Modelo SIR con Ecuaciones Diferenciales

D. S. Fernandez del Viso

2020.10.14

# Ecuaciones diferenciales para el modelo SIR

Las ecuaciones básicas del modelo SIR (sin demografía, o nacimientos iguales a muertes), para cada compartimento, son las siguientes:

## Descripción y parámetros del modelo

* El modelo SIR asume el contacto (directo o mediante vector) entre individuos susceptibles (*X*) e infectados (*Y*), lo cual significa una *transmisión por acción de masas*, y por lo tanto es dependiente de la frecuencia de cada grupo actuando (susceptibles e infectados). Por esta razón, y para estandarizar el modelo para cualquier tamaño poblacional (*N*), utilizamos las proporciones de cada grupo:

Igual lo hacemos para los recuperados en el modelo (*R*):

* El parámetro es el producto de la tasa de contacto y la probabilidad de transmisión de la infección. También se utiliza la expresión para indicar la *fuerza de infección*.
* El parámetro se denomina *tasa de remoción o recuperación* (usaremos recuperación); a partir de datos epidemiológicos se puede obtener el *período infeccioso*, que es el recíproco .
* Un parámetro derivado de los anteriores, es uno de los números más famosos de la epidemiología es la *razón reproductiva básica*, , que se cuantifica como el número promedio de casos secundarios que se derivan de un caso primario promedio, en una población totalmente susceptible.
* Podemos demostrar que cuando la proporción de susceptibles iniciales, , es menor que , la *tasa relativa de remoción*, la infección no procede (esto se conoce como el umbral de infección).
* El inverso de , por otra parte, expresa la *razón reproductiva básica*, .

## Ejercicios con el modelo SIR y ecuaciones diferenciales en R

### Uso de deSolve en R

* El paquete [**deSolve**](http://desolve.r-forge.r-project.org/) en R permite resolver sistemas de *ecuaciones diferenciales ordinarias* (ODE), asi como otras, cuando se conocen las condiciones iniciales.

#### Resolución de ecuaciones del modelo SIR

library(deSolve)  
#tamaño poblacional  
N = 1  
#estado inicial de los compartimentos  
init <- c(S = 1-1e-6,  
 I = 1e-6,  
 R = 0)  
#parámetros del modelo (coeficientes de las variables)  
param <- c(beta = 2.0,  
 gamma = 0.2)  
#crear la función con las ODE  
sir <- function(times, init, param) {  
 with(as.list(c(init, param)), {  
#ecuaciones diferenciales   
 dS <- -beta \* S \* I  
 dI <- beta \* S \* I - gamma \* I  
 dR <- gamma \* I  
#resultados de las tasas de cambio   
 return(list(c(dS, dI, dR)))  
 })  
}  
#intervalo de tiempo y resolución  
times <- seq(0, 20, by = .1)  
#resolver el sistema de ecuaciones con función 'ode'  
out <- ode(y = init, times = times, func = sir, parms = param)  
#cambiar out a un data.frame  
out <- as.data.frame(out\*N) #aqui puede multiplicar 'out' por N  
# ver resultados  
head(out)

## time S I R  
## 1 0.0 0.9999990 1.000000e-06 0.000000e+00  
## 2 0.1 0.9999988 1.197789e-06 2.197659e-08  
## 3 0.2 0.9999985 1.434672e-06 4.829699e-08  
## 4 0.3 0.9999982 1.718390e-06 7.982122e-08  
## 5 0.4 0.9999978 2.058215e-06 1.175796e-07  
## 6 0.5 0.9999974 2.465242e-06 1.628050e-07

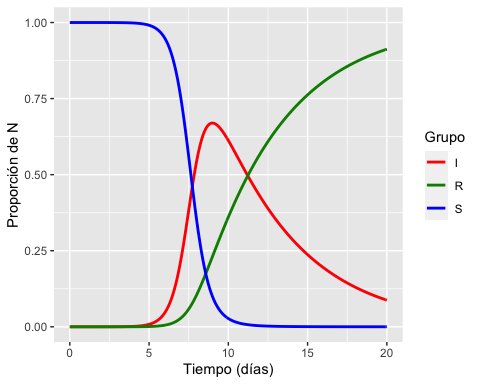
A continuación vamos a rearreglar los datos de los resultados (**out**), usando el paquete **tidyverse**. Esto facilita otros análisis y la gráficación de los resultados.

library(tidyverse)  
new.out <- as.data.frame(out) %>% gather(key, value, -time)   
head(new.out)

## time key value  
## 1 0.0 S 0.9999990  
## 2 0.1 S 0.9999988  
## 3 0.2 S 0.9999985  
## 4 0.3 S 0.9999982  
## 5 0.4 S 0.9999978  
## 6 0.5 S 0.9999974

En este nuevo objeto, *new.out*, la variable *key* contiene los grupos , , y , y *value* el respectivo valor del grupo en ese tiempo (*time*).

