# Código competition()

```
require(plotly) #carregue pacote necessário para elaboração do gráfico

competition = function(data,coef,NArm=TRUE, graphic=TRUE, table=TRUE){
#função
if (NArm==TRUE){ #se o usuário quer remover os NAs
  data.na = na.omit(data)  ##remove todos os NAs com na.omit
  na = as.numeric(dim(data)[1] - dim(data.na)[1]) #contabiliza os NAs
  cat(na,"registros NA's foram removidos","\n") #mensagem para o usuário
  data = data.na #padronizando o nome do objeto
    } else { #condição alternativa: se o usuário não quiser que os NAs sejam
removidos
    data = data #renomeia dataframe
  }
###Teste de premissas: conferindo se as entradas estão corretas
#1.conferir dataframe "data"
if (class(data) != "data.frame"){ #se a classe do objeto é diferente de
dataframe:
  stop("O objeto data não é um dataframe")} #pare e exiba uma mensagem

if (ncol(data) != 5){ #se o objeto não tem 5 colunas
  stop("O objeto data deve ter 5 colunas")} #pare e exiba uma mensagem
#2.conferir matriz "coef"
if (class(coef)!= "matrix"){ #se a classe do objeto é diferente de matriz:
stop("O objeto coef não é uma matriz")} #pare e exiba uma mensagem

if (dim(coef)[1]!=dim(coef)[2]){ #Se o nº de linhas ou colunas é diferente
do número de colunas
  stop("O objeto coef não apresenta a dimensão correta")}  #pare e exiba uma
mensagem

sum = sum(rownames(coef)[] != colnames(coef)[]) ##somar ocorrências de nome
de spp. diferentes entre linhas e colunas
if (sum > 0){
    stop("O nome das espécies não é igual para linhas e colunas da matriz")}
#pare e exiba uma mensagem

#Manipulação dos dados
#padronizando nome de linhas e colunas do objeto data
colnames(data) = c("individuals", "species","u","x","y")

#Criar a estrutura da matriz de distancia para todos os pares de indivíduos
da comunidade
nrow.data= dim(data)[1] #objeto indicando o tamanho que a matriz terá (total
de ind. na comunidade)
dist=matrix(NA,nrow = nrow.data,ncol = nrow.data) #estrutura da matriz
diag(dist)=0 #colocando zeros na diagonal principal
rownames(dist) = seq(1,nrow.data) #nomeia as linhas com o rótulo dos
```

```
indivíduos
colnames(dist) = seq(1,nrow.data) #nomeia as colunas com o rótulo dos
indivíduos

#Calcular distância
#Calculada com base no Teorema de Pitagóras (distância = raiz da soma dos
quadrado dos catetos)
i=1
for (i in 1:(nrow.data-1)){ #cria um contador para para cada indivíduo i
(colunas da matriz)
    j = i+1 #para cada indivíduo pegue o indivíduo j da posição posterior
    for (j in j:nrow.data){ #para cada par de indivíduos ij calcule:
      dx2 = (data$x[j]-data$x[i])^2 #quadrado do deltax entre ij
      dy2 = (data$y[j]-data$y[i])^2 #quadrado do delta y entre ij
      dist[i,j] = sqrt(dx2 + dy2) #raiz da soma dos quadrados é colocada na
posição ij da matriz
      dist[j,i] = sqrt(dx2 + dy2) #raiz da soma dos quadrados é colocada na
posição ji da matriz
    }
    }
#Atribuir uma identificação única (rótulo) para cada espécie da comunidade
#Essa identificação única será necessária para o próximo passo, quando eu
for manipular a matriz com a escala espacial de efeito das espécies
spp.name=sort(unique(data$species)) #vetor com nome das spp ordenado
spp.number= seq(from=1,to=length(spp.name),1) #cria vetor com a sequência de
números que servirão como código das espécie (caso o usuário esteja usando
variáveis do tipo string para os nomes. Número de códigos = número de spp.)
abund=as.numeric(table(data$species)) #extrai vetor de abundancias das spp
(em ordem alfabetica, assim como os códigos)
id=data.frame(spp.name,spp.number,abund) ##criando um objeto unindo as
informações (nome, código e abundância por spp.)

