Homework Week 8

Kerri Luttrell

3/3/2021

PART FOUR: Homework

To make sure you're set up, run the code from lab again from top to bottom. Include all the set up code in this chunk, to make sure you can generate a knitted document:

```
#variable set up
r <- 1  # growth rate of H
K <- 20         # carrying capacity of H
gamma <- .5     # positive effect of P on H's carrying capacity
a <- .01        # exploitation of H by P
b <- 1          # growth rate of P
m <- 1          # density-independent mortality of P
e <- 1          # factor weighting density-dependent mortality of P
tset <- seq(from = 0, to = 100, length.out = 5000)

a_mut <- 0
a_par <- 0.05

# create holding vectors and store initial conditions
H.simu.mut <- NaN*tset; H.simu.mut[1] <- 1
P.simu.mut <- NaN*tset; P.simu.mut[1] <- 1

# for each element i from the second to the last in tset
for(i in 2:length(tset)){
  # calculate change in time
    dt <- tset[i] - tset[i-1]

    # store dummy variables
    H <- H.simu.mut[i-1]
    P <- P.simu.mut[i-1]

    # calculate change in population size
    dH <- ( r*H*(1-H/(K+gamma*P))-a_mut*H*P )*dt
    dP <- ( b*H*P - m*P*(1+e*P) )*dt

    # calculate total population size
    H.simu.mut[i] <- H + dH
    P.simu.mut[i] <- P + dP
}
```

```
# store H and P at equilibrium
Hmax.mut <- H.simu.mut[length(tset)]
Pmax.mut <- P.simu.mut[length(tset)]

H.simu.par <- NaN*tset; H.simu.par[1] <- 1
P.simu.par <- NaN*tset; P.simu.par[1] <- 1

# for each element i from the second to the last in tset
for(i in 2:length(tset)){
  # calculate change in time
    dt <- tset[i] - tset[i-1]

    # store dummy variables
    H <- H.simu.par[i-1]
    P <- P.simu.par[i-1]

    # calculate change in population size
    dH <- ( r*H*(1-H/(K+gamma*P))-a_par*H*P )*dt
    dP <- ( b*H*P - m*P*(1+e*P) )*dt

    # calculate total population size
    H.simu.par[i] <- H + dH
    P.simu.par[i] <- P + dP
}
```

```
# store H and P at equilibrium
Hmax.mut <- H.simu.mut[length(tset)]
Pmax.mut <- P.simu.mut[length(tset)]
# store H and P at equilibrium
Hmax.par <- H.simu.par[length(tset)]
Pmax.par <- P.simu.par[length(tset)]


## a. Synchronous dispersal

Hcol <- 'red'
Pcol <- 'royalblue'


D_H <- 0.001
D_P <- 0.001

Xset <- seq(from = 1, to = 20)

# create a holding matrix for H and fill initial conditions
H.simu4 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
H.simu4[1,] <- c(Hmax.mut,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu4 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
P.simu4[1,] <- c(Pmax.mut,rep(0,length(Xset)-1))
```

```
# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu4[i-1, j]
        H <- H.simu4[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu4[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu4[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu4[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu4[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu4[i-1,j-1] + P.simu4[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu4[i-1,j-1] + H.simu4[i-1,j+1] - 2*H)
        }
        # calculate total population size and store in holding matrix
        P.simu4[i, j] <- P + dP
        H.simu4[i, j] <- H + dH
    }
}
```

```
D_H <- 0.01
D_P <- 0.001

# create a holding matrix for H and fill initial conditions
H.simu5 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
H.simu5[1,] <- c(Hmax.mut,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu5 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
P.simu5[1,] <- c(Pmax.mut,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu5[i-1, j]
        H <- H.simu5[i-1, j]
```

```
        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu5[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu5[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu5[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu5[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu5[i-1,j-1] + P.simu5[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu5[i-1,j-1] + H.simu5[i-1,j+1] - 2*H)
        }
        # calculate total population size and store in holding matrix
        P.simu5[i, j] <- P + dP
        H.simu5[i, j] <- H + dH
    }
}
```

```
D_H <- 0.001
D_P <- 0.1

# create a holding matrix for H and fill initial conditions
H.simu6 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
H.simu6[1,] <- c(Hmax.mut,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu6 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
P.simu6[1,] <- c(Pmax.mut,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu6[i-1, j]
        H <- H.simu6[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu6[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu6[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu6[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu6[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu6[i-1,j-1] + P.simu6[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - a_mut*H*P + D_H*(H.simu6[i-1,j-1] + H.simu6[i-1,j+1] - 2*H)
        }
```

```
        # calculate total population size and store in holding matrix
        P.simu6[i, j] <- P + dP
        H.simu6[i, j] <- H + dH
    }
}
```

```
tail(H.simu4)
```

```
##         [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [4995,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4996,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4997,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4998,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4999,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [5000,]   39   39   39   39   39   39   39   39   39    39    39    39    39
##         [,14]    [,15]    [,16]    [,17]    [,18]    [,19]    [,20]
## [4995,]    39 38.99999 38.99993 38.99937 38.99421 38.94678 38.53567
## [4996,]    39 38.99999 38.99993 38.99938 38.99427 38.94730 38.54023
## [4997,]    39 38.99999 38.99993 38.99939 38.99433 38.94781 38.54475
## [4998,]    39 38.99999 38.99993 38.99939 38.99438 38.94832 38.54923
## [4999,]    39 38.99999 38.99993 38.99940 38.99444 38.94882 38.55366
## [5000,]    39 38.99999 38.99994 38.99940 38.99449 38.94932 38.55805
```

```
tail(H.simu6)
```

```
##         [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [4995,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4996,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4997,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4998,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [4999,]   39   39   39   39   39   39   39   39   39    39    39    39    39
## [5000,]   39   39   39   39   39   39   39   39   39    39    39    39    39
##         [,14]    [,15]    [,16]    [,17]    [,18]    [,19]    [,20]
## [4995,]    39 38.99999 38.99992 38.99928 38.99360 38.94382 38.55259
## [4996,]    39 38.99999 38.99992 38.99928 38.99366 38.94436 38.55700
## [4997,]    39 38.99999 38.99992 38.99929 38.99372 38.94490 38.56136
## [4998,]    39 38.99999 38.99992 38.99930 38.99378 38.94542 38.56568
## [4999,]    39 38.99999 38.99992 38.99930 38.99384 38.94595 38.56996
## [5000,]    39 38.99999 38.99992 38.99931 38.99390 38.94646 38.57420
```

