Przedmiot: Podstawy modelowania i symulacji
TEMAT : Projekt PMIS
Imię Nazwisko Grupa : Filip Gędłek L2 MITI

Wstęp

Zmienne użyte w sprawozdaniu:
$C = 0,696$
$\delta = 0,070$
$m_1 = 6$
$m_2 = 5$
$k_1 = k_2 = 6,5$
$\frac{k}{m} = \omega^2$

Link do Githuba z kodami

0.1 Obwód RC z wymuszeniem sinusoidalnym

- Zapisz poprawnie równanie na ładunek w obwodzie.

$$q(t) = A\frac{\alpha \sin(\omega t) - \omega \cos(\omega t)}{\alpha^2 + \omega^2}$$

- Zaklasyfikuj typ równania.

Równanie nieliniowe jednorodne.

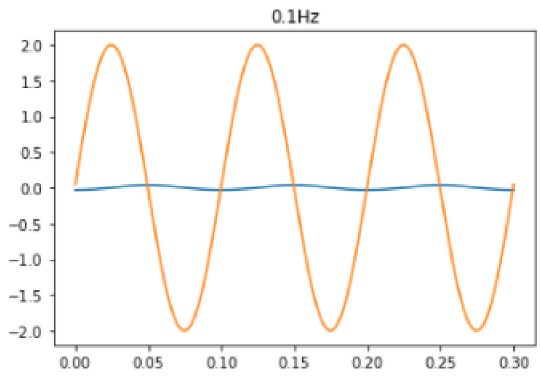
- Po jakim czasie w obwodzie bez wymuszenia napięcie spadnie do 0.1 napięcia początkowego?

$$\frac{1}{10} = e^{-\alpha * t}$$
$$\alpha = \frac{-1}{RC}$$
$$R = 1$$
$$C = 0.696$$
$$\alpha = \frac{-1}{0.696}$$
$$\frac{1}{10} = e^{-\frac{1}{0.696} * t}$$

Po rozwiązaniu równania otrzymujemy przybliżoną odpowiedź:
t = 1.602599

- Jakie jest maksymalne napięcie na oporniku i kondensatorze po długim czasie (brak odpowiedzi swobodnej) w przypadku wymuszenia źródłem napięciowym sinusoidalnym o amplitudzie 2V i częstościach odpowiednio 0.1,1,10 Hz

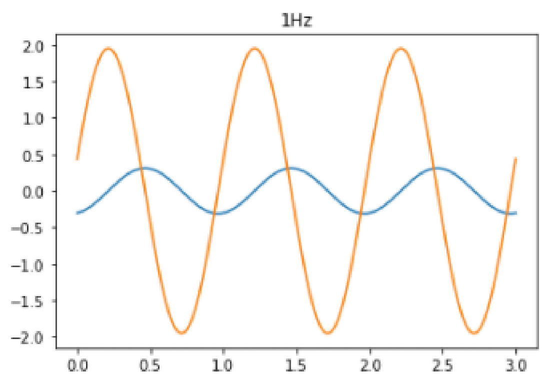
```python
import matplotlib.pyplot as plt
import numpy as np
R = 1;
C = 0.696;
a = 1/(R*C)
A = 2
w1=(2*np.pi)/0.1
# 0.1Hz
t=np.linspace(0, 3*2*np.pi/w1, 300)
s1=A*(a*np.sin(w1*t)-w1*np.cos(w1*t))/(w1**2+a**2)
sd1=A*(a*w1*np.cos(w1*t)+w1**2*np.sin(w1*t))/(w1**2+a**2)
fig,ax=plt.subplots()
ax.set_title('0.1Hz')
ax.plot(t,s1,t,sd1)
plt.show()
max(s1),max(sd1)
```




```
(0.031821476916925655, 1.9994268675722975)
```

Rys. 1. Wykres i funkcja obliczająca maksymalne napięcie na oporniku i kondensatorze dla częstości 0.1Hz.

