Report05.03.2025

Reed Scott

# Introduction

In the last update on this project we had agreed upon a method for simulating the species richness ~ abundance relationship when building our metacommunities. I have simulated 100 meta-communities for us to explore using the rules and parameters we have previously established

In the most recent addition to our manuscript (Metadisease1.3, sent in the same email as this document) I define T and (called R and B Mark’s 2020 supplementals). To start, here I provide the code necessary to calculate the T and matrices. I then show how I believe these functions should be used in our simulated metacommunities to calculate R0,L . My goal is that, assuming you approve of these methods, we can run simulation, calculate R0L for each metacommunity and explore the results.

# The Functions:

## The R (or T) matrix

To build the R matrix, we require 5 pieces of information: , I, N, S, and P. S defines the number of possible species in the system and P defines the number of patches in the system. is an S x S matrix define transmission rates: is intraspecific transmission while is interspecific transmission. I is an S x P matrix defining the abundance of infectious individuals in each patch. N is also an S x P matrix, but defines the total abundance of each species in each patch. Based on the method of calculating T and TP we can calculate R using the following code:

build\_R\_with\_data\_freq <- function(beta,I,N,S,P) {  
 fullR <- matrix(data = 0, nrow = P\*S, ncol = P\*S)  
 for (p in 1:P) {  
 Rmat <- matrix(data = 0, nrow = S, ncol = S)  
 for (s in 1:S){  
 for (i in 1:S){  
 Rmat[i,s] <- (beta[s,i]\*I[p,i])/sum(N[p,])  
 }  
 }  
 start <- ifelse(p == 1, p,(S\*p)-(S-1))  
 stop <- start + S - 1  
 fullR[start:stop, start:stop] <- Rmat  
 }  
 return(fullR)  
}

Here, we are defining the full matrix R, ofr”fullR” as a matrix with dimensions {P\*S,P\*S}. We then define a submatrix Rmat, for each patch, where where is the transmission coefficient between infectious individuals of species i and susceptible individuals of species s.

## The B (or matrix)

While the R matrix had to change its calculations compared to Wilber 2020, the B matrix remains largely unchanged. In the Metadisease1.3 we define 2 submatrices of the B matrix, D and E (originally defined in Mark’s 2020 supplementals). In addition to parameters S and P we also define the B matrix using Cmat (the matrix of connectivity between patches), (the dispersal rate of a species), and b (the loss rate of the disease, defined by recovery + death).

build\_B\_with\_data <- function(Cmat, phi, b, S, P) {  
 Cmat\_axis <- colSums(Cmat)  
 diag\_list <- vector("list",P^2)  
 diag\_list\_names <- array(dim = c(sqrt(length(diag\_list)), sqrt(length(diag\_list))))  
 for (p in 1:P) { #loop over columns  
 for (j in 1:P) { #loop over rows  
 diag\_list\_names[j,p] <- paste(p,j,sep="\_")  
 }  
 }  
 diag\_list\_names <- as.vector(diag\_list\_names)  
 names(diag\_list) <- diag\_list\_names  
 for (p in 1:P) { #loop over columns  
 for (j in 1:P) { #loop over rows  
 tZ <- matrix(0,nrow = S, ncol = S)  
 new\_diag <- array(dim = S)  
 new\_diag <- if(p == j){  
 #build the diagonal matrix  
 (-1)\*b - phi\*Cmat\_axis[p]  
 } else{  
 phi\*Cmat[j,p]  
 }  
 diag(tZ) <- new\_diag  
 x <- paste(p,j,sep="\_")  
 diag\_list[[x]] <- tZ   
 }  
 }  
 diag\_matrix <- matrix(diag\_list, nrow = P, ncol = P)  
 tB <- list()  
 for (j in 1:P) {  
 tB[[j]] <- matrix(data = unlist(diag\_matrix[j,]), nrow = S\*P, ncol = S, byrow = T) #looks like this worked!  
 }  
   
 B <- do.call(cbind, tB[1:P]) #need to determine if this is what B should look like  
 return(B)   
}

I’d say the most important thing to pay attention to is the if…else statement. If p = j, then we are along the diagonal of the matrix and thus use the formula . If we are off the diagonal we use the formula .

## The K matrix

The final product we need to calculate R0,L is the matrix K. this is defined by

We already have the process for defining R and B. Now we just need to use matrix multiplication to get K.

landscape\_R0\_freq <- function(beta,I,N,Cmat,b, phi,S,P){  
 R <- build\_R\_with\_data\_freq(beta = beta,  
 I = I,  
 N = N,  
 S = S,  
 P = P)  
 B <- build\_B\_with\_data(Cmat = Cmat,  
 phi = phi,  
 b = b,  
 S = S,  
 P = P)  
 K <- R %\*% solve(-1\*B)  
 return(list(K = K, R = R, B = B))  
}

## An example

Let’s take an example and see if it works as expected. We’ll define a 2 species, 2 patch system. We’ll randomly define values for b, , , and N. We’ll define S (abundance of susceptible individuals) as a proportion of N and I = N-S. With all of that defined we can then run each function and see if they work.

