**Lab 6: Vector-borne Disease Models**

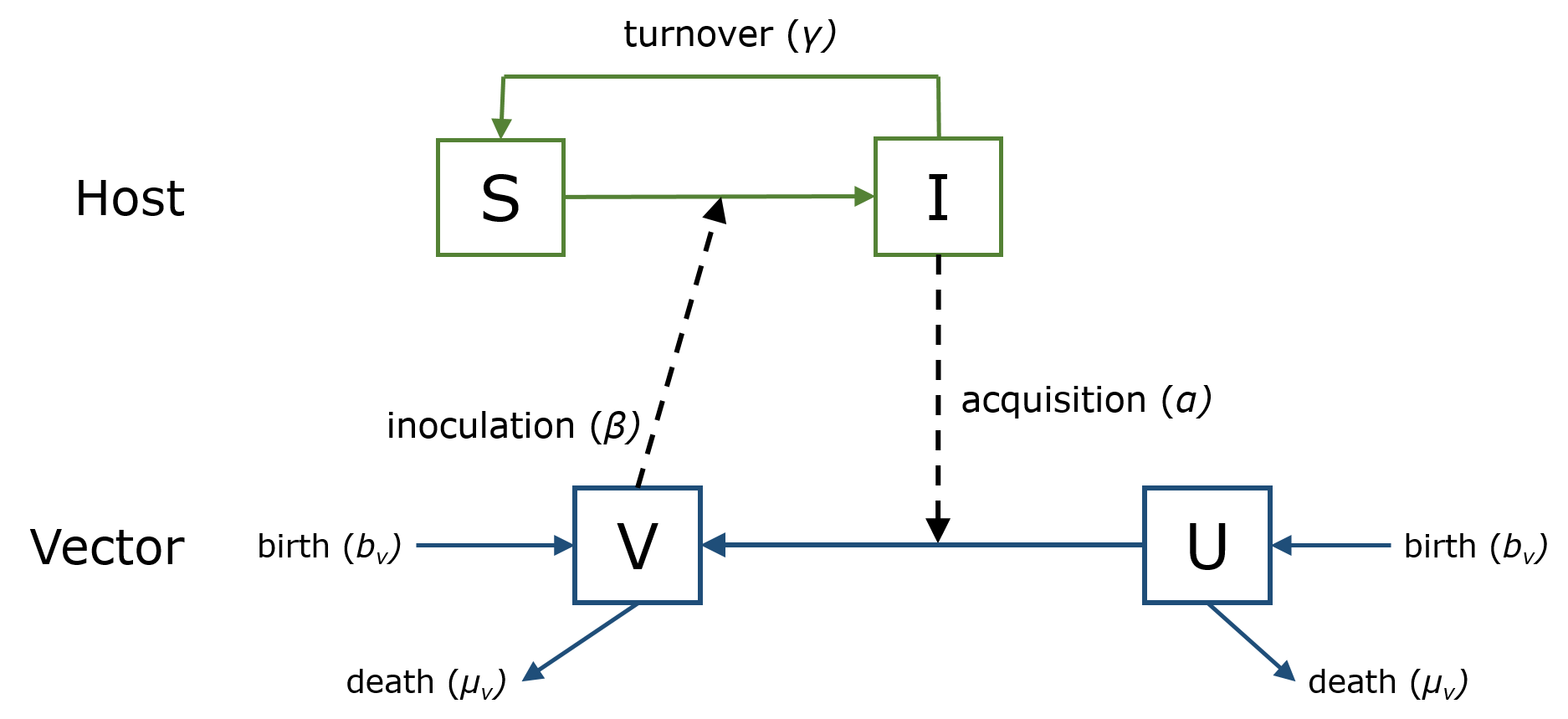
**Introduction**

In this lab, we’re going to cover continuous-time SI models that incorporate disease vectors[[1]](#footnote-1). The same mathematical concepts apply to vector SI models as more traditional SI models. However, the biology is more complicated because we have to consider two different species. Likewise, the same issues need to be considered with any modeling exercise:

