UNIVERSITETET I STAVANGER

Institutt for data- og elektroteknologi

Chaotic systems - analysis, simulation, and experiments on the effect of adding integral control

Participant

Name	Student number
Ullestad, Adrian Langvik	243237

Abstract

Model chaotic systems into Matlab/Simulink, or other programs that allow simulations. The systems should then be simulated for different disturbances to observer both the step response and how the amplitude, average value of amplitude and frequency, for one or more output variables, changes for different disturbance values. The results can be presented both as a time response or in bifurcation diagrams. Then the systems should be expanded to allow one or more I-controller and perform the same analysis again, to see the effect the I-controlled has on the system. Then if given the time create one of the systems as an electrical circuit.

The models are created in Matlab using the *ode*45 solver on a Matlab script, made from the differential equations, for each of the chaotic models used. To create the Simulink model we simply started implementing the differential equations into Simulink. The next part is the expanded Simulink model with an I-controller, which was made possible by adding the integral response into the differential equation for x. Then adding an closed-feedback loop from x to calculate the error. To perform the simulation of the expanded chaotic systems we needed to find the integral gain, K_i, which was done by finding the transfer function from u, integral response, to x, the output. After finding the transfer function we made some assumptions, and then by comparing the closed-loop function, $M(s)$, with a desired closed-loop function, $M_d(s)$, we can calculate the integral gain, K_i.

From the simulations done on simple chaotic systems we can say that there is not a big difference to the results we get from a simulation of the Matlab script and on a model made in Simulink. Taking a look at the results from the calculations for K_i and the simulations using this value gave us some satisfying results for most of the chaotic systems.

Contents

1. Introduction

In this there will be given an introduction to chaos, the characteristics of chaos and behaviour that occur in chaotic systems, what happens when an integrator controller is placed in a chaotic system, and an outline for the project.

1.1 Chaos

According to the book "Nonlinear dynamics and chaos" by Steven Strogatz [5] there is not an universally accepted definition of chaos. However people in the field mainly agree that a chaotic system contains at least these three:

- Aperiodic long-term behavior
- Deterministic
- Sensitive dependence on initial conditions

Where an aperiodic long-term behavior mean that some of the system's trajectories do not settle down toward any fixed points, periodic- or quasiperiodic orbits. Whereas deterministic means that there are no noisy inputs to the system and that all irregular behavior stems from the systems nonlinearity, instead of other forces. And lastly sensitive dependence on initial conditions means that the system has a positive Liapunov exponent, or in other words the systems dynamic changes with just a tiny change in the initial condition.

So what problems occur in chaotic systems, one of them is that because of the aperiodic long-term behavior it is not possible to predict the the systems response. This is also caused by the sensitive dependence on initial conditions, which makes a system diverge with small changes in the starting point, figure 1.1 shows the effect of a small change in the initial condition of a chaotic system. By observing the results in the figure, we can see that

after a time the solution trajectories diverges. The time before the signals diverges depends on the difference of the change in the initial conditions.

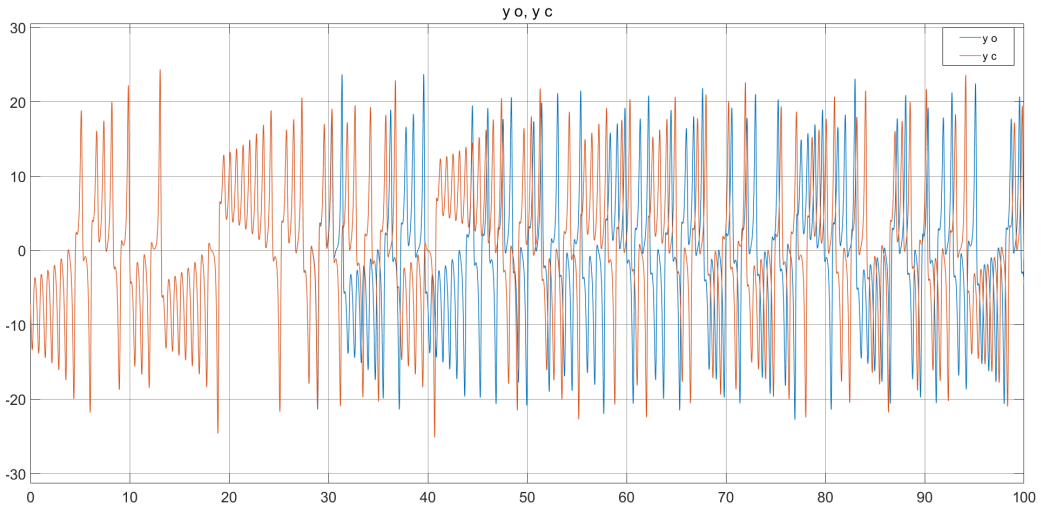


Figure 1.1: A chaotic system with a small change in the initial condition, change by 0.00000000001. The blue (y o) line shows the original signal, whereas the red (y c) is the same system with a change in the initial conditions.

Other problems is that because of the sensitivity of the chaotic system there is a possibility that when simulating, such a system can give different results by changing the tolerance of the solver. This is caused by the fact that with each new calculation, the previous calculated result is perceived as an initial condition.

1.2 I-control of a nonlinear system

So what happens when an integrator is placed as an controller for the nonlinear system, one of the goals of this thesis is to look for an answer to this question. However, we will start by explaining what an integral controller is; an integral controller is used to mainly remove steady-state error, and when the error is zero the controller will give a fixed value which is the same as the value when the error became zero. The error is given by the reference minus the current value of the system.
To find an answer to the effect we get from an I-controller we will in this work look at different chaotic models, which are implemented and simulated in Matlab/Simulink. The control parameters are also calculated using the

differential equations. So by using the controller and the Simulink models we want to use multiple steps in the reference at different times. We will assume that the step response will be oscillating. Then by taking the average of the response ones it reaches steady state, we will see that the response is equal to the each of the respective steps.

1.3 Project overview

Chapter 2: Chaotic systems and implementation

This chapter, explains how the first part of the thesis is performed and how the second part is done. The first part explain the implementation of the chaotic systems into Matlab and Simulink for each of the chaotic systems used. The second part explain how we found the transfer function for each of the chaotic systems and the calculations to find the parameters for the I-controller.

Chapter 3: Experiments and Results

This chapter, shows us the experiments done on part one and two. The first part gives us proof on whether the coefficients we use do make the systems chaotic. The second part uses the coefficients from part one to see the effect of an I-controller added to the system.

Chapter 4: Discussion

This chapter gives an conclusion to the various experiments done, improvements and further work.

2. Chaotic systems and implementation

In this chapter we will take a closer look at the chaotic systems that are used in this study, the calculations used to solve the two main objectives as described in chapter 1. We will therefore split this chapter into two parts. The first part takes a closer look at implementing a chaotic system in Matlab and in Simulink, and the second will take a closer look at the creation of the controller.

2.1 Lorenz

Starting of, we have probably one of the most well known chaotic systems the Lorenz system. The Lorenz system was discovered by Edward Lorenz during the 1960's when he was trying to model and simulate weather patterns [7].

The Lorenz system is given by the differential equations. Where x, y and z are the states of the system and σ, ρ and β are parameters.

$$\begin{aligned} \dot{x} =& \sigma(y - x) \\ \dot{y} =& \rho x - y - xz \\ \dot{z} =& xy - \beta z \end{aligned} \tag{2.1}$$

To make it simpler to follow when looking through the Matlab code the parameters have here been renamed, $\sigma = s$, $\rho = r$ and $\beta = b$, which results in the following equations

$$\begin{aligned} \dot{x} =& s(y - x) \\ \dot{y} =& rx - y - xz \\ \dot{z} =& xy - bz \end{aligned} \tag{2.2}$$

When analysing a system it is interesting to observe where the states are equal to zero, to do this we simply solve equation 2.3 for each state.

$$\begin{aligned} 0 =& s(y-x) \\ 0 =& rx - y - xz \\ 0 =& xy - bz \end{aligned} \tag{2.3}$$

After solving the equation we get the following fixed points: $(0,0,0)$, and $(\pm\sqrt{b(r-1)}, \pm\sqrt{b(r-1)}, r-1)$.
The parameters in the Lorenz system are also known with their own names: s is the Prandtl number, r is the Reayleigh number, and b has no name. Where r is

$$1 < r < r_H = \frac{s(s+b+3)}{s-b-1} \tag{2.4}$$

and $s - b - 1 > 0$, the system looses stability at $r = r_H$. So choosing an r larger then r_H will result in chaotic behavior.

One of the signatures of a Lorenz system is the Lorenz attractor, shown in figure 2.1. Obtaining the Lorenz attractor in both the Matlab and Simulink models is the objective for this part of the thesis.

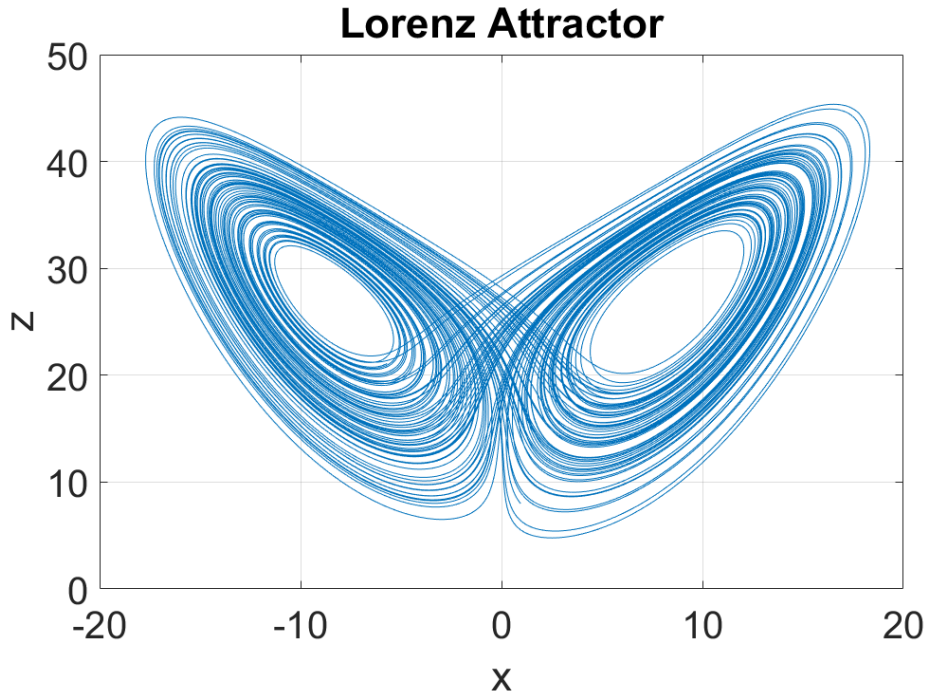


Figure 2.1: Lorenz attractor formed by the trajectories of the states. Here in the xz-plane. Using $s = 10$, $b = \frac{8}{3}$ and $r = 28$, starting at $(-5.7, -7.1, 21.1)$

2.1.1 Matlab

To simulate the Lorenz system in Matlab we use the *ode45* solver and a *self-made* function which contain the equations. The Matlab script that contain the equation was made by Kristian Thorsen in 2015, shown in listing 2.1.

Listing 2.1: LorenzSystem.m

```
function [dx] = LorenzSystem(t,x,s,r,b)
% LORENZSYSTEM ...
% Differential equations for the Lorenz System (Equations) based on
% Edqard N. Lorenz - Deterministic Nonperiodic Flow [1963]
%
% 3 dimensional system with equations (s = sigma, in origial paper)
% sigma = Prandtl number
% r = Reayleigh number
% b = noname
%
% dx(1) = s*(x(2) - x(1))                 % Linear
% dx(2) = r*x(1) - x(2) - x(1)*x(3)       % Nonlinear crossterm
% dx(3) = x(1)*x(2) - b*x(3)              % Nonlinear crossterm
%
% Where s, r, b > 0
% The system becomes chaotic as r increases, the Hopf bifurcation values rH is:
% rH = s(s+b+3)/(s-b-1)
%
% Lorenz Parameters
% s=10, b=8/3, r=28
%%

% AUTHOR     : Kristian Thorsen
% DATE       : 29-Nov-2015 00:22:05
% Revision   : 1.00 (29-Nov-2015 00:22:05)
% DEVELOPED  : 8.1.0.604 (R2013a)
% FILENAME   : LorenzSystem.m

%% Changelog
% v 1.00 (29-Nov-2015 00:22:05)
%       -Initial version
%%
dx = [s*(x(2)-x(1)); r*x(1) - x(2) - x(1)*x(3); x(1)*x(2) - b*x(3)];
end
```

This function takes in the time t, and the current value of x, y and z as the vector. As well as the parameters s, r and b, and returns the $\dot{x}$, $\dot{y}$ and $\dot{z}$.

Simulation of the Lorens system is then done by calling the ode45 solver, the ode45 solver needs the simulation start and stop time, along with the function that is going to be used, in this instant is *LorenzSystem.m*. As mentioned earlier the tolerance of the simulation can affect the results, so in this thesis an tolerance of 10^{-9} is used.

2.1.2 Simulink

The Simulink model developed for the Lorenz system is shown in figure 2.2, and is made based of the differential equation given in equation 2.2.

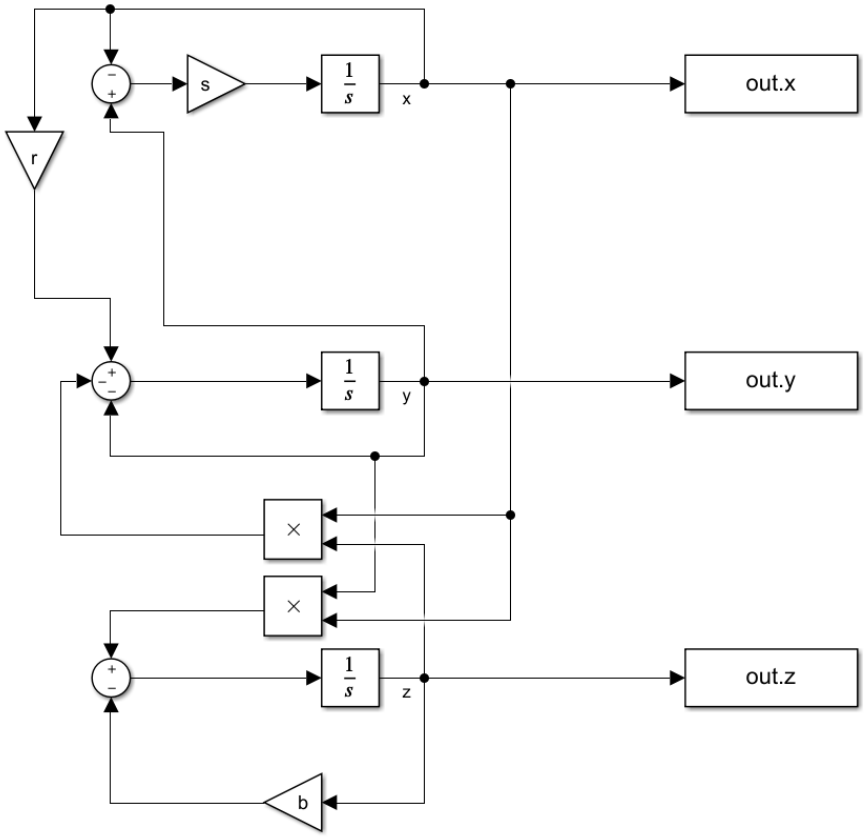


Figure 2.2: Simulink model of the Lorenz system

The Simulink model uses the solver ode45 and a tolerance of 10^{-9}, which is the same as for the Matlab functions.

2.2 Lorenz system as an electrical circuit

In this part of the thesis we implement the Lorenz system as an electrical circuit. We will base our implementation on the two research articles: "Circuit Implementation of Synchronization with Applications to Communication" [2], and "Synchronization of Lorenz-Based Chaotic Circuit with Applications to Communication" [3] both by Kevin Cuomo et al.. The electrical circuit, shown in figure 2.3, contains inverse buffers, inverse summation, multipliers. An inverse buffer has the same effect as a non-inverting buffer, but the output will be the opposite. So if the input is positive, the output will be negative, and opposite. The inverse sum-block will sum all the inputs of the operational amplifier and give an output value that is opposite of the input. The multipliers are used to obtain the nonlinear parts of the chaotic system, by multiplying the values together.

Since the electrical model is made based of real life components, there is a need to scale the variables to match a realistic range for power supplies. The easiest way to do this is to perform a simple transformation of variables. The transformation of variables is:

$$\begin{aligned} u &= \frac{x}{10} \\ v &= \frac{y}{10} \\ w &= \frac{z}{20} \end{aligned} \tag{2.5}$$

Which gives us a new set of differential equations which is now given by:

$$\begin{aligned} \dot{u} &= s(v-u) \\ \dot{v} &= ru - v - 20uw \\ \dot{w} &= 5uv - bw \end{aligned} \tag{2.6}$$

This equation can be used to make a new Simulink model which will be a scaled version compared to the model in figure 2.2. However this is not how this Simulink model will be made, we will use a Simulink library called Simscape, where we will be using models of electrical components. As mentioned this model is made based of the article made by Cuomo [2], the Lorenz-based circuit that Cuomo found is shown in figure 2.3.

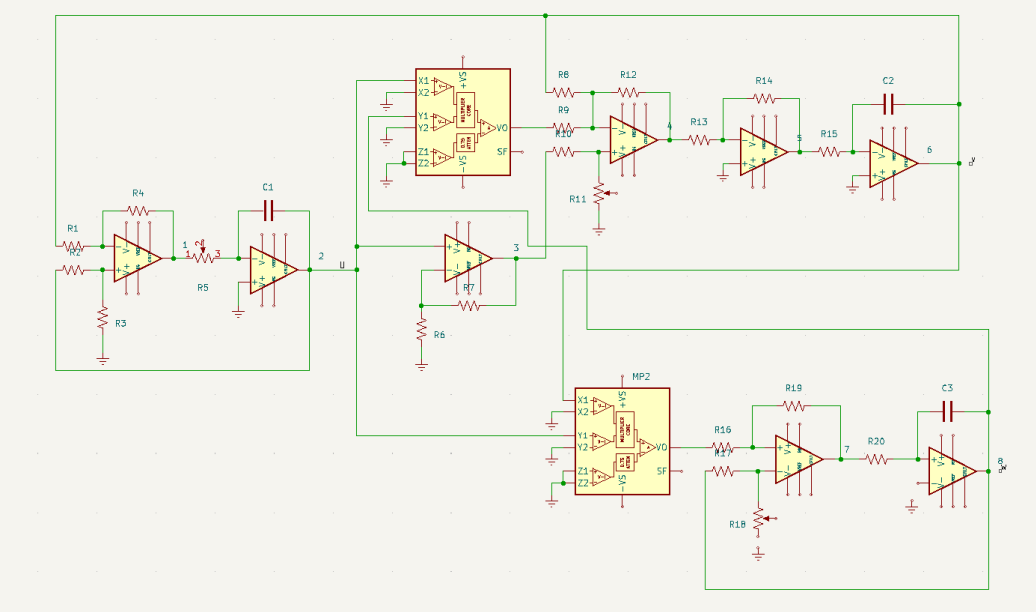


Figure 2.3: Lorenz-based circuit

By analysing the Lorenz-based circuit we will by observing the nodes in figure 2.3, and applying standard node analysis obtain the following equations:

$$\begin{aligned}
\dot{u} &= \frac{1}{R_5 C_1}\left[\frac{R_4}{R_1}v - \frac{R_3}{R_2+R_3}\left(1+\frac{R_4}{R_1}\right)\right] \\
\dot{v} &= \frac{1}{R_{15}C_2}\left[\frac{R_{11}}{R_{10}+R_{11}}\left(1+\frac{R_{12}}{R_8}+\frac{R_{12}}{R_9}\right)\left(1+\frac{R_7}{R_6}\right)u - \frac{R_{12}}{R_8}v - \frac{R_{12}}{R_9}uw\right] \\
\dot{w} &= \frac{1}{R_{20}C_3}\left[\frac{R_{19}}{R_{16}}uv - \frac{R_{18}}{R_{17}+R_{18}}\left(1+\frac{R_{19}}{R_{16}}\right)w\right]
\end{aligned} \tag{2.7}$$

The capacitors C_1, C_2 and C_3 determine the time scale for the circuit, which mean that we can easily change the time scale, as long as all three capacitors are changed by the same factor. So by comparing equation 2.6 and equation 2.7 we can obtain equations for the s, r and b value, giving the following equations:

$$
\begin{aligned}
s =& \frac{1}{R_5 C_1} \\
r =& \frac{1}{R_{15} C_2} \cdot \frac{R_{11}}{R_{10} + R_{11}} \left(1 + \frac{R_{12}}{R_8} + \frac{R_{12}}{R_9}\right) \left(1 + \frac{R_7}{R_6}\right) \\
b =& \frac{1}{R_{20} C_3} \cdot \frac{R_{18}}{R_{17} + R_{18}} \left(1 + \frac{R_{19}}{R_{16}}\right)
\end{aligned} \tag{2.8}
$$

So from the equation we can see that we can adjust the values of s, r and b by changing the values of R_5, R_{11} and R_{18}. To obtain the value of the coefficients that corresponds to equation 2.7 we have to rescale by a time factor. The rescaling factor can be obtained by looking at:

$$
20uw = \frac{1}{R_{15} C_2} \cdot \frac{R_{12}}{R_9} uw
$$

To solve this we need to divide the right hand side with the rescale factor, which results in:

$$
\begin{aligned}
20 &= \frac{\frac{1}{R_{15} C_2} \cdot \frac{R_{12}}{R_9}}{sf} \\
&\Downarrow \\
sf &= \frac{\frac{1}{R_{15} C_2} \cdot \frac{R_{12}}{R_9}}{20}
\end{aligned} \tag{2.9}
$$

Resulting in the following electrical simulink model:

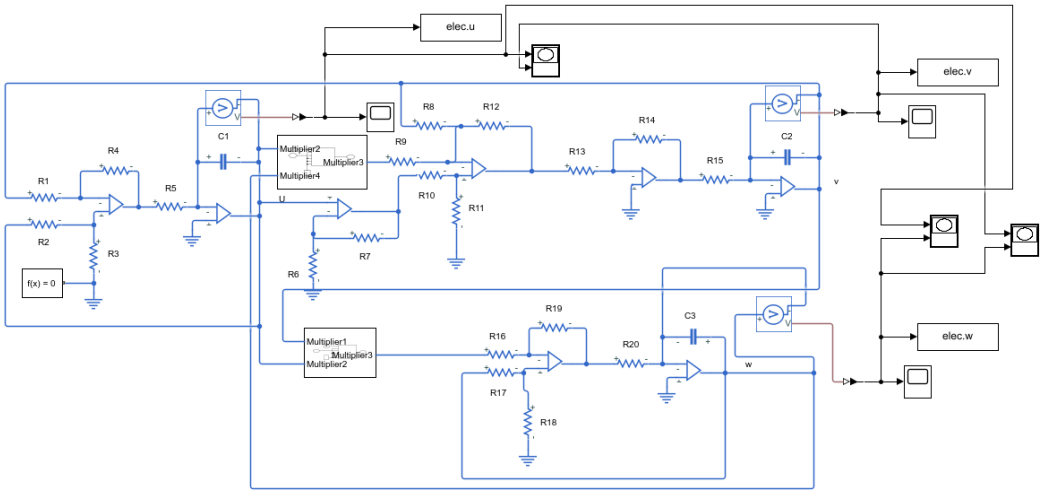


Figure 2.4: Simulink model of the Lorenz based circuit

2.3 Chua

Together with the Lorenz system the Chua system is looked at as one of simplest chaotic systems, and as a classic example of chaos. The Chua circuit was invented by Leon Chua in 1983 [6]. The Chua circuit is given by the differential equations:

$$\begin{aligned} \dot{x} =& \alpha(y - x - h(x)) \\ \dot{y} =& x - y + z \\ \dot{z} =& -\beta y \end{aligned} \tag{2.10}$$

where $h(x)$ is given by:

$$h(x) = m_1 \cdot x + \frac{1}{2}(m_0 - m_1)(|x + 1| - |x - 1|) \tag{2.11}$$

The function $h(x)$ describes the voltage-current characteristic of a nonlinear resistor. And α, β, m_1 and m_0 are coefficients, and we will rename $\alpha = a$ and $\beta = b$ from here on, giving us the following.

$$\begin{aligned} \dot{x} =& a(y - x - h(x)) \\ \dot{y} =& x - y + z \\ \dot{z} =& -by \end{aligned} \tag{2.12}$$

By observing the equations 2.10 and 2.12 we can see that the $\dot{x}$ part is nonlinear and that $\dot{y}$ and $\dot{z}$ are linear. The Chua cicuit has fixed point where $\dot{x}$ is equal to zero and we solve that equation:

$$a(y - x - h(x)) = 0$$

Solving that equation for x gives us the fixed point in x, calling this point $\bar{x}$, giving us the following set of points $(\bar{x},\ 0,\ -\bar{x})$.
The Chua circuit like the Lorenz system has a chaotic attractor, shown in figure 2.5, that we wish to recreate in both Matlab and Simulink.

