Calcul de l'indice de biodiversité : Comparaison de deux types de programmation

Annie Hofstetter

Calcul de l'indice de biodiversité : Comparaison de deux types de programmation

Annie Hofstetter[1]

Je suis Annie Hofstetter, ingénieure d'études à l'unité Centre d'Économie de l'Environnement de Montpellier en appui aux recherches en sciences sociales conduites dans cette unité pour aider les chercheurs dans le traitement des données et leur organisation en système d'information, pour contribuer au développement d'applications et accompagner dans le déploiement de solutions de calculs scientifiques. Depuis longtemps sensibilisée par l'organisation de l'informatique du département EcoSocio, je suis co-animatrice du Cati CITISES (Centre Informatisé du Traitement de l'Information en Sciences Économiques et Sociales).

Résumé. Ce document présente un outil de calcul d'un indice de biodiversité. L'algorithme a été défini et a permis de mettre en œuvre une application qui répondait au cahier des charges de 2007, basée sur les technologies de développement autour du web : HTML, PHP, MySQL et Javascript. En 2020, l'algorithme a été adapté afin de permettre un développement sur un seul logiciel, R, qui améliore considérablement les temps de calcul de l'indice de biodiversité. C'est ainsi qu'est née l'idée de faire une comparaison des deux versions.

Mots clés : indice de biodiversité, calcul, algorithme, développement d'application, comparaison

Abstract. This document presents a tool for calculating a biodiversity index. The algorithm has been defined and allowed the implementation of an application that met the specifications defined in 2007, based on web development technologies: HTML, PHP, MySQL and Javascript. In 2020, the algorithm was adapted to allow development on a single software, R, which considerably improves the calculation time of the biodiversity index. Thus was born the idea of making a comparison between the two versions.

Keywords : biodiversity index, calcul, algorithm, computer programming, comparison

1 UMR 1135 CEE-M, Campus SupAgro, 2 Place Viala, F-34060 Montpellier, France

Email : Annie.Hofstetter@inrae.fr

Il était une fois

J'étais loin de me douter qu'un jour mes cours de biologie me serviraient pour le développement d'applications informatiques. En l'occurrence, le mot mnémotechnique *RECOFGER* d'un professeur de biologie m'a permis de me rappeler la série *règne, embranchement, classe, ordre, famille, genre, espèce* et *race* ou *variété,* ce qui m'a été très utile dans mon travail d'appui à la recherche dont une application est décrite dans cet article. Les économistes parlent de la biodiversité spécifique en termes d'abondance d'une espèce, qu'elle soit relative ou absolue. Ils vont même jusqu'à évoquer le nombre efficace d'espèces dans un écosystème (Aulong et al, 2005 ; 2008). Les écologues commencent à parler d'extinction (resp. d'abondance) de variétés ou de races avant de parler d'extinction (resp. d'abondance) d'espèces ; ce qui est plus logique en termes de classification, puisque l'espèce n'est pas au bout de cette série de notions illustrant la classification. Le calcul de l'indice de biodiversité sur lequel nous avons travaillé et dont les applications font l'objet de comparaisons évoquées dans cet article, peut porter indifféremment sur des variétés ou des espèces. Par simplification du propos, il sera plus fréquemment utilisé le terme espèce quel que soit le niveau de classification pour lequel nous ferons un calcul. D'ailleurs, comme l'écrit Weitzman (1993), le mot « species » vaut donc pour n'importe quelle entité au niveau de la classification.

