NONLINEAR SYSTEMS, CHAOS AND CONTROL IN ENGINEERING

ASSIGNMENTS 2

Student: YI QIANG JI ZHANG

Professor: DR. CRISTINA MASOLLER, DR. ANTONIO PONS

Aerospace Engineering
The School of Industrial, Aerospace and Audiovisual Engineering of Terrassa
Polytechnic University of Catalonia | BarcelonaTech

April 22, 2021

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Contents

1 2D System Exercises

1.1 Probe solution of a linear system

1. Probe the following statement for a linear system of the form

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{1}$$

If $\mathbf{c}_1 = (x_1\ y_1)^T$ and $\mathbf{c}_2 = (x_2\ y_2)^T$ are solutions of the equation, then $\mathbf{c_3} = \alpha\mathbf{c_1} + \beta\mathbf{c_2}$ is also a solution of the equation.

To begin with, if both $\mathbf{c_1}$ and $\mathbf{c_2}$ are solutions to the linear system stated before, then $\alpha c_1 + \beta c_2$ are solutions to the system. Below is the proof,

Let

$$\dot{z} = \begin{pmatrix} x \\ y \end{pmatrix} \quad A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \tag{2}$$

Then it can be rewritten as,

$$\dot{z} = Az \tag{3}$$

Assume, z_1 and z_2 are solutions of 3, thereby, $\dot{z}_1 = Az_1$ and $\dot{z}_2 = Az_2$. Then for all α, $\beta \in \mathcal{R}$, $z = az_1 + bz_2$ is also a solution. Indeed,

$$\dot{z} = \alpha\dot{z}_1 + \beta\dot{z}_2 = \alpha Az_1 + \beta Az_2 = A(\alpha z_1 + \beta z_2) = \boxed{Az} \tag{4}$$

Another way of probing the solution is shown below.

If $\mathbf{c_1}$ is a solution to the differential equation, then, it satisfies that:

$$\dot{x} = ax_1 + by_1 \tag{5}$$
$$\dot{y} = ax_1 + by_1 \tag{6}$$

If $\mathbf{c_2}$ is a solution to the differential equation, then, it satisfies that:

$$\dot{x} = ax_2 + by_2 \tag{7}$$
$$\dot{y} = ax_2 + by_2 \tag{8}$$

Then,

$$c_3 = \begin{pmatrix} x_3 \\ y_3 \end{pmatrix} = \alpha \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \beta \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} \tag{9}$$

By first equation by arbitrary constants and sum the results,

$$k_1 \cdot (\dot{x_1} - ax_1 - by_1) + k_2 \cdot (\dot{x_2} - ax_2 - by_2) = 0 \tag{10}$$

and developing each term,

$$k_1\dot{x_1} + k_2\dot{x_2} + a\,(k_1x_1 + k_2x_2) + b\,(k_1y_1 + k_2y_2) = 0 \tag{11}$$

If k_1 and k_2 are substituted by α and β and the equation is rearranged

$$[\alpha x_1 + \beta x_2]' + a\,(\alpha x_1 + \beta x_2) + b\,(\alpha y_1 + \beta y_2) = 0 \tag{12}$$

Then, by replacing the above expression with x_3, we justify that the result is $\mathbf{c_3}$ is also a solution:

$$\boxed{x_3' + a\,(x_3) + b\,(x_3) = 0} \tag{13}$$

1.2 Numerical solution of a 2D Linear System

2. For the initial conditions $(x_1(0), y_1(0)) = (-1.25, 1.0)$ and $(x_2(0), y_2(0)) = (0.2, 1.5)$, solve numerically (using Matlab) the following systems:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{14}$$

and

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} 1 & -2 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{15}$$

Show, together, the time evolution of variables $x_i(t)$ and $y_i(t)$. One plot for each system. Then, plot, also, $y_i(t)$ as a function $x_i(t)$. Now we are going to consider the evolution in time of a linear combination of the two solutions: Set as an initial condition $(x_3(0), y_3(0)) = (\alpha x_1(0) + \beta x_2(0), \alpha y_1(0) + \beta y_2(0))$ for some α and β of your choice and solve numerically again the time evolution of the systems. Now compare the solutions, $(x_3(t), y_3(t))$ obtained numerically with the combination $(\alpha x_1(t) + \beta x_2(t), \alpha y_1(t) + \beta_{y2}(t))$ plotting them together in a single graph.

First, taking into consideration the first system (14). Considering $c_1 = (x_1(0), y_1(0)) = (-1.25, 1.0)$, the evolution of variables $x(t)$ and $y(t)$ are shown below (see Figure 1). This first system shows how the system remains stable but at the end of the time interval decreases rapidly. Notice how the systems are coupled since x variable appears in y-equation and vice-versa.

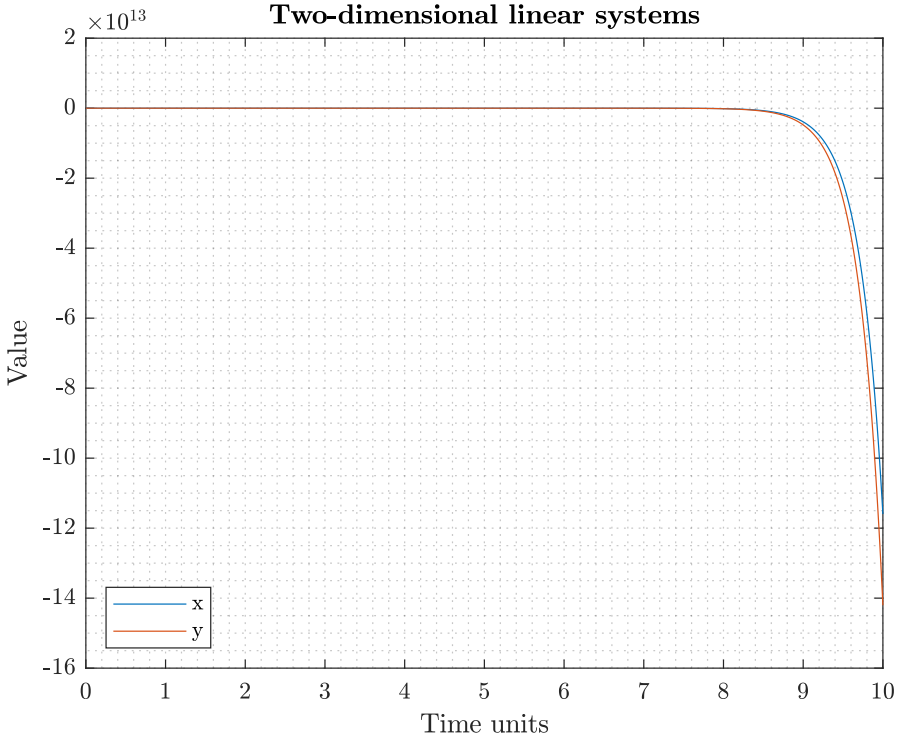


Figure 1 Evolution of $x(t)$ and $y(t)$ vs t. Source Own.

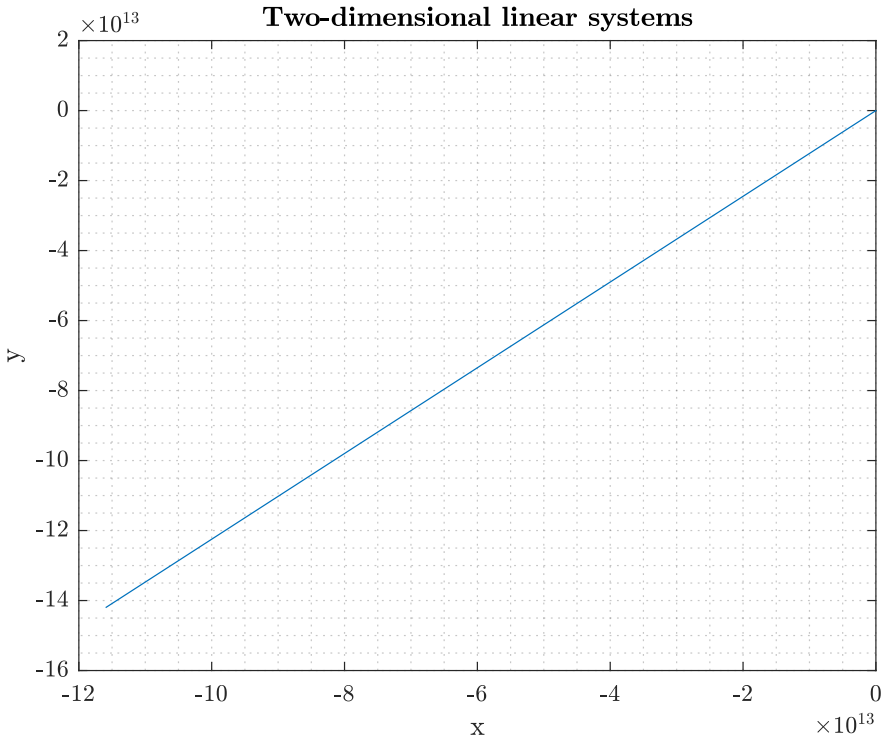


Figure 2 Phase Portrait. Source Own.

Next, using the second initial conditions $c_2 = (x_2(0), y_2(0)) = (0.2, 1.5)$, see how in this case, the behaviour differs and the system grows exponentially:

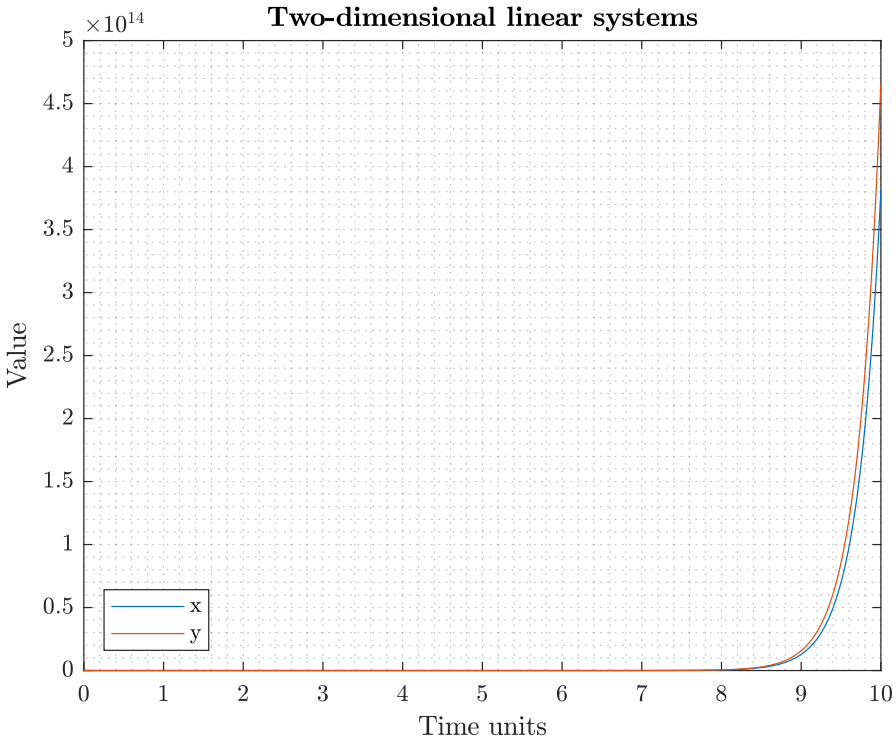


Figure 3 Evolution of $x(t)$ and $y(t)$ vs t. Source Own.

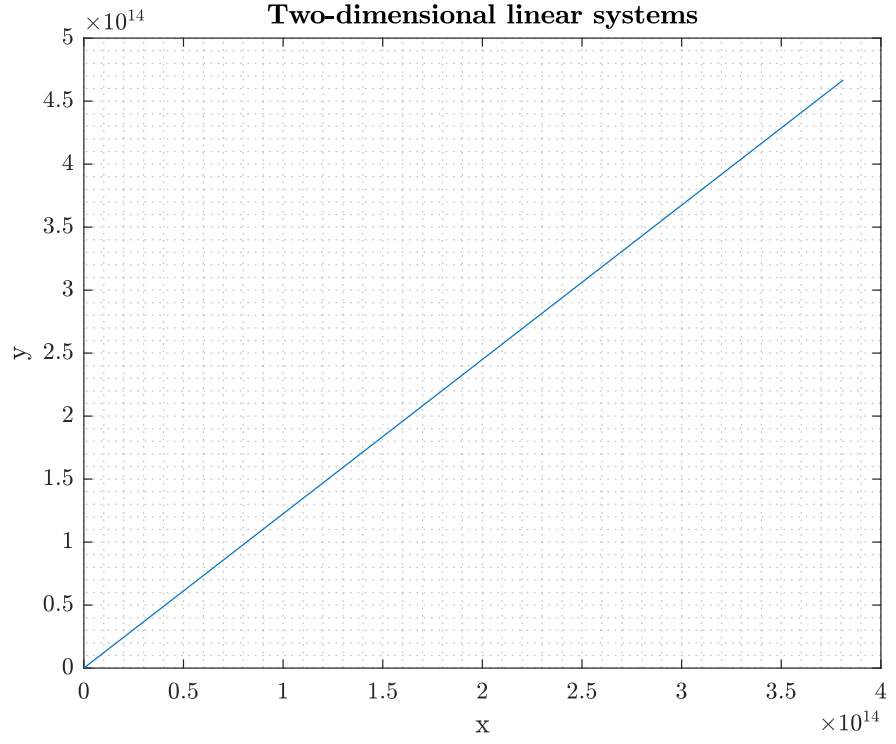


Figure 4 Phase Portrait. Source Own.

Now, considering the second system (15) with the initial conditions $c_2 = (x_2(0), y_2(0)) = (0.2, 1.5)$, we see how the system behaves differently from the first one. After a considerable time units, the systems oscillates (see Figure 5) and its phase portrait is a spiral. This means their eigenvalues are complex, and the fixed point can be either a center o a spiral. For instance, a harmonic oscillator is normally stable and centers. Spirals occurs when there are some damping mechanism, the trajectory would fail to close since the oscillator would lose some energy in each cycle.

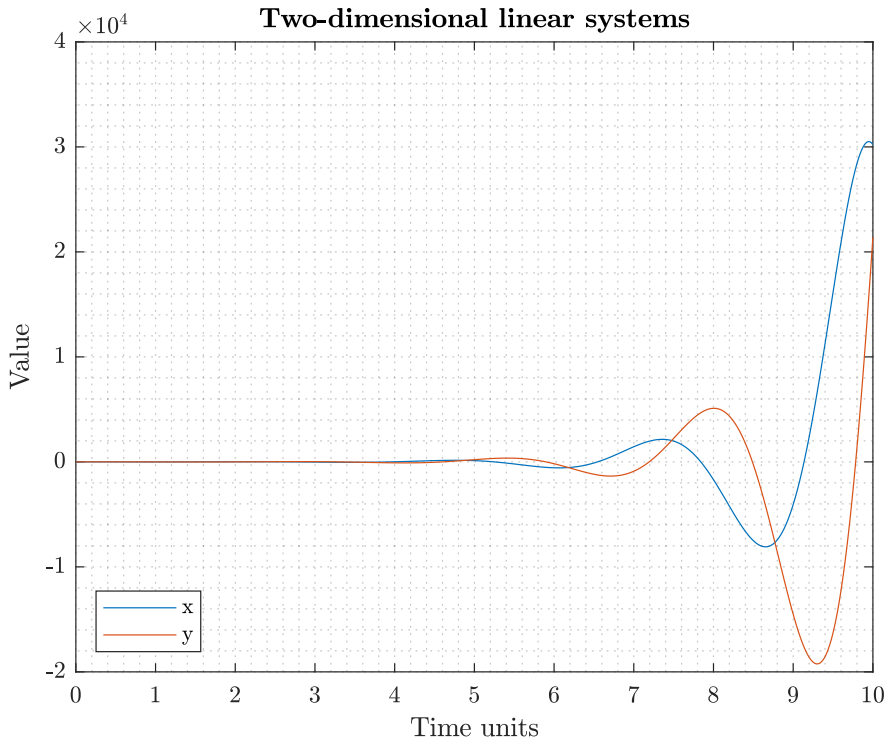


Figure 5 Evolution of $x(t)$ and $y(t)$ vs t. Source Own.

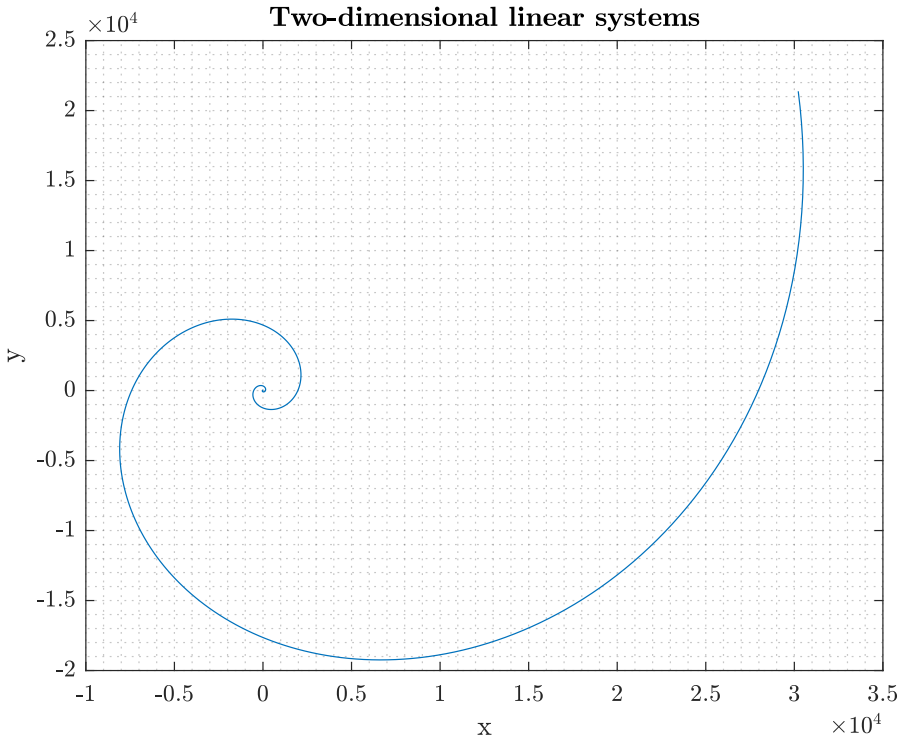


Figure 6 Phase Portrait. Source Own.

Using the same second system (15) but with the first initial conditions $c_1 = (x_1(0), y_1(0)) = (-1.25, 1.0)$ the following plots are obtained:

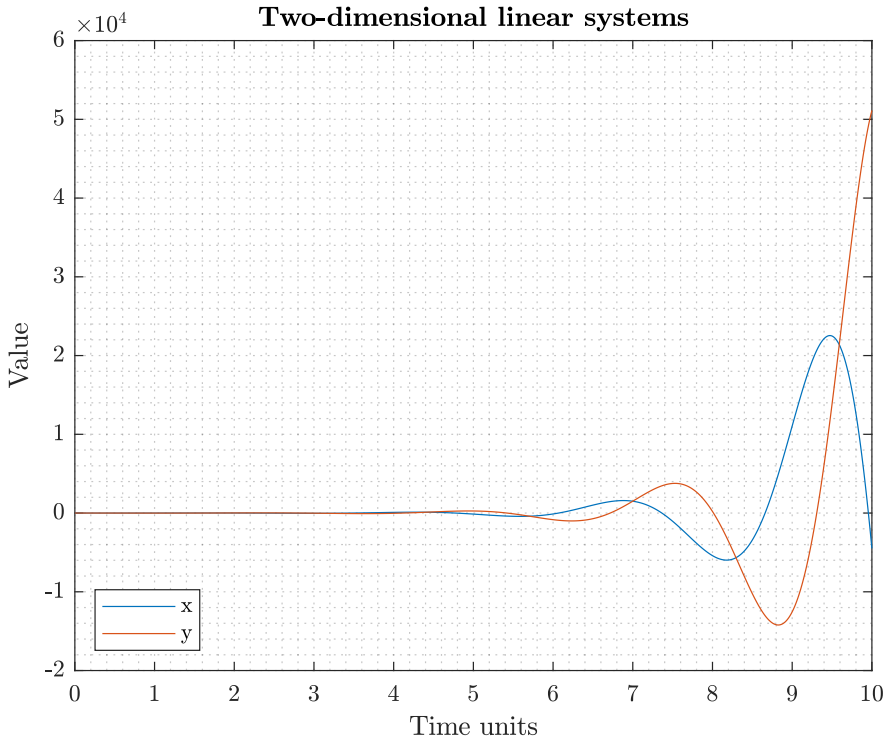


Figure 7 Evolution of $x(t)$ and $y(t)$ vs t. Source Own.

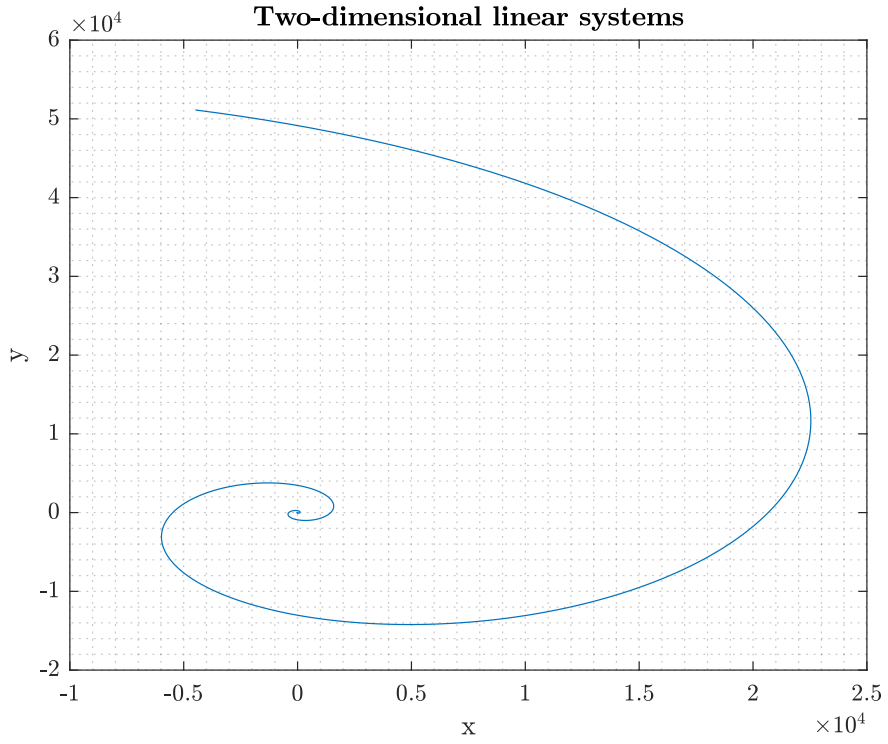
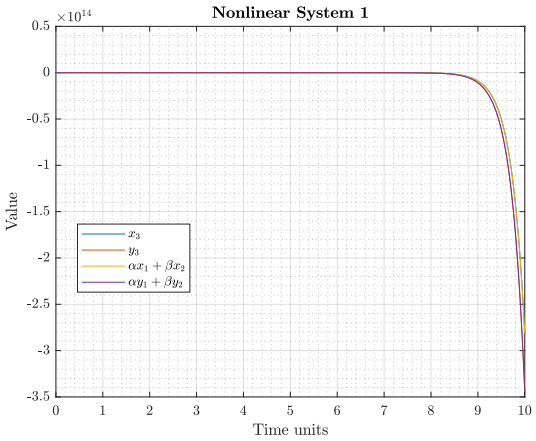
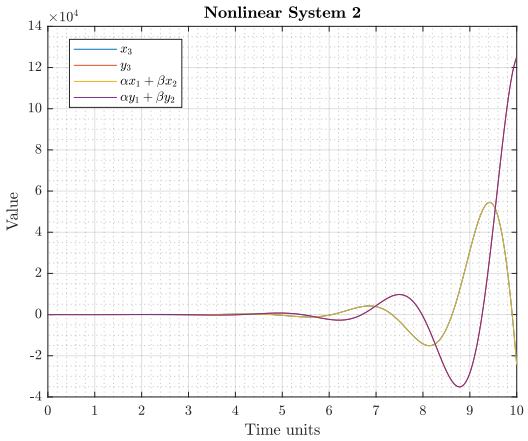


Figure 8 Phase Portrait. Source Own.

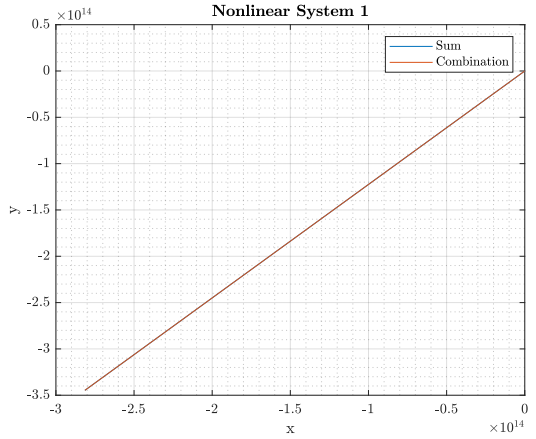
The final step is to analyse a linear combination of the first and second initial conditions. Considering the first set of 2D system (14). In this case, for the sake of simplicity $\alpha = 3$ and $\beta = 0.5$ are chosen. Thus, $c_3 = (x_3(0), y_3(0)) = (-3.65, 3.75)$. Figures 9 are obtained plotting all the results together in a single graph:



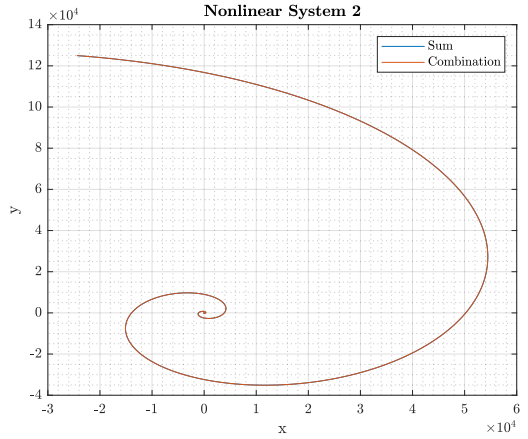
(a) System 1. Source: Own.



(b) System 2. Source: Own.



(c) System 1 Phase portrait. Source: Own.



(d) System 2 Phase portrait. Source: Own.

Figure 9 Linear combination solution

It is easily seen how the solutions are exactly the same by using the linear combination of the initial conditions.

1.3 Phase Portrait

Find all the fixed points of the system

$$\begin{aligned} \dot{x} &= -x + 4x^3 \\ \dot{y} &= -2x \end{aligned} \tag{16}$$

and use its linearization to classify them. Considering different initial conditions, derive, numerically, the phase portrait for the full nonlinear system. Hint: There are (many!!) different fixed points.

The first step is to finding the fixed points. Fixed points can be found by setting $\dot{x}$ and $\dot{y}$ equal to zero. Fixed points (x^*, y^*) are the following:

- $(0, y)$
- $(+0.5, y)$
- $(-0.5, y)$

That means, there are a set of infinite fixed points where y is arbitrary. Afterwards, it is seen how the Jacobian matrix does not depend on the y component either.

Once the fixed points are found, the next step is to analyse its stability. This shall be done by assaying the Jacobian matrix-s determinant and its trace.

