

# El modelo epidémico de SIR

Una descripción matemática simple de la propagación de una enfermedad en una población es el llamado modelo SIR, que divide la población (fija) de  $N$  individuos en tres "compartimentos" que pueden variar en función del tiempo,  $t$ :

- $S(t)$  son aquellos susceptibles pero aún no infectados con la enfermedad;
- $I(t)$  es el número de individuos infecciosos;
- $R(t)$  son aquellas personas que se han recuperado de la enfermedad y ahora tienen inmunidad.

El modelo SIR describe el cambio en la población de cada uno de estos compartimentos en términos de dos parámetros, beta y gamma.

- Beta describe la tasa de contacto efectiva de la enfermedad: un individuo infectado entra en contacto con  $\beta N$  otros individuos por unidad de tiempo (de los cuales la fracción que es susceptible a contraer la enfermedad es  $S/N$ ).
- Gamma es la tasa de recuperación promedio: es decir,  $1/\gamma$  es el período de tiempo promedio durante el cual una persona infectada puede transmitirlo.

Las ecuaciones diferenciales que describen este modelo fueron derivadas primero por Kermack y McKendrick [ Proc. R. Soc. A , 115 , 772 (1927)]:

$$\begin{aligned}\frac{dS}{dt} &= -\frac{\beta SI}{N}, \\ \frac{dI}{dt} &= \frac{\beta SI}{N} - \gamma I, \\ \frac{dR}{dt} &= \gamma I.\end{aligned}$$

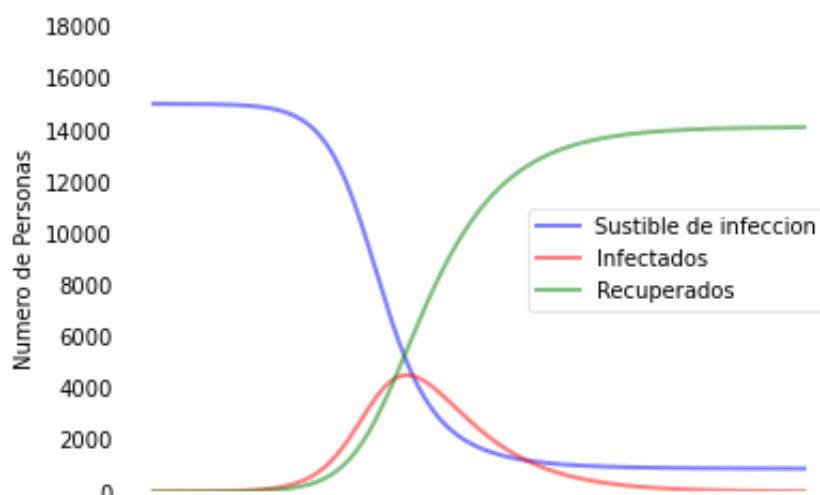
El siguiente código de Python integra estas ecuaciones para una enfermedad caracterizada por los parámetros  $\beta=0.2$ ,  $\gamma=10$  en una población de  $N=1000$  (quizás 'gripe en una escuela') El modelo se inicia con una sola persona infectada el día 0:  $I(0)=1$ . Las curvas trazadas de  $S(t)$ ,  $I(t)$  y  $R(t)$  están diseñadas para verse un poco mejor que los valores predeterminados de Matplotlib.

```
In [59]: 1 #Importar las librerías.
          2 import numpy as np
