q\_diversity\_functions.R

z3254626

2021-07-07

# Functions to calculate q-profile diversity measures  
# From a vector of allele proportions (p)  
# Assuming biallelic loci (i.e. p + (1-p) = 1)  
# Both Entropy (H) and Diversity (D)  
# Both Alpha (within population) and Beta (between population) measures  
# For Beta measures, H/D is calculated per locus,   
# Gamma calculations should use Alpha methods when all populations are pooled  
  
### Alpha Diversity Functions  
  
# For alpha diversity, take minor allele proportion of (p), and value of q  
# p can either be a single value, or a vector of values  
# each p is calculated independently,   
# as you are calculating the diversity of each LOCUS (not all loci together)  
# p can only range from 0 (100% one type) to 1 (100% alternate type)  
# Missing data (NA), returns NA  
get.Hq.alpha <- function (p, q){  
  
 #Return error if q is not 0, 1, or 2  
 if (!(q == 0|q == 1|q == 2)) {  
 stop("Invalid value of q. Must be one of: 0, 1, 2")  
 }  
   
 #Return error if p>1 or p<0  
 if (any(p > 1, na.rm = T) | any(p < 0, na.rm = T) ) {  
 stop("Invalid value(s) of p. Must be between 0 and 1")  
 }  
   
 if (q == 0) {  
 entropy <- ifelse((p == 0) | (p == 1), 0, 1)  
 }  
   
 if (q == 1) {  
 entropy <- ifelse((p == 0) | (p == 1), 0,   
 -(p \* log(p)) - ((1 - p) \* log(1 - p)))  
 }  
   
 if (q == 2) {  
 entropy <- 1 - p\*p - (1-p)\*(1-p)  
 }  
   
 return(entropy)  
}  
  
# Take entropy (H) and convert it to effective number diversity (D), for q = 0,1,2  
# Works with a single H, or H as a vector of values  
# Note: to get average values of D, you should average H THEN convert to D  
H.to.D.alpha <- function(H, q){  
   
 #Return error if q is not 0, 1, or 2  
 if (!(q == 0|q == 1|q == 2)) {  
 stop("Invalid value of q. Must be one of: 0, 1, 2")  
 }  
  
 if (q == 0) {  
 div <- H + 1  
 }  
   
 if (q == 1) {  
 div <- exp(H)  
 }  
   
 if (q == 2) {  
 div <- 1/(1-H)  
 }  
   
 return(div)  
}  
  
# Takes a vector of p (including a vector of length 1)  
# It will only return a \*single\* D value (mean of H values, THEN converted to D)  
# If you want to get average D measures, you should average H measures first  
# THEN convert to a D measure, NOT the other way round  
# (see Jensen's inequality for why)  
get.Dq.alpha <- function (p, q){  
  
 div <- H.to.D.alpha(mean(get.Hq.alpha(p, q), na.rm = T), q)  
   
 return(div)  
}  
  
### Beta calculations  
  
# Accepts two vectors of allele proportions to be compared (p1 and p2)  
# And the value of q you want to calculate (0, 1, or 2)  
# per.locus = T (locus variant)  
# Returns a vector of entropies  
# per.locus = F (global variant)  
# Returns only 1 value, not a vector  
# Optional variants for:  
# q0measure - "Jaccard"/"Sorenson"  
# q1measure - "Mutual Information"/"Shannon Differentiation"  
# q2measure - "Jost-D"/ "GST"  
get.Hq.beta <- function (p1, p2, q, per.locus = T,  
 q0measure = "Jaccard",   
 q1measure = "Mutual Information",   
 q2measure = "Jost-D"){  
   
 if (!(q == 0|q == 1|q == 2)) stop("Invalid value of q. Must be one of: 0, 1, 2")  
   
 p.av <- (p1 + p2)/2 #Average minor allele proportion  
   
 #When calculating entropy for each locus   
 if (per.locus == T) {  
   
 #Mean alpha diversity of localities, per locus  
 Hqa.mean <- (get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2  
   
 Hqa.mean.plus1 <- (get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q) + 2)/2  
   
 #Gamma diversity of localities, per locus  
 Hqgamma <- get.Hq.alpha(p.av, q)  
   
