Supplementary material of: Combining food-web theory and population dynamics to assess the impact of invasive species

R-code part 1: Inferences of trophic interactions

Chloé Vagnon, Rudolf P. Rohr

June 2021

Function definition

Function 1: get_niche_attributes

Infer the niche attributes of species in the species inventory based on their body size and their category. Niche parameters of primary producers are automatically set at 0. The function requires the package "stringr".

Input: Data are ordered by decreasing body sizes.

1. species_name = the name of the species to use
2. body_size = log10(vector of species body sizes in μm)
3. species_category = "vertebrate", "invertebrate", "zooplankton", or other. If "other" is mentionned, the species will automatically be considered as producer and not as consumer. Capital and lowercase letters are allowed for the species category. Species category can also be provided in French.

Outputs : A data frame with the Niche attributes of each species countaining:

1. name = the species name
2. n = the log10(species body sizes in μm)
3. low = the lower bound of the species diet range (QR at 5%)
4. high= the higher bound of the species diet range (QR at 95%)
5. c = the center of the species diet range

```
get_niche_attributes<-function(name,body_size, species_category) {

  Niche<-data.frame(name=NA,n=NA,c=NA,low=NA,high=NA)

  for(i in 1:length(body_size)){

    Niche2<-data.frame(name=NA,n=NA,c=NA,low=NA,high=NA)

    # For primary producers
    if(!str_detect(species_category[i], regex("vertebrate", ignore_case = TRUE)) |
       !str_detect(species_category[i], regex("invertebrate", ignore_case = TRUE)) |
       !str_detect(species_category[i], regex("vertebre", ignore_case = TRUE)) |
       !str_detect(species_category[i], regex("invertebre", ignore_case = TRUE))|
       !str_detect(species_category[i], regex("zoop", ignore_case = TRUE))){
```

```
      Niche2$name=name[i]
      Niche2$n = body_size[i]
      Niche2$low = 0
      Niche2$high = 0
      Niche2$c = 0
    }

    # For vertebrates
    if(str_detect(species_category[i], regex("vertebrate", ignore_case = TRUE))&
       !str_detect(species_category[i], regex("inv", ignore_case = TRUE))|
       str_detect(species_category[i], regex("vertebre", ignore_case = TRUE))&
       !str_detect(species_category[i], regex("inv", ignore_case = TRUE))){
      qrsup = Param_regvert[[2]]
      qrinf = Param_regvert[[3]]
      Niche2$name=name[i]
      Niche2$n = body_size[i]
      Niche2$low = qrinf[1] + qrinf[2]*body_size[i]
      Niche2$high = qrsup[1] + qrsup[2]*body_size[i]
      Niche2$c = Niche2$low+(Niche2$high-Niche2$low)/2
    }

    # For invertebrates
    if(str_detect(species_category[i], regex("invertebrate", ignore_case = TRUE))|
       str_detect(species_category[i], regex("invertebre", ignore_case = TRUE))|
       str_detect(species_category[i], regex("zoop", ignore_case = TRUE))){
      qrsup = Param_reginvert[[2]]
      qrinf = Param_reginvert[[3]]
      Niche2$name=name[i]
      Niche2$n = body_size[i]
      Niche2$low = qrinf[1] + qrinf[2]*body_size[i]
      Niche2$high = qrsup[1] + qrsup[2]*body_size[i]
      Niche2$c = Niche2$low+(Niche2$high-Niche2$low)/2

    }
    Niche<-rbind(Niche,Niche2)
  }

  return(na.omit(Niche))
}
```

Function 2: L_fn2

Transform the parameters from get_niche_attributes into an binary interaction matrix from Gravel et al. 2013

Inputs:

1. name = the species name
2. n = the log10(species body sizes in μm)
3. low = the lower bound of the species diet range (QR at 5%)
4. high= the higher bound of the species diet range (QR at 95%)
5. c = the center of the species diet range
6. table= "NO" to only obtain the binary matrix, "YES" to only obtain the binary matrix + a data.frame with one observation = one interaction

Outputs :

A binary interaction matrix with 0 indicating absence of a link and 1 indicating the presence of a link with consumers in columns (j) and resources in rows (i) OR A list with the binary interaction matrix + a table referencing each link with :

