Odwzorowanie logistyczne

Matejko Marek, Mazur Krzysztof, Paszkot Dawid

29 listopada 2021

Spis treści

1 Wprowadzenie

Naszym celem było zapoznanie się odwzorowanie logistyczny. Naszkicowanie wykresów wartości populacji funkcji x_n liczby generacji n. Określenie charakterystycznych punktów na otrzymanych wykresach. Naszkicowanie diagramu bifurkacyjnego.

Odwzorowanie logistyczne (ang. logistic map) jest to funkcja odwzorowująca przedział jednostkowy w siebie:

$$f : [0, 1] \to [0, 1]$$

dana wzorem:

$$f(x_{n+1}) = \mu x_n(1 - x_n)$$

gdy wartości parametru μ warunek:

$$0 < \mu < 4$$

Funkcja ta jest klasycznym przykładem prostego układu dynamicznego zachowującego się chaotycznie. Znajduje zastosowanie np. w badaniach dynamiki liczebności populacji.

Bifurkacja jes to zjawisko skokowej zmiany własności modelu matematycznego przy drobnej zmianie jego parametrów (np. warunków początkowych procesu albo warunków brzegowych). Szczególnie często spotykane i istotne jest to pojęcie przy rozwiązywaniu równań różniczkowych oraz badaniu fraktali (i teorii chaosu).

2 Materiały i metody

Materiały dydaktyczne

W celu zapoznania się z tematem i postawionymi przed nami zadaniami, przeczytaliśmy dostarczoną literaturę(ODE2.pdf).

Aby bardziej przybliżyć nam dany temat skorzystaliśmy również z materiałów dostępnych w internecie. Między innymi film "This equation will change how you see the world (the logistic map)" na portalu Youtube *(https://youtu.be/ovJcsL7vyrk)*, również posiłkowaliśmy się informacjami ze strony na portalu Wikipedia poświęconej językowi Python *(en.wikipedia.org/wiki/Python_(programming_language))* oraz strony na temat naszego głównego tematu, czyli odwzorowaniu logistycznemu *(pl.wikipedia.org/wiki/Odwzorowanie_logistyczne)*.

Metody

Programy pisaliśmy w języku Python w środowiskach Python(x,y) oraz Jupyter notebook. Użyliśmy bibliotek *numpy* i *matplotlib*

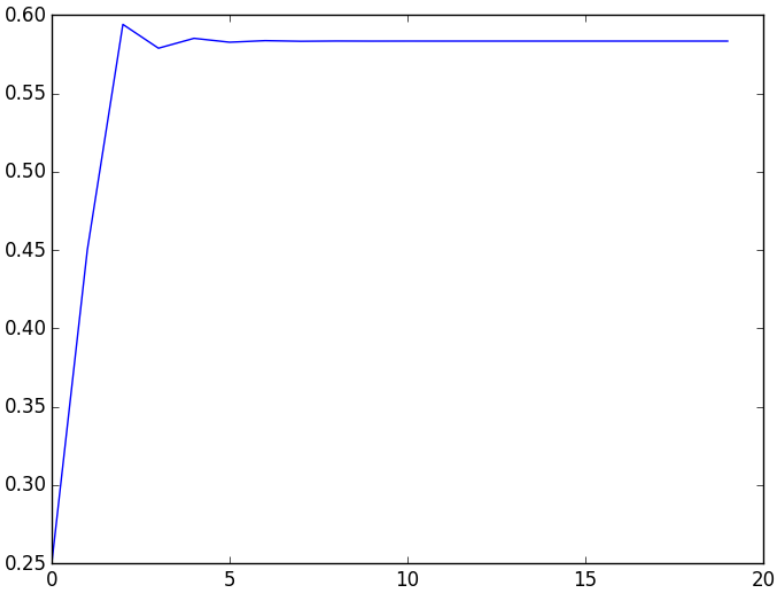
3 Rezultaty

Zobrazowaliśmy charakterystyczne zachowania populacji dla różnych wartośći μ oraz x_0. Później pokazaliśmy jak zachowują się atraktory od wartości paramatru wzrostu (tzw. diagram bifurkacyjny)