Let

$$f(x, y) = \dot{x} = -x + 4x^3 \tag{17}$$

$$g(x, y) = \dot{y} = -2x \tag{18}$$

Then the Jacobian matrix is set to be,

$$J = \begin{pmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{pmatrix} \tag{19}$$

Using the Jacobian matrix, be get the following:

$$J = \begin{pmatrix} -1 + 12x^2 & 0 \\ -12 & 0 \end{pmatrix} \tag{20}$$

After, we shall evaluate the Jacobian matrix at these fixed points. Thus, the eigenvalues can be obtained by using $\det(Ax - \lambda I) = 0$. Since the determinant is always 0 (see the Jacobian matrix):

- Considering $(0, y)$, evaluating the Jacobian matrix at that point:

$$\det\left[\begin{pmatrix} -1 & 0 \\ -12 & 0 \end{pmatrix}\right] = 0 \tag{21}$$

  And its trace is

$$\operatorname{tr}\left[\begin{pmatrix} -1 & 0 \\ -12 & 0 \end{pmatrix}\right] = -1 \tag{22}$$

  Thus, this set of fixed points are a set of **stable** fixed points nodes.

- Considering $(+0.5, y)$, evaluating the Jacobian matrix at that point:

$$\det\left[\begin{pmatrix} 2 & 0 \\ -12 & 0 \end{pmatrix}\right] = 0 \tag{23}$$

And its trace is

$$\operatorname{tr}\left[\begin{pmatrix} 2 & 0 \\ -12 & 0 \end{pmatrix}\right] = 2 \tag{24}$$

Thus, this set of fixed points are a set of **unstable** fixed points nodes.

- Considering $(-0.5, y)$, evaluating the Jacobian matrix at that point:

$$\det\left[\begin{pmatrix} 2 & 0 \\ -12 & 0 \end{pmatrix}\right] = 0 \tag{25}$$

And its trace is

$$\operatorname{tr}\left[\begin{pmatrix} 2 & 0 \\ -12 & 0 \end{pmatrix}\right] = 2 \tag{26}$$

Thus, this set of fixed points are a set of **unstable** fixed points nodes.

Since the determinant of all of the set of fixed points is 0, at least one of the eigenvalues is zero. Then the origin is not an isolated fixed point but a whole line of fixed points.

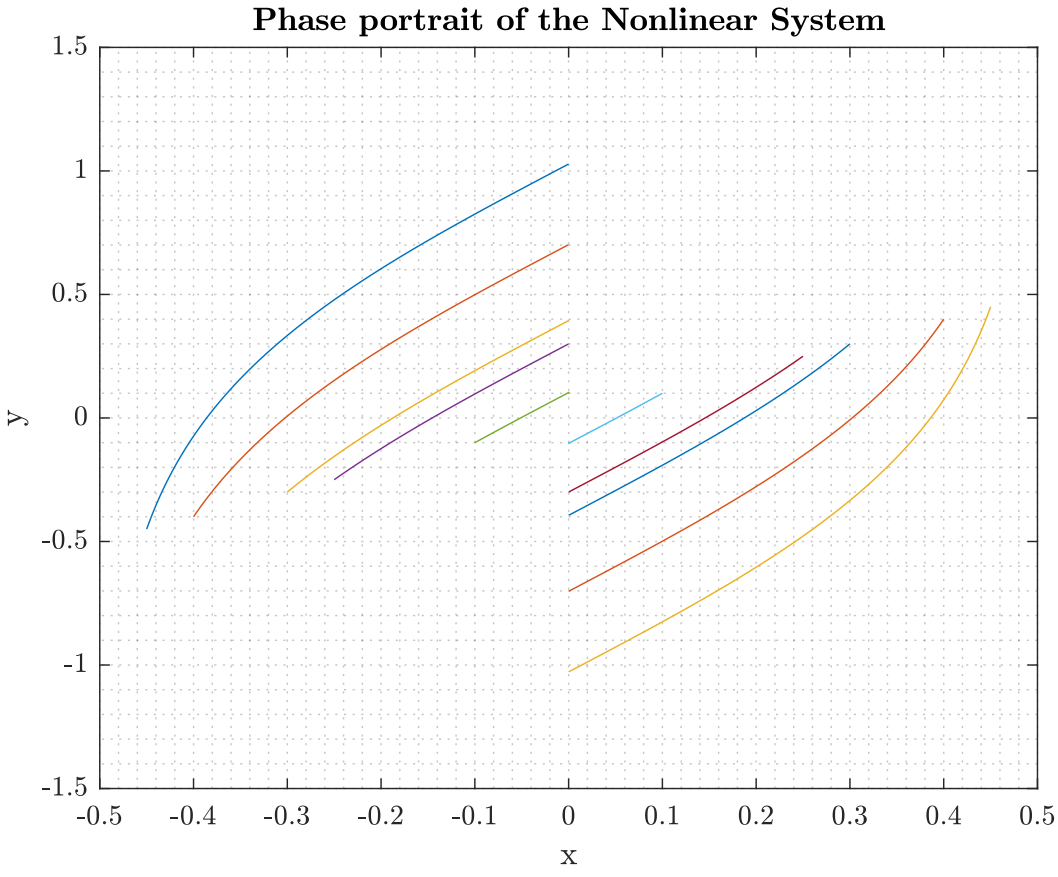


Figure 10 Phase Portrait. Source Own.

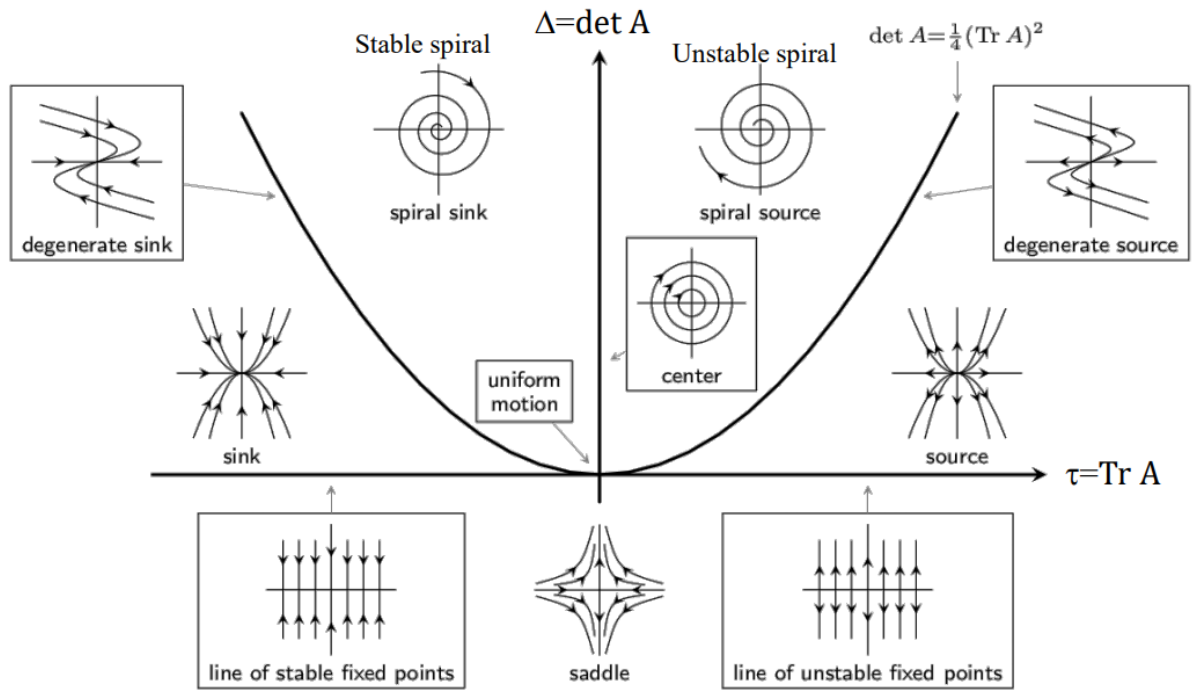


Figure 11 Classification of linear systems. Source [2].

1.4 Direction Field

Consider the system

$$\begin{aligned} \dot{x} &= x + \exp(-y) \\ \dot{y} &= -y \end{aligned} \tag{27}$$

Plot, in phase space, the direction field (using the Matlab function quiver) for this system. On top of it, plot a few trajectories.

The direction field of the above-mentioned system is presented below:

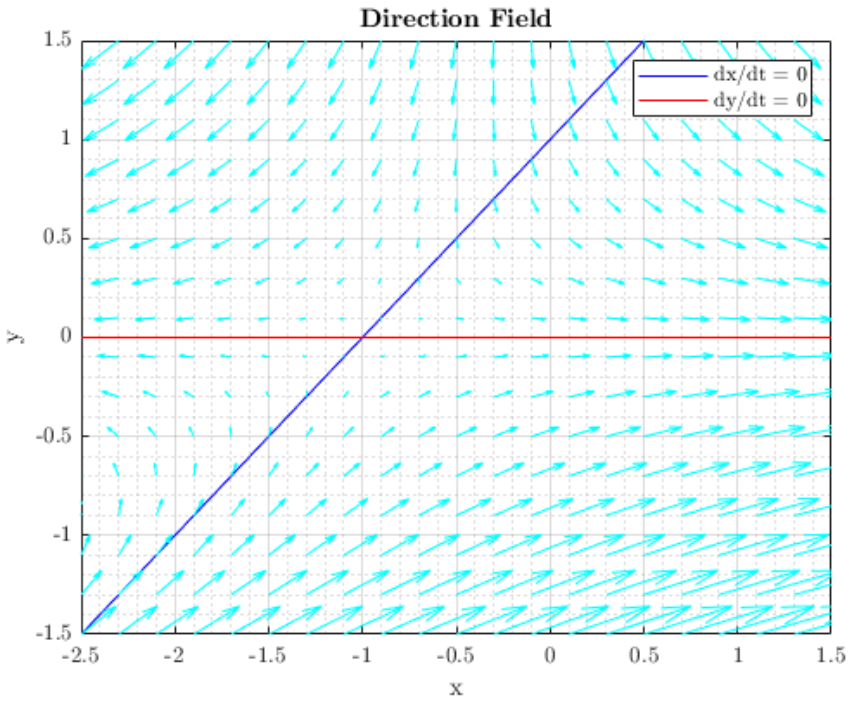


Figure 12 Direction Field. Source Own.

Direction fields enables to have a notion of the solution to the differential equations, this can be achieved by drawing the tangents of the function curves on a regular grid.

Figures 13 and 14 shows some trajectories of the system for various initial conditions $x0 = [02550100]$ and $y0 = [02550100]$, notice how y-variable decays exponentially while x variable grows exponentially. This is easily deduced as in the equation above, y variable only depends on y.

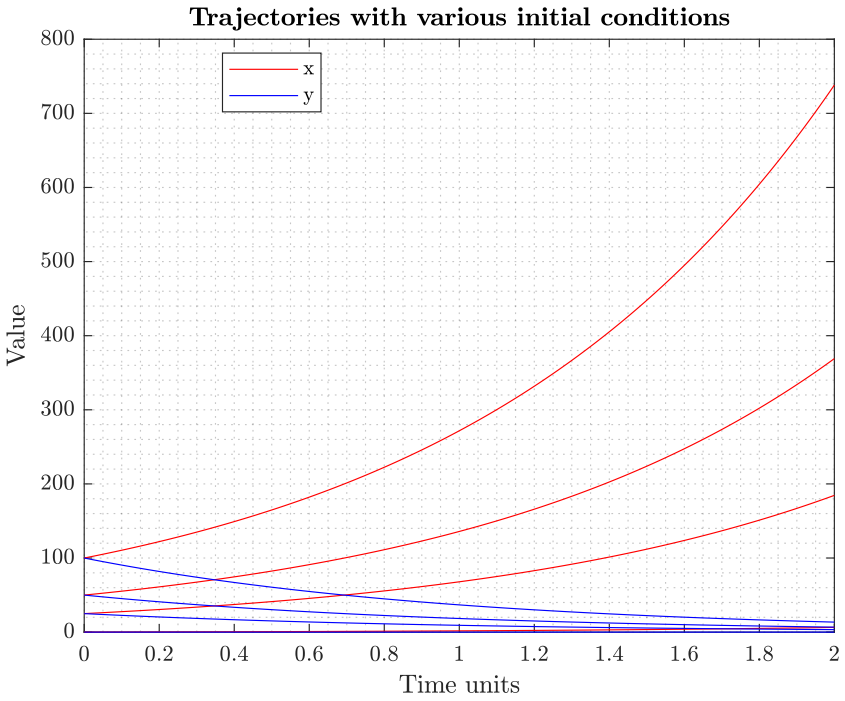


Figure 13 Trajectories plot. Source Own.

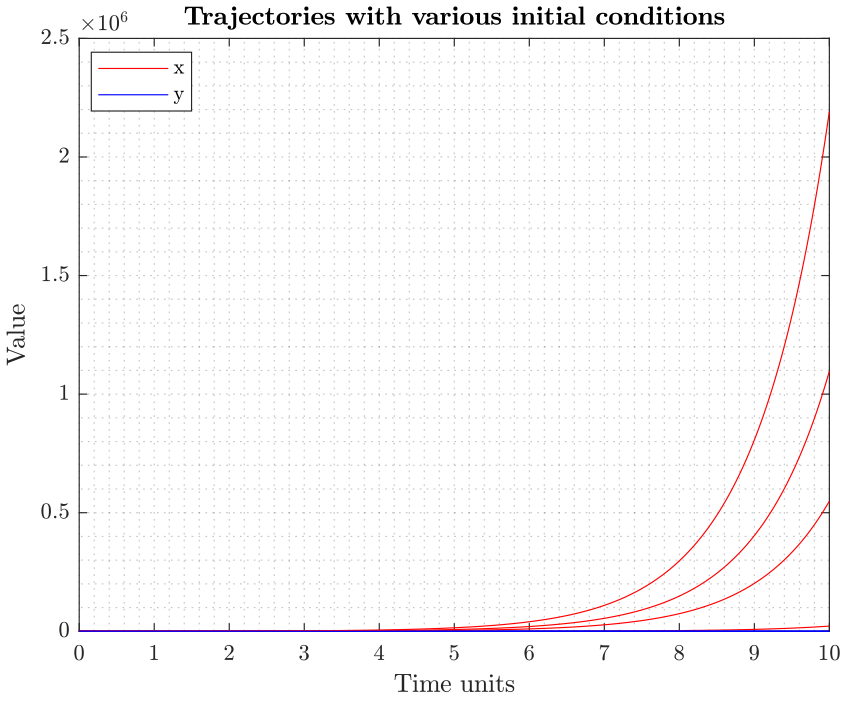


Figure 14 Trajectories plot (zoom-out). Source Own.

2 Chaotic systems Exercises

2.1 Lorenz System

Let us consider the Lorenz system:

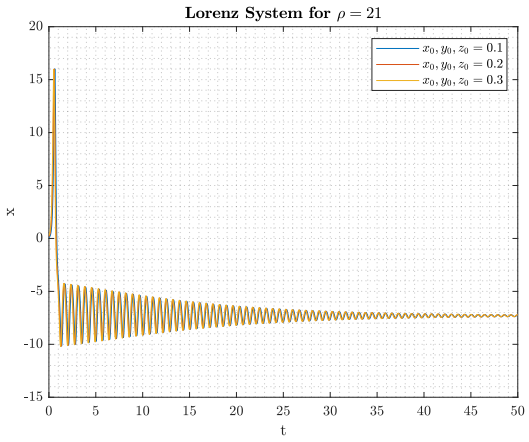
$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy \end{aligned} \tag{28}$$

with $\sigma = 10$ and $\beta = \frac{8}{3}$. Show the evolution in time of variables $x(t), y(t)$ and $z(t)$ for different initial conditions grouped in pairs which are very close (to check if the evolution is similar as time proceeds) for the following ρ values: $21, 24.15, 30$. Show also, x(t) versus z(t) for the same initial conditions. **Explain** to which dynamical regime corresponds each of the ρ considered.

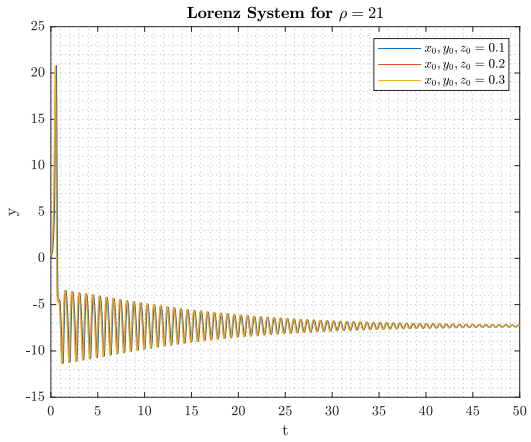
The Lorenz system is a system of ordinary differential equations that is notable of showing a chaotic solution for certain parameters. In fact, the Lorenz attractor is a set of chaotic solutions of the Lorenz system.

Lorenz discovered that this simple-looking deterministic system could have extremely erratic dynamics: over a wide range of parameters, the solutions oscillate irregularly, never exactly repeating but always remaining in a bounded region of phase space. When he plotted the trajectories in three dimensions, he discovered that they settled onto a complicated set, now called a strange attractor. Unlike stable fixed points and limit cycles, the strange attractor is not a point or a curve or even a surface. It is a fractal, with a fractional dimension between 2 and 3.

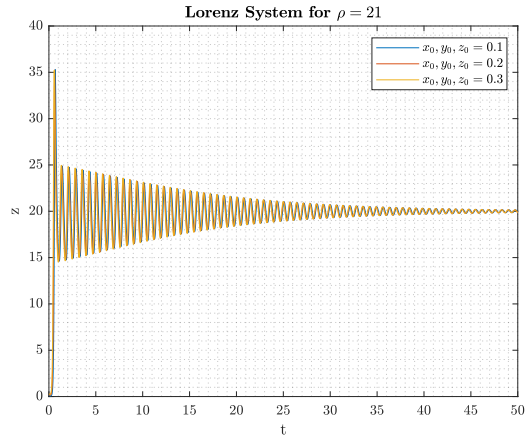
Below is presented the resultant numerical integration of the Lorenz system for various values of ρ (see Figures 15 17 19). For a bigger image, please go to A).



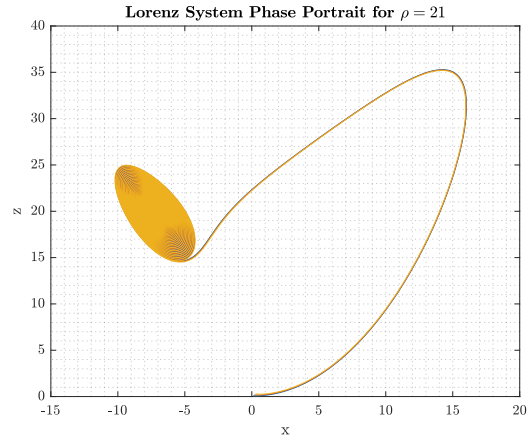
(a) Lorenz System for $\rho = 21$



(b) Lorenz System for $\rho = 21$



(c) Lorenz System for $\rho = 21$



(d) Lorenz System Phase Portrait for $\rho = 21$

Figure 15 Lorenz System for $\rho = 21$. Source: Own.

For small values of ρ, the system is stable and evolves to one of two fixed point attractors. When ρ is larger than 24.74, the fixed points become repulsors and the trajectory is repelled by them in a very complex way (see Figure 17 and 19).

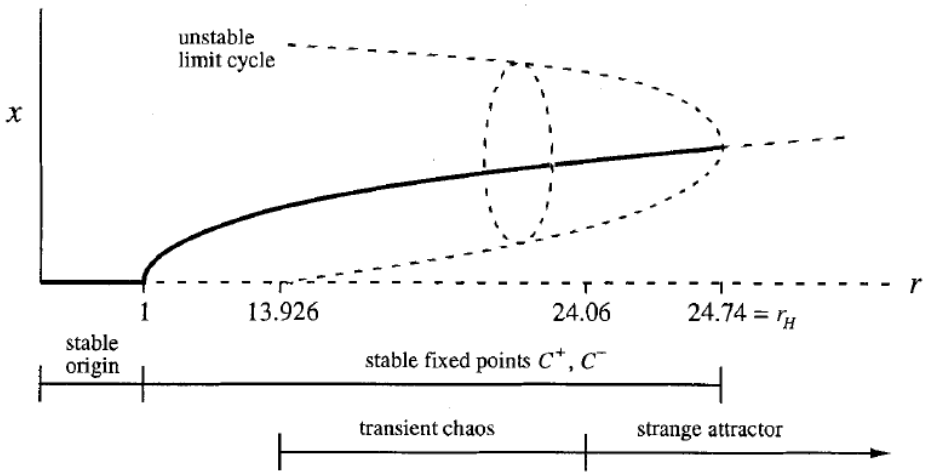


Figure 16 Lorenz parameter space. Source [4].

The analysis was made for various parameters, $\sigma = 10$, $\beta = 8/3$ and $rho = 21\,24.15\,30$. The question is, what happens when ρ goes from 21 to 30?

To respond this question see Figure 16, depending on ρ one can find exotic limit cycles tied in knots, pairs of limit cycles linked through each other, intermittent chaos, noisy periodicity.

In the analysis, values $\sigma = 10$, $\beta = 8/3$ were kept constant while varying ρ, the behaviour is summarized in Figure 16. It can be seen that the origin is globally stable for $r < 1$. At $r = 1$ the origin loses stability by a supercritical pitchfork bifurcation, and a symmetric pair of attracting fixed points is born (in our schematic, only one of the pair is shown).

If $r > 1$, then $\Delta < 0$ there is Saddle point with 2 incoming directions and 1 outgoing direction Besides, there are C^+ and C fixed point solutions.

It can be shown (with the standard techniques that we know), that they are linearly stable for $(\sigma - b - 1 > 0)$:

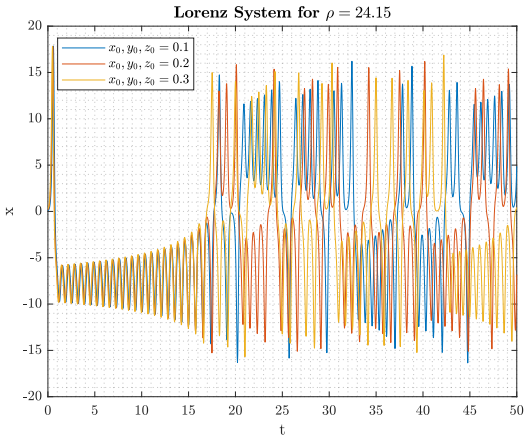
$$1 < r < r_{\mathrm{H}} = \frac{\sigma \cdot (\sigma + b + 3)}{(\sigma - b - 1)} \tag{29}$$

At $r_H = 24.74$ the fixed points lose stability by absorbing an unstable limit cycle in a subcritical Hopf bifurcation.

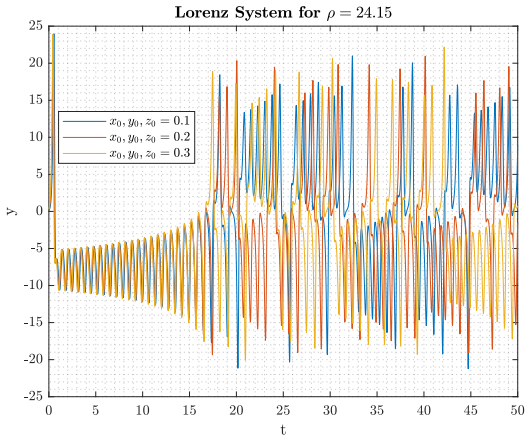
If r is increased from r_H, the unstable limit cycles expand and just at $r = 13.926$ the cycles touch the saddle point and there is a homoclinic bifurcation. There are no limit cycles below 13.926. Unstable limit cycles are created as r increases through $r = 13.926$. Thus, an invariant set is born at $r = 13.926$ which is a set of many saddle-cycles and very sensitive dependence on initial conditions.Trajectories can get hung up near this set, somewhat like wandering in a maze. Then they rattle around chaotically for a while, but eventually escape and settle down to C^+ or C^-. The time spent wandering near the set gets longer and longer as r increases.

Finally, at $r = 24.06$ the time spent wandering becomes infinite and the set becomes a strange attractor (see Figure 17).

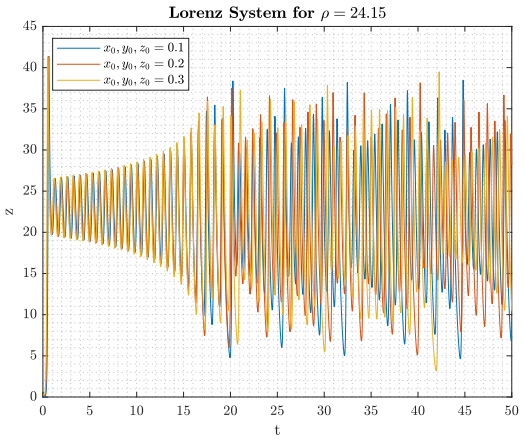
The trajectory seems to be a strange attractor but it stays on the right and spirals down toward the stable fixed point C^+. The plot of variables x, y and z over t shows a transient chaos.



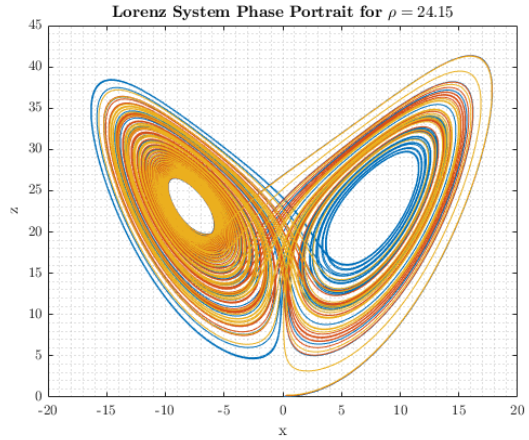
(a) Lorenz System for $\rho = 24.15$



(b) Lorenz System for $\rho = 24.15$



(c) Lorenz System for $\rho = 24.15$



(d) Lorenz System Phase Portrait for $\rho = 24.15$

Figure 17 Lorenz System for $\rho = 24.15$. Source: Own.

After an initial transient, the solution settles into an irregular oscillation that persists as $t \to \infty$, but never repeats exactly. The motion is aperiodic.

If the solution is plotted as a trajectory in phase space, a structure emerges, this structure is the famous butterfly pattern when $x(t)$ is plotted against $z(t)$.

Looking at the phase space, the trajectory appears to cross itself repeatedly, but that's just an artifact of projecting the three-dimensional trajectory onto a two-dimensional plane. In three dimensions no self-intersections occur. The trajectory starts near the origin, then swings to the right and delves into the center of a spiral on the left. After a very slow spiral outward, the trajectory comes back over the right side, spirals around a few times and shoots over the left side and vice-versa indefinitely. The number of circuits made on either side varies unpredictably from one side to the next. In fact, the sequence of the number of circuits shares many of the characteristics of a random sequence.

When the trajectory is viewed in all three dimensions, rather than in a two-dimensional projection, it appears to settle onto an exquisitely thin set that looks like a pair of butterfly wings.

Lorenz suggests that the geometrical structure of the strange attractor is a pair of surfaces that merge into one in the lower portion of 18. He explains that the surfaces only appear to merge.

This infinite complex of surfaces is known as a fractal. It is a set of points with zero volume but infinite surface area. And numerical experiments suggest that it has a dimension of about 2.05.