1. Res = the resource name
2. Cons = the consumer name
3. Log10Size_Res = log10(resource body size)
4. Log10Size_Cons = log10(consumer body size)

```
L_fn2 <- function(name,n,c,low,high,table) {

  S <- length(n)
  L <- matrix(0,nr=S,nc=S)

  for(j in 1:S)
    for(i in 1:S)
      if(n[i]>low[j] && n[i]<high[j]) L[i,j] = 1
  colnames(L) <-name
  rownames(L) <-name

  Table<-data.frame(Res=NA,Cons=NA,Log10Size_Res=NA,Log10Size_Cons=NA)
  for(i in 1:S){
    if(length(which(L[,j]==1))!=0){
      Table2 <- data.frame(Res=names(which(L[,j]==1)),
                           Cons=rep(colnames(L)[j],length(which(L[,j]==1))),
                           Log10Size_Res=n[which(L[,j]==1)],
                           Log10Size_Cons=n[j])}
    else{Table2<-data.frame(Res=NA,Cons=NA,Log10Size_Res=NA,Log10Size_Cons=NA)}
    Table <- rbind(Table,Table2)}

  if(table=="NO"){
    return(L)
  }
  if(table=="YES"){
    return(list(Bmat=L,Table=na.omit(Table)))
  }
}
```

Function3: Ref_L_Diet

Refine links for impossible results according to species diet trait. Fish do not eat primary producers. Carnivorous macroinvertebrates do not eat primary producers Other diet refinement can be implemented after applying the function as they are considered as site-specific.

Inputs:

1. Bmat = binary matrix of trophic links inferred from L_fn2 function with : colnames and rownames = names of species in the inventory
2. diet = “invP” or “inv” for invertebrates | “p”=piscivorous or “o”=omnivorous for fish | “prod” for primary producers
3. table = “YES” or “NO” to obtain or note the matrix with links refined
4. LinksTab = if Table is “YES” provide the table of links obtained with the function L_fn2

Outputs :

1. if table="NO", returns th Binary matrix after link refinement
2. if table="YES", returns th Binary matrix after link refinement + the table of links after links refinement

```
Ref_L_Diet <- function(Bmat, diet,table,LinksTab) {
  if(table=="NO"){

    for(j in 1:ncol(Bmat)){
      for (i in 1:nrow(Bmat)){
        if(diet[j]=="omnivorous" & diet[i]=="prod" |

           diet[j]=="piscivorous" & diet[i]=="prod"){
          Bmat[i,j]<-0
        }else{Bmat[i,j]<-Bmat[i,j]}

        if(diet[j]=="invP" &
           diet[i]=="prod"){
          Bmat[i,j]<-0
        }else{Bmat[i,j]<-Bmat[i,j]}
      }
    }
    return(Bmat_ref=Bmat)
  }

  if (table == "YES") {
    for(j in 1:ncol(Bmat)){
      for (i in 1:nrow(Bmat)){
        if(length(which(Bmat[,j]==1))!=0){

          if(diet[j]=="omnivorous" & diet[i]=="prod" |
             diet[j]=="piscivorous" & diet[i]=="prod"){
            Bmat[i,j]<-0
          }else{Bmat[i,j]<-Bmat[i,j]}

          if(diet[j]=="invP" & diet[i]=="prod"){
            Bmat[i,j]<-0
          }else{Bmat[i,j]<-Bmat[i,j]}

        }else{Bmat[,j]<-0}
      }
    }

    for (j in 1:ncol(Bmat)) {
      for (i in 1:nrow(Bmat)) {
        if(Bmat[i,j]==0){
          LinksTab$Prey[LinksTab$Prey==rownames(Bmat)[i]&
                          LinksTab$Pred==colnames(Bmat)[j]]<-NA
        }else{

        }
      }
    }
    return(list(Bmat_ref = Bmat,Table_ref=na.omit(LinksTab)))
  }
}
```

Function 4: Ref_L_Hab

Refine links for impossible results according to the habitat trait

Inputs:

1. Bmat = binary matrix of trophic links inferred from L_fn2 function with : colnames and rownames = names of species in the inventory
2. habitat = "pel" or "ben" or "pel/ben"
3. Table = "YES" or "NO" to obtain or note the matrix with links refined