Historiquement, les généticiens ont porté leur intérêt pour mesurer la diversité génétique à l'aide de l'indice proposé par l'économiste Weitzman. Cet indice (cf. encadré) a plusieurs défauts connus : en premier lieu, il est théorique (sans doute a-t-il été défini sans se soucier des données nécessaires); par ailleurs son temps de calcul est souvent conséquent ; enfin, il varie avec le nombre d'espèces dans l'échantillon. D'où l'idée au départ, de calculer une approximation de cet indice. À l'aube de l'année de la biodiversité (2010), certains chercheurs en sciences sociales ont travaillé sur cet indice de Weitzman et ont proposé, mieux qu'une approximation, une nouvelle façon de calculer l'indice, plus simple et plus rapide. Le point de départ est la « dissimilitude » entre les espèces qui est représentée par la distance génétique entre chaque couple d'espèces. L'algorithme consiste à classer les distances d'une espèce avec toutes les autres espèces en les triant par ordre croissant, puis à retenir la plus petite distance et à retirer de l'échantillon l'espèce correspondante. Ce tri est réeffectué en repartant de la matrice initiale sans l'espèce qui vient d'être retirée, et ce jusqu'à la dernière espèce (en réalité jusqu'à l'avant dernière puisque la toute dernière espèce présentera une distance nulle avec elle-même). La diversité totale est la somme de toutes les distances mises en mémoire. Le résultat du calcul est un nombre ; plus la distance entre les espèces est élevée, plus l'indice de diversité entre les espèces sera élevé.

Encadré : Indice de biodiversité de Weitzman

Soit un ensemble S d'espèces.
Soit le sous-ensemble Q' de S qui contient l'espèce i.
Soit D(Q') sa diversité et δ(i,Q'-i) la dissimilarité entre l'espèce i et l'ensemble
Q'-i.
L'indice de Weitzman s'obtient par la formule récursive suivante :

$$D(Q') = \underset{i \in Q'}{Max}\{D(Q' - i) + \delta(i, Q' - i)\}$$

Première version du calculateur V2007

Les premiers termes du cahier des charges de 2007 s'orientaient vers une application web prenant en charge l'utilisateur et rendant invisible un certain nombre de tâches. Il était également important de pouvoir réaliser le calcul avec un nombre important de données, d'où un couplage éventuel avec un système d'information.

Le calcul proposé travaille sur une matrice carrée, symétrique et dont la diagonale est nulle ; ce qui implique de garder un grand nombre de distances s'il y a un grand nombre d'espèces. Un système de gestion de base de données semblait approprié pour le traitement d'un grand nombre de couples de distances c'est-à-dire autant de tuples dans la table de la base de données sous MySQL.

L'utilisation d'un navigateur Internet a l'avantage d'être indépendant du système d'exploitation et le choix initial s'est porté sur un outil facilement portable (HTML).

L'application propose donc deux possibilités : soit la saisie à l'écran pour un nombre raisonnable d'espèces, soit à l'aide d'un fichier disponible. Tout d'abord, dans le cas où on autorise la saisie à l'écran, celle-ci se fait grâce à un formulaire. La saisie de la matrice se réduit aux seules distances strictement nécessaires afin de limiter les erreurs de saisie. En effet, la diagonale de zéros sera remplie automatiquement, de même que la symétrie de la matrice sera contrôlée par le remplissage automatique de la partie correspondante. Dans un second temps, la version de 2007 a très vite autorisé la possibilité de faire le calcul à partir d'un fichier texte de données plutôt qu'une saisie à l'écran. L'avantage étant de récupérer les données existantes tout en limitant les erreurs de saisie ; en revanche, il fallait s'assurer que les données respectaient le format de matrice carrée, symétrique et de diagonale nulle.

Les contrôles de saisie de formulaire sont réalisés en Javascript, pour être effectués côté client et gagner du temps de calcul en évitant les allers-retours avec le serveur. En contrepartie des fenêtres d'alerte de ces contrôles alourdissent l'application lorsque le client ne contourne pas la validation en Javascript. Le premier contrôle porte sur la saisie du nombre d'espèces : l'application ne permet pas la saisie manuelle pour plus de 20 espèces, l'écran n'étant pas le moyen le plus judicieux pour cela à cause du risque d'erreur élevé en cas de données nombreuses à saisir (pour 20 espèces, il faut saisir 190 valeurs). Le second contrôle porte sur le label à donner aux espèces retenues pour le calcul. On souhaite que tous les en-têtes soient renseignés. On interdit également que des libellés soient identiques. Le troisième contrôle porte sur le contenu de la matrice. Par simplification, les cases non renseignées sont considérées nulles. Les seules touches autorisées sont les chiffres et la marque décimale anglo-saxonne. Ainsi lorsque deux marques décimales sont saisies, le nombre est tronqué (par exemple une erreur de saisie avec deux marques décimales, 1.2.3 devient 1.2, ou encore 2..3 devient 2.).