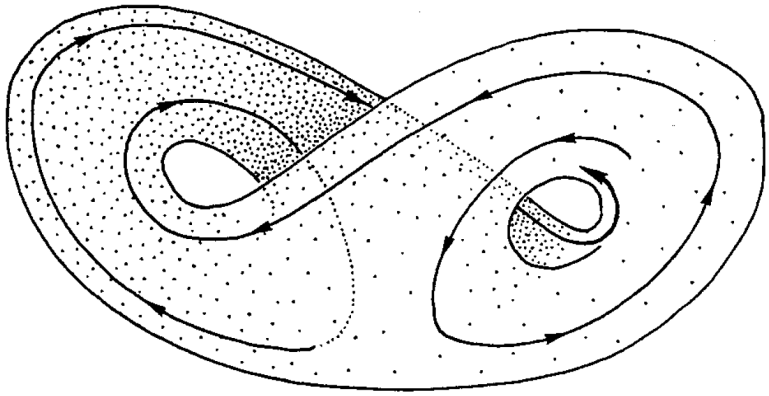
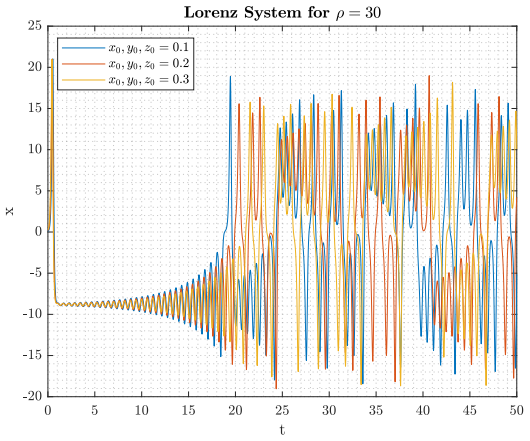
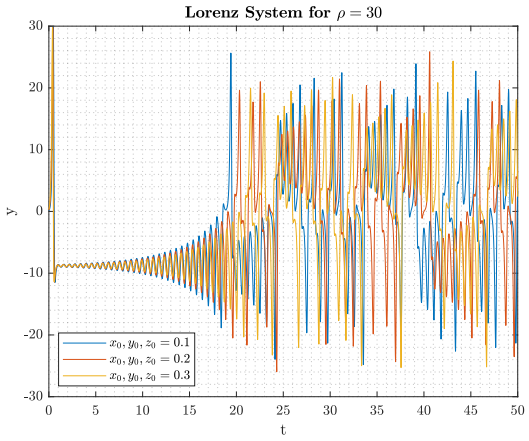


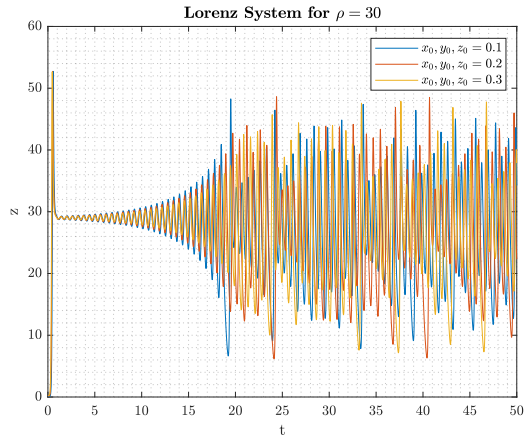
Figure 18 Lorenz structure. Source [4].



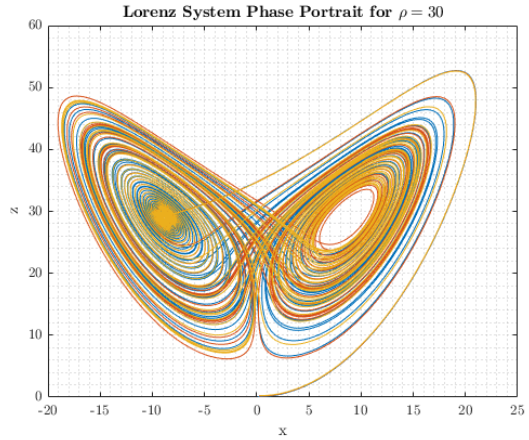
(a) Lorenz System for $\rho = 30$



(b) Lorenz System for $\rho = 30$



(c) Lorenz System for $\rho = 30$



(d) Lorenz System Phase Portrait for $\rho = 30$

Figure 19 Lorenz System for $\rho = 30$. Source: Own.

Lorenz showed that in a certain range of parameters, there could be no stable fixed points and no stable limit cycles, yet he also proved that all trajectories remain confined to a bounded region and are eventually attracted to a set of zero volume. That set is the strange attractor, and the motion on it is chaotic.

Some properties of the Lorenz systems are listed below:

- System (28) has only two nonlinearities, the qudratic termns and xy and xz.
- There is an important symmetry in the Lorenz equations. If (x, y) are replaced with $(-x, -y)$, the equations stay the same. Hence, if $(x(t), y(t), z(t))$ is a solution, so is $(-x(t), -y(t), z(t))$. In other words, all solutions are either symmetric themselves, or have a symmetric partner.
- Lorenz system is **dissipative**, volumes in phase space contract under the flow. Thus volumes in phase space shrink exponentially fast.

2.2 Maximum Lyapounov Exponent

Using the results of the previous exercise, compute the Maximum Lyapunov Exponent obtained from the evolution of the pairs of initial conditions. Explain the relation of the exponent obtained with the predictability of the trajectories obtained from your numerical simulations. Hint: To determine the exponent, use the fitting procedure explained in class selecting accurately the time range used to make the fit. Remember that the exponential amplification is only valid when trajectories are very close.

The motion on the attractor exhibits sensitive dependence on initial conditions. This means that two trajectories starting very close together will rapidly diverge from each other, and thereafter have totally different outcomes.

In numerical studies of the Lorenz attractor, one can find the following relationship:

$$\|\delta(t)\| \sim \|\delta_0\| e^{\lambda t} \tag{30}$$

If the plot of $\ln \delta(t)$ versus t, it is found a curve that is close to a straight line with a positive or negative slope of λ.

The Lyapunov exponent λ is a tool that is very useful for determining whether the system is strongly dependant on initial conditions or not.

Below is presented the Lyapunov exponent for different values of ρ and different variables. This exponent is obtained by plotting the difference $\ln \delta$ between pairs of initial conditions.

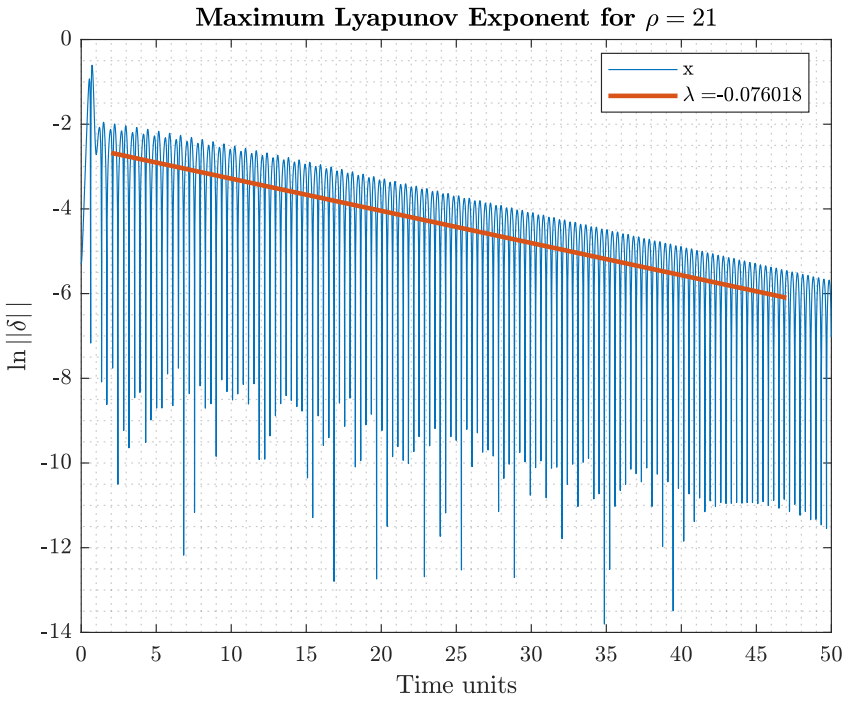


Figure 20 Maximum Lyapunov exponent for x for $\rho = 21$. Source Own.

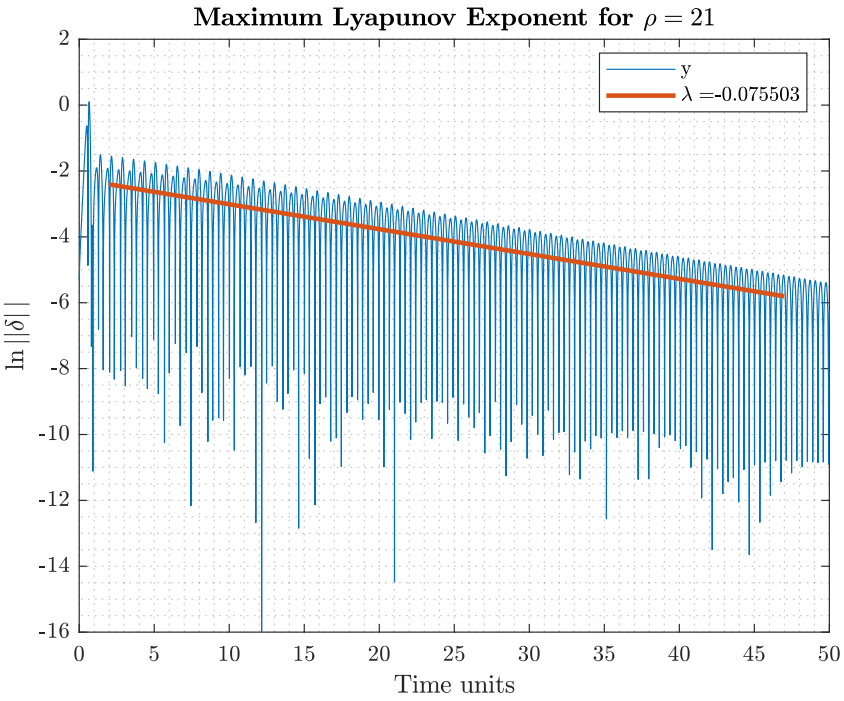


Figure 21 Maximum Lyapunov exponent for y for $\rho = 21$. Source Own.

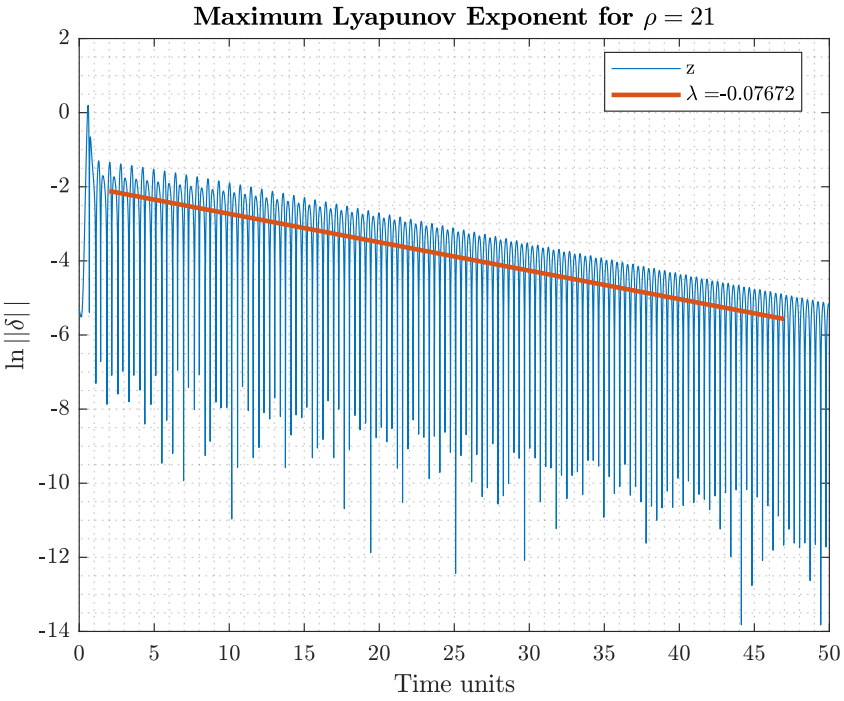


Figure 22 Maximum Lyapunov exponent for z for $\rho = 21$. Source Own.

For the first $\rho = 21$, the maximum Lyapunov exponent is in absolute value:

$$\lambda = -0.07672 \tag{31}$$

This exponent is negative, which means that the dynamical is either periodic or refers to a fixed point. Besides, it is not strongly dependant on initial conditions. If two initial conditions starts distance apart, that distance gets smaller over time.

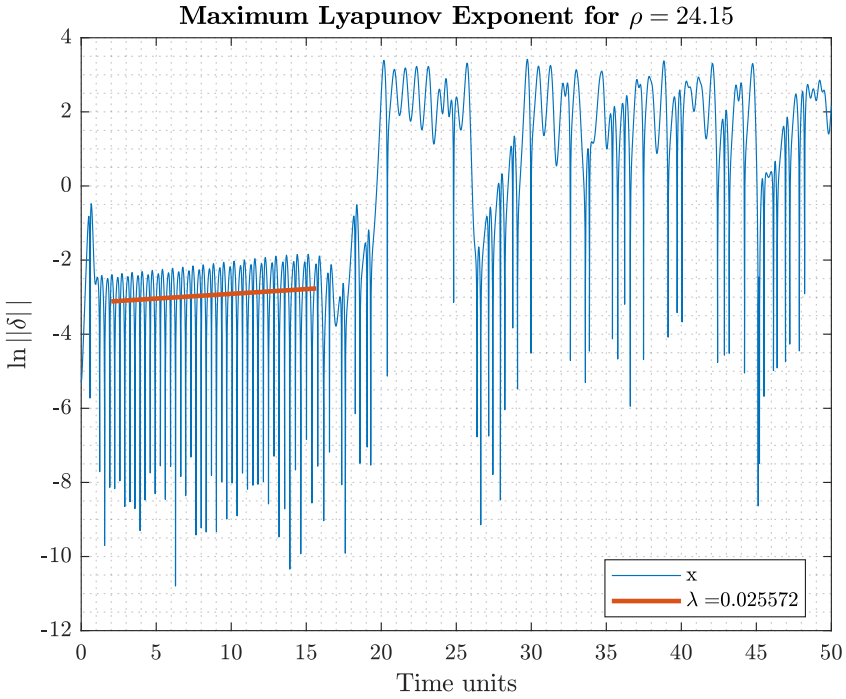


Figure 23 Maximum Lyapunov exponent for x for $\rho = 24.15$. Source Own.

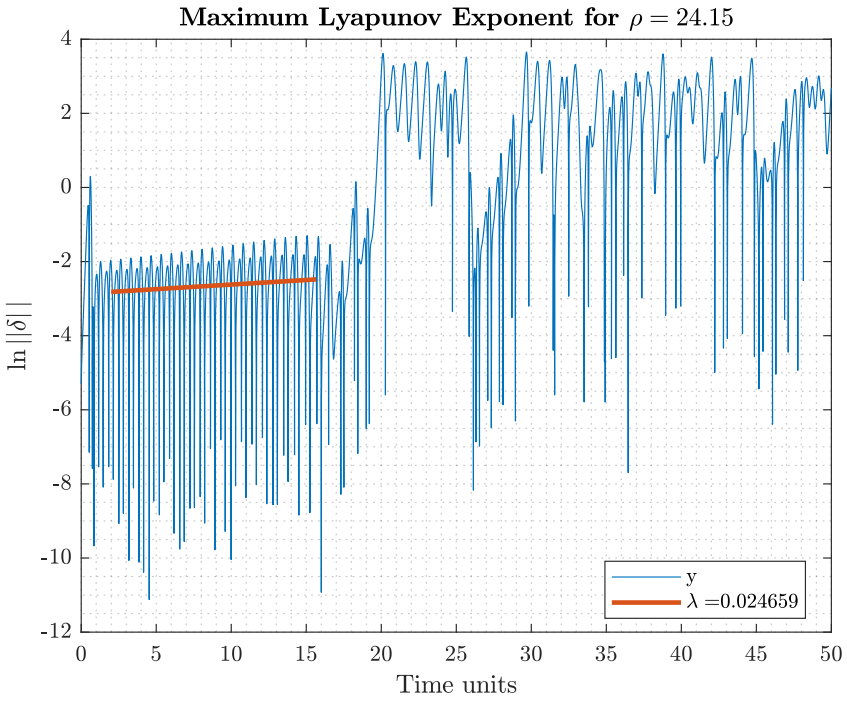


Figure 24 Maximum Lyapunov exponent for y for $\rho = 24.15$. Source Own.

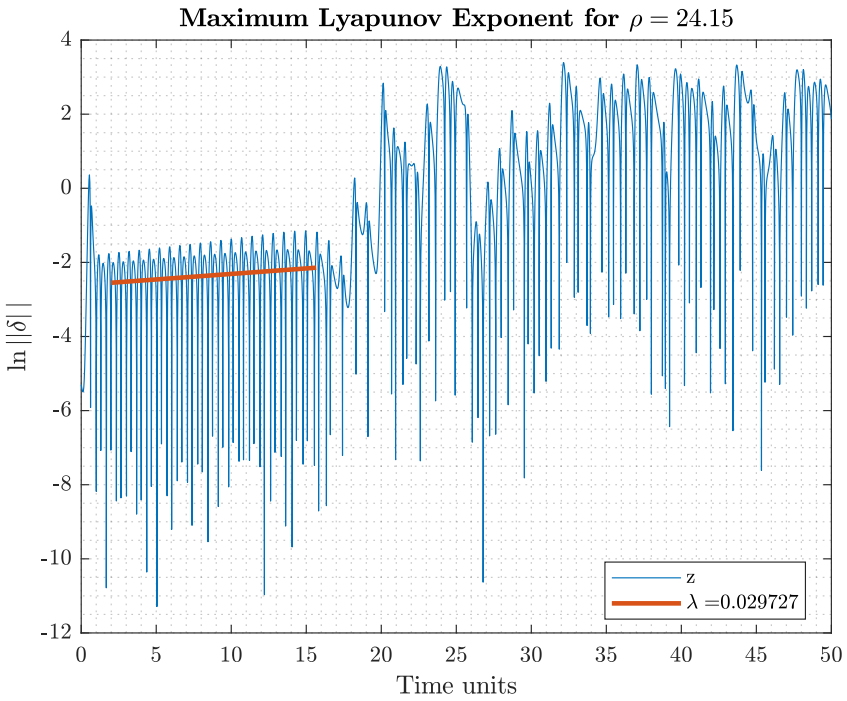


Figure 25 Maximum Lyapunov exponent for z for $\rho = 24.15$. Source Own.

For $\rho = 24.15$, the maximum Lyapunov exponent is positive ($\lambda = 0.029727$), really close to zero.

If the Lyapunov exponent happens to be $\lambda = 0$, an interesting phenomenon takes place, the dynamical system behaves as a bifurcation point.

For $\rho = 30$, the maximum Lyapunov exponent is positive ($\lambda = 0.22156$). A positive exponent means that the system is chaotic and strongly dependant on initial conditions. A small perturbation on the initial conditions will have exceptionally different outcomes.

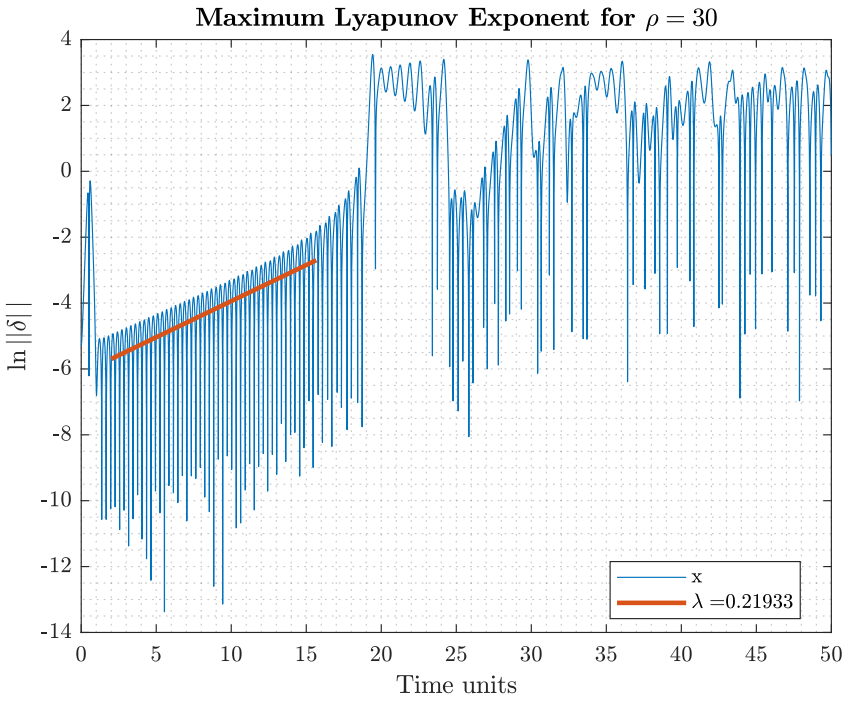


Figure 26 Maximum Lyapunov exponent for x for $\rho = 30$. Source Own.

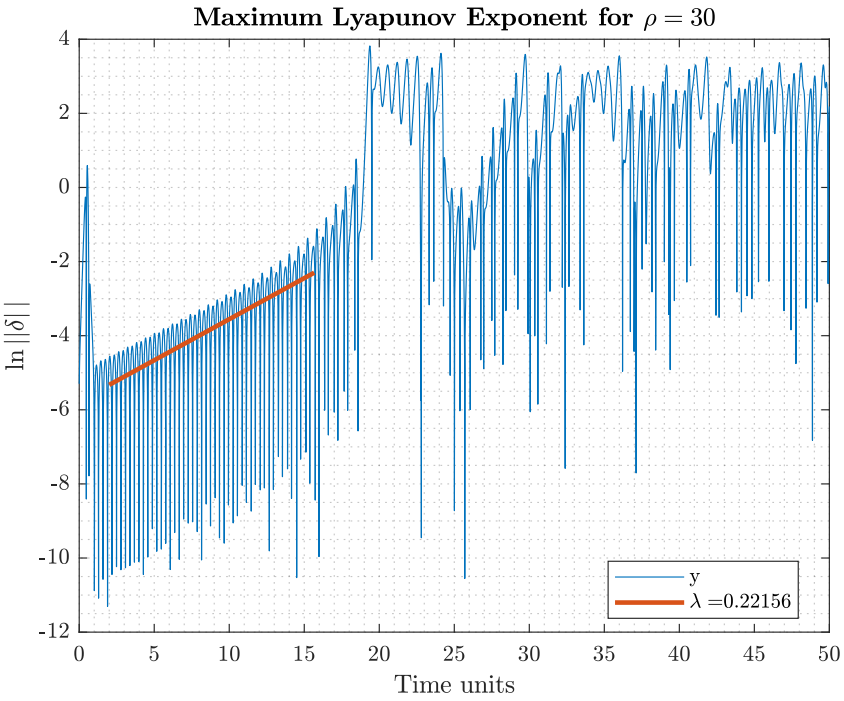


Figure 27 Maximum Lyapunov exponent for y for $\rho = 30$. Source Own.

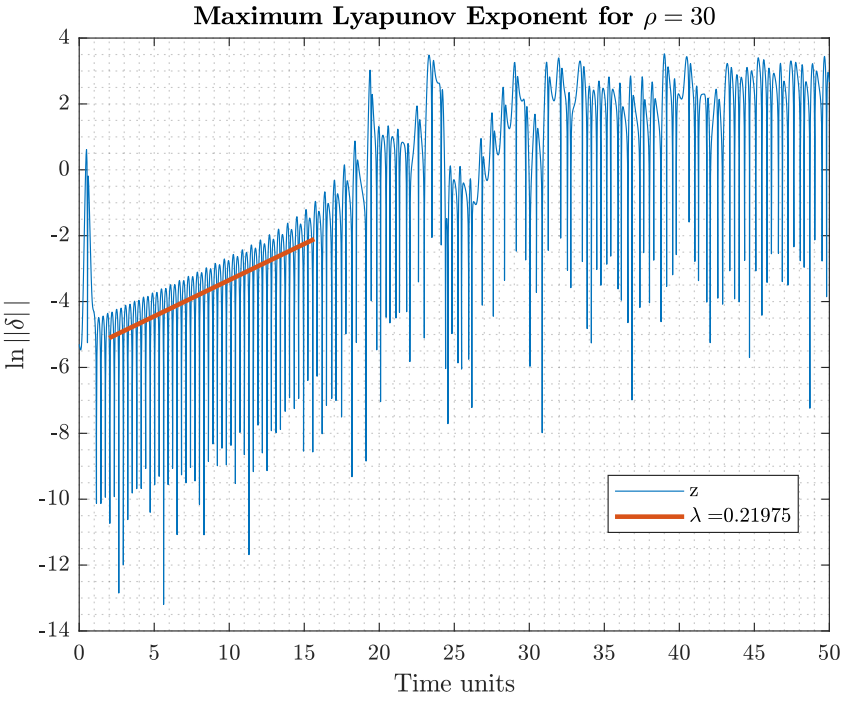


Figure 28 Maximum Lyapunov exponent for z for $\rho = 30$. Source Own.

Some observations can be made regarding the Lyapunov exponent:

- The curve is never exactly straight. It has wiggles because the strength of the exponential divergence varies somewhat along the attractor.
- The exponential divergence must stop when the separation is comparable to the "diameter" of the attractor-the trajectories obviously can't.

Talking about the predictability, when a system has a positive Lyapunov exponent, there is a time horizon beyond which prediction breaks down. Suppose we measure the initial conditions of an experimental system very accurately. Of course, no measurement is perfect there is always some error $\|\delta_0\|$ between our estimate and the true initial state.

pag 330

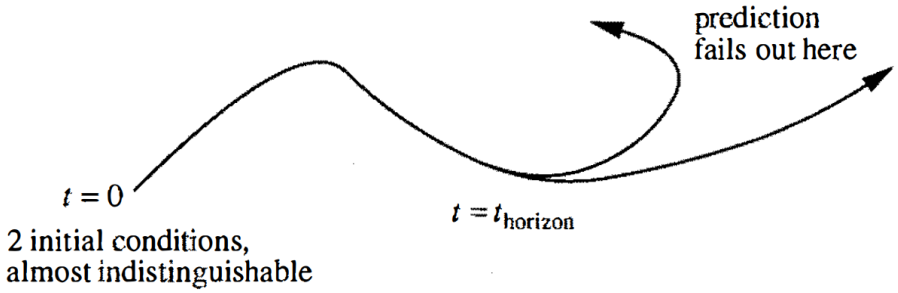


Figure 29 Predictability. Source [4].

After a time t, the discrepancy grows to $\|\delta(t)\| \sim \|\delta_0\| e^{\lambda t}$. Let a be the tolerance, so if a prediction is within a of the true state, we consider it acceptable. Then our prediction becomes intolerable when $\|\delta(t)\| \geq a$, this of occurs after a time:

$$t_{horizon} = O\left(\frac{1}{\lambda} \ln \frac{a}{\|\delta_0\|}\right) \tag{32}$$

The term that affects the most is the logarithmic dependence on $\|\delta_0\|$. No matter how small is is the initial measurement error, it is not possible to predict longer than a few multiples of $\frac{1}{\lambda}$.

This is why for example it is so difficult to predict the long term weather behaviour. The detailed prediction of a long-term behavior of a chaotic system is highly difficult.

2.3 Henon Map

Consider the Henon Map:

$$\begin{aligned} x_{n+1} &= y_n + 1 - a x_n^2 \\ y_{n+1} &= b x_n \end{aligned} \tag{33}$$

Considering $a = 1.4$ and $b = 0.3$, iterate the map for 10.000 successive iterates. Show, all the points in a x_n versus y_n graph. **Explain** the structure of the graph (discussed in the class) and the mechanism needed to produce it.

There are at least two maps known as the Hénon Map. The first is the two-dimensional dissipative quadratic map given by the coupled equations described above.

However, with$a = 1.4$ and $b = 0.3$ a strange attractor [1] appears.

