Problem set 1, Task 3
A route to chaos
Computational Biology
FFR110/FIM740

Mattias Berg, Anita Ullrich

February 11, 2021

a.)

Figure 1 shows the bifurcation diagram of the system. We can see that the dynamics behave at first in an equilibrium but then there occur clearly period-doubling several times, e.g. at $R \approx 7.5, 13, \ldots$. We observe also that the dynamics are behaving chaotic in between where period-doubling occurs infinitely often. But surprisingly, we return from chaotic dynamics to a 3-point cycle at $R \approx 22.5$. But after that for $R \approx 25$ we observe chaotic behavior again for all following values of R that we considered.

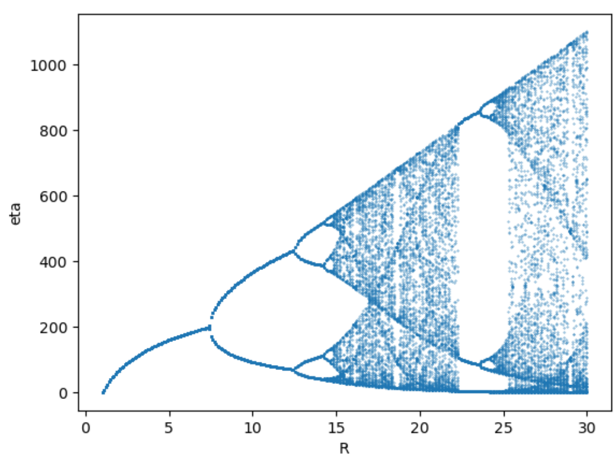

Figure 1: Bifurcation diagram of $\eta_{\tau+1} = R\eta_\tau e^{-\alpha\eta_\tau}$

b.)

Figure 2 shows the population dynamics for different values of R to visualize equilibrium, a 2-, 3- and 4-point cycle. We can here see that for larger τ values, when the function have stabilised it self, that the function will jump between different values. For example R = 23, we can see that it jumps between 3 different points, which is expected, as R = 23 have 3- points cycle according to Figure 1.

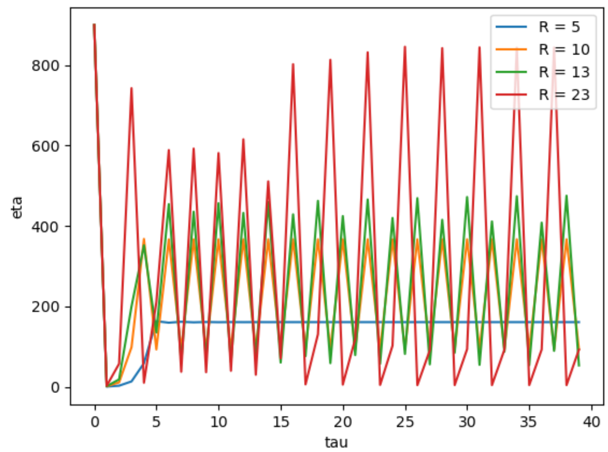

Figure 2: Population dynamics for 4 different R values. R=5 have a stable equilibrium, R=10 have a 2-point cycle, R=13 have a 4-point cycle and R=23 have a 3-point cycle

c.)

In figure 3 we visualized the bifurcation diagram and highlighted the interesting bifurcation points. From this we see that $R_1 \approx 7.38$ and $R_2 \approx 12.50$.

d.)

When zooming in around R_∞ it was found that a pattern repeats it self in sets of 4, one of these sets is plotted in Figure 4. From this it can be see that R_∞ happens between R 14.766 and 14.767. The R_∞ point is found by looking at Figure 4, and determining when we do not have clear "lines" anymore, meaning that we are having an infinite amount of solutions, instead of a point cycle.

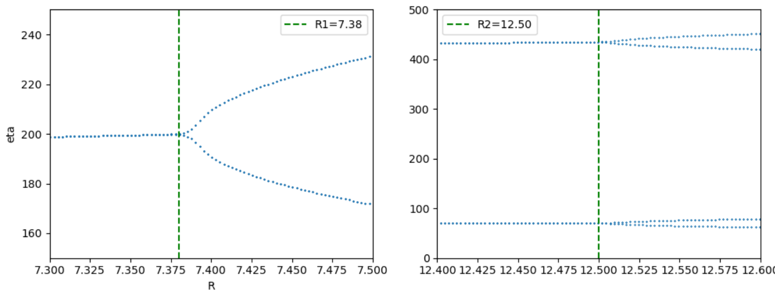

Figure 3: R values of staring point of 2-point cycle (R1) and starting point of the 4-point cycle (R2)