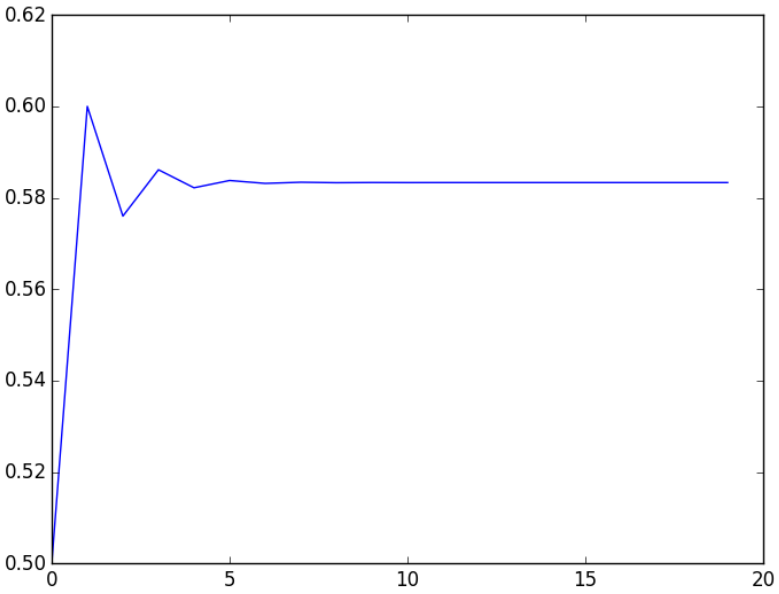
```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Sat Nov 06 12:58:47 2021
4
5  @author: user
6  """
7
8  import numpy as np
9  import matplotlib.pyplot as plt
10
11 r = 3.0
12
13 N = 1000
14 x = .75+np.zeros(N)
15
16 for n in range(N-1):
17     x[n+1] = r*x[n]*(1-x[n])
18
19 plt.plot(x,'-')
20 plt.show()
```

Rysunek 1: Za pomocą podanego kodu, zmieniając odpowiednie parametry, uzyskaliśmy poniższe wykresy

1. **Przejściowość**
   Nieregularne zachowania przed osiągnięciem stanurównowagi, które są inne dla różnych wartościpoczątkowych $\mu = 2.4$