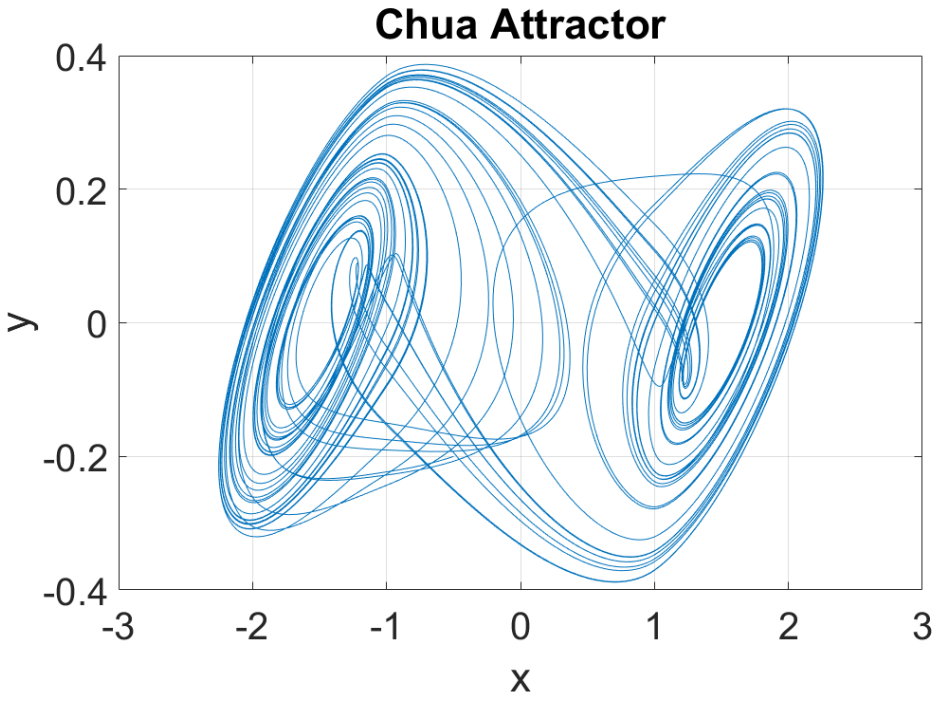


Figure 2.5: The Chua attractor formed by the trajectories of the states. Here in the xy-plane. Using $m_0 = -\frac{8}{7}$, $m_1 = -\frac{5}{7}$, $a = 15.6$ and $b = 28$, starting at $(-0.5, -0.2, 0)$

2.3.1 Matlab

Just as it was for the Lorenz system we are using the same solver and tolerance when using Matlab. The implementation of the differential equation into a function was based of a function found online at [4]. The resulting function can be seen in listing 2.2.

Listing 2.2: MathChua.m

```
function dx=MathChua(t,x, m0,m1,a,b)
%Chuasystem
% Differential equations based of Chua circuit
%
%
% dx(1)=a*(x(2)-x(1)-h(x(1)); % Nonlinear
% dx(2)=x(1)-x(2)+x(3); % Linear
% dx(3)=-b*x(2); % Linear
%
% where h is
% m1*x(1)+1/2*(m0-m1)*(abs(x(1)+1)-abs(x(1)-1))
%
% a, b > 0
%
%AUTHOR : Adrian Ullestad
%DATE : 28-Mai-2022
%DEVELOPED : 2020b
%FILENAME : MathChua.m
%

h=m1*x(1)+1/2*(m0-m1)*(abs(x(1)+1)-abs(x(1)-1));

dx=[a*(x(2)-x(1)-h); x(1)-x(2)+x(3);-b*x(2)];

end
```

The function works in the same way as the function for the Lorenz system.

2.3.2 Simulink

When it comes to the Simulink implementation there was only need for one model, since we did not make an electrical model. This means that the model was made based on the differential equations given in equation 2.12. Shown in figure 2.6.

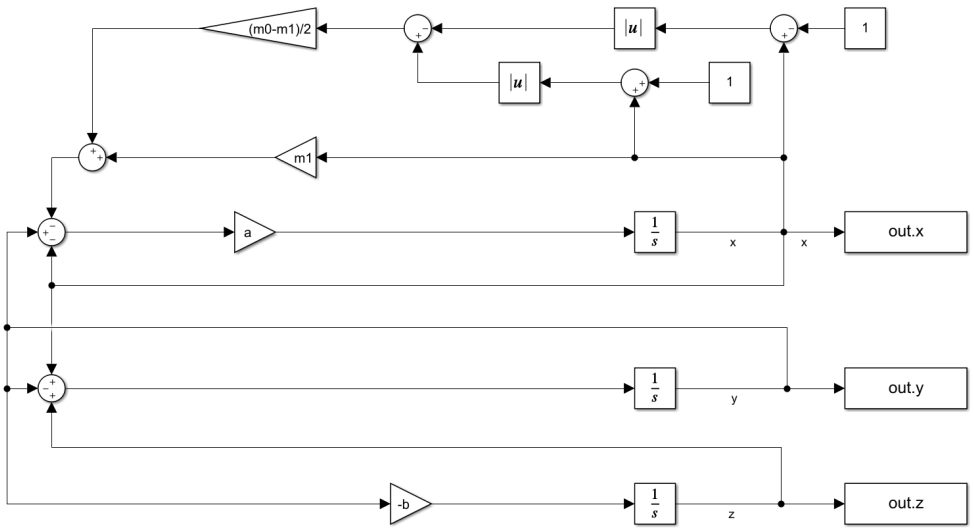


Figure 2.6: Simulink model of the Chua system

2.4 Rössler

The Rössler system is the last of the chaotic systems that we will take a look at in this thesis. The Rössler system was invented by Otto Rössler in the 1970s [8]. The differential equations for the Rössler system are.

$$\begin{aligned} \dot{x} &= -y - z \\ \dot{y} &= x + ay \\ \dot{z} &= b + xz - cz \end{aligned} \tag{2.13}$$

Where a, b and c are coefficients. The Rössler system has fixed points in

$$\begin{aligned} \bar{x} &= \frac{c \pm \sqrt{c^2 - 4ab}}{2} \\ \bar{y} &= \frac{-c \pm \sqrt{c^2 - 4ab}}{2a} \\ \bar{z} &= \frac{c \pm \sqrt{c^2 - 4ab}}{2a} \end{aligned} \tag{2.14}$$

The Rössler system has as the other two chaotic systems an chaotic attractor, shown in figure 2.7, which we wish to obtain.

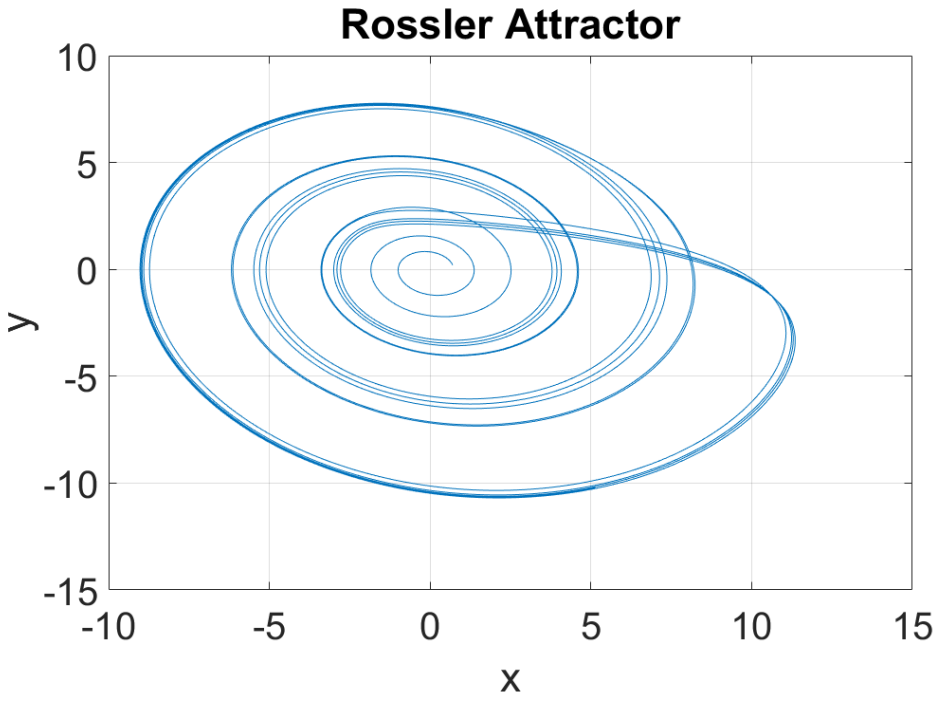


Figure 2.7: The Rössler attractor formed by the trajectories of the states. Here in the xy-plane. Using $a = 0.2$, $b = 0.2$ and $c = 5.7$, starting at $(0.7, 0.2, 0)$

2.4.1 Matlab

The Matlab function for the Rössler system is similar to that of the other Matlab functions. This means that other than the differential equation in the function the simulation uses the same tolerance and simulation time, as well as the ode45. The function is shown in listing 2.3.

Listing 2.3: RosslerMatlab.m

```
function dx=RosslerMatlab(t,x,a,b,c)
%Rossler system
% Differential equations based of the Rossler system
%
% dx(1)=-x(2)-x(3);           % Linear
% dx(2)=x(1)+a*x(2);          % Linear
% dx(3)=b+x(3)*x(1)-x(3)*c;   % Nonlinear
%
% a, b, c > 0
% Warying c will change the attractor.
%
%AUTHOR      : Adrian Ullestad
%DATE        : 28-Mai-2022
%DEVELOPED   : 2020b
%FILENAME    : RosslerMatlab.m
%


dx=[-x(2)-x(3);x(1)+a*x(2);b+x(3)*x(1)-x(3)*c];

end
```

2.4.2 Simulink

Just as it was for the Chua system there is only one model, which is the mathematical model. So the model is based on the equation given in equation 2.13, and is shown in figure 2.8.

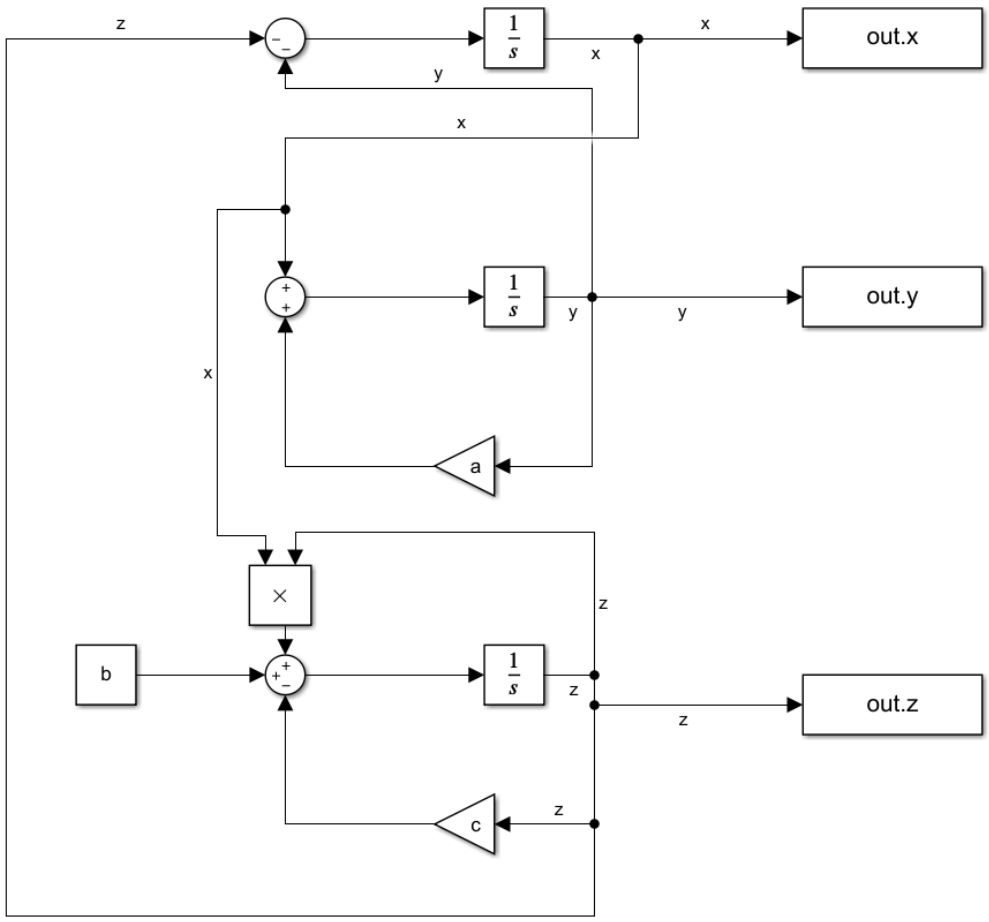


Figure 2.8: Simulink model of the Rössler system

2.5 The integrator-controller

The second part of the thesis is to see what effect an I-controller has on the chaotic system. To start of with the chaotic systems used in this part are the same as the ones used in the previous task. The controller is a typical PID-controller, that is used when we want to control a system to a desired value. A PID-controller is given by the following equation.

$$u(t) = K_p \cdot e(t) + \frac{K_p}{T_i} \int_0^t e(\tau)d\tau + K_p \cdot T_d \cdot \frac{de(t)}{dt} \tag{2.15}$$

Where $e(t)$ is the error given by the difference in the current value and the desired value, and is shown equation 2.16 where $x_{sp}(t)$ is the desired value and $x(t)$ is the current value.

$$e(t) = x_{sp}(t) - x(t) \tag{2.16}$$

However, since we are only going to use an integral controller we will use equation 2.17 to calculate u. The I part of the controller has the trait of continuously summing the error (e(t)), which will eventually result in no offset from the desired valued.

$$u(t) = \frac{K_p}{T_i} \int_0^t e(\tau) d\tau \tag{2.17}$$

So in the next sections we will see the calculations of the I-controllers coefficients based on the differential equations for the chaotic systems. And to calculate this coefficients we have used the IMC method [1]. Where we will be using

$$\tau_c = \frac{\tau}{3} \tag{2.18}$$

as the universal τ_c value for all the systems. Where τ_c is the desired response time for the closed loop response function M(s). So to start the process to acquire the coefficients we want, is to start with linearization of the differential equations around a fixed point, then perform Laplace transformation of the linearized equations, and then solve for the desired output.

2.5.1 Lorenz, linearization and transfer function

To start of with we choose the fixed points, and we choose:

$$\begin{aligned} \bar{x} &= \sqrt{b(r-1)} \\ \bar{y} &= \sqrt{b(r-1)} \\ \bar{z} &= r - 1 \end{aligned} \tag{2.19}$$

The reason why we use the fixed point shown in equation 2.19 is because we already know based of equation 2.3 already is zero and based on that we can simply set $\bar{u} = 0$. Additionally, we choose the fixed point of the three mentioned earlier is that we want to avoid getting imaginary values for our calculations. Where these values are denoted in Matlab as $\bar{x} = xA$, $\bar{y} = yA$ and $\bar{z} = zA$ as well as $\bar{u} = uA$. Changing the value of $\bar{u}$ will also change the fixed points of $\bar{x}$, $\bar{y}$ and $\bar{z}$. Additionally the addition of the controller will change the differential equations for the Lorenz system:

$$\begin{aligned} \dot{x} &= s(y - x) + u \\ \dot{y} &= rx - y - xz \\ \dot{z} &= xy - bz \end{aligned} \tag{2.20}$$

Where the control action u acts on the differential equation for x.

Deciding to use x as the output we want to control, and u is the input. The I-controller input will be the difference between a desired setpoint x_{sp} and x, $e(t) = x_{sp}(t) - x(t)$. We want to find a transfer function from u to x, meaning we want to solve for $\frac{x}{u}$, where u is given in equation 2.17.
We will start by linearizing equation 2.20, and then Laplace transform that equation before solving for the desired output. We introduce the delta variables $\Delta x = (x - \bar{x})$, $\Delta y = (y - \bar{y})$ and $\Delta z = (z - \bar{z})$. The linearizied differential equations are given in equation 2.21.

$$\begin{aligned} \Delta\dot{x} =& s\Delta y - s\Delta x + \Delta u \\ \Delta\dot{y} =& (r - \bar{z})\Delta x - \Delta y - \bar{x}\Delta z \\ \Delta\dot{z} =& \bar{y}\Delta x + \bar{x}\Delta y - b\Delta z \end{aligned} \tag{2.21}$$

Since we want to solve equation 2.21 with respect to $\frac{\Delta x}{\Delta u}$, because of the notation used when performing Laplace transformation we will from here reintroduce $s = \sigma$ for the parameter, and reserve s for use as the Laplace variable. The Laplace transformation is shown in equation 2.22.

$$\begin{gathered} \Delta\dot{x} = \sigma\Delta y - \sigma\Delta x + \Delta u \\ \Delta\dot{y} = (r - \bar{z})\Delta x - \Delta y - \bar{x}\Delta z \\ \Delta\dot{z} = \bar{y}\Delta x + \bar{x}\Delta y - b\Delta z \\ \Downarrow \mathscr{L} \\ s\Delta x = \sigma\Delta y - \sigma\Delta x + \Delta u \\ s\Delta y = (r - \bar{z})\Delta x - \Delta y - \bar{x}\Delta z \\ s\Delta z = \bar{y}\Delta x + \bar{x}\Delta y - b\Delta z \end{gathered} \tag{2.22}$$

Then taking a closer look at the first part of the new equation, and solving for Δx we get:

$$\Delta x = \frac{\sigma}{s + \sigma}\Delta y + \frac{1}{s + \sigma}\Delta u \tag{2.23}$$

and then a look at the Δy part we obtain

$$\Delta y = \frac{(r - \bar{z})}{s + 1}\Delta x - \frac{\bar{x}}{s + 1}\Delta z \tag{2.24}$$

and finally we look at Δz

$$\Delta z = \frac{\bar{y}}{s + b}\Delta x + \frac{\bar{x}}{s + b}\Delta y \tag{2.25}$$

Then substituting equation 2.25 into equation 2.24

$$\Delta y = \frac{r-\bar{z}}{s+1}\Delta x - \frac{\bar{x}}{s+1}\left(\frac{\bar{y}}{s+b}\Delta x + \frac{\bar{x}}{s+b}\Delta y\right)$$
$$\Downarrow$$
$$\Delta y = \frac{r-\bar{z}}{s+1}\Delta x - \frac{\bar{x}\bar{y}}{(s+1)(s+b)}\Delta x - \frac{\bar{x}^2}{(s+1)(s+b)}\Delta y$$
$$\Downarrow \tag{2.26}$$
$$\left(1 + \frac{\bar{x}^2}{(s+1)(s+b)}\right)\Delta y = \frac{(r-\bar{z})(s+b) - \bar{x}\bar{y}}{(s+1)(s+b)}\Delta x$$
$$\Downarrow$$
$$\Delta y = \frac{(r-\bar{z})(s+b) - \bar{x}\bar{y}}{(s+1)(s+b) + \bar{x}^2}$$

and then substituting equation 2.26 into equation 2.23.

$$\Delta x = \frac{\sigma}{s+\sigma}\left(\frac{(r-\bar{z})(s+b) - \bar{x}\bar{y}}{(s+1)(s+b) + \bar{x}^2}\Delta x\right) + \frac{1}{s+\sigma}\Delta u$$
$$\Downarrow$$
$$\Delta x = \frac{\sigma(r-\bar{z})(s+b) - \sigma\bar{x}\bar{y}}{(s+\sigma)(s+1)(s+b) + (s+\sigma)\bar{x}^2}\Delta x + \frac{1}{s+\sigma}\Delta u$$
$$\Downarrow \tag{2.27}$$
$$\left(1 - \frac{\sigma(r-\bar{z})(s+b) - \sigma\bar{x}\bar{y}}{(s+\sigma)(s+1)(s+b) + (s+\sigma)\bar{x}^2}\right)\Delta x = \frac{1}{s+\sigma}\Delta u$$
$$\Downarrow$$
$$\frac{\Delta x}{\Delta u} = \frac{(s+1)(s+b) + \bar{x}^2}{(s+\sigma)(s+1)(s+b) - \bar{x}(s+\sigma) - \sigma(r-\bar{z})(s+b) + \sigma\bar{x}\bar{y}}$$

And now we will take a look at the denominator to simpler get the coefficients for controller.

$$(s+\sigma)(s+1)(s+b) - \bar{x}(s+\sigma) - \sigma(r-\bar{z})(s+b) + \sigma\bar{x}\bar{y} =$$
$$s^3 + (b+1+\sigma)s^2 + (1+\sigma(b+1) - \sigma(r-\bar{z})b)s + \sigma(b - (r-\bar{z})b + \bar{x}\bar{y} \tag{2.28}$$

Given the result from equation 2.28 we will make some assumptions and some new variables. Let us start with the assumptions, which is that we will simplify the integrator gain as:

$$K_i = \frac{K_p}{T_i} \tag{2.29}$$

The next variables we will make are simply there to make writing and calculating the integral controller gain easier, these are:

$$\begin{aligned} D =& \sigma + b + 1 \\ E =& 1 + \sigma b + \sigma - \sigma r b + \sigma \bar{z} b \\ F =& \sigma(b - (r - \bar{z})b + \bar{x}\bar{y}) \end{aligned} \tag{2.30}$$

This gives us the following transfer function:

$$\frac{\Delta x}{\Delta u} = \frac{(s+1)(s+b) + \bar{x}^2}{s^3 + Ds^2 + Es + F} \tag{2.31}$$

The last assumption we will make is to use an I-controller that will cancel the numerator of equation 2.31. As mentioned earlier we wish to use the IMC-based method, unfortunately the transfer function in equation 2.31 is a third order function, so we can not use a table to look up the formula for K_i. To find the integral gain we will have to use the transfer function of the I-controller $\frac{K_i}{s}$ and calculate the closed loop function $M(s)$, which is given by.

$$\begin{aligned} M(s) &= \frac{\frac{K_i}{s((s+1)(s+b)+\bar{x}^2)} \cdot \frac{(s+1)(s+b)+\bar{x}^2}{s^3+Ds^2+Es+F}}{1 + \frac{K_i}{s((s+1)(s+b)+\bar{x}^2)} \cdot \frac{(s+1)(s+b)+\bar{x}^2}{s^3+Ds^2+Es+F}} \\ &\Downarrow \\ M(s) &= \frac{K_i}{s^4 + Ds^3 + Es^2 + Fs + K_i} \\ &\Downarrow \\ M(s) &= \frac{1}{\frac{1}{K_i}s^4 + \frac{D}{K_i}s^3 + \frac{E}{K_i}s^2 + \frac{F}{K_i}s + 1} \end{aligned} \tag{2.32}$$

And then we have the model from the IMC method, which is the model that is the closest to our transfer function:

$$\frac{K}{\tau^2 s^2 + 2\zeta\tau s + 1} \tag{2.33}$$

however, as we can see from equation 2.33 the model is only a second order model, and since $M(s)$, from equation 2.32, is a fourth order model. We have to multiply the model from equation 2.33 with another second order model. We choose to compare $M(s)$ with two systems in cascade to find the value for K_i, which gives us:

$$\frac{K}{\tau_1^2 s^2 + 2\zeta\tau_1 s + 1}\frac{1}{\tau_2^2 s^2 + 2\zeta\tau_2 s + 1}$$
$$\Downarrow$$
$$\frac{K}{\tau_1^2\tau_2^2 s^4 + (2\zeta\tau_1\tau_2^2 + 2\zeta\tau_2\tau_1^2)s^3 + (\tau_1^2 + 4\zeta^2\tau_1\tau_2 + \tau_2^2)s^2 + (2\zeta\tau_1 + 2\zeta\tau_2)s + 1} \tag{2.34}$$

Based of equation 2.34's denominator we can now solve for the unknowns K_i, τ_1 and τ_2, where ζ is a free variable that we can choose, which gives us:

$$\frac{1}{K_i} = \tau_1^2\tau_2^2 \tag{2.35}$$

$$\frac{D}{K_i} = 2\zeta\tau_1\tau_2^2 + 2\zeta\tau_2\tau_1^2 \tag{2.36}$$

$$\frac{E}{K_i} = \tau_1^2 + 4\zeta^2\tau_1\tau_2 + \tau_2^2 \tag{2.37}$$

$$\frac{F}{K_i} = 2\zeta\tau_1 + 2\zeta\tau_2 \tag{2.38}$$

To start we use equation 2.35 and solve for K_i, we then substitute that equation into equation 2.38 to find τ_1, and finally we will use equation 2.36 and substitutes the equations for τ_1 and K_i into it to find τ_2. To solve the equations we use the Matlab function *solve*, this gives us the following equations:

$$\begin{aligned} \tau_2 &= solve(\frac{D}{K_i} = 2\zeta\tau_1\tau_2^2 + 2\zeta\tau_2\tau_1^2, \tau_2) \\ \tau_1 &= \frac{\zeta \pm \sqrt{\zeta(2F\tau_2^3 + \zeta)}}{F\tau_2^2} \\ K_i &= \frac{F^2\tau_2^2}{(\zeta \pm \sqrt{\zeta(2F\tau_2^3 + \zeta)})^2} \end{aligned} \tag{2.39}$$

2.5.2 Chua, linearization and transfer function

To start we identify the fixed points for the Chua circuit, this is because we want to use them later after linearization. The Chua cicuit has fixed points

given by:

$$\begin{aligned} \bar{x} : a(-x - (m_1 x + \frac{1}{2}(m_0 - m_1)(abs(x+1) - abs(x-1)) = 0 \\ \bar{y} = 0 \\ \bar{z} = -\bar{x} \end{aligned} \tag{2.40}$$