Tous ces contrôles en saisie ont rapidement conduit à proposer la possibilité de déposer un fichier texte en guise de matrice, charge à l'utilisateur de saisir une matrice qui convienne, et, dans cette version, la redondance des libellés n'est alors pas vérifiée, pas plus que la marque décimale ; la symétrie et la diagonale de zéros sont vérifiées avant de commencer le calcul.

L'algorithme a été vérifié à l'aide d'exemples : les quatre espèces de primates, les 19 races de vaches ou les 15 espèces de grues de Weitzman (1993) ; ainsi que des exemples fictifs. Weitzman lui-même n'a pas proposé d'autres exemples de calcul sur un nombre plus important et annonçait la limite de son calcul pour 30 espèces.

Le temps de calcul est rapidement apparu comme un facteur limitant de l'application. Pour le vérifier, n'ayant pas de jeu de données suffisamment important, nous avons construit une matrice de distances juste avant de faire le calcul. La distribution des distances est aléatoire entre les valeurs 100 et 500 par exemple. Pour 100 espèces, la génération a créé 10000 lignes correspondant aux 10000 couples de distances, dans la table de travail. À partir de

300 espèces, quelques difficultés de paramétrage sont apparues pour redimensionner la mémoire utile ainsi que le temps alloué au traitement en PHP.

Afin d'assurer sa valorisation, l'application ainsi réalisée a été déposée à l'Agence de protection des programmes sous le numéro : IDDN.FR.001.080016.000.R.P.2009.000.31235.

Deuxième version du calculateur V2020

Les axes de recherche des équipes ayant changé, il aura fallu 13 ans et surtout la période du confinement pour retrouver du temps pour réécrire l'application de calcul de l'indice de Weitzman. R étant également largement utilisé dans les sciences sociales, il a été préféré à Python.

Le développement, la saisie ou l'entrée d'un fichier et le calcul sont maintenant tous réalisés sous R à l'aide de Rstudio. La principale différence vient du choix de ne pas saisir la matrice directement. La validation du calcul donc de l'algorithme a été réalisée avec les mêmes exemples. N'ayant pas davantage de données réelles pour faire les tests sur de grandes matrices, le calcul a été réalisé à partir de matrices de données aléatoires, similaires à celles construites pour simuler les calculs lors de l'utilisation de la version 2007 du calculateur.

Comparaison entre les deux versions

La première version n'est plus utilisée. L'objectif de ce paragraphe est d'exprimer en quoi la seconde version est plus pertinente que la première à l'aide de critères de comparaison objectifs présents dans les tableaux suivants.

Trois tableaux (Tableaux 1 à 3) permettent la comparaison : le premier permet un survol des principaux points de comparaison possibles en termes de développement de l'application, le second tableau affiche des résultats de temps de calcul selon le nombre d'espèces traitées avec de l'aléa ou non, et enfin le troisième tableau pose quelques limites de chaque version.

Tableau 1. Comparaison portant sur le développement

Développement	**Version V2007**	**Version V2020**
Temps de développement À nuancer par...	2 mois Application web à finaliser	10 jours Déroulement déjà en tête
Nombre de scripts	4, ce nombre est lié à ce qui était attendu dans le cahier des charges : application web	1

Nombre de lignes	Environ 1000 lignes	150 lignes
Outils Remarques	PHP, MySQL, HTML, Javascript Outils libres mais plusieurs technologies	R Un seul outil libre
Orientation du calcul	Application web intégrée Approche BD relationnelle Traitement algorithmique horizontal (tuple)	Approche matricielle Traitement algorithmique vertical (matrice)
Machine et logiciels	Processeur AMD Athlon 64 3.4GHz, Linux Mandrake 10.0, Xampp 1.6.1 pour Linux (base MySQL 5.0.37, serveur Apache 2.2.4, PHP 5.2.1)	Processeur 8 Cores Intel Core, 3,6GHz Linux Mageia RStudio 2009-2013 V99.9.9

Le tableau 1 présente quelques éléments de comparaison des deux versions concernant les contraintes matérielles spécifiques à chacune d'elle.

