Universidad de Sonora

División de Ciencias Exactas y Naturales

Licenciatura en Física

Física computacional

2016-1

Actividad 11: Apocalipsis Zombie

Carolina Valenzuela Córdova

1 de Mayo de 2016

1. Descripción de la actividad

Para esta actividad, se nos proporcionó un artículo y un código ejemplo a modificar, de tal manera que nos fuese posible graficar el comportamiento poblacional durante una apocalipsis zombie. A su vez, se incluyen las ecuaciones diferenciales que describen dicho comportamiento

El artículo nos propone cuatro modelos, los cuales consisten en lo siguiente:

1.1. Modelo Básico

En este modelo, los humanos pueden ser convertidos en zombies al perder contra uno. Las ecuaciones propuestas son las siguientes:

$$S' = \Pi - \beta SZ - \delta S$$

$$Z' = \beta SZ - \zeta R - \alpha SZ$$

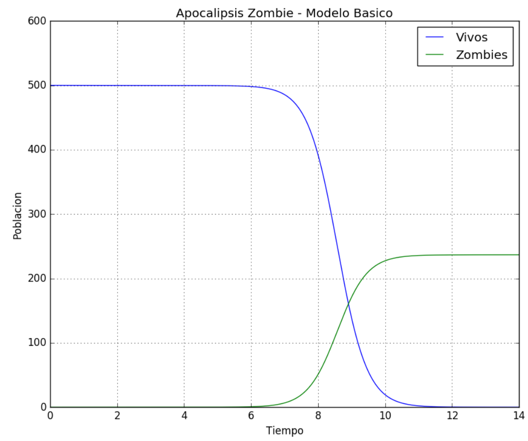
$$R' = \delta S + \alpha SZ - \zeta R$$

El código utilizado fue el siguiente:

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

Pi = 0        #Nacimientos
Del = 0.0001  # Muertes Naturales
```

```
Bet = 0.0095  # Transmisión
Zet = 0.0001  # Removidos
Alf = 0.005   # Destruidos

#Sistemilla de ecuacioncillas diferenciales
def f(y, t):
Si = y[0]
Zi = y[1]
Ri = y[2]
f0 = Pi - Bet*Si*Zi - Del*Si              #Si
f1 = Bet*Si*Zi + Zet*Ri - Alf*Si*Zi       #Zi
f2 = Del*Si + Alf*Si*Zi - Zet*Ri          #Ri
return [f0, f1, f2]

S0 = 500.
Z0 = 0
R0 = 0
y0 = [S0, Z0, R0]
t  = np.linspace(0, 14., 1000)

# Sol
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
#Estúpida gráfica hermosa
plt.figure()
plt.ylim(0,600)
plt.grid(True)
plt.plot(t, S, label='Vivos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Tiempo')
plt.ylabel('Poblacion')
plt.title('Apocalipsis Zombie - Modelo Basico')
plt.legend(loc="best")
```

y esta fue la gráfica que se obtuvo:



1.2. Modelo con infección latente

Aquí se plantea el tiempo que pasa después de que un individuo es mordido por un zombie y se observa de manera más realista la posibilidad de que este se convierta en un individuo de la clase Z después de este tiempo. Las ecuaciones propuestas son las siguientes:

$$S' = \Pi - \beta SZ - \delta S$$

$$I' = \beta SZ - \rho I - \delta I$$

$$Z' = \rho I + \zeta R - \alpha SZ$$

$$R' = \delta S + \delta I + \alpha SZ - \zeta R$$

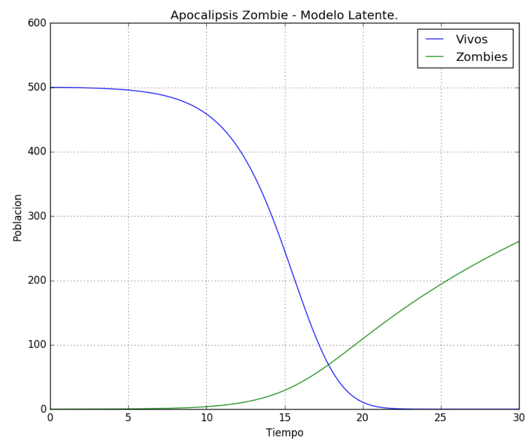
Y el código utilizado fue:

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

Pi = 0          # Nacimientos
Del = 0.0001    # Muertes Naturales
Bet = 0.0095    # Transmisión
Zet = 0.0001    # Removidos
```

```
Alf = 0.0001   # Destruidos
Rho = 0.05     # Infectados

#Sistemilla de ecuacioncillas diferenciales
def f(y, t):
Si = y[0]
Zi = y[1]
Ri = y[2]
Ii = y[3]

f0 = Pi - Bet*Si*Zi - Del*Si                #Si
f1 = Rho*Ii + Zet*Ri - Alf*Si*Zi            #Zi
f2 = Del*Si + Del*Ii + Alf*Si*Zi - Zet*Ri   #Ri
f3 = Bet*Si*Zi -Rho*Ii - Del*Ii             #Ii

return [f0, f1, f2, f3]

S0 = 500.
Z0 = 0.
R0 = 0.
I0 = 1.
y0 = [S0, Z0, R0, I0]
t  = np.linspace(0., 30., 1000)

# Sol
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]
# Estúpida gráfica hermosa
plt.figure()
plt.ylim(0,600)
plt.grid(True)
plt.plot(t, S, label='Vivos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Tiempo')
plt.ylabel('Poblacion')
```

```
plt.title('Apocalipsis Zombie - Modelo Latente.')
plt.legend(loc="best")
```