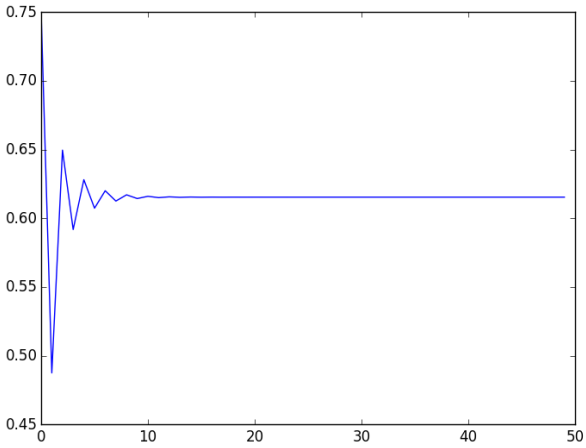


Rysunek 2: x = 0.25

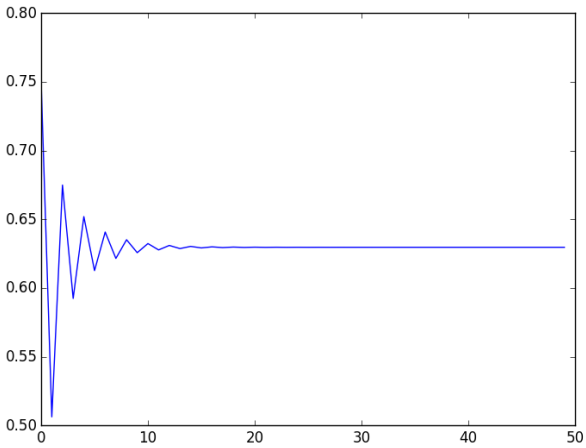


Rysunek 3: x = 0.5

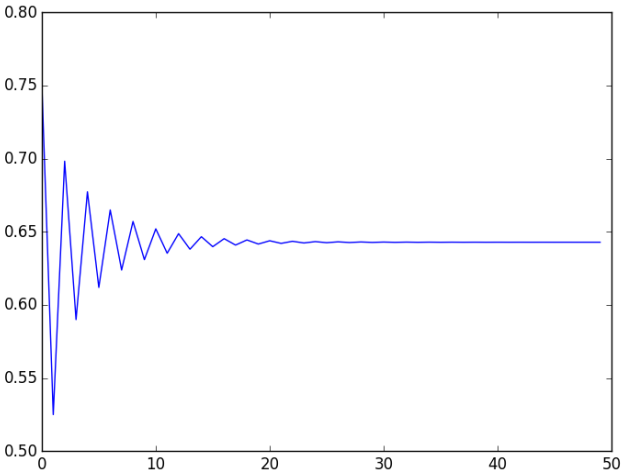
2. **Asymptoty**
   W niektórych przypadkach stan ustalony jest osiągany już po 20 pokoleniach, podczas gdy dla większych wartości μ mogą być potrzebne setki pokoleń. Te populacje w stanie równowagi są niezależne od wartości początkowych.