Where $\bar{x}$ is found solving the given equation and m_0 and m_1 are arbitrarily chosen coefficients. The notation used in Matlab is the same here as for the Lorenz system, $\bar{x} = xA$, $\bar{y} = yA$ and $\bar{z} = zA$, and the differential equations is now:

$$\begin{aligned} \dot{x} =& a(y - x - h(x)) + u \\ \dot{y} =& x - y + z \\ \dot{z} =& -by \end{aligned} \tag{2.41}$$

where $h(x)$ is still given by equation 2.11, and then we can move over to the linearization part. Here we have the input in x to make future calculations easier and the output is x. Hence, we will use the same methods to solve the equations as we did for the Lorenz system.

$$\begin{aligned} \Delta\dot{x} =& a\Delta y - (a + ag(x))\Delta x + \Delta u \\ \Delta\dot{y} =& \Delta x - \Delta y + \Delta z \\ \Delta\dot{z} =& -b\Delta y \end{aligned} \tag{2.42}$$

Where $g(x)$ is the linearization of $h(x)$, and is given by:

$$g(x) = m_1 + \frac{1}{2}(m_0 - m_1)\left(\frac{\bar{x}+1}{\sqrt{(\bar{x}+1)^2}} - \frac{\bar{x}-1}{\sqrt{(\bar{x}-1)^2}}\right) \tag{2.43}$$

After performing linearization we can now move over to Laplace transform equation 2.42, and then find the transferfunction.

$$\begin{aligned} \Delta\dot{x} &= a\Delta y - (a + ag(x))\Delta x + \Delta u \\ \Delta\dot{y} &= \Delta x - \Delta y + \Delta z \\ \Delta\dot{z} &= -b\Delta y \\ &\Downarrow \mathscr{L} \\ s\Delta x &= a\Delta y - (a + ag(x))\Delta x + \Delta u \\ s\Delta y &= \Delta x - \Delta y + \Delta z \\ s\Delta z &= -b\Delta y \end{aligned} \tag{2.44}$$

Looking at the x equation we solve for Δx, and thus we obtain:

$$\Delta x = \frac{a}{s + (a + ag)(x)}\Delta y + \frac{1}{s + (a + ag(x))}\Delta u \tag{2.45}$$

And similarly for Δy.

$$\Delta y = \frac{1}{s+1}\Delta x + \frac{1}{s+1}\Delta z \tag{2.46}$$

Finally we look at Δz.

$$\Delta z = -\frac{b}{s}\Delta y \tag{2.47}$$

Then we will substitute equation 2.47 into equation 2.46,

$$\begin{gathered}
\Delta y = \frac{1}{s+1}\Delta x + \frac{1}{s+1}\left(-\frac{b}{s}\Delta y\right) \\
\Downarrow \\
\left(1 + \frac{b}{(s+1)s}\right)\Delta y = \frac{1}{s+1}\Delta x \\
\Downarrow \\
\Delta y = \frac{s}{s(s+1)+b}\Delta x
\end{gathered} \tag{2.48}$$

and then equation 2.48 into equation 2.45 which gives us our transfer function:

$$\begin{gathered}
\Delta x = \frac{a}{s+a+ag(x)}\left(\frac{s}{s(s+1)+b}\Delta x\right) + \frac{1}{s+a+ag(x)}\Delta u \\
\Downarrow \\
\left(1 - \frac{as}{(s+a+ag(x))(s+1)s + b(s+a+ag(x))}\right)\Delta x = \frac{1}{s+a+ag(x)}\Delta u \\
\Downarrow \\
\frac{\Delta x}{\Delta u} = \frac{s(s+1)+b}{(s+a+ag(x))(s+1)s + b(s+a+ag(x)) - as}
\end{gathered} \tag{2.49}$$

After finding the transfer function $\frac{\Delta x}{\Delta u}$, we will look at the denominator and calculate it. The calculation gives us a denominator that looks like this:

$$\begin{gathered}
(s+a+ag(x))(s+1)s + b(s+a+ag(x)) - as = \\
s^3 + (1+a+ag(x))s^2 + (a+ag(x)+b-a)s + b(a+ag(x))
\end{gathered} \tag{2.50}$$

Which after writing the transfer function on standard form we obtain the following function which will allow us to calculate the parameters we need for the controller.

$$\frac{\Delta x}{\Delta u} = \frac{s(s+1)+b}{s^3 + (1+a+ag(x))s^2 + (ag(x)+b)s + b(a+ag(x))} \tag{2.51}$$

We will, like we did for the Lorenz system make, simplify equation 2.51 by renaming some of the variables with new variable names.

$$\begin{aligned} D =& a + ag(x) + a \\ E =& ag(x) + b \\ F =& b(a + ag(x)) \end{aligned} \tag{2.52}$$

We will also make the same assumption for K_i as we did for the Lorenz system, then using the new variables, given in equation 2.52, to calculate the closed loop transfer function with the integral controller included, $M(s)$. Then set $M(s)$ on standard form, to compare with the desired $M_d(s)$.

$$\begin{aligned} M(s) &= \frac{K_i}{s^4 + Ds^3 + Es^2 + Fs + K_i} \\ &\Downarrow \\ M(s) &= \frac{1}{\frac{1}{K_i}s^4 + \frac{D}{K_i}s^3 + \frac{E}{K_i}s^2 + \frac{F}{K_i} + 1} \end{aligned} \tag{2.53}$$

$M_d(s)$ is two second order system multiplied together, which is:

$$M_d(s) = \frac{K}{(\tau_1^2 s^2 + 2\zeta\tau_1 s + 1)(\tau_2^2 s^2 + 2\zeta\tau_2 s + 1)}$$

We will now compare the denominators and solve them for K_i, τ_1 and τ_2 respectively, ζ will still be a free variable, and then we can obtain our integral-controller. To calculate the equations we use Matlab's solve function, and we will thus have the following equations:

$$\frac{1}{K_i} = \tau_1^2\tau_2^2 \tag{2.54}$$

$$\frac{D}{K_i} = 2\zeta\tau_1\tau_2^2 + 2\zeta\tau_2\tau_1^2 \tag{2.55}$$

$$\frac{E}{K_i} = 2\tau_1^2 + 4\zeta^2\tau_1\tau_2 + \tau_2^2 \tag{2.56}$$

$$\frac{F}{K_i} = 2\zeta\tau_1 + 2\zeta\tau_2 \tag{2.57}$$

Then solving equation 2.54 for K_i, equation 2.55 for τ_2 and equation 2.57

for τ_1, thus we use the *solve* function and obtain:

$$\begin{aligned}
\tau_2 =& solve\left(\frac{D}{\frac{F^2\tau_2^2}{\left(\zeta-\sqrt{\zeta(2F\tau_2^3+\zeta)}\right)^2}} == 2\zeta\frac{\zeta-\sqrt{\zeta(2F\tau_2^3+\zeta)}}{F\tau_2^2}\tau_2^2 + 2\zeta\tau_2\frac{\zeta-\sqrt{\zeta(2F\tau_2^3+\zeta)}}{F\tau_2^2}^2\right)\\
\tau_1 =& \frac{\zeta-\sqrt{\zeta(2F\tau_2^3+\zeta)}}{F\tau_2^2}\\
K_i =& \frac{1}{\tau_1^2\tau_2^2}
\end{aligned} \tag{2.58}$$

2.5.3 Rössler, linearization and transfer function

To start of we identify the fixed points we are going to use when calculating the controller parameters, still using $\bar{u} = 0$, which gives us the following fixed points:

$$\begin{aligned}
\bar{x} =& \frac{c \pm \sqrt{c^2-4ab}}{2}\\
\bar{y} =& \frac{-c \pm \sqrt{c^2-4ab}}{2a}\\
\bar{z} =& \frac{c \pm \sqrt{c^2-4ab}}{2a}
\end{aligned} \tag{2.59}$$

Additionally, because of the addition of the controller the new differential equations are given by:

$$\begin{aligned}
\dot{x} =& -y-z+u\\
\dot{y} =& x+ay\\
\dot{z} =& b+zx-zc
\end{aligned} \tag{2.60}$$

For the Rössler system we have decided to use x as input and y as output, this means we want to acquire the transfer function $\frac{\Delta y}{\Delta u}$. To reach this point we will go through the same steps as for the Lorenz system and the Chua circuit.

$$\begin{aligned}
\Delta\dot{x} =& -\Delta y-\Delta z+\Delta u\\
\Delta\dot{y} =& \Delta x+a\Delta y\\
\Delta\dot{z} =& (\bar{x}-c)\Delta z+\bar{z}\Delta x
\end{aligned} \tag{2.61}$$

To start of we will perform Laplace transformation of equation 2.61, and find

the transfer function $\frac{\Delta y}{\Delta u}$.

$$\begin{aligned}
\Delta\dot{x} &= -\Delta y - \Delta z + \Delta u \\
\Delta\dot{y} &= \Delta x + a\Delta y \\
\Delta\dot{z} &= (\bar{x} - c)\Delta z + \bar{z}\Delta x \\
&\Downarrow \mathscr{L} \\
s\Delta x &= -\Delta y - \Delta z + \Delta u \\
s\Delta y &= \Delta x + a\Delta y \\
s\Delta z &= (\bar{x} - c)\Delta z + \bar{z}\Delta x
\end{aligned} \tag{2.62}$$

Taking a look at the x equation we obtain:

$$\Delta x = -\frac{1}{s}\Delta y - \frac{1}{s}\Delta z + \frac{1}{s}\Delta u \tag{2.63}$$

Then for y:

$$\Delta y = \frac{1}{s - a}\Delta x \tag{2.64}$$

and similarly for z we obtain:

$$\Delta z = \frac{\bar{z}}{s - (\bar{x} - c)}\Delta x \tag{2.65}$$

So by substituting equation 2.65 into equation 2.63, and solving for Δx. Then substituting equation 2.66 into equation 2.64 and then solving for $\frac{\Delta y}{\Delta u}$.

$$\begin{aligned}
\Delta x =& -\frac{1}{s}\Delta y - \frac{1}{s}\left(\frac{\bar{z}}{s - (\bar{x} - c)}\Delta x\right) + \frac{1}{s}\Delta u \\
\left(1 + \frac{\bar{z}}{s(s - (\bar{x} - c))}\right)\Delta x =& -\frac{1}{s}\Delta y + \frac{1}{s}\Delta u \\
\Delta x =& -\frac{s - (\bar{x} - c)}{s(s - (\bar{x} - c)) + \bar{z}}\Delta y + \frac{s - (\bar{x} - c)}{s(s - (\bar{x} - c)) + \bar{z}}\Delta u
\end{aligned} \tag{2.66}$$

Now that we have found the equation for Δx, we will as mentioned substitute the equation for Δx into the equation for Δy, this will result in our transfer

function.

$$\Delta y = \frac{1}{s-a}\left(-\frac{s-(\bar{x}-c)}{s(s-(\bar{x}-c)+\bar{z}}\Delta y + \frac{s-(\bar{x}-c)}{s(s-(\bar{x}-c)+\bar{z}}\Delta u\right)$$
$$\Downarrow$$
$$\left(1+\frac{s-(\bar{x}-c)}{(s-a)(s-(\bar{x}-c))s+\bar{z}(s-a)}\right)\Delta y = \frac{s-(\bar{x}-c)}{(s-a)(s-(\bar{x}-c))s+\bar{z}(s-a)}\Delta u$$
$$\Downarrow$$
$$\frac{\Delta y}{\Delta u} = \frac{s-(\bar{x}-c)}{(s-a)(s-(\bar{x}-c))s+\bar{z}(s-a)+s-(\bar{x}-c)} \tag{2.67}$$

Then calculating the denominator and then find $M(s)$ compare it with the desired transfer function. The denominator is:

$$s^3+(-a-(\bar{x}-c))s^2+(a(\bar{x}-c)+\bar{z}-(\bar{x}-c))s+(-\bar{z}a-(\bar{x}-c) \tag{2.68}$$

Additionally, we will with the calculation of the denominator find $M(s)$, but first we will introduce some new variables.

$$\begin{aligned} D =&(-a-(\bar{x}-c) \\ E =&a\bar{x}-ac+\bar{z}-\bar{x}+c \\ F =&-\bar{z}a-(\bar{x}-c) \end{aligned} \tag{2.69}$$

and we get:

$$M(s) = \frac{-\frac{1}{\bar{x}-c}s+1}{\frac{1}{-K_i(\bar{x}-c)}s^4+\frac{E}{-K_i(\bar{x}-c)}s^3+\frac{D}{-K_i(\bar{x}-c)}s^2+\frac{F+K_i}{-K_i(\bar{x}-c)}s+1} \tag{2.70}$$

We will use the same $M_d(s)$ as we use for the Lorenz System which is:

$$M_d(s) = \frac{K(-\beta s+1)}{(\tau_1^2 s^2+2\zeta\tau_2 s+1)(\tau_2^2 s^2+2\zeta\tau_2 s+1)} \tag{2.71}$$

Then we set the two denominators equal to each other and find four equation with three unknowns and we get the following equations:

$$-\frac{1}{K_i(\bar{x}-c)} = \tau_1^2\tau_2^2 \tag{2.72}$$

$$-\frac{D}{K_i(\bar{x}-c)} = 2\zeta\tau_1\tau_2^2+2\zeta\tau_2\tau_1^2 \tag{2.73}$$

$$-\frac{E}{K_i(\bar{x}-c)} = \tau_1^2+(2\zeta)^2\tau_1\tau_2+\tau_2^2 \tag{2.74}$$

$$-\frac{F+K_i}{K_i(\bar{x}-c)} = 2\zeta\tau_1 + 2\zeta\tau_2 \tag{2.75}$$

We decide to use the following three equations, equation 2.72 that we will solve for K_i, equation 2.74 that we solve for τ_1, and equation 2.75 which we solve for τ_2 with the help of Matlab's solve function, shown in equation 2.76. This gives us the final integral-controller parameters.

$$\frac{2\zeta\tau_2}{\tau_2 D - 2\zeta}^2 - \frac{1}{\bar{x}-c} = \frac{4\zeta^2\tau_2}{\tau_2 D - 2\zeta} + 2\zeta\tau_2 \tag{2.76}$$

Then the final results are.

$$\begin{aligned}
\tau_2 =& solve(\frac{2\zeta\tau_2}{\tau_2 D - 2\zeta}^2 - \frac{1}{\bar{x}-c} == \frac{4\zeta^2\tau_2}{\tau_2 D - 2\zeta} + 2\zeta\tau_2, \tau_2) \\
\tau_1 =& \frac{2\zeta\tau_2}{\tau_2 D - 2\zeta} \\
K_i =& -\frac{1}{\tau_1^2\tau_2^2(\bar{x}-c)}
\end{aligned} \tag{2.77}$$

3. Experiments and Results

In this chapter we will take a look at how implementing the data from the previous chapter went, and which results we got. The chapter consists of three section each section contains both the Matlab and Simulink constructed models and the Simulink model that contains the controller.

3.1 Lorenz

For the Lorenz system we have as mentioned one Matlab program and two Simulink models. To validate the different implementations and to compare their outputs, we have decided to use three different coefficients to see if all implementations produce a chaotic system. These coefficients are shown in table 3.1.

Coefficients	**Experiments**		
	1	2	3
s	10	16	22
b	$\frac{8}{3}$	4	6
r	28	45.6	56
r_H	24.74	33.46	45.47

Table 3.1: Table for coefficients used during experiments Lorenz

As mentioned in the chaotic systems and implementation chapter under subsection Lorenz 2.1 the system becomes chaotic when $r > r_H$. To prove this we have made a bifurcation diagram, figure 3.1, using the first coefficients given in table 3.1.

Just like in the previous section we will split the chapter into two parts, one which is using Matlab and Simulink to perform simple simulations, and the other part contains the controller part. Each of these part contains three experiments.

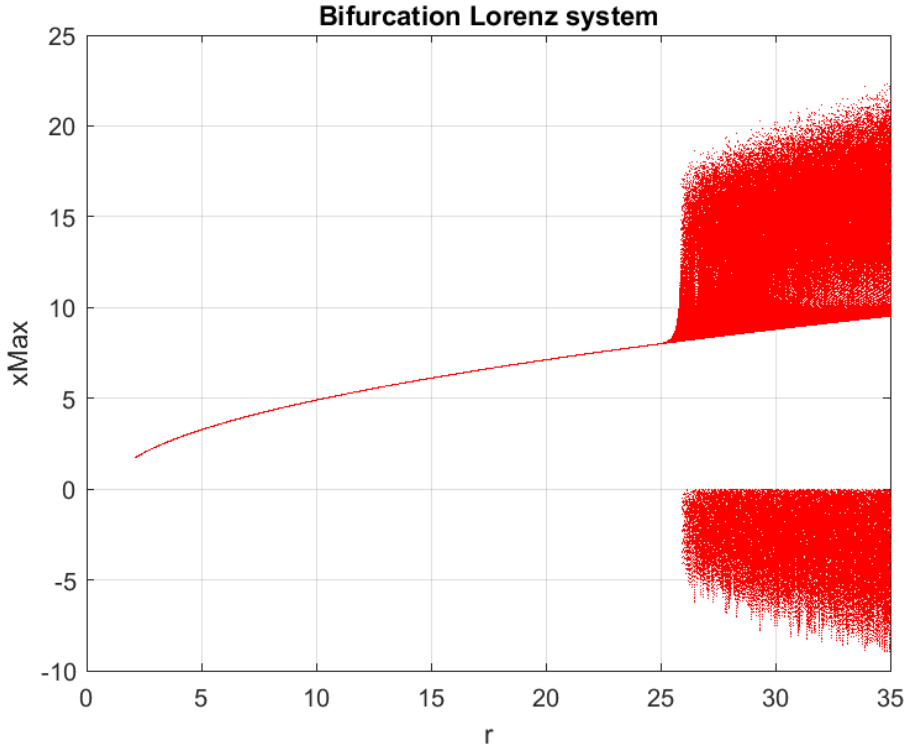
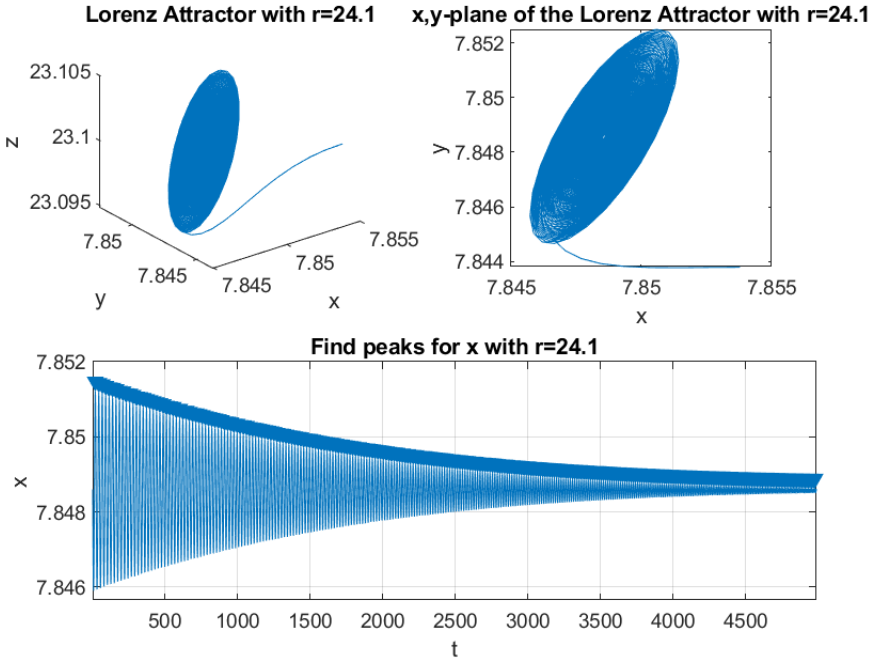


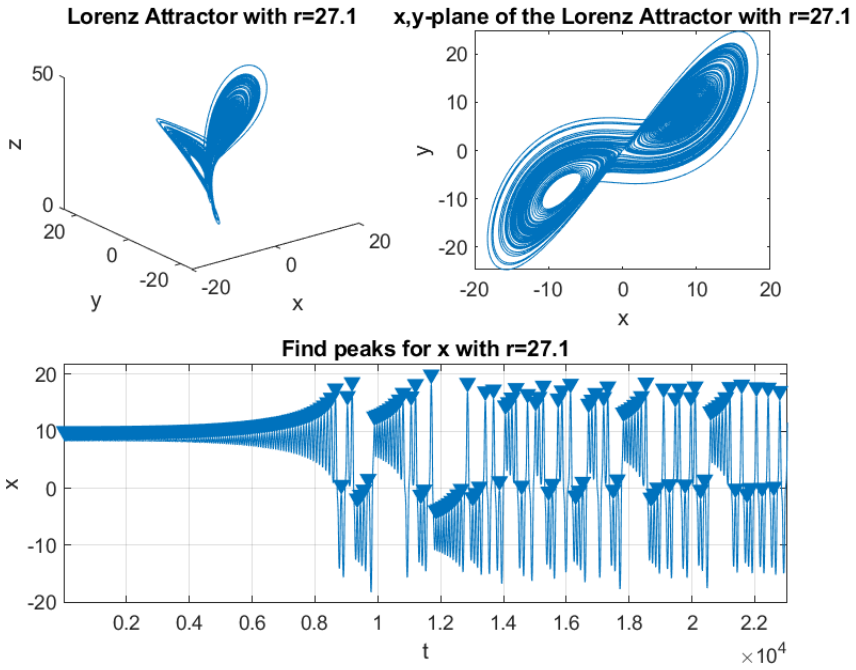
Figure 3.1: Bifurcation diagram using $s = 10$, $b = \frac{8}{3}$, r is varying and $r_H = 24.74$, with a change in r with 0.01. The reason we get negative values in once the system becomes chaotic is that the findpeak, will find peaks even if the value of said peak is less than 0.

Let's start by explaining what a bifurcation diagram is, a bifurcation diagram shows us the peaks of the response during a simulation for a set time, while increasing the value of r. In our case we use Matlabs function $findpeaks$ and plot the resulting peaks against the r value used during that simulation. A show case of one simulation during the creation of the bifurcation diagram is shown in figure 3.2a with $r = 24.1$ where the equilibrium points are stable but quite close to $r_H = 24.74$ and figure 3.2b with $r = 27.1$ where the equilibrium points are no longer stable and the system behaves chaotically.
As we can see from figure 3.1 there is no value plotted for $r < 1$, this is because the origin is the only stable point, as mentioned in Strogatz "Nonlinear Dynamics and Chaos"[5] page 315. We will therefore start by taking a look at $1 < r < r_H$ we can see that the the plot only shows that the peaks are concentrated at a single point, which is in line the fact that the equilibrium points $(\pm\sqrt{b(r-1)}, \pm\sqrt{b(r-1)}, r-1)$ are linearly stable for $r < r_H$. Fi-

nally we take a look at $r > r_H$, where we can see that we get a lot of points on the graph. These points shows us that if we choose any r larger than r_H the system will be chaotic.



(a) An example of the results of simulation and use of $findpeaks$ with $r = 24.1$. These are fast decaying oscillations toward a stable equilibrium point



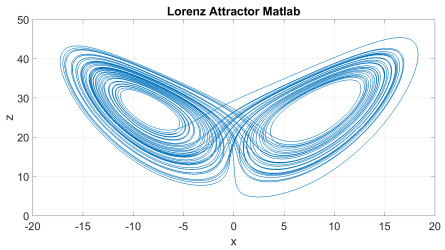
(b) An example of the results of simulation and use of $findpeaks$ with $r = 27.1$

Figure 3.2: Figure (a) shows the fast decaying oscillations toward a stable equilibrium point. Figure (b) shows an increase in oscillation until system becomes chaotic.

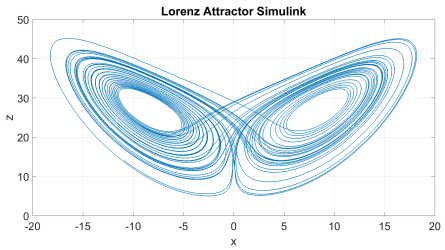
3.1.1 Model Validation

Experiment 1

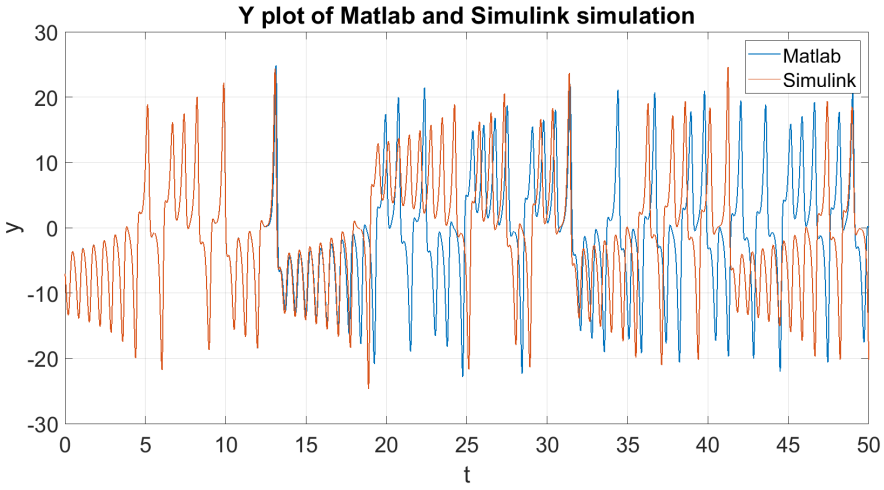
In this experiment we will use the values in table 3.1 under *Experiment 1* to first simulate the Matlab script, listing 2.1, and then plot the response such that we acquire the Lorenz Attractor. Then using the same coefficient values in the Simulink model from figure 2.2, which will give us room to compare the results, or see how similar the two different methods are. The results are shown in figure 3.3.