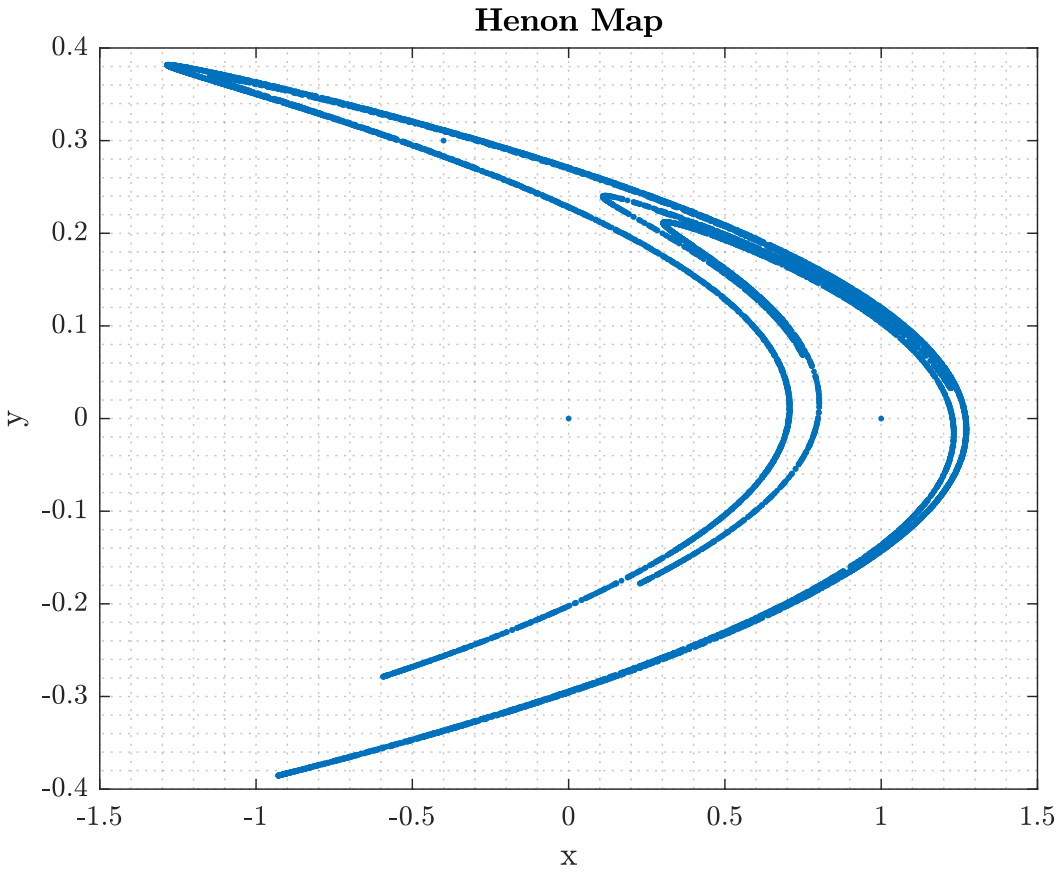


Figure 30 Henon Map. Source Own.

Henon approached the Lorenz system in a different way. Instead of tackling the Lorenz system directly, he sought a mapping that captures its essential features but which also had an adjustable amount of dissipation. Rather than studying differential equations, Henon chose to study mappings as maps are faster to simulate and their solutions can be followed more accurately and for

[1] An attracting set that has zero measure in the embedding phase space and has fractal dimension. Trajectories within a strange attractor appear to skip around randomly.

a longer time.

2.3.1 Structure of the Map

The Henon Map presents captures several essential properties of the Lorenz system. Hereunder are some of its properties:

- *The Henon map is invertible.* This is counterpart of the Lorenz system in which there is a unique trajectory through each point in phase space. T^{-1} exists FOr all $b \neq 0$, where T is the Henon map.
- *The Henon map is dissipative.* It contracts areas, and does so at the same rate everywhere in phase space. This property is the analog of constant negative divergence in the Lorenz system.
- *For certain parameter values, the Henon map has a trapping region.* This is, there is a region R that gets mapped inside itself. As in the Lorenz system, the strange attractor is enclosed in the trapping region.
- *Some trajectories of the Henon map escape to infinity.* In contrast, all trajectories of the Lorenz system are bounded; they all eventually enter and stay inside a certain large ellipsoid. This happens when the quadratic term dominates and repels orbits to infinity.

Regarding the parameters of the Henon Map, b should not be too close to since the area contraction will be excessive and the fine structure of the attractor will be invisible. Nevertheless, it should not be too large as well as the folding would not be strong enough. Thus, b is the parameter that controls both the dissipation and produces extra folding. Henon recommends $b = 0.3$.

On the other hand, if parameter a is either too small or too large, all trajectories escape to infinity and there is no attractor in these cases (in the logistic map, for $0 \leq r \leq 4$ all trajectories escape to infinity). For intermediate values of a, initial conditions dictates whether trajectories escape to infinity or approach an attractor. If a still increases, the attractor changes from a stable fixed point to a stable 2-cycle and the system undergoes a period-doubling route to chaos. Henon pickes $a = 1.4$.

Figure 30 shows the direct representation of the fractal structure of a strange attractor. The attractor is bent like a boomerang and is made of many parallel curves if zoomed. Taking a closer look, there seem to be 6 parallel curves that are grouped 1, 2, 3. And those curves themselves are made of smaller curves. The self-similarity continues to arbitrarily small scales.

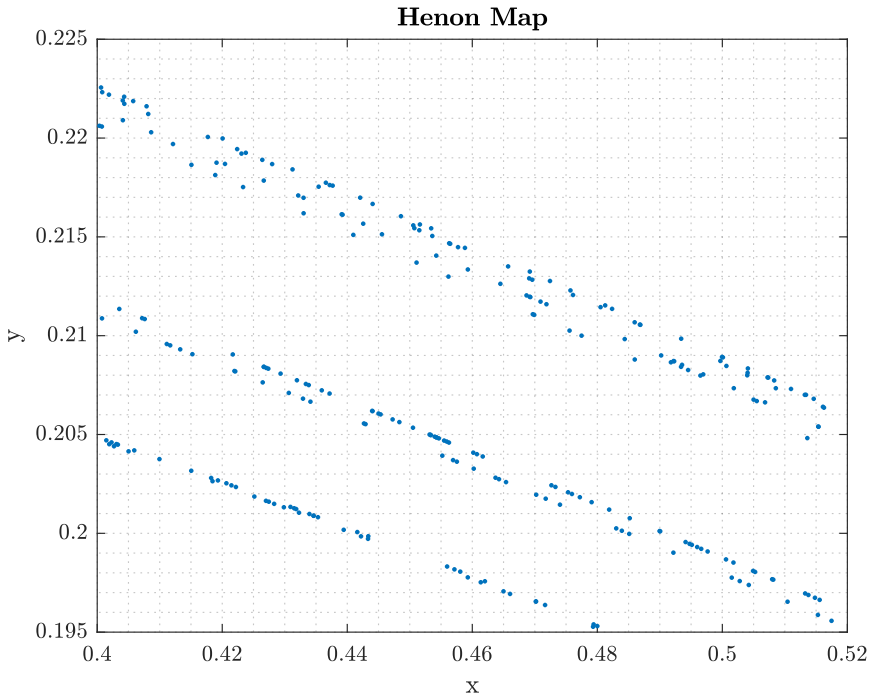


Figure 31 Henon Map (zoom). Source Own.

2.3.2 Procurement of the Map

As regards the mechanism to produce the graph, Henon arrived at this map by an elegant reasoning. To simulate the stretching and folding that occurs in the Lorenz System. Thereby, he considered a chain of transformations:

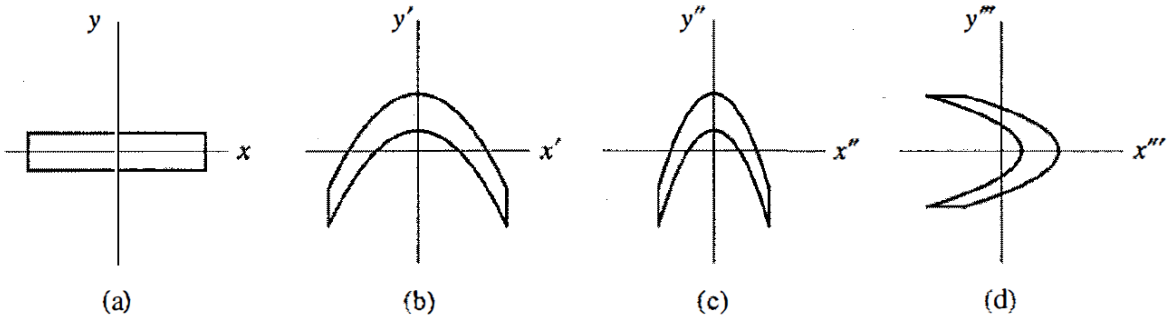


Figure 32 Henon Map Chain of transformations. Source Own.

1. First, start with a rectangular region elongated along the x-axis (Figure 32 a) and stretch and fold the rectangle by applying the following transformation (primes denote iteration, not differentiation):
$$T' : x' = x \quad y' = 1 + y - ax^2 \tag{34}$$

2. Next, the bottom and top of the rectangle gets mapped to parabolas (see Figure 32 b). Parameter a controls the folding. Now, fold the region even more by contracting it along

the x-axis.

$$T'' : x'' = bx' \quad y'' = y' \tag{35}$$

where $-1 < b < 1$. This produces Figure 32 c.

3. Finally, by reflecting across the line $y = x$ it shall come back to the orientation along x-axis.

$$T''' : x''' = y'' \quad y''' = x'' \tag{36}$$

Then the composite transformation $T\,T'''\,T''\,T'$ yields the Henon mapping, where the notation (x_n, y_n) is for (x, y) and (x_{n+1}, y_{n+1}) for (x''', y''') .

2.4 Rossler system

The Rōssler system is defined by the following equations:

$$\begin{aligned} \dot{x} &= -y - z \\ \dot{y} &= x + ay \\ \dot{z} &= b + 2(x - c) \end{aligned} \tag{37}$$

Consider this system for $b = 2, c = 4$ and a increasing slowly from 0 to 0.4 in small steps. For each a, plot the evolution in time for one of the variables and its Power Spectrum. **Comment** the results.

Rössler system are defined by a set of three differential equations. These equations define a continuous-time dynamical system that exhibits chaotic dynamics associated with the fractal properties of the Rössler attractor. Besides, the system of three differential equations presents a simpler strange attractor than Lorenz's. This is due to the fact that the Rössler system has only one quadratic non-linearity xz.

Rössler are a set of a simpler Lorenz equations in the sense that there are only one non linear term in thew third equation. This system also has an attractor but slightly different from the Lorenz butterfly. In particular, the flow is actually not confined to a folded 2D surface, but rather to a folded disk of finite width.

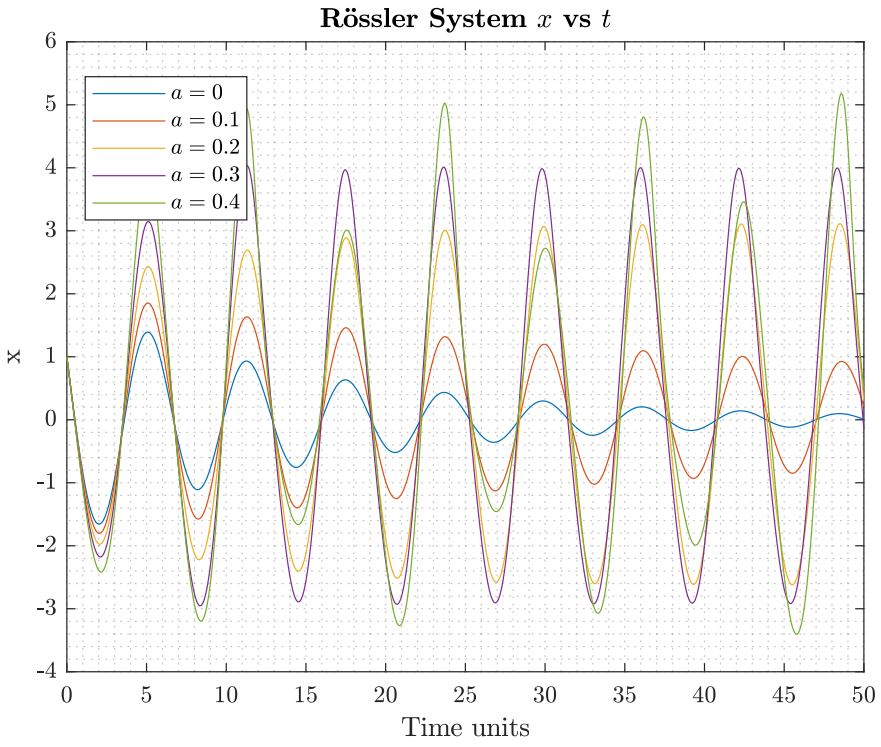


Figure 33 Rössler System x vs t. Source Own.

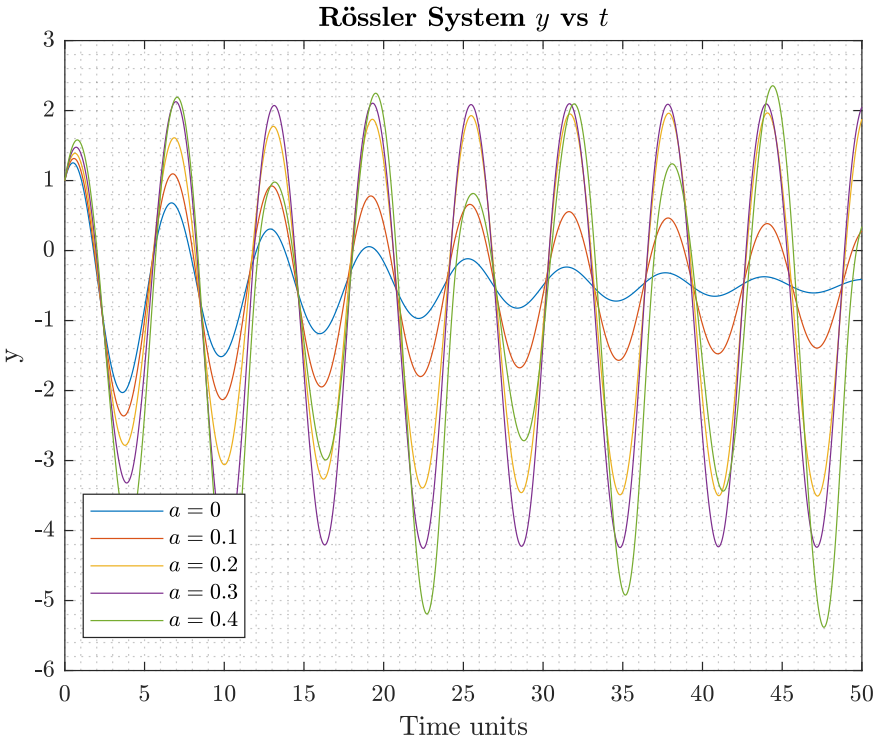


Figure 34 Rössler System y vs t. Source Own.

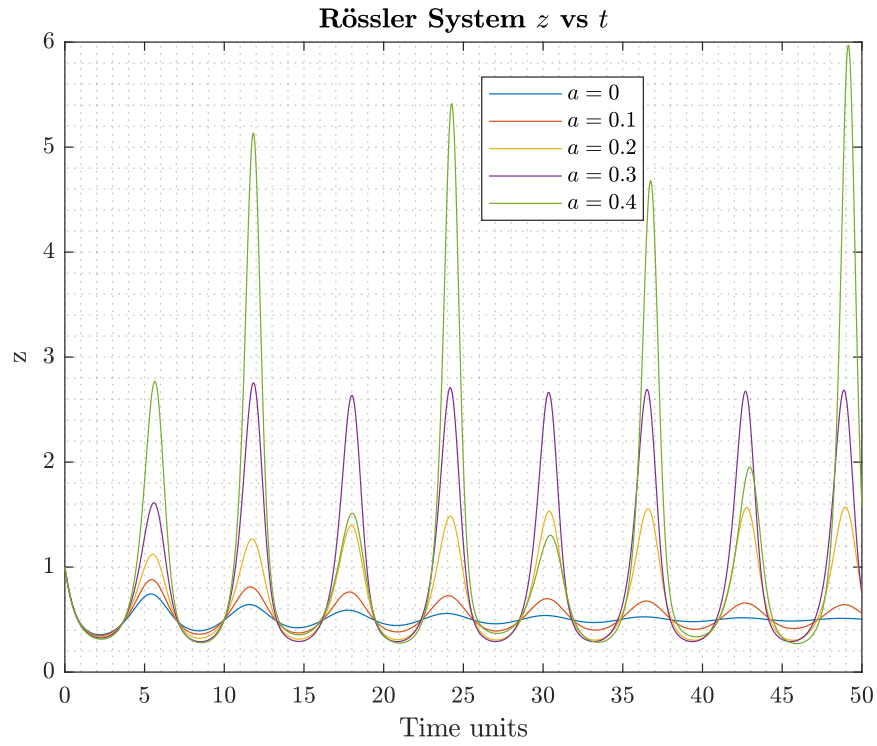


Figure 35 Rössler System z vs t. Source Own.

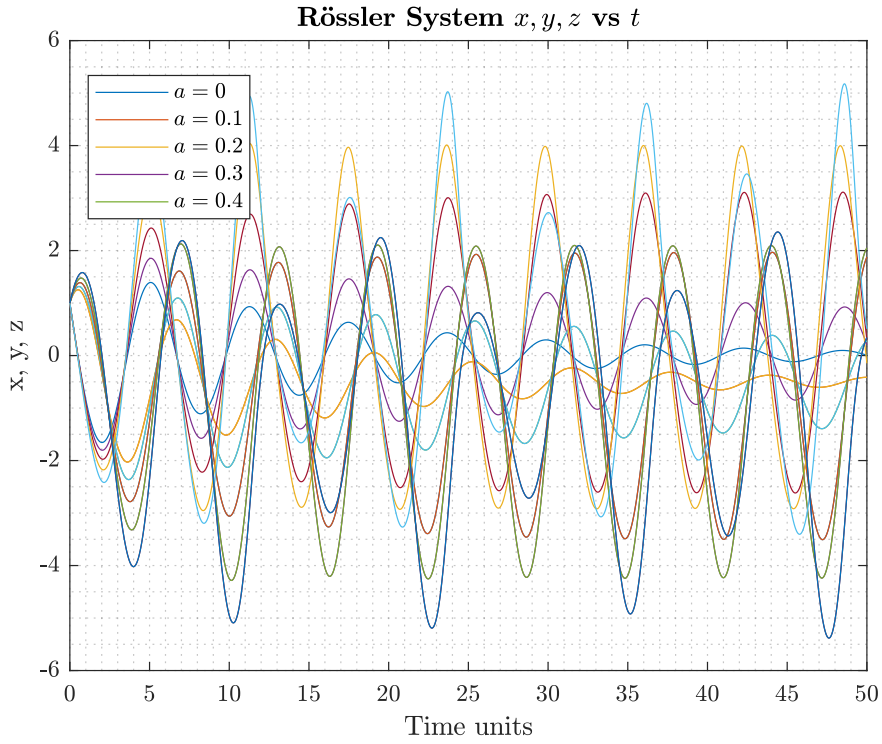


Figure 36 Rössler System x, y, z vs t. Source Own.

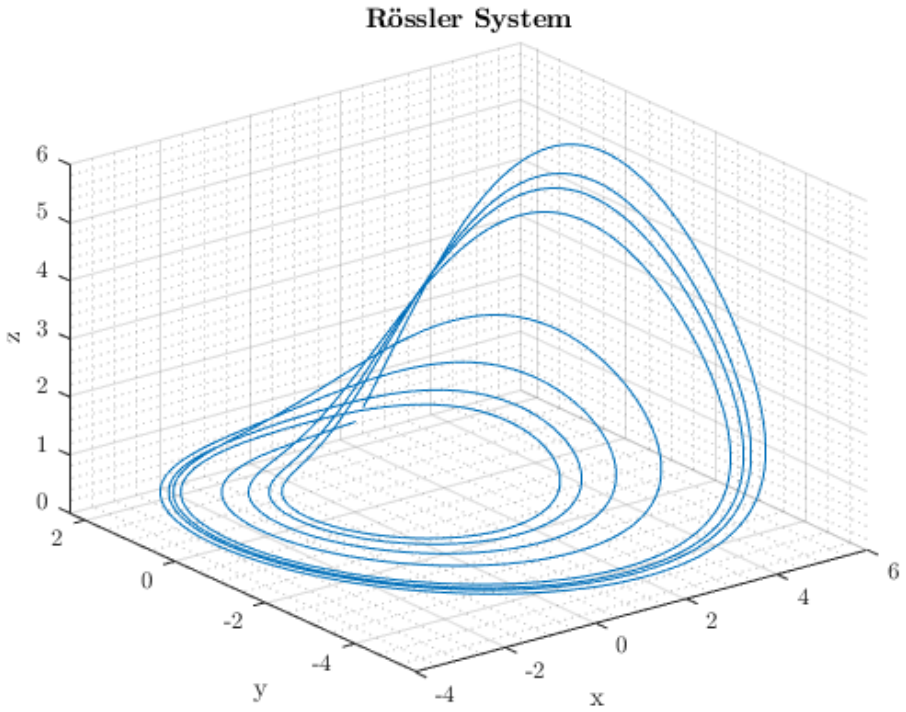


Figure 37 Rössler System 3D plot for $a = 0.4$. Source Own.

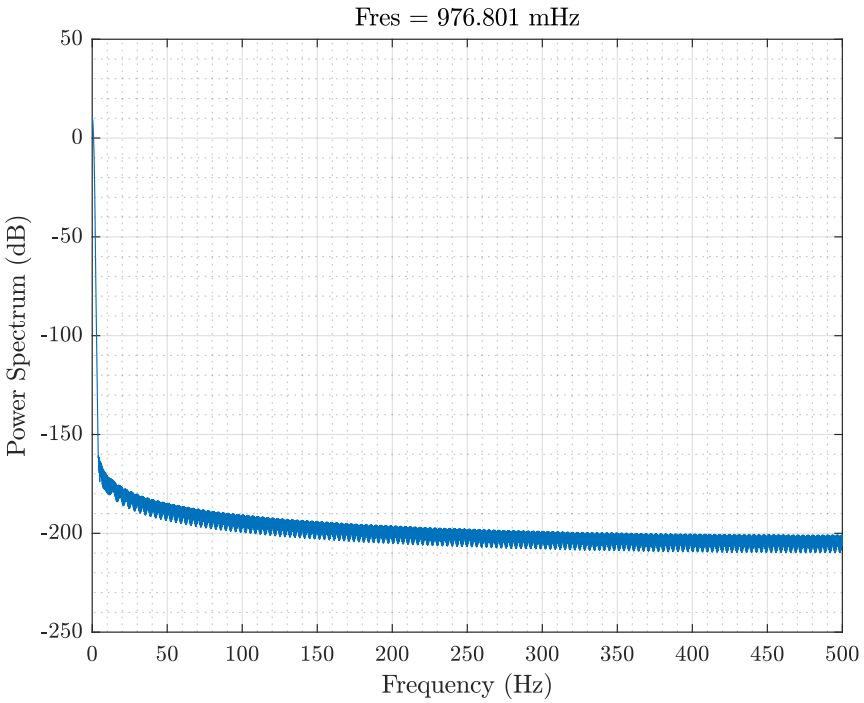


Figure 38 Rössler System power spectrum for x variable with $a = 0.4$. Source Own.

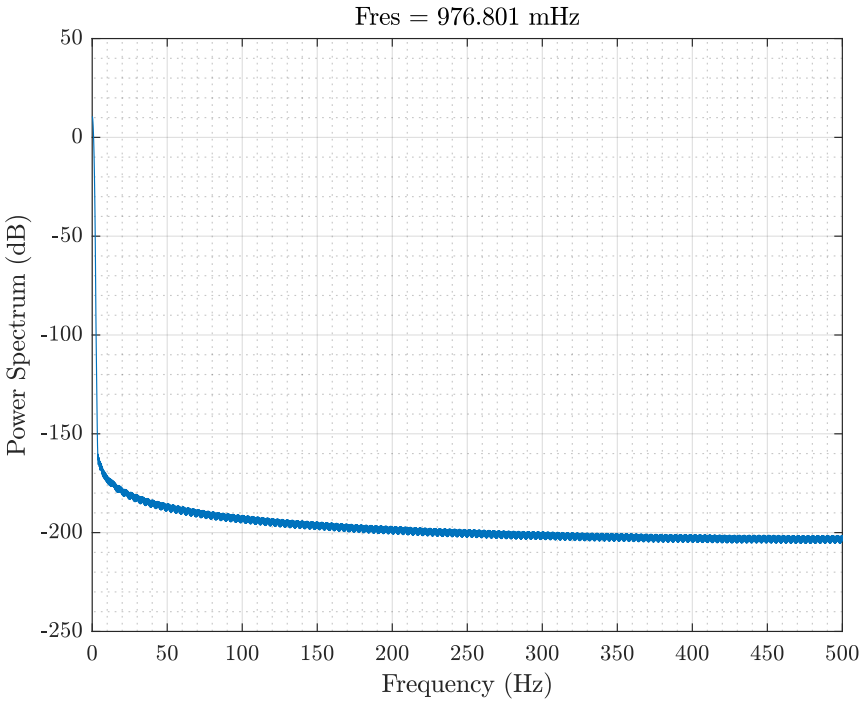


Figure 39 Rössler System power spectrum for y variable with $a = 0.4$. Source Own.

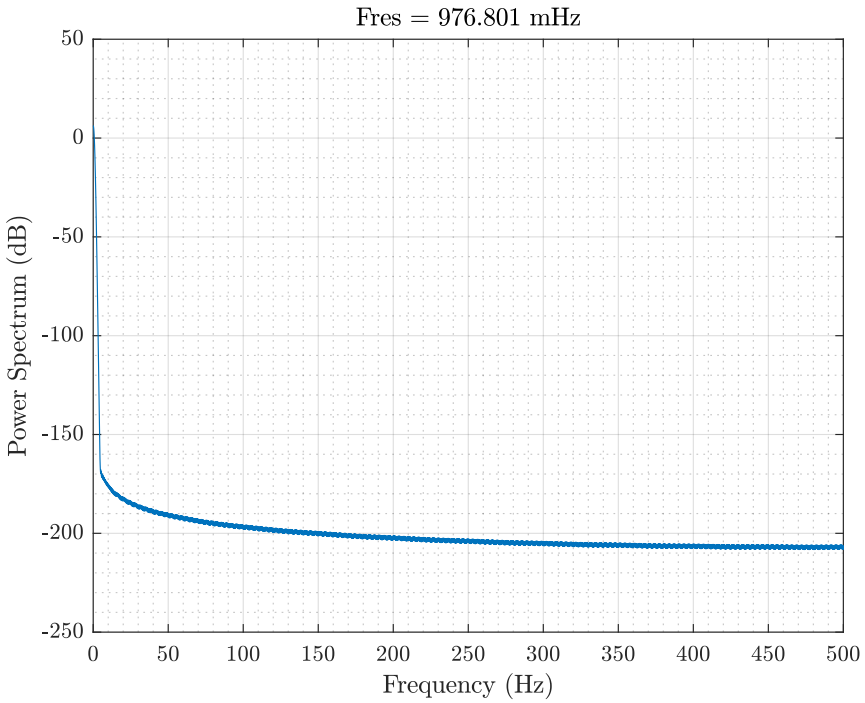


Figure 40 Rössler System power spectrum for z variable with $a = 0.4$. Source Own.