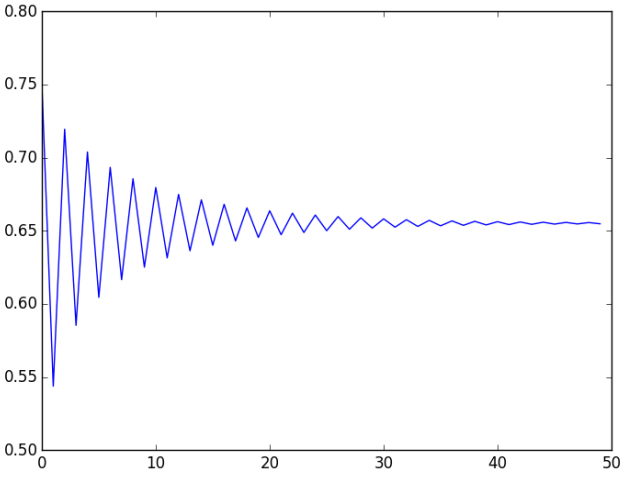
Rysunek 4: $\mu = 2.6$



Rysunek 5: $\mu = 2.7$

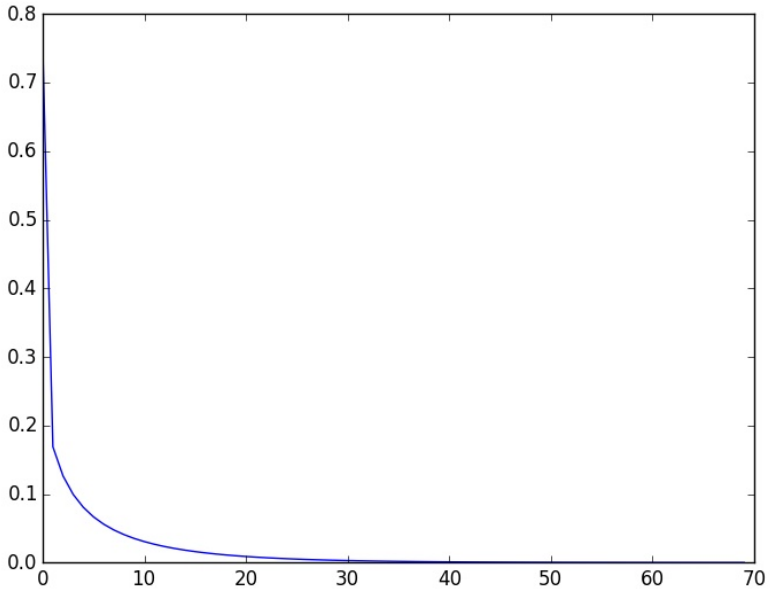


Rysunek 6: $\mu = 2.8$



Rysunek 7: $\mu = 2.9$

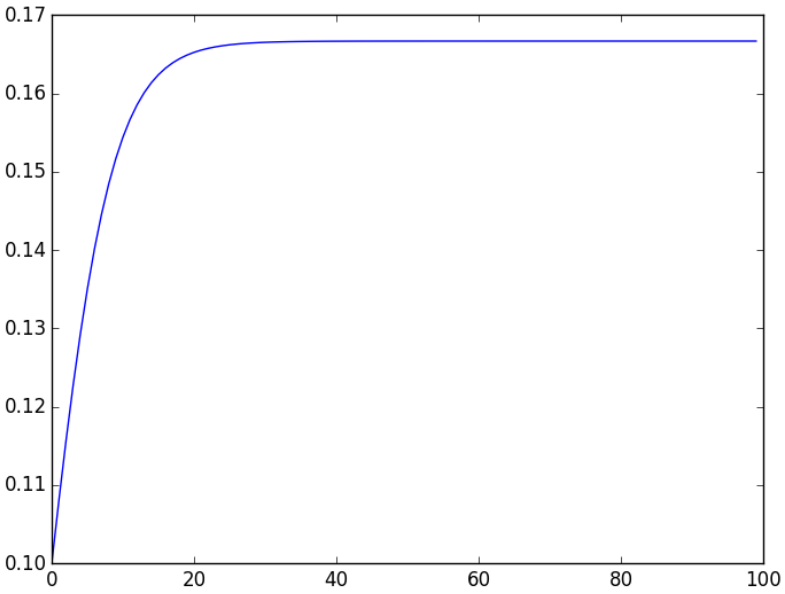
3. **Wyginięcie**
   Jeśli tempo wzrostu jest zbyt niskie , $\mu \leq 1$, populacja wymiera



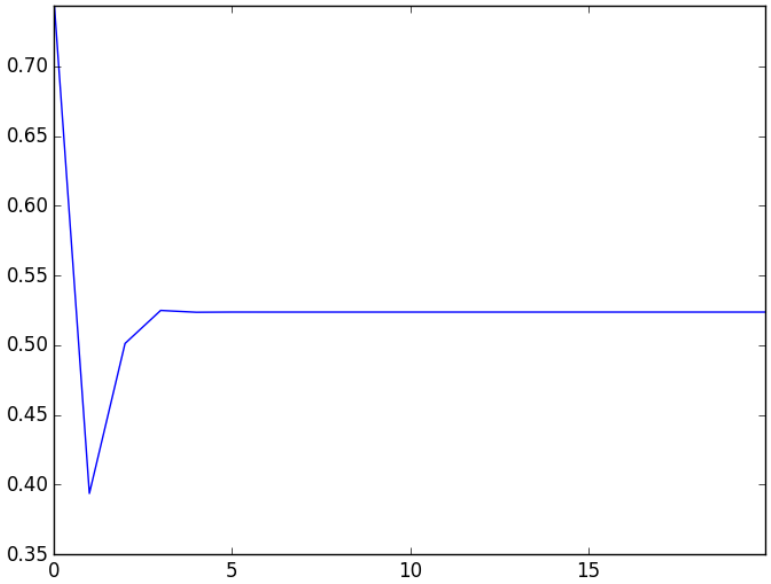
Rysunek 8: $\mu \leq 1$

4. **Stany stabilne**
   Stabilne stany pojedynczej populacji uzyskane dla $\mu < 3$ powinny zgadzać się z przewidywaniem:

