Branching-model-2.R

ruwan

2020-09-24

#################################################################  
## Branching process model for generating caseload of symptomatic  
## cholera cases in a small outbreak (ring), repeated 1000 times  
## Author: Ruwan Ratnayake, September 2020  
## Code for branching process adapted from Althaus, 2015   
## <doi.org/10.1016/S1473-3099(15)70135-0>  
#################################################################  
  
##############################  
# 0 - Load libraries  
##############################  
library(fitdistrplus)

## Loading required package: MASS

## Warning: package 'MASS' was built under R version 4.0.2

## Loading required package: survival

## Warning: package 'survival' was built under R version 4.0.2

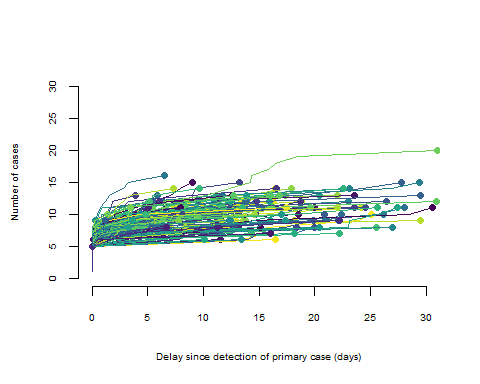
library(viridis)

## Loading required package: viridisLite

library(truncnorm)

## Warning: package 'truncnorm' was built under R version 4.0.2

##############################  
# 1 - Parameters  
##############################  
  
# Incubation period and serial interval (gamma distribution) (Kahn, 2019, Azman, 2015)   
r.0 <- 2.5 #Azman, 2015 (early reproduction number before intervention)  
#inc.per <- 1.4 #Azman, 2012 (median incubation period)  
ser.int <- 5 #Kahn, 2019; Azman, 2015  
ser.int.rate <- 0.1 #Kahn, 2019; Azman, 2015  
ser.int.shape<- 0.5 #Kahn, 2019; Azman, 2015  
#ser.int.distr<- rgamma(n=100, shape=ser.int.shape, rate=ser.int.rate, scale=1/ser.int.rate, lower.tail)  
k <- 4.5 #Moore, 2014 (dispersion parameter for low potential of superspreading)  
  
# Baseline scenario  
#num.initial.cases <- c(1, 5, 10, 15) #Clusters of symptomatic persons at detection can be varying sizes  
cap.max.days <- 30 #Days under observation  
cap.cases <- 1000 #Maximum number of cases within a ring  
  
# Assumptions for detection and response of cluster  
# Delays to cluster detection; e.g. first day of response  
# Minimum 0 days with next day response (+1 = 1)  
# Maximum 7 days with next day response (+1 = 8)  
#delay.distr.short <- rtruncnorm(n=100, a=1, b=3, mean=2, sd=1)  
#delay.distr.long <- rtruncnorm(n=100, a=4, b=8, mean=6, sd=2)  
#time.to.protection.vacc <- 7 #Delay to vibriocidal antibody response (Azman, 2016)  
  
# Estimates of reduction of R0  
antibiotic.eff <- 0.66 #ACP, Reveiz 2011 (meta)  
vacc.2m.eff <- 0.87 #OCV, 2 month, Azman, 2015 (case-cohort)  
#vacc.12m.eff <- 0.69 #OCV, 12 month, Bi 2017 (meta)  
water.tx.eff <- 0.26 #POUWT, Fewtrell, 2005 (meta)  
water.store.eff<- 0.21 #Safe storage, Roberts 2001 (RCT)  
pop.cover <- 0.8 #Coverage at the population level   
  
# Overall effectiveness of CATI  
# Impact (effected on 1st day; initial). Assume no impact from vaccination.  
# Estimated reduction of R0 by CATI without vaccination is 0.8  
cati.efficacy <- 1-((1-antibiotic.eff)\*(1-water.tx.eff)\*(1-water.store.eff))  
# Estimated impact within first week is to knock down R0 to 0.998  
cati.effectiveness <- pop.cover\*cati.efficacy  
re.cati.0.6.d <- r.0\*(1-(cati.effectiveness))  
  
# Impact (effected on 7th day; initial). Include impact from vaccination.  
# Estimated reduction of R0 by CATI without vaccination results in Re=0.97  
cati.vacc.efficacy <- 1-((1-vacc.2m.eff)\*(1-antibiotic.eff)\*(1-water.tx.eff)\*(1-water.store.eff))  
# Estimated reduction of R0 by CATI with vaccination results in Re=0.67  
cati.vacc.effectiveness <- pop.cover\*cati.vacc.efficacy  
re.cati.7.d <- r.0\*(1-(cati.vacc.effectiveness))  
   