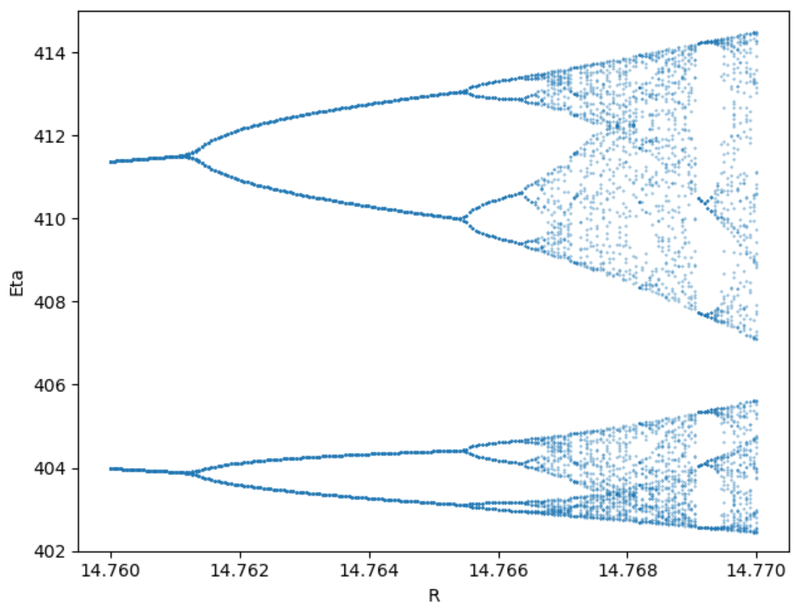

Figure 4: Zoomed in plot. In this we see R_∞ starting between R 14.766 and 14.767

Appendix

Code task 3

```
#Course : FFR110/FIM740 Computational Biology
#Problem : Problem set 1, Task 3. A route to chaos
#Code :  Python 3.8.5

import numpy as np
import matplotlib.pyplot as plt
import math

def equation(R,eta):
    alpha=0.01
    return R*eta*math.exp(-alpha*eta)

# The bifurcation_diagram code is based on https://github.com/Alain1405/bifurcat
# Create the bifurcation diagram
def bifurcation_diagram(seed, t_skip, t_iter, step=0.1, R_min=0, R_max=30):

    # Array of R values, the x axis of the bifurcation plot
    RList = []
    # Array of eta (eta_{tau}) values, the y axis of the bifurcation plot
    etaList = []

    # Create the R values to loop. For each Rr value we will plot t_iter points
    R_range = np.linspace(R_min, R_max, int((R_max-R_min)/step))

    for R in R_range:
        eta = seed
        # For each r, iterate the logistic function and collect datapoint if n_s
        for i in range(t_iter+t_skip+1):
            if i >= t_skip:
                RList.append(R)
                etaList.append(eta)

            eta = equation(R,eta)

    return RList, etaList

#Question 3a
if False:
    RList, etaList = bifurcation_diagram(900, 200, 100, step=0.1, R_min=1, R_max

    plt.scatter(RList, etaList, s=0.1)
    plt.xlabel('R')
    plt.ylabel('eta')
    plt.grid()
```

```
    plt.show()

#Question 3b
# Plot function for different R values.
if False:
    def plot_equation(seed, R=[30], t_end=40):

        # Array of eta (eta_{tau}) values, the y axis of the bifurcation plot
        etaList = []

        for r in R:
            eta = seed
            etaListTmp=[]
            print(r)
            for t in range(t_end):

                etaListTmp.append(eta)

                eta = equation(r,eta)

            etaList.append(etaListTmp)
        # Plot the data
        for r in range(len(R)):
            plt.plot(etaList[r], label='R = ' + str(R[r]))
        plt.xlabel('tau')
        plt.ylabel('eta')
        plt.legend()
        plt.show()

#Question 3c
if True:
    RList, etaList = bifurcation_diagram(900, 2000, 25, step=0.0025, R_min=6, R_

    fig, (axs1,axs2) = plt.subplots(1,2)
    axs1.scatter(RList, etaList, s=0.05)
    line1=axs1.axvline(x=7.38, c='g', linestyle='dashed')

    axs2.scatter(RList, etaList, s=0.05)
    line2=axs2.axvline(x=12.50, c='g', linestyle='dashed')

    axs1.set_ylim([150, 250])
    axs2.set_ylim([0, 500])

    axs1.set_xlim([7.3, 7.5])
    axs2.set_xlim([12.4, 12.6])
```

```
    axs1.set(xlabel='R', ylabel='eta')

    axs1.legend([line1], ['R1=7.38'])
    axs2.legend([line2], ['R2=12.50'])

    plt.show()

#Question 3d
#Show all the bifurcation zooms around r_inf.
if False:
    RList, etaList = bifurcation_diagram(900, 3000, 500, step=0.00005, R_min=14.

    fig, axs = plt.subplots(2,3)
    axs[0,0].scatter(RList, etaList, s=0.1)
    axs[0,1].scatter(RList, etaList, s=0.1)
    axs[0,2].scatter(RList, etaList, s=0.1)
    axs[1,0].scatter(RList, etaList, s=0.1)
    axs[1,1].scatter(RList, etaList, s=0.1)
    axs[1,2].scatter(RList, etaList, s=0.1)

    axs[0,0].set_ylim([505, 545])
    axs[0,1].set_ylim([400, 431])
    axs[0,2].set_ylim([364, 377])
    axs[1,0].set_ylim([128, 142])
    axs[1,1].set_ylim([85, 107.5])
    axs[1,2].set_ylim([34, 47])

    for ax in axs.flat:
        ax.set(xlabel='R', ylabel='eta')

    plt.subplots_adjust(left=0.125,
                        bottom=0.1,
                        right=0.9,
                        top=0.9,
                        wspace=0.2,
                        hspace=0.1)
    plt.show()

#Question 3d
#Show one of the bifurcation zooms around r_inf
if False:
    RList, etaList = bifurcation_diagram(900, 3000, 1000, step=0.00005, R_min=14
    plt.scatter(RList, etaList, s=0.1)
    plt.ylim([402, 415])
    plt.xlabel('R')
    plt.ylabel('Eta')
```

```
plt.show()
```