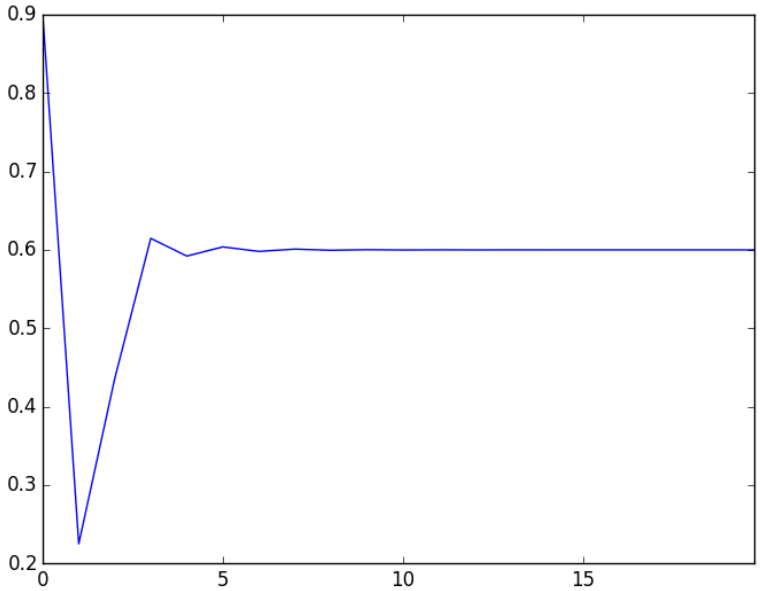
$$x_* = \frac{\mu - 1}{\mu}$$



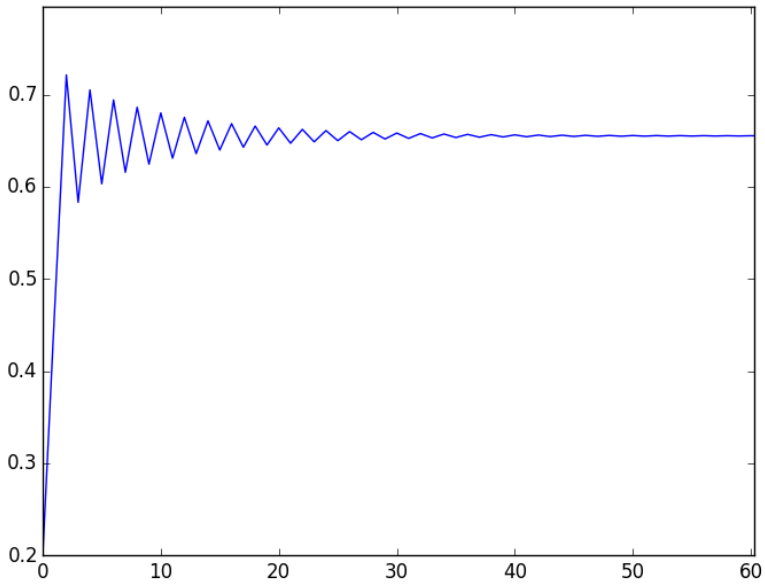
Rysunek 9: $x_* = \frac{1.2-1}{1.2}$



Rysunek 10: $x_* = \frac{2.1-1}{2.1}$

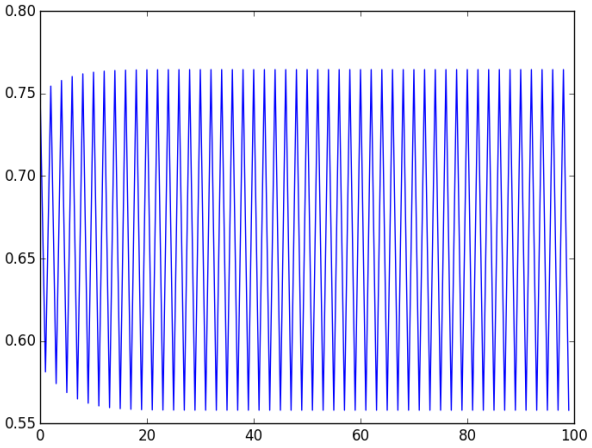


Rysunek 11: $x_* = \frac{2.5-1}{2.5}$