Outputs :

1. if Table="NO", returns th Binary matrix after link refinement
2. if Table="YES", returns th Binary matrix after link refinement and the table of links after links refinement

```
Ref_L_Hab = function(Bmat, habitat, table,LinksTab) {
  if(table=="NO"){
    for(i in 1:ncol(Bmat)){
      for (j in 1:nrow(Bmat)){
        if(length(which(Bmat[,i]==1))!=0){
          if(habitat[i]=="pel" &
             habitat[j]=="ben" |
             habitat[i]=="ben" &
             habitat[j]!="pel"){
            Bmat[j,i]<-0
          }else{Bmat[j,i]<-Bmat[j,i]}
        }else{Bmat[,i]<-0}
      }
    }
    return(Bmat_ref=Bmat)
  }

  if (table == "YES") {
    for (i in 1:ncol(Bmat)) {
      for (j in 1:nrow(Bmat)) {
        if (length(which(Bmat[, i] == 1)) != 0) {
          if (habitat[i] == "pel" &
              habitat[j] == "ben" |
              habitat[i] == "ben" &
              habitat[j] == "pel") {
            Bmat[j, i] <- 0
          } else{
            Bmat[j, i] <- Bmat[j, i]
          }
        } else{
          Bmat[, i] <- 0
        }
      }
    }

    for (i in 1:ncol(Bmat)) {
      for (j in 1:nrow(Bmat)) {
        if(Bmat[j,i]==0){
          LinksTab$Prey[LinksTab$Prey==rownames(Bmat)[j]& LinksTab$Pred==colnames(Bmat)[i]]<-NA
        }else{
```

```
        }
      }
    }
    return(list(Bmat_ref = Bmat,Table_ref=na.omit(LinksTab)))
  }
}
```

Function 5: Weighting

Used to obtain the likelyhood of an occuring trophic link between a consumer and a ressource. This "weighting" is scaled to the resource body size range obtained from diet range for each consumer.

Inputs :

1. Niche_attributes = data frame resulting from the function "get_niche_attribute" with :

- names = species name also used as colnames/rownames for the binary matrix
- n = log10(species body size (μm))
- c = optimal center of the niche (log10(μm))
- low = lower bound of the niche range (log10(μm))
- high =higher bound of the niche range (log10(μm))

2. Bmat = initial binary interaction matrix

Outputs :

1. the "weighted"" interaction matrix

```
Weighting <- function(Niche_attributes,Bmat){
    for(j in 1:ncol(Bmat)){
    if(length(which(Bmat[,j]==1))!=0){
      Bmat[which(Bmat[,j]==1),j] <- dnorm(Niche_attributes$n[which(Bmat[,j]==1)],
                  mean=Niche_attributes$c[j],
                  sd=sd(seq(from=Niche_attributes$low[j],to=Niche_attributes$high[j],
                  by=((Niche_attributes$high[j]-Niche_attributes$low[j])/100))))/
                  max(dnorm(Niche_attributes$n[which(Bmat[,j]==1)],
                   mean=Niche_attributes$c[j],
                   sd=sd(seq(from=Niche_attributes$low[j],to=Niche_attributes$high[j],
                           by=((Niche_attributes$high[j]-Niche_attributes$low[j])/100)))))
    }else{Bmat[which(Bmat[,j]!=1),j]<-0}
  }
  return(Bmat)
}
```

Example: Inferrence and refinement of the trophic links among species in an inventory

Loading library and data

```
library(stringr)

#Parameters to calculate the consumers diet range (regressions quantiles)
load("Param_reginvert.Rdata") # For invertebrates
load("Param_regvert.Rdata")   # For vertebrates

#Loading of the species inventory created for the example
load("Example_SpInventory.Rdata")
DATA2<-DATA[order(DATA$bs,decreasing=TRUE),] # species are ordered by decreasing body size
DATA2$Log10bs<-log10(DATA2$bs) #Body size is transformed to log10
```

A) The resource body sizes range are reconstructed for consumers based on :

1. species category (i.e., primary producers, zooplankton,invertebrate, vertebrate)
2. body size

Note that rows corresponding to primary producers are automatically filled with 0.

```
bs_r<-get_niche_attributes(name=DATA2$Species,
                           body_size = DATA2$Log10bs,
                           species_category = DATA2$Category)
```