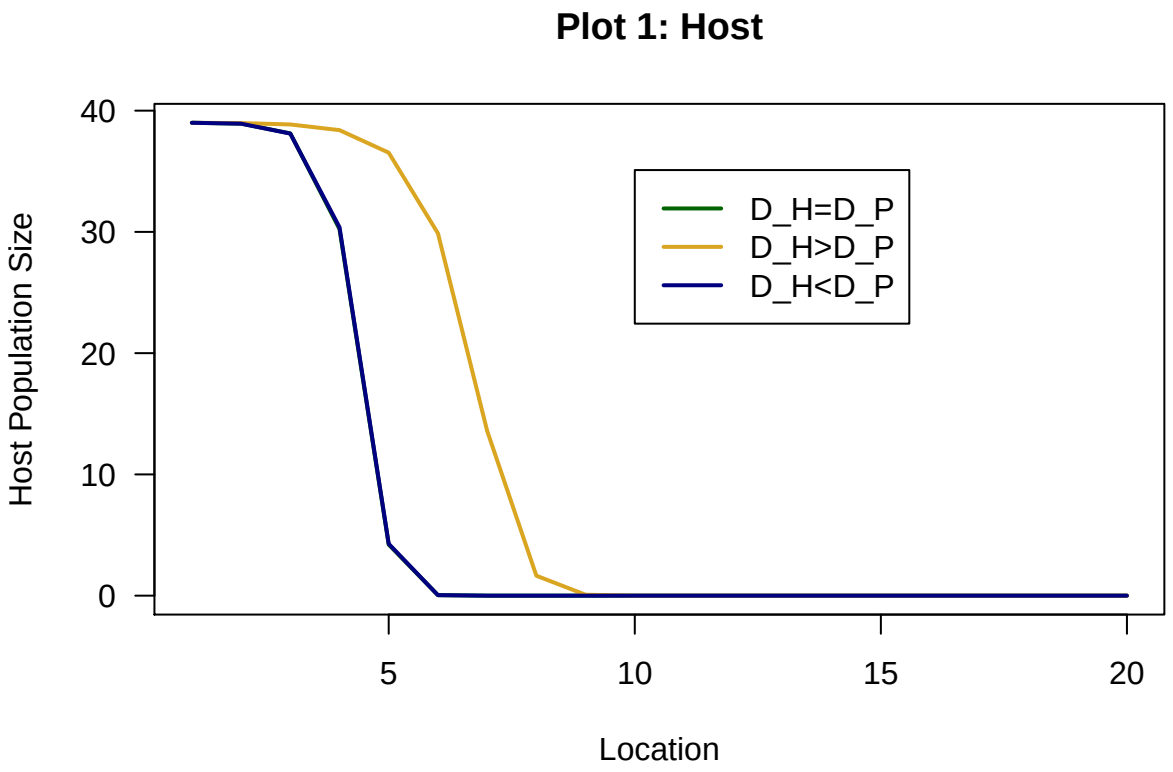
Question 1: Mutualist-enhanced invasion

For this part of the homework, the partner is a mutualist (a = a_mut = 0)

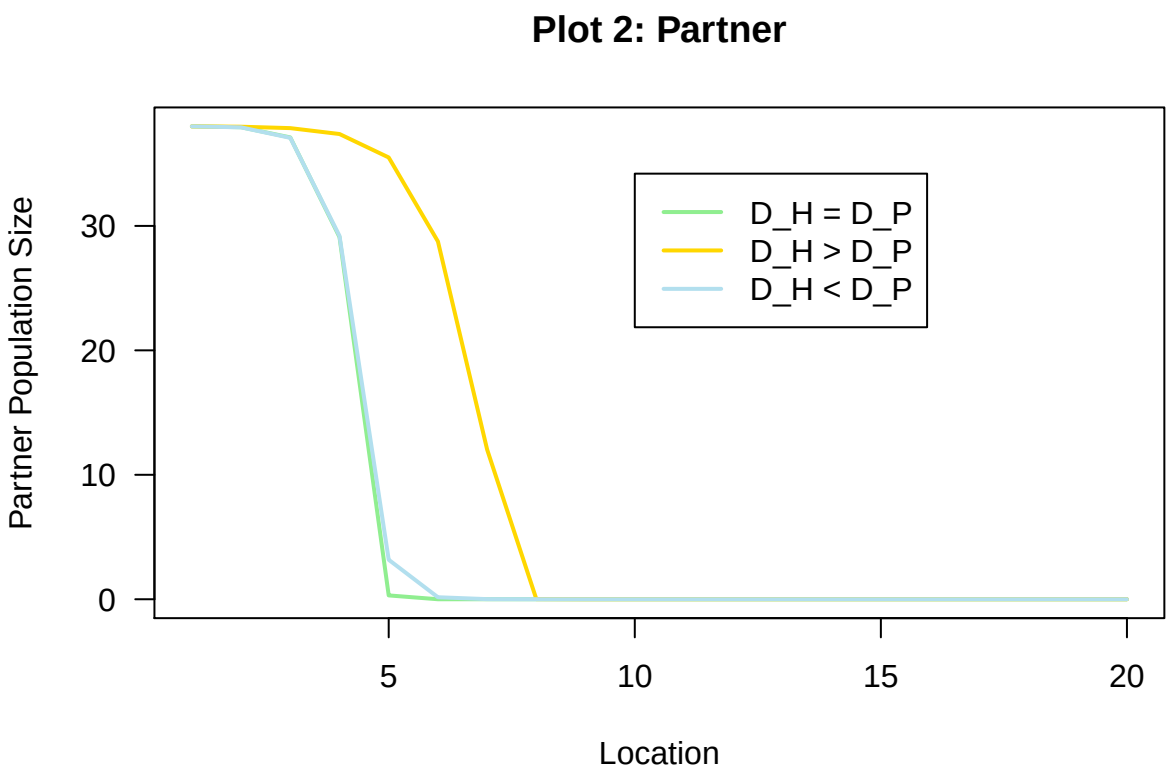
a. Make two plots that compare the host (plot 1) and partner's (plot 2) invasion fronts on Day 1000 for (a) D_H = D_P, (b) D_H > D_P, and (c) D_H < D_P. Be sure to include a legend!

