‘Epidemiology and Infection’

Persistent infections support maintenance of a coronavirus in a population of Australian bats (*Myotis* *macropus*)

J. Jeong, C. S. Smith, A. J. Peel, R. K. Plowright, D. H. Kerlin, J. McBroom, H. McCallum

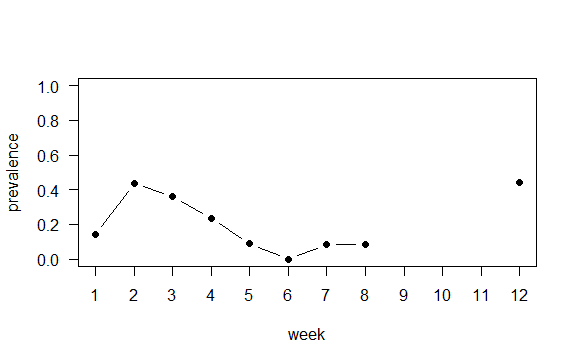
‘Supplementary Material’

**Supplementary Material Table**

Table S1. Detection of a putative novel Alphacoronaviruses in a 52 Myotis macropus from a Capture-Mark-Recapture study. This table was modified with permission from Smith [1].

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | Capturing occasion | | | | | | | | |
| Recaptured | Coronavirus RNA | Bat | Sex | Age | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Recaptured | Multiple Detections | 1 | Male | Unknown | + | - |  |  |  |  |  | + | + |
|  |  | 2 | Female | Adult | - | + | + |  |  |  | + |  | + |
|  |  | 3 | Female | Sub-adult |  | + | + |  | - |  |  |  |  |
|  |  | 4 | Female | Sub-adult |  | + |  |  |  |  | - | - | + |
|  |  | 5 | Male | Unknown |  |  | + | + | - |  | - |  | + |
|  |  | 6 | Male | Unknown |  |  |  | + |  |  |  |  | + |
|  |  | 7 | Male | Unknown |  |  |  | + |  |  |  |  | + |
|  | Single detection | 8 | Female | Sub-adult | + |  |  |  |  |  |  | - |  |
|  |  | 9 | Male | Unknown |  | + | - |  |  |  |  | - |  |
|  |  | 10 | Male | Unknown |  | + | - |  |  |  | - |  |  |
|  |  | 11 | Male | Unknown |  | + | - | - |  |  | - | - |  |
|  |  | 12 | Female | Sub-adult |  | + | - |  | - |  |  |  |  |
|  |  | 13 | Female | Sub-adult |  | + |  |  |  |  |  | - | - |
|  |  | 14 | Male | Unknown |  |  | + | - | - |  | - | - |  |
|  |  | 15 | Male | Unknown |  |  | + |  | - |  |  |  |  |
|  |  | 16 | Male | Unknown | - |  | + |  |  |  |  |  |  |
|  |  | 17 | Female | Adult |  | - | + |  |  |  |  |  |  |
|  |  | 18 | Female | Adult |  |  |  | - | + |  |  |  | - |
|  |  | 19 | Female | Adult | - |  |  | - |  |  |  |  | + |
|  |  | 20 | Female | Adult |  | - | - |  |  |  |  |  | + |
|  |  | 21 | Female | Adult |  | - | - | - |  |  |  |  | + |
|  |  | 22 | Female | Adult |  | - |  | - | - |  |  |  | + |
|  |  | 23 | Female | Adult |  |  | - | - | - |  |  |  | + |
|  | Not Detected | 24 | Female | Adult | - |  | - |  |  |  |  | - | - |
|  |  | 25 | Female | Adult | - | - | - |  |  |  | - |  |  |
|  |  | 26 | Female | Sub-adult | - | - |  |  |  |  |  |  |  |
|  |  | 27 | Male | Unknown | - | - |  |  |  |  |  |  |  |
|  |  | 28 | Female | Adult | - |  |  |  |  |  | - |  | - |
|  |  | 29 | Male | Unknown | - |  |  | - |  |  | - |  |  |
|  |  | 30 | Male | Unknown | - |  |  |  |  |  |  | - |  |
|  |  | 31 | Female | Sub-adult | - |  |  |  | - | - |  |  |  |
|  |  | 32 | Female | Adult |  | - |  |  | - |  |  |  | - |
|  |  | 33 | Female | Adult |  | - | - |  |  |  |  |  | - |
|  |  | 34 | Female | Sub-adult |  | - | - |  |  |  |  |  | - |
|  |  | 35 | Male | Unknown |  |  | - |  |  |  | - |  | - |
|  |  | 36 | Female | Adult |  |  | - |  |  |  | - |  | - |
|  |  | 37 | Female | Adult |  |  | - |  | - |  | - |  |  |
|  |  | 38 | Female | Adult |  |  | - |  |  |  |  |  | - |
|  |  | 39 | Female | Sub-adult |  |  |  | - |  |  |  | - | - |
|  |  | 40 | Male | Unknown |  |  |  | - |  |  |  | - | - |
|  |  | 41 | Male | Unknown |  |  |  | - |  |  |  | - | - |
|  |  | 42 | Male | Unknown |  |  |  | - |  |  |  |  | - |
| Not recaptured | Single Detection | 43 | Male | Unknown |  | + |  |  |  |  |  |  |  |
|  |  | 44 | Male | Unknown |  | + |  |  |  |  |  |  |  |
|  |  | 45 | Female | Sub-adult |  |  | + |  |  |  |  |  |  |
|  |  | 46 | Female | Sub-adult |  |  | + |  |  |  |  |  |  |
|  |  | 47 | Male | Unknown |  |  |  | + |  |  |  |  |  |
|  | Not Detected | 48 | Male | Unknown | - |  |  |  |  |  |  |  |  |
|  |  | 49 | Female | Adult |  | - |  |  |  |  |  |  |  |
|  |  | 50 | Male | Unknown |  | - |  |  |  |  |  |  |  |
|  |  | 51 | Male | Unknown |  |  | - |  |  |  |  |  |  |
|  |  | 52 | Female | Sub-adult |  |  |  | - |  |  |  |  |  |

