UNIVERZA V LJUBLJANI

FAKULTETA ZA MATEMATIKO IN FIZIKO

Poročilo

Domača naloga 2 - Epidemiološki model razširjanja bolezni

Luka Orlić

1. maj 2023

Kazalo

1 Uvod

V epidemiologiji obstajajo različni načini, kako modelirati in predvidevati rast oz širjenje bolezni. V tej nalogi bo obravnavan oziroma uporabljen model ŠIR", ki je akronim angleških besed Šucceptible Infected Recovered/Removed", ta model najpogosteje uporablja navadne diferencialne enačbe znane tudi kot NDE oziroma v angleščini kot ODE.

V nalogi bomo iz podanih začetnih stanj bolezni probali predvidevati potek bolezni iz epidemiološkega pogleda, kako je od reprodukcijskega števila, o katerem bomo več povedali kasneje, odvisno največje število okuženih v danem (enem) trenutku.

V nalogi imamo populacijo ($N = 1000$) 1.000 ljudi, od katerih je na začetku modela 990 susceptibilnih za okužbo, 10 okuženih in 0 prebolelih ($S = 990, I = 10, R = 0$). Opažamo, da je vsota po enačbi (1) vedno enaka začetni populaciji. Tu privzamemo, da so vse osebe v skupini prebolelih bodisi imune od bodisi preminile zaradi bolezni, ter ne morejo zbolet več kot enkrat.

Iz originalnih enačb (2), (3), in (4) smo dobili enačbe, kjer smo skalirali čas tako kot je zapisano v enačbi (5), ter v te enačbe vstavili tako imenovano reprodiukcijsko število, kot ga opisuje enačba (6), ter tako smo dobili enačbe (7), (8), in (9), ki jih bomo v programu uporabljali.

$$S(t) + I(t) + R(t) = N \tag{1}$$

$$\dot{S} = -\frac{\beta}{N} \cdot SI \tag{2}$$

$$\dot{I} = \frac{\beta}{N} \cdot SI - \gamma I \tag{3}$$

$$\dot{R} = \gamma I \tag{4}$$

$$t' = \gamma t \tag{5}$$

$$R_0 = \frac{\beta}{\gamma} \tag{6}$$

$$\dot{S} = -\frac{R_0}{N} \cdot SI \tag{7}$$

$$\dot{I} = \frac{R_0}{N} \cdot SI - I \tag{8}$$

$$\dot{R} = I \tag{9}$$

2 Reševanje naloge

Za doseganje ciljev smo sestavili naslednjo kodo:

```
import scipy
import numpy as np
import matplotlib.pyplot as plt
#------------------------------DATASTORAGE-----------------------------#

tstart = 0
tend = 20

s_0 = 990
i_0 = 10
r_0 = 0

n = 1000
r0 = 2

t = np.linspace(tstart,tend,5000)

#-----------------------------PODNALOGA A-----------------------------#

def epidemijaAB(t, w, r0, n):
    s,i,r = w
    return [-r0/n*s*i, r0/n*s*i-i, i]

sol1 = scipy.integrate.solve_ivp(epidemijaA, [tstart, tend],
[s_0,i_0,r_0], t_eval = t, args=(r0,n), dense_output=True)

w = sol1.sol(t)
plt.plot(sol1.t,w.T[:,0],'-',c='red',label='s(t)')
plt.plot(sol1.t,w.T[:,1],'-',c='blue',label='i(t)')
plt.plot(sol1.t,w.T[:,2],'-',c='purple',label='r(t)')

plt.title("S, I, R : primer A")
plt.xlabel("Time")
plt.ylabel("Stevilo ljudi")

plt.legend()
plt.grid()
plt.show()

#-----------------------------PODNALOGA B-----------------------------#

xos = []
yos = []


for r0 in range(11):

    sol1 = scipy.integrate.solve_ivp(epidemijaAB, [tstart, tend],
    [s_0,i_0,r_0], t_eval = t, args=(r0,n), dense_output=True)

    w = sol1.sol(t)
    a = (sol1.t,w.T[:,1])
    mm = max(a[1])
```

```
    xos.append(r0)
    yos.append(mm)

# --> Za vsak posamezni graf pokaze SIR graf
#
#    plt.plot(sol1.t,w.T[:,0],'-',c='red',label='s(t)')
#    plt.plot(sol1.t,w.T[:,1],'-',c='blue',label='i(t)')
#    plt.plot(sol1.t,w.T[:,2],'-',c='purple',label='r(t)')
#    plt.legend()
#    plt.title("S, I, R : primer B : r0 = " + str(r0))
#    plt.xlabel("Time")
#    plt.ylabel("Stevilo ljudi")
#    plt.grid()
#    plt.show()

plt.plot(xos, yos, '--bo')
plt.title("Graf SIR modela : max( I(t) )[R0]")
plt.xlabel("R0")
plt.ylabel("Stevilo okuzenih")

plt.grid()
plt.show()

#----------------------------drugi plot gr----------------------------#


#----------------------------PODNALOGA C----------------------------#
def epidemijaC(t, w, n):
    s,i,r,r0 = w
    return [-r0/n*s*i, r0/n*s*i-i, i,  0.6-0.01*i-0.00001*i*s ]

r00 = 1.001

sol1 = scipy.integrate.solve_ivp(epidemijaC, [tstart, tend],
[s_0,i_0,r_0,r00], t_eval = t, args=(n,), dense_output=True)

w = sol1.sol(t)
plt.plot(sol1.t,w.T[:,0],'-',c='red',label='s(t)')
plt.plot(sol1.t,w.T[:,1],'-',c='blue',label='i(t)')
plt.plot(sol1.t,w.T[:,2],'-',c='purple',label='r(t)')
plt.plot(sol1.t,w.T[:,3],'-',c='green',label="r_0(t)")

plt.title("S, I, R : primer C")
plt.xlabel("Time")
plt.ylabel("Stevilo ljudi")

plt.legend()
plt.grid()
plt.show()

#--------------

plt.plot(sol1.t,w.T[:,3],'-',c='green',label="r_0(t)")
plt.title("R_0(t)")
plt.xlabel("Time")
plt.ylabel("Stevilska vrednost")