          3 from scipy.integrate import odeint
          4 import matplotlib.pyplot as plt
          5
          6 # Total de la población
          7 N = 15000
          8 # Numero Inicial de Infectados
          9 I0 = 1
         10 # Numero de Recuperados
         11 R0 = 0
         12 # Todos los demás, S0, son susceptibles a la infección inicialm
         13 S0 = N - I0 - R0
```

```

14 # Tasa de contacto, beta (nivel de reproductividad del virus)
15 # La tasa de recuperación media, gamma,(1/días) Una persona se
16 beta, gamma = 0.2, 1.0/15
17 # Una cuadrícula de puntos de tiempo (en días)
18 t = np.linspace(0, 200, 200)
19
20
21 # Las ecuaciones diferenciales del modelo SIR..
22 def deriv(y, t, N, beta, gamma):
23     S, I, R = y
24     dSdt = -beta * S * I / N
25     dIdt = beta * S * I / N - gamma * I
26     dRdt = gamma * I
27     return dSdt, dIdt, dRdt
28
29 # Vector de condiciones iniciales
30 y0 = S0, I0, R0
31 # Integre las ecuaciones SIR en la cuadrícula de tiempo, t. A t
32 ret = odeint(deriv, y0, t, args=(N, beta, gamma))
33 S, I, R = ret.T # Obtencion de resultados
34
35
36 # Trace los datos en tres curvas separadas para S (t), I (t) y
37 fig = plt.figure(facecolor='w')
38 ax = fig.add_subplot(111, axisbelow=True)
39 ax.plot(t, S, 'b', alpha=0.5, lw=2, label='Sustible de infeccio
40 ax.plot(t, I, 'r', alpha=0.5, lw=2, label='Infectados')
41 ax.plot(t, R, 'g', alpha=0.5, lw=2, label='Recuperados')
42 ax.set_xlabel('Tiempo en dias')
43 ax.set_ylabel('Numero de Personas')
44 ax.set_ylim(0,N*1.2)
45 ax.yaxis.set_tick_params(length=0)
46 ax.xaxis.set_tick_params(length=0)
47 ax.grid(b=True, which='major', c='w', lw=2, ls='-')
48 legend = ax.legend()
49 legend.get_frame().set_alpha(0.5)
50 for spine in ('top', 'right', 'bottom', 'left'):
51     ax.spines[spine].set_visible(False)
52 plt.show()
53
54 #Ro = beta/gamma
55 #print(Ro)
56

