oscillation\_metric

Matthew Barbour

February 24, 2015

Load required R libraries for running the models.

require(deSolve)

## Loading required package: deSolve

require(seqinr)

## Loading required package: seqinr

Setup the Rosenzweig-MacArthur Consumer-Resource model as formulated in Johnson and Amarasekare 2015.

## Rosenzweig-MacArthur consumer-resource model   
# assumes logistic resource growth and a type 2 functional response by the consumer (i.e. consumption rate saturates with resource density). This function is necessary for the ordinary differential equation solver, ode()   
rmcr <- function(t,y,p) {  
 # t,y,p is the necessary format for solving using the ode() function in R  
 R <- y[1]  
 C <- y[2]  
 with(as.list(p), {  
 dR.dt <- r \* R \* (1 - q \* R) - a \* C \* R / (1 + a \* C \* h \* R)  
 dC.dt <- e \* a \* C \* R / (1 + a \* C \* h \* R) - d \* C  
 return(list(c(dR.dt,dC.dt)))  
 })  
}

Set intial state variable (Resource and Consumer densities) for running the model. Note that it is not immediately clear what their initial state variable were so I just played around until I got ones that were close.

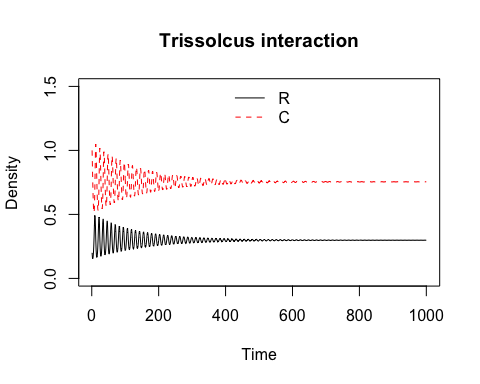
# state variable values (initial values at beginning of "experiments")  
R <- 0.2  
C <- 1  
i.state <- c(R = R,C = C)

Replicate Fig. 3a

## Experiment 1: a \* h / q < 1  
  
# parameter values for Harlequin bug and Trissolocus interaction, taken from Fig. 3 in Johnson & Amarasekare 2015  
r <- 0.85 # per capita rate of increase in resource  
q <- 1/16 # estimate competition coefficient from field data for Harlequin bug  
e <- 1 # predator conversion efficiency  
h <- 0.015 # handling time  
d <- 0.33 # mortality rate of predator  
a <- 1.11 # initial attack rate of 1.3 instead of 1.2 more closely replicates the figures in the book  
  
a \* h / q # 0.2664, which is lower than what was reported in the manuscript (0.35)

## [1] 0.2664

# run the experiment. This experiment essentially solves the model at the initial C and R densities, and takes the new C and R densities and reruns the model, and so on until the end of time.  
p.rm1 <- c(r = r, e = e, a = a, q = q, h = h, d = d) # create a vector for the parameters for experiment #1  
   
Time <- 1000 # set time scale  
rm1 <- ode(i.state,1:Time, rmcr, p.rm1) # run the experiment   
  
# replicate Fig. 3a  
matplot(rm1[,1], rm1[,c(2,3)],  
 type = "l", ylab="Density", xlab="Time", ylim=c(0, C\*1.5), main = "Trissolcus interaction")  
legend("top", c("R","C"), lty=1:2, col=1:2, bty="n")

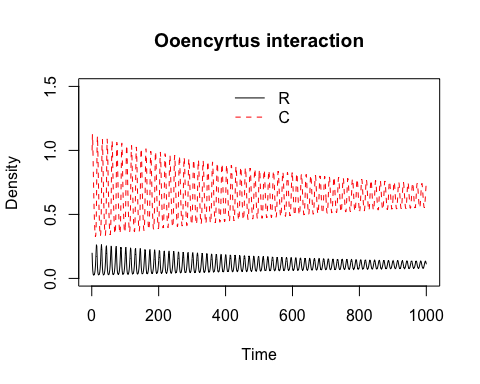


Replicate Fig. 3b.

## parameters for Ooencyrtus interaction  
h <- 0.05   
a <- 1.32   
e <- 1.82   
d <- 0.25   
  
a \* h / q # now is above threshold (i.e. > 1)

## [1] 1.056

# set parameters and run the model for the same amount of time.  
p.rm2 <- c(r = r, e = e, a = a, h = h, q = q)  
rm2 <- ode(i.state, 1:Time, rmcr, p.rm2)  
  
# plot densities  
matplot(rm2[,1],rm2[,c(2,3)],   
 type = "l", ylab="Density", xlab="Time", ylim=c(0,C\*1.5), main = "Ooencyrtus interaction")  
legend("top", c("R","C"), lty=1:2, col=1:2, bty="n")



After replicating these figures, it isn't exactly clear to me why a\*h/q < 1 is a threshold. It seems like both interactions are oscillating...