**Supplementary Material Figure**

Figure S1. Prevalence of a putative novel *Alphacoronaviruses* in a 52 *Myotis macropus* from a Capture-Mark-Recapture study [1]. Capturing of *M. macropus* was not performed from week 9 to 11. We used an equation, , to calculate the grow rate of the epidemic (Λ) [2]. I(t) and I(0) represent the number of infectious individuals at time t and the number of infectious individuals at the start. We assumed that week 7 was the start of the epidemic, and we assumed that week 12 as time t. Thus, we calculated the growth rate from week 7 to week 12. The number of infectious individuals was calculated by multiplying the prevalence of each week with the population size of 86.

As a result, Λ=0.3328.

**Supplementary Material R codes**

**R code S1. Analyses of Capture-Mark-Recapture data of *Myotis macropus* using R2OpenBUGS package in R*.* This R code analyzed model number 3 in table 3. The output of this R code generated the estimates or ranges of all infection type in Table 2.**

# text file

"E:/all.txt"

2 1 0 0 0 0 0 2 2

1 2 2 0 0 0 2 0 2

0 2 2 0 1 0 0 0 0

0 2 0 0 0 0 1 1 2

0 0 2 2 1 0 1 0 2

0 0 0 2 0 0 0 0 2

0 0 0 2 0 0 0 0 2

2 0 0 0 0 0 0 1 0

0 2 1 0 0 0 0 1 0

0 2 1 0 0 0 1 0 0

0 2 1 1 0 0 1 1 0

0 2 1 0 1 0 0 0 0

0 2 0 0 0 0 0 1 1

0 0 2 1 1 0 1 1 0

0 0 2 0 1 0 0 0 0

1 0 2 0 0 0 0 0 0

0 1 2 0 0 0 0 0 0

0 0 0 1 2 0 0 0 1

1 0 0 1 0 0 0 0 2

0 1 1 0 0 0 0 0 2

0 1 1 1 0 0 0 0 2

0 1 0 1 1 0 0 0 2

0 0 1 1 1 0 0 0 2

#####################################

##  Without grouping of bats into two groups based on frequency of coronavirus detection

library(R2OpenBUGS)

# Import data

CH <- as.matrix(read.table(file = "E:/all.txt", sep = " "))

n.occasions<-dim(CH)[2]

#Compute vector with occasion of first capture

f<-numeric()

for(i in 1:dim(CH)[1]){f[i]<-min(which(CH[i,]!=0))}

#Recode CH matrix: note, a 0 is not allowed by OpenBUGS!

