# Actividad 11: Ecuaciones diferenciales modelando un apocalipsis zombie

Valenzuela Chaparro Hugo de Jesús

8 de mayo de 2016

## 1. Introducción

Un zombie es un cadáver humano reanimado originado de la creencia espiritual de Vudú en el Caribe. Los seguidores del Vudú creen que un muerto puede ser revivido por una bruja o hechicero con un 'polvo zombie' que contiene una neurotoxina extremadamente poderosa que paraliza temporalmente el sistema nervioso humano y crea un estado de hibernación. Zombies modernos, como los que son ilustrados en películas, libros y videojuegos, son muy diferentes del vudú. Son lentos y muestran señales de descomposición en la piel, ojos y cortadas profundas. Estos zombies frecuentemente son relacionados a un apocalipsis, donde la civilización colapsa debido a la plaga de los muertos vivientes.

## 2. Modelo matemático

Estas figuras populares de la cultura pop generalmente surgen a través del brote de una epidemia. Modelamos un ataque de zombie utilizando suposiciones biológicas basadas en películas populares. En este programa, introducimos un modelo básico para una infección de zombie, determina un equilibrio y estabilidad, e ilustramos el resultado con una solución numérica. Después, el modelo introduce un periodo latente de zombificación, donde los seres humanos son infectados mas no infeccioso, antes de convertirse en zombie. Se modifica el modelo posteriormente para incluir el efecto de una cuarentena o una cura. Finalmente se examina el impacto regular de reducción impulsivo en el numero de zombies y derivamos la condición en la cual la erradicación puede ocurrir. Este experimento nos muestra que solo ataques prontos y agresivos colapsa la sociedad mientras los zombies se apoderan del Día del Juicio Final.

## 3. Modelo básico

En el modelo básico se manejan tres clases:

- Humanos [propensos a infectarse] (S)

- Zombie (Z)
- Removidos [personas muertas por ataques zombies o causas naturales] (R)

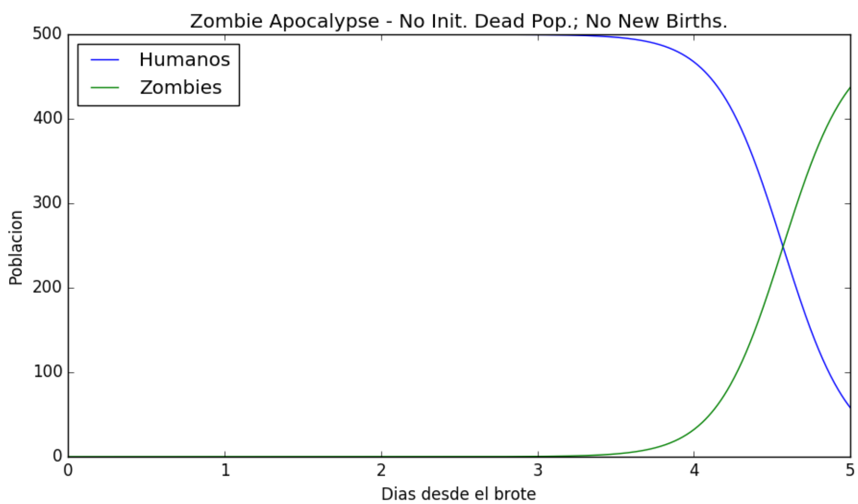
Los nuevos zombies solo pueden venir de los humanos resucitados (grupo R) o de humanos que tuvieron un encuentro perdido con un zombie.

## 3.1. Código Python

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import matplotlib

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 zombie population
R0 = 0                     # initial death population
y0 = [S0, Z0, R0]   # initial condition vector
t  = np.linspace(0, 5., 1000)       # time grid

#Resolver ecuacion diferencial
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
```

```
#Grafica
plt.figure()
plt.plot(t, S, label='Humanos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Dias desde el brote')
plt.ylabel('Poblacion')
plt.title('Zombie Apocalypse - No Init. Dead Pop.; No New Births.')
plt.legend(loc=0)