Now, taking a 3D plot of all Rossler system for the different values of a:

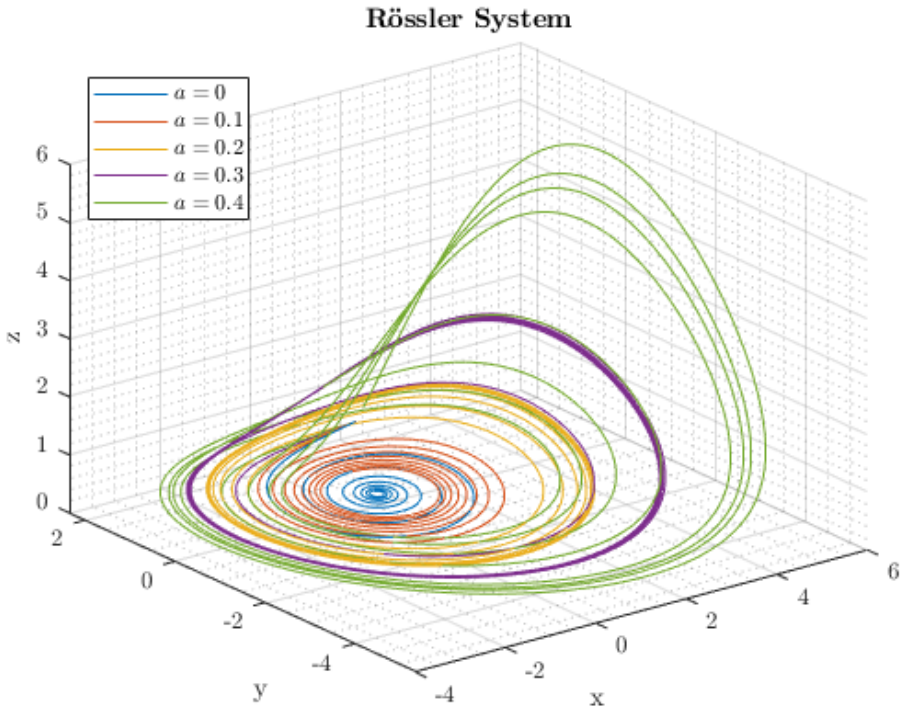


Figure 41 Rössler System 3D plot. Source Own.

Every cross-section through the flow is therefore two-dimensional (rather than one-dimensional). It assumes the form of a horseshoe between one transition and the next. This becomes more evident if one follows the course of one rectangular cross-section as it is "stretched;; and then "folded" before it is mapped back onto itself.

In fact, it is this stretching and folding action that makes the motion chaotic, nearby points can move very far from each other.

Numerical integration shows that this system has a strange attractor. Notice how neighbouring trajectories separate by spiraling out, i.e. "stretching" and then cross without intersecting by using the third dimension z to circulate back near their starting point, also known as "reinjection".

This explains why three dimensions are needed for a flow to be chaotic.

By perusing Figure 41, in one direction there's compression towards the attractor while in the other direction there's divergence along the attractor.

Taking a *Poincaré section* of the attractor, by slicing the attractor with a plane and exposing its cross section. Consequently, if a further one-dimensional slice is taken (*Lorenz section*), there are an infinite set of points separated by gaps of various sizes. This patter of dots and gaps is a topological Cantor set.

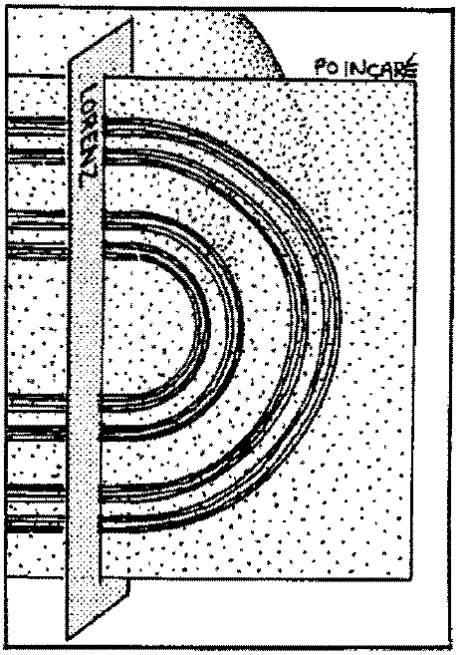


Figure 42 Rössler System 3D plot. Source Own.

The power spectrum $S_{xx}(f)$ of a time series $x(t)$ describes the distribution of power into frequency components composing that signal. According to Fourier analysis, any physical signal can be decomposed into a number of discrete frequencies, or a spectrum of frequencies over a continuous range. Thus, an statistical average of a signal analyzed in frequency space is known as its spectrum.

The spectrum of a signal indicates for which frequencies are the one that dominates over the rest of the signals. As seen in Figures 43, 44 45, the maximum resonance frequency always happens near the 1 Hz frequency. Then the power spectrum decays rapidly and tends to a constant -200 dB.

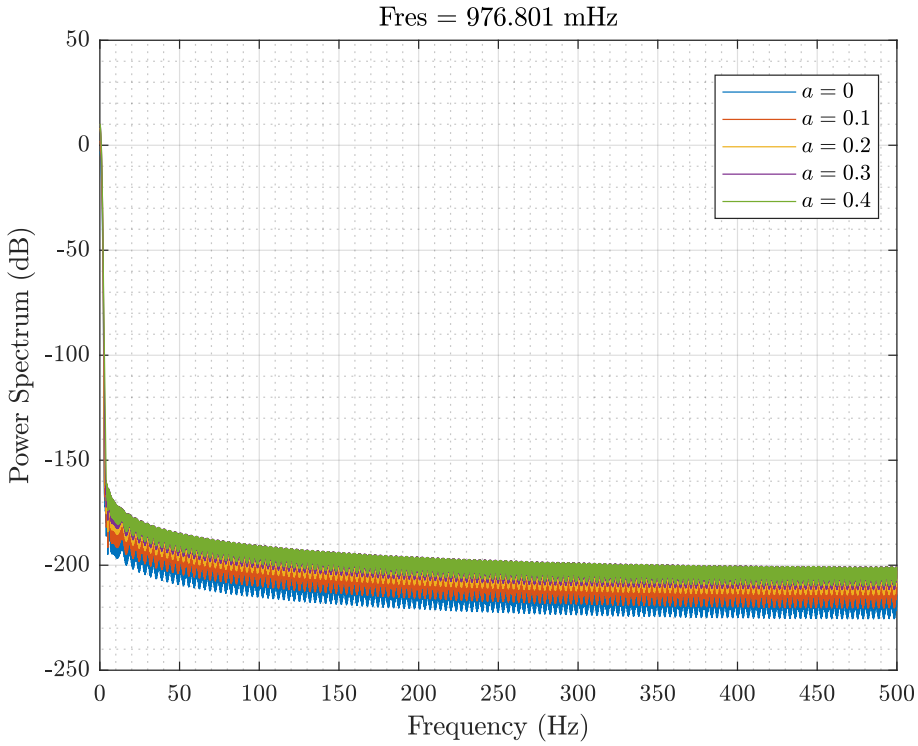


Figure 43 Power spectrum for x. Source Own.

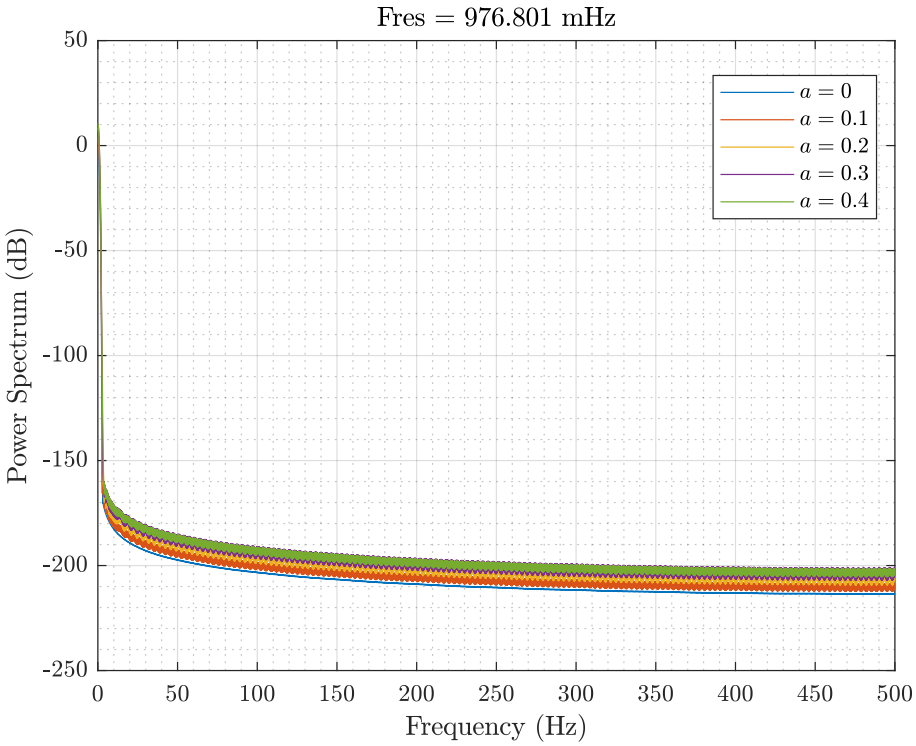


Figure 44 Power spectrum for y. Source Own.

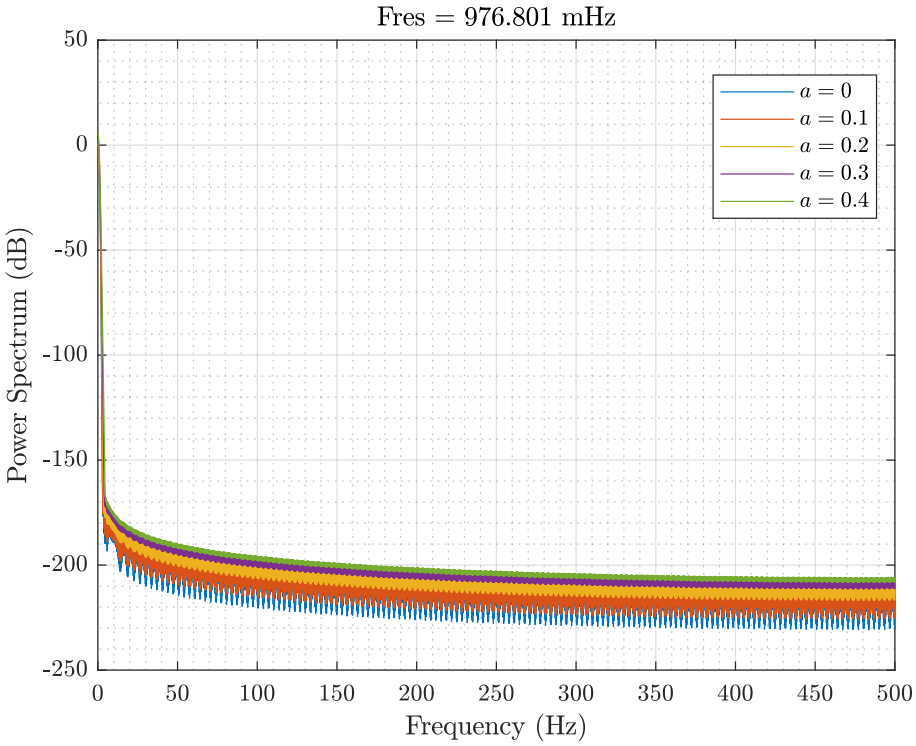


Figure 45 Power spectrum for z. Source Own.

3 Matlab Codes (Runge Kutta 4th Order Method)

3.1 2D System Exercises

3.1.1 Proof

N/A

3.1.2 2D Linear System

```
%% Exercise 2

%-------------------------------------------------------------------------%
% Two-dimensional linear systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Function handle
a = 1;
b = 2;
c = 3;
d = 1;

% Numerical data
h = 0.001; % Time steps of RK4
t_final = 10; % time units
N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil

% Declaration of solution vectors
t = 0:h:t_final;
x = zeros(1, length(t));
y = zeros(1, length(t));

% Initial conditions
t(1) = 0; % We begin at t=0 s
x(1) = -1.25;
y(1) = +1.0;

% Function handle
f1 = @(t,x,y) a*x + b*y;
f2 = @(t,x,y) c*x + d*y;

% RK4 update loop
for i=1:N_steps

    % Update function f1 and f2
    % Compute coefficients sub 1
    k1_f1 = f1(t(i)          ,x(i)                    ,y(i)                 );
    k1_f2 = f2(t(i)          ,x(i)                    ,y(i)                 );

    % Compute coefficients sub 2
    k2_f1 = f1(t(i) + h/2    ,x(i) + k1_f1*h/2        ,y(i) + k1_f2*h/2     );
    k2_f2 = f2(t(i) + h/2    ,x(i) + k1_f1*h/2        ,y(i) + k1_f2*h/2     );

```

```
58      % Compute coefficients sub 3
59      k3_f1 = f1(t(i) + h/2   ,x(i) + k2_f1*h/2       ,y(i) + k2_f2*h/2   );
60      k3_f2 = f2(t(i) + h/2   ,x(i) + k2_f1*h/2       ,y(i) + k2_f2*h/2   );
61
62      % Compute coefficients sub 3
63      k4_f1 = f1(t(i) + h     ,x(i) + k3_f1*h         ,y(i) + k3_f2*h     );
64      k4_f2 = f2(t(i) + h     ,x(i) + k3_f1*h         ,y(i) + k3_f2*h     );
65
66      % Compute next step
67      x(i+1) = x(i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
68      y(i+1) = y(i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
69
70  end
71
72  plot_pdf = figure(1);
73  plot(t,x,t,y);
74  xlabel('Time units')
75  ylabel('Value')
76  title('\textbf{Two-dimensional linear systems}')
77  legend('x','y','location','southwest')
78  box on
79  grid minor
80  hold off;
81
82
83  % Save pdf
84  set(plot_pdf, 'Units', 'Centimeters');
85  pos = get(plot_pdf, 'Position');
86  set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
87      'PaperSize',[pos(3), pos(4)]);
88  print(plot_pdf, 'exercise2_2DLinearSystem.pdf', '-dpdf', '-r0');
89
90  % Save png
91  print(gcf,'exercise2_1_2DLinearSystem.png','-dpng','-r600');
92
93
94  plot_pdf2 = figure(2);
95  plot(x,y);
96  xlabel('x')
97  ylabel('y')
98  title('\textbf{Two-dimensional linear systems}')
99  box on
100 grid minor
101
102 % Save pdf
103 set(plot_pdf2, 'Units', 'Centimeters');
104 pos = get(plot_pdf2, 'Position');
105 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
106     'PaperSize',[pos(3), pos(4)]);
107 print(plot_pdf2, 'exercise2_2DLinearSystem_xy.pdf', '-dpdf', '-r0');
108
109 % Save png
110 print(gcf,'exercise2_2DLinearSystem_xy.png','-dpng','-r600');
```

```
1  %% Exercise 2
2
3  %-------------------------------------------------------------------------%
4  % Two-dimensional linear systems
5  %-------------------------------------------------------------------------%
```

```
6
7  % Date: 09/03/2021
8  % Author/s: Yi Qiang Ji Zhang
9  % Subject: Nonlinear Systems, Chaos and Control in Engineering
10 % Professor: Antonio Pons & Cristina Masoller
11
12 % Clear workspace, command window and close windows
13 clear;
14 close all;
15 clc;
16 % Set interpreter to latex
17 set(groot,'defaultAxesTickLabelInterpreter','latex');
18 set(groot,'defaulttextinterpreter','latex');
19 set(groot,'defaultLegendInterpreter','latex');
20
21 % Function handle
22 a = 1;
23 b = -2;
24 c = 3;
25 d = 1;
26
27 % Numerical data
28 h = 0.001; % Time steps of RK4
29 t_final = 10; % time units
30 N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil
31
32 % Declaration of solution vectors
33 t = 0:h:t_final;
34 x = zeros(1, length(t));
35 y = zeros(1, length(t));
36
37 % Initial conditions
38 t(1) = 0; % We begin at t=0 s
39 x(1) = 0.2;
40 y(1) = +1.5;
41
42 % Function handle
43 f1 = @(t,x,y) a*x + b*y;
44 f2 = @(t,x,y) c*x + d*y;
45
46
47 % RK4 update loop
48 for i=1:N_steps
49
50     % Update function f1 and f2
51     % Compute coefficients sub 1
52     k1_f1 = f1(t(i)          ,x(i)                    ,y(i)                 );
53     k1_f2 = f2(t(i)          ,x(i)                    ,y(i)                 );
54
55     % Compute coefficients sub 2
56     k2_f1 = f1(t(i) + h/2    ,x(i) + k1_f1*h/2        ,y(i) + k1_f2*h/2     );
57     k2_f2 = f2(t(i) + h/2    ,x(i) + k1_f1*h/2        ,y(i) + k1_f2*h/2     );
58
59     % Compute coefficients sub 3
60     k3_f1 = f1(t(i) + h/2    ,x(i) + k2_f1*h/2        ,y(i) + k2_f2*h/2     );
61     k3_f2 = f2(t(i) + h/2    ,x(i) + k2_f1*h/2        ,y(i) + k2_f2*h/2     );
62
63     % Compute coefficients sub 3
64     k4_f1 = f1(t(i) + h      ,x(i) + k3_f1*h          ,y(i) + k3_f2*h       );
65     k4_f2 = f2(t(i) + h      ,x(i) + k3_f1*h          ,y(i) + k3_f2*h       );
66
```

```
67      % Compute next step
68      x(i+1) = x(i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
69      y(i+1) = y(i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
70
71  end
72
73  % Perturbation
74  plot_pdf = figure(1);
75  plot(t,x,t,y);
76  xlabel('Time units')
77  ylabel('Value')
78  title('\textbf{Two-dimensional linear systems}')
79  legend('x','y','location','southwest')
80  box on
81  grid minor
82  hold off;
83
84
85  % Save pdf
86  set(plot_pdf, 'Units', 'Centimeters');
87  pos = get(plot_pdf, 'Position');
88  set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
89      'PaperSize',[pos(3), pos(4)]);
90  print(plot_pdf, 'exercise2_1_2DLinearSystem.pdf', '-dpdf', '-r0');
91
92  % Save png
93  print(gcf,'exercise2_1_2DLinearSystem.png','-dpng','-r600');
94
95
96
97  plot_pdf2 = figure(2);
98  plot(x,y);
99  xlabel('x')
100 ylabel('y')
101 title('\textbf{Two-dimensional linear systems}')
102 box on
103 grid minor
104
105 % Save pdf
106 set(plot_pdf2, 'Units', 'Centimeters');
107 pos = get(plot_pdf2, 'Position');
108 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
109     'PaperSize',[pos(3), pos(4)]);
110 print(plot_pdf2, 'exercise2_1_2DLinearSystem_xy.pdf', '-dpdf', '-r0');
111
112 % Save png
113 print(gcf,'exercise2_1_2DLinearSystem_xy.png','-dpng','-r600');
```

```
1  %% Exercise 2
2
3  %-----------------------------------------------------------------------%
4  % Two-dimensional linear systems
5  %-----------------------------------------------------------------------%
6
7  % Date: 09/03/2021
8  % Author/s: Yi Qiang Ji Zhang
9  % Subject: Nonlinear Systems, Chaos and Control in Engineering
10 % Professor: Antonio Pons & Cristina Masoller
11
```

```
% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Function handle
a = 1;
b = -2;
c = 3;
d = 1;


% Numerical data
h = 0.001; % Time steps of RK4
t_final = 2; % time units
N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil

% Declaration of solution vectors
t = 0:h:t_final;
x = zeros(1, length(t));
y = zeros(1, length(t));

% Initial conditions
t(1) = 0; % We begin at t=0 s
x(1) = 0.2;
y(1) = +1.5;

% Function handle
f1 = @(t,x,y) a*x + b*y;
f2 = @(t,x,y) c*x + d*y;


% Initial conditions
% x0 = -1.25;
% y0 = +1.0;
% x0 = 0.20;
% y0 = +1.5;
x0 = [-1.25 1.0 -1.05];
y0 = [0.2 1.5 +2.5];


% For each h (dt)
for j=1:length(x0)
    % Initial conditions
    t(1) = 0; % We begin at t=0 s
    x(j,1) = x0(j);
    y(j,1) = y0(j);

    % RK4 update loop
    for i=1:N_steps

        % Update function f1 and f2
        % Compute coefficients sub 1
        k1_f1 = f1(t(i)         ,x(j,i)                    ,y(j,i) ...
                           );
        k1_f2 = f2(t(i)         ,x(j,i)                    ,y(j,i) ...
                           );
```

```
71
72          % Compute coefficients sub 2
73          k2_f1 = f1(t(i) + h/2   ,x(j,i) + k1_f1*h/2       ,y(j,i) + k1_f2*h/2 ...
                );
74          k2_f2 = f2(t(i) + h/2   ,x(j,i) + k1_f1*h/2       ,y(j,i) + k1_f2*h/2 ...
                );
75
76          % Compute coefficients sub 3
77          k3_f1 = f1(t(i) + h/2   ,x(j,i) + k2_f1*h/2       ,y(j,i) + k2_f2*h/2 ...
                );
78          k3_f2 = f2(t(i) + h/2   ,x(j,i) + k2_f1*h/2       ,y(j,i) + k2_f2*h/2 ...
                );
79
80          % Compute coefficients sub 3
81          k4_f1 = f1(t(i) + h     ,x(j,i) + k3_f1*h         ,y(j,i) + k3_f2*h ...
                  );
82          k4_f2 = f2(t(i) + h     ,x(j,i) + k3_f1*h         ,y(j,i) + k3_f2*h ...
                  );
83
84          % Compute next step
85          x(j,i+1) = x(j,i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
86          y(j,i+1) = y(j,i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
87
88      end
89          plot_pdf = figure(1);
90          plot(t,x(j,:),t,y(j,:));
91          hold on;
92          plot_pdf2 = figure(2);
93          plot(x(j,:),y(j,:));
94          hold on;
95  end
96
97
98  plot_pdf = figure(1);
99  xlabel('Time units')
100 ylabel('Value')
101 title('\textbf{Two-dimensional linear systems}')
102 legend('$x_{01} \ \mathbf{c_1}$','$y_{01} \ \mathbf{c_1}$','$x_{02} \ ...
        \mathbf{c_2}$', ...
103     '$y_{02}  \ \mathbf{c_2}$','$x_{03} \ \mathbf{c_3}$','$y_{03} \ ...
            \mathbf{c_3}$', ...
104     'location','southwest')
105 box on
106 grid minor
107 hold off;
108
109
110 % Save pdf
111 set(plot_pdf, 'Units', 'Centimeters');
112 pos = get(plot_pdf, 'Position');
113 set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
114     'PaperSize',[pos(3), pos(4)]);
115 print(plot_pdf, '2DLinearSystem_2_all_init_cond.pdf', '-dpdf', '-r0');
116
117 % Save png
118 print(gcf,'2DLinearSystem_2_all_init_cond.png','-dpng','-r600');
119
120
121 plot_pdf2 = figure(2);
122 xlabel('x')
123 ylabel('y')
```

```
title('\textbf{Two-dimensional linear systems}')
legend('$\mathbf{c_1}$','$\mathbf{c_2}$', '$\mathbf{c_3}$', ...
    'location','best')
box on
grid minor

% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, '2DLinearSystem_all_init_cond_xy.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'2DLinearSystem_all_init_cond_xy.png','-dpng','-r600');
```

```
%% Exercise 2

%-------------------------------------------------------------------------%
% Two-dimensional linear systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');


%% Comparison

alpha = 3;
beta  = 0.5;

load('Ex2_S1_IC1.mat');
x_S1_IC1 = alpha*x;
y_S1_IC1 = alpha*y;

load('Ex2_S1_IC2.mat');
x_S1_IC2 = beta*x;
y_S1_IC2 = beta*y;

load('Ex2_S2_IC1.mat');
x_S2_IC1 = alpha*x;
y_S2_IC1 = alpha*y;

load('Ex2_S2_IC2.mat');
x_S2_IC2 = beta*x;
y_S2_IC2 = beta*y;

x_S1_compare = x_S1_IC1 + x_S1_IC2;
```

```
y_S1_compare = y_S1_IC1 + y_S1_IC2;
x_S2_compare = x_S2_IC1 + x_S2_IC2;
y_S2_compare = y_S2_IC1 + y_S2_IC2;

load('Ex2_S1_IC3.mat');
x_S1_IC3 = x;
y_S1_IC3 = y;

load('Ex2_S2_IC3.mat');
x_S2_IC3 = x;
y_S2_IC3 = y;

% Time domain

plot_pdf = figure(1);
plot(t,x_S1_IC3,t,y_S1_IC3,t,x_S1_compare,t,y_S1_compare);
grid on;
grid minor;
box on;
title("\textbf{Nonlinear System 1}");
ylabel("Value");
xlabel("Time units");
legend("$x_3$","$y_3$","$\alpha x_1 + \beta x_2$","$\alpha y_1 + \beta y_2$", ...
    'location', 'best');

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'exercise2_system1.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'exercise2_system1.png','-dpng','-r600');


plot_pdf2 = figure(2);
plot(t,x_S2_IC3,t,y_S2_IC3,t,x_S2_compare,t,y_S2_compare);
grid on;
grid minor;
box on;
title("\textbf{Nonlinear System 2}");
ylabel("Value");
xlabel("Time units");
legend("$x_3$","$y_3$","$\alpha x_1 + \beta x_2$","$\alpha y_1 + \beta y_2$", ...
    'location', 'best');

% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, 'exercise2_system2.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'exercise2_system2.png','-dpng','-r600');



% Phase portrait

```

```
plot_pdf3 = figure(3);
plot(x_S1_IC3,y_S1_IC3,x_S1_compare,y_S1_compare);
grid on;
grid minor;
box on;
title("\textbf{Nonlinear System 1}");
ylabel("y");
xlabel("x");
legend("Sum","Combination");


% Save pdf
set(plot_pdf3, 'Units', 'Centimeters');
pos = get(plot_pdf3, 'Position');
set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf3, 'exercise2_system1_combination.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'exercise2_system1_combination.png','-dpng','-r600');


plot_pdf4 = figure(4);
plot(x_S2_IC3,y_S2_IC3,x_S2_compare,y_S2_compare);
grid on;
grid minor;
box on;
title("\textbf{Nonlinear System 2}");
ylabel("y");
xlabel("x");
legend("Sum","Combination");

% Save pdf
set(plot_pdf4, 'Units', 'Centimeters');
pos = get(plot_pdf4, 'Position');
set(plot_pdf4, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf4, 'exercise2_system2_combination.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'exercise2_system2_combination.png','-dpng','-r600');
```