# 1=seen negative, 2= positive, 3= not seen

rCH<-CH #Recoded CH

rCH[rCH==0]<-3

# Specify model in BUGS language

sink("E:/corona\_multistate\_all.txt")

cat("

    model {

    # -------------------------------------------------

    # Parameters:

    # phiA: survival probability at Negative

    # phiB: survival probability at Positive

    # psiAB: transition probability from Negative to Positive

    # psiBA: transition probability from Positive to Negative

    # pA: recapture probability at Negative

    # pB: recapture probability at Positive

    # -------------------------------------------------

    # States (S):

    # 1 alive Negative

    # 2 alive Positive

    # 3 dead

    # Observations (O):

    # 1 seen Negative

    # 2 seen Positive

    # 3 not seen

    # -------------------------------------------------

    # Priors and constraints

    for (t in 1:(n.occasions-1)){

    phiA[t] <- mean.phi[1]

    phiB[t] <- mean.phi[2]

    psiAB[t] <- mean.psi[1]

    psiBA[t] <- mean.psi[2]

    p[t] <- mean.p

    }

    mean.p ~ dunif(0, 1)      # Priors for mean state-spec. recapture

    for (u in 1:2){

    mean.phi[u] ~ dunif(0, 1)    # Priors for mean state-spec. survival

    mean.psi[u] ~ dunif(0, 1)    # Priors for mean transitions

    }

    # Define state-transition and observation matrices

    for (i in 1:nind){

    # Define probabilities of state S(t+1) given S(t)

    for (t in f[i]:(n.occasions-1)){

    ps[1,i,t,1] <- phiA[t] \* (1-psiAB[t])

    ps[1,i,t,2] <- phiA[t] \* psiAB[t]

    ps[1,i,t,3] <- 1-phiA[t]

    ps[2,i,t,1] <- phiB[t] \* psiBA[t]

    ps[2,i,t,2] <- phiB[t] \* (1-psiBA[t])

    ps[2,i,t,3] <- 1-phiB[t]

    ps[3,i,t,1] <- 0

    ps[3,i,t,2] <- 0

    ps[3,i,t,3] <- 1

    # Define probabilities of O(t) given S(t)

    po[1,i,t,1] <- p[t]

    po[1,i,t,2] <- 0

    po[1,i,t,3] <- 1-p[t]

    po[2,i,t,1] <- 0

    po[2,i,t,2] <- p[t]

    po[2,i,t,3] <- 1-p[t]

    po[3,i,t,1] <- 0

    po[3,i,t,2] <- 0

    po[3,i,t,3] <- 1

    } #t

    } #i

    # Likelihood

    for (i in 1:nind){

    # Define latent state at first capture

    z[i,f[i]] <- y[i,f[i]]

    for (t in (f[i]+1):n.occasions){

    # State process: draw S(t) given S(t-1)

    z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])

    # Observation process: draw O(t) given S(t)

    y[i,t] ~ dcat(po[z[i,t], i, t-1,])

    } #t

    } #i

    }

    ",fill = TRUE)

sink()

# Function to create known latent states z

[known.state.ms](http://known.state.ms/) <- function(ms, notseen){

  state <- ms

  state[state==notseen] <- NA

  for (i in 1:dim(ms)[1]){

    m <- min(which(![is.na](http://is.na/)(state[i,])))

    state[i,m] <- NA

  }

  return(state)

}

# Function to create initial values for unknown z

ms.init.z <- function(ch, f){

  for (i in 1:dim(ch)[1]){ch[i,1:f[i]] <- NA}

  states <- max(ch, na.rm = TRUE)

  known.states <- 1:(states-1)

  v <- which(ch==states)

  ch[-v] <- NA

  ch[v] <- sample(known.states, length(v), replace = TRUE)

  return(ch)

}

# Bundle data

bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2], nind = dim(rCH)[1], z = [known.state.ms](http://known.state.ms/)(rCH, 3))

# Initial values

inits <- function(){list(mean.phi = runif(2, 0, 1), mean.psi = runif(2, 0, 1), mean.p = runif(2, 0, 1), z = ms.init.z(rCH, f))}