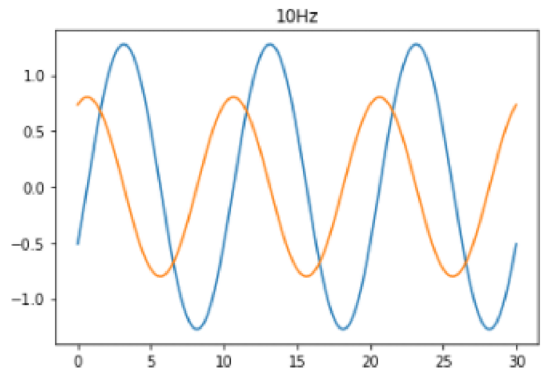
```python
[19] # 1Hz
     w2=(2*np.pi)/1
     t=np.linspace(0, 3*2*np.pi/w2, 300)
     s2=A*(a*np.sin(w2*t)-w2*np.cos(w2*t))/(w2**2+a**2)
     sd2=A*(a*w2*np.cos(w2*t)+w2*w2*np.sin(w2*t))/(w2**2+a**2)
     fig,ax=plt.subplots()
     ax.set_title('1Hz')
     ax.plot(t,s2,t,sd2)
     plt.show()
     max(s2),max(sd2)
```




```
(0.3102977110969568, 1.9496737105622215)
```

Rys. 2. Wykres i funkcja obliczająca maksymalne napięcie na oporniku i *kondensatorze dla częstości 1Hz.*

```
# 10Hz
w3=(2*np.pi)/10
t=np.linspace(0, 3*2*np.pi/w3, 300)
s3=A*(a*np.sin(w3*t)-w3*np.cos(w3*t))/(w3**2+a**2)
sd3=A*(a*w3*np.cos(w3*t)+w3*w3*np.sin(w3*t))/(w3**2+a**2)
fig,ax=plt.subplots()
ax.set_title('10Hz')
ax.plot(t,s3,t,sd3)
plt.show()
max(s3),max(sd3)
```




```
(1.2753420036175007, 0.8013191320412988)
```

Rys. 3. Wykres z funkcja obliczająca maksymalne napięcie na oporniku i kondensatorze dla częstości 10Hz.

Maksymalne napięcie na oporniku i kondensatorze wynosi:

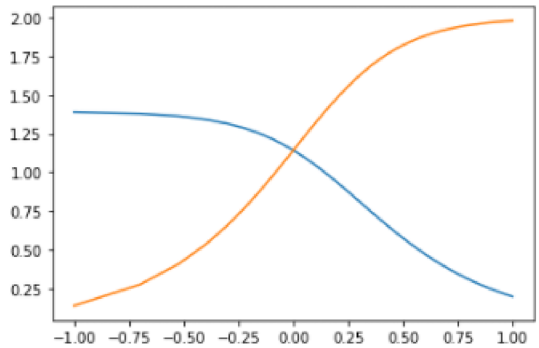
dla częstośi 0.1Hz: kondensator - 0.031821476916925655, opornik - 1.9994268675722975,

dla częstośi 1Hz: kondensator - 0.3102977110969568, opornik - 1.9496737105622215,

dla częstośi 10Hz: kondensator - 1.2753420036175007, opornik - 0.8013191320412988,

- Wykonaj wykres charakterystyki amplitudowo-częstościowej w przedziale częstości (0.1, 10) . Oś częstości przedstaw w skali logarytmicznej o podstawie 10

```
logw = []
ss=[]
ssd=[]
ww=np.linspace(0.1,10,100)
for w in ww:
  t = np.linspace(0, 2 * np.pi / w, 100)
  s=A*(a*np.sin(w*t)-w*np.cos(w*t))/(w**2+a**2)
  sd=A*(a*w*np.cos(w*t)+w*w*np.sin(w*t))/(w**2+a**2)
  ss.append(max(s))
  ssd.append(max(sd))
  logw.append(np.log10(w))
fig,ax=plt.subplots()
ax.plot(logw,ss,logw,ssd)
plt.show()
```



Rys. 4. Funkcja i wykres charakterystyki amplitudowo-częstościowej w przedziale częstości (0.1, 10)

0.2 Oscylator harmoniczny z tłumieniem bez wymuszenia

- Zapisz poprawnie równanie dla masy m1 na sprężynie k1 z tłumieniem δ.

$$\ddot{x} + 2\delta\dot{x} + \omega^2 x = 0$$
$$\omega^2 = \frac{k}{m}$$

Równanie po przekształceniu$= \ddot{x} + 2\delta\dot{x} + \frac{k}{m}x = 0$

- Zaklasyfikuj typ równania.

Równanie różniczkowe jednorodne rzędu drugiego.