 #Number of shared alleles, per locus  
 shared <- ifelse(p1 %% 1 > 0 & p2 %% 1 > 0 ,2, #both alleles shared  
 ifelse((p1 == 0 & p2 == 1)| (p1 == 1 & p2 == 0 ), 0, #no shared alleles  
 1)) #else, one shared allele  
 }  
   
 #When calculating global variant of diversity  
 #Average entropies for each locus before calculating beta  
 if (per.locus == F) {  
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 Hqa.mean <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, na.rm = T)  
   
 Hqa.mean.plus1 <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q) + 2)/2, na.rm = T)  
   
 #AVERAGE Gamma diversity of localities, across all loci  
 Hqgamma <- mean(get.Hq.alpha(p.av, q), na.rm = T)  
   
 shared <- mean(ifelse(p1 %% 1 > 0 & p2 %% 1 > 0 , 2, #both alleles shared  
 ifelse((p1 == 0 & p2 == 1)|(p1 == 1 & p2 == 0 ), 0, #no shared alleles  
 1)), #else, one shared alleles  
 na.rm = T)  
 }  
   
 if (q == 0) {  
 #Jaccard  
 if (q0measure == "Jaccard") entropy <- 1 - (shared/(Hqgamma + 1))  
 #Sorenson  
 if (q0measure == "Sorenson") entropy <- 1 - (shared/Hqa.mean.plus1)  
 }  
   
 if (q == 1) {  
 #Mutual Information (I)  
 if (q1measure == "Mutual Information") entropy <- (Hqgamma - Hqa.mean)  
 #Shannon differentiation - I normalised to a [0,1] scale  
 if (q1measure == "Shannon Differentiation") entropy <- (Hqgamma - Hqa.mean)/log(2)  
 }  
   
 if (q == 2) {  
 #Jost-D  
 if (q2measure == "Jost-D") entropy <- ((Hqgamma - Hqa.mean)/(1 - Hqa.mean)) \* 2   
 #Gst  
 if (q2measure == "GST") entropy <- ifelse(Hqgamma == 0, 0, (Hqgamma - Hqa.mean)/Hqgamma)  
 }  
   
 return(entropy)  
}  
  
get.Hq.beta.sd <- function (p1, p2, q,   
 q0measure = "Jaccard",   
 q1measure = "Mutual Information",   
 q2measure = "Jost-D"){  
   
 if (!(q == 0|q == 1|q == 2)) stop("Invalid value of q. Must be one of: 0, 1, 2")  
   
 p.av <- (p1 + p2)/2 #Pooled allele proportion  
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 A\_mean <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, na.rm = T)  
 #sd  
 A\_var <- var((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, na.rm = T)  
   
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 S\_mean <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2 + 1, na.rm = T)  
 #sd  
 S\_var <- var((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2 + 1, na.rm = T)  
   
 #AVERAGE Gamma diversity of localities, across all loci  
 G\_mean <- mean(get.Hq.alpha(p.av, q), na.rm = T)  
 #sd  
 G\_var <- var(get.Hq.alpha(p.av, q), na.rm = T)  
   
   
 R\_mean <- mean(ifelse(p1 %% 1 > 0 & p2 %% 1 > 0 , 2, #both alleles shared  
 ifelse((p1 == 0 & p2 == 1)|(p1 == 1 & p2 == 0 ), 0, #no shared alleles  
 1)), #else, one shared alleles  
 na.rm = T)  
   
 R\_var <- var(ifelse(p1 %% 1 > 0 & p2 %% 1 > 0 , 2, #both alleles shared  
 ifelse((p1 == 0 & p2 == 1)|(p1 == 1 & p2 == 0 ), 0, #no shared alleles  
 1)), #else, one shared alleles  
 na.rm = T)  
   
   
 if (q == 0) {  
 #Jaccard  
 if (q0measure == "Jaccard") entropy\_var <- R\_var/((G\_mean +1)^2 )+ ((R\_mean^2)\*G\_var)/((G\_mean +1)^4)  
 #Sorenson  
 if (q0measure == "Sorenson") entropy\_var <- 4\*R\_var/(S\_mean^2) + 4\*((R\_mean^2)\*R\_var)/(S\_mean^4)  
 }  
   