1. Natural History. As with any biological model, the model should reflect what is understood about the natural history of the system. In vector disease models, we have to consider the biology of the host, the vector, and their interactions. We generally construct separate compartments for the hosts and vectors, and model their interactions through the transmission terms. We usually are interested in including susceptible (*S*) and infected (*I*) compartments for both the host and vector. We can also include recovered (*R*) and latent or exposed (*E*) compartments for host and/or vector.
2. Time period of the model. In nature, vectors commonly have much shorter generation times than hosts (e.g., mosquitoes vs. humans). Because of this, it is common for demographics (birth and death) of the vector to be included in the model and for demographics of the host to be ignored. When demographics of the host are ignored, it’s considered to be a closed host population. Of course, modeling host demographics is also possible.
3. Research question. As always, effective modeling balances the desire to include more biological processes—which almost always increases complexity—and the desire to make models understandable and elegant—which comes from simplifying models. The level of complexity should reflect the research questions that motivates the model.

**The basic vector SIS model**

The basic vector SIS model consists of 4 compartments or state variables: susceptible compartments for host (*S*) and vector (*U*) and infected compartments for host (*I*) and vector (*V*):



We have 5 parameters: inoculation rate (*β*), acquisition rate (*α*), host turnover rate (*γ*), vector birth rate (*bv*), and vector death rate (*µv*). The diagram (or qualitative model) above is translated into the follow system of ordinary differential equations:

**Questions**

1. In the SIS model above, what parameters represent the interaction between hosts and vectors?
2. We have not including birth and death of the host, only the vector. What assumption about time are we making about the system?
3. We’ve included birth rates (*bv*) for both the susceptible vectors (*U*) and infected vectors (*V)*. What assumption about the biology of vector-pathogen interactions are we making?

**Numerical simulation of the basic SIS model**

To analyze the model, we will use the numerical solver in the deSolve package. We first load the deSolve package, then set up a function that includes the differential equations.

> library(deSolve)

> SISvector <- function(dt, inits, parms){

+ with(as.list(c(parms, inits)), {

+ # Specify the model equations

+ dS = -beta\*S\*V/(S + I) + gamma\*I

+ dI = beta\*S\*V/(S + I) - gamma\*I

+ dU = -alpha\*U\*I/(S + I) + b\*U - mu\*U

+ dV = alpha\*U\*I/(S + I) + b\*V - mu\*V

+ # Specify which variables to output or return from function

+ return(list(c(dS, dI, dU, dV)))

+ })

+ }

We’ve named our function SISvector and we’ve defined 3 inputs that must go into the function: dt, inits, and parms. This is the familiar, right?

Next, we need to supply the function with the 3 inputs, dt, init, and parms. dt is defined using the seq() function:

# Define time period and intervals for simulation

> dt <- seq(0, 150, by = 1)

The input init is a vector of starting values for the state variables. parms is a vector of parameter values. I find it good practice to name the state variables and parameters in the R environment then concatenate them into vectors:

> # Define starting values for state variables

> S0 <- 99 # Susceptible hosts

> I0 <- 1 # Infected hosts

> U0 <- 200 # Susceptible vectors

> V0 <- 0 # Infective vectors

> inits <- c(S = S0, I = I0, U = U0, V = V0) # Concatenate starting values

> # Define parameters

> alpha <- 0.5 # acquisition rate

> beta <- 0.6 # inoculation rate

> gamma <- 0.5 # removal rate

> mu <- 0.15 # vector death rate

> b <- 0.1 # vector birth rate

> # Concatenate parameter values into a vector

> parms <- c(alpha = alpha, beta = beta, gamma = gamma, mu =

mu, b = b)

Note that the starting values have names of the state variables with 0’s after them: S0, I0, U0, and V0. Now let’s run the numerical solver ode() and examine the output:

> # Numerical simulation

> model.out <- as.data.frame(ode(y = inits, times = dt, func = SISvector, parms = parms))

> tail(model.out)

time S I U V

496 495 100 5.089461e-09 5.410662e-13 3.817096e-09

497 496 100 4.847106e-09 5.153011e-13 3.635330e-09

498 497 100 4.616292e-09 4.907630e-13 3.462219e-09

499 498 100 4.396468e-09 4.673933e-13 3.297351e-09

500 499 100 4.187112e-09 4.451365e-13 3.140334e-09

501 500 100 3.987726e-09 4.239395e-13 2.990795e-09

You’re results should match the table above. The entire table is a 151 row x 5 column table called model.out. We can plot results easily with the matplot() function. All we need to do is supply the model.out table excluding the first column, which is time.