# Parameters monitored

parameters <- c("mean.phi", "mean.psi", "mean.p")

# MCMC settings

ni <- 10000

nt <- 6

nb <- 1000

nc <- 3

# Call OpenBUGS from R

multistate.total<-bugs(data=bugs.data,inits=inits,  parameters.to.save=parameters,n.iter=ni,model.file = "E:/corona\_multistate\_all.txt", n.chains=nc, n.burnin=nb, n.thin=nt,  debug=TRUE)

print(multistate.total, digits=4)

**R code S2. Analyses of Capture-Mark-Recapture data of *Myotis macropus* using R2OpenBUGS package in R*.* This R code analyzed model number 1 in table 3. The output of this R code generated the estimates or ranges of persistent and transient infection types in Table 2.**

# text file

“E:/persistent.txt”

2 1 0 0 0 0 0 2 2

1 2 2 0 0 0 2 0 2

0 2 2 0 1 0 0 0 0

0 2 0 0 0 0 1 1 2

0 0 2 2 1 0 1 0 2

0 0 0 2 0 0 0 0 2

0 0 0 2 0 0 0 0 2

"E:/transient.txt"

2 0 0 0 0 0 0 1 0

0 2 1 0 0 0 0 1 0

0 2 1 0 0 0 1 0 0

0 2 1 1 0 0 1 1 0

0 2 1 0 1 0 0 0 0

0 2 0 0 0 0 0 1 1

0 0 2 1 1 0 1 1 0

0 0 2 0 1 0 0 0 0

1 0 2 0 0 0 0 0 0

0 1 2 0 0 0 0 0 0

0 0 0 1 2 0 0 0 1

1 0 0 1 0 0 0 0 2

0 1 1 0 0 0 0 0 2

0 1 1 1 0 0 0 0 2

0 1 0 1 1 0 0 0 2

0 0 1 1 1 0 0 0 2

#######################################

#######################################

##  With grouping of bats into two groups based on frequency of coronavirus detection

library(R2OpenBUGS)

CHm<-as.matrix(read.table("E:/persistent.txt")) #bats with multiple detections

CHs<-as.matrix(read.table(file = "E:/transient.txt", sep = " ")) #bats with single detection

# Merge capture-histories by row

CH <- rbind(CHm, CHs)

group<-c(rep(1,dim(CHm)[1]),rep(2,dim(CHs)[1]))

n.occasions<-dim(CH)[2]

#Compute vector with occasion of first capture

f<-numeric()

for(i in 1:dim(CH)[1]){f[i]<-min(which(CH[i,]!=0))}

#Recode CH matrix: note, a 0 is not allowed by OpenBUGS!

# 1=seen negative, 2= positive, 3= not seen

rCH<-CH #Recoded CH

rCH[rCH==0]<-3

# Specify model in BUGS language

sink("E:/corona\_multistate\_persistent+transient.txt")

cat("

    model {

    # -------------------------------------------------

    # Parameters:

    # phiA: survival probability at Negative

    # phiB: survival probability at Positive

    # psiAB: transition probability from Negative to Positive

    # psiBA: transition probability from Positive to Negative

    # p: recapture probability

    # -------------------------------------------------

    # States (S):

    # 1 alive Negative

    # 2 alive Positive

    # 3 dead

    # Observations (O):

    # 1 seen Negative

    # 2 seen Positive

    # 3 not seen

    # -------------------------------------------------

    #####################################

    # Priors and constraints

    for (i in 1:nind){

    for (t in 1:(n.occasions-1)){

    phiA[i,t] <- mean.phi.g[1,group[i]]

    phiB[i,t] <- mean.phi.g[2,group[i]]

    psiAB[i,t] <- mean.psi.g[1,group[i]]

    psiBA[i,t] <- mean.psi.g[2,group[i]]

    p[i,t] <- mean.p.g[group[i]]