(a) Lorenz attractor, Matlab



(b) Lorenz attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

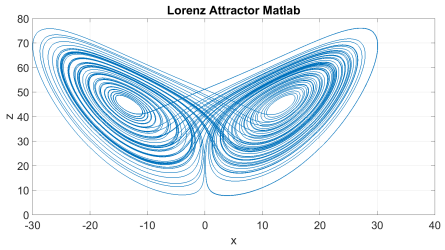
Figure 3.3: (a) and (b) shows the Lorenz attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(-5.1, -7.1, 21.1)$

We start by comparing the Lorenz attractor in figure 3.3a and figure 3.3b; we can from the figures see that they both have the characteristic look of a Lorenz attractor, and can therefore conclude based only on the attractors that their response should be close to the same. Moving on to figure 3.3c we

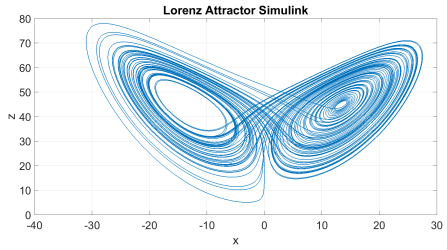
can see that the response starts similar before it diverges after a time, this is most likely caused by the fact that the solver might give small deviations when performing calculations, which in turn can be seen as small changes in the initial conditions. There is also the chance that it is caused by difference in step size for the solver. We can however assume that the responses are close to the same and conclude that we have in fact managed to implement the Lorenz system into both Matlab and Simulink.

Experiment 2

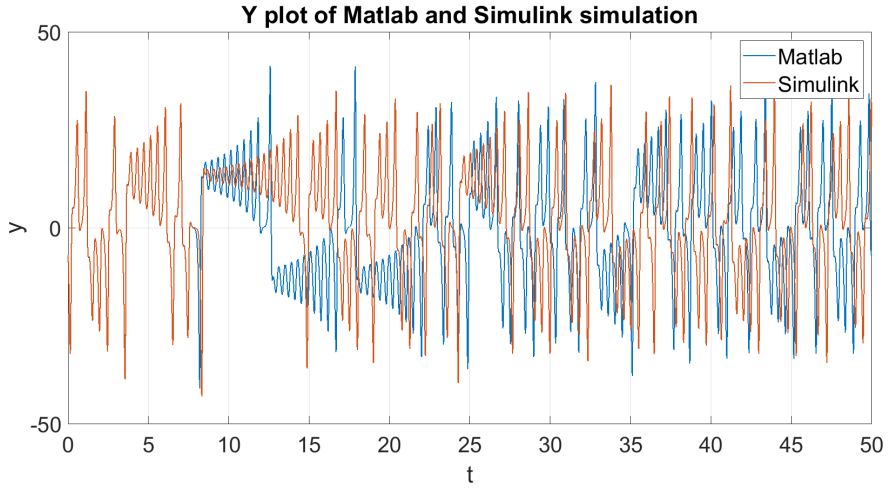
Comparably to experiment 1, we use the the coefficients from table 3.1 to simulate the same Matlab script and Simulink model. Which gives us the result shown in figure 3.4.



(a) Lorenz attractor, Matlab



(b) Lorenz attractor, Simulink



(c) Plot of y for both Matlab(blue) and Simulink(red).

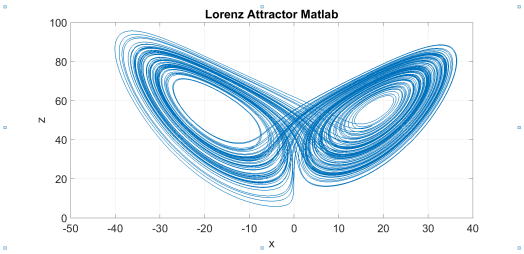
Figure 3.4: (a) and (c) shows the result from the Matlab simulations and (b) and (d) show the result from the Simulink simulation

Looking at the attractors in figure 3.4a and figure 3.4b, and can see there is a bigger difference between the two attractors than there was in experiment

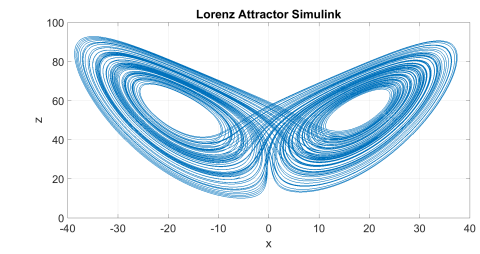
1. The reason can be seen in figure 3.4c; the fact that the response diverges will, as explained in the last experiment, cause the attractors to change. So by looking at the results we can assume that the implementation is as good as it can be, for the step size and tolerance that is used.

Experiment 3

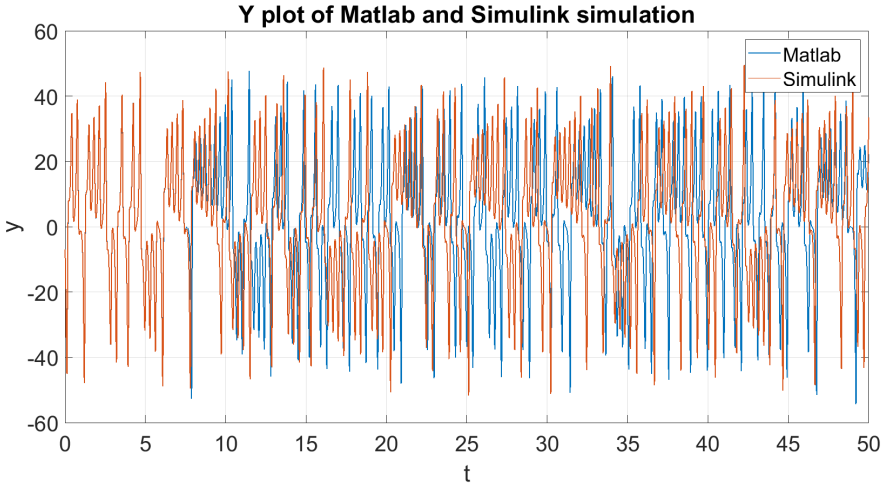
Finishing of the first batch of experiments, we use like in experiment 1 and 2 the values given in table 3.1, giving us the following result, shown in figure 3.5.



(a) Lorenz attractor Matlab



(b) Lorenz attractor Simulink



(c) Plot of y for both Matlab(blue) and Simulink(red).

Figure 3.5: (a) and (c) shows the result from the Matlab simulations and (b) and (d) show the result from the Simulink simulation

The Lorenz attrators, figure 3.5a and figure 3.5b, are a lot more similar than the case was for experiment 2, this is likely caused by the fact that the response, figure 3.5c, does not diverge as much as it does in experiment 2, and because the response keeps the same shape after divergence.

The results are as expected as the systems are the same, so we expected small changes from the simulation in Simulink compared to the simulation in Matlab. Additionally, the only data we changes between the experiments are only the coefficients and not the initial conditions. This can be see in the responses, figure 3.3c, figure 3.4c and figure 3.5c, has the same shape but the larger the coefficient are the bigger the peaks are. The number of oscillations increases, meaning we have more peaks the larger the coefficients are.

3.1.2 Controller

As mentioned in the introduction, chapter 1, we want to prove that the average of a step response at steady state is constant to the step. To achieve this we will use the equations we found in chapter 2.5.1, and the coefficients we chose in table 3.1. We will use equation 2.30 and equation 2.39, the results are placed in table 3.2.

Coefficients	**Experiments**		
	1	2	3
D	13.67	21	29
E	11	17	23
F	720	2854.4	7260
ζ	0.5	0.5	0.5
τ_2	3.98	0.0073	0.004
τ_1	0.019	6.56	8.7
K_i	180	435	834

Table 3.2: Table for coefficients used to calculate the integral gain

Then using the value from table 3.2 we will simulate the models for 5000 seconds and plot the response. During the simulation we will use different steps every 1000 seconds, and then we calculate the average of the last 300 seconds before the next step happens. This should mean that the response is in a steady state; we will do this for the five steps we have. To find the index where our response have reached the correct time, we use Matlabs $find$ function, the $find$ function searches trough all the value until it reaches a threshold set by the user. Figure 3.6 shows us the steps that we will use during the simulations.

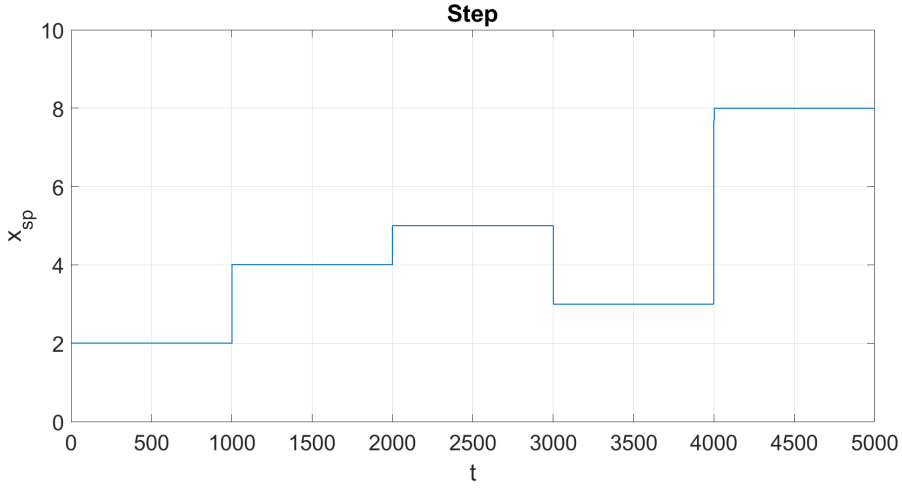
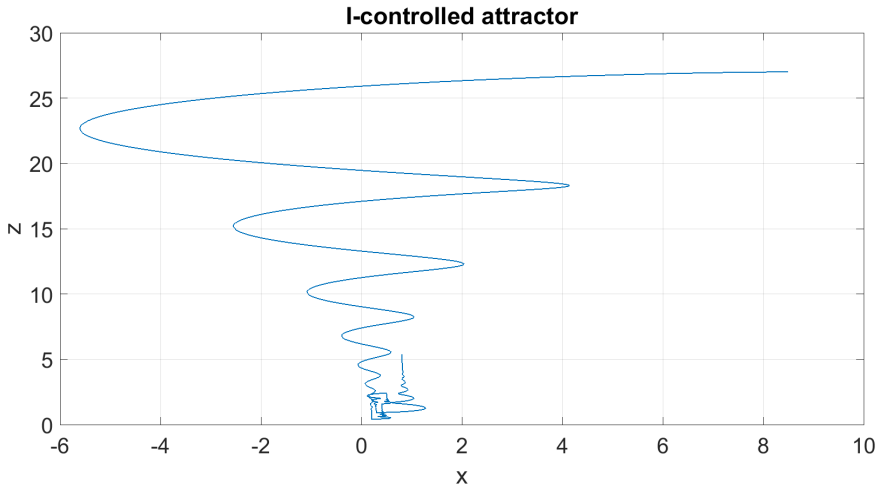


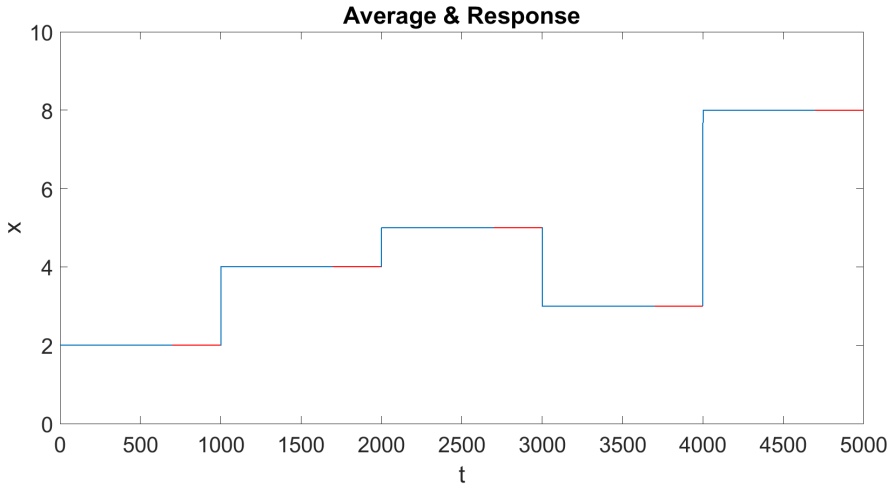
Figure 3.6: The steps used during simulation

Experiment 1

The first experiment consists of using the values in table 3.2 to simulate the Lorenz system and plot the average of the step response. This can be seen in figure 3.7, where we have the mean of the response, and the steps we use.



(a) The attractor we get for the I-controlled Lorenz system



(b) The average of the response (red) and the step used during simulation (blue)

Figure 3.7: Figure (a) shows the attractor we get using the I-controlled Lorenz system. Figure (b) shows the average of the response (red) and the step used during simulation (blue)

By observing figure 3.7a we can see that we do not get the characteristic Lorenz attractor that we usually get during simulations, that is because of

the effect of the integral controller. Moving on to figure 3.7b we can see that the average of the response, in red, is equal to the steps after reaching steady state.

This was the desired result, however the response is not entirely as expected as we expected the response to have chaotic oscillations. This is however not the case, by observing the response in figure 3.8 we see the integral controller gives us a good regulation.

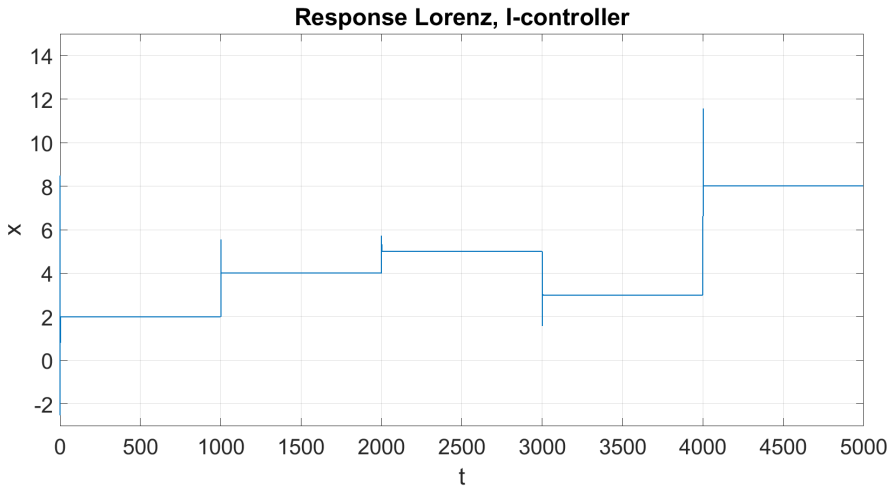
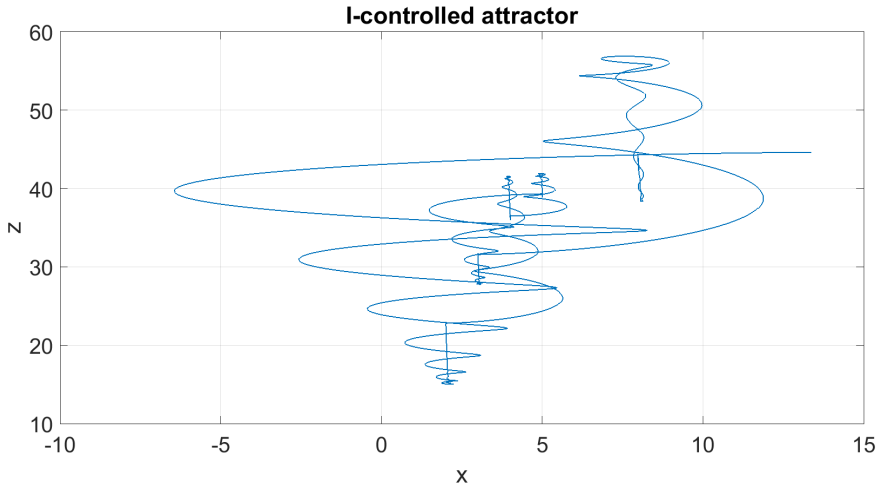


Figure 3.8: Response of the Lorenz system while using an I-controller

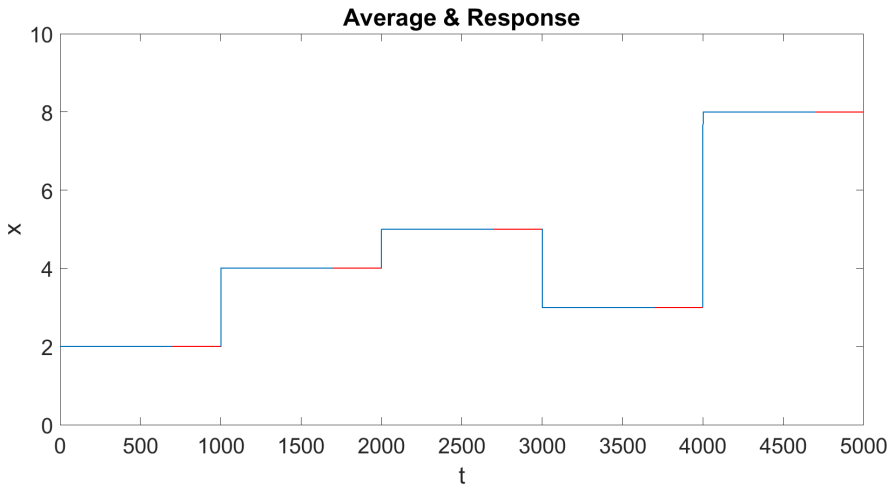
Experiment 2

The next experiment we will use the next values from table 3.1 and table 3.2. We can probably expect that we will get the same type of response as in experiment 1, however we will proceed with the experiment.

The attractor and the response is shown in figure 3.9.



(a) The attractor we get for the I-controlled Lorenz system



(b) The average of the response (red) and the step (blue)

Figure 3.9: Figure (a) shows the attractor we get using the I-controller on the Lorenz system. Figure (b) show the average of the response (red) and the steps (blue)

We can ones again see that the attractor, shown in figure 3.9a, does not have the characteristic Lorenz shape, but this is as expected. Taking a look at the average of the response and the steps, figure 3.9b, we see that we get a perfect match.

This is, as mentioned in experiment 1, exactly the result we wanted. However, as in experiment 1 we did not expect it to be because the of how good the regulations actually was. As proof of how good the controller actually is we have, in figure 3.10, the full response.

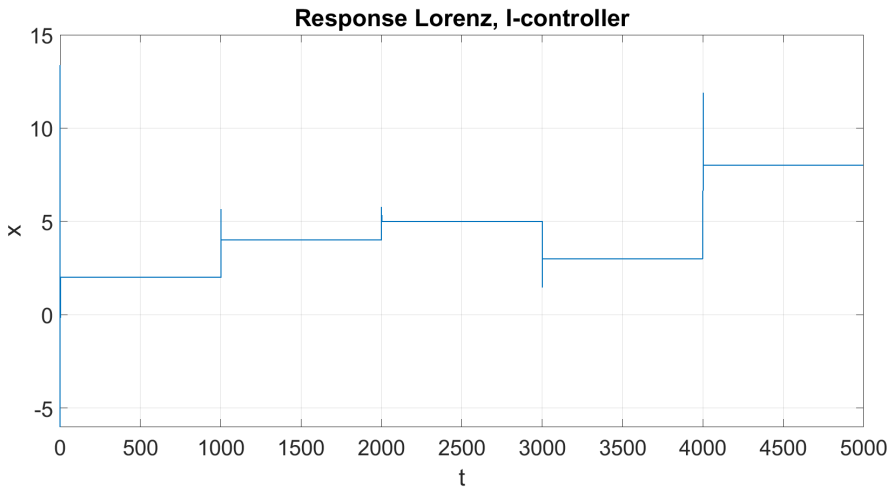
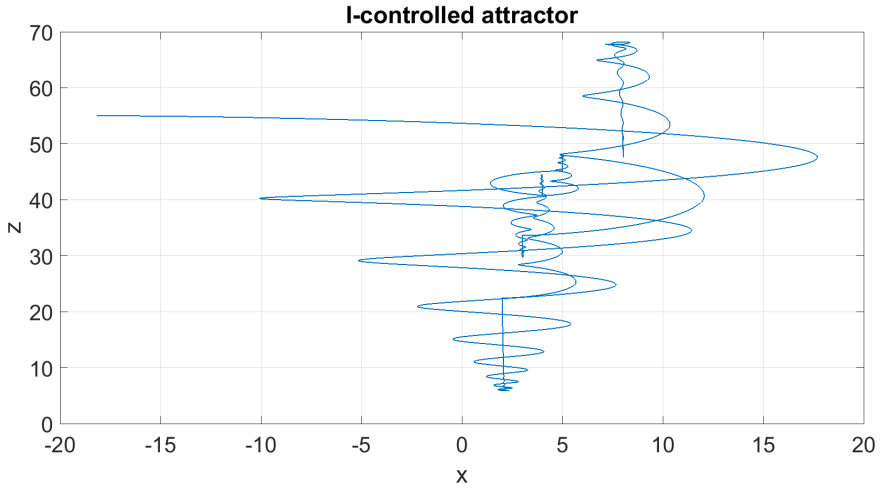


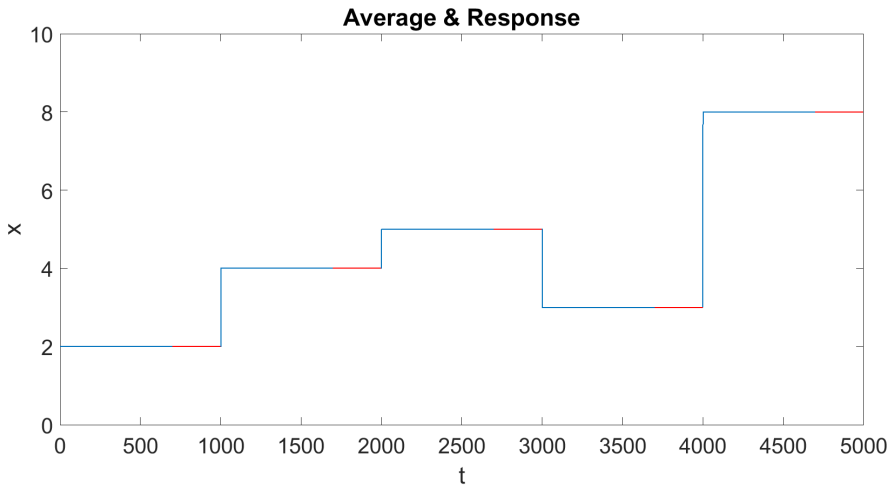
Figure 3.10: Step response of the I-controlled Lorenz system

Experiment 3

The third and last experiment on the Lorenz system is done similar to experiment 1 and 2, using values from table 3.1 and table 3.2. We start by finding the attractor and then the response, both shown in figure 3.11.



(a) Attractor of the I-controlled Lorenz system



(b) The average of the response (red) and step (blue)

Figure 3.11: Figure (a) shows the attractor we get using the I-controller on the Lorenz system. Figure (b) shows the average of the response (red) and the steps (blue)

Similar to the other experiments we have exactly the answer we wanted, however just as for the other experiments we can see that the response is not as expected, figure 3.12.

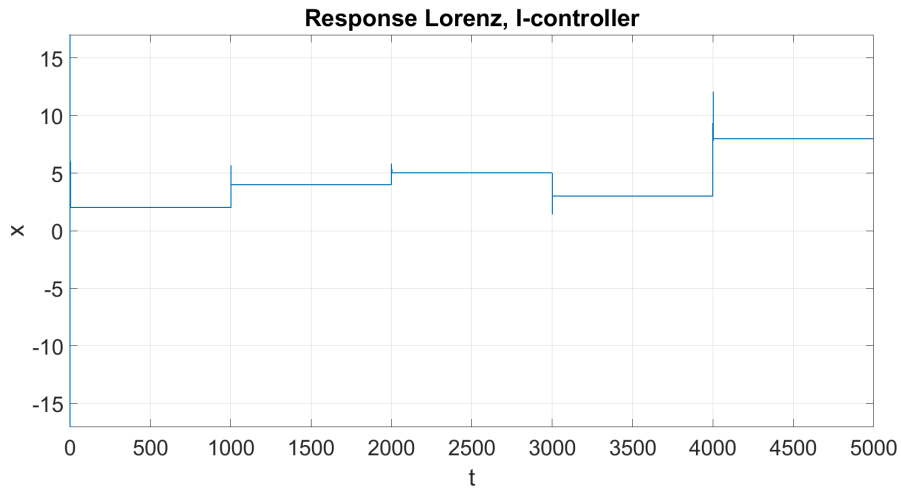


Figure 3.12: Step response of the I-controlled Lorenz system

As mentioned in the introduction, chapter 1, we wanted to use an I-controller on the Lorenz system and then from what we assumed would be chaotic oscillations take the average. The average should then resemble the step used to get the response. However, in our case we get a response that resembles the step without us finding the average of the response. This in itself is actually just as valid as the expected result. So to conclude these experiments we can say that the result are more than good enough.

