BirthDeath\_újszülöttek S-be  
kész 😊 R0 jó, plot jó

Bori

3 mai 2019

##########################################################  
# Starting script to the module 'SIR models of epidemics'  
##########################################################  
  
# implements the basic SIR model, and plots simulation results  
  
## Here you only need to know basic things:  
## # To make comments  
## <- Assigning a value to a symbol  
## c To create a vector c(v1, v2, ..., vn)  
## e is \*10^  
## seq(a,b,c) To create a sequence from a to b with intervals c  
  
  
###################################  
# FUNCTION DEFINITIONS  
###################################  
  
###  
# SIR <- function(t, x, parms)  
# Use: calculates the derivatives for the SIR model  
# Input:   
# t: time (not used here, because there is no explicit time dependence)  
# x: vector of the current values of all variables (S, I, R)  
# parms: vector of all parameter values  
# Output:  
# der: vector of derivatives  
  
# To use the lsoda function, the model function has to be a function of t (time),  
# x (the vector of the variables) and parms (the parameters of the model).  
  
SIR <- function(t, x, parms){  
   
 # Beta and r are not global variables. This means that if you type beta in R, the output will be 'beta',  
 # and not its value. You have to specify that you want to use the value of beta from 'parms' to solve the ODEs.  
 # Similarly, the variables of the model are taken from the vector x. This is done by the 'with' function.  
   
 with(as.list(c(parms,x)),{  
 dS <- d\*(S+I+R) +g\*I - beta\*S\*I - d\*S   
 dI <- + beta\*S\*I - r\*I - d\*I -g\*I  
 dR <- r\*I - d\*R   
# Note: B=d\*(S+I+R)+g\*I =születés D=d\*S+d\*I+d\*R+g\*I =halál 🡪 születés=halál, mindenki, aki születik fogékony lesz, N itt is konstans  
  
# Note: because S+I+R=constant, this equation could actually be omitted,  
 # and R at any time point could simply be calculated as N-S-I.  
 der <- c(dS, dI,dR)  
 list(der) # the output must be returned   
 }) # end of 'with'  
   
} # end of function definition  
  
  
###########################  
# MAIN PROGRAM  
###########################  
  
### LOAD LIBRARIES  
#load R library for ordinary differential equation solvers  
library(deSolve)  
  
### INITIALIZE PARAMETER SETTINGS  
for (i in 0:25){  
parms <- c(beta=1e-2, r=1e-1, d=0.1, g=i\*0.2) # set the parameters of the model  
inits <- c(S=499, I=1, R=0) # set the initial values  
dt <- seq(0,60,0.1) # set the time points for evaluation  
  
# Calculate and print R\_0 on the screen  
  
N <- sum(inits)  
R\_0 <- with(as.list(parms),{beta\*N/(r+d+g)}) # Note: mert R0 már nem cask r-től függ fordítottan, hanem a halálozási rátáktól is, I egyenlete miatt  
  
  
### SIMULATE THE MODEL  
  
## Use lsoda to solve the differential equations numerically. The syntax should be  
## lsoda(initial values, time points, function, parameters)  
  
simulation <- as.data.frame(lsoda(inits, dt, SIR, parms=parms)) # this way our set 'parms' will be used as default  
  
### PLOT THE OUTPUT  
  
# If you remove the # before pdf(...) and dev.off(), the output will be written in a pdf file,  
# in the working directory. If you don't, a window containing your graph will just pop up.  
  
#pdf("startingscript.pdf")  
#par(cex=1.7)  
# Plot S according to time, in blue, and add the graph I and R according to time,  
# in red and dark green respectively. Call help(plot) for further details.  
  
### PLOT THE OUTPUT  
library(ggplot2)  
  
#itt írom be, hogy kiírja nekem a beta-kat meg r-eket is a plotra  
g=parms[4]  
title = paste("Infected\_extradeath = ",g, sep = "")  
beta = parms[1]  
r = parms[2]  
d=parms[3]  
beta2 = paste("beta = ",beta, sep = "")  
r2 = paste("r = ",r,sep = "")  
d2=paste("death = ",d, sep="")  
r0= paste("R\_0 = ",round(R\_0,2), sep = "") # Note: round(x,2)->két tizedesjegy pontossággal adja meg az x-et  
subtitle = paste(d2,beta2,r2,r0, sep = ", ")  
  
#plot script  
plot = ggplot(simulation, aes(x = time)) +   
 geom\_line(aes(y = S, colour = "Susceptible"), size=2, alpha=0.45) +   
 geom\_line(aes(y = I, colour = "Infected"), size=2, alpha=0.45) +   
 geom\_line(aes(y = R, colour = "Recovered"), size=2, alpha=0.45) +  
 ylab(label="Number of individuals") +   
 xlab("Time") +  
 ggtitle(title, subtitle = subtitle) +  
 scale\_color\_manual(name = " ",  
 values = c("Susceptible" = "turquoise4", "Infected" = "tomato3", "Recovered" = "goldenrod2")) +  
 theme(plot.background = element\_rect(fill = "antiquewhite4"),  
 legend.background = element\_rect(fill = "antiquewhite4"),  
 plot.title = element\_text(size = 18, lineheight=.8, hjust=0.5, face="bold", colour="antiquewhite"),  
 plot.subtitle = element\_text(size = 16, lineheight=.8, hjust=0.5, face="italic", colour="antiquewhite"),  
 axis.title = element\_text(size = 15, face="bold",colour="antiquewhite"),  
 axis.text = element\_text(size = 12,colour="antiquewhite"),  
 legend.position="top",  
 legend.key = element\_rect(fill = "seashell"),  
 legend.text = element\_text(size = 12,colour="antiquewhite"),  
 legend.box.background = element\_rect(colour = "antiquewhite", size=1.5),  
 panel.background = element\_rect(fill = "seashell", colour = "seashell", size = 0.5, linetype = "solid"),  
 panel.grid.major = element\_line(size = 0.5, linetype = 'solid', colour = "antiquewhite"),   
 panel.grid.minor = element\_line(size = 0.25, linetype = 'solid', colour = "antiquewhite"))  
print(plot)  
}

## Warning: package 'ggplot2' was built under R version 3.5.3