    }#t

    }#i

    for (u in 1:g){

    mean.p.g[u] ~ dunif(0, 1)      # Priors for mean state-spec. recapture

    }

    for (u in 1:g){

    for (v in 1:2){

    mean.phi.g[v,u] ~ dunif(0, 1)    # Priors for mean state-spec. survival

    mean.psi.g[v,u] ~ dunif(0, 1)    # Priors for mean transitions

    }}

   ##### Define state-transition and observation matrices

    for (i in 1:nind){

    # Define probabilities of state S(t+1) given S(t)

    for (t in f[i]:(n.occasions-1)){

    ps[1,i,t,1] <- phiA[i,t] \* (1-psiAB[i,t])

    ps[1,i,t,2] <- phiA[i,t] \* psiAB[i,t]

    ps[1,i,t,3] <- 1-phiA[i,t]

    ps[2,i,t,1] <- phiB[i,t] \* psiBA[i,t]

    ps[2,i,t,2] <- phiB[i,t] \* (1-psiBA[i,t])

    ps[2,i,t,3] <- 1-phiB[i,t]

    ps[3,i,t,1] <- 0

    ps[3,i,t,2] <- 0

    ps[3,i,t,3] <- 1

    # Define probabilities of O(t) given S(t)

    po[1,i,t,1] <- p[i,t]

    po[1,i,t,2] <- 0

    po[1,i,t,3] <- 1-p[i,t]

    po[2,i,t,1] <- 0

    po[2,i,t,2] <- p[i,t]

    po[2,i,t,3] <- 1-p[i,t]

    po[3,i,t,1] <- 0

    po[3,i,t,2] <- 0

    po[3,i,t,3] <- 1

    } #t

    } #i

    # Likelihood

    for (i in 1:nind){

    # Define latent state at first capture

    z[i,f[i]] <- y[i,f[i]]

    for (t in (f[i]+1):n.occasions){

    # State process: draw S(t) given S(t-1)

    z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])

    # Observation process: draw O(t) given S(t)

    y[i,t] ~ dcat(po[z[i,t], i, t-1,])

    } #t

    } #i

    }

    ",fill = TRUE)

sink()

# Function to create known latent states z

[known.state.ms](http://known.state.ms/) <- function(ms, notseen){

  # notseen: label for ‘not seen?

  state <- ms

  state[state==notseen] <- NA

  for (i in 1:dim(ms)[1]){

    m <- min(which(![is.na](http://is.na/)(state[i,])))

    state[i,m] <- NA

  }

  return(state)

}

# Function to create initial values for unknown z

ms.init.z <- function(ch, f){

  for (i in 1:dim(ch)[1]){ch[i,1:f[i]] <- NA}

  states <- max(ch, na.rm = TRUE)

  known.states <- 1:(states-1)

  v <- which(ch==states)

  ch[-v] <- NA

  ch[v] <- sample(known.states, length(v), replace = TRUE)

  return(ch)}

# Bundle data

bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2], nind = dim(rCH)[1], z = [known.state.ms](http://known.state.ms/)(rCH, 3),g=length(unique(group)),group=group)

# Initial values

inits <- function(){list(mean.phi.g = runif(2, 0, 1), mean.psi.g = runif(2, 0, 1), mean.p.g = runif(2, 0, 1), z = ms.init.z(rCH, f))}

# Parameters monitored

parameters <- c("mean.phi.g", "mean.psi.g", "mean.p.g")

# MCMC settings

ni <- 10000

nt <- 6

nb <- 1000

nc <- 3

# Call OpenBUGS from R

multistate.single.multiple<-bugs(bugs.data,inits,parameters,"E:/corona\_multistate\_persistent+transient.txt", n.chains=nc, n.thin=nt, n.iter=ni, n.burnin=nb, debug=TRUE)

print(multistate.single.multiple, digits=3)

**R code S3. Simulation of epidemic models that contained scenario 1 to 6*.* The simulation of this R code generated probabilities of viral maintenance within the study population in scenario 1 to 6. R code 1 and 2 generated parameter values that were used in R code 3.**

library(deSolve)

library(mc2d)

week=12 # number of weeks

N=10000 # number of iteration

n=86 #number of individuals

prev=.278571 #mean prevalence from the CMR data

prop<- 7/23 #proportion of persistently infected bats

t=seq(0,week,by=1)

#########################

#SIRS model for 1 group model

SIRS1<- function(t, state, par) {

with(as.list(c(state, par)),{

dS<- -beta\*S\*I+omega\*R-mu\*S

dI<- beta\*S\*I-gamma\*I-mi\*I

dR<- +gamma\*I-omega\*R-mu\*R

floor(S)

floor(I)

floor(R)

return(list(c(dS, dI, dR)))