#guardar grafica
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(10.5,5.5)
fig.savefig('ModeloBasico.png',dpi=100)
```

### 3.2. Gráfica




## 4. Modelo de infección latente

Hay un periodo de tiempo (aproximadamente 24 horas) a partir de que el humano ha sido mordido, antes de que se conviertan en zombie. En este modelo lo extendemos para hacerlo más realista, e incluimos la posibilidad de que un humano se infecte antes de ese periodo de tiempo.
Cambios del modelo básico:

- Los humanos, una vez infectados, primero se clasifican en una clase de infectados (I) y permanecen ahí por un periodo de tiempo.

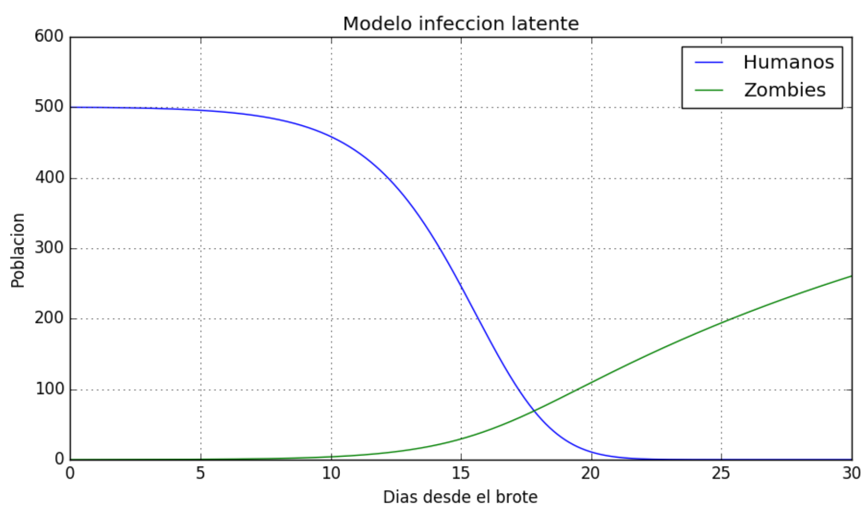
- Los individuos infectados aún pueden morir por causas naturales antes de convertirse en zombies, es decir, mientras están en el periodo de tiempo antes mencionado; de lo contrario, se convierten en muertos vivientes.

### 4.1. Código Python

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import matplotlib

plt.ion()
plt.rcParams['figure.figsize'] = 10, 8

Pi = 0         # birht rate
Del = 0.0001   # Muertes Naturales %  (Por dia)
Bet = 0.0095   # Transmision       %  (Por dia)
Zet = 0.0001   # Resucitados         %  (Por dia)
Alf = 0.0001   # Destruidos        %  (Por dia)
Rho = 0.05     # Infectados          %  (Por dia)

# solve the system dy/dt = f(y, t)
def f(y, t):
    Si = y[0]
    Zi = y[1]
    Ri = y[2]
    Ii = y[3]
    # the model equations (see Munz et al. 2009)
    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]


#condiciones iniciales
S0 = 500.                         # Poblacion Inicial
Z0 = 0.                           # Zombie Inicial
R0 = 0.                           # Muertos Inicial
I0 = 1.                           # Infectados Inicial
y0 = [S0, Z0, R0, I0]             #vector
t  = np.linspace(0., 30., 1000)  # Tiempo

#Resolver ecuacion diferencial
```

```
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]

# Grafica
plt.figure()
plt.ylim(0,600)
plt.grid(True)
plt.plot(t, S, label='Humanos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Dias desde el brote')
plt.ylabel('Poblacion')
plt.title('Modelo infeccion latente')
plt.legend(loc="best")


#guardar grafica
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(10.5,5.5)
fig.savefig('InfeccionLatente.png',dpi=100)
```

### 4.2. Gráfica



# 5. Modelo cuarentena

En este modelo, asumimos que los individuos en cuarentena son removidos de la población y no pueden infectar nuevos humanos mientras permanecen allí. Los cambios con el modelo de la infección latente serían:

