Guide to coevo_mut_antag code

Lucas Arantes Camacho

27/06/2018

Introduction

Hello everyone.

I write this guide to help the understanding of the model that we gonna work and explain the code to run the simulations. For that, we gonna start speaking a little bit about our model, how it works and after i will explain step-by-step what the R code does. Hope it helps.

General idea

Coevolution is defined as reciprocal evolutionary change of interacting species in a community and is a major process breaking and creating interactions between species in communities[1]. In the last 20 years, the study of coevolution and ecological interactions use network theory tools to help answering questions about the mechanism of coevolution acting in species traits. Nowadays, we already understand how the coevolution creates complementarity of interacting species and convergence of species in the same trophic level in mutualistic networks of interactions[2]. We also know that asymmetry in selective pressures and network structure has an effect in the coevolutionary dynamics of antagonistic networks[3].

Despite the fact that these studies helps the creation of a theoretical map of how coevolution act in traits of species and how interactions change the coevolutionary process, there is compelling evidence that ecological interactions are context dependent in nature[4]. The interactions are context-dependent if the outcomes or the magnitude change depending on the biotic of abiotic context[7].

Using networks only with mutualism or antagonism is a convention to disentangle the effects of these types of interactions outcomes in evolutionary process, but in real communities, species are experiencing mutualistic and antagonistic interactions at the same time. More than that, ecological interactions are not static in nature, but variable in space and time. In this way, we are trying to take a step further in coevolutionary studies, merging the evolutionary effects of antagonisms and mutalisms in the same network of ecological interactions and explore the shifts in interaction outcomes of mutualism and antagonism. Here we gonna answer 2 questions about antagonism in networks of mutualisms and the context dependency of interactions:

1. What happends with the coevolutionary process of mutualisms in the presence of antagonisms?
2. How context-dependent interactions influences the coevolutionary process?

For that, we gonna use computational simulations of a mathematical model that merge the

coevolutionary effects of mutualism, antagonism and other seletive pressures, like abiotic factors.

The model

We are trying to describe how a certain average trait Z of a specie i change in time due to coevolution and other selective pressures (abiotic conditions for example). For that, we have our model.

$$Z_i^{(t+1)} = Z_i^t + \varphi_i[\sum_j m_{ij}^t(Z_j^t - Z_i^t) + \sum_j v_{ij}^t(Z_j^t \pm \epsilon_{ij} - Z_i^t) + p_i(\theta_i - Z_i^t)]$$

This trait Z of specie i in the timestep $t + 1$ depends of Z one timestep further Z_i^t. The traits change also depends on the heritability and additive genetic variance of Z which is given by the φ_i parameter. After that we have 3 components changing the trait Z: the mutualism, the antagonism and the environment. Let's see the mutualism component first.

$$\sum_j m_{ij}^t(Z_j^t - Z_i^t)$$

In this component we have m_{ij}^t that is the mutualistic evolutionary effects of a species j to the specie i. These component favours the trait matching between Z_i and Z_j. In other words, these values Z_i and Z_j tend to become more similar in time. The second component is the antagonist component

$$\sum_j v_{ij}^t(Z_j^t \pm \epsilon_{ij} - Z_i^t)$$

Similar to mutualism, we also have an evolutionary antagonistic effect of j in i which is v_{ij}^t. Note that we have a parameter ϵ, which is our "trait barrier" or the maximum differente of Z_i and Z_j for this antagonistic interaction happen. Finally, we have the environment component

$$p_i(\theta_i - Z_i^t)$$

which θ_i are the vector with the environmental trait optima for each specie.

For the antagonism and mutualism component we have the evolutionary effects of these interactions, that we called v_{ij}^t and m_{ij}^t. These evolutionary effects are defined as a relative effect of specie j in i normalized by all the other interactions that i has with other species. We have

$$m_{ij}^{(t)} = M_{ij}^{(t)} \frac{e^{-\alpha(Z_j^{(t)} - Z_i^{(t)})^2}}{\sum_{k,i \neq k}^N e^{-\alpha(Z_k^{(t)} - Z_i^{(t)})^2}}$$

$$v_{ij}^{(t)} = V_{ij}^{(t)} \frac{e^{-\alpha(Z_j^{(t)} - Z_i^{(t)})^2}}{\sum_{k,i \neq k}^{N} e^{-\alpha(Z_k^{(t)} - Z_i^{(t)})^2}}$$

where $M_{ij}^{(t)}$ and $V_{ij}^{(t)}$ are the elements from the adjacency matrix that give us the species interactions and α is the sensibility of selection to trait matching.