- sprowadź równanie układu do układu równań stopnia pierwszego, zapisz równanie w postaci macierzowej

$$\ddot{x} + 2\delta\dot{x} + \omega^2 x = 0$$

$$\begin{cases} \dot{x} = v \\ \dot{v} = -2\delta v - \omega^2 x \end{cases}$$
$$\begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -2\delta \end{bmatrix}$$
$$\det \begin{bmatrix} -\lambda & 1 \\ \delta^2 & -2\delta - \lambda \end{bmatrix}$$

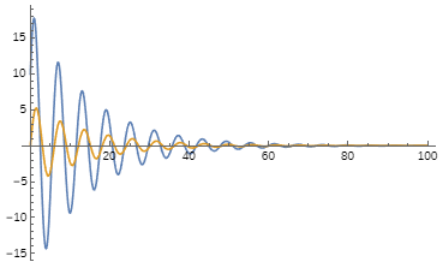
- wyznacz częstość własną drgań

$$-\lambda(-2\lambda - \lambda) + \omega^2 = 0$$
$$2\delta\lambda + \lambda^2 + \omega^2 = 0$$
$$\lambda^2 + 2\delta\lambda + \omega^2 = 0$$
$$a = 1$$
$$b = 2\delta$$
$$\delta = 0,07$$
$$\text{k=6.5}$$
$$\text{m=6}$$
$$\frac{k}{m} = \omega^2$$
$$\Delta = 4\delta^2 - 4\omega^2$$
$$\Delta = 4 * 0,07^2 - 4\left(\frac{k}{m}\right)$$
$$\Delta = 4 * 0,07^2 - 4\left(\frac{6,5}{6}\right)$$
$$\Delta = -4.314$$
$$\Delta = 4.314i^2$$
$$\sqrt{\Delta} = 2.077i$$
$$\lambda_1 = -0,07 - 1.039i$$
$$\lambda_2 = -0,07 + 1.039i$$

Częstości własne drgań w przybliżeniu to:
-0,07 - 1.039i
-0,07 + 1.039i.

- narysuj przebieg wychylenia w czasie dla dwóch wybranych warunków początkowych. Skomentuj interpretację tych warunków Warunki początkowe wynoszą
x[0] = 10
$\dot{x}[0] = 16$
$\delta = 0.07$
$\omega = 1.041$

```
In[109]:= ω = 1.041
δ = 0.07
sol = DSolve[{x''[t] + 2 δ x'[t] + ω^2 x[t] == 0, x[0] == 10, x'[0] == 16}, x[t], {t, 0, 100}];
sol2 = DSolve[{x2''[t] + 2 δ x2'[t] + ω^2 x2[t] == 0, x2[0] == 0, x2'[0] == 6}, x2[t], {t, 0, 200}];
Plot[{x[t] /. sol, x2[t] /. sol2}, {t, 0, 100}, PlotRange → All]
Out[109]= 1.041
Out[110]= 0.07
Out[113]=
```

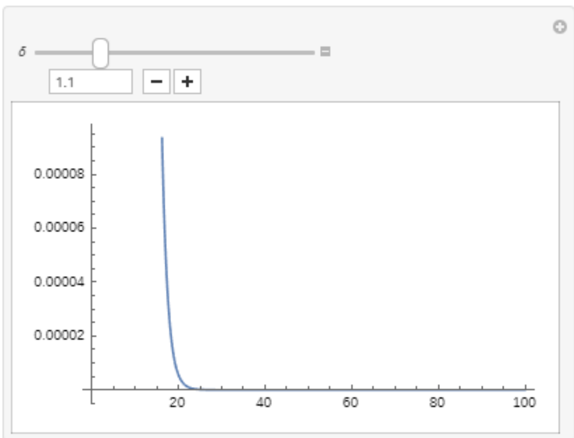


Rys. 5. Funkcja obliczająca przebieg wychylenia w czasie dla dwóch wybranych warunków początkowych.

Mimo dużej różnicy początkowej w wychyleniu - w krótkim czasie następuje wyrównanie funkcji.