})}

state<-c(S=n\*(1-prev), I=n\*prev, R=0) #Initial number of bats in two group models

#Scenario 1 one-group

inf.sta1<-numeric() # number of infected bats at the end of simulation

persist1<-numeric() # the probability of viral persistence

for (i in 1:N){

par<-c(beta= 0.0104,# transmission rate

gamma=rpert(1,.3236,.5638,.8031 ), # recovery rate

omega=0.08333 , #immunity loosing rate

mu=rpert(1,0,1-.9848,1-.9268), # mortality rate of uninfected infected bats

mi=rpert(1,0,1-.9731,1-.8866) # mortality rate of infected infected bats

)

out<-ode(y=state, times=t, func=SIRS1, parms=par)

inf.sta1[i]<-floor(out[week,3]) #the number of infected ones in the stabilized state

persist1[i] <- if (inf.sta1[i] <= 1) 0 else 1 # infected less than 1 or larger than 1

}

out1<-out #ode simulation result of scenario 1

#########################

#SIRS model for 2 group model

SIRS2<- function(t, state, par) {

with(as.list(c(state, par)),{

dS<- -beta\*S\*(Ip+It)+omega\*R-mut\*S\*(1-f)-mup\*S\*f

dIp<- f\*beta\*S\*(Ip+It)-gamma.p\*Ip-mip\*Ip

dIt<- (1-f)\*beta\*S\*(Ip+It)-gamma.t\*It-mit\*It

dR<- gamma.p\*Ip+gamma.t\*It-omega\*R-mut\*R\*(1-f)-mup\*R\*f

floor(S)

floor(Ip)

floor(It)

floor(R)

return(list(c(dS, dIp, dIt, dR)))

})}

state<-c(S=n\*(1-prev), Ip=n\*prev\*prop,It=n\*prev\*(1-prop), R=0) #Initial number of bats in two group models

#Scenario 2: two group

inf.sta2<-numeric() # number of infected bats at the end of simulation

persist2<-numeric() # the probability of viral persistence

for (i in 1:N){

par<-c(beta= 0.0104 , gamma.p=rpert(1,0.121, 0.3354,0.6518 ), gamma.t=rpert(1,0.4985, 0.8582 ,0.9943 ),

omega= 0.0833333,

mup=rpert(1,0,1-.9848,1-.9268), mip=rpert(1,0,1-.9731,1-.8866),

mut=rpert(1,0,1-.9848,1-.9268), mit=rpert(1,0,1-.9731,1-.8866),f= 7/23)

out<-ode(y=state, times=t, func=SIRS2, parms=par)

out.mod<-matrix(NA, ncol=4, nrow=week+1)

out.mod[,1]<-out[,1];out.mod[,2]<-out[,2];out.mod[,3]<-out[,3]+out[,4];out.mod[,4]<-out[,5]

inf.sta2[i]<-floor(out.mod[week,3]) #the number of infected ones in the stabilized state

persist2[i] <- if (inf.sta2[i] <= 1) 0 else 1 # infected less than 1 or larger than 12.9815 2.9815

}

out2<-out #ode simulation result of scenario 2

#Scenario 3: 5 weeks of persistent infectious period

inf.sta3<-numeric()

persist3<-numeric()

for (i in 1:N){

par<-c(beta= 0.0104, gamma.p=1/5, gamma.t=rpert(1,0.4985, 0.8582 ,0.9943 ),omega=0.0833333 , mup=rpert(1,1-0.9983,1- 0.9383, 1-0.7183),

mip=rpert(1,1-1,1-.9748,1-.8711), mut=rpert(1,0,1-.9834,1-.9185), mit=rpert(1,1-1, 1-.9834,1-.9185), f= 7/23)

out<-ode(y=state, times=t, func=SIRS2, parms=par)

out.mod<-matrix(NA, ncol=4, nrow=week+1)

out.mod[,1]<-out[,1];out.mod[,2]<-out[,2];out.mod[,3]<-out[,3]+out[,4];out.mod[,4]<-out[,5]

inf.sta3[i]<-floor(out.mod[week,3]) #the number of infected ones in the stabilized state

persist3[i] <- if (inf.sta3[i] <= 1) 0 else 1 # infected less than 1 or larger than 1