The code

Now, let's construct a code in R that simulates the coevolution process following these model. First, we start by set our parameters of interest and construct our matrices of mutualistic (M) and antagonistic (V) interactions.

```r
# fixate the random parameters (necessary only for this guide)
set.seed(42)
# number of species
n_sp = 5
# probability of an interaction become antagonistic
antprob = 0.25
# herdability and additive genetic variance parameter
phi = 0.2
# sensibility of evolution to trait matching parameter
alpha = 0.2
# optimas of Z`s to environment
theta = runif(n_sp, 0, 5)
# initial Z values
init = runif(n_sp, 0, 5)
# importance of environment to changes in Z
p = 0.01
 # value of max difference between Z`s (barrier)
epsilon = 3
# difference between Z to stop simulation
eq_dif = 0.0001
# max timesteps of simulation
t_max = 1000

# Mutualistic adjacency matrix
M = matrix(1, ncol = n_sp, nrow = n_sp)
# no intraespecific interactions
diag(M) = 0
# Antagonistic adjacency matrix filled with 0 (no interactions)
V = M * 0
```

Great, we have our parameters defined, a matrix of mutualism completely connected and an empty antagonism matrix. Now, we have to transform some links of our M network in

3

antagonists links. For each interaction in our mutualistic matrix, we are going to "roll a dice" and, following the probability of a certain link became antagonist which is our *antprob* parameter, transform or not the interaction in antagonism.

```r
for(i in 1:dim(M)[1]){
  for(j in 1:dim(M)[2]){
    # for each element of M equal 1
    if(M[i,j] == 1){
      # sample a random value
      p = runif(1, 0, 1)

      # if this value is lower than antprob
      if(p <= antprob){
        # transform positive effect in negative
        M[j,i] = 0
        V[j,i] = 1
      }
    }
  }
}
```

Let's look to our matrices and see which interactions became antagonistic

M

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    1    1    1    1
## [2,]    1    0    1    1    1
## [3,]    1    1    0    0    1
## [4,]    1    1    1    0    1
## [5,]    1    0    0    1    0
```

V

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0    0
## [2,]    0    0    0    0    0
## [3,]    0    0    0    1    0
## [4,]    0    0    0    0    0
## [5,]    0    1    1    0    0
```

Great, but there a couple of things we have to see at this point. First, when we transform an interaction in antagonism we are defining an explorer and a victim. In these scenario, we introduce an antagonistic effect of j in i, but not necessarily an antagonistic effect of i in j. The V matrix tell us the antagonistic interactions of species in the collums to the species in the rows. Second (and more intuitive), the V and M matrix are complementar. The sum of V and M gives us all the interactions in the network.

Ok, we have our interactions, now we can start thinking in the simulations. For start, let's create a empty matrix called *z_mat* which is our results matrix. In the matrix z_mat we have the species in the collums and the timesteps of simulation in the rows. The first line of z_mat is the initial trait values.

```
# matrix to store z values
z_mat = matrix(NA, nrow = t_max, ncol = n_sp)
# initial trait values
z_mat[1, ] = init
```

Ok, know we are entering in a loop, which means that these process will happend several times. Let's start by just saying to R which are our initial values of Z. After this, we get all the interactions that we have in our network just getting the sum of M and V.

```
# current z values
z = z_mat[1, ]
# matrix with all interactions (mutualistic and antagonistic)
A = M + V
```

We start our simulation by calculating the evolutionary effects between the species in the network. That's our famous **Q matrix**. To get the Q matrix, we gonna calculate the trait difference between all species in the network. See how our differences of traits matrix (z_dif) looks like.

```
# matrix with all trait differences
z_dif = t(A * z) - A * z
z_dif
```

```
##            [,1]      [,2]       [,3]        [,4]       [,5]
## [1,]  0.0000000 1.0874618 -1.922147  0.6894817  0.9298442
## [2,] -1.0874618 0.0000000 -3.009609 -0.3979801 -0.1576177
## [3,]  1.9221468 3.0096086  0.000000  2.6116285  2.8519909
## [4,] -0.6894817 0.3979801 -2.611628  0.0000000  0.2403625
## [5,] -0.9298442 0.1576177 -2.851991 -0.2403625  0.0000000
```