3.1.3 Phase Portrait

```
%% Exercise 3

%-------------------------------------------------------------------------%
% Two-dimensional linear systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');



% Numerical data
h = 0.001; % Time steps of RK4
t_final = 10; % time units
N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil

% Declaration of solution vectors
t = 0:h:t_final;
x = zeros(1, length(t));
y = zeros(1, length(t));

% Function handle
f1 = @(t,x,y) -x + 4*x^3;
f2 = @(t,x,y) -2*x;


% Initial conditions
x0 = [-0.45 -0.40 -0.30 -0.25 -0.10 0.10 0.25 0.30 0.40 0.45];
y0 = [-0.45 -0.40 -0.30 -0.25 -0.10 0.10 0.25 0.30 0.40 0.45];


% For each h (dt)
for j=1:length(x0)
    % Initial conditions
    t(1) = 0; % We begin at t=0 s
    x(j,1) = x0(j);
    y(j,1) = y0(j);

    % RK4 update loop
    for i=1:N_steps

        % Update function f1 and f2
        % Compute coefficients sub 1
        k1_f1 = f1(t(i)         ,x(j,i)                    ,y(j,i) ...
                           );
```

```
56          k1_f2 = f2(t(i)          ,x(j,i)                    ,y(j,i) ...
                            );
57
58          % Compute coefficients sub 2
59          k2_f1 = f1(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                  );
60          k2_f2 = f2(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                  );
61
62          % Compute coefficients sub 3
63          k3_f1 = f1(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                  );
64          k3_f2 = f2(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                  );
65
66          % Compute coefficients sub 3
67          k4_f1 = f1(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                    );
68          k4_f2 = f2(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                    );
69
70          % Compute next step
71          x(j,i+1) = x(j,i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
72          y(j,i+1) = y(j,i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
73
74      end
75          plot_pdf = figure(1);
76          plot(x(j,:),y(j,:));
77          hold on;
78  end
79
80
81  plot_pdf = figure(1);
82  xlabel('x')
83  ylabel('y')
84  title('\textbf{Phase portrait of the Nonlinear System}')
85  % legend('x','y','location','southwest')
86  box on
87  grid minor
88  hold off;
89
90
91  % Save pdf
92  set(plot_pdf, 'Units', 'Centimeters');
93  pos = get(plot_pdf, 'Position');
94  set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
95      'PaperSize',[pos(3), pos(4)]);
96  print(plot_pdf, 'exercise3_phase_portrait.pdf', '-dpdf', '-r0');
97
98  % Save png
99  print(gcf,'exercise3_phase_portrait.png','-dpng','-r600');
```

3.1.4 Direction Field

```
%% Exercise 4

%-------------------------------------------------------------------------%
% Direction Field
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

%% Map

% Field
x = -2.5:0.01:2.5;
y = -2.5:0.01:2.5;

dxdt = zeros(1,length(x));
dydt = zeros(1,length(y));

% Lines
i = 1;
while i ≤ length(x)
    dxdt(i) = x(i)+exp(0);
    dydt(i) = 0;
    i = i+1;
end

% Position of the quiver vectors
pos_x = size(length(x),length(y));
pos_y = size(length(x),length(y));


for i = 1:20:length(x)
   for j = 1:20:length(y)
       pos_x(i,j) = x(i);
       pos_y(i,j) = y(j);
       value_dx_dt(i,j) = x(i) + exp(-y(j));
       value_dy_dt(i,j) = -y(j);
   end
end

% Figure plot
plot_pdf = figure(1);
plot(x,dxdt,'b',y,dydt,'r');
hold on
qv = quiver(pos_x,pos_y,value_dx_dt,value_dy_dt,'c');
set(qv,'AutoScale','on', 'AutoScaleFactor', 100)
```

```
58 xlim([-2.5 1.5])
59 ylim([-1.5 1.5])
60 xlabel('x')
61 ylabel('y')
62 legend("dx/dt = 0","dy/dt = 0", 'location', 'northeast');
63 title('\textbf{Direction Field}');
64 grid on;
65 grid minor;
66 box on;
67
68 % Save pdf
69 set(plot_pdf, 'Units', 'Centimeters');
70 pos = get(plot_pdf, 'Position');
71 set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
72     'PaperSize',[pos(3), pos(4)]);
73 print(plot_pdf, 'direction_field.pdf', '-dpdf', '-r0');
74
75 % Save png
76 print(gcf,'direction_field.png','-dpng','-r600');
77
78 %% Trajectories
79
80
81 % Numerical data
82 h = 0.001; % Time steps of RK4
83 t_final = 2; % time units
84 N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil
85
86 % Declaration of solution vectors
87 t = 0:h:t_final;
88 x = zeros(1, length(t));
89 y = zeros(1, length(t));
90
91
92 % Function handle
93 f1 = @(t,x,y) x + exp(-y);
94 f2 = @(t,x,y) -y;
95
96
97 % Initial conditions
98 x0 = [0 25 50 100];
99 y0 = [0 25 50 100];
100
101
102 % For each h (dt)
103 for j=1:length(x0)
104     % Initial conditions
105     t(1) = 0; % We begin at t=0 s
106     x(j,1) = x0(j);
107     y(j,1) = y0(j);
108
109     % RK4 update loop
110     for i=1:N_steps
111
112         % Update function f1 and f2
113         % Compute coefficients sub 1
114         k1_f1 = f1(t(i)         ,x(j,i)                    ,y(j,i) ...
                            );
115         k1_f2 = f2(t(i)         ,x(j,i)                    ,y(j,i) ...
                            );
116
```

```
117         % Compute coefficients sub 2
118         k2_f1 = f1(t(i) + h/2   ,x(j,i) + k1_f1*h/2       ,y(j,i) + k1_f2*h/2 ...
                );
119         k2_f2 = f2(t(i) + h/2   ,x(j,i) + k1_f1*h/2       ,y(j,i) + k1_f2*h/2 ...
                );
120
121         % Compute coefficients sub 3
122         k3_f1 = f1(t(i) + h/2   ,x(j,i) + k2_f1*h/2       ,y(j,i) + k2_f2*h/2 ...
                );
123         k3_f2 = f2(t(i) + h/2   ,x(j,i) + k2_f1*h/2       ,y(j,i) + k2_f2*h/2 ...
                );
124
125         % Compute coefficients sub 3
126         k4_f1 = f1(t(i) + h     ,x(j,i) + k3_f1*h         ,y(j,i) + k3_f2*h ...
                  );
127         k4_f2 = f2(t(i) + h     ,x(j,i) + k3_f1*h         ,y(j,i) + k3_f2*h ...
                  );
128
129         % Compute next step
130         x(j,i+1) = x(j,i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
131         y(j,i+1) = y(j,i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
132
133     end
134         plot_pdf2 = figure(2);
135         plot(t,x(j,:), 'r', t,y(j,:), 'b');
136         hold on;
137 end
138
139
140 plot_pdf2 = figure(2);
141 xlabel('Time units')
142 ylabel('Value')
143 title('\textbf{Trajectories with various initial conditions}')
144 legend('x','y','location','best')
145 box on
146 grid minor
147 hold off;
148
149
150 % Save pdf
151 set(plot_pdf2, 'Units', 'Centimeters');
152 pos = get(plot_pdf2, 'Position');
153 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
154     'PaperSize',[pos(3), pos(4)]);
155 print(plot_pdf2, 'trajectories.pdf', '-dpdf', '-r0');
156
157 % Save png
158 print(gcf,'trajectories.png','-dpng','-r600');
```

3.2 Chaos Exercises

3.2.1 Lorenz System

```
1 %% Exercise 1
2
3 %-----------------------------------------------------------------------%
4 % Chaotic systems
```

```
5  %------------------------------------------------------------------------%
6
7  % Date: 10/04/2021
8  % Author/s: Yi Qiang Ji Zhang
9  % Subject: Nonlinear Systems, Chaos and Control in Engineering
10 % Professor: Antonio Pons & Cristina Masoller
11
12 % Clear workspace, command window and close windows
13 clear;
14 close all;
15 clc;
16 % Set interpreter to latex
17 set(groot,'defaultAxesTickLabelInterpreter','latex');
18 set(groot,'defaulttextinterpreter','latex');
19 set(groot,'defaultLegendInterpreter','latex');
20
21 % Constants
22 sigma = 10;
23 beta = 8/3;
24 rho_vec = [21 24.15 30];
25 rho = rho_vec(3);
26
27 % Numerical data
28 h = 0.001; % Time steps of RK4
29 t_final = 50; % time units
30 N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil
31
32 % Declaration of solution vectors
33 t = 0:h:t_final;
34 x = zeros(1, length(t));
35 y = zeros(1, length(t));
36 z = zeros(1, length(t));
37
38 % Function handle
39 f1 = @(t,x,y,z) sigma*(y - x);
40 f2 = @(t,x,y,z) rho*x - y -x*z;
41 f3 = @(t,x,y,z) -beta*z + x*y;
42
43
44 % Initial conditions
45 x0 = 0.1:0.1:0.3;
46 y0 = 0.1:0.1:0.3;
47 z0 = 0.1:0.1:0.3;
48
49
50 % For each x0
51 for j=1:length(x0)
52     % Initial conditions
53     t(1) = 0; % We begin at t=0 s
54     x(j,1) = x0(j);
55     y(j,1) = y0(j);
56     z(j,1) = z0(j);
57
58     % RK4 update loop
59     for i=1:N_steps
60
61         % Update function f1 and f2
62         % Compute coefficients sub 1
63         k1_f1 = f1(t(i)          ,x(j,i)                    ,y(j,i) ...
                          ,z(j,i));
```

```
64          k1_f2 = f2(t(i)          ,x(j,i)                    ,y(j,i) ...
                          ,z(j,i));
65          k1_f3 = f3(t(i)          ,x(j,i)                    ,y(j,i) ...
                          ,z(j,i));
66
67          % Compute coefficients sub 2
68          k2_f1 = f1(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                ,z(j,i) + k1_f3*h/2);
69          k2_f2 = f2(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                ,z(j,i) + k1_f3*h/2);
70          k2_f3 = f3(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                ,z(j,i) + k1_f3*h/2);
71
72          % Compute coefficients sub 3
73          k3_f1 = f1(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                ,z(j,i) + k2_f3*h/2);
74          k3_f2 = f2(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                ,z(j,i) + k2_f3*h/2);
75          k3_f3 = f3(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                ,z(j,i) + k2_f3*h/2);
76
77          % Compute coefficients sub 4
78          k4_f1 = f1(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                  ,z(j,i) + k3_f3*h);
79          k4_f2 = f2(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                  ,z(j,i) + k3_f3*h);
80          k4_f3 = f3(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                  ,z(j,i) + k3_f3*h);
81
82          % Compute next step
83          x(j,i+1) = x(j,i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
84          y(j,i+1) = y(j,i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
85          z(j,i+1) = z(j,i) + h/6*(k1_f3 + 2*k2_f3 + 2*k3_f3 + k4_f3);
86
87      end
88      figure(1)
89      plot(t, x(j,:));
90      hold on;
91      figure(2)
92      plot(t, y(j,:));
93      hold on;
94      figure(3)
95      plot(t, z(j,:));
96      hold on;
97      figure(4)
98      plot(x(j,:), z(j,:));
99      hold on;
100 end
101
102
103 str0 = {strcat('$\rho = $' , num2str(rho))};
104 str0 = [str0 , strcat('$\rho = $' , num2str(rho))];
105
106 plot_pdf = figure(1);
107 xlabel('t')
108 ylabel('x')
109 title('\textbf{Lorenz System for}',str0{2})
110 legend('$x_0, y_0, z_0 = 0.1$','$x_0, y_0, z_0 = 0.2$','$x_0, y_0, z_0 = ...
        0.3$',...
111 'location','best')
112 box on
```

```
grid minor
hold off;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'lorenz_system_rho_30_x.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'lorenz_system_rho_30_x.png','-dpng','-r600');


plot_pdf2 = figure(2);
xlabel('t')
ylabel('y')
title('\textbf{Lorenz System for}',str0{2})
legend('$x_0, y_0, z_0 = 0.1$','$x_0, y_0, z_0 = 0.2$','$x_0, y_0, z_0 = ...
    0.3$',...
'location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, 'lorenz_system_rho_30_y.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'lorenz_system_rho_30_y.png','-dpng','-r600');

plot_pdf3 = figure(3);
xlabel('t')
ylabel('z')
title('\textbf{Lorenz System for}',str0{2})
legend('$x_0, y_0, z_0 = 0.1$','$x_0, y_0, z_0 = 0.2$','$x_0, y_0, z_0 = ...
    0.3$',...
'location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf3, 'Units', 'Centimeters');
pos = get(plot_pdf3, 'Position');
set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf3, 'lorenz_system_rho_30_z.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'lorenz_system_rho_30_z.png','-dpng','-r600');


plot_pdf4 = figure(4);
xlabel('x')
ylabel('z')
title('\textbf{Lorenz System Phase Portrait for}',str0{2})
```

```
172 box on
173 grid minor
174 hold off;
175
176 % Save pdf
177 set(plot_pdf4, 'Units', 'Centimeters');
178 pos = get(plot_pdf4, 'Position');
179 set(plot_pdf4, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
180     'PaperSize',[pos(3), pos(4)]);
181 print(plot_pdf4, 'lorenz_system_rho_30_xz.pdf', '-dpdf', '-r0');
182
183 % Save png
184 print(gcf,'lorenz_system_rho_30_xz.png','-dpng','-r600');
```

3.2.2 Maximum Lyapunov Exponent

```
%% Exercise 2

%-------------------------------------------------------------------------%
% Maximum Lyapunov Exponent
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
sigma = 10;
beta = 8/3;
rho_vec = [21 24.15 30];
rho = rho_vec(3);

% Numerical data
h = 0.001; % Time steps of RK4
t_final = 50; % time units
N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil

% Declaration of solution vectors
t = 0:h:t_final;
x = zeros(1, length(t));
y = zeros(1, length(t));
z = zeros(1, length(t));

% Function handle
f1 = @(t,x,y,z) sigma*(y - x);
f2 = @(t,x,y,z) rho*x - y -x*z;
f3 = @(t,x,y,z) -beta*z + x*y;

% Initial conditions
x0 = [0.1 0.105];
y0 = [0.1 0.105];
z0 = [0.1 0.105];


% For each x0
for j=1:length(x0)
    % Initial conditions
    t(1) = 0; % We begin at t=0 s
    x(j,1) = x0(j);
    y(j,1) = y0(j);
    z(j,1) = z0(j);

    % RK4 update loop
```

```
58      for i=1:N_steps
59
60          % Update function f1 and f2
61          % Compute coefficients sub 1
62          k1_f1 = f1(t(i)          ,x(j,i)                    ,y(j,i) ...
                            ,z(j,i));
63          k1_f2 = f2(t(i)          ,x(j,i)                    ,y(j,i) ...
                            ,z(j,i));
64          k1_f3 = f3(t(i)          ,x(j,i)                    ,y(j,i) ...
                            ,z(j,i));
65
66          % Compute coefficients sub 2
67          k2_f1 = f1(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                  ,z(j,i) + k1_f3*h/2);
68          k2_f2 = f2(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                  ,z(j,i) + k1_f3*h/2);
69          k2_f3 = f3(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + k1_f2*h/2 ...
                  ,z(j,i) + k1_f3*h/2);
70
71          % Compute coefficients sub 3
72          k3_f1 = f1(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                  ,z(j,i) + k2_f3*h/2);
73          k3_f2 = f2(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                  ,z(j,i) + k2_f3*h/2);
74          k3_f3 = f3(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + k2_f2*h/2 ...
                  ,z(j,i) + k2_f3*h/2);
75
76          % Compute coefficients sub 4
77          k4_f1 = f1(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                    ,z(j,i) + k3_f3*h);
78          k4_f2 = f2(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                    ,z(j,i) + k3_f3*h);
79          k4_f3 = f3(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + k3_f2*h ...
                    ,z(j,i) + k3_f3*h);
80
81          % Compute next step
82          x(j,i+1) = x(j,i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
83          y(j,i+1) = y(j,i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
84          z(j,i+1) = z(j,i) + h/6*(k1_f3 + 2*k2_f3 + 2*k3_f3 + k4_f3);
85
86      end
87  end
88
89
90  %% Plots
91
92  % X
93  plot_pdf = figure(1);
94  Δ = log(abs(x(2,:)-x(1,:)));
95  plot(t,Δ);
96  hold on;
97
98  init=2000;
99  final=length(t)/3-1000;
100 str0 = {strcat('$\rho = $' , num2str(rho))};
101 str0 = [str0 , strcat('$\rho = $' , num2str(rho))];
102
103 p = polyfit(t(init:final),Δ(init:final),1);
104 f1 = polyval(p,t(init:final));
105 plot(t(init:final),f1,'LineWidth',2);
106
```

```
xlabel('Time units')
ylabel('$\ln{\left||\Δ\right||}$')
title('\textbf{Maximum Lyapunov Exponent for}',str0{2})
str = {strcat('$\lambda = $' , num2str(p(1)))};
str = [str , strcat('$\lambda = $' , num2str(p(1)))];
legend('x',str{2},'location','best');
box on
grid minor
hold off;


% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'MLE_rho_30_x.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'MLE_rho_30_x.png','-dpng','-r600');



% Y
plot_pdf2 = figure(2);
Δ = log(abs(y(2,:)-y(1,:)));
plot(t,Δ);
hold on;

p = polyfit(t(init:final),Δ(init:final),1);
f1 = polyval(p,t(init:final));
plot(t(init:final),f1,'LineWidth',2);

xlabel('Time units')
ylabel('$\ln{\left||\Δ\right||}$')
title('\textbf{Maximum Lyapunov Exponent for}',str0{2})
str = {strcat('$\lambda = $' , num2str(p(1)))};
str = [str , strcat('$\lambda = $' , num2str(p(1)))];
legend('y',str{2},'location','best');
box on
grid minor
hold off;

% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, 'MLE_rho_30_y.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'MLE_rho_30_y.png','-dpng','-r600');


% Z
plot_pdf3 = figure(3);
Δ = log(abs(z(2,:)-z(1,:)));
plot(t,Δ);
hold on;

p = polyfit(t(init:final),Δ(init:final),1);
```

```
168 f1 = polyval(p,t(init:final));
169 plot(t(init:final),f1,'LineWidth',2);
170
171 xlabel('Time units')
172 ylabel('$\ln{\left|\Δ\right|}$')
173 title('\textbf{Maximum Lyapunov Exponent for}',str0{2})
174 str = {strcat('$\lambda = $' , num2str(p(1)))};
175 str = [str , strcat('$\lambda = $' , num2str(p(1)))];
176 legend('z',str{2},'location','best');
177 box on
178 grid minor
179 hold off;
180
181
182 % Save pdf
183 set(plot_pdf3, 'Units', 'Centimeters');
184 pos = get(plot_pdf3, 'Position');
185 set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
186     'PaperSize',[pos(3), pos(4)]);
187 print(plot_pdf3, 'MLE_rho_30_z.pdf', '-dpdf', '-r0');
188
189 % Save png
190 print(gcf,'MLE_rho_30_z.png','-dpng','-r600');
```

3.2.3 Henon Map

```
%% Exercise 3

%-------------------------------------------------------------------------%
% Henon Map
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
a = 1.4;
b = 0.3;

% Initial conditions
x(1) = 0;
y(1) = 0;

% For 10000 iterations
N = 10000;

for i=1:N
    x(i+1) = y(i) + 1 - a*(x(i)^2);
    y(i+1) = b*x(i);
end

% Plot
plot_pdf = figure(1);
plot(x,y,'.');
xlabel('x')
ylabel('y')
title('\textbf{Henon Map}')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'henon_map.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'henon_map.png','-dpng','-r600');
```

3.2.4 Rössler System

```
%% Exercise 4

%-------------------------------------------------------------------------%
% Chaotic systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
a = 0:0.1:0.4;
b = 2;
c = 4;

% Numerical data
h = 0.001; % Time steps of RK4
t_final = 50; % time units
N_steps = ceil(t_final/h); % Number of steps, rounds up with ceil

% Declaration of solution vectors
t = 0:h:t_final;
x = zeros(1, length(t));
y = zeros(1, length(t));
z = zeros(1, length(t));

% Function handle
f1 = @(t,x,y,z,a) -y-z;
f2 = @(t,x,y,z,a) x + a*y;
f3 = @(t,x,y,z,a) b + z*(x - c);


% Initial conditions
x0 = [1];
y0 = [1];
z0 = [1];

% For each a
for l=1:length(a)
    % For each x0
    for j=1:length(x0)
        % Initial conditions
        t(1) = 0; % We begin at t=0 s
        x(j,1) = x0(j);
        y(j,1) = y0(j);
        z(j,1) = z0(j);

```

```
58          % RK4 update loop
59          for i=1:N_steps
60
61              % Update function f1 and f2
62              % Compute coefficients sub 1
63              k1_f1 = f1(t(i)          ,x(j,i)                    ,y(j,i) ...
                                    ,z(j,i), a(l));
64              k1_f2 = f2(t(i)          ,x(j,i)                    ,y(j,i) ...
                                    ,z(j,i), a(l));
65              k1_f3 = f3(t(i)          ,x(j,i)                    ,y(j,i) ...
                                    ,z(j,i), a(l));
66
67              % Compute coefficients sub 2
68              k2_f1 = f1(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + ...
                    k1_f2*h/2    ,z(j,i) + k1_f3*h/2, a(l));
69              k2_f2 = f2(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + ...
                    k1_f2*h/2    ,z(j,i) + k1_f3*h/2, a(l));
70              k2_f3 = f3(t(i) + h/2    ,x(j,i) + k1_f1*h/2        ,y(j,i) + ...
                    k1_f2*h/2    ,z(j,i) + k1_f3*h/2, a(l));
71
72              % Compute coefficients sub 3
73              k3_f1 = f1(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + ...
                    k2_f2*h/2    ,z(j,i) + k2_f3*h/2, a(l));
74              k3_f2 = f2(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + ...
                    k2_f2*h/2    ,z(j,i) + k2_f3*h/2, a(l));
75              k3_f3 = f3(t(i) + h/2    ,x(j,i) + k2_f1*h/2        ,y(j,i) + ...
                    k2_f2*h/2    ,z(j,i) + k2_f3*h/2, a(l));
76              % Compute coefficients sub 4
77              k4_f1 = f1(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + ...
                    k3_f2*h      ,z(j,i) + k3_f3*h, a(l));
78              k4_f2 = f2(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + ...
                    k3_f2*h      ,z(j,i) + k3_f3*h, a(l));
79              k4_f3 = f3(t(i) + h      ,x(j,i) + k3_f1*h          ,y(j,i) + ...
                    k3_f2*h      ,z(j,i) + k3_f3*h, a(l));
80
81              % Compute next step
82              x(j,i+1) = x(j,i) + h/6*(k1_f1 + 2*k2_f1 + 2*k3_f1 + k4_f1);
83              y(j,i+1) = y(j,i) + h/6*(k1_f2 + 2*k2_f2 + 2*k3_f2 + k4_f2);
84              z(j,i+1) = z(j,i) + h/6*(k1_f3 + 2*k2_f3 + 2*k3_f3 + k4_f3);
85
86          end
87          figure(1)
88          plot(t, x(j,:));
89          hold on;
90          figure(2)
91          plot(t, y(j,:));
92          hold on;
93          figure(3)
94          plot(t, z(j,:));
95          hold on;
96          figure(4)
97          plot(t, x(j,:), t, y(j,:),t, y(j,:));
98          hold on;
99          plot_pdf5 = figure(5);
100         plot3(x,y,z);
101         hold on;
102         plot_pdf6 = figure(6);
103         pspectrum(x(j,:),t)
104         hold on;
105         plot_pdf6 = figure(7);
106         pspectrum(y(j,:),t)
```

```
        hold on;
        plot_pdf6 = figure(8);
        pspectrum(z(j,:),t)
        hold on;
    end
end


plot_pdf = figure(1);
xlabel('Time units')
ylabel('x')
title('\textbf{R\"ossler System $x$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'rossler_system_x.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_x.png','-dpng','-r600');


plot_pdf2 = figure(2);
xlabel('Time units')
ylabel('y')
title('\textbf{R\"ossler System $y$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, 'rossler_system_y.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_y.png','-dpng','-r600');

plot_pdf3 = figure(3);
xlabel('Time units')
ylabel('z')
title('\textbf{R\"ossler System $z$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf3, 'Units', 'Centimeters');
pos = get(plot_pdf3, 'Position');
set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
```

```
print(plot_pdf3, 'rossler_system_z.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_z.png','-dpng','-r600');

plot_pdf4 = figure(4);
xlabel('Time units')
ylabel('x, y, z')
title('\textbf{R\"ossler System $x, y, z$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf4, 'Units', 'Centimeters');
pos = get(plot_pdf4, 'Position');
set(plot_pdf4, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf4, 'rossler_system_xyz.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_xyz.png','-dpng','-r600');


plot_pdf5 = figure(5);
xlabel('x')
ylabel('y')
zlabel('z')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
title('\textbf{R\"ossler System}')
grid on;
grid minor

% Save pdf
set(plot_pdf5, 'Units', 'Centimeters');
pos = get(plot_pdf5, 'Position');
set(plot_pdf5, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf5, 'rossler_system_3D_all.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_3D_all.png','-dpng','-r600');


plot_pdf6 = figure(6);
grid minor
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')

% Save pdf
set(plot_pdf6, 'Units', 'Centimeters');
pos = get(plot_pdf6, 'Position');
set(plot_pdf6, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf6, 'rossler_system_power_spec_x_all.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_power_spec_x_all.png','-dpng','-r600');

plot_pdf7 = figure(7);
grid minor
```

```
229 legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
230
231 % Save pdf
232 set(plot_pdf7, 'Units', 'Centimeters');
233 pos = get(plot_pdf7, 'Position');
234
235 set(plot_pdf7, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
236     'PaperSize',[pos(3), pos(4)]);
237 print(plot_pdf7, 'rossler_system_power_spec_y_all.pdf', '-dpdf', '-r0');
238
239 % Save png
240 print(gcf,'rossler_system_power_spec_y_all.png','-dpng','-r600');
241
242 plot_pdf8 = figure(8);
243 grid minor
244 legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
245
246 % Save pdf
247 set(plot_pdf8, 'Units', 'Centimeters');
248 pos = get(plot_pdf8, 'Position');
249
250 set(plot_pdf8, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
251     'PaperSize',[pos(3), pos(4)]);
252 print(plot_pdf8, 'rossler_system_power_spec_z_all.pdf', '-dpdf', '-r0');
253
254 % Save png
255 print(gcf,'rossler_system_power_spec_z_all.png','-dpng','-r600');
```