3.2 Lorenz as an electrical circuit

In this chapter we will take a look at how the Lorenz system will work as an electrical circuit, chapter 2.2, and use information from that chapter to calculate the coefficients and run some simulations. To make it easier we will use the same coefficients as in table 3.1, and compare the results with the ones from chapter 3.1.1.

3.2.1 Simulink

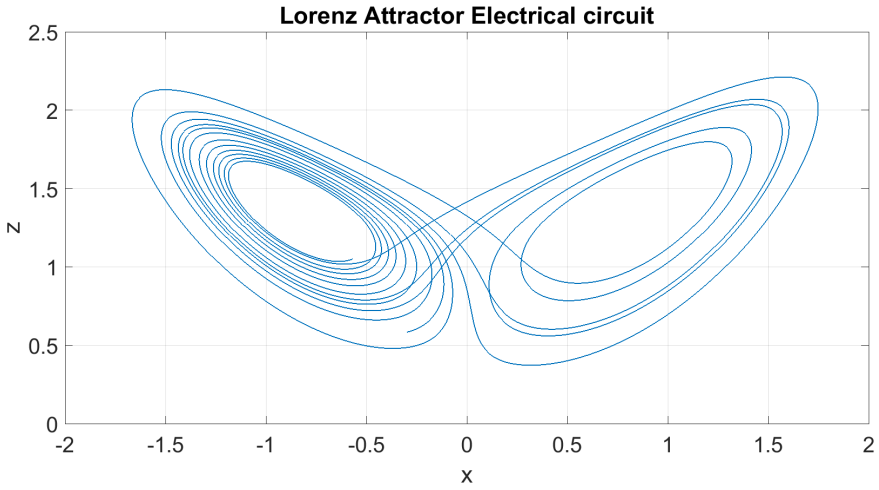
As mentioned in chapter 2.2 we can change the value of R_5, R_{11} and R_{18} to change the values for s, b and r. So using equation 2.8 we can solve for R_5, R_{11} and R_{18} using $sf = 2505$. We will then get the results shown in table 3.3 and can then move on to the experiments.

Coefficients	**Experiments**		
	1	2	
R_5	$7.9840 \cdot 10^4$	49900	Ω
R_{11}	$2.6151 \cdot 10^4$	63400	Ω
R_{18}	$3.5844 \cdot 10^4$	66500	Ω

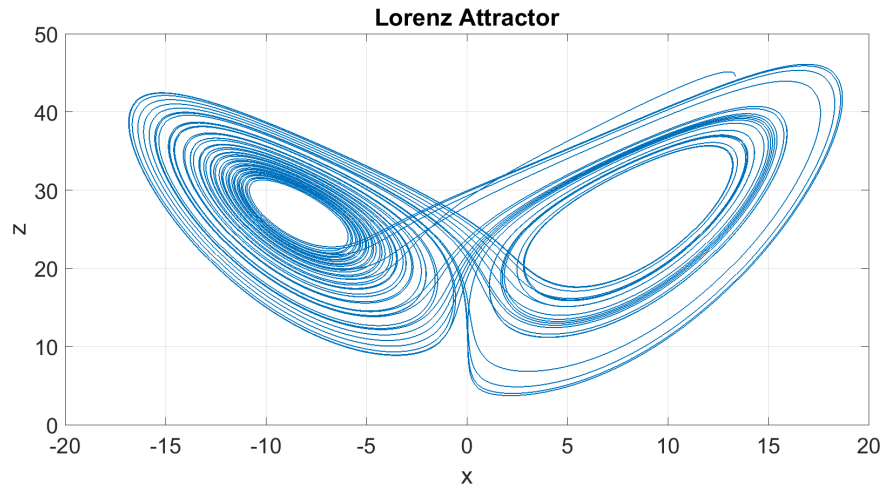
Table 3.3: Table for resistors value to find correct value for s, b and r.

Experiment 1

Since we now have the resistor values, we can start simulating the model and comparing them. The biggest problem with comparing the responses is that the time axes are not the same. We start by finding the attractor of experiment 1 and experiment 2 and compare them to see if both of the are Lorenz systems with chaotic behaviour, the attractors are shown in figure 3.13.



(a) Lorenz attactor of experiment 1

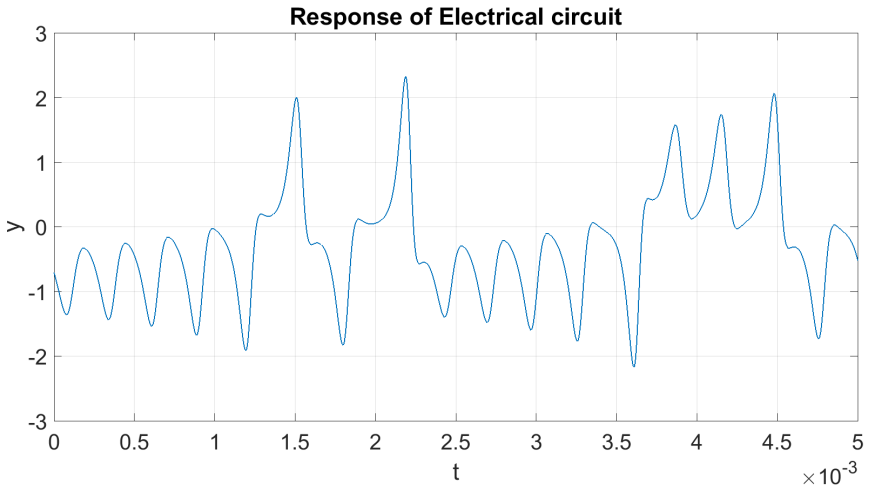


(b) Lorenz attractor of an ordinary Lorenz system

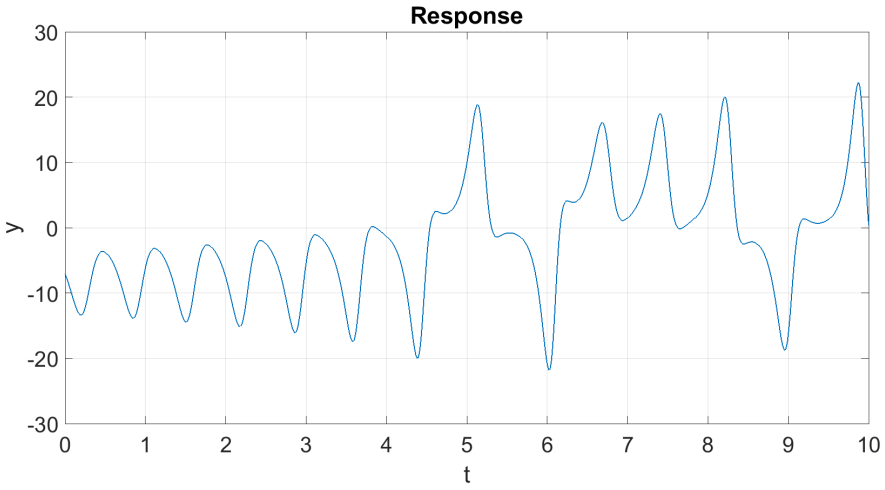
Figure 3.13: The Lorenz attractor of experiment 1 and Lorenz attractor from a normal Lorenz system

In figure 3.13a we have the Lorenz attractor, using the resistor values from table 3.3. In figure 3.13b we have an attractor using the values from table 3.1 for s, b and r. Examining the attractors we can see that they have similar shape, the biggest different is the simulation time. Which is the reason why figure 3.13b has denser coloring than the Lorenz attractor in figure 3.13a, another difference is the scale of the figures. This however is caused by the fact that the electrical circuit is a scaled version of the original Lorens system. Moving on from the attractor we will now take a look at the response of the electrical circuit, figure 3.14, where we will take a look at

how the response compare with the response of an ordinary Lorenz system.



(a) Response of experiment 1



(b) Response of an ordinary Lorenz system

Figure 3.14: The response of experiment 1 and ordinary Lorenz system

Figure 3.14a shows us the response of the electrical circuit, and figure 3.14b shows the response of an ordinary Lorenz system. As we mentioned earlier an in chapter 2.2 we know that the electrical system is a scaled model, so we can assume that this change in variable would result in difference in response, according to the definition of chaos. However, we can also assume that the fact that the initial condition is different from each other, scaled by the values given in chapter 2.2, but we can assume that the system is a good representation of a Lorenz system. It would be better to compare the elec-

trical circuit to a model of the scaled system, using scaled initial conditions. We can assume that the response would start the same before the response will diverge, like we got in chapter 3.1.1.

To prove the assumption we will make a Simulink model based of equation 2.7, and check the response of that model.

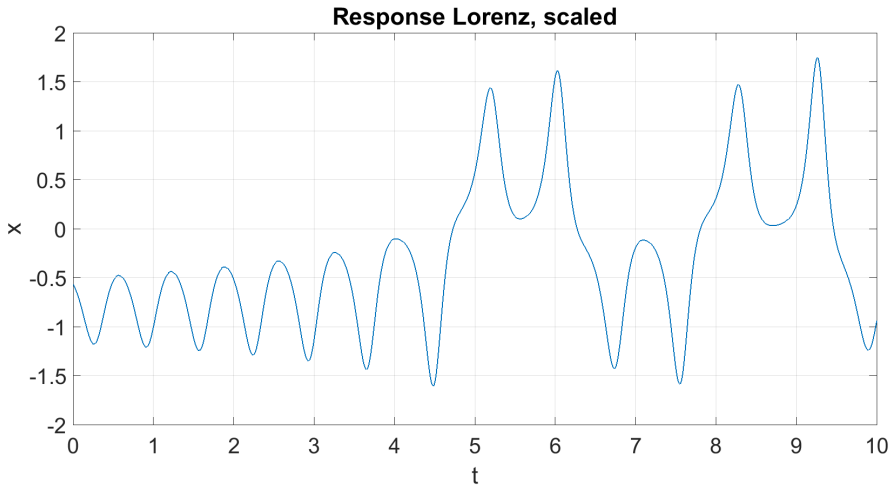
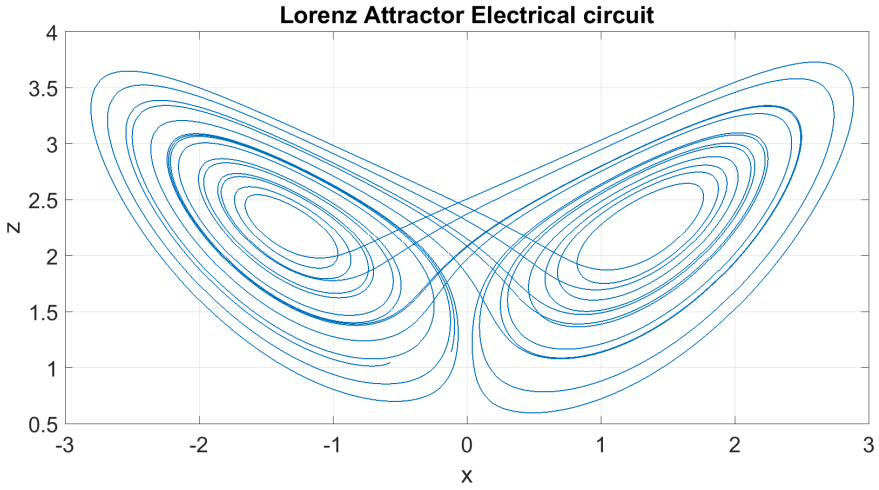


Figure 3.15: Response of the scaled model using $s = 10$, $b = 8/3$ and $r = 28$

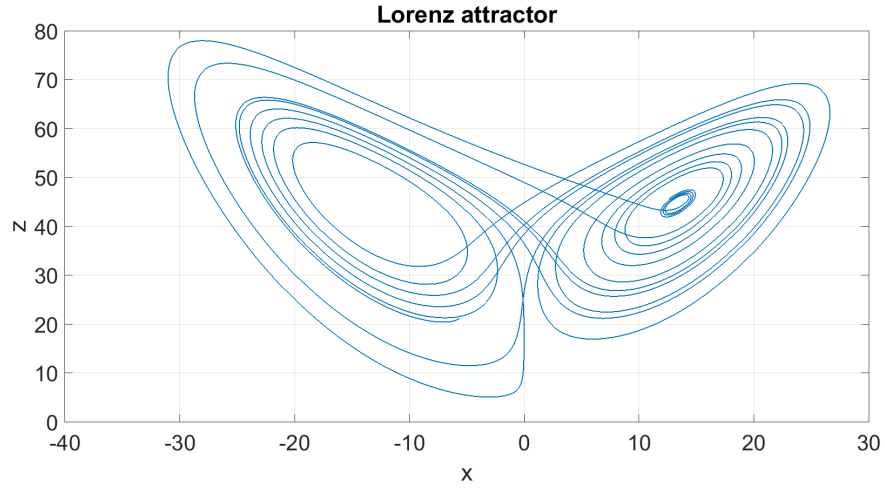
We can see in figure 3.15 that our assumtion is not entirely correct as the response look a lot closer to the response from figure 3.14a, however we can still conclude with a relatively good recreation.

Experiment 2

In the second experiment we use the second set of coefficient values from table 3.1, and then simulate the model and compare it to an ordinary Lorenz system. We start by looking at the attractor, figure 3.16, and compare them.



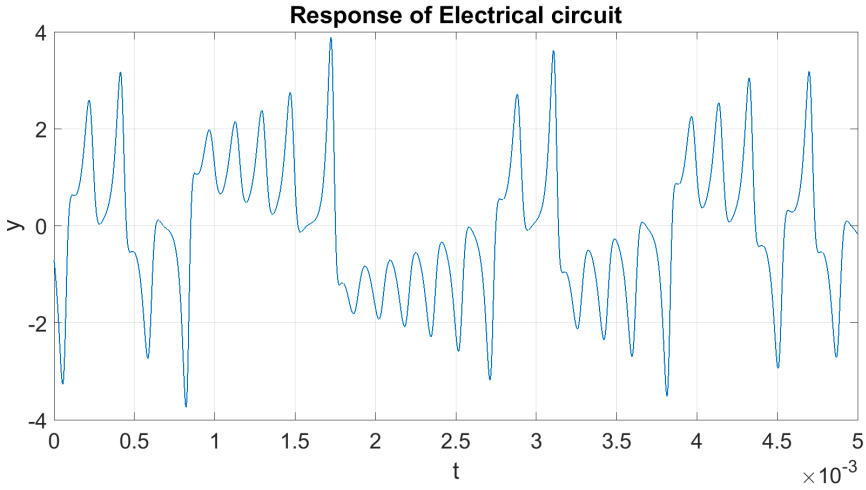
(a) Lorenz attractor experiment 2



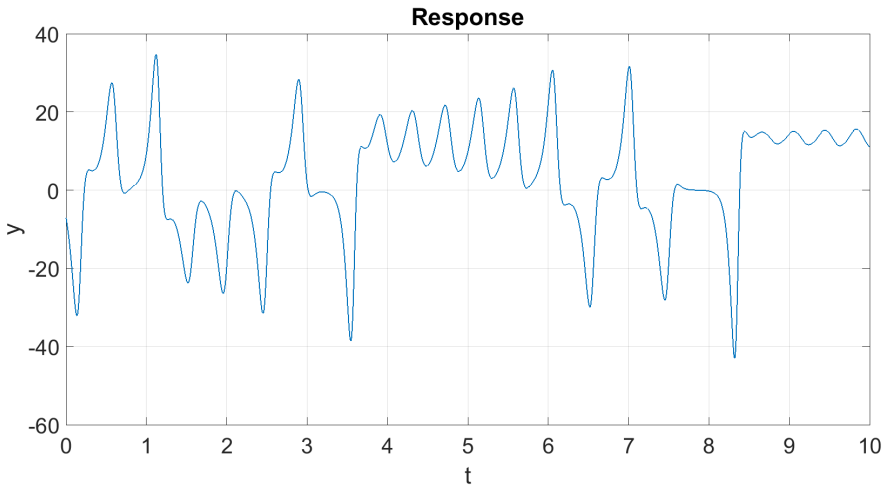
(b) Lorenz attractor of an ordinary Lorenz system

Figure 3.16: Lorenz attractor of experiment 2 and an ordinary Lorenz system

Comparing figure 3.16a and figure 3.16b we can see that both of them do have the classic Lorenz attractor butterfly look. This means that both of them are Lorenz systems. However, we can also see that the attractor are far from similar, this is as mentioned in experiment 1 most likely caused by the fact that the electrial circuit is a scaled version of the Lorenz system and we have therefore different initial condition. Now moving on to the response of the electrical circuit, figure 3.17.



(a) Response experiment 2



(b) Response of an ordinary Lorenz system

Figure 3.17: Response for experiment 2 and an ordinary Lorenz system

Analysing the figure 3.17a and figure 3.17b we see that the response is far from similar. Thus we can make the assumption, as we did for experiment 1, that the fact that the electrical circuit is a scaled version. So comparing the response of the electrical circuit with the response of an ordinary Lorenz system where we use different initial condition, will as expected not give the desired result. However, we can like we did for experiment 1 use a scaled Simulink model, figure 3.18.

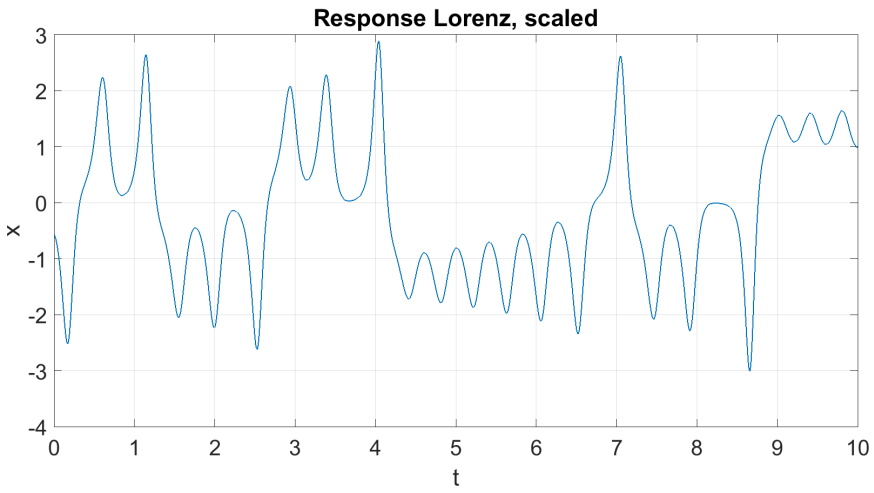


Figure 3.18: Response of a scaled Lorenz system using $s = 16$, $b = 4$ and $r = 45.6$

From the response in figure 3.18 we can see that the response are as expected and that the electrical circuit is a good representation of a Lorenz system.

3.3 Chua

The Chua section of this thesis consists of one Matlab program and one Simulink model. The goal of this section is to first validate the Matlab program as a good approach to the Chua circuit, then do the same for the Simulink model. To do these validations we will need value for our coefficients, we will therefore use the same set up as we did for the Lorenz system. Thus we will perform three experiment each with a change in a and b, to see which value we want to use we have made a bifurcation diagram, figure 3.19.

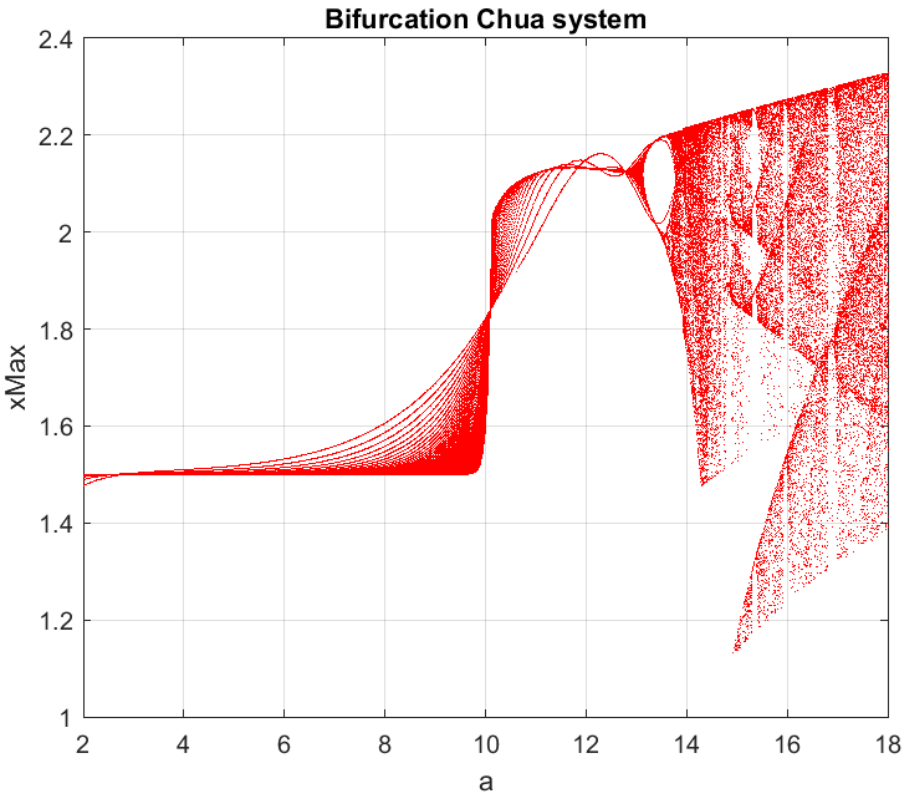


Figure 3.19: Bifurcation diagram of the Chua circuit using $m_0 = -\frac{8}{7}$, $m_1 = -\frac{5}{7}$ and $b = 28$, a is varying. With a change in a of 0.01

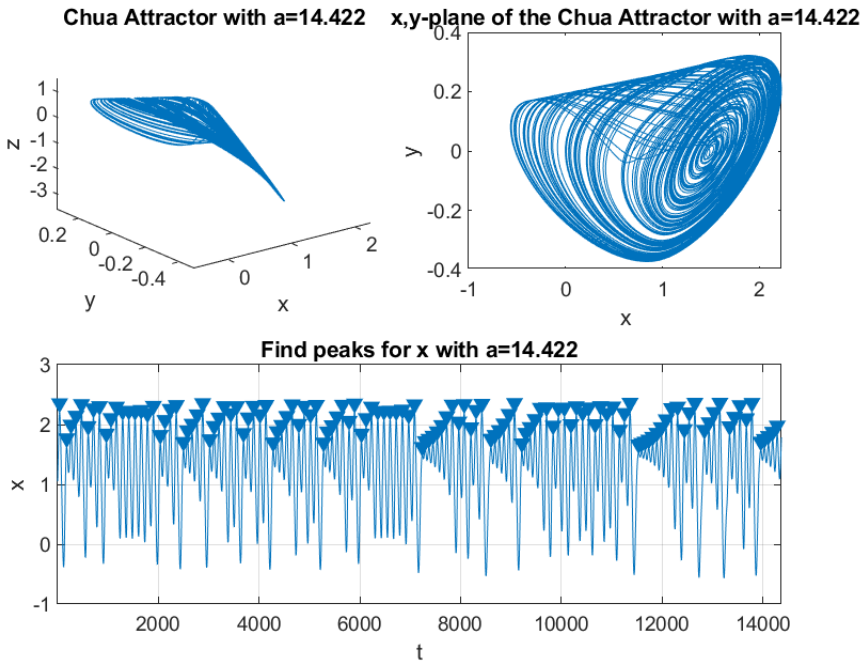
The bifurcation diagram is made by using the same methods as for the Lorenz system. We can by observing the bifurcation diagram see that the system does not truly become chaotic until $a > 14.422$, from this observation we can assume that choosing an value for a smaller than this will not result in a chaotic system. Thus we will choose larger values for a, the coefficient

values we have chosen to use are shown in table 3.4.

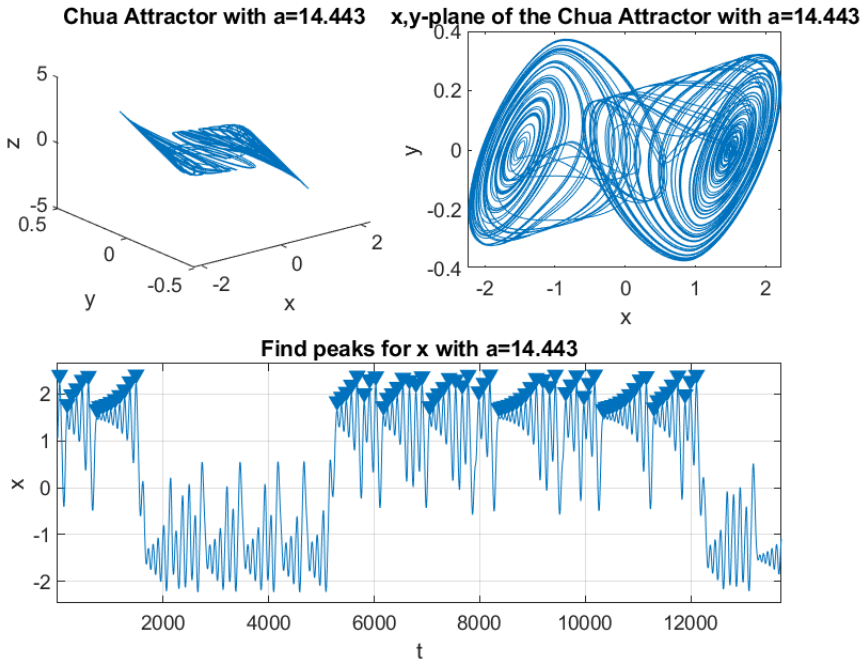
Coefficients	**Experiments**		
	1	2	3
m_0	$-\frac{8}{7}$	$-\frac{8}{7}$	$-\frac{8}{7}$
m_1	$-\frac{5}{7}$	$-\frac{5}{7}$	$-\frac{5}{7}$
a	15.6	16.9	18
b	28	27	34

Table 3.4: Table for coefficients used during experiments, Chua

However, making the assumptions that a is not chaotic is $a < 14.422$ can be proven by taking a look at the figure 3.20.