Ok, now we calculate the $e^{-\alpha(Z_j^{(t)}-Z_i^{(t)})^2}$ component using the trait differences that we already have. We are not interested in intraespecific interactions, so we gonna insert "0" in the diagonal of the matrix.

```
# matrix Q
Q = A * (exp(-alpha * (z_dif ^ 2)))
Q
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 0.0000000 0.7893743 0.4776249 0.9093029 0.8412032
## [2,] 0.7893743 0.0000000 0.1634009 0.9688188 0.9950437
## [3,] 0.4776249 0.1634009 0.0000000 0.2556053 0.1965634
## [4,] 0.9093029 0.9688188 0.2556053 0.0000000 0.9885117
```

```
## [5,] 0.8412032 0.9950437 0.1965634 0.9885117 0.0000000
```

```r
    # intraespecific effects are not allowed
    diag(Q) = 0
    Q
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 0.0000000 0.7893743 0.4776249 0.9093029 0.8412032
## [2,] 0.7893743 0.0000000 0.1634009 0.9688188 0.9950437
## [3,] 0.4776249 0.1634009 0.0000000 0.2556053 0.1965634
## [4,] 0.9093029 0.9688188 0.2556053 0.0000000 0.9885117
## [5,] 0.8412032 0.9950437 0.1965634 0.9885117 0.0000000
```

Now we gonna normalize our Q matrix and get $\dfrac{e^{-\alpha(Z_j^{(t)} - Z_i^{(t)})^2}}{\sum_{k,i\neq k}^{N} e^{-\alpha(Z_k^{(t)} - Z_i^{(t)})^2}}$

```r
    # normalizing the Q matrix
    Q_n = Q / apply(Q, 1, sum)
    Q_n
```

```
##           [,1]      [,2]       [,3]      [,4]      [,5]
## [1,] 0.0000000 0.2615983 0.15828468 0.3013426 0.2787744
## [2,] 0.2706453 0.0000000 0.05602371 0.3321698 0.3411612
## [3,] 0.4369075 0.1494710 0.00000000 0.2338151 0.1798064
## [4,] 0.2912343 0.3102962 0.08186605 0.0000000 0.3166035
## [5,] 0.2784222 0.3293405 0.06505873 0.3271785 0.0000000
```

Great, we already have our normalized matrix of evolutionary effects between the interacting species in the network. But, we have a parameter that says how much the environment is important to trait changes (p). If we are considering p as the importance of environment, *(1 - p)* should be the importance of interactions to trait changes. We gonna multiply these importance of interactions to our Q matrix.

```r
    # multiplying each row i of matrix Q by (1 - p)
    Q_m = Q_n * (1 - p)
    Q_m
```

```
##            [,1]       [,2]        [,3]       [,4]       [,5]
## [1,] 0.00000000 0.02465875 0.014920210 0.02840512 0.02627779
## [2,] 0.02551153 0.00000000 0.005280900 0.03131094 0.03215849
## [3,] 0.04118372 0.01408942 0.000000000 0.02203985 0.01694889
## [4,] 0.02745229 0.02924910 0.007716847 0.00000000 0.02984364
## [5,] 0.02624460 0.03104425 0.006132557 0.03084046 0.00000000
```

Ok, we are ready to calculate our selection differentials. Basically, calculate how much the Z trait will change. We start doing it for the environment. For that, we compute $p_i(\theta_i - Z_i^t)$

```r
    # response to selection related to the environment
    r_env = phi * p * (theta - z)
```

6

Great, now the mutualism part. First we get only the mutualistic interactions and multiply the evolutionary effects by the trait differences between interacting species. We are just computing $\sum_j m_{ij}^t (Z_j^t - Z_i^t)$.

```
# calculating selection differentials to mutualism
sel_dif_mut = M * Q_m * z_dif
# response to selection related to mutualism
r_mut = phi * apply(sel_dif_mut, 1, sum)
```

Ok, finally we compute the antagonism part of trait changes. Before that, we have some conditions in antagonistic interactions. First, the trait difference between i and j has to be minor than our parameter ϵ. The second condition tell us that if the trait of i is minor than j, we have the sum of ϵ in the trait differences $(Z_j^t + \epsilon_{ij} - Z_i^t)$. Otherwise, we have the subtract of ϵ in trait differences $Z_j^t - \epsilon_{ij} - Z_i^t$. In other words, in one situation we are increasing the trait values for the victins and another situation we are decreasing these trait values.

```
# create V_m to use in antagonism selection differential
V_m = V
# excluding interactions of traits that are larger than the barrier
V_m[abs(z_dif) > epsilon] = 0
# matrix with barrier (epsilon) values
epsilon_plus = (z_dif < 0) * matrix(epsilon, n_sp, n_sp)
# matrix wih -epsilon values
epsilon_minus = (z_dif > 0) * matrix(-epsilon, n_sp, n_sp)
# adding barrier values to trait differences
z_dif_a = z_dif + epsilon_plus + epsilon_minus
```