- wyznacz tłumienie krytyczne dla tego układu

```
In[141]:= δ = 0.07
w = 1.041
Manipulate[
 sol = DSolve[{x''[t] + 2 δ x'[t] + w^2 x[t] == 0, x[0] == 6, x'[0] == 3}, x[t], {t, 0, 50}];
 Plot[Evaluate[x[t] /. sol], {t, 0, 100}], {δ, 0, 5}]
Out[141]= 0.07
Out[142]= 1.041
Out[143]=
```



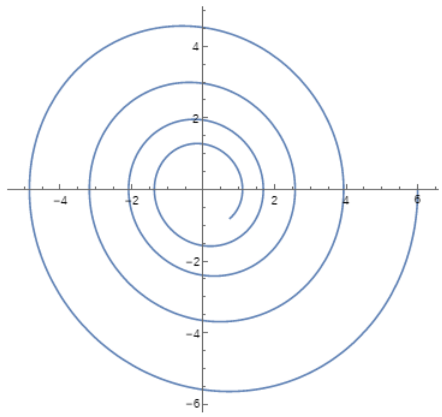
Rys. 6. Funkcja obliczająca przebieg wychylenia w czasie dla dwóch wybranych warunków początkowych.

Tłumienie krytyczne wynosi - 1.1

- narysuj ruch układu w przestrzenie fazowej (x,v)

```
In[159]:= δ = 0.07
w = 1.041
sol = NDSolve[{x''[t] + 2 δ x'[t] + w^2 x[t] == 0, x[0] == 6, x'[0] == 0}, {x[t], x'[t]}, {t, 25}];
ParametricPlot[Evaluate[{x[t], x'[t]} /. sol], {t, 0, 25}]

Out[159]= 0.07

Out[160]= 1.041
```

Out[162]=



Rys. 7. Ruch układu w przestrzeni fazowej (x,v).

0.3 Oscylator harmoniczny tłumieniem i wymuszeniem harmonicznym

- Zapisz poprawnie równanie dla masy m1 na sprężynie k1 z tłumieniem i wymuszeniem harmonicznym o częstości ω i amplitudzie 0.1m.

$$\omega^2 = \frac{k}{m}$$
$$A=0.1$$

$$\ddot{x} + 2\delta\dot{x} + \omega^2 x = A\sin(\omega t)$$
$$\ddot{x} + 2\delta\dot{x} + \frac{k}{m}x = 0.1\sin(\sqrt{\frac{k}{m}}t)$$

- Zaklasyfikuj typ równania.

Równanie różniczkowe niejednorodne rzędu drugiego.

- Sprowadź równanie układu do układu równań stopnia pierwszego, zapisz równanie w postaci macierzowej

$$\ddot{x} + 2\delta\dot{x} + \omega^2 x = A\sin(\omega t)$$

$$
\begin{cases}
\dot{x} = \dot{v} \\
v = A\sin(\omega t) - 2\delta\dot{v} - \omega^2 x
\end{cases}
$$

$$
\begin{bmatrix} x' \\ v' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -2\delta \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ Asin(\omega t) \end{bmatrix}
$$

- narysuj przebieg wychylenia w czasie dla zerowych warunków początkowych i wymuszenia z częstością 0.5ω, ω oraz 2ω, (ω-częstość drgań własnych). Skomentuj interpretację tych przebiegów.

```
In[89]:=
δ = 0.07
w = 1.041
tmax = 100
A = 0.1

sol = NDSolve[{x''[t] + 2 δ x'[t] + w^2 x[t] == A Sin[0.5 t], x[0] == 0, x'[0] == 0}, {x[t], x'[t]}, {t, 0, tmax}];
Plot[Evaluate[{x[t], x'[t]} /. sol], {t, 0, tmax}]

sol2 = NDSolve[{x''[t] + 2 δ x'[t] + w^2 x[t] == A Sin[t], x[0] == 0, x'[0] == 0}, {x[t], x'[t]}, {t, 0, tmax}];
Plot[Evaluate[{x[t], x'[t]} /. sol2], {t, 0, tmax}]

sol3 = NDSolve[{x''[t] + 2 δ x'[t] + w^2 x[t] == A Sin[2 t], x[0] == 0, x'[0] == 0}, {x[t], x'[t]}, {t, 0, tmax}];
Plot[Evaluate[{x[t], x'[t]} /. sol3], {t, 0, tmax}]

Out[89]= 0.07
```

Rys. 8. Kody do narysowania wychylenia w czasie dla wymuszenia z częstościami kolejno: 0.5ω, ω, 2ω.