B) Inferrences of all trophic links (binary matrix)

```
Bmat<-L_fn2(name=bs_r$name,n=bs_r$n,c=bs_r$c,low=bs_r$low,high=bs_r$high,table="NO")
```

C) Links refinement based on consumers' diet

```
Bmat_Diet<-Ref_L_Diet(Bmat=Bmat, diet=DATA2$Diet,table="NO")
```

D) Links refinement based on consumers' habitat

```
Bmat_Hab<-Ref_L_Hab(Bmat=Bmat_Diet,habitat =DATA2$Habitat,table="NO")
# Note that supplementary refinement can be implemented depending on the ecosystems or the
# species studied.In this example fish and predatory invertebrates are not allowed to consume
# primary producers.
# Final binary matrix Mb is given by:
Mb<-Bmat_Hab
```

E) Weighting of refined trophic links

```
Mb_W<-Weighting(Niche_attributes = bs_r,Bmat=Mb)
```

F) Use weightings as resources i proportions in diet of consumer j

$$\omega_{ji} = \frac{WeightedLink_{ji}}{\sum_{i \in resources\ of\ j} WeightedLink_{ji}} \ (eq.1)$$

```
Wji<-Mb_W
Wji[,which(colSums(Wji)!=0)]<-t((1/colSums(Wji[,which(colSums(Wji)!=0)])) *
                                 t(Wji[,which(colSums(Wji)!=0)]))
```

G) Save matrices used in the simulations of population dynamics

```
save(DATA2,file="DATA2.Rdata")# Species inventory with supplementary data
save(Mb,file="Mb.Rdata") # Binary interaction matrix
save(Wji,file="Wji.Rdata") # Matrix of resources i proportions in diet of consumer j
```

Supplementary material of: Combining food-web theory and population dynamics to assess the impact of invasive species

R-code part 2: Simulations of population dynamics

Chloé Vagnon, Rudolf P. Rohr

June 2021

Example: Simulations of population dynamics

Loading library and data

```
library("deSolve") # Package for calculations

load("DATA2.Rdata") # Species inventory created in the 1st part of the code example
load("Mb.Rdata")    # Binary interaction matrix
load("Wji.Rdata")   # Species proportions in consumer diet

# Load the parameters to convert body size to metabolic rate
load("bodymass.Rdata") # For conversion from body size to bodymass
load("metabolic.Rdata") # For conversion from body mass to metabolic rate
```

A) Parametrization of the dynamic model with constants from literature and allometric relationships

```
S <- length(DATA2$Species)       # Define the number of species (S)
Iprod <- which(colSums(Mb) == 0) # Define the primary producers (Iprod)
Cons <- which(colSums(Mb) > 0)   # Define the consumers (Cons)

# Log10(body size) is converted in Log10(body mass)
DATA2$Log10bm <- m_bodymass$coefficients[1] + m_bodymass$coefficients[2] * DATA2$Log10bs
# Log10(body mass) is converted in metabolic rate
DATA2$Log10mr <- m_metabolic$coefficients[1] +
  m_metabolic$coefficients[2]*DATA2$Log10bm + m_metabolic$coefficients[3]*DATA2$Log10bm^2

#Biotic capacity of primary producers
DATA2$K[Iprod] <- 10^(-0.77*DATA2$Log10bm[Iprod]-6)

# Maximum consumption rate
DATA2$y[DATA2$Category == 'Vertebrate'] <- 4
DATA2$y[DATA2$Category == 'Invertebrate' & DATA2$Diet=="invP"] <- 8
DATA2$y[DATA2$Category == 'Invertebrate' & DATA2$Diet=="inv"] <- 5
DATA2$y[DATA2$Category == 'Zooplankton'] <- 6.5
DATA2$y[DATA2$Diet == 'prod'] <- 1.69
```

```
# Allometric constant(Brose et al., 2006)
DATA2$ax[DATA2$Category %in% 'Vertebrate'] <- 0.88
DATA2$ax[DATA2$Category != 'Vertebrate'] <- 0.314

# Efficiency of predator consumption
DATA2$epsilon[DATA2$Carnivorous == 1] <- 0.85
DATA2$epsilon[DATA2$Carnivorous ==0] <- 0.45

# Handling time
h<-t(Mb)
for (j in 1:nrow(h)){
  #for invertebrates
 if(is.element(rownames(h)[j],
               DATA2$Species[DATA2$Category%in%c("Invertebrate","Zooplankton")])){
   h[j,which(h[j,]!=0)]<-1/DATA2$y[j]
   }
 else{
 h[j,which(h[j,]!=0)]<-(4.084*10^5)*(10^DATA2$Log10bm[is.element(DATA2$Species,
                                                                names(which(h[j,]!=0)))])*
   (10^(DATA2$Log10bm[j])^-0.75)
  }

}
```