4 Matlab Codes (Euler Method)

4.1 2D System Exercises

4.1.1 Proof

N/A

4.1.2 2D Linear System

```
%% Exercise 2

%--------------------------------------------------------------------------%
% Two-dimensional linear systems
%--------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Function handle
a = 1;
b = 2;
c = 3;
d = 1;


f1 = @(t,x,y) a*x + b*y;
f2 = @(t,x,y) c*x + d*y;


% 1.1 Numerical data
dt = [0.001]; % Time steps (dt)
t_final = 10; % time units

for j=1:length(dt)
% N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
N_steps(j) = length(0:dt:t_final)-1;
end

% Initial conditions
x0 = -1.25;
y0 = +1;

% For each h (dt)
for j=1:length(dt)
    t(j,1) = 0; % We begin at t=0 s
    x(j,1) = x0; % x(0)
    y(j,1) = y0; % y(0)
        % Euler method's update loop
        for i=1:N_steps(j)
            t(j,i+1) = t(j,i) + dt(j);
            x(j,i+1) = x(j,i) + f1(t(j,i),x(j,i),y(j,i))*dt(j);
            y(j,i+1) = y(j,i) + f2(t(j,i),x(j,i),y(j,i))*dt(j);
        end
        plot(t(j,:),x(j,:),t(j,:),y(j,:));
        hold on;
```

```
58 end
59
60 plot_pdf = figure(1);
61 xlabel('Time units')
62 ylabel('Value')
63 title('\textbf{Two-dimensional linear systems}')
64 legend('x','y','location','southwest')
65 box on
66 grid minor
67 hold off;
68
69
70 % Save pdf
71 set(plot_pdf, 'Units', 'Centimeters');
72 pos = get(plot_pdf, 'Position');
73 set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
74     'PaperSize',[pos(3), pos(4)]);
75 print(plot_pdf, 'exercise2_2DLinearSystem.pdf', '-dpdf', '-r0');
76
77 % Save png
78 print(gcf,'exercise2_1_2DLinearSystem.png','-dpng','-r600');
79
80
81 plot_pdf2 = figure(2);
82 plot(x(j,:),y(j,:));
83 xlabel('x')
84 ylabel('y')
85 title('\textbf{Two-dimensional linear systems}')
86 box on
87 grid minor
88
89 % Save pdf
90 set(plot_pdf2, 'Units', 'Centimeters');
91 pos = get(plot_pdf2, 'Position');
92 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
93     'PaperSize',[pos(3), pos(4)]);
94 print(plot_pdf2, 'exercise2_2DLinearSystem_xy.pdf', '-dpdf', '-r0');
95
96 % Save png
97 print(gcf,'exercise2_2DLinearSystem_xy.png','-dpng','-r600');
```

```
1  %% Exercise 2
2
3  %-------------------------------------------------------------------------%
4  % Two-dimensional linear systems
5  %-------------------------------------------------------------------------%
6
7  % Date: 09/03/2021
8  % Author/s: Yi Qiang Ji Zhang
9  % Subject: Nonlinear Systems, Chaos and Control in Engineering
10 % Professor: Antonio Pons & Cristina Masoller
11
12 % Clear workspace, command window and close windows
13 clear;
14 close all;
15 clc;
16 % Set interpreter to latex
17 set(groot,'defaultAxesTickLabelInterpreter','latex');
18 set(groot,'defaulttextinterpreter','latex');
```

```
19 set(groot,'defaultLegendInterpreter','latex');
20
21 % Function handle
22 a = 1;
23 b = -2;
24 c = 3;
25 d = 1;
26
27
28 f1 = @(t,x,y) a*x + b*y;
29 f2 = @(t,x,y) c*x + d*y;
30
31
32 % 1.1 Numerical data
33 dt = [0.01]; % Time steps (dt)
34 t_final = 10; % time units
35
36 for j=1:length(dt)
37 % N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
38 N_steps(j) = length(0:dt:t_final)-1;
39 end
40
41 % Initial conditions
42 x0 = 0.2;
43 y0 = +1.5;
44
45 % For each h (dt)
46 for j=1:length(dt)
47     t(j,1) = 0; % We begin at t=0 s
48     x(j,1) = x0; % x(0)
49     y(j,1) = y0; % y(0)
50         % Euler method's update loop
51         for i=1:N_steps(j)
52             t(j,i+1) = t(j,i) + dt(j);
53             x(j,i+1) = x(j,i) + f1(t(j,i),x(j,i),y(j,i))*dt(j);
54             y(j,i+1) = y(j,i) + f2(t(j,i),x(j,i),y(j,i))*dt(j);
55         end
56         plot(t(j,:),x(j,:),t(j,:),y(j,:));
57         hold on;
58 end
59
60 plot_pdf = figure(1);
61 xlabel('Time units')
62 ylabel('Value')
63 title('\textbf{Two-dimensional linear systems}')
64 legend('x','y','location','southwest')
65 box on
66 grid minor
67 hold off;
68
69
70 % Save pdf
71 set(plot_pdf, 'Units', 'Centimeters');
72 pos = get(plot_pdf, 'Position');
73 set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
74     'PaperSize',[pos(3), pos(4)]);
75 print(plot_pdf, 'exercise2_1_2DLinearSystem.pdf', '-dpdf', '-r0');
76
77 % Save png
78 print(gcf,'exercise2_1_2DLinearSystem.png','-dpng','-r600');
79
```

```
80
81
82 plot_pdf2 = figure(2);
83 plot(x(j,:),y(j,:));
84 xlabel('x')
85 ylabel('y')
86 title('\textbf{Two-dimensional linear systems}')
87 box on
88 grid minor
89
90 % Save pdf
91 set(plot_pdf2, 'Units', 'Centimeters');
92 pos = get(plot_pdf2, 'Position');
93 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
94     'PaperSize',[pos(3), pos(4)]);
95 print(plot_pdf2, 'exercise2_1_2DLinearSystem_xy.pdf', '-dpdf', '-r0');
96
97 % Save png
98 print(gcf,'exercise2_1_2DLinearSystem_xy.png','-dpng','-r600');
```

```
1  %% Exercise 2
2
3  %-------------------------------------------------------------------------%
4  % Two-dimensional linear systems
5  %-------------------------------------------------------------------------%
6
7  % Date: 09/03/2021
8  % Author/s: Yi Qiang Ji Zhang
9  % Subject: Nonlinear Systems, Chaos and Control in Engineering
10 % Professor: Antonio Pons & Cristina Masoller
11
12 % Clear workspace, command window and close windows
13 clear;
14 close all;
15 clc;
16 % Set interpreter to latex
17 set(groot,'defaultAxesTickLabelInterpreter','latex');
18 set(groot,'defaulttextinterpreter','latex');
19 set(groot,'defaultLegendInterpreter','latex');
20
21 % Function handle
22 a = 1;
23 b = -2;
24 c = 3;
25 d = 1;
26
27
28 f1 = @(t,x,y) a*x + b*y;
29 f2 = @(t,x,y) c*x + d*y;
30
31
32 % 1.1 Numerical data
33 dt = 0.01; % Time steps (dt)
34 t_final = 10; % time units
35
36 for j=1:length(dt)
37 % N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
38 N_steps(j) = length(0:dt:t_final)-1;
39 end
```

```
40
41 % Initial conditions
42 % x0 = -1.25;
43 % y0 = +1.0;
44 % x0 = 0.20;
45 % y0 = +1.5;
46 x0 = [-1.25 1.0 -1.05];
47 y0 = [0.2 1.5 +2.5];
48
49 % For each h (dt)
50 for j=1:length(x0)
51     t(1) = 0; % We begin at t=0 s
52     x(j,1) = x0(j); % x(0)
53     y(j,1) = y0(j); % y(0)
54         % Euler method's update loop
55         for i=1:N_steps
56             t(i+1) = t(i) + dt;
57             x(j,i+1) = x(j,i) + f1(t(i),x(j,i),y(j,i))*dt;
58             y(j,i+1) = y(j,i) + f2(t(i),x(j,i),y(j,i))*dt;
59         end
60         plot_pdf = figure(1);
61         plot(t,x(j,:),t,y(j,:));
62         hold on;
63         plot_pdf2 = figure(2);
64         plot(x(j,:),y(j,:));
65         hold on;
66 end
67
68 plot_pdf = figure(1);
69 xlabel('Time units')
70 ylabel('Value')
71 title('\textbf{Two-dimensional linear systems}')
72 legend('$x_{01} \ \mathbf{c_1}$','$y_{01} \ \mathbf{c_1}$','$x_{02} \ ...
       \mathbf{c_2}$', ...
73     '$y_{02}  \ \mathbf{c_2}$','$x_{03} \ \mathbf{c_3}$','$y_{03} \ ...
          \mathbf{c_3}$', ...
74     'location','southwest')
75 box on
76 grid minor
77 hold off;
78
79
80 % Save pdf
81 set(plot_pdf, 'Units', 'Centimeters');
82 pos = get(plot_pdf, 'Position');
83 set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
84     'PaperSize',[pos(3), pos(4)]);
85 print(plot_pdf, '2DLinearSystem_2_all_init_cond.pdf', '-dpdf', '-r0');
86
87 % Save png
88 print(gcf,'2DLinearSystem_2_all_init_cond.png','-dpng','-r600');
89
90
91 plot_pdf2 = figure(2);
92 xlabel('x')
93 ylabel('y')
94 title('\textbf{Two-dimensional linear systems}')
95 legend('$\mathbf{c_1}$','$\mathbf{c_2}$', '$\mathbf{c_3}$', ...
96     'location','best')
97 box on
98 grid minor
```

```
99
100 % Save pdf
101 set(plot_pdf2, 'Units', 'Centimeters');
102 pos = get(plot_pdf2, 'Position');
103 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
104     'PaperSize',[pos(3), pos(4)]);
105 print(plot_pdf2, '2DLinearSystem_all_init_cond_xy.pdf', '-dpdf', '-r0');
106
107 % Save png
108 print(gcf,'2DLinearSystem_all_init_cond_xy.png','-dpng','-r600');
```

4.1.3 Phase Portrait

```
%% Exercise 3

%-------------------------------------------------------------------------%
% Two-dimensional linear systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Function handle
f1 = @(t,x,y) -x + 4*x^3;
f2 = @(t,x,y) -2*x;


% 1.1 Numerical data
dt = [0.01]; % Time steps (dt)
t_final = 10; % time units

for j=1:length(dt)
% N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
N_steps(j) = length(0:dt:t_final)-1;
end

% Initial conditions
x0 = [-0.45 -0.40 -0.30 -0.25 -0.10 0.10 0.25 0.30 0.40 0.45];
y0 = [-0.45 -0.40 -0.30 -0.25 -0.10 0.10 0.25 0.30 0.40 0.45];

% For each initial condition
for k=1:length(x0)
    % For each h (dt)
    for j=1:length(dt)
        t(j,1) = 0; % We begin at t=0 s
        x(j,1) = x0(k); % x(0)
        y(j,1) = y0(k); % y(0)
            % Euler method's update loop
            for i=1:N_steps(j)
                t(j,i+1) = t(j,i) + dt(j);
                x(j,i+1) = x(j,i) + f1(t(j,i),x(j,i),y(j,i))*dt(j);
                y(j,i+1) = y(j,i) + f2(t(j,i),x(j,i),y(j,i))*dt(j);
            end
            plot(x(j,:),y(j,:));
            hold on;
    end
end

plot_pdf = figure(1);
```

```
58 xlabel('x')
59 ylabel('y')
60 title('\textbf{Phase portrait of the Nonlinear System}')
61 % legend('x','y','location','southwest')
62 box on
63 grid minor
64 hold off;
65
66
67 % Save pdf
68 set(plot_pdf, 'Units', 'Centimeters');
69 pos = get(plot_pdf, 'Position');
70 set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
71     'PaperSize',[pos(3), pos(4)]);
72 print(plot_pdf, 'exercise3_phase_portrait.pdf', '-dpdf', '-r0');
73
74 % Save png
75 print(gcf,'exercise3_phase_portrait.png','-dpng','-r600');
```

4.1.4 Direction Field

```
%% Exercise 4

%-------------------------------------------------------------------------%
% Direction Field
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

%% Map

% Field
x = -2.5:0.01:2.5;
y = -2.5:0.01:2.5;

dxdt = zeros(1,length(x));
dydt = zeros(1,length(y));

% Lines
i = 1;
while i ≤ length(x)
    dxdt(i) = x(i)+exp(0);
    dydt(i) = 0;
    i = i+1;
end

% Position of the quiver vectors
pos_x = size(length(x),length(y));
pos_y = size(length(x),length(y));


for i = 1:20:length(x)
   for j = 1:20:length(y)
       pos_x(i,j) = x(i);
       pos_y(i,j) = y(j);
       value_dx_dt(i,j) = x(i) + exp(-y(j));
       value_dy_dt(i,j) = -y(j);
   end
end

% Figure plot
plot_pdf = figure(1);
plot(x,dxdt,'b',y,dydt,'r');
hold on
qv = quiver(pos_x,pos_y,value_dx_dt,value_dy_dt,'c');
set(qv,'AutoScale','on', 'AutoScaleFactor', 100)
```

```
xlim([-2.5 1.5])
ylim([-1.5 1.5])
xlabel('x')
ylabel('y')
legend("dx/dt = 0","dy/dt = 0", 'location', 'northeast');
title('\textbf{Direction Field}');
grid on;
grid minor;
box on;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'direction_field.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'direction_field.png','-dpng','-r600');

%% Trajectories

% Function handle
f1 = @(t,x,y) x + exp(-y);
f2 = @(t,x,y) -y;

% 1.1 Numerical data
dt = 0.001; % Time steps (dt)
t_final = 2; % time units

for j=1:length(dt)
% N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
N_steps(j) = length(0:dt:t_final)-1;
end


% Initial conditions
x0 = [0 25 50 100];
y0 = [0 25 50 100];

% For each initial condition
for k=1:length(x0)
        t(1) = 0; % We begin at t=0 s
        x(k,1) = x0(k); % x(0)
        y(k,1) = y0(k); % y(0)
            % Euler method's update loop
            for i=1:N_steps(j)
                t(i+1) = t(i) + dt;
                x(k,i+1) = x(k,i) + f1(t(i),x(k,i),y(k,i))*dt;
                y(k,i+1) = y(k,i) + f2(t(i),x(k,i),y(k,i))*dt;
            end
            plot_pdf2 = figure(2);
            plot(t,x(k,:), 'r', t,y(k,:), 'b');
            hold on;
end

plot_pdf2 = figure(2);
xlabel('Time units')
ylabel('Value')
title('\textbf{Trajectories with various initial conditions}')
legend('x','y','location','best')
```

```
box on
grid minor
hold off;


% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, 'trajectories.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'trajectories.png','-dpng','-r600');
```

4.2 Chaos Exercises

4.2.1 Lorenz System

```
%% Exercise 1

%-------------------------------------------------------------------------%
% Chaotic systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
sigma = 10;
beta = 8/3;
rho = [21 24.15 30];


% Function handle
f1 = @(t,x,y,z) sigma*(y - x);
f2 = @(t,x,y,z) rho(3)*x - y -x*z;
f3 = @(t,x,y,z) -beta*z + x*y;


% 1.1 Numerical data
dt = [0.001]; % Time steps (dt)
t_final = 50; % time units

for j=1:length(dt)
% N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
```

```
N_steps(j) = length(0:dt:t_final)-1;
end

% Initial conditions
x0 = 0.1:0.1:0.3;
y0 = 0.1:0.1:0.3;
z0 = 0.1:0.1:0.3;

% For each initial condition
for k=1:length(x0)
    % For each h (dt)
    for j=1:length(dt)
        t(j,1) = 0; % We begin at t=0 s
        x(j,1) = x0(k); % x(0)
        y(j,1) = y0(k); % y(0)
        z(j,1) = z0(k); % y(0)
            % Euler method's update loop
            for i=1:N_steps(j)
                t(j,i+1) = t(j,i) + dt(j);
                x(j,i+1) = x(j,i) + f1(t(j,i),x(j,i),y(j,i),z(j,i))*dt(j);
                y(j,i+1) = y(j,i) + f2(t(j,i),x(j,i),y(j,i),z(j,i))*dt(j);
                z(j,i+1) = z(j,i) + f3(t(j,i),x(j,i),y(j,i),z(j,i))*dt(j);
            end
            figure(1)
            plot(t(j,:), x(j,:));
            hold on;
            figure(2)
            plot(t(j,:), y(j,:));
            hold on;
            figure(3)
            plot(t(j,:), z(j,:));
            hold on;
            figure(4)
            plot(x(j,:), z(j,:));
            hold on;
    end
end

plot_pdf = figure(1);
xlabel('t')
ylabel('x')
title('\textbf{Lorenz System for $\rho = 30$}')
legend('$x_0, y_0, z_0 = 0.1$','$x_0, y_0, z_0 = 0.2$','$x_0, y_0, z_0 = ...
    0.3$',...
'location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'lorenz_system_rho_30_x.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'lorenz_system_rho_30_x.png','-dpng','-r600');


plot_pdf2 = figure(2);
```

```
99  xlabel('t')
100 ylabel('y')
101 title('\textbf{Lorenz System for $\rho = 30$}')
102 legend('$x_0, y_0, z_0 = 0.1$','$x_0, y_0, z_0 = 0.2$','$x_0, y_0, z_0 = ...
        0.3$',...
103 'location','best')
104 box on
105 grid minor
106 hold off;
107
108 % Save pdf
109 set(plot_pdf2, 'Units', 'Centimeters');
110 pos = get(plot_pdf2, 'Position');
111 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
112     'PaperSize',[pos(3), pos(4)]);
113 print(plot_pdf2, 'lorenz_system_rho_30_y.pdf', '-dpdf', '-r0');
114
115 % Save png
116 print(gcf,'lorenz_system_rho_30_y.png','-dpng','-r600');
117
118 plot_pdf3 = figure(3);
119 xlabel('t')
120 ylabel('z')
121 title('\textbf{Lorenz System for $\rho = 30$}')
122 legend('$x_0, y_0, z_0 = 0.1$','$x_0, y_0, z_0 = 0.2$','$x_0, y_0, z_0 = ...
        0.3$',...
123 'location','best')
124 box on
125 grid minor
126 hold off;
127
128 % Save pdf
129 set(plot_pdf3, 'Units', 'Centimeters');
130 pos = get(plot_pdf3, 'Position');
131 set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
132     'PaperSize',[pos(3), pos(4)]);
133 print(plot_pdf3, 'lorenz_system_rho_30_z.pdf', '-dpdf', '-r0');
134
135 % Save png
136 print(gcf,'lorenz_system_rho_30_z.png','-dpng','-r600');
137
138
139 plot_pdf4 = figure(4);
140 xlabel('x')
141 ylabel('z')
142 title('\textbf{Lorenz System Phase Portrait for $\rho = 30$}')
143 box on
144 grid minor
145 hold off;
146
147 % Save pdf
148 set(plot_pdf4, 'Units', 'Centimeters');
149 pos = get(plot_pdf4, 'Position');
150 set(plot_pdf4, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
151     'PaperSize',[pos(3), pos(4)]);
152 print(plot_pdf4, 'lorenz_system_rho_30_xz.pdf', '-dpdf', '-r0');
153
154 % Save png
155 print(gcf,'lorenz_system_rho_30_xz.png','-dpng','-r600');
```

4.2.2 Maximum Lyapunov Exponent

```
%% Exercise 2

%-------------------------------------------------------------------------%
% Maximum Lyapunov Exponent
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
sigma = 10;
beta = 8/3;
rho = [21 24.15 30];


% Function handle
f1 = @(t,x,y,z) sigma*(y - x);
f2 = @(t,x,y,z) rho(3)*x - y -x*z;
f3 = @(t,x,y,z) -beta*z + x*y;


% 1.1 Numerical data
dt = 0.001; % Time steps (dt)
t_final = 50; % time units

for j=1:length(dt)
% N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
N_steps(j) = length(0:dt:t_final)-1;
end

% Initial conditions
x0 = [0.1 0.105];
y0 = [0.1 0.105];
z0 = [0.1 0.105];

% For each initial condition
for k=1:length(x0)
        t(1) = 0; % We begin at t=0 s
        x(k,1) = x0(k); % x(0)
        y(k,1) = y0(k); % y(0)
        z(k,1) = z0(k); % y(0)
            % Euler method's update loop
            for i=1:N_steps(j)
                t(i+1) = t(i) + dt;
                x(k,i+1) = x(k,i) + f1(t(i),x(k,i),y(k,i),z(k,i))*dt;
                y(k,i+1) = y(k,i) + f2(t(i),x(k,i),y(k,i),z(k,i))*dt;
```

```
                z(k,i+1) = z(k,i) + f3(t(i),x(k,i),y(k,i),z(k,i))*dt;
            end
end

%% Plots

% X
plot_pdf = figure(1);
Δ = log(abs(x(2,:)-x(1,:)));
plot(t,Δ);
hold on;

init=2000;
final=length(t)/3-1000;


p = polyfit(t(init:final),Δ(init:final),1);
f1 = polyval(p,t(init:final));
plot(t(init:final),f1,'LineWidth',2);

xlabel('Time units')
ylabel('$\ln{\left||\Δ\right||}$')
title('\textbf{Maximum Lyapunov Exponent for $\rho = 30$}')
str = {strcat('$\lambda = $' , num2str(p(1)))};
str = [str , strcat('$\lambda = $' , num2str(p(1)))];
legend('x',str{2},'location','best');
box on
grid minor
hold off;


% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'MLE_rho_30_x.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'MLE_rho_30_x.png','-dpng','-r600');



% Y
plot_pdf2 = figure(2);
Δ = log(abs(y(2,:)-y(1,:)));
plot(t,Δ);
hold on;

p = polyfit(t(init:final),Δ(init:final),1);
f1 = polyval(p,t(init:final));
plot(t(init:final),f1,'LineWidth',2);

xlabel('Time units')
ylabel('$\ln{\left||\Δ\right||}$')
title('\textbf{Maximum Lyapunov Exponent for $\rho = 30$}')
str = {strcat('$\lambda = $' , num2str(p(1)))};
str = [str , strcat('$\lambda = $' , num2str(p(1)))];
legend('y',str{2},'location','best');
box on
grid minor
```

```
119 hold off;
120
121 % Save pdf
122 set(plot_pdf2, 'Units', 'Centimeters');
123 pos = get(plot_pdf2, 'Position');
124 set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
125     'PaperSize',[pos(3), pos(4)]);
126 print(plot_pdf2, 'MLE_rho_30_y.pdf', '-dpdf', '-r0');
127
128 % Save png
129 print(gcf,'MLE_rho_30_y.png','-dpng','-r600');
130
131
132 % Z
133 plot_pdf3 = figure(3);
134 Δ = log(abs(z(2,:)-z(1,:)));
135 plot(t,Δ);
136 hold on;
137
138 p = polyfit(t(init:final),Δ(init:final),1);
139 f1 = polyval(p,t(init:final));
140 plot(t(init:final),f1,'LineWidth',2);
141
142 xlabel('Time units')
143 ylabel('$\ln{\left|\Δ\right|}$')
144 title('\textbf{Maximum Lyapunov Exponent for $\rho = 30$}')
145 str = {strcat('$\lambda = $' , num2str(p(1)))};
146 str = [str , strcat('$\lambda = $' , num2str(p(1)))];
147 legend('z',str{2},'location','best');
148 box on
149 grid minor
150 hold off;
151
152
153 % Save pdf
154 set(plot_pdf3, 'Units', 'Centimeters');
155 pos = get(plot_pdf3, 'Position');
156 set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
157     'PaperSize',[pos(3), pos(4)]);
158 print(plot_pdf3, 'MLE_rho_30_z.pdf', '-dpdf', '-r0');
159
160 % Save png
161 print(gcf,'MLE_rho_30_z.png','-dpng','-r600');
```

4.2.3 Henon Map

```
%% Exercise 3

%-------------------------------------------------------------------------%
% Henon Map
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
a = 1.4;
b = 0.3;

% Initial conditions
x(1) = 0;
y(1) = 0;

% For 10000 iterations
N = 10000;

for i=1:N
    x(i+1) = y(i) + 1 - a*(x(i)^2);
    y(i+1) = b*x(i);
end

% Plot
plot_pdf = figure(1);
plot(x,y,'.');
xlabel('x')
ylabel('y')
title('\textbf{Henon Map}')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'henon_map.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'henon_map.png','-dpng','-r600');
```

4.2.4 Rössler System

```
%% Exercise 4

%-------------------------------------------------------------------------%
% Chaotic systems
%-------------------------------------------------------------------------%

% Date: 10/04/2021
% Author/s: Yi Qiang Ji Zhang
% Subject: Nonlinear Systems, Chaos and Control in Engineering
% Professor: Antonio Pons & Cristina Masoller

% Clear workspace, command window and close windows
clear;
close all;
clc;
% Set interpreter to latex
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

% Constants
a = 0:0.1:0.4;
b = 2;
c = 4;

% Function handle
f1 = @(t,x,y,z,a) -y-z;
f2 = @(t,x,y,z,a) x + a*y;
f3 = @(t,x,y,z,a) b + z*(x - c);


% Numerical data
dt = [0.001]; % Time steps (dt)
t_final = 50; % time units

for j=1:length(dt)
% N_steps(j) = ceil(t_final/h(j)); % Number of steps, rounds up with ceil
N_steps(j) = length(0:dt:t_final)-1;
end

% Initial conditions
x0 = [1];
y0 = [1];
z0 = [1];

% For each a
for l=1:length(a)
    % For each initial condition
    for k=1:length(x0)
        % For each h (dt)
        for j=1:length(dt)
            t(j,1) = 0; % We begin at t=0 s
            x(j,1) = x0(k); % x(0)
            y(j,1) = y0(k); % y(0)
            z(j,1) = z0(k);
                % Euler method's update loop
                for i=1:N_steps(j)
```

```
                    t(j,i+1) = t(j,i) + dt(j);
                    x(j,i+1) = x(j,i) + ...
                        f1(t(j,i),x(j,i),y(j,i),z(j,i),a(l))*dt(j);
                    y(j,i+1) = y(j,i) + ...
                        f2(t(j,i),x(j,i),y(j,i),z(j,i),a(l))*dt(j);
                    z(j,i+1) = z(j,i) + ...
                        f3(t(j,i),x(j,i),y(j,i),z(j,i),a(l))*dt(j);
                end
                plot_pdf = figure(1);
                plot(t(j,:), x(j,:));
                hold on;
                plot_pdf2 = figure(2);
                plot(t(j,:), y(j,:));
                hold on;
                plot_pdf3 = figure(3);
                plot(t(j,:), z(j,:));
                hold on;
                plot_pdf4 = figure(4);
                plot(t(j,:), x(j,:), t(j,:), y(j,:),t(j,:), y(j,:));
                hold on;
                plot_pdf5 = figure(5);
                plot3(x,y,z);
                hold on;
                plot_pdf6 = figure(6);
                pspectrum(x(k,:),t)
                hold on;
                plot_pdf6 = figure(7);
                pspectrum(y(k,:),t)
                hold on;
                plot_pdf6 = figure(8);
                pspectrum(z(k,:),t)
                hold on;
        end
    end
end

plot_pdf = figure(1);
xlabel('Time units')
ylabel('x')
title('\textbf{R\"ossler System $x$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf, 'Units', 'Centimeters');
pos = get(plot_pdf, 'Position');
set(plot_pdf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf, 'rossler_system_x.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_x.png','-dpng','-r600');


plot_pdf2 = figure(2);
xlabel('Time units')
ylabel('y')
title('\textbf{R\"ossler System $y$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
```

```
box on
grid minor
hold off;

% Save pdf
set(plot_pdf2, 'Units', 'Centimeters');
pos = get(plot_pdf2, 'Position');
set(plot_pdf2, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf2, 'rossler_system_y.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_y.png','-dpng','-r600');

plot_pdf3 = figure(3);
xlabel('Time units')
ylabel('z')
title('\textbf{R\"ossler System $z$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf3, 'Units', 'Centimeters');
pos = get(plot_pdf3, 'Position');
set(plot_pdf3, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf3, 'rossler_system_z.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_z.png','-dpng','-r600');

plot_pdf4 = figure(4);
xlabel('Time units')
ylabel('x, y, z')
title('\textbf{R\"ossler System $x, y, z$ vs $t$}')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
box on
grid minor
hold off;

% Save pdf
set(plot_pdf4, 'Units', 'Centimeters');
pos = get(plot_pdf4, 'Position');
set(plot_pdf4, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
    'PaperSize',[pos(3), pos(4)]);
print(plot_pdf4, 'rossler_system_xyz.pdf', '-dpdf', '-r0');

% Save png
print(gcf,'rossler_system_xyz.png','-dpng','-r600');


plot_pdf5 = figure(5);
xlabel('x')
ylabel('y')
zlabel('z')
legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
title('\textbf{R\"ossler System}')
grid on;
grid minor
```

```
177
178 % Save pdf
179 set(plot_pdf5, 'Units', 'Centimeters');
180 pos = get(plot_pdf5, 'Position');
181 set(plot_pdf5, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
182     'PaperSize',[pos(3), pos(4)]);
183 print(plot_pdf5, 'rossler_system_3D_all.pdf', '-dpdf', '-r0');
184
185 % Save png
186 print(gcf,'rossler_system_3D_all.png','-dpng','-r600');
187
188
189 plot_pdf6 = figure(6);
190 grid minor
191 legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
192
193 % Save pdf
194 set(plot_pdf6, 'Units', 'Centimeters');
195 pos = get(plot_pdf6, 'Position');
196 set(plot_pdf6, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
197     'PaperSize',[pos(3), pos(4)]);
198 print(plot_pdf6, 'rossler_system_power_spec_x_all.pdf', '-dpdf', '-r0');
199
200 % Save png
201 print(gcf,'rossler_system_power_spec_x_all.png','-dpng','-r600');
202
203 plot_pdf7 = figure(7);
204 grid minor
205 legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
206
207 % Save pdf
208 set(plot_pdf7, 'Units', 'Centimeters');
209 pos = get(plot_pdf7, 'Position');
210
211 set(plot_pdf7, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
212     'PaperSize',[pos(3), pos(4)]);
213 print(plot_pdf7, 'rossler_system_power_spec_y_all.pdf', '-dpdf', '-r0');
214
215 % Save png
216 print(gcf,'rossler_system_power_spec_y_all.png','-dpng','-r600');
217
218 plot_pdf8 = figure(8);
219 grid minor
220 legend('$a=0$','$a=0.1$','$a=0.2$','$a=0.3$','$a=0.4$','location','best')
221
222 % Save pdf
223 set(plot_pdf8, 'Units', 'Centimeters');
224 pos = get(plot_pdf8, 'Position');
225
226 set(plot_pdf8, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Centimeters', ...
227     'PaperSize',[pos(3), pos(4)]);
228 print(plot_pdf8, 'rossler_system_power_spec_z_all.pdf', '-dpdf', '-r0');
229
230 % Save png
231 print(gcf,'rossler_system_power_spec_z_all.png','-dpng','-r600');
```

Appendices

A Lorenz System

Below is presented the plots of section 2.1 in higher size:

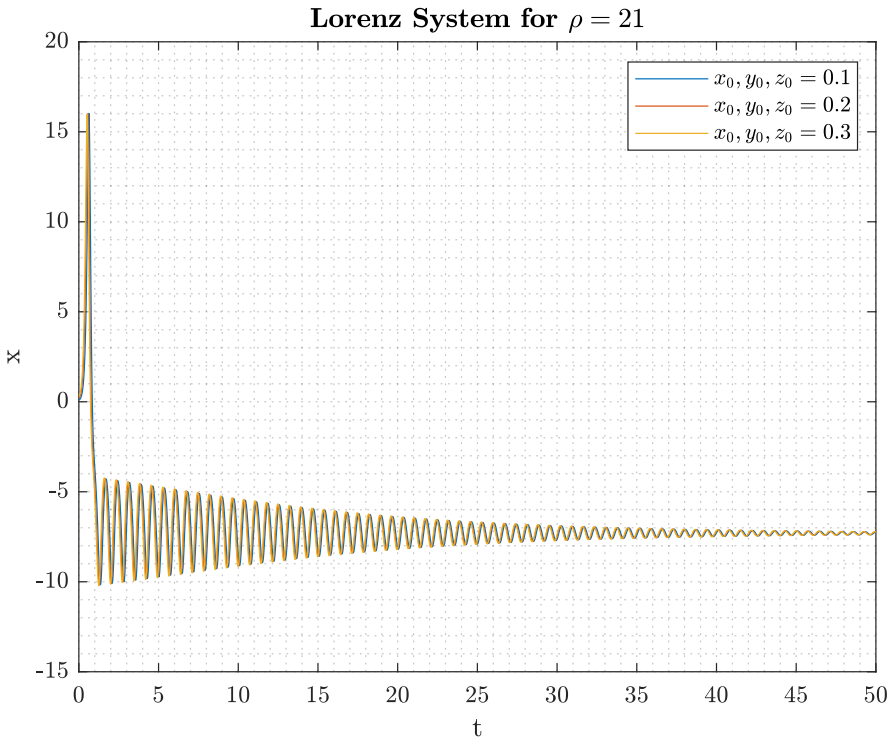


Figure 46 Lorenz System for $\rho = 21$. Source Own.