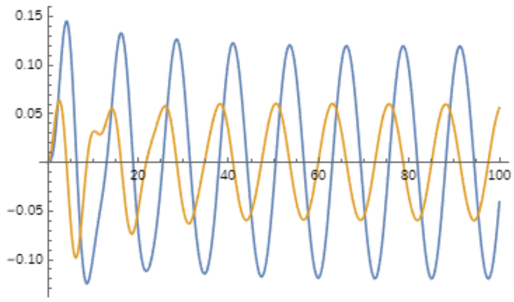
Out[89]= 0.07

Out[90]= 1.041

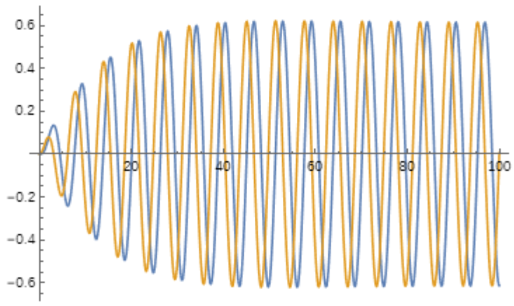
Out[91]= 100

Out[92]= 0.1

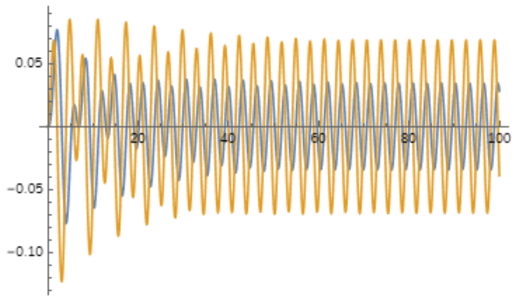
Out[94]=



Out[96]=



Out[98]=



Rys. 9. Wykres wychylenia w czasie dla wymuszenia z częstościami kolejno: 0.5ω, ω, 2ω.

Na podstawie przebiegów, możemy stwierdzić że w każdym przypadku po ok. 30 sekundach mamy do czynienia ze zjawiskiem rezonansu

- Na czym polega zjawisko rezonansu?

Rezonans to zjawisko wzrastania amplitudy drgań układu drgającego, w którym częstotliwość siły wymuszającej te drgania jest podobna do częstości własnej układu, czyli częstości drgań swobodnych tego układu.

0.4 Oscylator harmoniczny podwójny bez tłumienia i bez wymuszenia

- Zapisz poprawnie równanie dla masy m1 na sprężynie k1 i połączonej z nią masy m2 na sprężynie k2 (przykład z wykładu).

$$\begin{cases} m_1 * \ddot{x}_1 = -k1x1 = +k2(x2 - x1) \\ m_2 * \ddot{x}_2 = -k_2(x_2 - x_1) \end{cases}$$

$$\begin{cases} \ddot{x}_1 = \frac{1}{m_1}(-k_1 x_1 + k_2(x_2 - x_1)) \\ \ddot{x}_2 = \frac{1}{m_2}(-k_2(x_2 - x_1)) \end{cases}$$

$$\begin{cases} \ddot{x}_1 = -\frac{k_1}{m_1}x_1 + \frac{k_2}{m_1}x_2 - \frac{k_2}{m_1}x_1 \\ \ddot{x}_2 = -\frac{k_2}{m_2}x_2 - \frac{k_2}{m_2}x_1 \end{cases}$$

- Zaklasyfikuj typ równania.

Równanie różniczkowe niejednorodne rzędu drugiego

- sprowadź równanie układu do układu równań stopnia pierwszego, zapisz równanie w postaci macierzowej.

$$x' = V \quad x'' = V'$$

$$\begin{cases} \dot{v}_1 = v_1 \\ \dot{v}_1 = \frac{-k_1}{m_1}x_1 + \frac{k_2}{m_1}\dot{x}_2 - \frac{k_2}{m1}x_1 \\ \dot{x}_2 = v_2 \\ \dot{v}_2 = -\frac{k_2}{m_2}x_2 + \frac{k_2}{m_2}x_1 \end{cases}$$

$$\begin{bmatrix} x_1 \\ \dot{v} \\ \ddot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1}{m_1} \cdot \frac{k_2}{m_1} & 0 & \frac{k_2}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & 0 & \frac{k_2}{m_2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ v_1 \\ x_2 \\ v_2 \end{bmatrix}$$