(a) An example of results of simulation and use of $findpeaks$ with $a = 14.422$



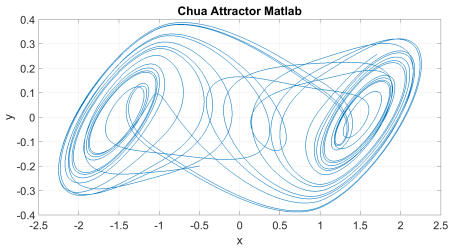
(b) An example of results of simulation and use of $findpeaks$ with $a = 14.443$

Figure 3.20: Figure (a) shows the use an almost chaotic, response from the Chua circuit. Figure (b) shows a chaotic response.

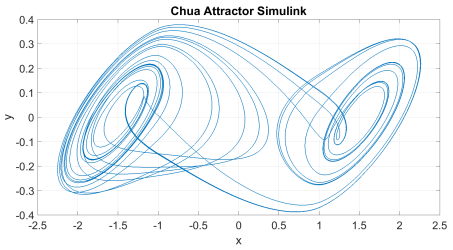
3.3.1 Model validation

Experiment 1

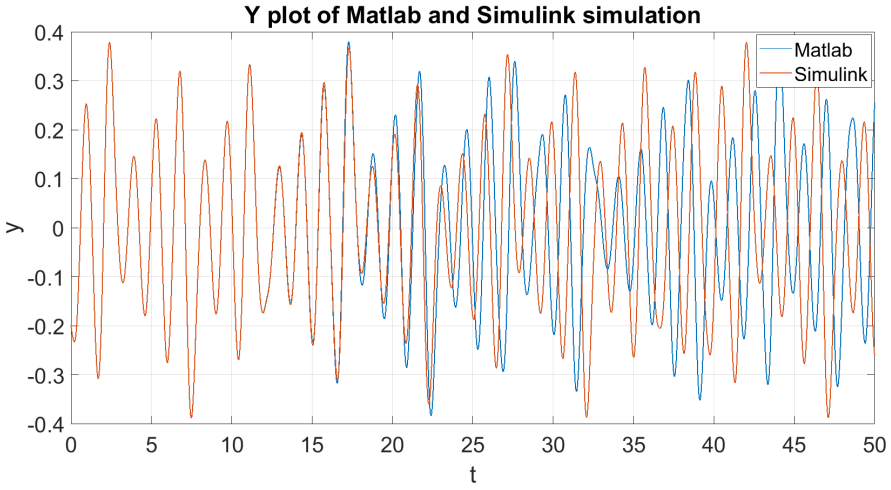
This experiment will use the values from column 1 in table 3.4 to simulate the Matlab program, listing 2.2, and then simulate the Simulink model. After the simulation we will compare the attractors and response to see if the models are good representation of the Chua circuit. The results are shown in figure 3.21.



(a) Chua attractor, Matlab



(b) Chua attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

Figure 3.21: (a) and (b) shows the Chua attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(-0.5, -0.2, 0)$

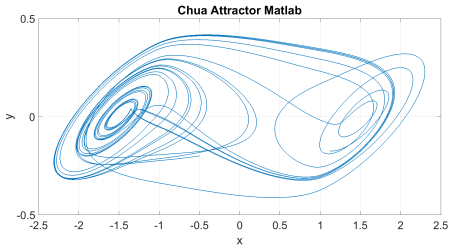
We can from figure 3.21a and figure 3.21b see that the attrators both have the characteristic appearance of the Chua attractor. However, we can see there is a difference between the two attractors, we can safely say that both of the models, Matlab and Simulink, are chaotic Chua models.

Moving onto figure 3.21c we can see that the response diverges after some time, even if the coefficients and the initial conditions have the same value, this can also be seen in the attractors. The cause is most likely the solver used in the two models, even if the tolerance is the same for both systems.

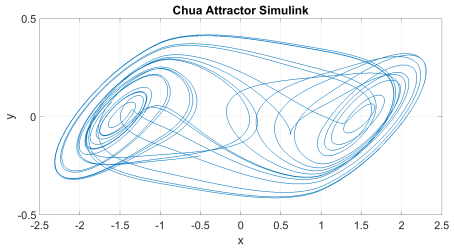
We see that the response starts the same before they after a certain time the response diverges, this is similar as it was for the Lorenz system. So we will make the same assumptions as we did for the Lorenz system, and concluded that with the use of the coefficients used in this experiment is implemented in both Matlab and Simulink.

Experiment 2

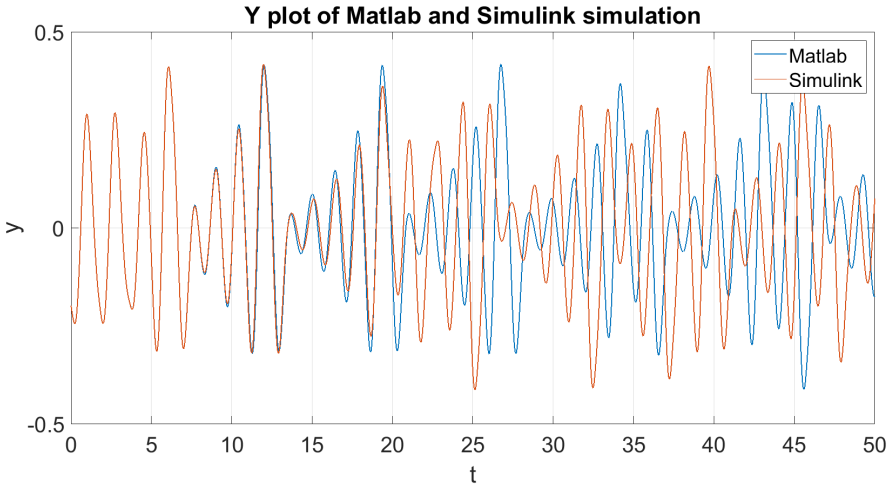
The next experiment uses $a = 16.9$ and $b = 27$, and we want to check if the Matlab model and Simulink model is a good implementation of the Chua circuit. So to get results we put them into the Matlab and simulate both the Matlab and Simulink model, this gives us the results shown in figure 3.22.



(a) Chua attractor, Matlab



(b) Chua attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

Figure 3.22: (a) and (b) shows the Chua attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(-0.5, -0.2, 0)$

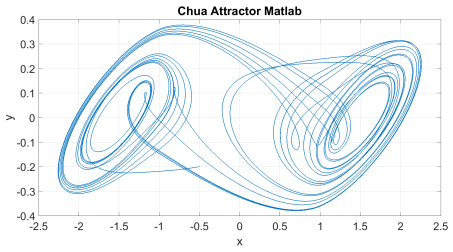
We start by comparing figure 3.22a and figure 3.22b, the attractors, and see that both of them have, like in experiment 2, the characteristic shape and form of the Chua attractor. However, we can also see that there is a relative big difference in the two attractors.

Moving onto figure 3.22c we see that the response diverges after a time, and comparing the response to the one in experiment 1 we can see that the divergence happens earlier. The observation of the first two experiments we can assume the time before divergence is caused by the size of a and b.

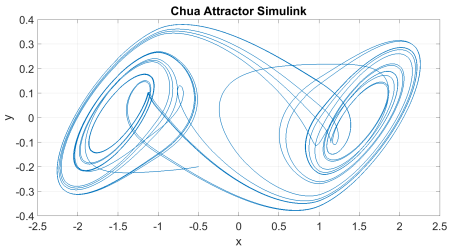
We can also assume the same as in experiment 1, that the divergence is caused by the way the solver in Matlab and the solver in Simulink calculate the response. We can therefore conclude that both of the systems are good representation of the Chua circuit.

Experiment 3

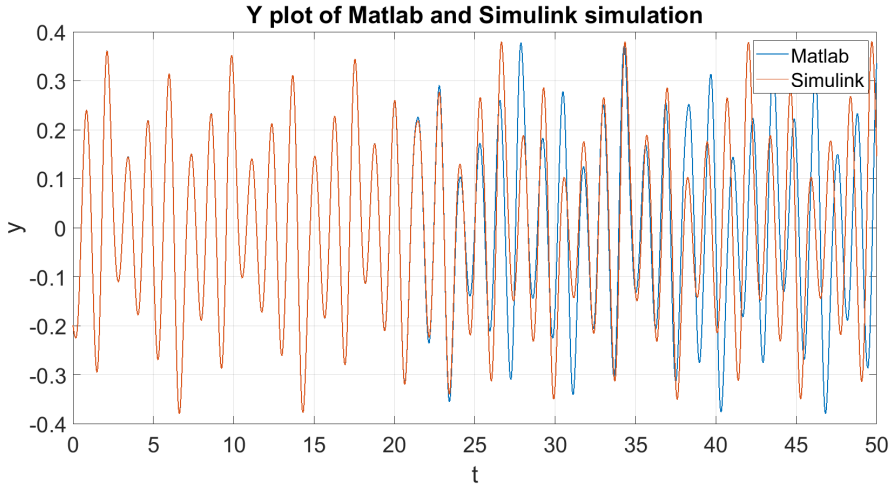
In this experiment, we will increase a and b with a larger amount. To see if this causes any problems or changes to the response, the result is shown in figure 3.23.



(a) Chua attractor, Matlab



(b) Chua attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

Figure 3.23: (a) and (b) shows the Chua attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(-0.5, -0.2, 0)$

Taking a look at the attractor, figure 3.23a and figure 3.23b, we can see that they look more alike than the attractors from experiment 1 and 2. While the response uses longer time before diverging, figure 3.23c, and we can like we did for the other experiments conclude with that both of the models are good implementation of the Chua circuit.

We can therefore conclude with that, at least with the use of the coefficients shown in table 3.4, the Matlab program and the Simulink model are good representation of the Chua circuit. So we can assume that the use of the Simulink model can be used during the implementation of the I-controller.

3.3.2 Controller

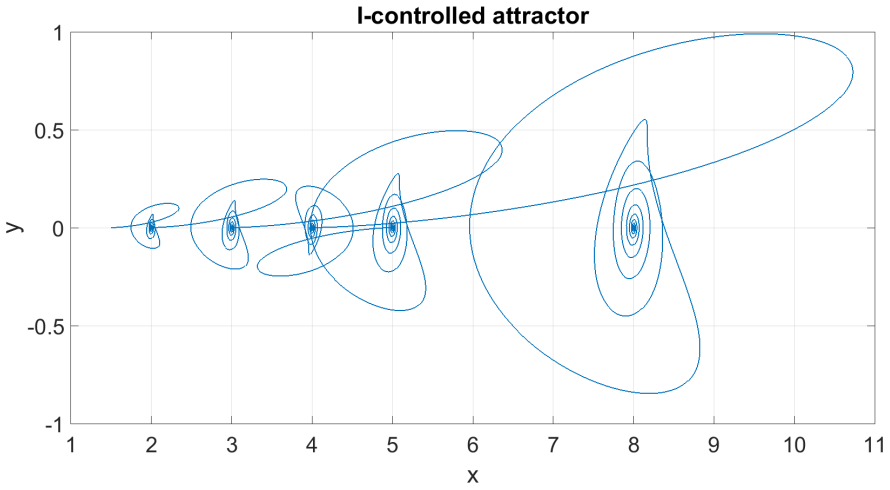
We will in this section make the calculations and simulate the I-controller Chua circuit, the calculations and coefficient values are shown in table 3.5. This will work the same way as it did during our experiments for the Lorenz system.

Coefficients	**Experiments**		
	1	2	3
D	5.46	5.8	6.14
E	16.86	15	21.14
F	124	130	175
ζ	0.5	0.5	0.5
τ_2	1.02	1.02	1.01
τ_1	0.085	0.083	0.07
K_i	134	140	186

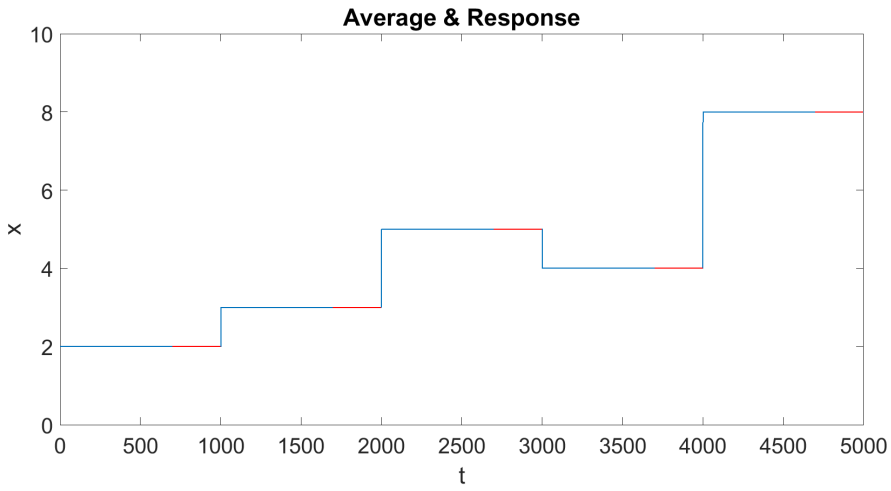
Table 3.5: Table for coefficients used to calculate the integral gain

Experiment 1

Using the values in table 3.5, we will simulate the Simulink model and see how the average of the response compare to the step and the attractor, figure 3.24.



(a) Attractor for I-controlled Chua circuit



(b) The average of the response in steady state (red) and step (blue)

Figure 3.24: Figure (a) Shows the attractor of the I-controlled Chua circuit. Figure (b) shows the the average of the response in steady state (red) and the step (blue)

We start by observing the attractor, figure 3.24a, and see that we do not have the attractor that is associated with the Chua circuit, this is however as expected. Since the attractor is dependent on the response of the system we will with the change in the response, because of the I-controller, not have a typical Chua response. Moving onto the response, figure 3.24b, we see that the red lines that indicates the value of the average of the response in steady state is on the line of the step. This is exactly what we wanted, however, this is because the response in itself is good, figure 3.25.

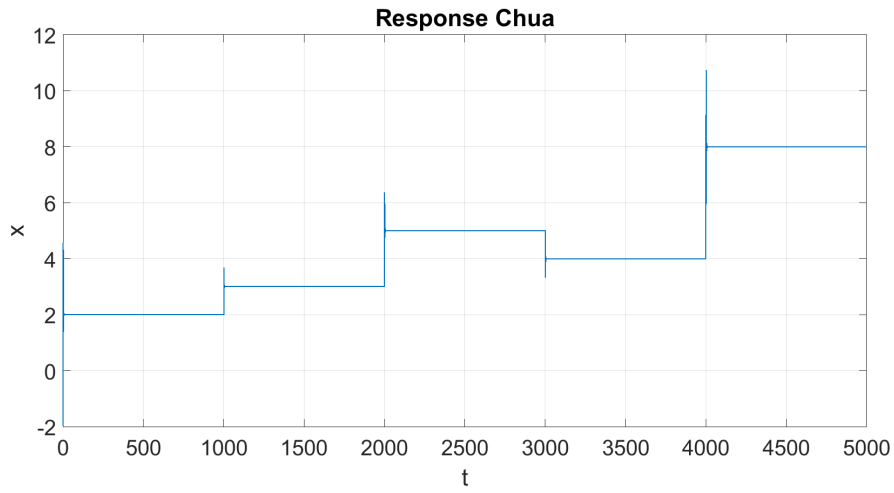


Figure 3.25: The step response of the I-controlled Chua circuit

We expected the response to oscillate around the set point. However, by reducing the step size we can, unlike for the Lorenz system, obtain a response that oscillate around the set points. The response of a simulation using this change in step size is shown in figure 3.26.

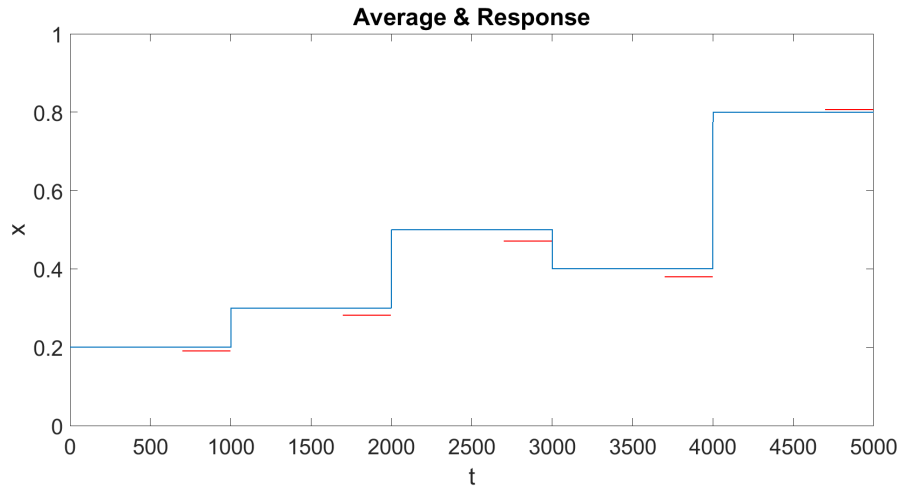


Figure 3.26: The average of the response in steady state (red) and step (blue)

By observing figure 3.26 we see that the result is not as good compared to the result we get when we use a bigger step size.

However, this do prove the assumption we had about the I-controller. That when a chaotic system is under the effect of the integral controller the

average of the oscillations from the response is close to or equal to the step, the step response is shown in figure 3.27.

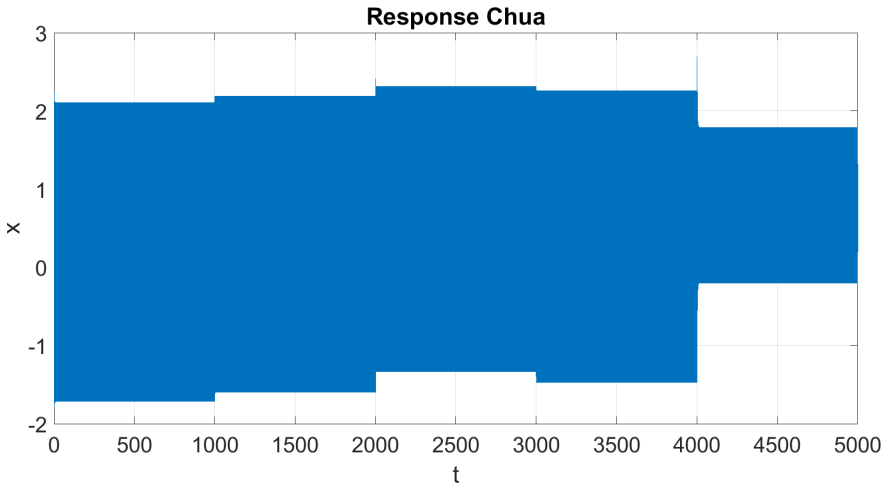
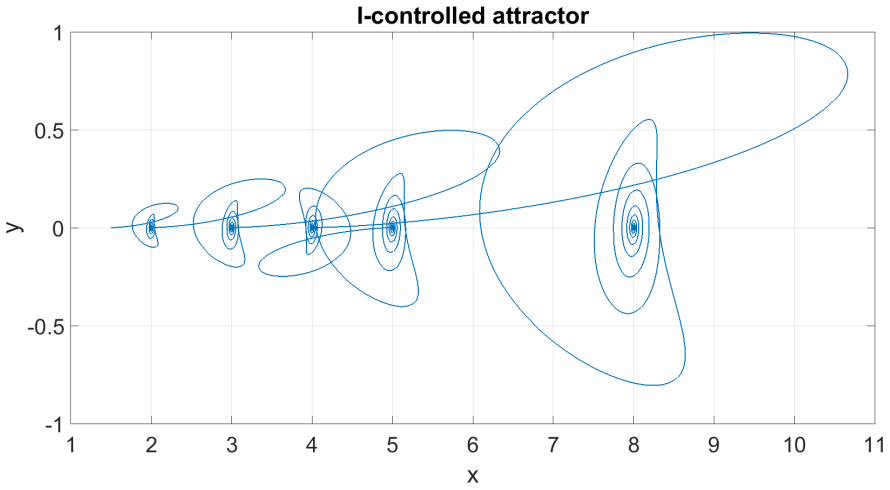


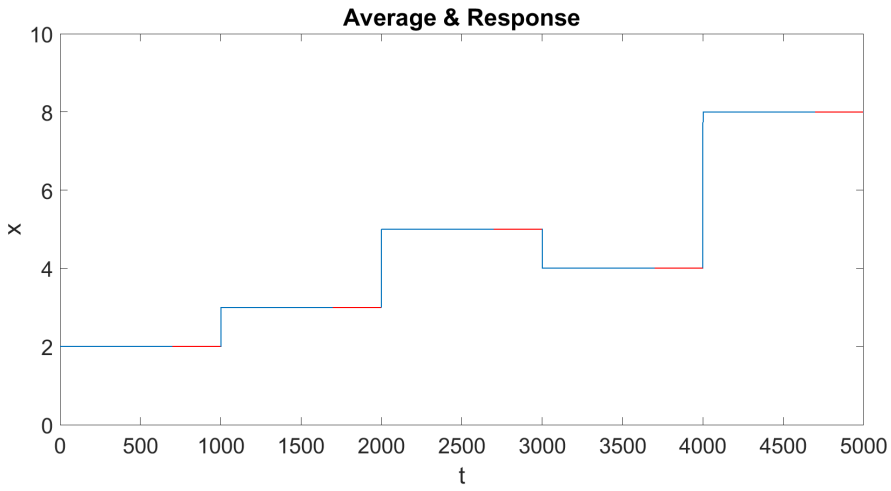
Figure 3.27: Step response for the I-controlled Chua circuit using a smaller step size

Experiment 2

We will then move on to the next experiment and here we will use $= 16.9$ and $b = 27$, using these values we will reproduce the experiment 1. Starting with finding the attractor and the response, figure 3.28.



(a) Attractor for I-controlled Chua circuit



(b) The average of the response in steady state (red) and step (blue)

Figure 3.28: Figure (a) Shows the attractor of the I-controlled Chua circuit. Figure (b) shows the the average of the response in steady state (red) and the step (blue)

Figure 3.28a and figure 3.28b are comparable to the results from experiment 1, and we can therefore assume that the response will have the same level of regulations. The response is shown in figure 3.29.

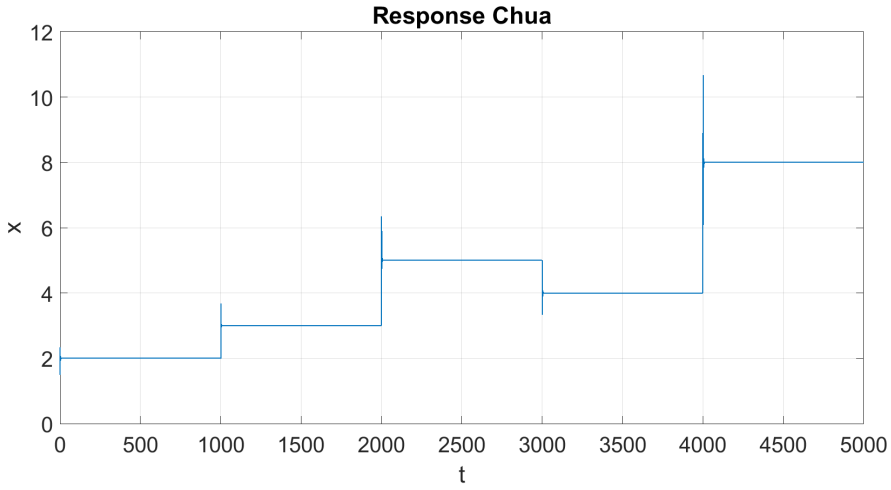


Figure 3.29: Response of the I-controlled Chua circuit

We will also check if we get the same type off effect by reducing the step size as we did in the experiment 1. The results from figure 3.30 shows us that this is actually the case.