Spp <- 2 #number of species  
Patches <- 2 #number of patches  
b <- rnorm(n = Spp, mean = 1, sd = 0.1)  
phi <- rnorm(n = 2, mean = 0.5, sd = 0.1)  
#transmission  
beta <- matrix(data = NA, nrow = Spp, ncol = Spp)  
for (i in 1:nrow(beta)) {  
 for (j in 1:ncol(beta)) {  
 beta[i,j] <- ifelse(i == j, rbeta(n = 1, shape1 = 2, shape2 = 10),rbeta(n = 1, shape1 = 1, shape2 = 1))  
 }  
}  
#connectivity  
Cmat <- matrix(data = rnorm(n = Patches^2, mean = 0.5, sd = 0.1),  
 nrow = Patches,  
 ncol = Patches)  
N <- matrix(rep(NA,4), nrow = 2, ncol = 2)  
N[,1] <- runif(2, min = 10, max = 20)  
for(i in 1:2){  
 N[i,2] <- runif(1, min = 1, max = N[i,1])  
}  
S <- N\*c(0.8,0.95)  
I <- N - S  
r\_freq <- build\_R\_with\_data\_freq(beta = beta,  
 I = I,  
 N = N,  
 S = Spp,  
 P = Patches)  
B <- build\_B\_with\_data(Cmat = Cmat,  
 phi = phi,  
 b = b,   
 S = Spp,  
 P = Patches)  
# Not currently working. Need to probably adjust B matrix  
K\_freq <- landscape\_R0\_freq(beta = beta,  
 I = I,  
 N = N,  
 Cmat = Cmat,  
 b = b,  
 phi = phi,  
 S = Spp,  
 P = Patches)

Now let’s check each and see if they worked:

r\_freq

[,1] [,2] [,3] [,4]  
[1,] 0.02966232 0.091686107 0.000000000 0.000000000  
[2,] 0.02211440 0.007612382 0.000000000 0.000000000  
[3,] 0.00000000 0.000000000 0.006044955 0.018684928  
[4,] 0.00000000 0.000000000 0.007338430 0.002526089

B

[,1] [,2] [,3] [,4]  
[1,] -1.3992400 0.000000 0.2500682 0.0000000  
[2,] 0.0000000 -1.254567 0.0000000 0.1090908  
[3,] 0.2751875 0.000000 -1.3979568 0.0000000  
[4,] 0.0000000 0.120049 0.0000000 -1.2540068

K\_freq[[1]]

[,1] [,2] [,3] [,4]  
[1,] 0.021971861 0.0736953671 0.003930354 0.0064110401  
[2,] 0.016380863 0.0061186732 0.002930229 0.0005322866  
[3,] 0.000881434 0.0014377608 0.004481807 0.0150252569  
[4,] 0.001070040 0.0001943766 0.005440807 0.0020313237

Looks like about what I’d expect! And if we take the max eigenvalue of that, that should be our R0,L

# Applying functions to our metacommunities

Our goal in this project is to assess how **dissimilarity of communities** affects R0,L. Previously, we have talked about how to simulate community structure so I’m not going to discuss it much here. You can see the methods section of Metadisease1.3, “Establishing meta-communities” and Figure 1 to see that this uses a saturated curve.

Here, I provide the methods in terms of the code. I am using I am using a two patch meta-community to start, although I think this can be upscaled relatively easily. We have to set the number of patches (2), number of species (6), birth (0 because this is a single season model), death (d), recovery (v), (dispersal), (transmission coefficient), and c(connectivity). I do not show the code here but am happy to continue discussion of parameter value selection.

I do want to take a second to discuss the simulation of . Originally, I had defined both intraspecific and interspecific using the beta distribution. however, when I did, I found that the values of interspecific were incredibly small. This could easily be adjusted, and for now I have arbitrarily defined interspecific as 0.85\*intraspecific . However, I think this could showcase how important the specifications for are (and that I may need to do a bit more literature review).

trans\_rate <- function(n = 1,x = seq(0,1,length = 100),a,b){  
 trans <- rbeta(n = n, shape1 = a, shape2 = b)  
 return(trans)  
}  
  
  
# PREG  
intra\_PREG <- trans\_rate(a = 4,b = 2)  
inter\_PREG <- intra\_PREG\*0.85 #trans\_rate(a = 4, b = 2.5)  
  
# TGRAN  
intra\_TGRAN <- trans\_rate(a = 4, b = 2.25)  
inter\_TGRAN <- intra\_TGRAN\*0.85 #trans\_rate(a = 4, b = 2.5)  
  
#TTOR  
intra\_TTOR <- trans\_rate(a = 3, b = 2.5)  
inter\_TTOR <- intra\_TTOR\*0.85 #trans\_rate(a = 3, b = 2.75)  
  
#ABOR  
intra\_ABOR <- trans\_rate(a = 2.5, b = 2.75)  
inter\_ABOR <- intra\_ABOR\*0.85 #trans\_rate(a = 2.5, b = 3.0)  
  
#RCAT  
intra\_RCAT <- trans\_rate(a = 2.0, b = 3.0)  
inter\_RCAT <- intra\_RCAT\*0.85 #trans\_rate(a = 2.0, b = 3.25)  
  