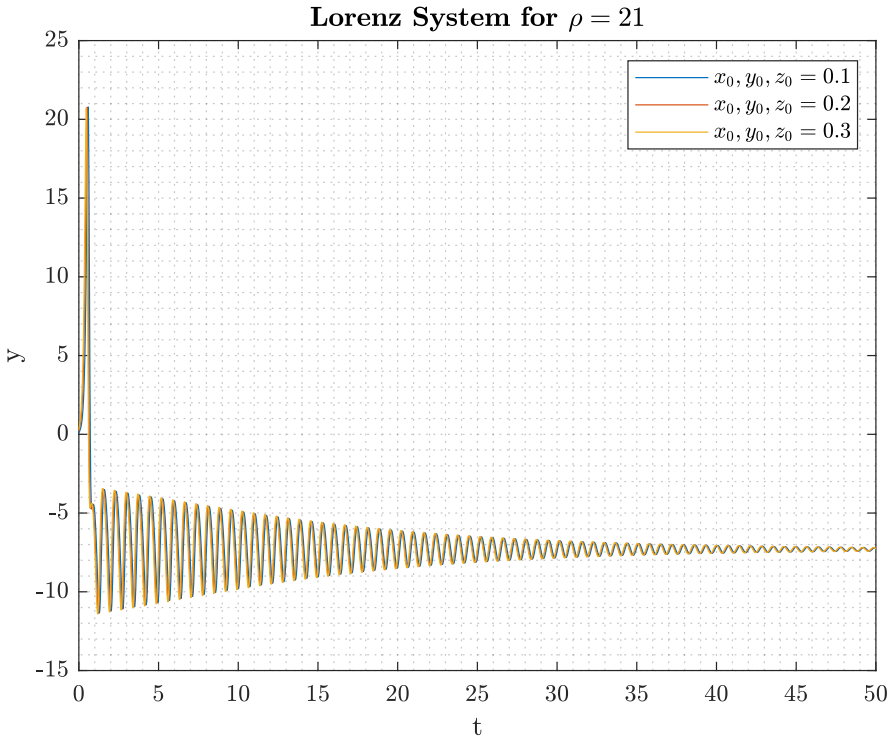


Figure 47 Lorenz System for $\rho = 21$. Source Own.

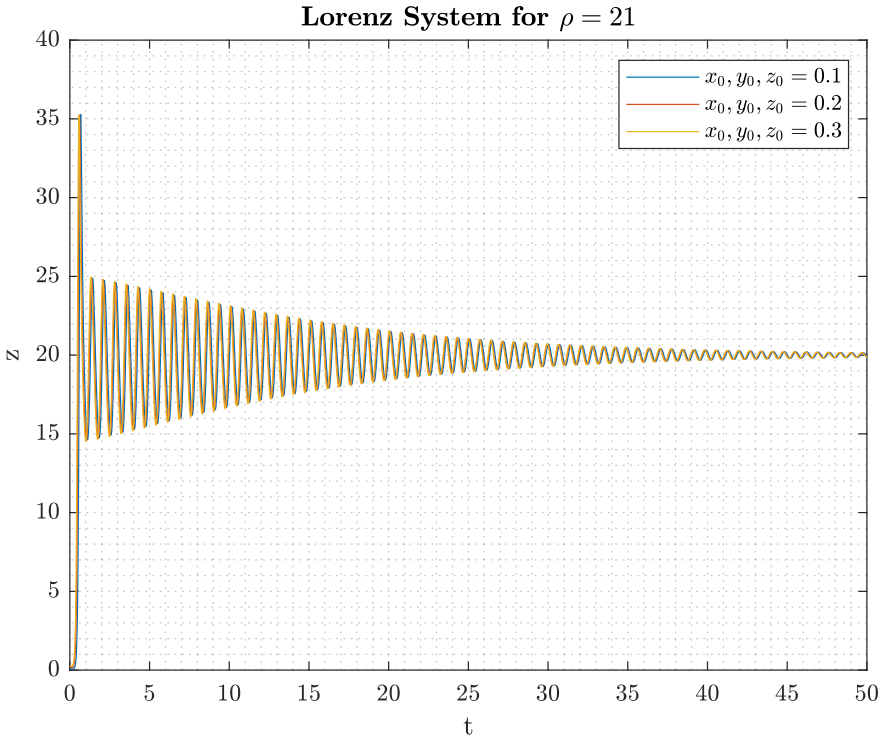


Figure 48 Lorenz System for $\rho = 21$. Source Own.

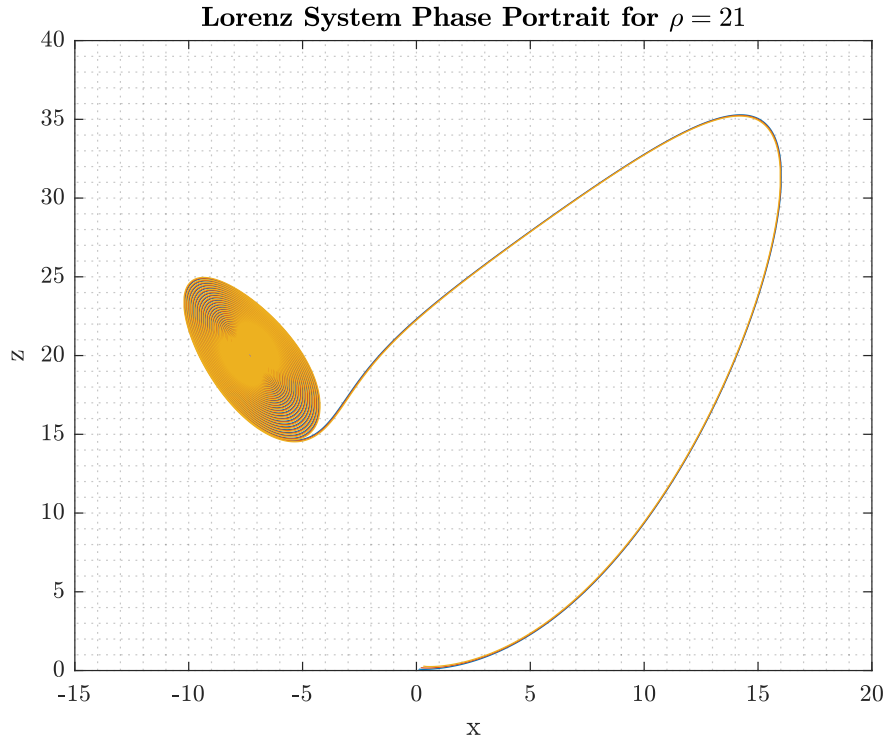


Figure 49 Lorenz System for $\rho = 21$. Source Own.

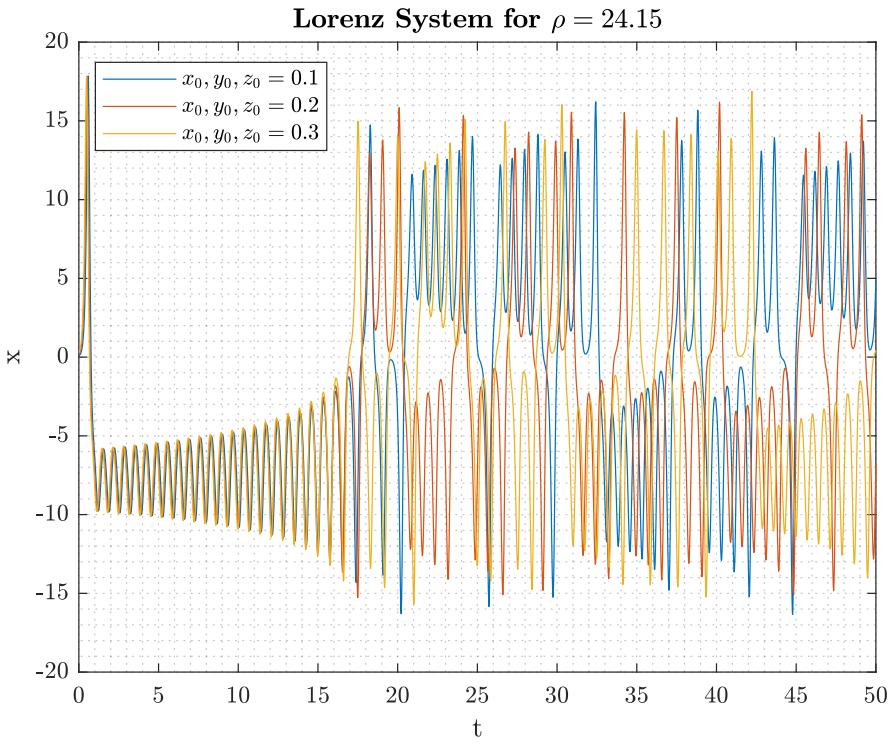


Figure 50 Lorenz System for $\rho = 24.15$. Source Own.

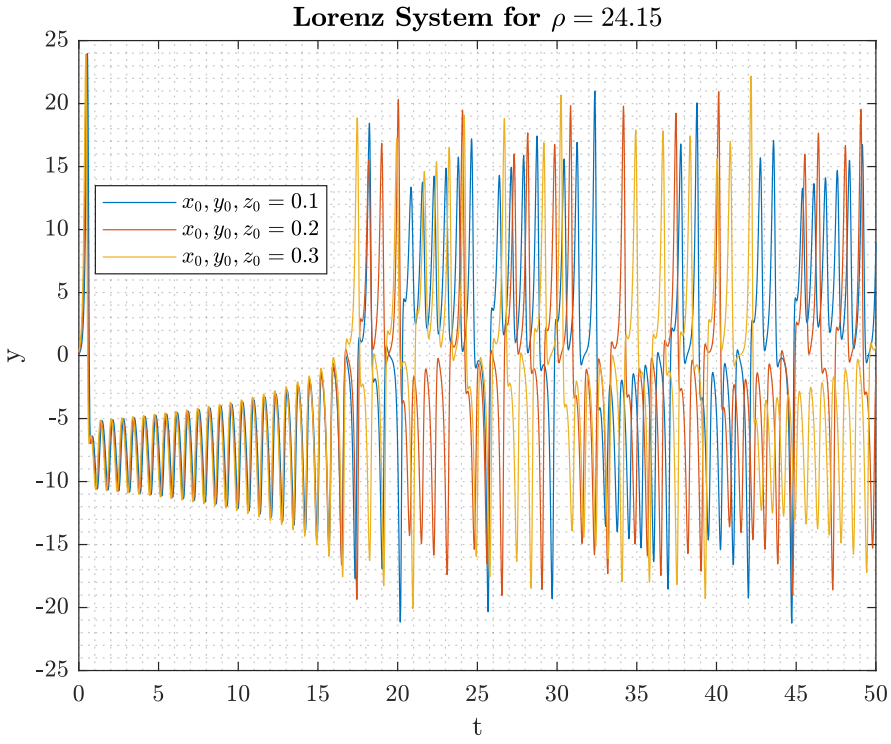


Figure 51 Lorenz System for $\rho = 24.15$. Source Own.

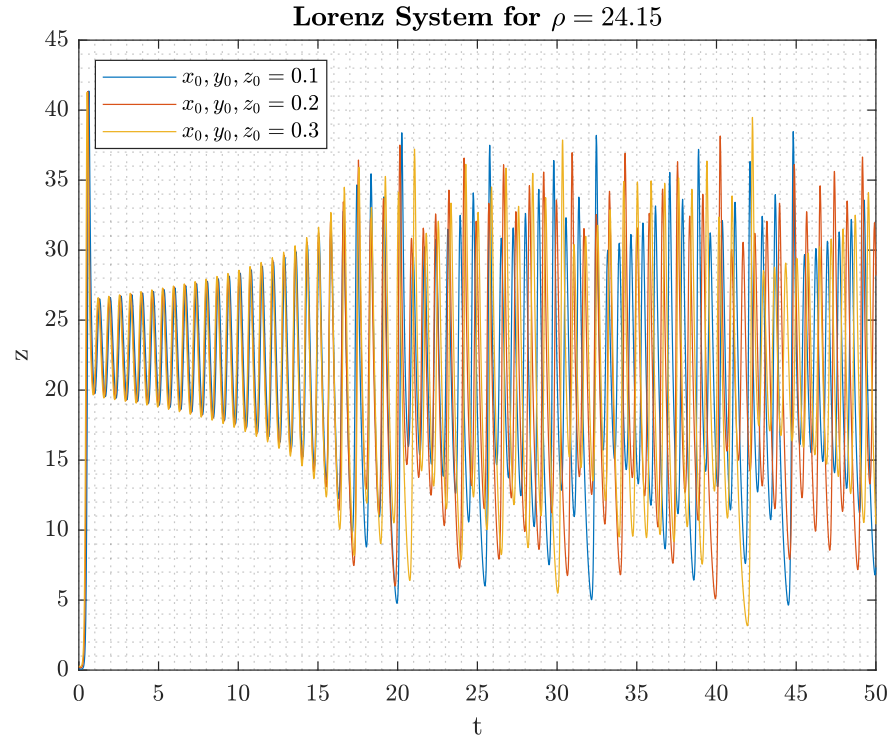


Figure 52 Lorenz System for $\rho = 24.15$. Source Own.

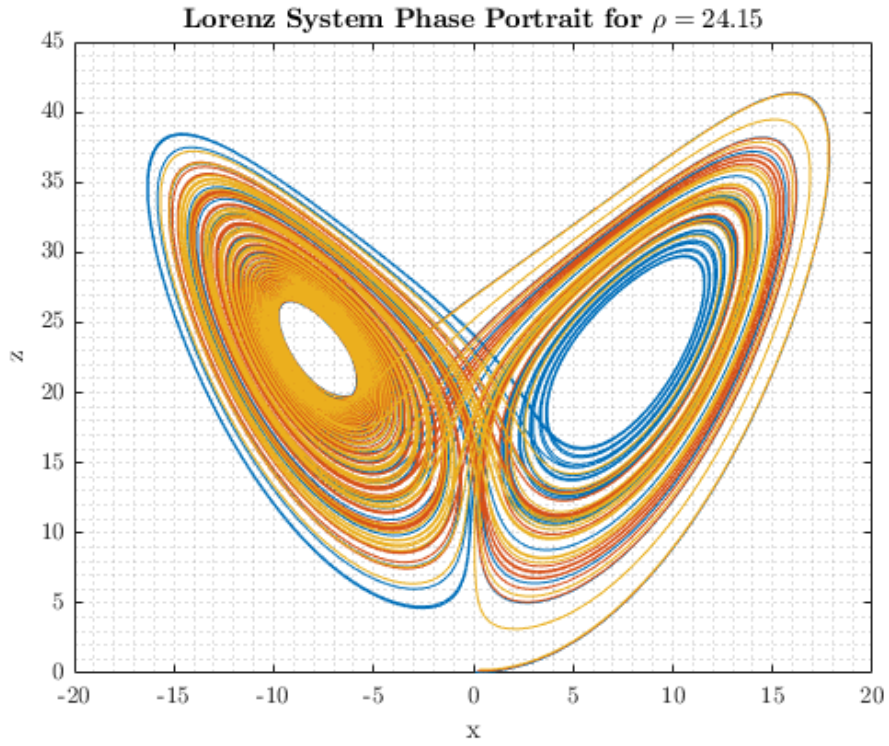


Figure 53 Lorenz System for $\rho = 24.15$. Source Own.

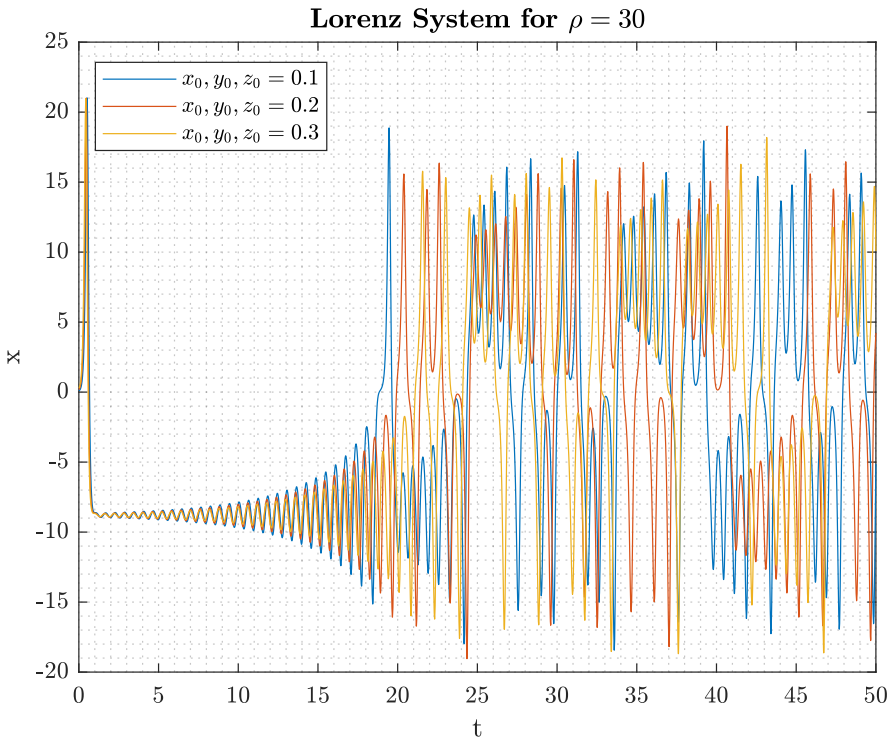


Figure 54 Lorenz System for $\rho = 30$. Source Own.

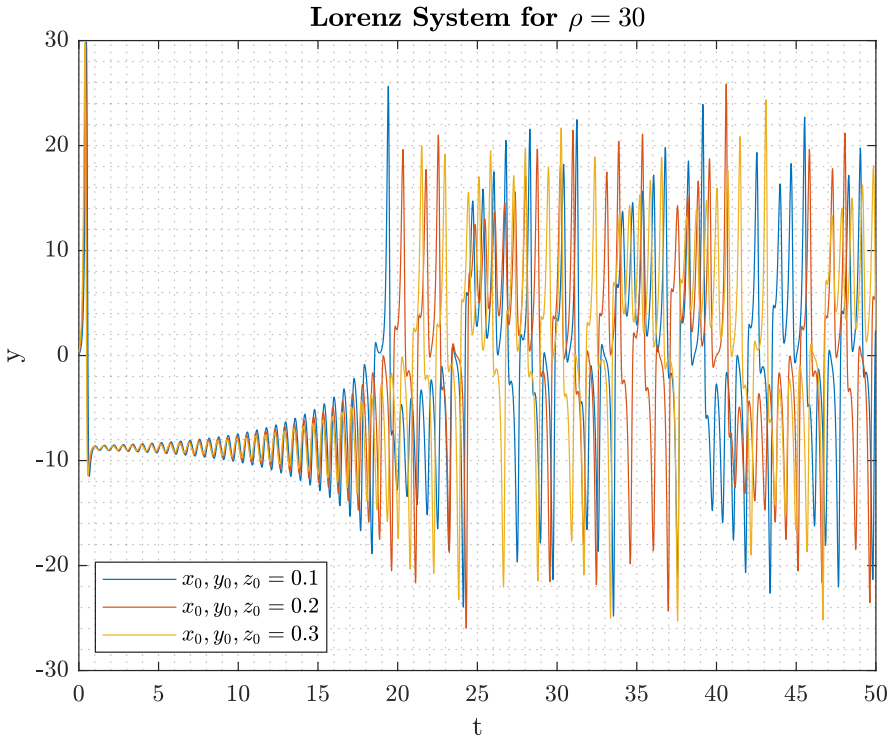


Figure 55 Lorenz System for $\rho = 30$. Source Own.

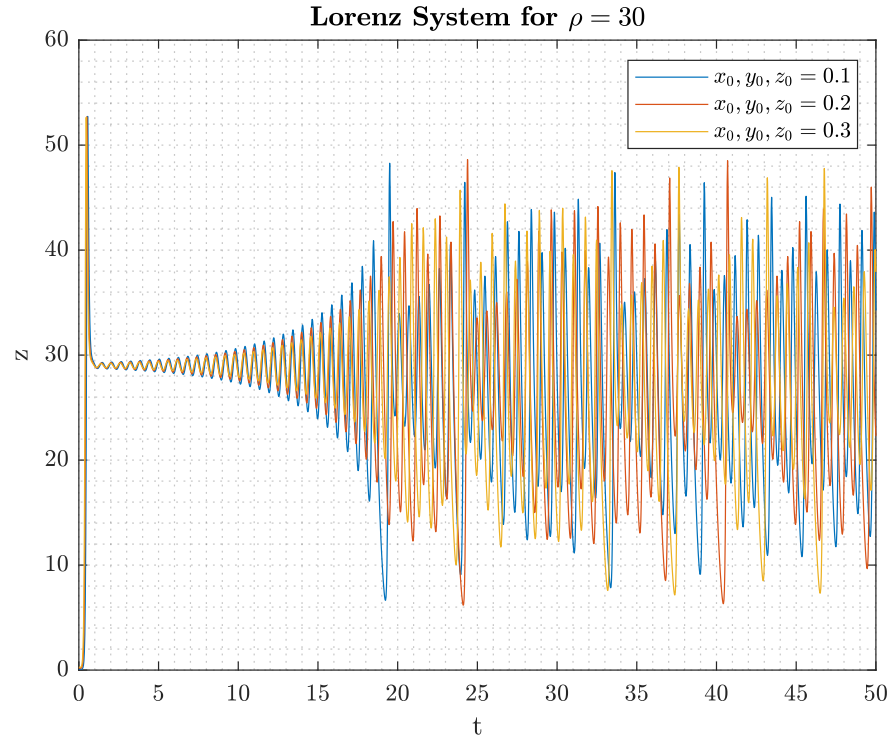


Figure 56 Lorenz System for $\rho = 30$. Source Own.

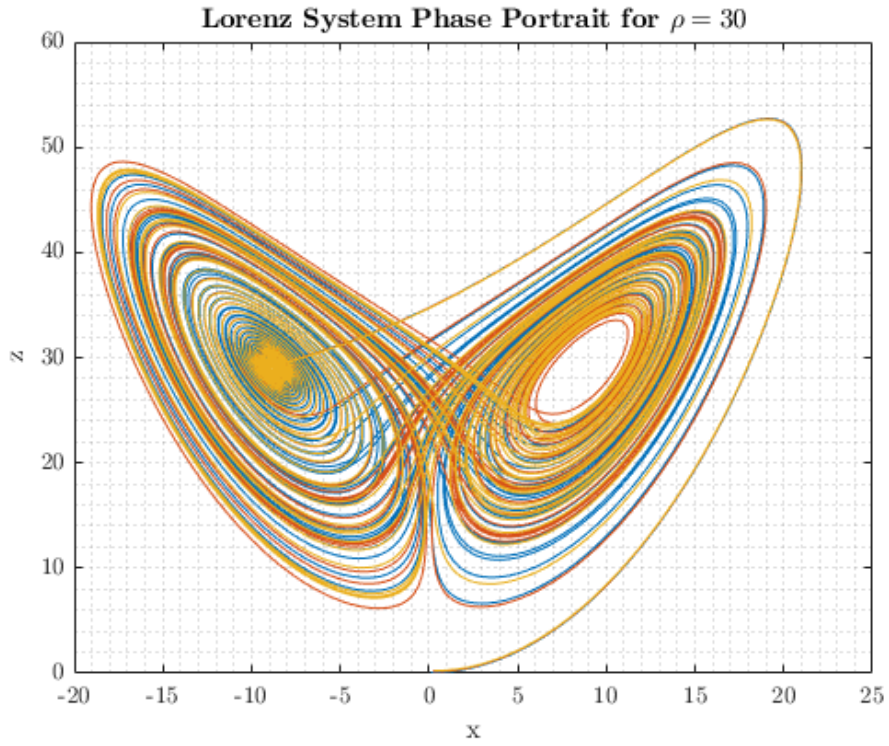


Figure 57 Lorenz System for $\rho = 30$. Source Own.

References

[1] Pons, Antonio J. “CHAOS”. In: 1st ed. UPC, 2021, pp. 1–37.

[2] Pons, Antonio J. “LINEAR SYSTEMS”. In: 1st ed. UPC, 2021, pp. 1–40.

[3] Pons, Antonio J. “NONLINEAR SYSTEMS”. In: 1st ed. UPC, 2021, pp. 1–37.

[4] Steven H Strogatz. Nonlinear dynamics and chaos with student solutions manual: With applications to ph
CRC press, 2018.