- wyznacz wartości własne macierzy układu.

$$\det(A - \lambda f) = 0$$

$$det = \begin{bmatrix} -\lambda & 1 & 0 & 0 \\ (-\frac{k_1}{m_1} - \frac{k_2}{m_1}) & -\lambda & \frac{k_2}{m_1} & 0 \\ 0 & 0 & -\lambda & 1 \\ \frac{k_2}{m_2} & 0 & \frac{k_2}{m_2} & -\lambda \end{bmatrix} = 0$$

$\lambda_1 = 0. - 0.732912i$ $\lambda_2 = 0. + 0.732912i$ $\lambda_3 = 0. - 1.77375i$ $\lambda_4 = 0. + 1.77375i$

- Jaki jest sens fizyczny tych wielkości ?

Przedstawiają one częstość drgań układu.

- narysuj przebiegi w przestrzeni fazowej (x,v) nakładając na siebie ruch obu mas. Warunki początkowe jak w poprzednim punkcie (trzy przypadki).

0.5 Wahadło matematyczne

Dane jest wahadło matematyczne o parametrach: Długość wahadła l=C (z tabeli parametrów) kąt początkowego wychylenia (0) = Rozwiąż numerycznie równanie wahadła. Wykonaj wykres drgań wahadła (1 okres) i porównaj z przypadkiem małych drgań (czyli założenie sin() =) Definiując błąd jako różnicę między rozwiązaniem równania wahadła a równaniem oscylatora , narysuj wykres tego błędu w czasie. Przedyskutuj czy jest sens wprowadzać pojęcie błędu względnego w takim przypadku (np. w odniesieniu do rozwiązania wahadła)? Przygotuj wykres przedstawiający okres drgań wahadła w funkcji wielkości warunku początkowego.

0.6 Modele rozwoju epidemii

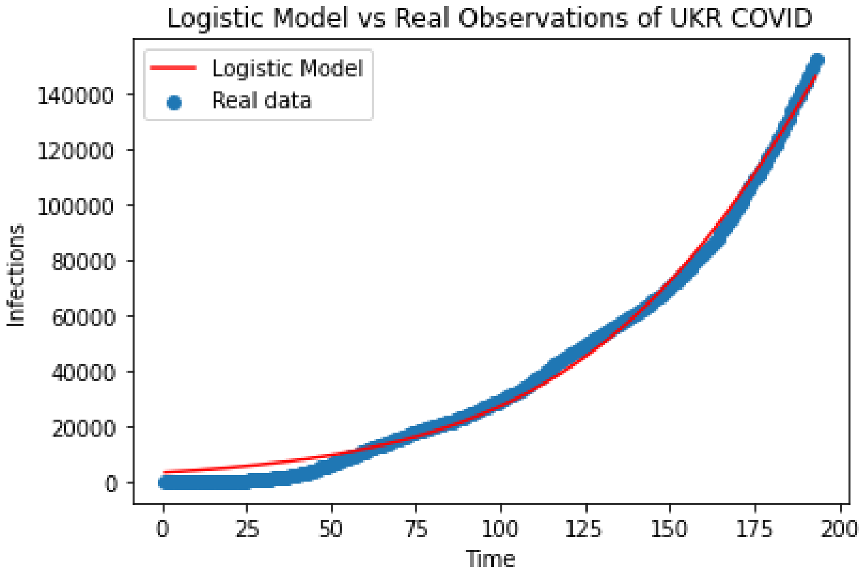
Ćwiczenie polega na dobieraniu parametrów modelu do danych doświadczalnych dotyczących COVID-19. Dane można znaleźć m. in tu: https://ourworldindata. org/coronavirus Każda osoba analizuje inny kraj (proszę to uzgodnić w grupie) * na ocenę 3 - model Malthusa (pasuje tylko do początku epidemii) * na ocenę 4 model logistyczny (obejmuje wysycanie ale nie ozdrowienia) * na ocenę 5 - porównanie modeli logistycznych Verhulst - Gompertz, lub eksperymentalne dobieranie parametrów do modelu SIR Fitowanie krzywej należy wykonać przy pomocy dowolnej biblioteki

Wybrany kraj: **Ukraina**, przedział czasowy **od 08.03.2020 do 29.11.2020**.