   
################################  
# 2 - Setup model and functions  
################################  
  
#Functions  
#serial.int<-function(x){rgamma(x,shape=0.1, rate=0.5)}  
#incub.param=c(1,4, 1.98) #Azman, 2013   
#inf.param=c(2, 1) #Weil, 2014  
  
#incub.fn<-function(x){rgamma(x,shape=incub.param[1]/(incub.param[2]^2/incub.param[1]), scale=(incub.param[2]^2/incub.param[1]))} # incubation period  
#infec.fn<-function(x){rgamma(x,shape=inf.param[1]/(inf.param[2]^2/inf.param[1]), scale=(inf.param[2]^2/inf.param[1]))} # infectious period  
  
# time to reporting (and CATI implementation)  
#reporting.fn<-function(x){ #0 # Set to zero, as incorporated using function below instead  
# rtruncnorm(n, a=1, b=7, mean=2, sd=1)  
#}  
  
###########################################  
# 3 - Run the branching process model  
###########################################  
set.seed(123)  
runs <- 200  
seed <- 5  
  
# Simulate outbreak trajectory given time to CATI implementation of 1 day  
# and vaccine effects after 7 days  
plot(NA,xlim=c(0,30),ylim=c(0,30),xlab="Delay since detection of primary case (days)",  
 ylab="Number of cases", cex.lab=0.6, cex.axis=0.6, frame=FALSE)  
cols <- sample(viridis(runs))  
  
# Model exponential growth given varying R values as a result of R0 (day 1),   
# then CATI without vaccination protection (days 2-6), followed by CATI with   
# vaccination protection (days 7-30)  
total.cases <- integer(runs)  
  
for(i in 1:runs) {  
 cases <- seed  
 t <- rep(0,seed)  
 times <- t  
  
 while(cases > 0)   
   
 if (length(times)<2)   
 {  
 secondary <- rnbinom(cases,size=k,mu=r.0)  
 t.new <- numeric()  
 for(j in 1:length(secondary)) {  
 t.new <- c(t.new,t[j] + rgamma(secondary[j],shape=ser.int.shape[1],  
 scale=(1/ser.int.rate)))  
 }  
 cases <- length(t.new) & cases<cap.cases  
 t <- t.new  
 times <- c(times[times<31],t.new)  
 }   
 else if (length(times)>1 & length(times)<7) {  
 secondary <- rnbinom(cases,size=k,mu=re.cati.0.6.d)  
 t.new <- numeric()  
 for(j in 1:length(secondary)) {  
 t.new <- c(t.new,t[j] + rgamma(secondary[j],shape=ser.int.shape[1],  
 scale=(1/ser.int.rate)))  
 }  
 cases <- length(t.new) & cases<cap.cases  
 t <- t.new  
 times <- c(times[times<31],t.new)  
 }   
 else if (length(times)>6 & length(times)<31){  
 secondary <- rnbinom(cases,size=k,mu=re.cati.7.d)  
 t.new <- numeric()  
 for(j in 1:length(secondary)) {  
 t.new <- c(t.new,t[j] + rgamma(secondary[j],shape=ser.int.shape[1],  
 scale=(1/ser.int.rate)))  
 }  
 cases <- length(t.new) & cases<cap.cases  
 t <- t.new  
 times <- c(times[times<31],t.new)  
 }   
 lines(sort(times),1:length(times),col=cols[i],lwd=0.5)  
 points(max(times),length(times),col=cols[i],pch=16)  
 total.cases[i] <-length(times)  
}



# Generate final epidemic size and parameters  
print(total\_cases.median <- median(total.cases))

## [1] 10

print(total\_cases.sd <- sd(total.cases))

## [1] 2.269837

print(total\_cases.range <- range(total.cases))

## [1] 5 20

#Transform into log scale to estimate proportion under 5, 10, and 20 cases  
# using a log normal distribution  
lsm=log(total\_cases.median)-(1/2)\*log((total\_cases.sd/total\_cases.median)^2+1)  
lssd=sqrt(log((total\_cases.sd/total\_cases.median)^2+1))  
plnorm(5, lsm, lssd)

## [1] 0.00143927

plnorm(10, lsm, lssd)

## [1] 0.5446158

plnorm(20, lsm, lssd)

## [1] 0.9993237

# Create dataframe with final epidemic size and incidence across 200 runs.  
# Incidence = (epidemic size/500 population) \* 1000  
cases.df <- data.frame("run.num"=1:200, "symp.chol.cases.30d" = total.cases,  
 "chol.inc.1000.30d" = (total.cases/500)\*1000)  
View(cases.df)  
write.csv(cases.df,"C:\\Users\\Ruwan\\Desktop\\Sample-size-simulation\\Incidence.csv", row.names = FALSE)