/2 axes for host and partner graphs
/6 three scenarios for host and partner
/2 legends for host and partner
= /10 points total

```
plot(x = Xset, y = H.simu4[1000,], type = 'l', lwd = 2, col= 'darkgreen' , xlab='Location',ylab='Host P
lines(x = Xset, y = H.simu5[1000,],lwd=2,col='goldenrod')
lines(x = Xset, y = H.simu6[1000,],lwd=2,col='navy')
legend(x = 10, y = Hmax.mut*.9, legend=c('D_H=D_P', 'D_H>D_P','D_H<D_P'), lwd=2, col=c('darkgreen','gol
```




```
plot(x = Xset, y = P.simu4[1000,], type = 'l', lwd = 2, col='lightgreen', xlab='Location',ylab='Partner
lines(x = Xset, y = P.simu5[1000,],lwd=2,col='gold')
lines(x = Xset, y = P.simu6[1000,],lwd=2,col='lightblue2')
legend(x = 10, y = Pmax.mut*.9, legend=c('D_H = D_P','D_H > D_P', 'D_H < D_P'),lwd=2,col=c('lightgreen'
```



b. Under what scenario does invasion happen fastest?

Invasion happens fastest under host first dispersal (D_H > D_P). This is visualized by seeing the right shift in the graph at day 1000 compared to the other two, meaning individuals of the population had dispersed further by day 1000.

= /2 points total

c. Why does (or does not) a faster partner dispersal rate accelerate invasion?

The faster partner dispersal does not accelerate rate because the partner is reliant on its mutualistic interaction with the host (as can be seen from the +bHP term in its growth equation). Without a positive host population in the new locations that P disperses into, there will be no positive term in its growth equation and the pioneer population will soon die out.

= /2 points total

Question 2: Parasite-limited invasion

For this part of the homework, the partner is a parasite (a = a_par = 0.05).

```
#setup chunk
a_par = 0.05
```

Run three simulations (remember to change the number for your `.simu` matrices!):
a. Same-speed: D_H = D_P = 0.001

```
D_H <- 0.001
D_P <- 0.001

Xset <- seq(from = 1, to = 20)

# create a holding matrix for H and fill initial conditions
H.simu7 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
H.simu7[1,] <- c(Hmax.par,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu7 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
P.simu7[1,] <- c(Pmax.par,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu7[i-1, j]
        H <- H.simu7[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu7[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu7[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu7[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu7[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu7[i-1,j-1] + P.simu7[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu7[i-1,j-1] + H.simu7[i-1,j+1] - 2*H)
        }
        # calculate total population size and store in holding matrix
        P.simu7[i, j] <- P + dP
        H.simu7[i, j] <- H + dH
    }
}
```

b. Faster host: D_H = 0.01; D_P = 0.001

```
D_H <- 0.01
D_P <- 0.001

# create a holding matrix for H and fill initial conditions
H.simu8 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
H.simu8[1,] <- c(Hmax.par,rep(0,length(Xset)-1))
```

```
# create a holding matrix for P and fill initial conditions
P.simu8 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
P.simu8[1,] <- c(Pmax.par,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu8[i-1, j]
        H <- H.simu8[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu8[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu8[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu8[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu8[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu8[i-1,j-1] + P.simu8[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu8[i-1,j-1] + H.simu8[i-1,j+1] - 2*H) )
        }
        # calculate total population size and store in holding matrix
        P.simu8[i, j] <- P + dP
        H.simu8[i, j] <- H + dH
    }
}
```