Tableau 2. Comparaison portant sur les temps de calcul

Temps de calcul		Version V2007	Version V2020
Matrice de données simples (1,0)	100 espèces	≈ 1 min	≈ 2 sec
	500 espèces	≈ 2 h	≈ 4 min
	1000 espèces	> 6 j	≈ 35 min
Matrice de données aléatoires	100 espèces	≈ 1 h	< 1 sec
	500 espèces	≈ 19 h	≈ 15 sec
	1000 espèces	≈ 6 j	≈ 2 min

Les premiers calculs avec la seconde version montrent clairement un gain de temps par rapport aux calculs faits à l'aide de la première version. Cependant la comparaison n'est pas conforme à une expérience traditionnelle puisque les tests n'ont pas été réalisés dans des conditions similaires (ordinateur, processeur, mémoire). Les temps de calcul, qui restent fonction du nombre d'espèces, sont significativement plus courts avec la nouvelle version. Le tableau 2 des temps de calcul met bien en évidence le gain de temps octroyé par la seconde version par rapport à la première. Par ailleurs, le traitement des matrices où ne figurent que des 0 et des 1 est plus lent que le traitement de matrices remplies de données aléatoires, car il faut aller jusqu'à la dernière espèce pour différencier celles qui sont les plus proches. En effet, les données identiques obligent à analyser chaque vecteur pour s'assurer qu'il s'agit bien de la distance la plus petite lorsque la dernière distance est triée.
Avec la version de 2020, les calculs sont toujours réalisés grâce aux traitements de matrices importantes remplies de données aléatoires pour simuler de véritables calculs sur des données d'échantillons conséquents, et les temps de calculs sont améliorés.

Tableau 3. Comparaison portant sur quelques limites

Limites	Version V2007	Version V2020
Saisie	Plus ergonomique en web, mais limitée en taille	Fichier .txt convenable et il existe des mots réservés : par exemple T pour TRUE ne peut pas être utilisé comme libellé
Déploiement	Aurait pu être très utilisée par une large diffusion sur le net	Dépendante de l'utilisation de l'outil R (disponibilité, maîtrise)
Valorisation	Dépôt APP de cette version	La première déclaration vaut pour la méthode de calcul comme pour l'algorithme sur le code

Le tableau 3 affiche quelques limites observées. Les développements de chaque version ont été réalisés sur la base de mes connaissances sur les outils utilisés dans la communauté des sciences sociales au moment du déploiement.

Conclusion

L'application développée en 2007 permettait déjà un calcul de l'indice de biodiversité beaucoup plus rapide que la démarche proposée par Weitzman. Le gain portait autant sur le temps de calcul, Weitzman utilisant un algorithme récursif gourmand en temps, que sur le nombre d'espèces, puisque l'algorithme de Weitzman limitait le nombre d'espèces à environ 30. Le calcul proposé par la version V2020 permet des accélérations importantes et pourrait donc être utilisé, par exemple, à des fins de comparaison d'échantillons entre eux, afin d'analyser la biodiversité selon Weitzman entre deux lieux critiques de biodiversité pour ne citer que ces exemples d'utilisation du calculateur.

Bibliographie

Aulong S, Erdlenbruch K, Figuières C (2005) Un tour d'horizon des critères d'évaluation de la biodiversité biologique. *Public economics / Institut d'économie publique (IDEP)* URL : http://journals.openedition.org/economiepublique/1713, 16.

Aulong S, Erdlenbruch K, Figuières C (2008) Les critères d'évaluation de la biodiversité : propriétés et difficultés d'usage. *Inra Sciences Sociales* 4-5, Septembre.

Weitzman M (1993) What to preserve? An application of diversity theory to crane conservation. *The Quarterly Journal of Economics* 108(1), 157-183.