#RDRAY  
intra\_RDRAY <- trans\_rate(a = 1.5, b = 3.25)  
inter\_RDRAY <- intra\_RDRAY\*0.85#trans\_rate(a = 1.5, b = 3.5)  
  
beta <- matrix(data = NA, nrow = num\_spp, ncol = num\_spp)  
for (i in 1:nrow(beta)) {  
 for (j in 1:ncol(beta)) {  
 beta[i,j] <- if(i == 1 & j == 1){  
 intra\_PREG}else if(i != 1 & j == 1){  
 inter\_PREG} else if(i == 2 & j == 2){  
 intra\_TGRAN} else if(i != 2 & j == 2){  
 inter\_TGRAN} else if(i == 3 & j == 3){  
 intra\_TTOR} else if(i != 3 & j == 3){  
 inter\_TTOR} else if(i == 4 & j == 4){  
 intra\_ABOR} else if(i != 4 & j == 4){  
 inter\_ABOR} else if(i == 5 & j == 5){  
 intra\_RCAT} else if(i != 5 & j ==5){  
 inter\_RCAT} else if(i == 6 & j == 6){  
 intra\_RDRAY} else if(i != 6 & j == 6){  
 inter\_RDRAY}  
 }  
}

Now with that in mind I can show the for loops used to actually calculate R0,L . I first defined a dataframe that we will use to store our results. I then ran a for loop where for each metacommunity “a” we established the total (N) susceptible (S) Infectious (I) abundance for each species. We can calculate total abundance for each metacommunity (TotalAbundance) as well as the abundance of each species. We can then measure **dissimilarity** of the two patches within the metacommunity (which I call beta\_diversity… that may not be correct). I also measure gamma diversity for good measure. With all of this, we then have the necessary variables to calculate R0,L using the landscape\_R0\_freq() function. And I believe… that’s everything required. That should be the whole model.

result <- data.frame(matrix(data = NA, nrow = length(meta\_comm\_list), ncol = 17))  
colnames(result) <- c("TotalAbundance","PREG","TGRAN","TTOR","ABOR","RCAT","RDRAY","BetaDiversity","Gamma\_diversity","LandscapeR0", "MetaCommID")  
  
for (a in 1:length(meta\_comm\_list)) {  
 S <- meta\_comm\_list[[a]][,1:6]\*c(0.5,0.6,0.70,0.8,0.80,.9) #value of susceptibles  
 I <- meta\_comm\_list[[a]][,1:6] - S #value of infecteds  
 N <- S+I #total pop of a patch  
 S <- as.matrix(S)  
 I <- as.matrix(I)  
 N <- as.matrix(N)  
 N\_meta <- colSums(N)  
 b <- v + d # total loss rate  
  
 # Calculate variables  
 result[a,1] <- sum(N) #total abundance  
 #abdunance of each species  
 for (i in 2:(1+num\_spp)) {  
 result[a,i] <- sum(N[,i-1])  
 result[a,(i+num\_spp)] <- result[a,i]/result[a,1]  
 }  
   
 #add beta diversity  
 beta\_diversity <- betadiver(N, method = 'sor')  
 result[a,14] <- mean(beta\_diversity, na.rm =T) #beta diversity  
 result[a,15] <- sum(ifelse(result[a,1:6] > 0, 1,0)) #gamma diversity  
   
 #calculate landscape R0  
 r0\_landscape <- landscape\_R0\_freq(beta = beta,  
 I = I,  
 N = N,  
 Cmat = c,  
 b = b,  
 phi = phi,  
 S = num\_spp,  
 P = num\_patches)  
   
 result[a,16] <- max(abs(eigen(r0\_landscape[[1]])$values)) #landscape R0  
 result[a,17] <- a #metacommunity ID  
}

# Next steps

First, I believe that I need to reassess how transmission is determined. Although I don’t show it here, I redid some of the sensitivity analysis I did earlier and found that R0,L is strongly correlated with . As I said above, this could include some further literature review.

Additionally, my chosen values for S and I are currently arbitrary. I think we’re far enough along that I need to do some digging to determine realistic values of prevalence for each host species. I also think that this has led to a change in an underlying question you both asked earlier: I think that this is going to be a simulation assessing a system at equilibrium and not necessarily exploring invasion dynamics.

Now, assuming my math is correct, and that my implementation via code is correct, I believe that the next step is analysis of the model. We could just use a GLM. Although Mihaljevic et al. (2014) used a GAM I’m not sure we’re expecting any non-linear relationship so not sure that’s a good fit? I am open to either though. And I think we could fit 3 models:

1. a null model
2. ~ and
3. ~

and see how the models perform. I think it would also make sense to include graphs of ~ and ~ .

Mihaljevic, Joseph R., Maxwell B. Joseph, Sarah A. Orlofske, and Sara H. Paull. 2014. “The Scaling of Host Density with Richness Affects the Direction, Shape, and Detectability of Diversity-Disease Relationships.” Edited by Delmiro Fernandez-Reyes. *PLoS ONE* 9 (5): e97812. <https://doi.org/10.1371/journal.pone.0097812>.