```

```
111 plt.grid()
112 plt.show()
```

V delu kode imenovan "datastorageshranjujemo spremenljivke.

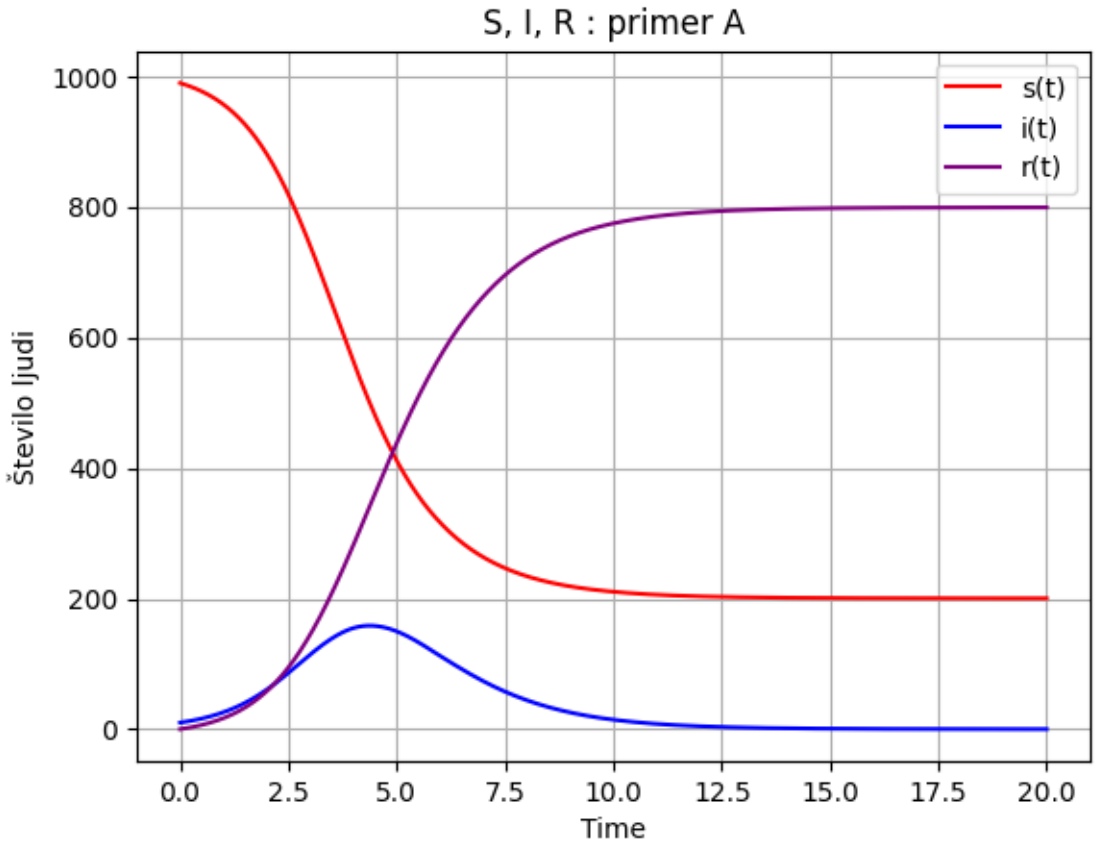
Python funkcija "`epidemija<PODNALOGA>`"je, zapis pripadajočih NDE za vsako podnalogo, tako da jih lahko knjižnica ščipy"uporabi za svoje delovcanje.