Annexes

Programme V2020

Fichier R calculind.R

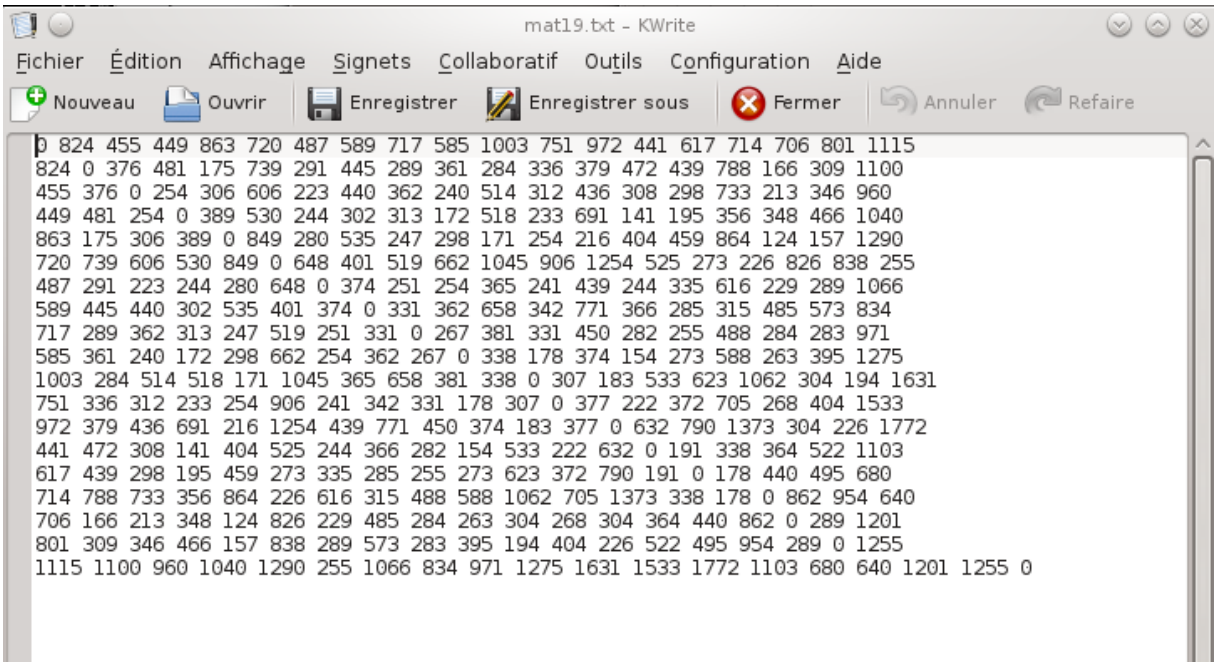
```
# calculind.R
# calcul de l'indice de biodiversite
# auparavant calcule avec appli web, HTML + PHP + MySQL
# mai 2020
# calcul a partir d'une matrice sur fichier txt
# calcul si la matrice est diagonale
# matrice.txt avec separateur espace attention fin de ligne sans entete

# fichier d'entree passe en argument par la commande Rscript
# Rscript calculind mat100.tct

rm(list=ls()) # remise a zero de toutes les variables
args <- commandArgs(TRUE)
T1 <- Sys.time()

############################## matrice ##############################
# remplissage a partir d un fichier txt
setwd(dir="~/test/R")

data_txt <- read.table(args[1], header = FALSE,sep=" ") # lecture
mat_init <- data.matrix(data_txt)  # transformation en matrice

# verification de la faisabilite du calcul a partir de la matrice initiale

if (ncol(mat_init)==nrow(mat_init)) {
  # print("la matrice carree sera prise en compte ")
  nbes <-ncol(mat_init)
} else {
  print("matrice non carree")
}

diago <- 1 # verification de la diagonalisation avec des 0 de la matice initiale

if (max(diag(mat_init)) != 0) {
  print("la diagonale n'est pas avec des 0")
  diago <- 0}

for (j in 1:nbes){
  for (i in 1: nbes) {
    if (mat_init[j,i] != mat_init[i,j]) {
      diago <- 0 # au moins un elements de la matrice fait qu'elle n'est pas diaognale
    }
  }
}

############################## calcul ##############################

nbes <-ncol(mat_init)
indicebio <- 0
val_dist_retenue_petite <- 0
nbcol <- 0
nblg <- 0
j <- 2
i <- 1
rejete <- 0

if (diago == 0 ) {
  print("La matrice n'est pas diagonale-zéro ni symétrique ni carree") # on ne fait rien
} else { # sinon on calcule

  print("La matrice est diagonale, symétrique et carrée")
  while (nbes>2) { # tant qu'on a plus de 2 especes ; matrice autre que 0 1dim
    mat_tempo <- apply(mat_init,1,sort,decreasing=F) # matrice triee
    nbcol <- ncol(mat_tempo)
    cptmin_glob <- nbes
    val_dist_retenue_petite <- min(mat_tempo[2,]) # recup pour calcul indice
    garde <- 0
```

```
while (j<nbcol & garde==0) { # tant qu'on peut descendre dans les lignes
  cptmin_lg <- 0
  minlg <- min(mat_tempo[j,])
  min_temp <- min(mat_tempo[j,1])
  garde_i <- cbind(0)

  for (i in 1:nbcol) { # prendre les colonnes pour trouver le min des min
    if (mat_tempo[j,i]==min(mat_tempo[j,])) {
      cptmin_lg <- cptmin_lg + 1
      garde_i <- cbind(garde_i,i)