#criar nova coluna no conjunto de dados atribuindo os rótulos espécie-
específicos a cada indivíduo
data$id.spp=rep(NA,nrow.data) #estrutura da coluna
i=1 #inicia o contador de espécies em 1
for (i in 1:length(id$spp.name)){ #estabelece a condição para o loop:
enquanto i for menor ou igual ao número de spp., faça:
  x = which(data$species==id$spp.name[i]) #extrai a posição de cada spp.i em
função do seu nome
  data$id.spp[x] = id$spp.number[i] #coloca o código da espécie i em todas
as posições onde ela ocorre
    i=i+1 #incrementa o contador em 1 para continuar o loop
  }
#Para facilitar a manipulação e os cálculos intermediários até chegar ao
Wij, preciso que as matrizes de distancia (dist) e de escala de efeito
(coef) tenham a mesma dimensão
#a matriz de escala de efeito está entre pares de especies
#a matriz de distâncias está entre pares de individuos
#Assim vou manipular os objetos para obter uma matriz coef que mostre os
```

```
valores da escala espacial de efeito por indivíduo (de acordo com a spp a
qual pertencem). Ou seja, indivíduos da mesma espécie receberão o mesmo
valor.

#Substituir nome das spp. pelos códigos numéricos nas linhas e colunas da
matriz por espécie
i=1 #criando contador de spp.
for (i in 1:length(id$spp.name)){ #Para cada spp i do número total de spp:
spp = id$spp.name[i] #extrai o nome da espécie i
number = id[spp.name==spp,2] #extrai o código que corresponde a spp. i
x = which(colnames(coef)==spp) #extrai a posição da spp. i no nome das
colunas da matriz
colnames(coef)[x]=number ##substitui o nome pelo código
i=i+1 #incrementa o contador em 1 para continuar o loop
}

rownames(coef)=colnames(coef) #igualando o nome das linhas e colunas da
matriz

coef.ind = matrix(NA,nrow = nrow.data,ncol = nrow.data)  #criar a estrutura
da matriz por indivíduo
rownames(coef.ind) = data$id.spp #renomeando os indivíduos nas linhas de
acordo com o código das especies na ordem em que aparecem nos dados
colnames(coef.ind) = data$id.spp #renomeando os indivíduos nas colunas


#Cada linha da matriz representa a escala espacial de efeito de uma espécie
(m) sobre cada espécie j (colunas)
#Assim, vou repetir os coef.da matriz de especies para a matriz  de
individuos repetindo os valores das escalas de efeito espécie-específicos
para cada par de individuo, de acordo com a identidade deles
m=1 #iniciando o contador de espécies
for (m in 1:dim(coef)[1]){ #condição que determina a parada do loop para as
espécies que exercem o efeito (linhas)
    j=m #contador para espécies j que recebem o efeito da spp. m
    for (j in 1:dim(coef)[2]){ #condição que determina a parada do loop para
as espécies que recebem o efeito
      mj = coef[rownames(coef)==m,colnames(coef)==j] #extrai o coefiente de
interaçao das spp. mj
      coef.ind[rownames(coef.ind)==m,colnames(coef.ind)==j] = mj #atribui o
valor do coefiente para todos os pares de individuos mj na matriz por
indivíduo
      j=j+1 #incrementa o contador para voltar para o loop
      }
  }

#Iniciando o cálculo do W (efeito de vizinhança em função da identidade,
tamanho e distância dos vizinhos)
#Etapa1: calcular Wijkm (efeito de cada indivíduo k da spp.m sobre cada
vizinho i da spp. j)
#Wijkm = tamanho.km*e^(-escalajm*dijkm)
```

```