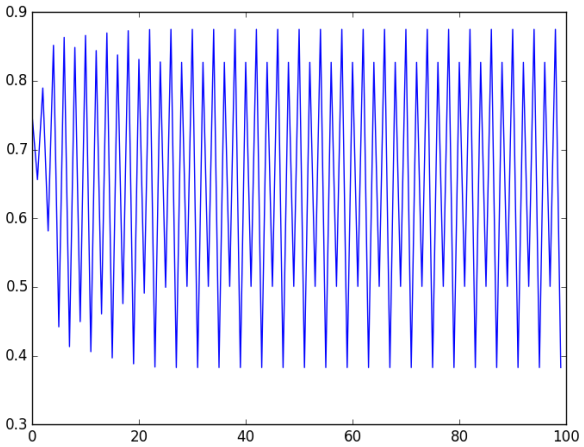


Rysunek 12: $x_* = \frac{2.9-1}{2.9}$

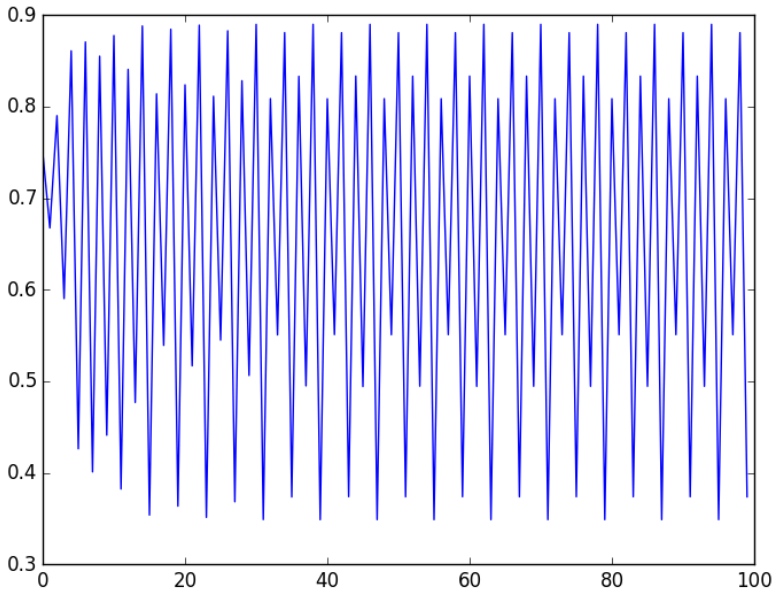
5. **Wiele cykli (eng.: Multiple cycles)**
   Badamy odwzorowanie pod kątem parametru wzrostu μ rosnącego od 3. Obserwujemy, jak system kontynuuje podwajanie okresów wraz ze wzrostem μ.