Vsaka podnaloga uporablja funkcijo knjižnice ščipy"namreč "`scipy.integrate.solve_ivp`"ta funkcija rešuje sistem NDE z začetnimi vrednostmi. Pomembno za tretjo podnalogo je, da argumente "`args=(n,)`"moramo nujno podati v obliki "`tupple`". Sicer funkcija ne deleuje. Pri podnalogi B smo se ozirali le na cela števila "R_0", čeprav bi lahko to pogledali za več števil.

V drugi podnalogi smo dodali seznama, ki shranjujeta vrednosti $max(I(t)(R_0))$ in R_0, ter ju potem grafirali, za to smo uporabili knjižnico "matplotlib.pyplot"kot "plt". Ukazi in argumenti večinoma pojasnijo sami sebe, edino "`plt.plot(xos, yos, '--bo')`", lahko povemo, da "`'--boo'`", ureja stil, ki je točke s črtano črto. Za preverjanje vsake vrednosti smo ustvarili zanko, ki za vsako vrednost "R_0"izračuna potrebne vrednosti.

V tretji podnalogi r_0 ni več argument funkcije temveč še ena funkcija v sistemu NDE, ki jih rešuje program. Funkcija spreminjanja temperature je bila določena z dvema vrhovoma, namreč podobno, četudi manj izrazito zgleda graf povprečne temperature skozi leto na svetovnem merilu. To je zaradi različnih sezon na obeh hemisfero zemlje, ter so tako "najnižje"temperature pravzaprav spomladi/jeseni. "Najvišje"pa poletji/pozimi.

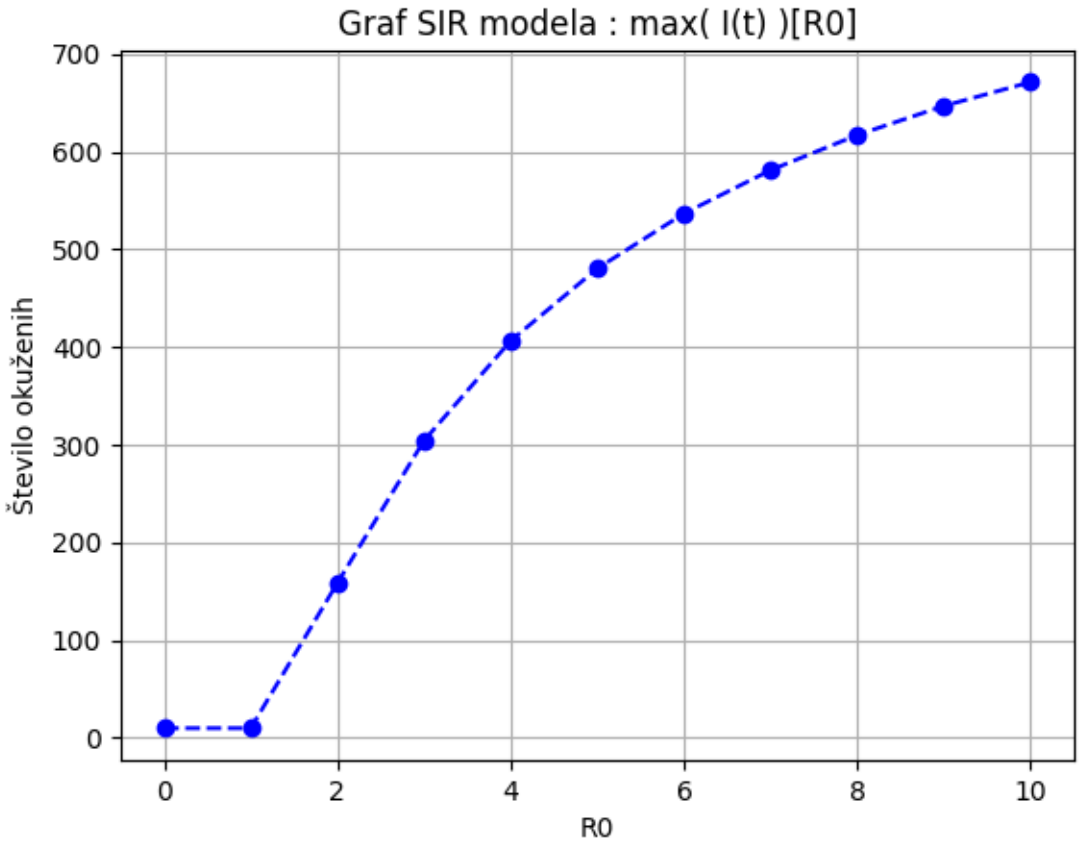
Z drugimi besedami, na primeru za severno hemisfero predvideno število "R_0še poveča pozimi, kajti takrat je zunaj mrzlo in večino časa predvidevamo, da potemtakem ljudje preživijo v notranjosti, kjer se teoretično poveča verjetnost za okužbo, spomladi/jeseni je mešam režim, kjer se ljudje družijo zunaj in znotraj, to nekoliko zmanjša reprodukcijsko število, poletji se pa to število drastično zveča interakcije med ljudmi, ter tudi povečajo se gostota ljudi oboje zunaj in znotraj, kar poveča reprodukcijsko število. Sicer naloga prosi za reprodukcijsko število od temperature, lahko argumentiramo, da je vse zgoraj tudi odvisno od temperature, ter potemtakem lahk tudi zgoraj naštetp upoštevamo. V resnici, če to upoštevamo ali ne, se spremeni samo navpični razteg grafa ter fazni zamik vrhov, od samega temperaturnega grafa (grafa povprečne temperature na zemlji na letni skali).