}

out3<-out #ode simulation result of scenario 3

#Scenario 4: 7 weeks of persistent infectious period

inf.sta4<-numeric()

persist4<-numeric()

for (i in 1:N){

par<-c(beta=0.0104, gamma.p=1/7, gamma.t=rpert(1,0.4985, 0.8582 ,0.9943 ),omega=0.0833333, mup=rpert(1,1-0.9983,1- 0.9383, 1-0.7183),

mip=rpert(1,1-1,1-.9748,1-.8711), mut=rpert(1,0,1-.9834,1-.9185), mit=rpert(1,1-1, 1-.9834,1-.9185), f= 7/23)

out<-ode(y=state, times=t, func=SIRS2, parms=par)

out.mod<-matrix(NA, ncol=4, nrow=week+1)

out.mod[,1]<-out[,1];out.mod[,2]<-out[,2];out.mod[,3]<-out[,3]+out[,4];out.mod[,4]<-out[,5]

inf.sta4[i]<-floor(out.mod[week,3]) #the number of infected ones in the stabilized state

persist4[i] <- if (inf.sta4[i] <= 1) 0 else 1 # infected less than 1 or larger than 1

}

out4<-out #ode simulation result of scenario 4

#Scenario 5: 9 weeks of persistent infectious period

inf.sta5<-numeric()

persist5<-numeric()

for (i in 1:N){

par<-c(beta=0.0104, gamma.p=1/9, gamma.t=rpert(1,0.4985, 0.8582 ,0.9943 ), omega=0.0833333, mup=rpert(1,1-0.9983,1- 0.9383, 1-0.7183),

mip=rpert(1,1-1,1-.9748,1-.8711), mut=rpert(1,0,1-.9834,1-.9185), mit=rpert(1,1-1, 1-.9834,1-.9185), f= 7/23)

out<-ode(y=state, times=t, func=SIRS2, parms=par)

out.mod<-matrix(NA, ncol=4, nrow=week+1)

out.mod[,1]<-out[,1];out.mod[,2]<-out[,2];out.mod[,3]<-out[,3]+out[,4];out.mod[,4]<-out[,5]

inf.sta5[i]<-floor(out.mod[week,3]) #the number of infected ones in the stabilized state

persist5[i] <- if (inf.sta5[i] <= 1) 0 else 1 # infected less than 1 or larger than 1

}

out5<-out #ode simulation result of scenario 5

#Scenario 6: 11 weeks of persistent infectious period

inf.sta6<-numeric()

persist6<-numeric()

for (i in 1:N){

par<-c(beta=0.0104, gamma.p=1/11, gamma.t=rpert(1,0.4985, 0.8582 ,0.9943 ), omega=0.0833333 ,mup=rpert(1,1-0.9983,1- 0.9383, 1-0.7183),

mip=rpert(1,1-1,1-.9748,1-.8711), mut=rpert(1,0,1-.9834,1-.9185), mit=rpert(1,1-1, 1-.9834,1-.9185), f= 7/23)

out<-ode(y=state, times=t, func=SIRS2, parms=par)

out.mod<-matrix(NA, ncol=4, nrow=week+1)

out.mod[,1]<-out[,1];out.mod[,2]<-out[,2];out.mod[,3]<-out[,3]+out[,4];out.mod[,4]<-out[,5]

inf.sta6[i]<-floor(out.mod[week,3]) #the number of infected ones in the stabilized state

persist6[i] <- if (inf.sta6[i] <= 1) 0 else 1 # infected less than 1 or larger than 1

}

out6<-out #ode simulation result of scenario 6

###########################################################################

#the probability of viral persistence

p1<-sum(persist1)/length(persist1)

p2<-sum(persist2)/length(persist2)

p3<-sum(persist3)/length(persist3)

p4<-sum(persist4)/length(persist4)

p5<-sum(persist5)/length(persist5)

p6<-sum(persist6)/length(persist6)

p<-c(p1,p2,p3,p4,p5,p6);p

**REFERENCES**

(1) **Smith CS**, Australian bat coronaviruses (PhD thesis). Brisbane, QLD, Australia: University of Queensland, 2015. 137pp.

(2) **Vynnycky E, White R**. *An introduction to infectious disease modelling*: Oxford University Press, 2010.