See in the code above how we create a matrix for each situation of increasing and decreasing the trait value of the victim, them we just apply these computations to our trait differences. Ok, now we are ready to calculate the selection differentials due to antagonism using our V matrix, the Q matrix and the trait differences between species.

```
# calculating selection differentials
sel_dif_ant = V_m * Q_m * z_dif_a
# response to selection related to antagonisms
r_ant = phi * apply(sel_dif_ant, 1, sum)
```

We have our selection differentials. Now we update our initial z values. The next line of our result matrix is equal to the sum of initial z values and our already calculated selection differentials.

```
# updating z values
z_mat[2, ] = z + r_env + r_mut + r_ant
```

Remember we are on a loop? These process will be repeated several times. There's a certain moment in the simulation where the traits doesn't change too much. We can define a certain moment, where the changing in Z is very low that we can just stop our simulations, the traits is in equilibrium. We define in this way

```
# computing the mean difference between old and new z values
dif = mean(abs(z - z_mat[2, ]))
if (dif < eq_dif) # if the difference in very low
  break # stop the simulation
```

Of course, maybe these equilibrium state doesn't happend, so the simulation will go until we reach the maximum time of simulations (t_max = 1.000)

Ok, now we just gonna say to R show to us our resulting matrix z_mat, now with a second line of trait values.

```
# return z_matrix
return(z_mat[1:2, ])
```

```
##           [,1]     [,2]      [,3]     [,4]     [,5]
## [1,] 2.595480 3.682942 0.6733330 3.284961 3.525324
## [2,] 2.962321 3.852297 0.8427964 3.438013 3.444144
```