Do wykonania zadania wykorzystam model logistyczny **Verhulsta**.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
import math
import scipy.optimize as optim
warnings.filterwarnings('ignore')

# import pliku
# plik został przeze mnie wyczyszczony ze zbędnych danych
data = pd.read_csv('/content/sample_data/start_covid_ukr.csv', sep=';')
data = data['total_cases']
data = data.reset_index(drop=False)
data.columns = ['Timestep', 'Total Cases']

# funkcje logistyczne
def my_logistic(t, a, b, c):
  return c / (1 + a * np.exp(-b*t))
def my_logistic2(t):
  return c / (1 + a * np.exp(-b*t))

# losowa inicjalizacja parametrów a, b i c
p0 = np.random.exponential(size=3)

#ustawienie górnych i dolnych ograniczeć dla a, b i c
bounds = (0, [100000., 3., 1000000000.])

# dopasowanie krzywej
x = np.array(data['Timestep']) + 1
y = np.array(data['Total Cases'])
(a,b,c),cov = optim.curve_fit(my_logistic, x, y, bounds=bounds, p0=p0)

# wykreślenie funkcji w porównaniu z rzeczywistymi danymi
plt.scatter(x, y)
plt.plot(x, my_logistic2(x), color="red")
plt.title('Logistic Model vs Real Observations of UKR COVID')
plt.legend(['Logistic Model', 'Real data'])
plt.xlabel('Time')
plt.ylabel('Infections')
```

Rys. 10. Kod realizujący model logistyczny Verhulsta oraz generujący wykres porównujący ten model z właściwymi danymi



Rys. 11. Wykres porównujący model logistyczny Verhulsta z właściwymi danymi

Podsumowanie: Na podstawie powyższego wykresu widzimy, że udało się znaleźć funkcję logistyczną, bardzo zbliżoną do właściwych danych COVID z Ukrainy.

0.7 Automat komórkowy Snowflake

```
size = 10
def neigh():
  nearest=[[0 for _ in range(size)] for _ in range(size)]
  for y in range(size):
    for x in range(size):
      nearest[y][x] = count(y,x)
  return nearest

def count(y, x):
  dy=[0,0,1,-1]
  dx=[-1,1,0,0]
  count = 0
  for k in range(len(dx)):
    kx=(x+dx[k]) % size
    ky=(y+dy[k]) % size
    count=count+tab[ky][kx]
  return count

def inicialize(y,x):
  tab=[[0 for _ in range(y)] for _ in range(x)]
  tab[4][4]=1
  tab[4][5]=1
  return tab

def print_tab(tab):
  for y in range(size):
    for x in range(size):
      print(tab[x][y], end=' ')
    print(' ')


tab=inicialize(size,size)
print_tab(tab)
print('====================')
for x in range(size):
  nearest = neigh()
  res=tab
  for y in range(size):
      for x in range(size):
        if tab[y][x] == 0:
          if nearest[y][x]==1:
            res[y][x]=1
  tab=res
  print_tab(tab)
  print('====================')
```

Rys. 12. Kod realizujący Automat komórkowy Snowflake

```
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
====================
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 1 1 1 0 0 0 0
 0 0 0 1 1 1 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
====================
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 1 1 1 1 1 0 0 0
 0 0 1 1 1 1 1 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
====================
 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 1 1 1 0 0 0 0
 0 0 1 0 1 0 1 0 0 0
 0 1 1 1 1 1 1 1 0 0
 0 1 1 1 1 1 1 1 0 0
 0 0 1 0 1 0 1 0 0 0
 0 0 0 1 1 1 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0
====================
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 1 1 1 0 0 0 0
 0 0 1 0 1 0 1 0 0 0
 1 1 1 1 1 1 1 1 1 0
 1 1 1 1 1 1 1 1 1 0
 0 0 1 0 1 0 1 0 0 0
 0 0 0 1 1 1 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
====================
 0 0 0 1 1 1 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 1 1 1 0 0 0 0
 1 0 1 0 1 0 1 0 1 0
 1 1 1 1 1 1 1 1 1 0
 1 1 1 1 1 1 1 1 1 0
 1 0 1 0 1 0 1 0 1 0
 0 0 0 1 1 1 0 0 0 0
 0 0 0 0 1 0 0 0 0 0
 0 0 0 1 1 1 0 0 0 0
```

Rys. 13. Automat komórkowy Snowflake