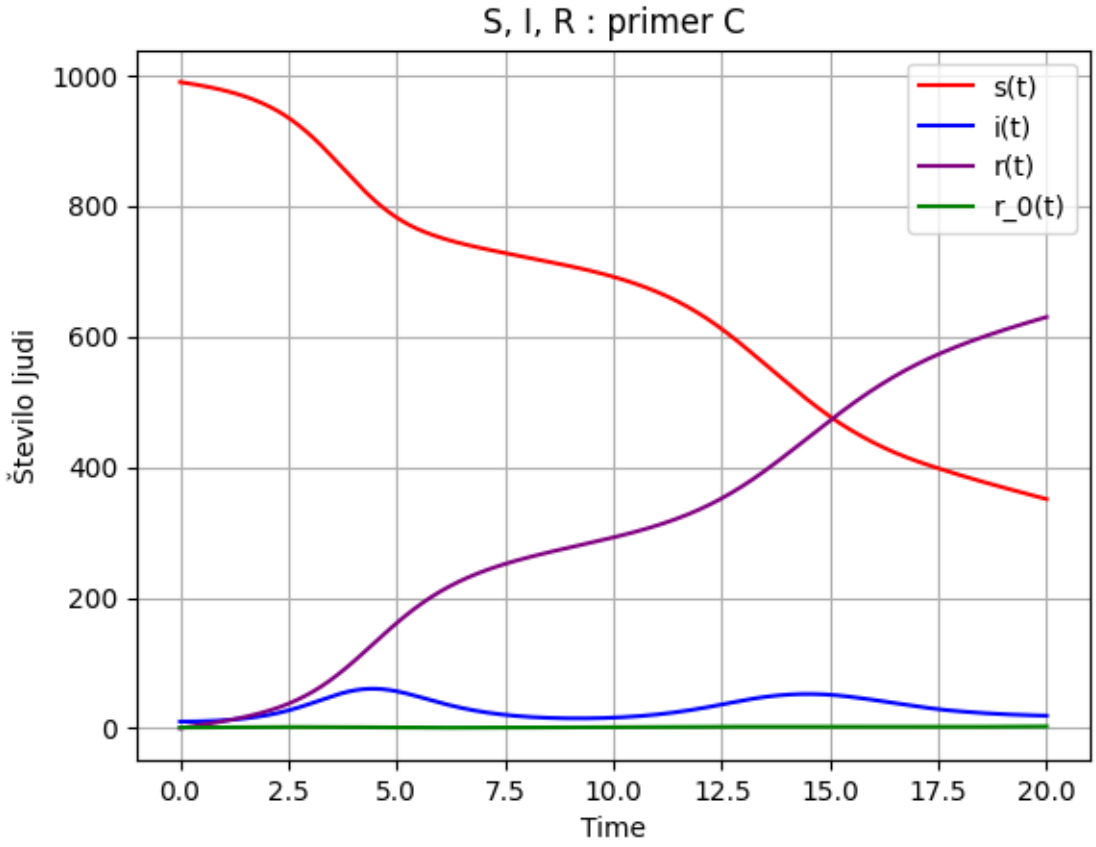
Slika 1: Graf modela SIR(t) za podnalogo A