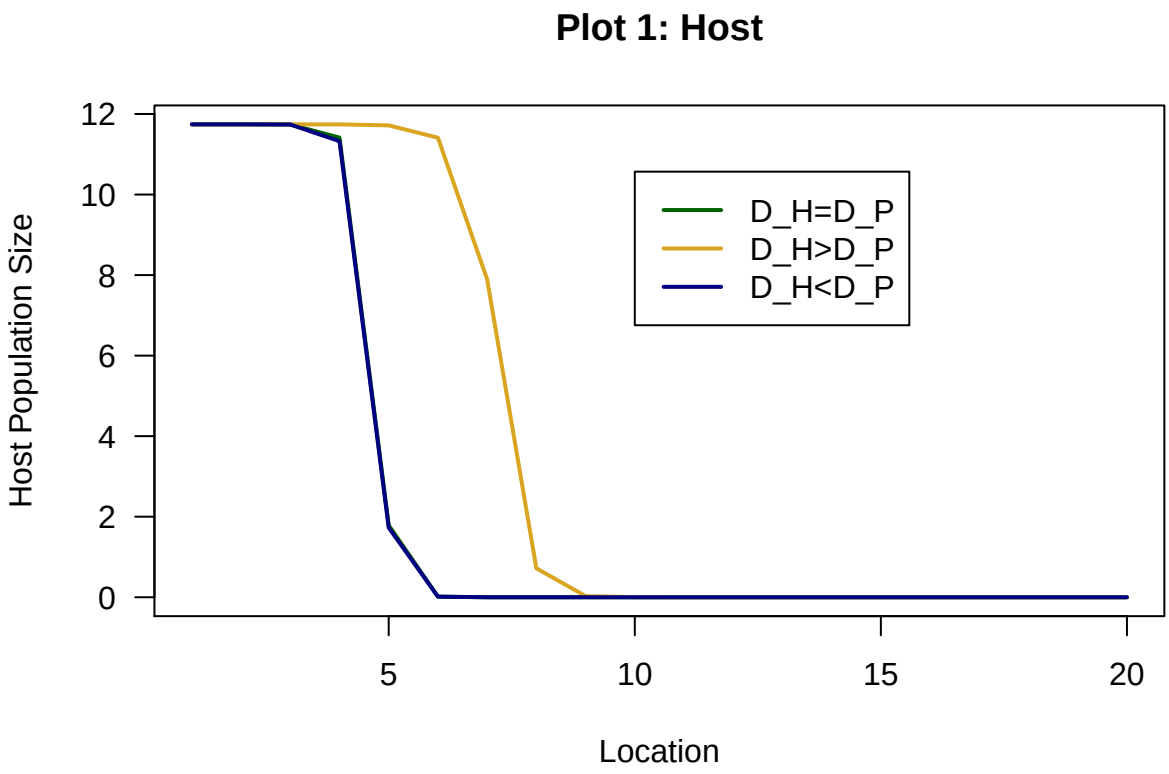
c. Faster parasite: D_H = 0.001; D_P = 0.01

```
D_H <- 0.001
D_P <- 0.01

# create a holding matrix for H and fill initial conditions
H.simu9 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
H.simu9[1,] <- c(Hmax.par,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu9 <- matrix(data=NA,
                  nrow = length(tset), ncol = length(Xset))
P.simu9[1,] <- c(Pmax.par,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
```

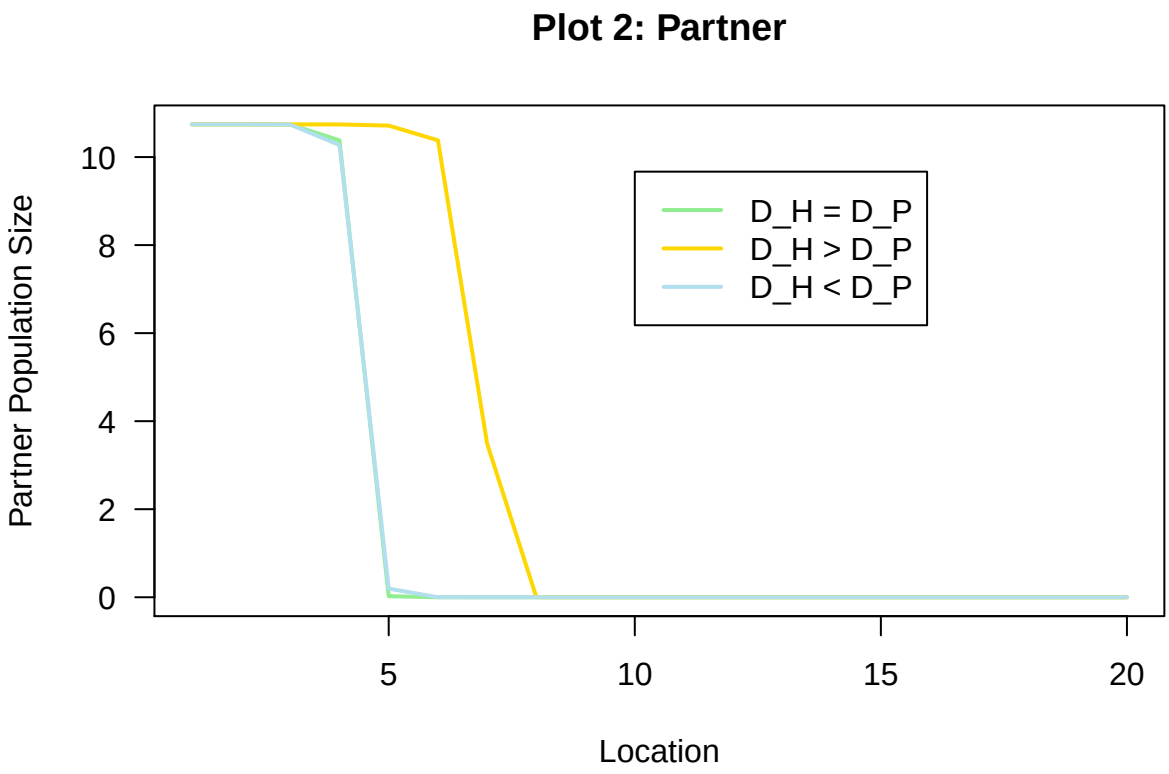
```
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu9[i-1, j]
        H <- H.simu9[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu9[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu9[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu9[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu9[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu9[i-1,j-1] + P.simu9[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - a_par*H*P + D_H*(H.simu9[i-1,j-1] + H.simu9[i-1,j+1] - 2*H)
        }
        # calculate total population size and store in holding matrix
        P.simu9[i, j] <- P + dP
        H.simu9[i, j] <- H + dH
    }
}
```

