Graficzne przedstawienie rozprzestrzeniania się COVID-19

Projekt zaliczeniowy - Programowanie dla fizyków

Emil Śmiech, Damian Łącz, Amelia Królczyk

Październik 2020

Spis treści:

1. Wstęp
2. Opisanie metody Monte Carlo
3. Opisanie modelu SEIRD
4. Podsumowanie
5. Bibliografia

1) Wstęp

Skupiamy się na zjawisku epidemiologicznym znanym nam szerzej od marca 2020r., jako koronawirus SARS-CoV-2 (COVID-19), w celu wprowadzenia technik modelowania epidemiologicznego oraz technik COVID, od w pełni połączonej, ciągłej dynamiki czasu do pełnej skali. Używamy wyrażenia analitycznego w formie zamkniętej dla w pełni połączonego równania różniczkowego. Ćwiczenia przeprowadzono w ramach zajęć Programowanie dla Fizyków, celem dokładnego pokazania rozprzestrzeniania się wirusa COVID-19 na przykładach Monte Carlo oraz SEIRD.

2) Opisanie metody Monte Carlo

Metoda stosowana do modelowania matematycznego procesów zbyt złożonych (obliczania całek, łańcuchów procesów statystycznych), aby można było przewidzieć ich wyniki za pomocą podejścia analitycznego. Istotną rolę w tej metodzie odgrywa losowanie (wybór przypadkowy) wielkości charakteryzujących proces, przy czym losowanie dokonywane jest zgodnie z rozkładem, który musi być znany.

Dokładność wyniku uzyskanego tą metodą jest zależna od liczby sprawdzeń i jakości użytego generatora liczb pseudolosowych. Zwiększanie liczby prób nie zawsze zwiększa dokładność wyniku, ponieważ generator liczb pseudolosowych ma skończenie wiele liczb losowych w cyklu. Przykładowo całkowanie tą metodą jest używane w przypadkach, kiedy szybkość otrzymania wyniku jest ważniejsza od jego dokładności (np. obliczenia inżynierskie).

Przykładem, który najlepiej obrazuje metodę Monte Carlo i liczbę π , jest opuszczenie ołówka na wyznaczony obszar – kwadrat. Powtarzamy taką czynność n razy. Liczba π , to liczba punktów w kole w stosunku do liczby wszystkich punktów . Im więcej razy powtórzymy czynność, tym bliższą otrzymamy wartość liczby π

Przykładowy kod Monte Carlo, generujący liczbę π :

```
import math
import random
random.seed()

n = 10000

for i in range(n):
    x = random.random()
    y = random.random()
    z = (x,y)

    if x** 2+y** 2 <= 1:
        print z
    else:
        del z
```

Metodą Monte Carlo można obliczyć pole figury zdefiniowanej nierównością:

$$x^2 + y^2 = R^2$$

czyli koła o promieniu R i środku w punkcie (0,0).

1. Losuje się n punktów z opisanego na tym kole kwadratu – dla koła o R = 1 współrzędne wierzchołków (-1,-1), (-1,1), (1,1), (1,-1).

2. Po wylosowaniu każdego z tych punktów trzeba sprawdzić czy jego współrzędne spełniają powyższą nierówność (tj. czy punkt należy do koła).

Wynikiem losowania jest informacja, że ze wszystkich prób k było trafionych, zatem pole koła wynosi:

$$P_k = P\frac{k}{n},$$

gdzie P jest polem kwadratu opisanego na tym kole (dla $R = 1 : P = 4$).