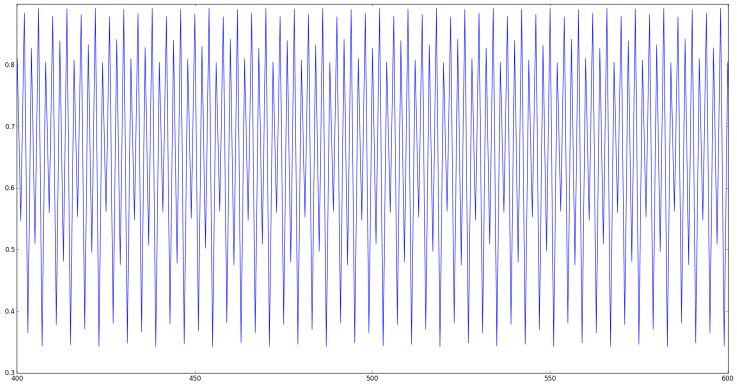
Rysunek 13: $\mu = 3,1$; dwa cykle



Rysunek 14: $\mu = 3,5$; cztery cykle

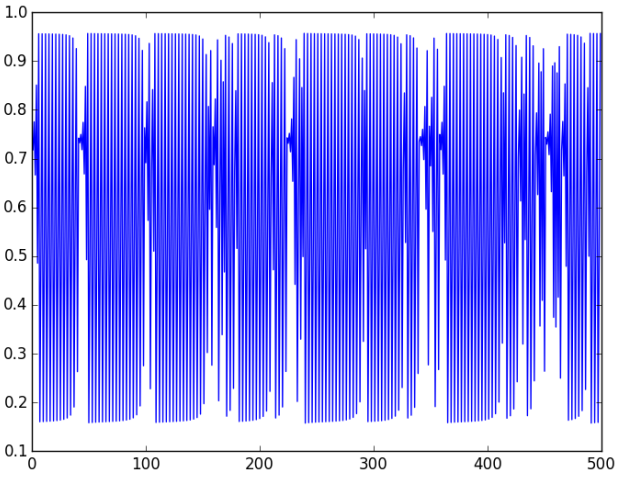


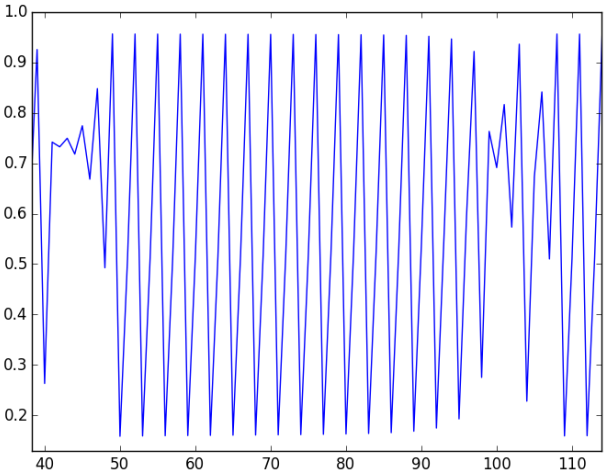
Rysunek 15: $\mu = 3,56$; osiem cykli



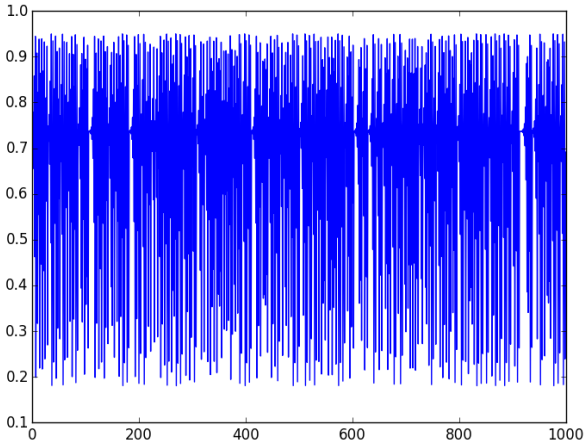
Rysunek 16: $\mu = 3,57$; szesnaście cykli

6. **Intermitencja (eng.: Intermittency)** Obserwujemy symulacje dla $3,8264 < \mu < 3,8304$. Tutaj system jest stabilny przez pewną liczbę pokoleń, a następnie skacze chaotycznie, by znów stać się stabilnym