# Time series plot

matplot(model.out[,-1],

# Some additional arguments to make the plot look nice

type = "l", xlab = "Time", ylab = "Population",

lty = 1:2, col = c("green", "green", "blue", "blue"), lwd = 2)

legend("topright", c("S", "I", "U", "V"),

lty = 1:2, col = c("green", "green", "blue", "blue"), box.lwd = 0, lwd = 2)

**Exercise**

Paste the resulting plot here. Describe the dynamics of the system. Does the pathogen become endemic in the system, or does the epidemic have a peak and then die out?

**Investigating the effect of vector population growth**

Next we will investigate the effects of vector population growth on disease prevalence. We would like to calculate the Infected host density (*I*) over a range of vector birth rates over a long time period. We will do this with the sapply() function. sapply() is similar to apply() which you learned about earlier, except that it simplifies the output. We first need to make a vector of values for the vector birth rate parameter b. We also need to extend the time period that we’re simulating:

> # Vector of birth rates

> bVec <- seq(0, 0.2, by = 0.01)

> # Define new time period and intervals for simulation

> dt <- seq(0, 500, by = 1)

Next we need to create a function that can pass these birth rate values to the ode() solver automatically. We’ll call this function SISVectorFunc:

> SISVectorFunc <- function(x){

+ # Pass the variable x to the "parms" vector, to represent birth rate values

+ parms <- c(alpha = alpha, beta = beta, gamma = gamma, mu = mu, b = x)

+ # Run ode() on the SISVector model using the new birth rate value

+ model.out2 <- as.data.frame(ode(y = inits, times = dt, func = SISvector, parms = parms))

+ # Extract the value for Infected hosts (I) from the last time point (t = 500)

+ # And calculate the % of infected hosts

+ infectFinal <- (model.out2$I[501]/(S0 + I0))\*100 # % of host population infected

+ # Return the birth rate value and the % of infected hosts

+ return(c(x, infectFinal))

+ }

Finally we call sapply():

> # Run sapply()

> bResults <- as.data.frame(t(sapply(bVec, SISVectorFunc, simplify = TRUE)))

> names(bResults) <- c("b", "I")

In the first line (after the comment), we have 3 nested functions. First, sapply() to “apply” each element of bVec to the function SISVectorFunc. The output of sapply() is a 2 row x 21 column matrix. Second, the t() function transposes this matrix to become a 21 row x 2 column matrix. Third, as.data.frame turns the matrix into a data.frame object. We then name the columns of this data.frame using the names() function.

So to review, using sapply(), we’ve run the ode() solver 21 times with different values for vector birth rate. We then extracted only the infected host results. Finally we made a table with just the vector birth rate values and the resulting % of hosts infected with the pathogen.

We can now easily plot this data:

> plot(I ~ b, data = bResults,

+ # More arguments to make the plot look nice

+ type = "b", lwd = 2, col = "green",

+ ylab = "Percent infected hosts",

+ xlab = "Vector birth rate (b)")

**Exercise**

Paste the resulting plot here. Describe the dynamics of the system. What happens at low birth rates? What happens at high birth rates? Is there a threshold birth rate at which the pathogen becomes endemic? What is this value? What is the value of the vector death rate, *µv*? (Hint: type mu into the R console.) How does the threshold birth rate value compare to the death rate? What is the overall impact of increasing the vector birth rate (and thus population size)?

1. We will be dealing with two very different kinds of vectors: disease vectors (e.g., mosquitoes) and mathematical vectors (e.g., c(1,2,3,4)); make sure not to get them mixed up! [↑](#footnote-ref-1)