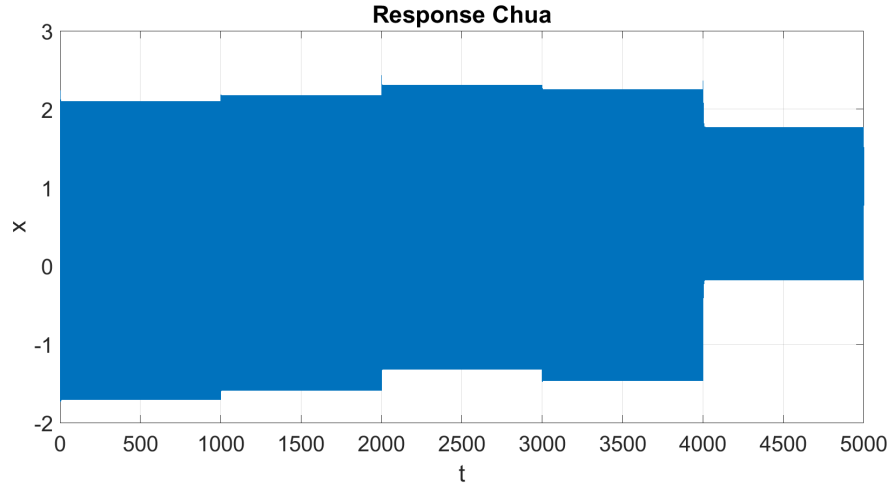


Figure 3.30: Response of the I-controlled Chua circuit with reduced step size

From the average of the response in steady state, figure 3.31, we see that there is a bit of deviation from the average to the actual step. Therefore we will conclude with that the result is quite good for larger step sizes and slightly worse for results where the step size is smaller than 1.

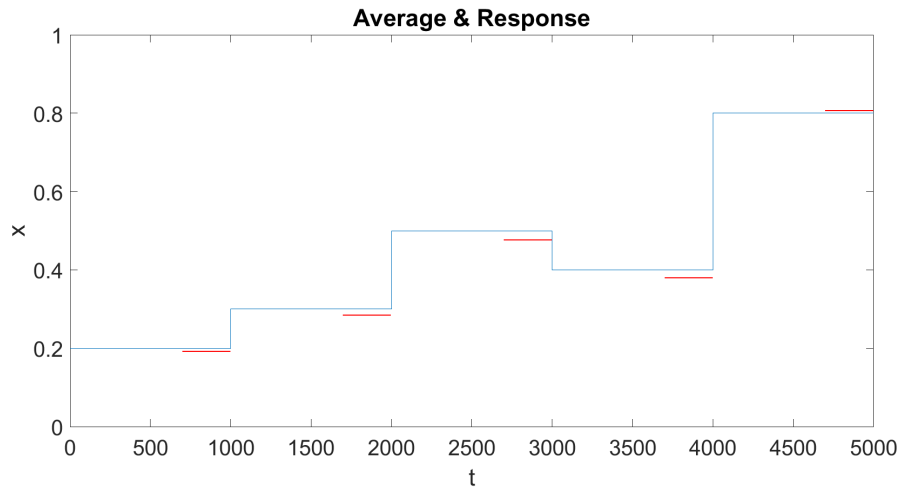
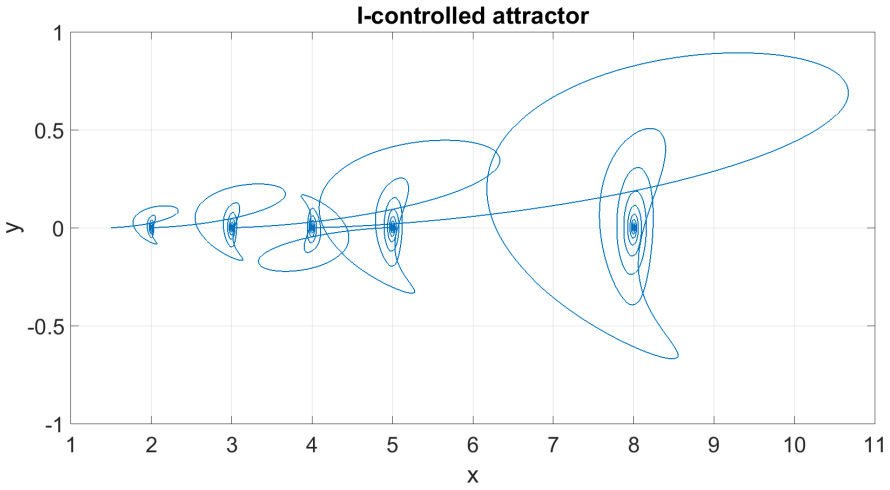


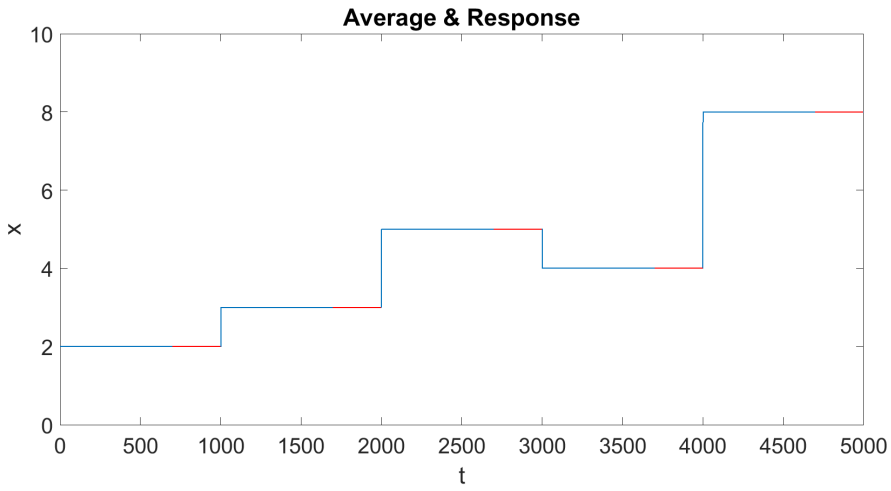
Figure 3.31: Average of the step response in steady state (red) step (blue)

Experiment 3

The last experiment will like experiment 1 and 2 use the coefficient value from table 3.4 and table 3.5 to simulate the Simulink model. Giving us the response and attractor shown in figure 3.32.



(a) Attractor for I-controlled Chua circuit



(b) The average of the response in steady state (red) and step (blue)

Figure 3.32: Figure (a) Shows the attractor of the I-controlled Chua circuit. Figure (b) shows the the average of the response in steady state (red) and the step (blue)

From figure 3.32a and figure 3.32b we that the results are really good, just as they were in experiment 1 and 2. From the fact that they are similar we can make the assumption that the results are good. The response of the simulation is shown in figure 3.33.

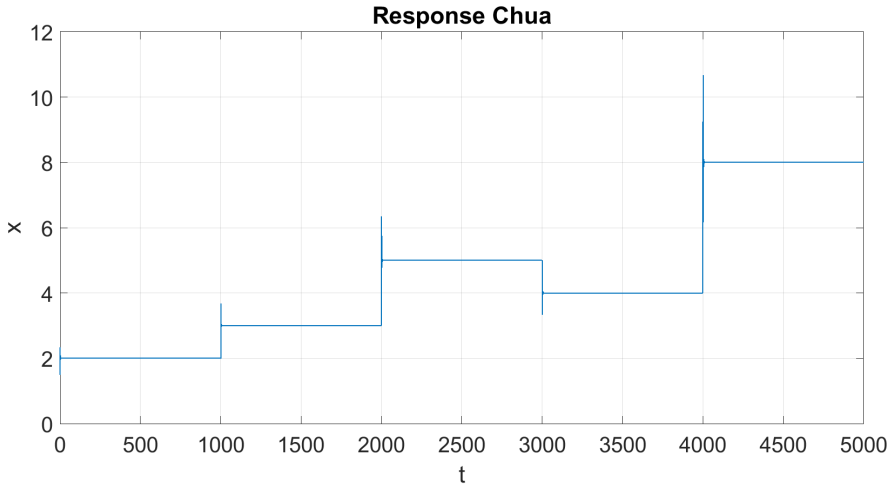


Figure 3.33: Response of the I-controlled Chua circuit

Additionally, we will take a look at the response using a smaller step size like we did for experiment 1 and 2, the average of the response is shown in figure 3.34 and the response is shown in figure 3.35.

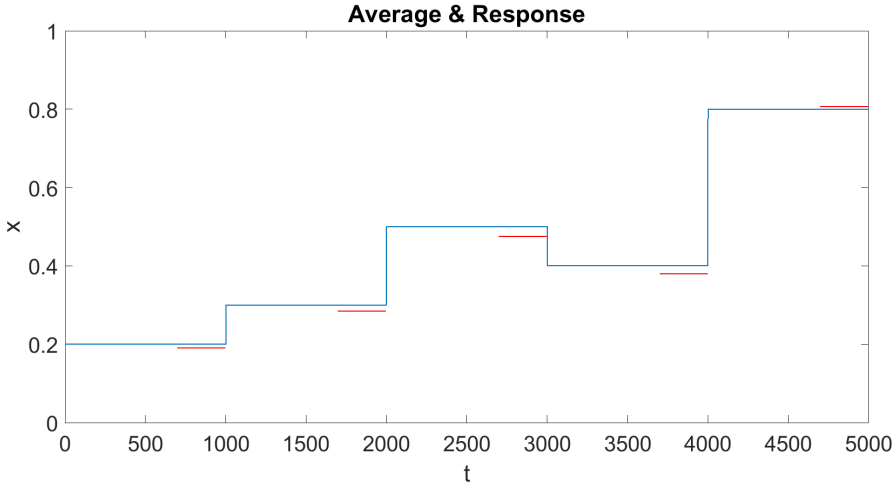


Figure 3.34: Average of the response (red) and the step (blue) for reduced step size

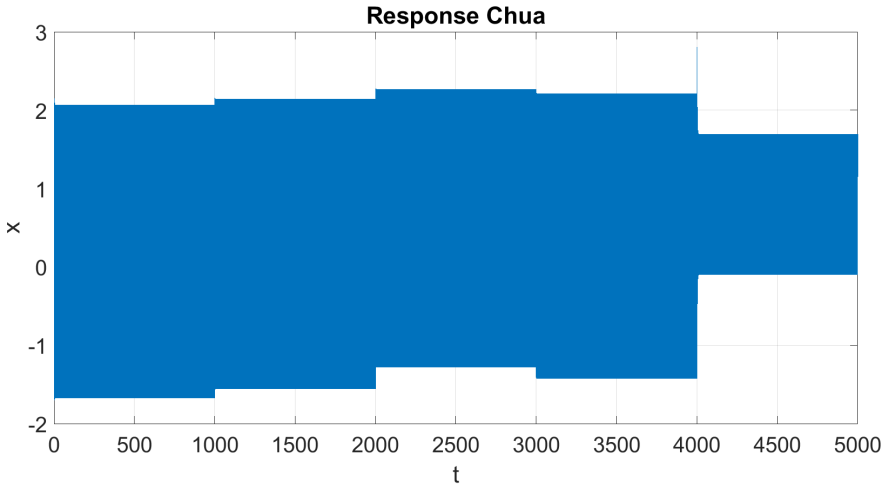


Figure 3.35: Response of the I-controlled Chua circuit for reduced step size

We can conclude with that the integral controller's parameter values are good for any value, as long as they are chaotic and the step size are big enough. Similar to the results from the experiments done on the Lorenz system we can say that the I-controller is good, however there might be room to improve the controller for smaller step sizes.

3.4 Rössler

The last chaotic system we will take a look at is the Rössler system, we will validate the programs and model from Matlab and Simulink respectively. We are going to use the coefficients from table 3.6, to find the response and attractor.

Coefficients	**Experiments**		
	1	2	3
a	0.2	0.1	0.3
b	0.2	0.1	0.3
c	9	14	18

Table 3.6: Table for coefficients used during experiments Lorenz

We have like we did for both the Lorenz system and Chua circuit made bifurcation diagrams. However, we do not only have one diagram we have two, one for c in figure 3.36 and one for b in figure 3.37.

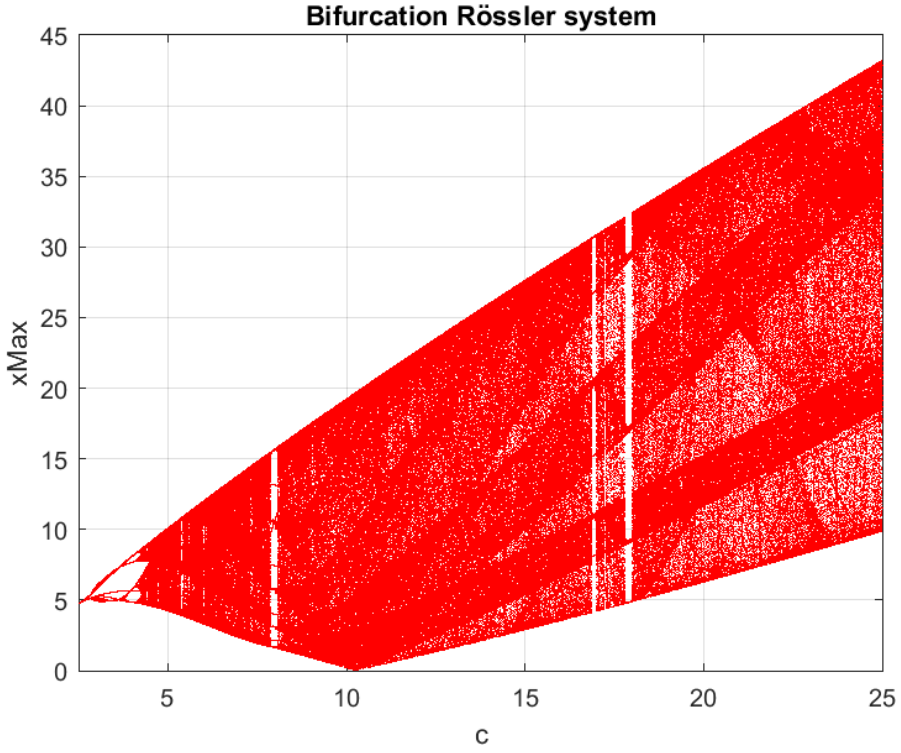


Figure 3.36: Bifurcation diagram for Rössler system with varying c

We start by taking a look at the bifurcation diagram with a varying c, figure 3.36. By observing the figure we can assume that the system starts stable but the more c increase the more chaotic the system becomes.

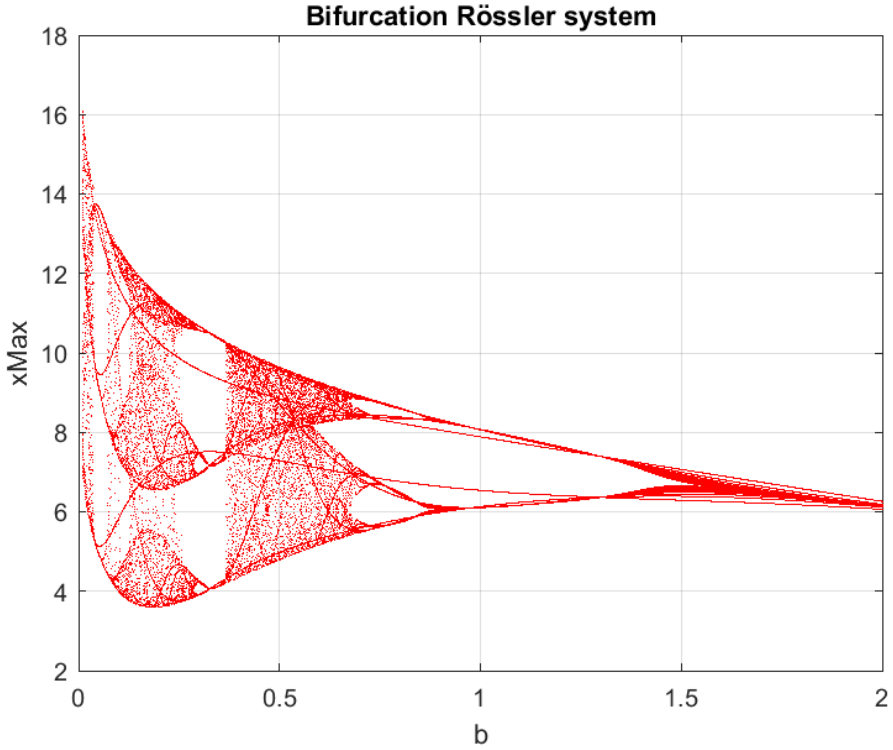
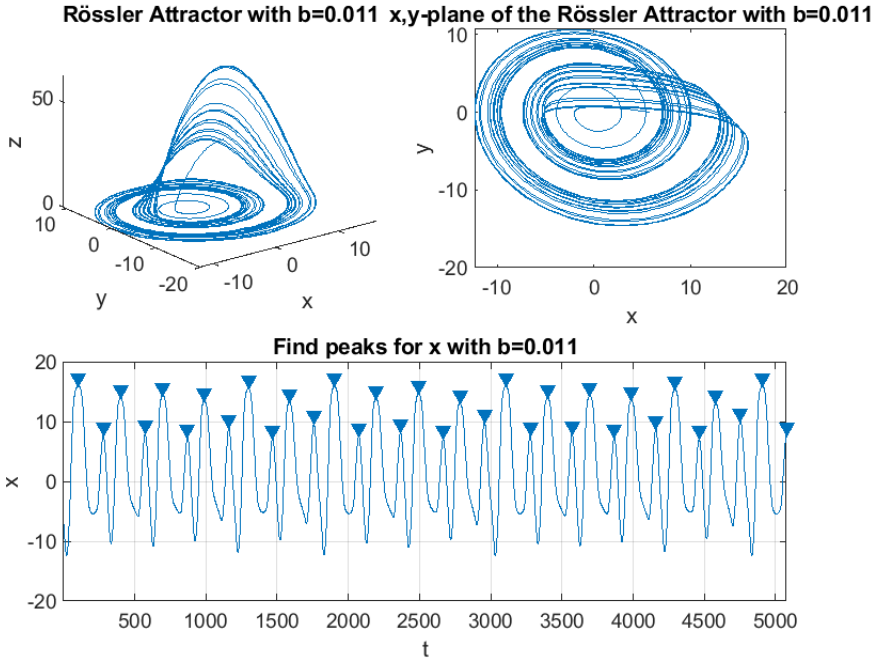
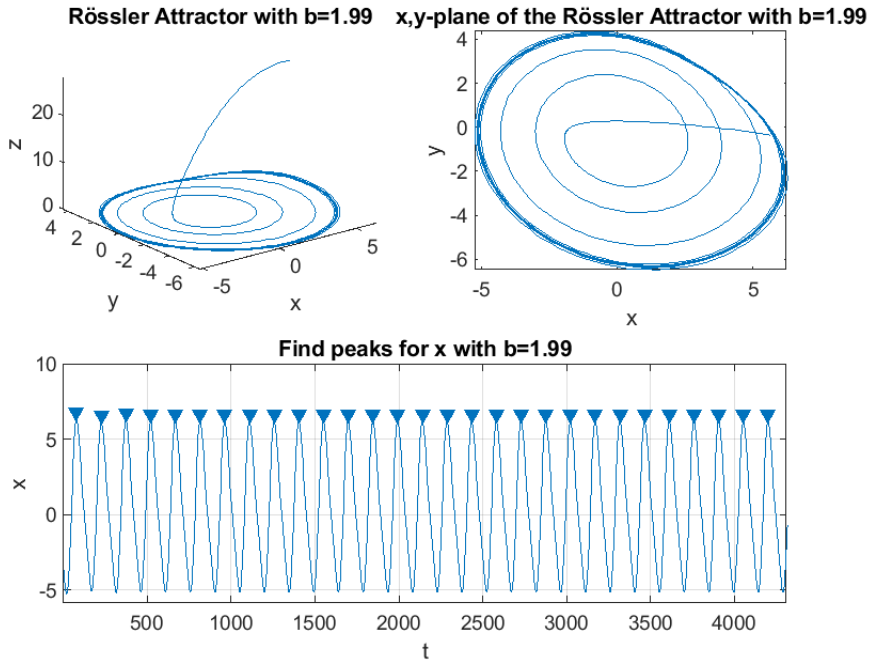


Figure 3.37: Bifurcation diagram for Rössler system with varying b

The bifurcation diagram for varying b, figure 3.37, works the opposite of c, meaning that the bigger b get the less chaotic the response will become. This can be proven by looking at figure 3.38, where we have the response of the system where we have a small b, and the response where we have a bigger b.



(a) An example of results of simulation and use of $findpeaks$ with $b = 0.011$



(b) An example of results of simulation and use of $findpeaks$ with $b = 1.99$

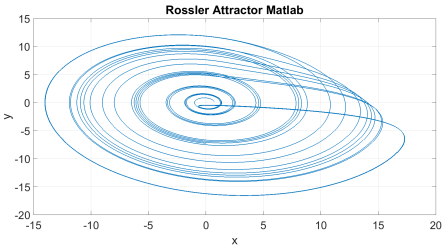
Figure 3.38: Figure (a) shows the use a chaotic response from the Rössler. Figure (b) shows a stable response.

This means we will choose coefficient value that results in chaotic behavior.

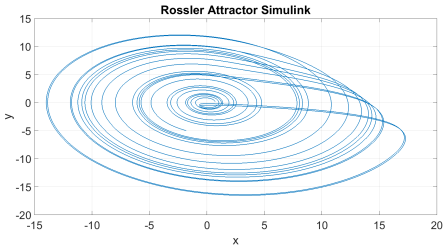
3.4.1 Model validation

Experiment 1

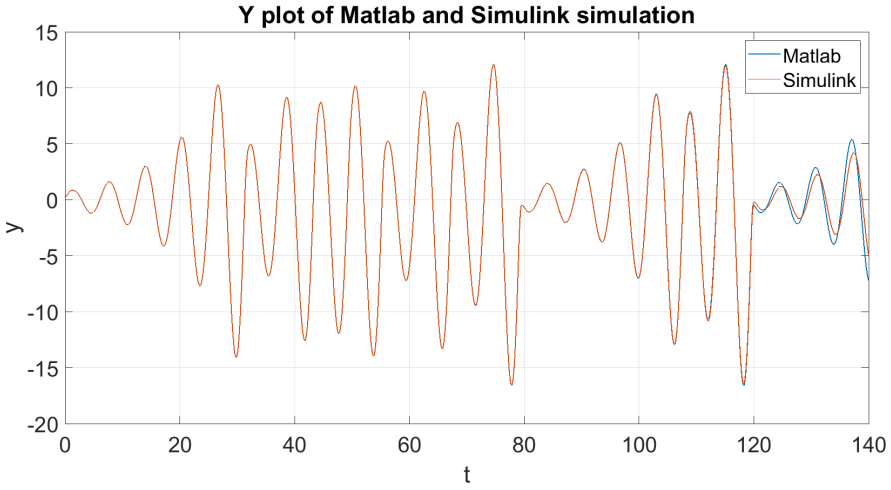
The first experiment for the Rössler system is done by using the coefficients from table 3.6, this gives us $a = b = 0.2$ and $c = 9$. We will use the Matlab script in listing 2.3 with the given coefficients, to obtain the Rössler attractor and response. Then we will use the same coefficients during a simulation of the Simulink model, to obtain an attractor and response. After finding the attractors and the response of both simulations, we will compare them to see if the Simulink model is a good representation of the Rössler system, the results are shown in figure 3.39.



(a) Rössler attractor, Matlab



(b) Rössler attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

Figure 3.39: (a) and (b) shows the Rössler attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(0.7, 0.2, 0)$

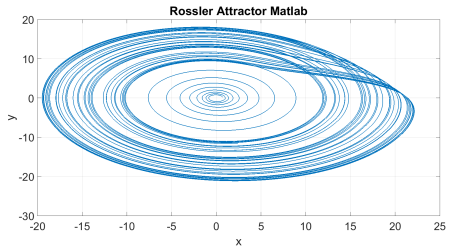
Observing figure 3.39a and figure 3.39b shows us that we have managed to create the Rössler attractor. We can also see that there is no big difference between the two attractors.

Figure 3.39c show us that the response of the Rössler system starts as the same value and that it stays like that for a long time before divergence. The time before the divergence is longer than it where for both the Lorenz system and the Chua circuit, which could ultimately mean that the representation is better. We conclude with that the Matlab and Simulink models are a good representation of each other.

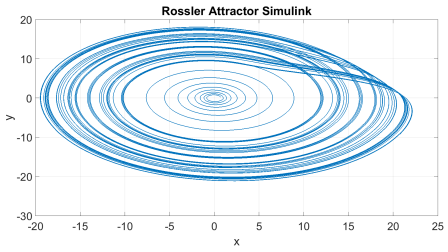
Experiment 2

We will now move on to experiment 2 where we will use the next batch of coefficients, thus we get $a = b = 0.1$ and $c = 14$. Then perform a validation

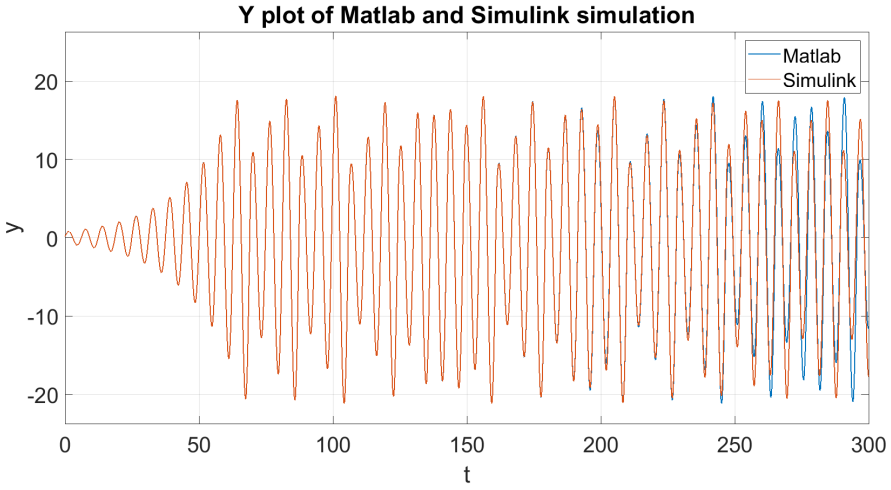
of the Matlab and Simulink model, and find one attractor for each of the models and the response for the model, and use them to make a conclusion of whether or not the model are good representations. The attractors and the response are shown in figure 3.40.