Ahora construiremos una gráfica usando el paquete **ggplot2**:



Un dato importante es el valor máximo de infectados:

# valor máximo de I y tiempo usando un pipe  
new.out %>%   
 filter(key=="I") %>%   
 filter(value==max(value)) %>%   
 mutate(maxI = round(N \* value, 2)) %>%  
 select(time, maxI)

## time maxI  
## 1 9 0.67

# valor de S\_umbral  
out$S[which.max(out$I)]

## [1] 0.09936161

## Modelo SIR para sarampión en Humacao

**El problema**: si se declara una epidemia de sarampión en Humacao, es muy probable no contar con las camas suficientes para los enfermos. Hay que comprobarlo.

*Datos a buscar*:

* población de Humacao
* número de camas en hospitales
* parámetros del sarampión

*Usar el modelo SIR en R*.

*Proponer soluciones*.

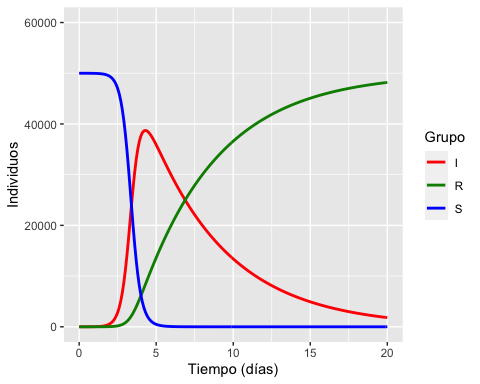
library(deSolve)  
#tamaño poblacional  
N = 50000  
#estado inicial de los compartimentos  
init <- c(S = 1-1/50000,  
 I = 1/50000,  
 R = 0)  
#parámetros del modelo (coeficientes de las variables)  
param <- c(beta = 3.4,  
 gamma = 0.2)  
#crear la función con las ODE  
sir <- function(times, init, param) {  
 with(as.list(c(init, param)), {  
#ecuaciones diferenciales   
 dS <- -beta \* S \* I  
 dI <- beta \* S \* I - gamma \* I  
 dR <- gamma \* I  
#resultados de las tasas de cambio   
 return(list(c(dS, dI, dR)))  
 })  
}  
#intervalo de tiempo y resolución  
times <- seq(0, 20, by = .1)  
#resolver el sistema de ecuaciones con función 'ode'  
out <- ode(y = init, times = times, func = sir, parms = param)  
#cambiar out a un data.frame  
out <- as.data.frame(out)  
head(out, 3)

## time S I R  
## 1 0.0 0.9999800 2.000000e-05 0.000000e+00  
## 2 0.1 0.9999719 2.761384e-05 4.758773e-07  
## 3 0.2 0.9999608 3.811587e-05 1.132278e-06

#rearreglo del output  
new.out <- as.data.frame(out) %>%  
 gather(key, value, -time) %>%  
 mutate(value=value\*N) #de proporción a individuos  
head(new.out, 3)

## time key value  
## 1 0.0 S 49999.00  
## 2 0.1 S 49998.60  
## 3 0.2 S 49998.04

#gráfica conjunta  
ggplot(data=new.out,   
 aes(x = time,  
 y = value,  
 group = key,  
 col = key  
 )) +   
 ylab("Individuos") + xlab("Tiempo (días)") +  
 geom\_line(size = 1) +   
 scale\_colour\_manual(values = c("red", "green4", "blue"), name = "Grupo") +  
 scale\_y\_continuous(labels = waiver(), limits = c(0, 60000))



# nivel máximo de infección  
new.out %>%   
 filter(key=="I") %>%   
 filter(value==max(value)) %>%   
 mutate(maxI = round(value, 0)) %>%  
 select(time, maxI)