d. Make two plots that compare the host (plot 1) and partner's (plot 2) invasion fronts on Day 1000 for (a) D_H = D_P, (b) D_H > D_P, and (c) D_H < D_P. Be sure to include a legend!

```
plot(x = Xset, y = H.simu7[1000,], type = 'l', lwd = 2, col= 'darkgreen' , xlab='Location',ylab='Host P
lines(x = Xset, y = H.simu8[1000,],lwd=2,col='goldenrod')
lines(x = Xset, y = H.simu9[1000,],lwd=2,col='navy')
legend(x = 10, y = Hmax.par*.9, legend=c('D_H=D_P', 'D_H>D_P','D_H<D_P'), lwd=2, col=c('darkgreen','gol
```




```
plot(x = Xset, y = P.simu7[1000,], type = 'l', lwd = 2, col='lightgreen', xlab='Location',ylab='Partner
lines(x = Xset, y = P.simu8[1000,],lwd=2,col='gold')
lines(x = Xset, y = P.simu9[1000,],lwd=2,col='lightblue2')
legend(x = 10, y = Pmax.par*.9, legend=c('D_H = D_P','D_H > D_P', 'D_H < D_P'),lwd=2,col=c('lightgreen'
```



= /2 points total

Question 4: Model applications

Imagine you are a State Parks natural resource manager tasked with slowing (or, ideally, stopping!) the invasion of species H into a native California habitat. You are working with scientists who have genetically engineered a range of potential biocontrol agents. They offer you three options:
Agent A: a = 0.05, D_P = 1
Agent B: a = 0.5, D_P = 1
Agent C: a = 0.05, D_P = 10
Which one do you choose, and why? Plot some invasion fronts to support your choice.

Agent B. As shown previously, a higher partner/parasite dispersal rate has no effect on host dispersal because the P population dies out without a host population to support it. It is only by increasing the attack/exploitation rate of the partner/parasite on the host that the host dispersal is altered.

/5 at least one invasion front plot with all three scenarios
/2 answer and rationale for control agent selection
= /7 points total

```
# agent A
aA = 0.05
D_P = 1
D_H <- 0.001

H.simu.parA <- NaN*tset; H.simu.parA[1] <- 1
P.simu.parA <- NaN*tset; P.simu.parA[1] <- 1
```

```
# for each element i from the second to the last in tset
for(i in 2:length(tset)){
  # calculate change in time
    dt <- tset[i] - tset[i-1]

    # store dummy variables
    H <- H.simu.parA[i-1]
    P <- P.simu.parA[i-1]

    # calculate change in population size
    dH <- ( r*H*(1-H/(K+gamma*P))-aA*H*P )*dt
    dP <- ( b*H*P - m*P*(1+e*P) )*dt

    # calculate total population size
    H.simu.parA[i] <- H + dH
    P.simu.parA[i] <- P + dP
}

Hmax.parA <- H.simu.parA[length(tset)]
Pmax.parA <- P.simu.parA[length(tset)]


Xset <- seq(from = 1, to = 20)

# create a holding matrix for H and fill initial conditions
H.simu10 <- matrix(data=NA,
                   nrow = length(tset), ncol = length(Xset))
H.simu10[1,] <- c(Hmax.parA,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu10 <- matrix(data=NA,
                   nrow = length(tset), ncol = length(Xset))
P.simu10[1,] <- c(Pmax.parA,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu10[i-1, j]
        H <- H.simu10[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu10[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - aA*H*P + D_H*(H.simu10[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu10[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - aA*H*P + D_H*(H.simu10[i-1,j-1] - H) )*dt
```

```
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu10[i-1,j-1] + P.simu10[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - aA*H*P + D_H*(H.simu10[i-1,j-1] + H.simu10[i-1,j+1] - 2*H) )
        }
        # calculate total population size and store in holding matrix
        P.simu10[i, j] <- P + dP
        H.simu10[i, j] <- H + dH
    }
}
```