(a) Rössler attractor, Matlab



(b) Rössler attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

Figure 3.40: (a) and (b) shows the Rössler attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(0.7, 0.2, 0)$

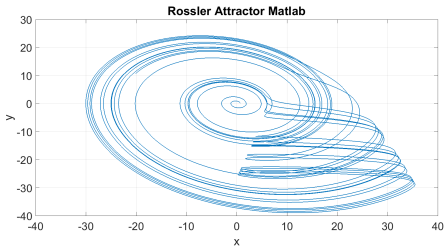
Starting by observing figure 3.40a and figure 3.40b, we see that the attractors we have is the Rössler attractor. Additionally, we see that there is no big differences or deviation in the shape of the attractor.

Figure 3.40c shows us the response from both the Matlab program and the Simulink model. By observing the graph we see that the response start with the same value, initial condition, and keep close to the same value for a time. After a while the signals diverge, likely by the same cause as for the Lorenz system and Chua circuit, and it comes down to difference in the solver. However, we can still conclude that the use of coefficients $a = b = 0.1$

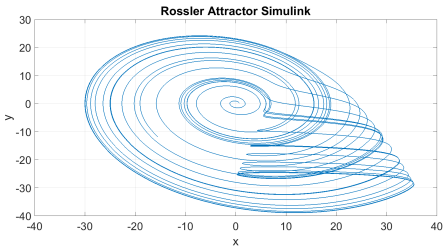
and $c = 14$ gives us a good approximation of each other.

Experiment 3

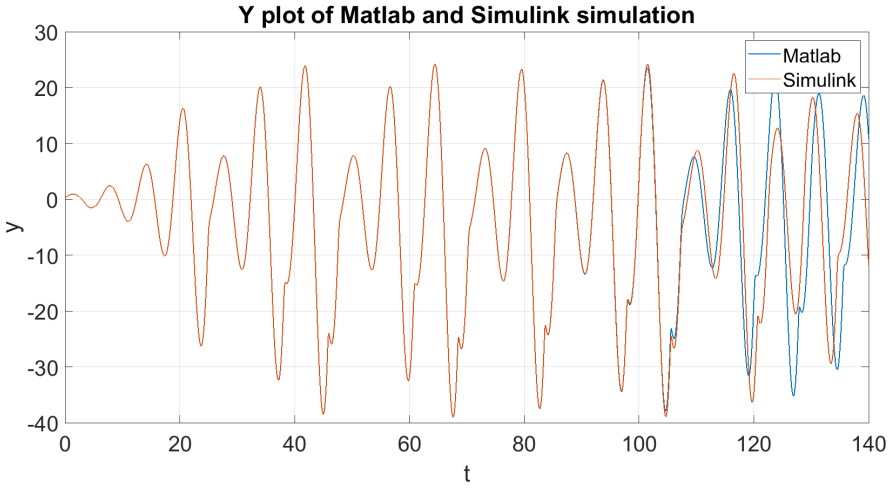
In the last of our experiments for validation of the Rössler system we use the next coefficient values from tale 3.6, $a = b = 0.3$ and $c = 18$. Using these value we will ones again take a look at attractors and the response of the Matlab program and the Simulink model, shown in figure 3.41.



(a) Rössler attractor, Matlab



(b) Rössler attractor, Simulink



(c) Plot of y for both Matlab (blue) and Simulink (red)

Figure 3.41: (a) and (b) shows the Rössler attractor for Matlab and Simulink respectively. (c) shows the response for both Matlab (blue) and Simulink (red). Staring at $(0.7, 0.2, 0)$

We start by taking a look at the attractors, figure 3.41a and figure 3.41b, we can although the shape is different from the Rössler attractors from experiment 1 and 2, we do still have the Rössler attractor. Even if they are different from experiment 1 and 2 we can see that they are close to the same attractor.

Moving on to the response, figure 3.41c, we can see that they have the same start before deviating after a time. We can also see that the divergence start earlier than in experiment 1 and 2, which is likely caused by the values of a and b. However, we can conclude that the model are good representation.

By looking at the validation for each of the models we can say that the results are satisfying, and that we can use the model later during testing of the integral controller.

3.4.2 Controller

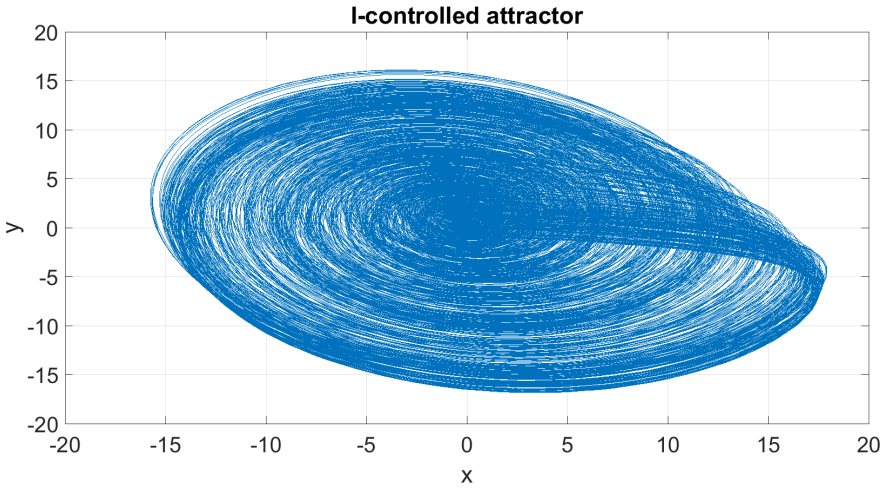
This part will contain experiments related to the implementation, testing and results of the effect of the I-controller on the Rössler system. The coefficients are shown in table 3.7.

Coefficients	**Experiments**		
	1	2	3
D	-0.2	-0.1	-0.3
E	45	140	60
F	-9	-14	-18
ζ	0.5	0.5	0.5
τ_2	255	$1.5 \cdot 10^3$	214
τ_1	5	10	3.33
K_i	$1.4 \cdot 10^{-4}$	$6.1 \cdot 10^{-6}$	$3.91 \cdot 10^{-4}$

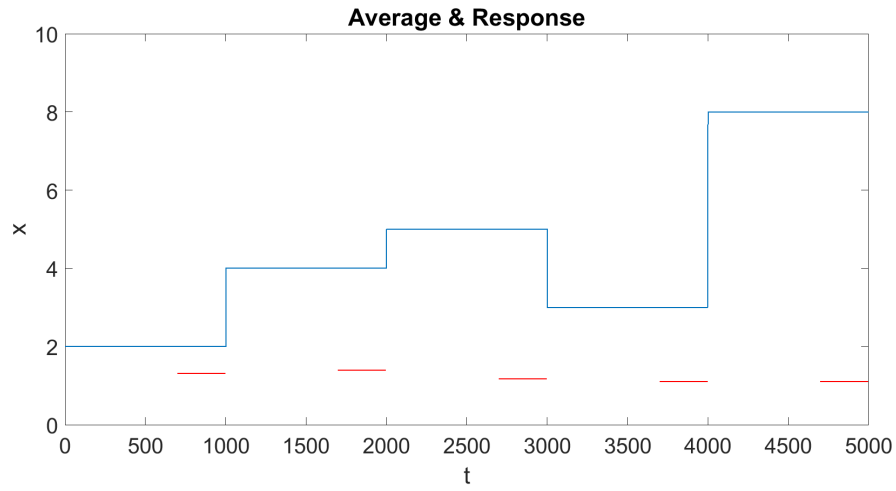
Table 3.7: Table for coefficients used to calculate the integral gain

Experiment 1

We start the experiments with using the K_i value from table 3.7, and then we check the attractor we get and the response, to see if the answers satisfy the condition we made back in chapter 1. To check this we place both the attractor and the response in figure 3.42.



(a) Attractor for I-controlled Rössler system



(b) The average of the response in steady state (red) and step (blue)

Figure 3.42: Figure (a) Shows the attractor of the I-controlled Rössler system. Figure (b) shows the the average of the response in steady state (red) and the step (blue)

Contrary to the attractor we got during the experiment for the Lorenz system and the Chua circuit, the attractor we got, figure 3.42a, do resemble the Rössler attractor. This could if we look at the attractor we got from the experiments for the Lorenz system and Chua circuit be a red flag, but that does not have to be the case as our system could still give us a respectable response. To really see if our system has a good controller we have to take a look at the response.

So we will now move on to the response, figure 3.42b, and the result is

not good. We expected the average of the response, in red, to be equal to the steps, blue. This is not the case and would mean that our controller probably is to small to have an actual impact on the system. We have to unfortunately say that we have not gotten the desired result as the deviation is to big. The response is shown in figure 3.43.

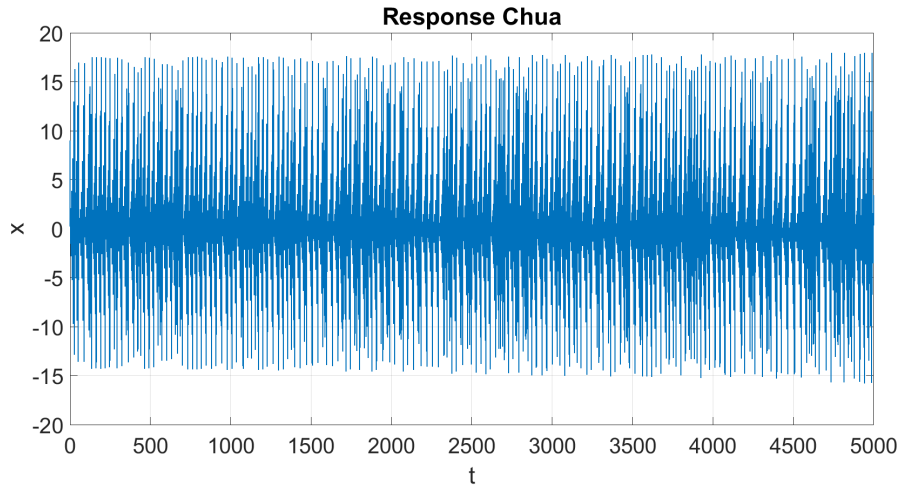
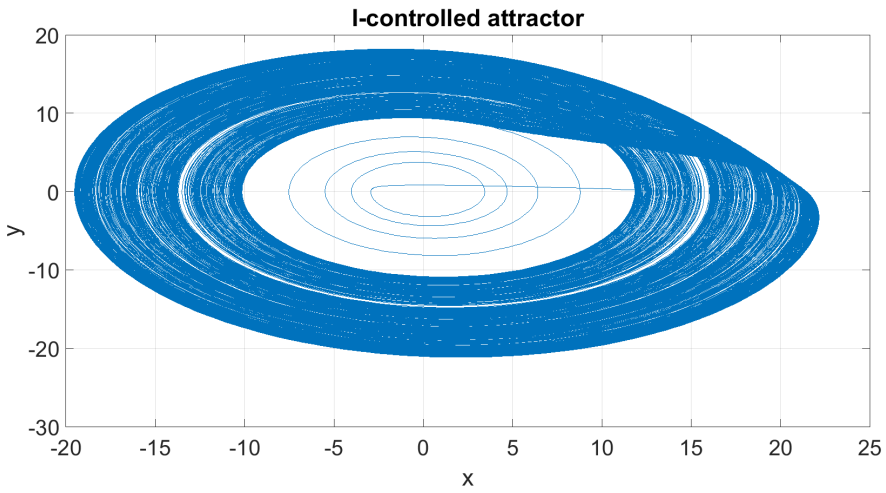


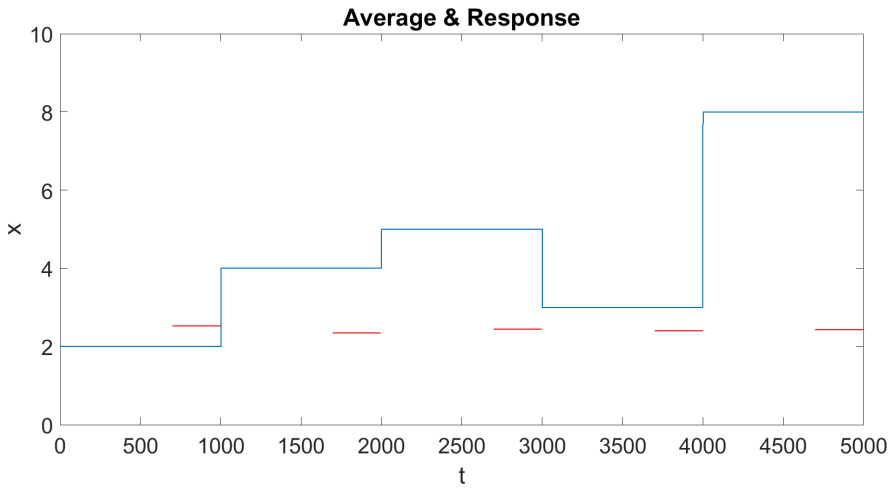
Figure 3.43: The response of the I-controllerd Rössler system

Experiment 2

After the terrible results from experiment 1 we move on to experiment 2, using the next batch of value from table 3.7.



(a) Attractor for I-controlled Rössler system



(b) The average of the response in steady state (red) and step (blue)

Figure 3.44: Figure (a) Shows the attractor of the I-controlled Rössler system. Figure (b) shows the the average of the response in steady state (red) and the step (blue)

We have ones again an attractor, figure 3.44a, that looks like the actual Rössler attractor, as mentioned this does not necessarily mean that our actual regulation is bad. So we have to like in experiment 1 take a look at the actual response.

Moving on to the response, figure 3.44b, and we can see that the average of the response, red, is closer to the steps than they were in experiment 1. This is however not because the regulation has become better, but because of the increase in coefficient c, that has increased the oscillations in the response,

figure 3.45. Thus we have to conclude with that the I-controller does not have the desired effect.

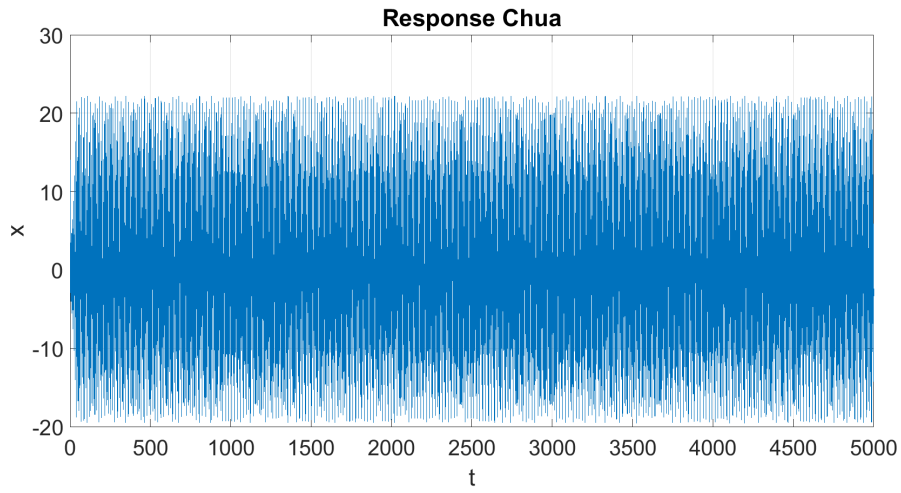
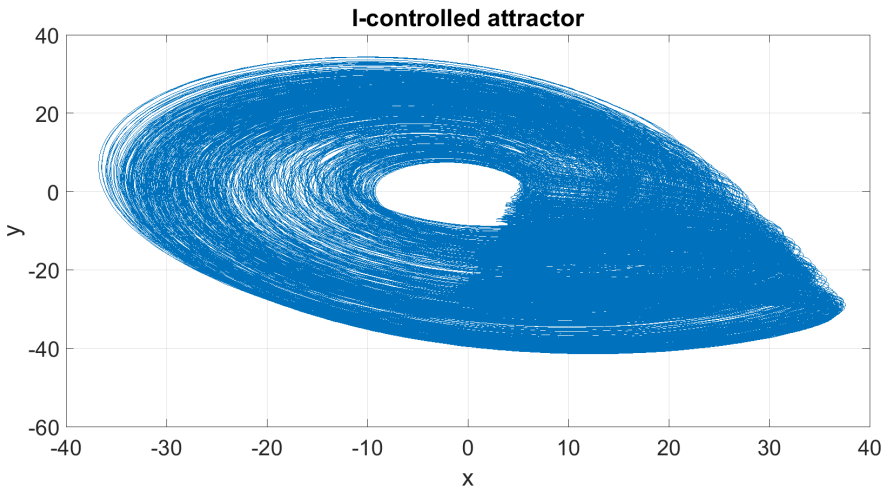


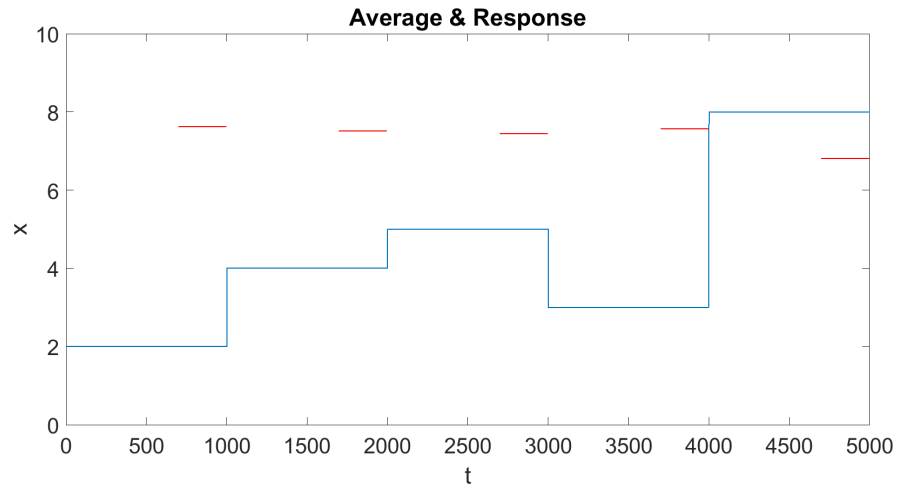
Figure 3.45: Response of the I-controlled Rössler system.

Experiment 3

The last experiment we will do in this thesis is also the third one for the controller part of the Rössler system. We will now choose the next batch of coefficients and repeat the experiment 1 and 2. Starting with the attrator and the response, figure 3.46, and see if the results are better than the last two experiments.



(a) Attractor for I-controlled Rössler system



(b) The average of the response in steady state (red) and step (blue)

Figure 3.46: Figure (a) Shows the attractor of the I-controlled Rössler system. Figure (b) shows the the average of the response in steady state (red) and the step (blue)

Taking a look at the attractor, figure 3.46a, and we see that it resemble the attractor we got in figure 3.41b, which means it still look like the Rössler attractor.

So we will quickly move on to the response in figure 3.46b. One again we can see that the response is not good, and that the only reason that the average of each step is higher is the increase in coefficients. This can be proven by taking a look at the response in figure 3.47.

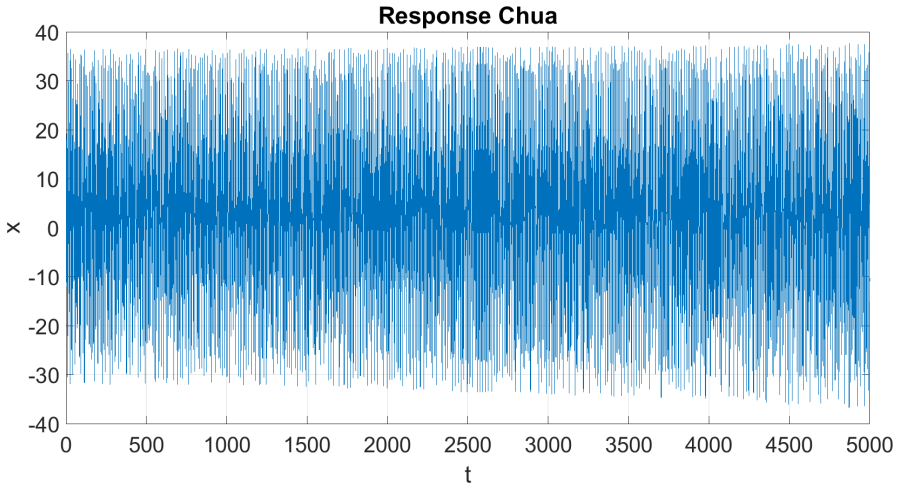


Figure 3.47: The response of the I-controlled Rössler system

To conclude the experiment on the Rössler system we can see that the results are not as desired or expected. This is most likely because of the small value of K_i. Meaning that the calculations are most likely not correct, and can be improved in the future. One change that can be done is to change the output from y to x and then find the transfer function $\frac{\Delta x}{\Delta u}$, then do similar assumptions as we did for the Lorenz system and Chua circuit.

4. Discussion

This chapter will give a review on the results from chapter 3. Achieved results, improvements. Suggestions for further works will also be presented.

4.1 Model validation

This section takes a closer look at the result from the experiments done in chapter 3, and mainly the model validation part.

Lorenz

By observing the results from experiment 1, 2 and 3 we can say that the Simulink model is a good representation of the Matlab program and vice versa. Even if the divergence happen after a short time, this is however caused by a small difference in the solver.

Lorenz as an electrical circuit

The electrical circuit is a good representation of the Lorenz system. Not only based of the results given in this thesis but also from the article made by Kevin Cuomo et al.[2]. So even if the response is not as expected we can see based of the attractors that it is still a chaotic Lorenz system.

Chua

The results have similar response and the attractors are similar so we can safely say that the model are a good representation. However, like the case was for the Lorenz system the response diverges fast compared to what we thought would be the case.

Rössler

The models based of the Rössler system are really good and uses longer time before it starts to diverge. This means that the Rössler models are probably the best representation.

4.2 I-controller

This chapter will review the experiments to check the effect of the I-controller has on the chaotic systems.

Lorenz

The results are surprising, since we actually expected the step response to oscillate. However since we got a stable response we can conclude with that we have managed to get results that satisfy our initial task of this thesis. Additionally we can add a disturbance to the system that can cause an oscillating step response.

Chua

Very similar results to the Lorenz system, except with the fact that we get an oscillating step response if we a small step size. We will therefore conclude with that the results are satisfying.

Rössler

The result we get from using an I-controller are not good. The cause of this is most likely because of the calculations done to find K_i.

4.3 Improvements

One of the improvements that can be done is on the Rössler system, and this will mainly be to got through the calculation and find the transfer function $\frac{\Delta x}{\Delta u}$. Changing the transfer function might give us better results than we actually got in this thesis.

4.4 Further work

Other than the improvements to the Rössler system, there is the opportunity to upgrade/add an integral controller to the Simulink model of the Lorenz system as an electrical circuit. Then make a real life model of the system using the schematic from appendix C.1, and then using the results from the Simulink model and the electrical circuit to compare the two model to each other.

5. References

[1] Edgar Seborg et. al. *Process Dynamics and Control*. 2016, pp. 199–229.

[2] Kevin M. Cuomo and Alan V. Oppenheim. "Circuit Implementation of Synchronized Chaos with Applications to Communications". In: (1993).

[3] Alan V. Oppenheim Kevin M. Cuomo and Steven H. Strogatz. "Synchronization of Lorenz-Based Chaotic Circuits with Applications to Communications". In: (1993).

[4] V. Siderskiy. *Matlab simulation of Chua's circuit*. URL: `https://www.chuacircuits.com/matlabsim.php`. (accessed: 11.03.2022).

[5] Steven Strogatz. *Nonlinear dynamics and chaos*. 1994, pp. 301–347.

[6] Wikipedia. *Chua System*. URL: `https://en.wikipedia.org/wiki/Chua%5C%27s_circuit`. (accessed: 26.04.2022).

[7] Wikipedia. *Lorenz System*. URL: `https://en.wikipedia.org/wiki/Lorenz_system`. (accessed: 26.04.2022).

[8] Wikipedia. *Rossler System*. URL: `https://en.wikipedia.org/wiki/R%5C%C3%5C%B6ssler_attractor`. (accessed: 26.04.2022).

Appendices

A. Matlab

B. Simulink

B.1 LorenzBifurcationModel.slx

B.2 ElecLorenz.slx

B.3 LorenzScaled.slx

B.4 LorenzSystemSimulink.slx

B.5 ChuaBifurcationModel.slx

B.6 ChuaSystem.slx

B.7 RosslerBifurcationModel.slx

B.8 Rossler.slx

C. Electrical Schematics

C.1 LorenzCircuitWithIcontroller

C.2 ParameterValues

C.3 Poster