## time maxI  
## 1 4.3 38726

### Vacunación

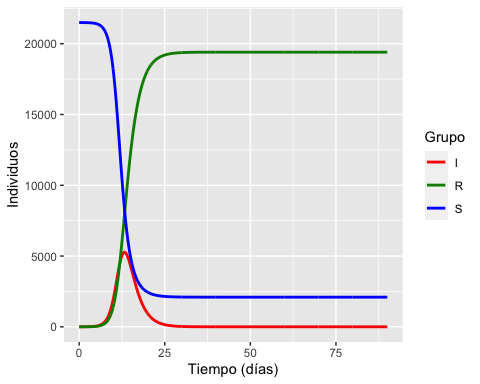
library(deSolve)  
#tamaño poblacional  
N = 50000  
pV = 0.6  
effV = 0.95  
#estado inicial de los compartimentos  
init <- c(S = 1-1/50000-pV\*effV,  
 I = 1/50000,  
 R = 0,  
 V = pV\*effV)  
#parámetros del modelo (coeficientes de las variables)  
param <- c(beta = 3,  
 gamma = 0.5)  
#crear la función con las ODE  
sirv <- function(times, init, param) {  
 with(as.list(c(init, param)), {  
#ecuaciones diferenciales   
 dS <- -beta \* S \* I  
 dI <- beta \* S \* I - gamma \* I  
 dR <- gamma \* I  
 dV <- 0  
#resultados de las tasas de cambio   
 return(list(c(dS, dI, dR, dV)))  
 })  
}  
#intervalo de tiempo y resolución  
times <- seq(0, 90, by = .1)  
#resolver el sistema de ecuaciones con función 'ode'  
out <- ode(y = init, times = times, func = sirv, parms = param)  
#cambiar out a un data.frame  
out <- as.data.frame(out)  
#eliminar la variable 'V' en out  
out$V <- NULL  
#mostrar 10 primeros datos  
head(out, 3)

## time S I R  
## 1 0.0 0.4299800 2.000000e-05 0.000000e+00  
## 2 0.1 0.4299773 2.164483e-05 1.041118e-06  
## 3 0.2 0.4299744 2.342491e-05 2.167855e-06

#rearreglo del output  
new.out <- as.data.frame(out) %>%  
 gather(key, value, -time) %>%  
 mutate(value=value\*N) #de proporción a individuos  
head(new.out, 3)

## time key value  
## 1 0.0 S 21499.00  
## 2 0.1 S 21498.87  
## 3 0.2 S 21498.72

#gráfica conjunta  
ggplot(data=new.out,   
 aes(x = time,  
 y = value,  
 group = key,  
 col = key  
 )) +   
 ylab("Individuos") + xlab("Tiempo (días)") +  
 geom\_line(size = 1) +   
 scale\_colour\_manual(values = c("red", "green4", "blue"), name = "Grupo") +  
 scale\_y\_continuous(labels = waiver(), limits = c(0, N-N\*pV\*effV))



# nivel máximo de infección  
new.out %>%   
 filter(key=="I") %>%   
 filter(value==max(value)) %>%   
 mutate(maxI = round(value, 0)) %>%  
 select(time, maxI)

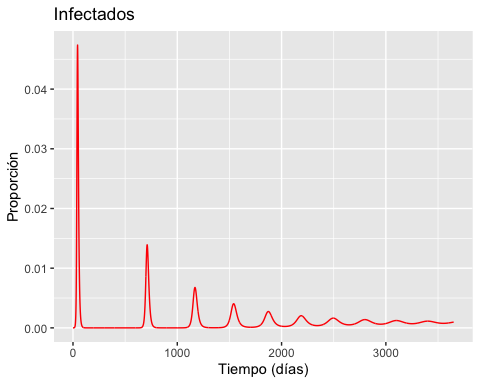
## time maxI  
## 1 13.2 5268

### Modelo SIR con parámetro demográfico

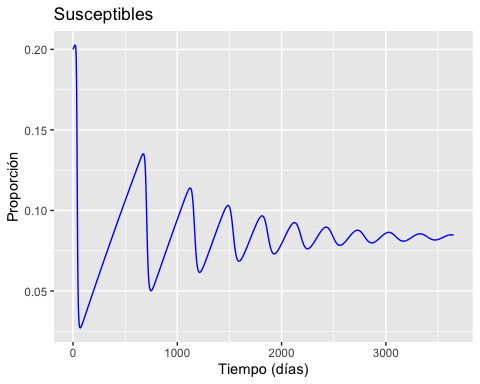
library(deSolve)  
library(tidyverse)  
#tamaño poblacional  
N = 1  
#estado inicial de los compartimentos  
init <- c(S = 0.2,  
 I = 1e-6,  
 R = 0.8 - 1e-6)  
#parámetros del modelo (coeficientes de las variables)  
param <- c(beta = 2.4,  
 gamma = 0.2,  
 mu = 2e-4)  
#crear la función con las ODE  
sir <- function(times, init, param) {  
 with(as.list(c(init, param)), {  
#ecuaciones diferenciales   
 dS <- mu - beta \* S \* I - mu \* S  
 dI <- beta \* S \* I - gamma \* I - mu \* I  
 dR <- gamma \* I - mu \* R  
#resultados de las tasas de cambio   
 return(list(c(dS, dI, dR)))  
 })  
}  
#intervalo de tiempo y resolución  
times <- seq(0, 3650, by = 1)  
#resolver el sistema de ecuaciones con función 'ode'  
out <- ode(y = init, times = times, func = sir, parms = param)  
#cambiar out a un data.frame  
out <- as.data.frame(out)  
#mostrar 3 primeros datos  
head(out, 3)