 if (q == 1) {  
 #Mutual Information (I)  
 if (q1measure == "Mutual Information") entropy\_var <- G\_var + A\_var  
 #Shannon differentiation - I normalised to a [0,1] scale  
 if (q1measure == "Shannon Differentiation") entropy\_var <- (G\_var + A\_var) / (log(2)^2)  
 }  
   
 if (q == 2) {  
 #Jost-D  
 if (q2measure == "Jost-D") entropy\_var <- 4 \* (G\_var/(A\_mean +1)^2) + (4\*(G\_mean -1)^2 \*A\_var)/ ((A\_mean +1)^4)  
 #Gst  
 if (q2measure == "GST") entropy\_var <- (A\_mean^2 \* G\_var)/(G\_mean^4) + G\_var/(G\_mean^2)  
 }  
   
 return(sqrt(entropy\_var)) #standard deviation  
}  
  
# Accepts two vectors of allele proportions to be compared  
# per.locus = T  
# Returns a vector of diversities  
# per.locus = F  
# Returns only 1 value, not a vector  
get.Dq.beta <- function (p1, p2, q, per.locus = T){  
   
 if (!(q == 0|q == 1|q == 2)) stop("Invalid value of q. Must be one of: 0, 1, 2")  
   
 p.av <- (p1 + p2)/2 #Pooled allele proportion  
   
 if (per.locus == T) {  
 #Mean alpha diversity of localities, per loci  
 #Then converted to D, per loci  
 Dqa.mean <- H.to.D.alpha((get.Hq.alpha(p1, q) + (get.Hq.alpha(p2, q)))/2, q)  
   
 #Gamma diversity of localities, per loci  
 #Then converted to D  
 Dqgamma <- H.to.D.alpha(get.Hq.alpha(p.av, q), q)  
 }  
   
 if (per.locus == F) {  
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 Hqa.mean <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, na.rm = T)  
   
 #Then converted to D  
 Dqa.mean <- H.to.D.alpha(Hqa.mean, q)  
   
 #AVERAGE Gamma diversity of localities, across all loci  
 #Then converted to D  
 Dqgamma <- H.to.D.alpha(mean(get.Hq.alpha(p.av, q), na.rm = T), q)  
 }  
  
 #Calculate diversity (D) measure   
 #works for vectors (per.locus) or single values (global)  
 div <- Dqgamma/Dqa.mean  
   
 return(div)  
}  
  
get.Dq.beta.sd <- function (p1, p2, q){  
   
 if (!(q == 0|q == 1|q == 2)) stop("Invalid value of q. Must be one of: 0, 1, 2")  
   
 p.av <- (p1 + p2)/2 #Pooled allele proportion  
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 A\_mean <- mean(H.to.D.alpha((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, q), na.rm = T)  
   
 A\_var <- var(H.to.D.alpha((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, q), na.rm = T)  
   
   
 #AVERAGE Gamma diversity of localities, across all loci  
 #Then converted to D  
 G\_mean <- mean(H.to.D.alpha(get.Hq.alpha(p.av, q), q), na.rm = T)  
   
 G\_var <- var(H.to.D.alpha(get.Hq.alpha(p.av, q), q), na.rm = T)  
   
   
 #Calculate diversity (D) measure   
 #works for vectors (per.locus) or single values (global)  
 div\_var <- (G\_var)/(A\_mean^2) + (G\_mean ^2 \* A\_var)/(A\_mean^4)  
   
 return(sqrt(div\_var))  
}  
  
get.Hq.relative.beta <- function (p1, p2, q, per.locus = T,  
 q0measure = "Jaccard",   
 q1measure = "Mutual Information",   
 q2measure = "Jost-D"){  
   
 if (!(q == 0|q == 1|q == 2)) stop("Invalid value of q. Must be one of: 0, 1, 2")  
   
 p.av <- (p1 + p2)/2 #Pooled allele proportion  
   
 #When calculating entropy for each locus   
 if (per.locus == T) {  
   
 #Mean alpha diversity of localities, per locus  
 Hqa.mean <- (get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2  
   
 Hqa.mean.plus1 <- (get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q) + 2)/2  
   