B) Initialization of the simulations
The dynamic of primary producer i is given by:

$$\frac{dN_i}{dt} = r_i \cdot \left(1 - \frac{N_i}{K_i}\right) \cdot N_i - \sum_{j \in consumers\ of\ i} F_{ji}(\overrightarrow{N}) \cdot N_j \quad (eq.2)$$

Where the functional response (*i.e.*, the per capita consumption rate of consumer j on resource i) is given by:

$$F_{ji}(\overrightarrow{N}) = \frac{x_j \cdot y_j \cdot \omega_{ji} \cdot N_i}{1 + \sum_{k \in resources\ of\ j} h_{jk} \cdot x_j \cdot y_j \cdot \omega_{jk} \cdot N_k} \quad (eq.3)$$

The dynamic of consumer j is given by:

$$\frac{dN_j}{dt} = -m_i \cdot \left(\frac{N_j}{K_i}\right) \cdot N_i + \varepsilon_j \sum_{i \in resources\ of\ j} F_{ji}(\overrightarrow{N}) \cdot N_j - \sum_{j \in consumers\ of\ i} F_{lj}(\overrightarrow{N}) \cdot N_l \quad (eq.4)$$

```
# Model used to calculate abundances at each time step
dN <- function(t,N,p){
  Fij <- p$alpha / as.vector(1 + (p$h *p$alpha)  %*%  N)
  out <- N * (p$r - p$alpha_intra * N  +  p$epsilon * Fij %*% N - t(Fij) %*% N)
  return(list(out))
}
# Note that alpha is the product of the metabolic rate mass dependant (xj), the maximum
# consumption rate (yj) and the matrix of resource proportions indiet of consumers (wij).
# It represents an interaction force, allometrically parameterized in our study but that
# can be parameterized with other technics.
```

C) Calulation of the abundances along time

```
N_rand <- 10                # Number of simulations
time_step <- seq(0,19999,1) # Time steps
Alive_sp <- rep(NA,N_rand)  # Vector allocation for alive species
Output<- array(NA,c(N_rand,length(time_step),S)) # Large array for stocking outputs


for (i in 1:N_rand){
  assign("last.warning", NULL, envir = baseenv())

  #Metabolic rate and noise simulated with the rnorm
  xj<- 10^(DATA2$Log10mr-DATA2$Log10bm)+ rnorm(S, mean=0, sd=0.001*(10^DATA2$Log10mr))


  # Growth
  r <- xj
  r[Cons] <- -r[Cons] * DATA2$ax[Cons]


  # alpha
  alpha <- (xj * DATA2$y) * t(Wji)
  alpha_intra <- rep(0,nrow(alpha))
  alpha_intra[Iprod] <- r[Iprod]/DATA2$K[Iprod] #intraspecific regulation for Iprod

  #List of parameters to induce in the equa diff computing
  p <- list(alpha = alpha, alpha_intra = alpha_intra, r = r, epsilon = DATA2$epsilon, h=h)

  # Initialization of initial  abundances
  N0 <- sort(round(runif(S, min=0.15, max=1), digits=5), decreasing = F)

  # Calculatin of results from differential equations
  out <- ode(y = N0, times = time_step, func = dN, parms = p)

  if (length(warnings())==0){    #Avoid warning messages
    N_equ <- out[dim(out)[1],-1] # N at the equilibrium
    alive <- N_equ > 1e-6 # Computing of species still alive

    Alive_sp[i] <- sum(alive)
    Output[i,,] <- as.matrix(out[,-1])
  }
}
```