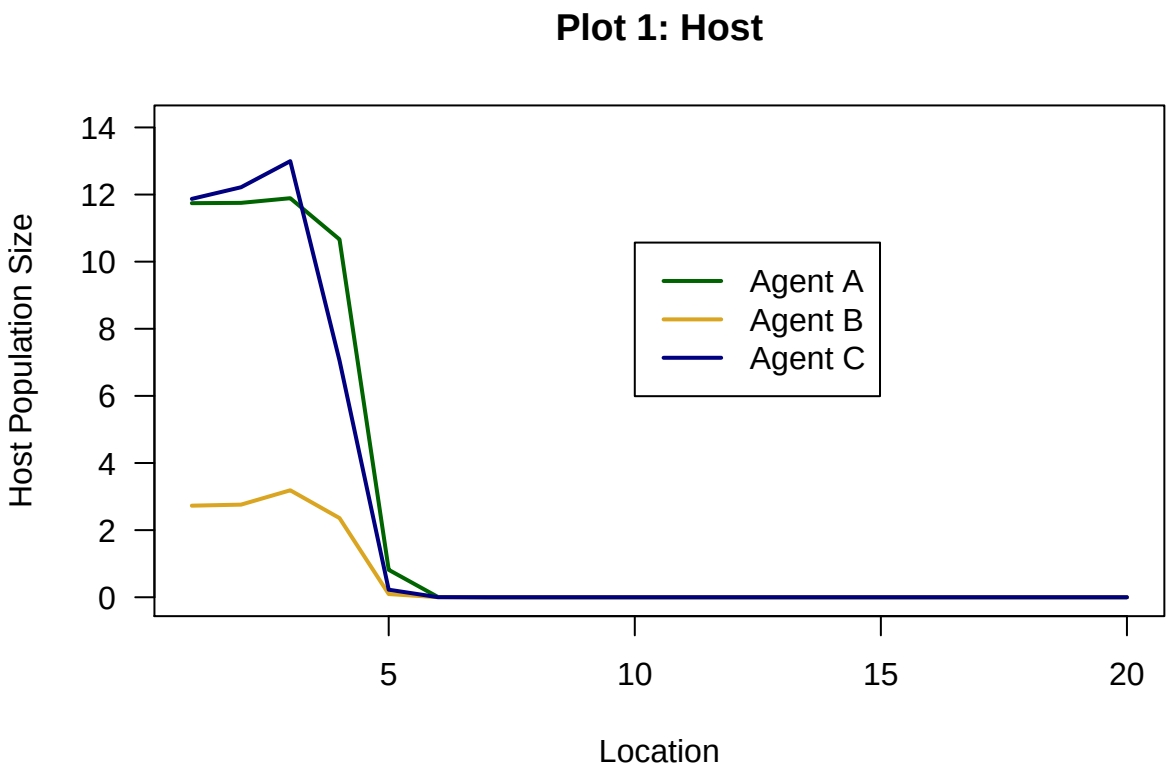
```
# agent B
aB = 0.5
D_P = 1
D_H <- 0.001

H.simu.parB <- NaN*tset; H.simu.parB[1] <- 1
P.simu.parB <- NaN*tset; P.simu.parB[1] <- 1

# for each element i from the second to the last in tset
for(i in 2:length(tset)){
  # calculate change in time
    dt <- tset[i] - tset[i-1]

    # store dummy variables
    H <- H.simu.parB[i-1]
    P <- P.simu.parB[i-1]

    # calculate change in population size
    dH <- ( r*H*(1-H/(K+gamma*P))-aB*H*P )*dt
    dP <- ( b*H*P - m*P*(1+e*P) )*dt

    # calculate total population size
    H.simu.parB[i] <- H + dH
    P.simu.parB[i] <- P + dP
}

Hmax.parB <- H.simu.parB[length(tset)]
Pmax.parB <- P.simu.parB[length(tset)]


Xset <- seq(from = 1, to = 20)

# create a holding matrix for H and fill initial conditions
H.simu11 <- matrix(data=NA,
                   nrow = length(tset), ncol = length(Xset))
H.simu11[1,] <- c(Hmax.parB,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu11 <- matrix(data=NA,
                   nrow = length(tset), ncol = length(Xset))
P.simu11[1,] <- c(Pmax.parB,rep(0,length(Xset)-1))

# for each timestep
```

```
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch

      # store dummy variables
        P <- P.simu11[i-1, j]
        H <- H.simu11[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu11[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - aB*H*P + D_H*(H.simu11[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu11[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - aB*H*P + D_H*(H.simu11[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu11[i-1,j-1] + P.simu11[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - aB*H*P + D_H*(H.simu11[i-1,j-1] + H.simu11[i-1,j+1] - 2*H) )
        }
        # calculate total population size and store in holding matrix
        P.simu11[i, j] <- P + dP
        H.simu11[i, j] <- H + dH
    }
}
```

```
# agent C
aC = 0.05
#aC = aA
D_P = 10
D_H <- 0.001

Xset <- seq(from = 1, to = 20)

# create a holding matrix for H and fill initial conditions
H.simu12 <- matrix(data=NA,
                   nrow = length(tset), ncol = length(Xset))