#Antes disso é necessário Voltar os nomes das linhas e das colunas dos três objetos envolvidos no cálculo (u,dist,coef.ind) para a identificação por indivíduos (Sequência numérica)
u = data$u #criar vetor tamanho
names(u) = seq(1,length(u)) #renomear vetor tamanho
rownames(coef.ind) = seq(1,nrow.data) #renomear linhas da matriz
colnames(coef.ind) = seq(1,nrow.data) #renomear colunas da matriz
rownames(dist) #OK, padronizado
colnames(dist) #OK, padronizado

matrix.wijkm=matrix(NA,nrow = nrow.data,ncol = nrow.data) #estrutura da matriz
rownames(matrix.wijkm) = seq(1,nrow.data) #renomear linhas da matriz
colnames(matrix.wijkm) = seq(1,nrow.data) #renomear colunas da matriz

i=1 #inicia o contador para cada individuo i que recebe o efeito
for (i in 1:nrow.data){ #para cada individuo focal i:
  k=1 #crie um k correspondendo a cada vizinho de i
  for (k in 1:nrow.data){ #para cada vizinho k calcule
    wik = u[k]*exp(-(coef.ind[k,i])*dist[k,i]) #valor do efeito de k sobre i
    matrix.wijkm[k,i] = wik #guarda na linha k coluna i da matriz
    k=k+1 #incrementa o contador para voltar para o loop
    }
}

diag(matrix.wijkm)=0 #diagonal = efeito de cada individuo sobre si = 0

#Etapa2:calcular Wijm (efeitos de cada especie m sobre o individuo i da especie j)
#Wijm = soma do efeito de todos os indivíduos k da espécie m sobre indivíduo i da espécie j
wijm = matrix(NA,nrow = dim(coef)[1],ncol=nrow.data) #estrutura da matriz
rownames(wijm) = seq(1,dim(coef)[1]) #nomeando as linhas
colnames(wijm) = seq(1,nrow.data) #nomeando as linhas

#trocando a identificação de indivíduos para espécies na matriz Wijkm (para que eu possa somar as linhas dos indivíduos da mesma spp)
rownames(matrix.wijkm) = data$id.spp

#Somando os valores dos indivíduos da mesma spp.
i=1 #iniciando contador de indivíduos
for (i in 1:nrow.data){ #para cada indivíduo i (colunas):
  m=1 #crie um m para espécie que faz o efeito (linhas)
  for (m in 1:dim(coef)[1]){ #para cada espécie:
  wijm[m,i]=sum(matrix.wijkm[which(rownames(matrix.wijkm)==m),i]) #some os valores do efeito de todos os indivíduos da mesma spp. e guarde na linha m da coluna i
  m+1 #incrementa o contador para voltar para o loop
  }
  }
```

```
#E,finalmente, calcular o Wij (efeito total de vizinhança sobre o indivíduo
i)
#Wij = soma dos efeitos de todas as espécies sobre o indivíduo i
wij = round(apply(wijm,2,sum),4) #soma de cada uma das colunas com 4 casas
decimais
index.wij = data.frame(seq(1:nrow.data),wij)#acrescentando uma coluna com o
código dos indivíduos
colnames(index.wij) = c("Individuals","Wij") #renomeando colunas


#Fazendo o gráfico
if (graphic==TRUE){ #se o usuário quer o gráfico:
  data.graph = cbind(data,index.wij$Wij) #concatene informações necessárias
  colnames(data.graph) =
c("Indivíduo","Spp.","Tamanho","CoordenadaX","CoordenadaY","Id_Spp.","Wij")
#renomeie as colunas para o eixo do gráfico
graphic = plot_ly(data.graph, x=~CoordenadaX, y = ~CoordenadaY,
type="scatter", mode = "markers", color = ~Wij, size = ~Wij,colors="YlOrRd")
#faça o gráfico
print(graphic)
}

#Salvando ou não o arquivo no diretório do usuário, de acordo com a escolha
do usuário
if (table == TRUE){ #se o usuário quer o arquivo salvo:
write.table(index.wij,file="Wij.txt",sep = "\t",row.names = FALSE) #exporte
o arquivo em .txt para o diretório de trabalho
}

return(index.wij)
}
```

##########################################################