Obliczamy całkę metodą Monte Carlo przy pomocy kodu:

```
# ! /usr/bin/env python

from random import uniform
from math import pi

import matplotlib.pyplot as plt

xpoints = []
ypoints = []

xcircle = []
ycircle = []

Ncircle = 0
Nsquare = 0

Ntot = 100000

for i in range(Ntot):
   x = uniform(-1,1)
   y = uniform(-1,1)

   xpoints.append(x)
   ypoints.append(y)
```

```
    Nsquare += 1
    if(x* x + y* y < 1):
        Ncircle +=1
        xcircle.append(x)
        ycircle.append(y)


mcpi = 4.0 * float(Ncircle) / float(Nsquare)

print("mcpi = ", mcpi )

print("pi-mcpi = ", pi - mcpi)


plt.plot(xpoints,ypoints,'or')
plt.plot(xcircle,ycircle,'ob')
plt.show()
```

3) Opisanie modelu SEIRD

Przy pomocy modelu SEIRD (Susceptible-Exposed-Infected-Recovered-Dead – Podatni-Narażeni-Zakażeni-Ozdrowieńcy-Zmarli) możemy przeprowadzić symulacje rozprzestrzeniania się nowych,nieznanych chorób. Dzięki temu możemy w przybliżeniu przewidzieć jak będzie zachowywać się wirus, jak szybko będzie się rozprzestrzeniać i ile osób się nim zakazi.

Algorytm metody SEIRD możemy przedstawić za pomocą równań:

$$\frac{dS}{dt} = -\beta SI,$$

$$\frac{dE}{dt} = -\beta_D SI - T_E I,$$

$$\frac{dI}{dt} = T_I E - \beta_I I,$$

$$\frac{dR}{dt} = \beta_D I\left(1 - f\right),$$

$$\frac{dD}{dt} = \beta_D I f,$$

gdzie:

β_I , β_D – wskaźniki szybkości zakażeń (Infected, Dead)

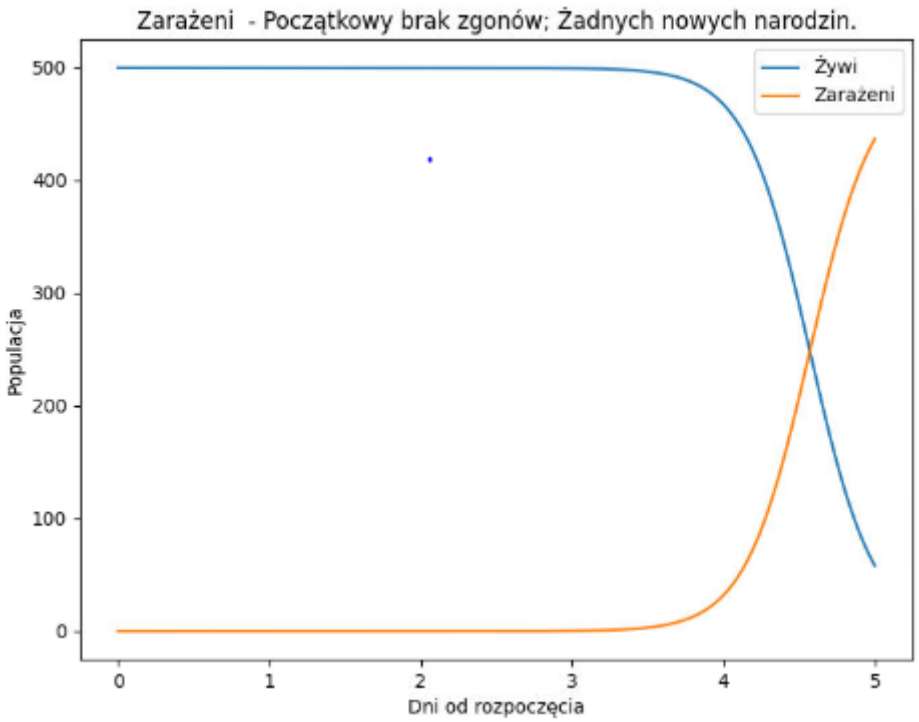
T_E , T_I – okresy trwania choroby pomiędzy odpowiednimi etapami (Exposed, Infected)