H.simu12[1,] <- c(Hmax.parA,rep(0,length(Xset)-1))

# create a holding matrix for P and fill initial conditions
P.simu12 <- matrix(data=NA,
                   nrow = length(tset), ncol = length(Xset))
P.simu12[1,] <- c(Pmax.parA,rep(0,length(Xset)-1))

# for each timestep
for (i in 2:length(tset)) {

  # calculate change in time
    dt <- tset[i] - tset[i-1]

    for (j in 1:length(Xset)) { # for each patch
```

```
      # store dummy variables
        P <- P.simu12[i-1, j]
        H <- H.simu12[i-1, j]

        # calculate change in population size
        if (j == 1) { # If you're in the leftmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu12[i-1,j+1] - P) )*dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - aA*H*P + D_H*(H.simu12[i-1,j+1] - H) )*dt
        } else if (j == length(Xset)) { # If you're in the rightmost patch
            dP <- (b*H*P - m*P*(1 + e*P) + D_P*(P.simu12[i-1,j-1] - P) ) * dt
            dH <- (r*H*(1 - H/(K+gamma*P)) - aA*H*P + D_H*(H.simu12[i-1,j-1] - H) )*dt
        } else { # If you're anywhere else
            dP <- (b*H*P - m*P*(1+e*P) + D_P*(P.simu12[i-1,j-1] + P.simu12[i-1,j+1] - 2*P) )*dt
            dH <- (r*H*(1-H/(K+gamma*P)) - aA*H*P + D_H*(H.simu12[i-1,j-1] + H.simu12[i-1,j+1] - 2*H) )
        }
        # calculate total population size and store in holding matrix
        P.simu12[i, j] <- P + dP
        H.simu12[i, j] <- H + dH
    }
}
```

```
#invasion front plot
plot(x = Xset, y = H.simu10[1000,], type = 'l', lwd = 2, col= 'darkgreen' , xlab='Location',ylab='Host
lines(x = Xset, y = H.simu11[1000,],lwd=2,col='goldenrod')
lines(x = Xset, y = H.simu12[1000,],lwd=2,col='navy')
legend(x = 10, y = Hmax.parA*.9, legend=c('Agent A', 'Agent B','Agent C'), lwd=2, col=c('darkgreen','go
```




```
plot(x = Xset, y = P.simu10[1000,], type = 'l', lwd = 2, col='lightgreen', xlab='Location',ylab='Partne
lines(x = Xset, y = P.simu11[1000,],lwd=2,col='gold')
lines(x = Xset, y = P.simu12[1000,],lwd=2,col='lightblue2')
legend(x = 10, y = Pmax.parA*.9, legend=c('Agent A', 'Agent B','Agent C'),lwd=2,col=c('lightgreen','gol
```

= /34 points total