    }
    garde_lg <- garde_i[,-1]
    #print(garde_lg)
  } # fin pour ligne

  if (length(garde_lg)==1) { # on a trouve un min
    rejete <- garde_lg
    garde <- 1
  } else { # on a trouve plusieurs min
    jj <- j+1

    while (jj<=nbes & length(garde_lg)>1) { # tant qu'il y a des especes ou rejete pas switche
      kfin <- length(garde_lg)  #
      #min_temp <- mat_tempo[jj,garde_lg[1]] # min temp est le premier element a inspecter
      cptminlg_temp <- 0

      # recup min de la ligne jj parmi les colonnes a etudier
      minlg_temp_tot <- cbind(0) # on met le zero pour initier
      for (k in 1:kfin) {
        minlg_temp_tot <- cbind(minlg_temp_tot,mat_tempo[jj,garde_lg[k]])
      }
      minlg_temp_tot <- minlg_temp_tot[,-1] # on enleve le 0 qu'on a mis
      minlg_temp <- min(minlg_temp_tot)

      garde_i_temp <- cbind(0)

      for (k in 1:kfin) { # pour chaque objet des col retenues de la ligne suivante
        if (mat_tempo[jj,garde_lg[k]] == minlg_temp) {
          cptminlg_temp <- cptminlg_temp + 1
          garde_i_temp <- cbind(garde_i_temp, garde_lg[k])
        }

      }

      garde_lg_temp <- garde_i_temp[,-1] # enlever le 0 du cbind initial
      #print(garde_lg_temp)

      # faire pseudo raz pour reprendre la boucle jj
      garde_lg <- garde_lg_temp
      jj <-jj+1

      } # fin du while sur colonne jj

    if (length(garde_lg_temp)==1) {
      rejete <- garde_lg_temp # on prend celui qui reste
      garde <-1
    } else {

      garde_i_temp <- garde_i_temp[,-1]
      rejete <- sample(garde_i_temp,size=1) # alea parmi ceux qui restent
      garde <-1
    }
```

```
        cptmin_glob <- cptmin_lg
        j <- j+1
        remove(jj)
      } # fin du si sur rejete

      indicebio <- indicebio + val_dist_retenue_petite
      mat_init <- mat_init[-rejete,-rejete]
      i <- 1
      j <- 2
      nbes <- ncol(mat_init)
      remove(garde_i)
    } # fin du while sur espece tant qu'on peut descendre dans les lignes

  }

  indicebio <- indicebio + mat_init[2,1] # dernier tour avec 2 especes
  print("L'indice de biodiversit est : ")
  print(indicebio)

  remove(indicebio, val_dist_retenue_petite,i,j,k,kfin)

} # fin si diago donc calcul

T2 <- Sys.time()
Tdiff = difftime(T2,T1)
print(Tdiff) # temps de calcul
```

Matrice 19 races de vaches cf. Weitzman

Fichier mat19.txt passé en argument de la commande.



Exécution

Ligne_de_commande_Linux% **Rscript calculind.R mat19.txt**