```



## Generar la predicción del modelo SIR

Se debe estimar el valor de

- $\beta$
- $\gamma$

Para ajustar el modelo SIR con los casos confirmados reales (el número de personas infecciosas) del Ecuador.

Para ello deben seguir el siguiente tutorial <https://www.lewuathe.com/covid-19-dynamics-with-sir-model.html> (<https://www.lewuathe.com/covid-19-dynamics-with-sir-model.html>)

```
In [18]: 1 import numpy as np
2 from time import time
3 import pandas as pd
4 from datetime import datetime, timedelta
5 from scipy.optimize import minimize
6 from scipy.integrate import solve_ivp
7 from scipy.integrate import odeint
```

```
In [32]: 1 link_Confirmed = 'https://raw.githubusercontent.com/Lewuathe/COVID-19-dynamics-with-sir-model/master/data/confirmed.csv'
2 link_Deaths = 'https://raw.githubusercontent.com/Lewuathe/COVID-19-dynamics-with-sir-model/master/data/deaths.csv'
3 link_Recovered = 'https://raw.githubusercontent.com/Lewuathe/COVID-19-dynamics-with-sir-model/master/data/recovered.csv'
```

```
In [93]: 1 START_DATE = {'Ecuador': '3/20/20'}
```

```
In [125]: 1 # Implementar y explicar la predicción del modelo SIR para el Ecuador
2
3 # 1. Implementar solo teniendo en cuenta los casos confirmados
4
5 print("Start time: "+str(datetime.now()))
6
7 def fx(point, data):
8     size = len(data)
9     beta, gamma = point
10     def SIR(t, y):
11         S = y[0]
12         I = y[1]
13         R = y[2]
14         return [-beta*S*I, beta*S*I-gamma*I, gamma*I]
15     solution = solve_ivp(SIR, [0, size], [S0, I0, R0], t_eval=np.linspace(0, size, 100))
16     return np.sqrt(np.mean((solution.y[1] - data)**2))
17
18
19 df = pd.read_csv(link_Recovered)
20 country_df = df[df['Country/Region'] == country]
21 recovered = country_df.iloc[0].loc[START_DATE[country]:]
22
23
```

```

23
24 df = pd.read_csv(link_Deaths)
25 country_df = df[df['Country/Region'] == country]
26 death = country_df.iloc[0].loc[START_DATE[country]:]
27
28 df = pd.read_csv(link_Confirmed)
29 country_df = df[df['Country/Region'] == country]
30 confirmed = country_df.iloc[0].loc[START_DATE[country]:]
31
32 data = (confirmed - death)
33
34 # Total de la poblacion
35 #N = 193582
36 N = 100000
37 # Numero Inicial de Infectados
38 I0 = 1
39 # Numero de Recuperados
40 R0 = 0
41 # Todos los demás, S0, son susceptibles a la infección inicial
42 S0 = N - I0 - R0
43
44
45 fxmin = minimize(fx, [0.001, 0.001], args=(data), method='L-BFGS')
46
47 beta, gamma = fxmin.x
48
49 print("Beta: %0.10f " % beta)
50 print("Gamma: %0.10f " % gamma)
51
52 beta *= 10000
53 #gamma *= 100000
54 # Tasa de contacto, beta (nivel de reproductividad del virus)
55 # La tasa de recuperación media, gamma, (1/días) Una persona se
56 #beta, gamma = 0.589, 0.045
57 # Una cuadrícula de puntos de tiempo (en días)
58 t = np.linspace(0, 200, 200)
59
60 # Las ecuaciones diferenciales del modelo SIR..
61 def deriv(y, t, N, beta, gamma):
62     S, I, R = y
63     dSdt = -beta * S * I / N
64     dIdt = beta * S * I / N - gamma * I
65     dRdt = gamma * I
66     return dSdt, dIdt, dRdt
67
68 # Vector de condiciones iniciales
69 y0 = S0, I0, R0
70 # Integre las ecuaciones SIR en la cuadrícula de tiempo, t. A
71 ret = odeint(deriv, y0, t, args=(N, beta, gamma))
72 S, I, R = ret.T # Obtencion de resultados
73
74 # Trace los datos en tres curvas separadas para S (t), I (t) y
75 fig = plt.figure(facecolor='w')
76 ax = fig.add_subplot(111, axisbelow=True)
77 ax.plot(t, S, 'b', alpha=0.5, lw=2, label='Susceptible de infección')
78 ax.plot(t, I, 'r', alpha=0.5, lw=2, label='Infectados')
79 ax.plot(t, R, 'g', alpha=0.5, lw=2, label='Recuperados')
80 ax.plot(range(len(confirmed)), confirmed, 'black', alpha=0.5,

