simulate\_data.R

z3254626

2021-07-07

Simulate genetic data and check for steps

@title simulate\_data @param step strength of step: 0 (linear) to x (steep step e.g. 50) @param p.start starting allele proportion, 0 to 1 @param p.end end allele proportion, 0 to 1 @param n.samples number of genomes sampled from a locality @param n.loci number of loci per genome @param n.pops number of localities sampled along gradient @param detect\_step T/F for detecting steps

Calculated variables d = distance from 0 to 1, increments of increment.size - 1/(n.pops - 1)

@return outputdata\_summary table of simulated data @return output$step\_results table of step detection results

@author Alex Sentinella

simulate\_data <- function(step = 0, p.start = 0, p.end = 1, n.samples = 20,   
 n.loci = 1000, n.pops = 10, detect\_step = T) {  
   
 #Make a table of the variables used for this run  
 data\_variables <- tibble(step, n.samples, p.start, p.end, n.loci, n.pops)  
   
 #Add variables to output  
 output <- list(variables = data\_variables)  
   
 #Make a variable which is the distance between each value of d (distance)  
 increment.size = 1/(n.pops - 1)  
  
 #Create a table with input variables and calculate p along distance (d)  
 data <- tibble(  
 step = step,  
 n.samples = n.samples,  
 i = increment.size,  
 n.pops = n.pops,  
 p.start = p.start,  
 p.end = p.end,  
 d = seq(0, 1, increment.size)) %>% #Location of population along distance (0 to 1)  
 mutate(p = qbeta(d, 1 / (1 + step), 1 / (1 + step)) \* #true population allele frequency  
 (p.end - p.start) + p.start) #offsets allele proportion from 0 and 1  
   
 ## Take random samples from p to get variable allele frequencies  
 # Take n.samples from a binomial distribution around allele proportion p  
 # Divide by n.samples to get 'measured' allele frequency  
 # e.g. n.samples = 5, allele proportion p = 0.1  
 # rbinom(1, 5, 0.1) / 5  
 # Can only be 0, 0.2, 0.4, 0.6, 0.8, 1  
 # But would more likely be 0/0.2   
 # repeat over n.loci  
 data <- data %>%   
 .[rep(1:nrow(.), times = n.loci),] %>% #replicate over multiple loci with same p  
 rowwise() %>% #allows for mutate to work row by row (rather than as a vector)  
 mutate(p.binom = rbinom(1, n.samples, p)/n.samples) %>% #p.binom  
 ungroup() #stops rowwise operations  
   
   
 ## Alpha diversities  
  
 #Calculate alpha diversities of each p.binom (i.e. sampled allele frequency)  
 #Calculated at the locus level  
 data <- data %>%  
 mutate(  
 H0a = get.Hq.alpha(p.binom, 0),  
 H1a = get.Hq.alpha(p.binom, 1),  
 H2a = get.Hq.alpha(p.binom, 2),  
 D0a = H.to.D.alpha(H0a, 0),  
 D1a = H.to.D.alpha(H1a, 1),  
 D2a = H.to.D.alpha(H2a, 2)  
 )  
  