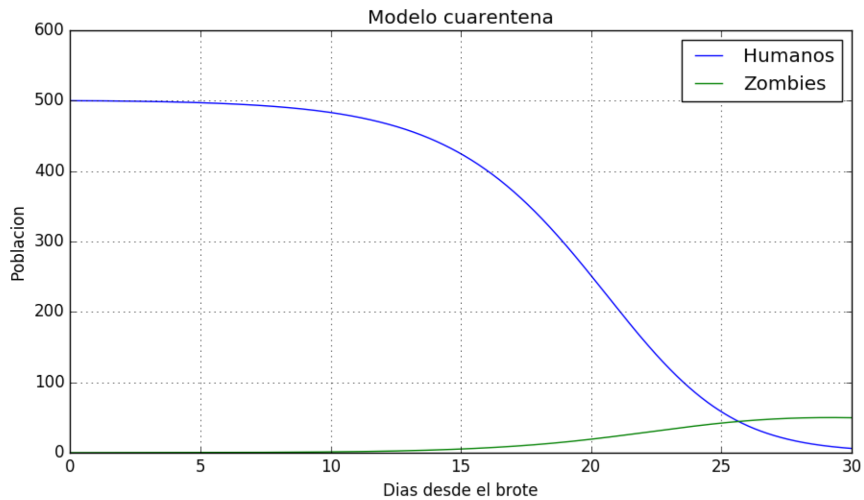
- El área de cuarentena solo contiene miembros de los infectados (I) o de los zombies (Z).
- Hay una posibilidad de que algunos reclusos traten de escapar, pero cualquiera que lo intente es asesinado antes de lograrlo.
- Estos individuos asesinados entran a la clasificación de removidos (R) y podrían después ser reanimados como zombies ”libres”.

## 5.1. Código Python

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import matplotlib

plt.ion()
plt.rcParams['figure.figsize'] = 10, 8

Pi = 0         # Birth rate
Del = 0.0001   # Muertes Naturales %  (Por dia)
Bet = 0.0095   # Transmision       %  (Por dia)
Zet = 0.0001   # Removidos         %  (Por dia)
Alf = 0.0001   # Destruidos        %  (Por dia)
Rho = 0.05     # Infected          %  (Por dia)
Kap = 0.15     # Infectados Q      %  (Por dia)
Sig = 0.10     # Infected          %  (Por dia)
Gam = 0.001    # Infected          %  (Por dia)

# solve the system dy/dt = f(y, t)
def f(y, t):
    Si = y[0]
    Zi = y[1]
    Ri = y[2]
    Ii = y[3]
    Qi = y[4]
    # the model equations (see Munz et al. 2009)
    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]

#condiciones iniciales
S0 = 500.                        # Poblacion Inicial
Z0 = 0.                          # Zombie Inicial
R0 = 0.                          # Muertos Inicial
I0 = 1.                          # Infectados Inicial
Q0 = 0.  # Cuarentena Inicial
y0 = [S0, Z0, R0, I0, Q0]        # vector
t  = np.linspace(0., 30., 1000)  # Tiempo

#Resolver ecuacion diferencial
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]
Q = soln[:, 4]


# Grafica
plt.figure()
plt.ylim(0,600)
plt.grid(True)
plt.plot(t, S, label='Humanos')
plt.plot(t, Z, label='Zombies')
plt.xlabel('Dias desde el brote')
plt.ylabel('Poblacion')
plt.title('Modelo cuarentena')
plt.legend(loc="best")