La gráfica obtenida:



1.3. Modelo con cuarentena

Aquí se simula una cuarentena parcial, durante los zombies son encerrados en un área y no pueden infectar a otros individuos.
En este modelo se considera que en el área de cuarentena solo hay zombies y personas infectadas, además los individuos que intenten escapar serán asesinados o removidos antes de que puedan salir.
Las ecuaciones propuestas son:

$$S' = \Pi - \beta SZ - \delta S$$

$$I' = \beta SZ - \rho I - \delta I - \kappa I$$

$$Z' = \rho I + \zeta R - \alpha SZ - \sigma Z$$

$$R' = \delta S + \delta I + \alpha SZ - \zeta R + \gamma Q$$

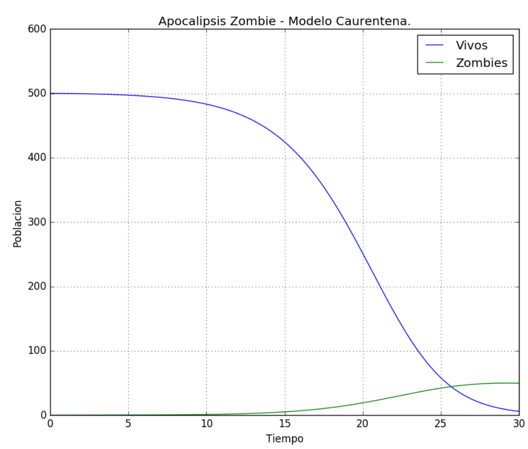
$$Q' = \kappa I + \sigma Z - \gamma Q$$

El código que se utilizó fue el siguiente:

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

Pi = 0          # Nacimientos
Del = 0.0001    # Muertes Naturales
Bet = 0.0095    # Transmisión
Zet = 0.0001    # Removidos
Alf = 0.0001    # Destruidos
Rho = 0.05      # Infectados
Kap = 0.15      # Infectados en cuarentena
Sig = 0.10      # Infectados
Gam = 0.001     # Infectados

#Sistemilla de ecuacioncillas diferenciales
def f(y, t):
Si = y[0]
Zi = y[1]
Ri = y[2]
Ii = y[3]
Qi = y[4]

f0 = Pi - Bet*Si*Zi - Del*Si                        #Si
f1 = Rho*Ii + Zet*Ri - Alf*Si*Zi - Sig*Zi           #Zi
f2 = Del*Si + Del*Ii + Alf*Si*Zi - Zet*Ri + Gam*Qi  #Ri
f3 = Bet*Si*Zi -Rho*Ii - Del*Ii - Kap*Ii            #Ii
f4 = Kap*Ii + Sig*Zi - Gam*Qi                       #Qi
return [f0, f1, f2, f3, f4]

S0 = 500.
Z0 = 0.
R0 = 0.
I0 = 1.
Q0 = 0.
y0 = [S0, Z0, R0, I0, Q0]
```

```
t  = np.linspace(0., 30., 1000)

# Sol
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]
Q = soln[:, 4]
# Estúpida gráfica hermosa
plt.figure()
plt.ylim(0,600)
plt.grid(True)
plt.plot(t, S, label='Vivos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Tiempo')
plt.ylabel('Poblacion')
plt.title('Apocalipsis Zombie - Modelo Caurentena.')
plt.legend(loc="best")
```

Y la gráfica obtenida fue:



1.4. Modelo con tratamiento

Para este modelo se considera que se encontró una cura para combatir a los zombies, y que estos individuos pueden volver a ser humanos.
Teniendo en cuenta esto ya no es necesaria la cuarentena, aunque si bien la cura ayuda a volver a la forma humana, no existe la inmunidad a contraer la infección de nuevo. Las ecuaciones propuestas fueron las siguientes:

$$S' = \Pi - \beta SZ - \delta S + cZS$$

$$I' = \beta SZ - \rho I - \delta I$$

$$Z' = \rho I + \zeta R - \alpha SZ - cZ$$

$$R' = \delta S + \delta I + \alpha SZ - \zeta R$$

El código utilizado:

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

Pi  = 0         # Nacimientos
Del = 0.0001    # Muertes Naturales
Bet = 0.0095    # Transmisión
Zet = 0.0001    # Removidos
Alf = 0.0001    # Destruidos
Rho = 0.05      # Infectados
Ce  = 0.05      # Tratamiento

#Sistemilla de ecuacioncillas diferenciales
def f(y, t):
Si = y[0]
Zi = y[1]
Ri = y[2]
Ii = y[3]

f0 = Pi - Bet*Si*Zi - Del*Si +Ce*Zi             #Si
f1 = Rho*Ii + Zet*Ri - Alf*Si*Zi -Ce*Zi         #Zi
```

```
f2 = Del*Si + Del*Ii + Alf*Si*Zi - Zet*Ri       #Ri
f3 = Bet*Si*Zi -Rho*Ii - Del*Ii                 #Ii

return [f0, f1, f2, f3]

S0 = 500.
Z0 = 0.
R0 = 0.
I0 = 1.
y0 = [S0, Z0, R0, I0]
t  = np.linspace(0., 30., 1000)

# Sol
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]
# Estúpida gráfica hermosa
plt.figure()
plt.ylim(0,500)
plt.grid(True)
plt.plot(t, S, label='Vivos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Tiempo')
plt.ylabel('Poblacion')
plt.title('Apocalipsis Zombie - Modelo Tratamiento.')
plt.legend(loc="best")
```

Índice