 ## Beta diversities - per locus - AvLast variants  
   
 # Calculate beta diversities of each p.binom (i.e. sampled allele frequencies)  
 # with the lead p.binom (distance + increment.size)  
 #  
 # Calculated at the locus level  
 #  
 # Don't calculate betas comparing distance 1 and 0 (d = 1), return NA instead  
 data <- data %>%  
 mutate(p.lead = lead(p.binom)) %>% #Add a column containing next p.binom  
 mutate(  
 H0b.Jac.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.beta(p.binom, p.lead, 0, q0measure = "Jaccard")),  
 H0b.Sor.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.beta(p.binom, p.lead, 0, q0measure = "Sorenson")),  
 H1b.MI.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.beta(p.binom, p.lead, 1, q1measure = "Mutual Information")),  
 H1b.ShD.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.beta(p.binom, p.lead, 1, q1measure = "Shannon Differentiation")),  
 H2b.JOST.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.beta(p.binom, p.lead, 2, q2measure = "Jost-D")),  
 H2b.GST.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.beta(p.binom, p.lead, 2, q2measure = "GST")),  
 D0b.A.AvLast = if\_else(d == 1, NA\_real\_, get.Dq.beta(p.binom, p.lead, 0)),  
 D0b.B.AvLast = H.to.D.alpha(H0b.Jac.AvLast, 0),  
 D1b.A.AvLast = if\_else(d == 1, NA\_real\_, get.Dq.beta(p.binom, p.lead, 1)),  
 D1b.B.AvLast = H.to.D.alpha(H1b.MI.AvLast, 1),  
 D2b.A.AvLast = if\_else(d == 1, NA\_real\_, get.Dq.beta(p.binom, p.lead, 2)),  
 D2b.B.AvLast = H.to.D.alpha(H2b.JOST.AvLast, 2),  
 BC.AvLast = if\_else(d == 1, NA\_real\_, get.BC(p.binom, p.lead)),  
 RBC.AvLast = if\_else(d == 1, NA\_real\_, get.RBC(p.binom, p.lead)),  
 H0b.Jac.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.relative.beta(p.binom, p.lead, 0, q0measure = "Jaccard")),  
 H0b.Sor.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.relative.beta(p.binom, p.lead, 0, q0measure = "Sorenson")),  
 H1b.MI.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.relative.beta(p.binom, p.lead, 1, q1measure = "Mutual Information")),  
 H1b.ShD.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.relative.beta(p.binom, p.lead, 1, q1measure = "Shannon Differentiation")),  
 H2b.JOST.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.relative.beta(p.binom, p.lead, 2, q2measure = "Jost-D")),  
 H2b.GST.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Hq.relative.beta(p.binom, p.lead, 2, q2measure = "GST")),  
 D0b.A.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Dq.relative.beta(p.binom, p.lead, 0)),  
 D0b.B.rel.AvLast = H.to.D.alpha(H0b.Jac.rel.AvLast, 0),  
 D1b.A.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Dq.relative.beta(p.binom, p.lead, 1)),  
 D1b.B.rel.AvLast = H.to.D.alpha(H1b.MI.rel.AvLast, 1),  
 D2b.A.rel.AvLast = if\_else(d == 1, NA\_real\_, get.Dq.relative.beta(p.binom, p.lead, 2)),  
 D2b.B.rel.AvLast = H.to.D.alpha(H2b.JOST.rel.AvLast, 2),  
 )  
   