 #Gamma diversity of localities, per locus  
 Hqgamma <- get.Hq.alpha(p.av, q)  
   
 #Number of shared alleles, per locus  
 shared <- ifelse(p1 %% 1 > 0 & p2 %% 1 > 0 , 2, #both alleles shared  
 ifelse((p1 == 0 & p2 == 1)|(p1 == 1 & p2 == 0 ), 0, #no shared alleles  
 1)) #else, one shared allele  
 }  
   
 #When calculating global entropy  
 #Average entropies for each locus before calculating beta  
 if (per.locus == F) {  
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 Hqa.mean <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, na.rm = T)  
   
 Hqa.mean.plus1 <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q) + 2)/2, na.rm = T)  
   
 #AVERAGE Gamma diversity of localities, across all loci  
 Hqgamma <- mean(get.Hq.alpha(p.av, q), na.rm = T)  
   
 shared <- mean(ifelse(p1 %% 1 > 0 & p2 %% 1 > 0 , 2, #both alleles shared  
 ifelse((p1 == 0 & p2 == 1)|(p1 == 1 & p2 == 0 ), 0, #no shared alleles  
 1)), #else, one shared alleles  
 na.rm = T)  
 }  
   
 if (q == 0) {  
 #Jaccard  
 if (q0measure == "Jaccard") entropy <- 1 - (shared/(Hqgamma + 1))  
 #Sorenson  
 if (q0measure == "Sorenson") entropy <- 1 - (shared/Hqa.mean.plus1)  
 }  
   
 if (q == 1) {  
 #Mutual Information (I)  
 if (q1measure == "Mutual Information") entropy <- (Hqgamma - Hqa.mean)  
 #Shannon differentiation - I normalised to a [0,1] scale  
 if (q1measure == "Shannon Differentiation") entropy <- (Hqgamma - Hqa.mean)/log(2)  
 }  
   
 if (q == 2) {  
 #Jost-D  
 if (q2measure == "Jost-D") entropy <- ((Hqgamma - Hqa.mean)/(1 - Hqa.mean)) \* 2   
 #Gst  
 if (q2measure == "GST") entropy <- ifelse(Hqgamma == 0, 0, (Hqgamma - Hqa.mean)/Hqgamma)  
 }  
   
 return(entropy/Hqa.mean)  
}  
  
  
get.Dq.relative.beta <- function (p1, p2, q, per.locus = T){  
   
 if (!(q == 0|q == 1|q == 2)) stop("Invalid value of q. Must be one of: 0, 1, 2")  
   
 p.av <- (p1 + p2)/2 #Pooled allele proportion  
   
 if (per.locus == T) {  
 #Mean alpha diversity of localities, per loci  
 #Then converted to D, per loci  
 Dqa.mean <- H.to.D.alpha((get.Hq.alpha(p1, q) + (get.Hq.alpha(p2, q)))/2, q)  
   
 #Gamma diversity of localities, per loci  
 #Then converted to D  
 Dqgamma <- H.to.D.alpha(get.Hq.alpha(p.av, q), q)  
 }  
   
 if (per.locus == F) {  
   
 #AVERAGE Mean alpha diversity of localities, across all loci  
 Hqa.mean <- mean((get.Hq.alpha(p1, q) + get.Hq.alpha(p2, q))/2, na.rm = T)  
   
 #Then converted to D  
 Dqa.mean <- H.to.D.alpha(Hqa.mean, q)  
   
 #AVERAGE Gamma diversity of localities, across all loci  
 #Then converted to D  
 Dqgamma <- H.to.D.alpha(mean(get.Hq.alpha(p.av, q), na.rm = T), q)  
 }  
   
 #Calculate diversity (D) measure   
 #works for vectors (per.locus) or single values (global)  
 div <- Dqgamma/(Dqa.mean^2)  
   
 return(div)  
}  
  
  
# Accepts two vectors of allele proportions to be compared  
# Returns a vector of Bray-Curtis  
get.BC <- function (p1, p2) abs(p1-p2)  
  
#Calculate relative Bray-Curtis  
get.RBC <- function (p1, p2) abs((p1-p2)/((p1+p2)/2))