```

```

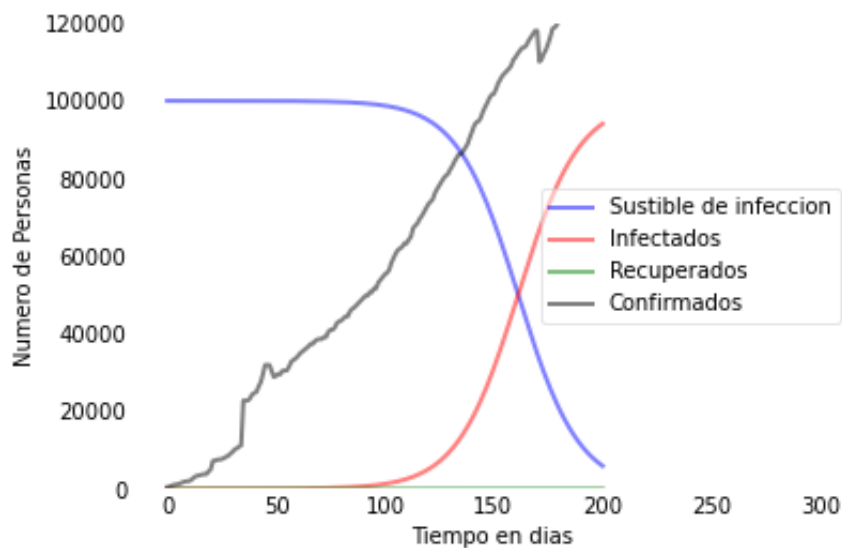
81 ax.set_xlabel('Tiempo en dias')
82 ax.set_ylabel('Numero de Personas')
83 ax.set_ylim(0,N*1.2)
84 ax.yaxis.set_tick_params(length=0)
85 ax.xaxis.set_tick_params(length=0)
86 ax.grid(b=True, which='major', c='w', lw=2, ls='--')
87 legend = ax.legend()
88 legend.get_frame().set_alpha(0.5)
89 for spine in ('top', 'right', 'bottom', 'left'):
90     ax.spines[spine].set_visible(False)
91 elapsed_time = (time() - start_time)/60
92 plt.show()
93
94
95 fig.savefig(f"Ecuador {datetime.now()}.png")
96
97 #Ro = beta/gamma
98 #print(Ro)
99
100 print("Elapsed time: %0.10f seconds." % elapsed_time)
101
102

```

Start time: 2021-08-06 02:51:29.514395

Beta: 0.0000071428

Gamma: 0.0000071428



Elapsed time: 53.1945629517 seconds.

In [126]:

```

1  # 2. Implementar teniendo en cuenta los casos confirmados y rec
2
3  # 2. Implementar teniendo en cuenta los casos confirmados y rec
4  print("Start time: "+str(datetime.now()))
5
6  def fx(point, data, recovered, s_0, i_0, r_0):
7      size = len(data)
8      beta, gamma = point
9      def SIR(t, y):
10         S = y[0]
11         I = y[1]
12         R = y[2]

```

```

13         return [-beta*S*I, beta*S*I-gamma*I, gamma*I]
14     solution = solve_ivp(SIR, [0, size], [s_0,i_0,r_0], t_eval=
15         l1 = np.sqrt(np.mean((solution.y[1] - data)**2))
16         l2 = np.sqrt(np.mean((solution.y[2] - recovered)**2))
17         alpha = 0.1
18         return alpha * l1 + (1 - alpha) * l2
19
20 df = pd.read_csv(link_Recovered)
21 country_df = df[df['Country/Region'] == country]
22 recovered = country_df.iloc[0].loc[START_DATE[country]:]
23
24
25 df = pd.read_csv(link_Deaths)
26 country_df = df[df['Country/Region'] == country]
27 death = country_df.iloc[0].loc[START_DATE[country]:]
28
29 df = pd.read_csv(link_Confirmed)
30 country_df = df[df['Country/Region'] == country]
31 confirmed = country_df.iloc[0].loc[START_DATE[country]:]
32
33
34 data = (confirmed - recovered - death)
35
36 # Total de la poblacion
37 #N = 193582
38 N = 100000
39 #N =15000
40 # Numero Inicial de Infectados
41 I0 = 1
42 # Numero de Recuperados
43 R0 = 0
44 # Todos los demás, S0, son susceptibles a la infección inicial
45 S0 = N - I0 - R0
46
47
48 fxmin = minimize(fx, [0.001, 0.001], args=(data, recovered, S0,
49
50 beta, gamma = fxmin.x
51
52 print("Beta: %0.10f " % beta)
53 print("Gamma: %0.10f " % gamma)
54
55 beta *= 100000/2
56 #gamma *= 100000
57 # Tasa de contacto, beta (nivel de reproductividad del virus)
58 # La tasa de recuperación media, gamma,(1/días) Una persona se
59 #beta, gamma = 0.589,0.045
60 # Una cuadrícula de puntos de tiempo (en días)
61 t = np.linspace(0, 200, 200)
62
63 # Las ecuaciones diferenciales del modelo SIR..
64 def deriv(y, t, N, beta, gamma):
65     S, I, R = y
66     dSdt = -beta * S * I / N
67     dIdt = beta * S * I / N - gamma * I
68     dRdt = gamma * I
69     return dSdt, dIdt, dRdt
70