#guardar grafica
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(10.5,5.5)
fig.savefig('Cuarentena.png',dpi=100)
```

## 5.2. Gráfica




# 6. Modelo con cura

Si producimos una cura podremos regresar a un individuo de zombie a humano, sin embargo, esto no da la inminidad, pues estaría propenso a infectarse de nuevo. Los zombies que resucitaron de la muerta y se les dio una cura, pudieron volver al estado humano de nuevo. Cambios a considerar:

- Ahora que hay cura, no se ocupa cuarentena.
- La cura permitirá a los muertos vivientes ser humanos vivos, independientemente de cómo se convirtieron en zombies.
- Como lo mencionamos, cualquier zombie curado podrá infectarse de nuevo, no hay inmunidad.

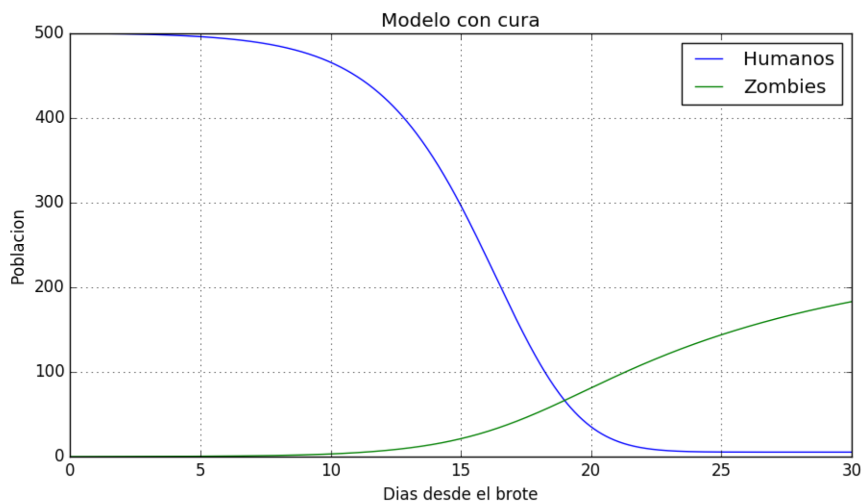
## 6.1. Código Python

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import matplotlib


plt.ion()
plt.rcParams['figure.figsize'] = 10, 8
```

```
Pi  = 0        # birth rate
Del = 0.0001   # Muertes Naturales %  (Por dia)
Bet = 0.0095   # Transmision       %  (Por dia)
Zet = 0.0001   # Removidos         %  (Por dia)
Alf = 0.0001   # Destruidos        %  (Por dia)
Rho = 0.05     # Infected          %  (Por dia)
Ce  = 0.05     # Cura              %  (Por dia)

# solve the system dy/dt = f(y, t)
def f(y, t):
    Si = y[0]
    Zi = y[1]
    Ri = y[2]
    Ii = y[3]
    # the model equations (see Munz et al. 2009)
    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]


#condiciones iniciales
S0 = 500.                         # Poblacion Inicial
Z0 = 0.                           # Zombie Inicial
R0 = 0.                           # Muertos Inicial
I0 = 1.                           # Infectados Inicial
y0 = [S0, Z0, R0, I0]             # vector
t  = np.linspace(0., 30., 1000)   # Tiempo

#Resolver ecuacion dierencial
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]


# Grafica
plt.figure()
plt.ylim(0,500)
plt.grid(True)
plt.plot(t, S, label='Humanos')
plt.plot(t, Z, label='Zombies')
```

```
plt.xlabel('Dias desde el brote')
plt.ylabel('Poblacion')
plt.title('Modelo con cura')
plt.legend(loc="best")


#guardar grafica
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(10.5,5.5)
fig.savefig('Cura.png',dpi=100)
```

### 6.2. Gráfica




## 7. Modelo con erradicación impulsiva

Finalmente, intentamos controlar la población de zombies destruyéndolos estratégicamente tantas veces como nuestros recursos nos lo permitan. Se asume que será dificil tener recursos y cordinación, así que se ocupará atacar más de una vez, tratando de destruir más zombies con cada ataque; esto resulta en un efecto impulsivo.

### 7.1. Código Pyton

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
```

```
import matplotlib

plt.ion()
plt.rcParams['figure.figsize'] = 10, 8

Pi  = 0        # birth rate
Del = 0.0001   # Muertes Naturales %  (Por dia)
Bet = 0.0055   # Transmision       %  (Por dia)
Zet = 0.0900   # Removidos         %  (Por dia)
Alf = 0.0075   # Destruidos        %  (Por dia)
k = 0.25
n=4

# 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 = 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
    f3 = -k*n*Zi                                  #DZi

    return [f0, f1, f2, f3]

#condiciones iniciales
S0 = 500.                         # Poblacion Inicial
Z0 = 0.                           # Zombie Inicial
R0 = 0.                           # Muertos Inicial
DZ0 = 0.                           # Infectados Inicial
y0 = [S0, Z0, R0, DZ0]             # vector
t  = np.linspace(0., 130., 1000)  # Tiempo

#Resolver ecuacion diferencial
soln = odeint(f, y0, t)
S = soln[:, 0]
Z = soln[:, 1]
R = soln[:, 2]
I = soln[:, 3]

# grafica
plt.figure()
plt.ylim(0,500)
plt.grid(True)
plt.plot(t, S, label='Humanos')
```

```
plt.plot(t, Z, label='Zombies')
plt.xlabel('Dias desde el brote')
plt.ylabel('Poblacion')
plt.title('Modelo de erradicacion impulsiva')
plt.legend(loc="best")


#guardar grafica
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(10.5,5.5)
fig.savefig('ErradicacionImpulsiva.png',dpi=100)
```

## 7.2. Gráfica