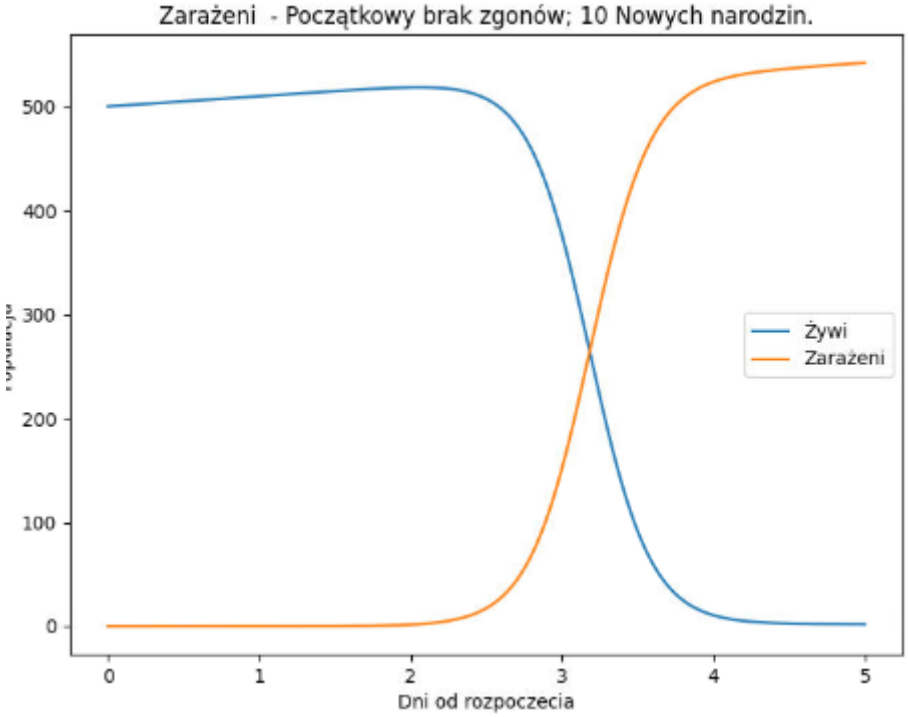
Powyższe równania znajduja swoje zastosowanie w kodzie:

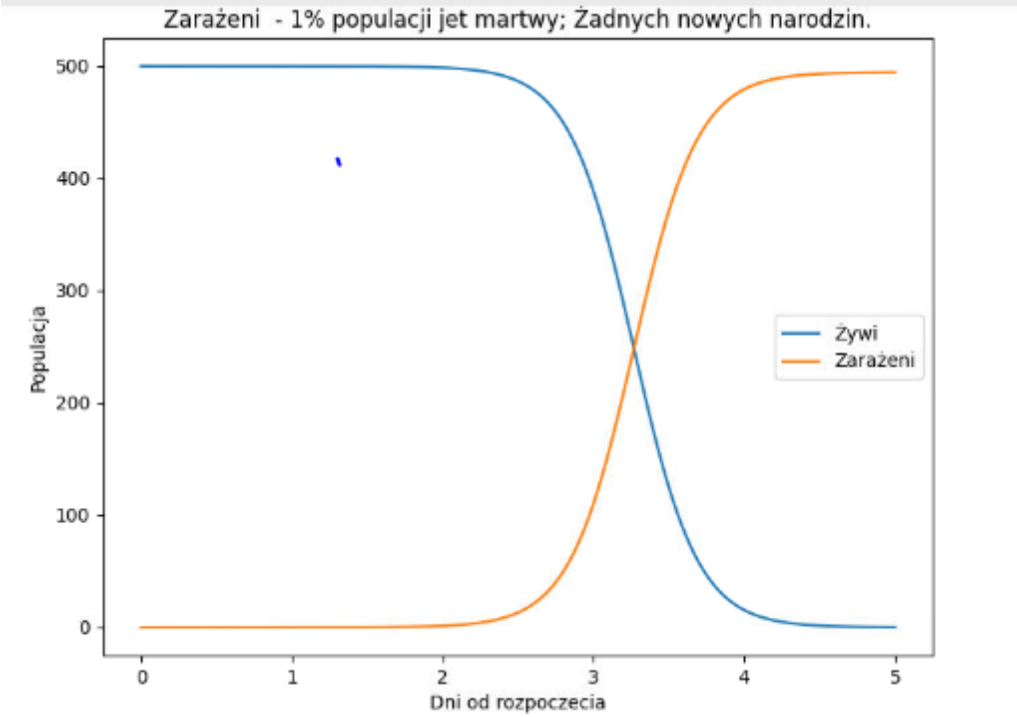
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
plt.ion()
plt.rcParams['figure.figsize'] = 10, 8