```

```

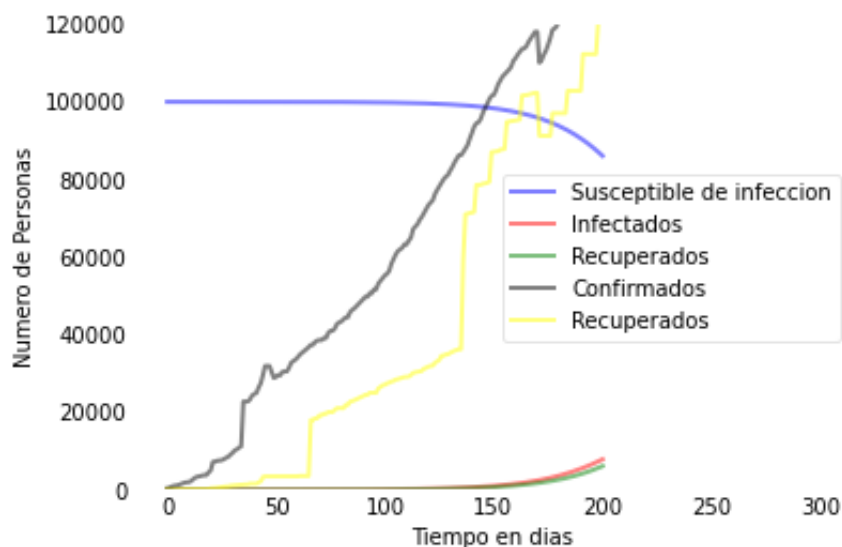
71 # Vector de condiciones iniciales
72 y0 = S0, I0, R0
73 # Integre las ecuaciones SIR en la cuadrícula de tiempo, t. A t
74 ret = odeint(deriv, y0, t, args=(N, beta, gamma))
75 S, I, R = ret.T # Obtencion de resultados
76
77
78 # Trace los datos en tres curvas separadas para S (t), I (t) y
79 fig = plt.figure(facecolor='w')
80 ax = fig.add_subplot(111, axisbelow=True)
81 ax.plot(t, S, 'b', alpha=0.5, lw=2, label='Susceptible de infecc
82 ax.plot(t, I, 'r', alpha=0.5, lw=2, label='Infectados')
83 ax.plot(t, R, 'g', alpha=0.5, lw=2, label='Recuperados')
84 ax.plot(range(len(confirmed)), confirmed, 'black', alpha=0.5,
85 ax.plot(range(len(confirmed)), recovered, 'yellow', alpha=0.5,
86 ax.set_xlabel('Tiempo en dias')
87 ax.set_ylabel('Numero de Personas')
88 ax.set_ylim(0, N*1.2)
89 ax.yaxis.set_tick_params(length=0)
90 ax.xaxis.set_tick_params(length=0)
91 ax.grid(b=True, which='major', c='w', lw=2, ls='-')
92 legend = ax.legend()
93 legend.get_frame().set_alpha(0.5)
94 for spine in ('top', 'right', 'bottom', 'left'):
95     ax.spines[spine].set_visible(False)
96 elapsed_time = (time() - start_time)/60
97 plt.show()
98
99 fig.savefig("Confirmados-y-Recuperados.png")
100
101 #Ro = beta/gamma
102 #print(Ro)
103
104 print("Elapsed time: %0.10f seconds." % elapsed_time)
105
106

```

Start time: 2021-08-06 02:52:04.525529

Beta: 0.0000015545

Gamma: 0.0315953595



Elapsed time: 50.2044500204 seconds

Elapsed time: 38.3844509204 seconds.