And that's end our simulation process. We just simulate one timestep, these process will happend until reach our t_max already defined in the parameters or will stop when the traits become very similar. Below we have our complete code inside a *for* loop and the final graph showing the timesteps in the x axis and the trait values in y axis.

```
# simulation runs for a maximum of t_max timesteps
for (r in 1:(t_max - 1)) {
  # current z values
  z = z_mat[r, ]
  # matrix with all interactions (mutualistic and antagonistic)
  A = M + V
  # matrix with all trait differences
  z_dif = t(A * z) - A * z
  # matrix Q
  Q = A * (exp(-alpha * (z_dif ^ 2)))
  # intraespecific effects are not allowed
  diag(Q) = 0
  # normalizing the matrix
  Q_n = Q / apply(Q, 1, sum)
  # multiplying each row i of matrix Q by (1 - p[i])
  Q_m = Q_n * (1 - p)

  # response to selection related to the environment
  r_env = phi * p * (theta - z)

  # calculating selection differentials to mutualism
  sel_dif_mut = M * Q_m * z_dif
  # response to selection related to mutualism
  r_mut = phi * apply(sel_dif_mut, 1, sum)
```

```r
    V_m = V
    # excluding interactions of traits that are larger than the barrier
    V_m[abs(z_dif) > epsilon] = 0

    # matrix with barrier (epsilon) values
    epsilon_plus = (z_dif < 0) * matrix(epsilon, n_sp, n_sp)
    # matrix wih -epsilon values
    epsilon_minus = (z_dif > 0) * matrix(-epsilon, n_sp, n_sp)
    # adding barrier values to trait differences
    z_dif_a = z_dif + epsilon_plus + epsilon_minus
    # calculating selection differentials
    sel_dif_ant = V_m * Q_m * z_dif_a
    # response to selection related to antagonisms
    r_ant = phi * apply(sel_dif_ant, 1, sum)

    # updating z values
    z_mat[r+1, ] = z + r_env + r_mut + r_ant

    # computing the mean difference between old and new z values
    dif = mean(abs(z - z_mat[r+1, ]))
    if (dif < eq_dif)
      break
  }

  return(z_mat[1:(r+1), ])
```
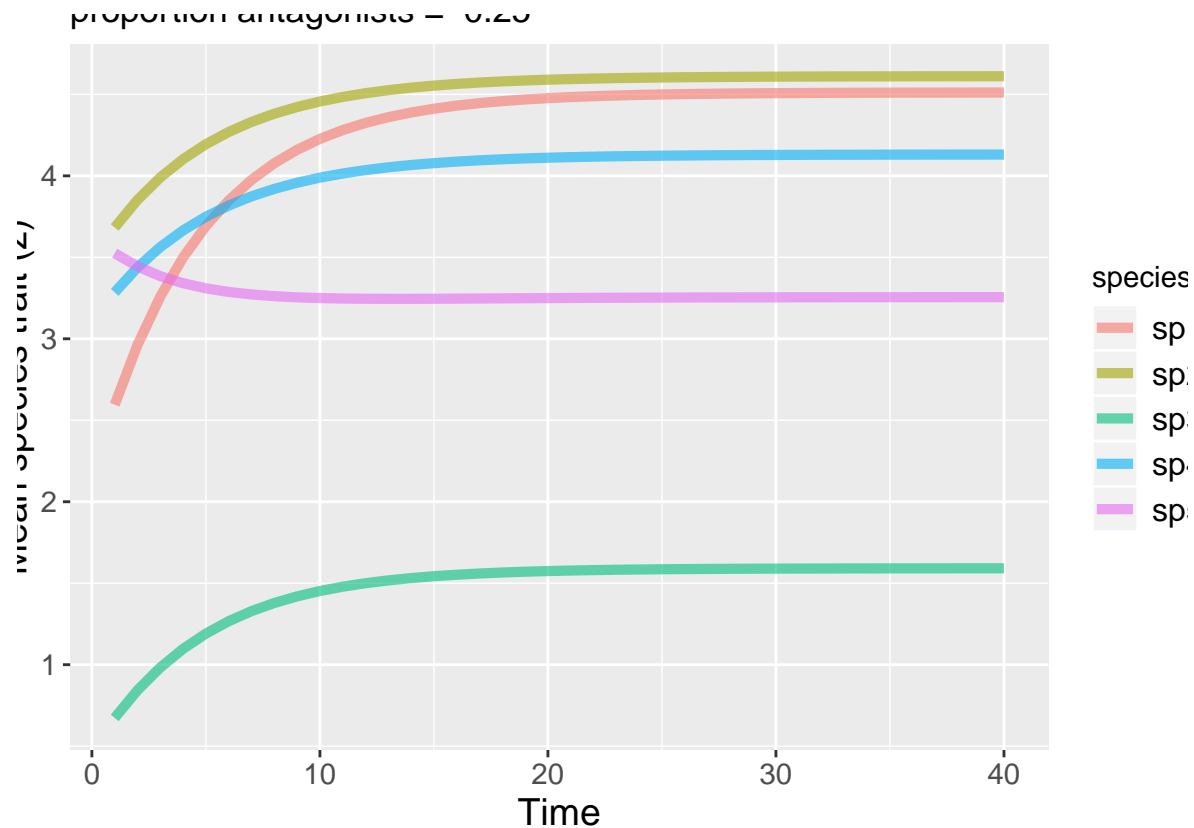
Final statements

That's all for now. Please, let me know if you find some error in my code or model by sending an e-mail to lucas.camacho@usp.br. You can find the R code in my GITHUB repository, in the CoevoMutAntag.R file.

Also, you can find in this repository all the others functions and scripts that i made.

Good Bye

References

[1] J. Bascompte, P. Jordano, *Annu. Rev. Ecol. Evol. Syst.* **2007**, *38*, 567.

[2] P. R. Guimarães Jr, P. Jordano, J. N. Thompson, *Ecology Letters* **2011**, *14*, 877.

[3] C. S. Andreazzi, J. N. Thompson, P. R. Guimarães, *The American Naturalist* **2017**, *190*, 99.

[4] M. Ushio, C.-h. Hsieh, R. Masuda, E. R. Deyle, H. Ye, C.-W. Chang, G. Sugihara, M. Kondoh, *Nature* **2018**, *554*, 360.

[5] A. Kolb, J. Ehrlén, O. Eriksson, *Perspectives in Plant Ecology, Evolution and Systematics*

2007, *9*, 79.

[6] J. Bronstein, *Trends in ecology & evolution* **1994**, *9*, 214.

[7] S. A. Chamberlain, J. L. Bronstein, J. A. Rudgers, *Ecology Letters* **2014**, *17*, 881.