 ## Beta diversities - AvFirst (average gamma, alphas before calculating beta)  
   
 # Calculate beta diversities of each p.binom (i.e. sampled allele frequencies)  
 # with the lead p.binom (distance + increment.size)  
 #  
 # Calculated overall per group (each site)  
 #  
 #Don't calculate betas comparing distance 1 and 0 (d =1), return NA instead   
 AvFirst\_data <- data %>%  
 group\_by(d) %>%  
 filter(d != 1) %>% #Avoid calculating at d = 1  
 mutate(  
 H0b.Jac.AvFirst\_mean = get.Hq.beta(p.binom, p.lead, 0, per.locus = F, q0measure = "Jaccard"),  
 H0b.Sor.AvFirst\_mean = get.Hq.beta(p.binom, p.lead, 0, per.locus = F, q0measure = "Sorenson"),  
 H1b.MI.AvFirst\_mean = get.Hq.beta(p.binom, p.lead, 1, per.locus = F, q1measure = "Mutual Information"),  
 H1b.ShD.AvFirst\_mean = get.Hq.beta(p.binom, p.lead, 1, per.locus = F, q1measure = "Shannon Differentiation"),  
 H2b.JOST.AvFirst\_mean = get.Hq.beta(p.binom, p.lead, 2, per.locus = F, q2measure = "Jost-D"),  
 H2b.GST.AvFirst\_mean = get.Hq.beta(p.binom, p.lead, 2, per.locus = F, q2measure = "GST"),  
 D0b.A.AvFirst\_mean = get.Dq.beta(p.binom, p.lead, 0, per.locus = F),  
 D0b.B.AvFirst\_mean = H.to.D.alpha(H0b.Jac.AvFirst\_mean, 0),  
 D1b.A.AvFirst\_mean = get.Dq.beta(p.binom, p.lead, 1, per.locus = F),  
 D1b.B.AvFirst\_mean = H.to.D.alpha(H1b.MI.AvFirst\_mean, 1),  
 D2b.A.AvFirst\_mean = get.Dq.beta(p.binom, p.lead, 2, per.locus = F),  
 D2b.B.AvFirst\_mean = H.to.D.alpha(H2b.JOST.AvFirst\_mean, 2)  
 ) %>%  
 mutate(  
 H0b.Jac.AvFirst\_sd = get.Hq.beta.sd(p.binom, p.lead, 0, q0measure = "Jaccard"),  
 H0b.Sor.AvFirst\_sd = get.Hq.beta.sd(p.binom, p.lead, 0, q0measure = "Sorenson"),  
 H1b.MI.AvFirst\_sd = get.Hq.beta.sd(p.binom, p.lead, 1, q1measure = "Mutual Information"),  
 H1b.ShD.AvFirst\_sd = get.Hq.beta.sd(p.binom, p.lead, 1, q1measure = "Shannon Differentiation"),  
 H2b.JOST.AvFirst\_sd = get.Hq.beta.sd(p.binom, p.lead, 2, q2measure = "Jost-D"),  
 H2b.GST.AvFirst\_sd = get.Hq.beta.sd(p.binom, p.lead, 2, q2measure = "GST"),  
 D0b.A.AvFirst\_sd = get.Dq.beta.sd(p.binom, p.lead, 0),  
 D0b.B.AvFirst\_sd = H0b.Jac.AvFirst\_sd, #same sd as H  
 D1b.A.AvFirst\_sd = get.Dq.beta.sd(p.binom, p.lead, 1),  
 D1b.B.AvFirst\_sd = H1b.MI.AvFirst\_sd, #same sd as H  
 D2b.A.AvFirst\_sd = get.Dq.beta.sd(p.binom, p.lead, 2),  
 D2b.B.AvFirst\_sd = H2b.JOST.AvFirst\_sd #same sd as H  
 ) %>%  
 summarise(across( everything(), mean)) %>%  
 select("d", ends\_with("\_mean"), ends\_with("\_sd"))  
   
  
 #Create data summary table,   
 #what are the mean and sd of each measure at each distance (d)  
 data\_summary <- data %>%  
 group\_by(d) %>%  
 summarise\_each(list(mean = mean, sd = sd, var = var)) %>%  
 left\_join(AvFirst\_data, by = "d")  
   
 #Add data summary to output  
 output <- c(output, list(data\_summary = data\_summary))  
  
 #Check for presence and location of step (optional)  
 if(detect\_step == T){  
   
 #Names of each beta measure  
 beta\_measure\_names <- c("H0b.Jac", "H0b.Sor", "H1b.MI", "H1b.ShD",   
 "H2b.JOST", "H2b.GST", "D0b.A", "D0b.B",   
 "D1b.A", "D1b.B", "D2b.A", "D2b.B")  
  
 #Names of each beta measure including their by AvLast and AvFirst variant  
 beta\_measures <- c(paste0(beta\_measure\_names, ".AvLast"),  
 paste0(beta\_measure\_names, ".rel.AvLast"),   
 paste0(beta\_measure\_names, ".AvFirst"),   
 "BC.AvLast", "RBC.AvLast")  
  
 #Detect for a step for each measure, also calculate coefficient of variation  
 step\_results <- map\_dfr(beta\_measures, step\_check, data = data\_summary, n.loci)  
  
 #Add step results to output  
 output <- c(output, list(step\_results = step\_results))  
 }  
  
return(output)  
}