,2)*z),z,-1/2,1/2);
    ixenB = int(izenB,x,0,pi/q);
    eq(3+shift) = ixenB;
    shift = 3;
end
equ1=subs(eq(1),{diff(psi(1),t),psi(1),psi(2),...
thetaa(1),thetaa(2),thetab(1),thetab(2),sigma,Ra,q}, ...
{Xd,x,u,y,v,z,w,10,5*(3/2*pi^2)^3/q^2,double(sqrt(1/2)*pi)});
equ2=subs(eq(2),{diff(thetaa(1),t),psi(1),psi(2), ...
thetaa(1),thetaa(2),thetab(1),thetab(2),sigma,Ra,q},...
{Yd,x,u,y,v,z,w,10,5*(3/2*pi^2)^3/q^2,double(sqrt(1/2)*pi)});
equ3=subs(eq(3),{diff(thetab(1),t),psi(1),psi(2), ...
thetaa(1),thetaa(2),thetab(1),thetab(2),sigma,Ra,q}, ...
{Zd,x,u,y,v,z,w,10,5*(3/2*pi^2)^3/q^2,double(sqrt(1/2)*pi)});
equ4=subs(eq(4),{diff(psi(2),t),psi(1),psi(2), ...
thetaa(1),thetaa(2),thetab(1),thetab(2),sigma,Ra,q}, ...
{Ud,x,u,y,v,z,w,10,5*(3/2*pi^2)^3/q^2,double(sqrt(1/2)*pi)});
equ5=subs(eq(5),{diff(thetaa(2),t),psi(1),psi(2), ...
thetaa(1),thetaa(2),thetab(1),thetab(2),sigma,Ra,q}, ...
{Vd,x,u,y,v,z,w,10,5*(3/2*pi^2)^3/q^2,double(sqrt(1/2)*pi)});
equ6=subs(eq(6),{diff(thetab(2),t),psi(1),psi(2),...
thetaa(1),thetaa(2),thetab(1),thetab(2),sigma,Ra,q}, ...
{Wd,x,u,y,v,z,w,10,5*(3/2*pi^2)^3/q^2,double(sqrt(1/2)*pi)});
g(1) = simple(solve(equ1,Xd));
g(2) = simple(solve(equ2,Yd));
g(3) = simple(solve(equ3,Zd));
g(4) = simple(solve(equ4,Ud));
g(5) = simple(solve(equ5,Vd));
g(6) = simple(solve(equ6,Wd));
```

tangentspace.m

```
function tsp = tangentspace()
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %
% This tool calculates the tangent space of a dynamical system (9D)  %
% Input variables: none                                              %
% By Franz Bozsak                                                    %
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %


% head
clc
clear all
close all

% specify dimension of the system
dim = 9;

% specify variables

syms r  b sigma c1 c2

for k=1:dim
  x(k) = sym(sprintf('x%d', k));
  dx(k,1) = sym(sprintf('dx%d', k));
end

% specify equations of the dynamical system

dynsys = [%put in euations here!%];

% calculate the jacobian
jac0 = jacobian(dynsys,x);

% calculate the tangent space
tsp = jac0*dx;
```

Bibliography

[1] F Christiansen and H H Rugh: *Computing Lyapunov spectra with continuous Gram - Schmidt orthonormalization.* Nonlinearity, Vol. 10, No. 5, p. 1063–1072, 1997.

[2] James P. Crutchfield and David P. Feldman: *Regularities unseen, randomness observed: Levels of entropy convergence.* Chaos: An Interdisciplinary Journal of Nonlinear Science, Vol. 13, No. 1, p. 25–54, 2003.

[3] R. DU PUITS, C. RESAGK, A. TILGNER, F. H. BUSSE and A. THESS: *Structure of thermal boundary layers in turbulent Rayleigh?B?nard convection.* Journal of Fluid Mechanics, Vol. 572, No. -1, p. 231–254, 2007.

[4] Alexander Gluhovsky, Christopher Tong and Ernest Agee: *Selection of Modes in Convective Low-Order Models.* Journal of the Atmospheric Sciences, Vol. 59, No. 8, p. 1383–1393, April 2002.

[5] D. Roy and Z.E. Musielak: *Generalized Lorenz models and their routes to chaos. I. Energy-conserving vertical mode truncations.* Chaos, Solitons & Fractals, Vol. 32, No. 3, p. 1038 – 1052, 2007.

[6] D. Roy and Z.E. Musielak: *Generalized Lorenz models and their routes to chaos. II. Energy-conserving horizontal mode truncations.* Chaos, Solitons & Fractals, Vol. 31, No. 3, p. 747 – 756, 2007.

[7] D. Roy and Z.E. Musielak: *Generalized Lorenz models and their routes to chaos. III. Energy-conserving horizontal and vertical mode truncations.* Chaos, Solitons & Fractals, Vol. 33, No. 3, p. 1064 – 1070, 2007.

[8] B. SALTZMAN: *Finite Amplitude Free Convection As An Initial Value Problem .1.* Journal Of The Atmospheric Sciences, Vol. 19, No. 4, p. 329–342, 1962.

[9] Cosma Rohilla Shalizi and James P. Crutchfield: *Computational Mechanics: Pattern and Prediction, Structure and Simplicity.* Journal of Statistical Physics, Vol. 104, No. 3, p. 817–879, August 2001.

[10] Steven Henry Strogatz: *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering.* Da Capo Press, 2001.

[11] Y. M. Treve and O. P. Manley: *Energy conserving Galerkin approximations for 2-D hydrodynamic and MHD Bénard convection.* Physica D: Nonlinear Phenomena, Vol. 4, No. 3, p. 319 – 342, 1982.