## Calculos de incidencia

Para obtener metricas de incidencia se debe calcular la tasa de prevalencia, incidencia y la relacion, para esto leer y obtener estos datos con la ultima lectura.

[https://www.paho.org/hq/index.php?option=com\\_content&view=article&id=14402:indicadores-de-salud-aspectos-conceptuales-y-operativos-seccion-2&catid=9894&limitstart=2&Itemid=101&lang=es](https://www.paho.org/hq/index.php?option=com_content&view=article&id=14402:indicadores-de-salud-aspectos-conceptuales-y-operativos-seccion-2&catid=9894&limitstart=2&Itemid=101&lang=es)  
([https://www.paho.org/hq/index.php?option=com\\_content&view=article&id=14402:indicadores-de-salud-aspectos-conceptuales-y-operativos-seccion-2&catid=9894&limitstart=2&Itemid=101&lang=es](https://www.paho.org/hq/index.php?option=com_content&view=article&id=14402:indicadores-de-salud-aspectos-conceptuales-y-operativos-seccion-2&catid=9894&limitstart=2&Itemid=101&lang=es))

In [127]:

```
1 # Implementar
2 print("Tasa de incidencia: "+str((confirmed[len(confirmed)-1]/N
3 print("Tasa de prevalencia: "+str((confirmed[len(confirmed)-1]/
4 print("Relacion: "+str(((confirmed[len(confirmed)-1]/N)*np.powe
```

Tasa de incidencia: 22.4315

Tasa de prevalencia: 129.90213110956682

Relacion: 672.9449999999999



## Analisis

El analisis de datos para una epidemia puede resultar muy complejo sobre todo cuando el numero de datos crece respecto a los días que pasa. El metodo SIR trata de predecir cuando tiempo tendra que pasar hasta que una poblacion se infecte con el patogeno, asi mismo, aunque cabe recalcar que en este metodo no se tienen en cuenta las medidas que puede aplicar la autoridad de dicha localidad o una posible vacuna.

La tasa de incidencia se define como el número de casos nuevos de una enfermedad u otra condición de salud dividido por la población en riesgo de la enfermedad (población expuesta) en un lugar específico y durante un período específico.

La tasa de prevalencia se define como el número de casos existentes de una enfermedad u otro evento de salud dividido por el número de personas de una población en un período específico.

La tasa de prevalencia de una enfermedad (u otro trastorno) es directamente proporcional al producto de su tasa de incidencia por la duración media de la enfermedad.

## Conclusiones

Como se puede observar en los modelos se predice los modelos los valores de confirmados y recuperados siguen en incremento por la recta que tiene, con unos datos de 10000, ya que no es el proceso para todos los datos obtenidos tiene un tiempo muy largo de ejecución. El gamma y betta cambiarian de acuerdo a os datos que se obtengan.

## Opinion

Con respecto a los datos se obtuvo los mas actualizados posibles. Los graficos nos demuestran que seguiremos con la misma curva , pero dado la situacion del pais esta se espera que la curva llegue a su pico en unas semanas o meses.

## Referencias:

- <https://www.agenciasinc.es/Reportajes/Un-modelo-un-teorema-y-teoria-de-juegos-contra-el-coronavirus> (<https://www.agenciasinc.es/Reportajes/Un-modelo-un-teorema-y-teoria-de-juegos-contra-el-coronavirus>)
- <https://rpubs.com/dsfernandez/422937> (<https://rpubs.com/dsfernandez/422937>)
- <https://towardsdatascience.com/modelling-the-coronavirus-epidemic-spreading-in-a-city-with-python-babd14d82fa2> (<https://towardsdatascience.com/modelling-the-coronavirus-epidemic-spreading-in-a-city-with-python-babd14d82fa2>)