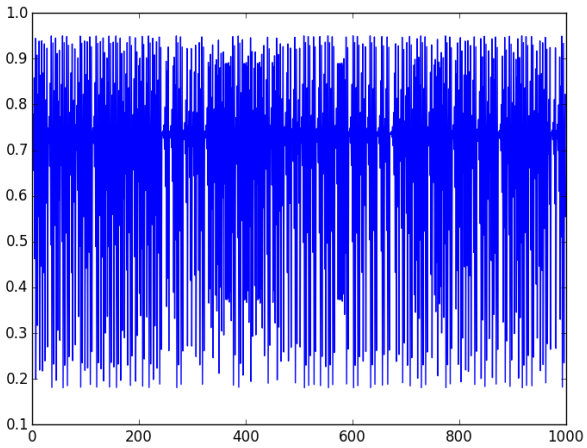




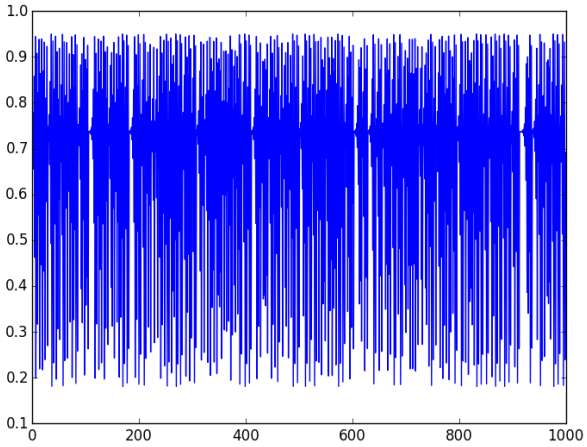
7. **Chaos**
   Zachowanie systemu w obszarze chaotycznym jest bardzo zależne od dokładnych wartości początkowych μ i x_0



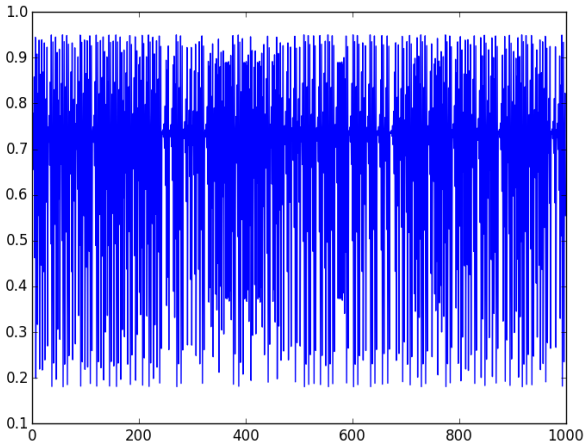
Rysunek 17: x = 0.75



Rysunek 18: $x = 0,75(1 + \epsilon)$



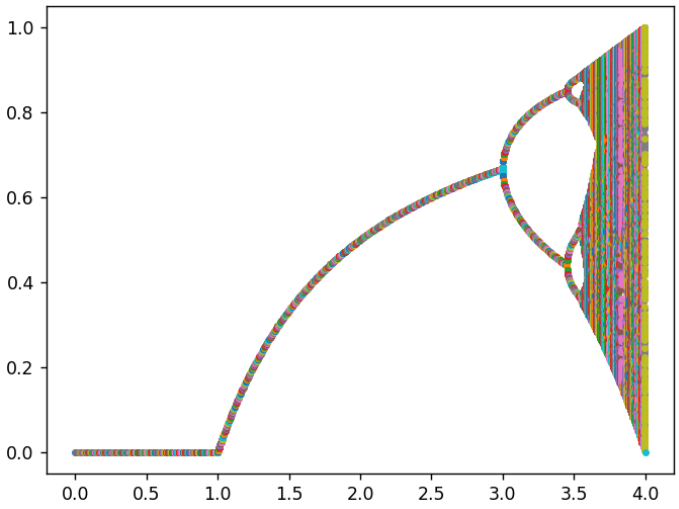
Rysunek 19: $\mu = 3,8$



Rysunek 20: $\mu = 3,8(1 - \epsilon)$

8. **Diagram bifurkacyjny**

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

%matplotlib notebook

us = np.linspace(1,4,1000)

N = 1000

x = .5+np.zeros(N)

endcap = np.arange(round(N*.8),N)



for ui in range (len(us)):

    for n in range(N-1):
        x[n+1] = us[ui]*x[n]*(1-x[n])

    k = np.unique(x[endcap])
    u = us[ui]*np.ones(len(k))
    plt.plot(u,k,'.')

plt.show()
```


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

4 Dyskusja

Udało nam się otrzymać rezultaty zgodne z przewidywaniami teoretycznymi. Wykresy wykazują odpowiednie zachowania, jak również diagram bifurkacyjny poprawnie ukazuje, znany nam z innych źródeł, chaotyczne zachowanie dla określonych wartości paramteru wzrostu.