3 Rezultati programa in analiza podatkov

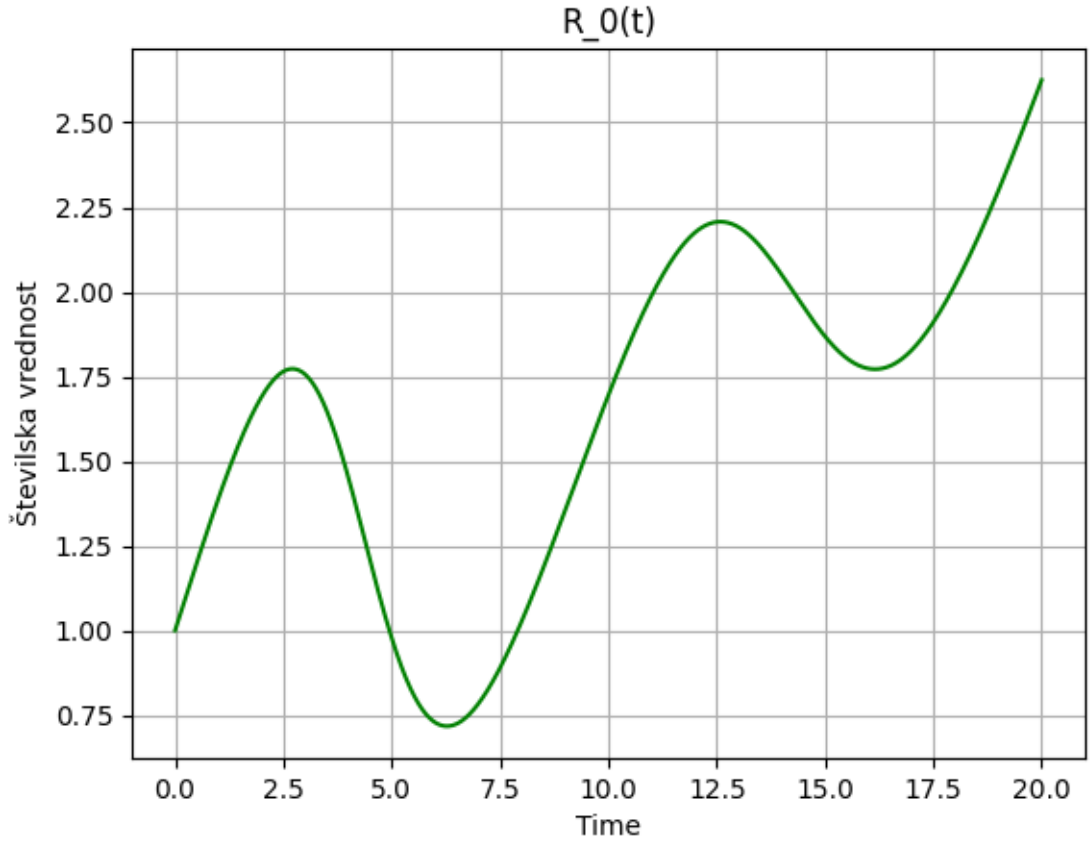
V prvi podnalogi, smo na osnovi podatkov, s programom predvidevali, kakšen bo potek bolezni. Graf (1) prikazuje ta potek. V drugi podnalogi smo dobili graf (2), iz katerega je jasno razvidno, da je največje število okuženih v danem trenutko sorazmerno z reprodukcijskim številom, za reprodukcijsko število večje od ena, ta graf spominja na korensko obliko, vendar to ne pomeni, da je nujno korenska oblika, kajti tega nismo raziskali. Za tretjo podnalogo smo dobili dva grafa, namreč graf (3) in graf (4). Graf (3), prikazuje model $SIR(t)$,tu vidimo, kako izgleda bolezen z dvema vrhovoma oziroma lokalnima maksimuma okuženih. Spet to je sorazmerno z reprodukcijskim številom, torej drži, da je $I(t) \propto R_0(t)$. Graf (4) prikazuje $R_0(t)$, ki prikazuje pojave pojasnjene v prejšnjem odstavku.



Slika 2: Graf modela SIR za podnalogo B : $max(I(t))(R_0)$



Slika 3: Graf modela SIR(t) za podnalogo C



Slika 4: Graf modela SIR za podnalogo C : $R_0(t)$