## time S I R  
## 1 0 0.2000000 1.000000e-06 0.7999990  
## 2 1 0.2001594 1.325479e-06 0.7998392  
## 3 2 0.2003186 1.757321e-06 0.7996796

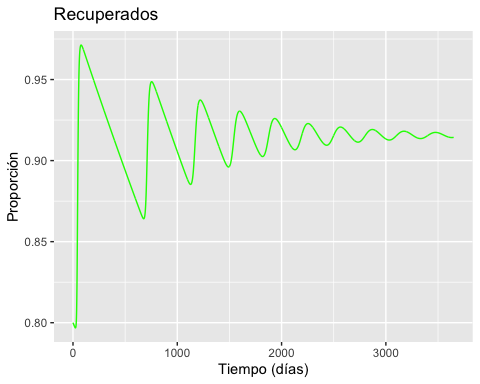
#gráficas individuales  
#infectados  
ggplot(data=out,   
 aes(x = time,  
 y = I)) +  
 ylab("Proporción") + xlab("Tiempo (días)") +  
 ggtitle("Infectados") +  
 geom\_line(size = 0.5, col = "red") +   
 scale\_y\_continuous(labels = waiver())



#susceptibles  
ggplot(data=out,   
 aes(x = time,  
 y = S)) +  
 ylab("Proporción") + xlab("Tiempo (días)") +  
 ggtitle("Susceptibles") +  
 geom\_line(size = 0.5, col = "blue") +   
 scale\_y\_continuous(labels = waiver())



#recuperados  
ggplot(data=out,   
 aes(x = time,  
 y = R)) +  
 ylab("Proporción") + xlab("Tiempo (días)") +  
 ggtitle("Recuperados") +  
 geom\_line(size = 0.5, col = "green") +   
 scale\_y\_continuous(labels = waiver())

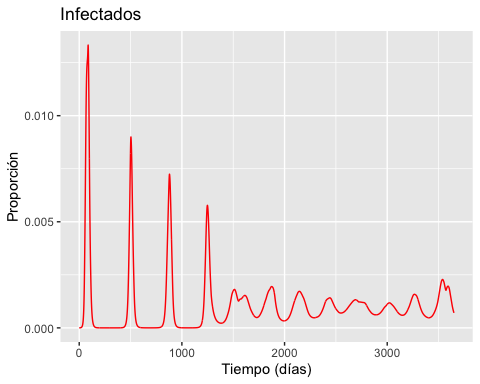


### Modelo SIR demográfico y aleatorio

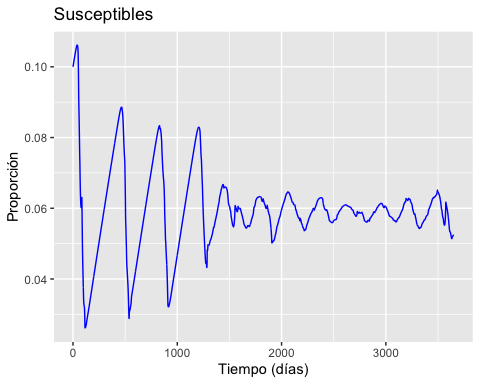
library(deSolve)  
#tamaño poblacional  
N = 1  
#estado inicial de los compartimentos  
init <- c(S = 0.1,  
 I = 1e-6,  
 R = 0.9 - 1e-6)  
#parámetros del modelo (coeficientes de las variables)  
param <- c(beta = 3.4,  
 gamma = 0.2,  
 mu = 2e-4)  
#crear la función con las ODE  
sir <- function(times, init, param) {  
 with(as.list(c(init, param)), {  
#ecuaciones diferenciales   
 dS <- mu - beta \* S \* I \* runif(1, 0.999, 1.001) - mu \* S  
 dI <- beta \* S \* I - gamma \* I - mu \* I  
 dR <- gamma \* I - mu \* R  
#resultados de las tasas de cambio   
 return(list(c(dS, dI, dR)))  
 })  
}  
#intervalo de tiempo y resolución  
times <- seq(0, 3650, by = 1)  
#resolver el sistema de ecuaciones con función 'ode'  
out <- ode(y = init, times = times, func = sir, parms = param)  
#cambiar out a un data.frame  
out <- as.data.frame(out\*N) #aqui puede multiplicar 'out' por N  
#mostrar 6 primeros datos  
head(out)