P = 0 # birth rate
d = 0.0001 # natural death percent (per day)
B = 0.0095 # transmission percent (per day)
G = 0.0001 # resurect percent (per day)
A = 0.0001 # destroy percent (per day)



# solve the system dy/dt = f(y, t)
def f(y, t):
   Si = y[0]
   Zi = y[1]
   Ri = y[2]
   # the model equations (see Munz et al. 2009)
   f0 = P - B* Si* Zi - d* Si
   f1 = B* Si* Zi + G* Ri - A* Si* Zi
   f2 = d* Si + A* Si* Zi - G* Ri
   return [f0, f1, f2]



# initial conditions
S0 = 500. # initial population
Z0 = 0 # initial coronavirus population
R0 = 0 # initial death population
y0 = [S0, Z0, R0] # initial condition vector
t = np.linspace(0, 5., 1000) # time grid
# solve the DEs
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
# plot results
plt.figure()
plt.plot(t, S, label='Living')
plt.plot(t, Z, label='coronavirus')
plt.xlabel('Days from outbreak')
plt.ylabel('Population')
plt.title('coronavirus - No Init. Dead Pop.; No New Births.')
plt.legend(loc=0)
plt.show()
```

```
# change the initial conditions
R0 = 0.01* S0 # 1% of initial pop is dead
y0 = [S0, Z0, R0]
# solve the DEs
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
plt.figure()
plt.plot(t, S, label='Living')
plt.plot(t, Z, label='coronavirus')
plt.xlabel('Days from outbreak')
plt.ylabel('Population')
plt.title('coronavirus - 1% Init. Pop. is Dead; No New Births.')
plt.legend(loc=0)
plt.show()

# change the initial conditions
R0 = 0.01* S0 # 1% of initial pop is dead
P = 10 # 10 new births daily
y0 = [S0, Z0, R0]

# solve the DEs
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
plt.figure()
plt.plot(t, S, label='Living')
plt.plot(t, Z, label='coronavirus')
plt.xlabel('Days from outbreak')
plt.ylabel('Population')

plt.title('coronavirus - 1% Init. Pop. is Dead; 10 Daily Births')
plt.legend(loc=0)
plt.show()
```

Dzięki metodzie SEIRD otrzymujemy wizualizacje:




Zarażeni - Początkowy brak zgonów; 10 Nowych narodzin.
500
400
300
200
100
0
0
1
2
3
4
5
Dni od rozpoczecia
Żywi
Zarażeni



4) Podsumowanie

Metoda Monte Carlo jest bardzo wiarygodna i dokładnie pozwala określić rozwój pandemii. Im więcej danych posiadamy, tym wyniki są dokładniejsze. Model SEIRD mimo ilości wprowadzonych danych wydaje się być bardziej dokładny. Powyższe sposoby sprawdzą się także w przypadku modelowania zachorowań innych wirusów. Ich wspólną cechą jest prosta konstrukcja, która ogranicza możliwość wystąpienia błędu.

5) Bibliografia

1) https://arxiv.org/pdf/1503.01104.pdf?fbclid=IwAR31jR3R_JQlalCoQk6s8A4uhCxfhsrtvBosUrfo
k

2) https://www.mdpi.com/2076-3417/10/15/5162/htm?fbclid=IwAR3XFHxlqJ8wSWpUECUUtDq

3) https://pl.wikipedia.org/wiki/Metoda_Monte_Carlo