## time S I R  
## 1 0 0.1000000 1.000000e-06 0.8999990  
## 2 1 0.1001796 1.152997e-06 0.8998192  
## 3 2 0.1003587 1.330245e-06 0.8996396  
## 4 3 0.1005380 1.536152e-06 0.8994599  
## 5 4 0.1007171 1.779988e-06 0.8992804  
## 6 5 0.1008923 2.056943e-06 0.8991010

#gráficas individuales  
#infectados  
ggplot(data=out,   
 aes(x = time,  
 y = I)) +  
 ylab("Proporción") + xlab("Tiempo (días)") +  
 ggtitle("Infectados") +  
 geom\_line(size = 0.5, col = "red") +   
 scale\_y\_continuous(labels = waiver())



#susceptibles  
ggplot(data=out,   
 aes(x = time,  
 y = S)) +  
 ylab("Proporción") + xlab("Tiempo (días)") +  
 ggtitle("Susceptibles") +  
 geom\_line(size = 0.5, col = "blue") +   
 scale\_y\_continuous(labels = waiver())



#recuperados  
ggplot(data=out,   
 aes(x = time,  
 y = R)) +  
 ylab("Proporción") + xlab("Tiempo (días)") +  
 ggtitle("Recuperados") +  
 geom\_line(size = 0.5, col = "green") +   
 scale\_y\_continuous(labels = waiver())



# Modelo con disminución de contactos (distanciamiento físico)

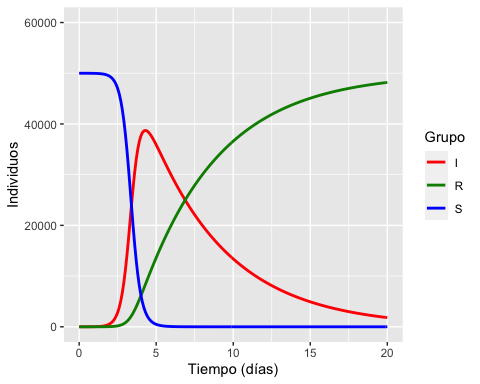
library(deSolve)  
#tamaño poblacional  
N = 50000  
#estado inicial de los compartimentos  
init <- c(S = 1-1/50000,  
 I = 1/50000,  
 R = 0)  
#parámetros del modelo (coeficientes de las variables)  
param <- c(beta = 3.4,  
 gamma = 0.2,  
 rho = 1)  
#crear la función con las ODE  
sir <- function(times, init, param) {  
 with(as.list(c(init, param)), {  
#ecuaciones diferenciales   
 dS <- -rho \* beta \* S \* I  
 dI <- rho \* beta \* S \* I - gamma \* I  
 dR <- gamma \* I  
#resultados de las tasas de cambio   
 return(list(c(dS, dI, dR)))  
 })  
}  
#intervalo de tiempo y resolución  
times <- seq(0, 20, by = .1)  
#resolver el sistema de ecuaciones con función 'ode'  
out <- ode(y = init, times = times, func = sir, parms = param)  
#cambiar out a un data.frame  
out <- as.data.frame(out)  
head(out, 3)

## time S I R  
## 1 0.0 0.9999800 2.000000e-05 0.000000e+00  
## 2 0.1 0.9999719 2.761384e-05 4.758773e-07  
## 3 0.2 0.9999608 3.811587e-05 1.132278e-06

#rearreglo del output  
new.out <- as.data.frame(out) %>%  
 gather(key, value, -time) %>%  
 mutate(value=value\*N) #de proporción a individuos  
head(new.out, 3)

## time key value  
## 1 0.0 S 49999.00  
## 2 0.1 S 49998.60  
## 3 0.2 S 49998.04

#gráfica conjunta  
ggplot(data=new.out,   
 aes(x = time,  
 y = value,  
 group = key,  
 col = key  
 )) +   
 ylab("Individuos") + xlab("Tiempo (días)") +  
 geom\_line(size = 1) +   
 scale\_colour\_manual(values = c("red", "green4", "blue"), name = "Grupo") +  
 scale\_y\_continuous(labels = waiver(), limits = c(0, 60000))



# nivel máximo de infección  
new.out %>%   
 filter(key=="I") %>%   
 filter(value==max(value)) %>%   
 mutate(maxI = round(value, 0)) %>%  
 select(time, maxI)

## time maxI  
## 1 4.3 38726