FST_prep_and_analysis

Grace Saville

26/04/2022

Preamble

```
library(corrplot)
library(ggplot2)
library(tidyr)
getwd()
setwd("C:/Users/airhe/OneDrive/Documents/Masters/Project 3/kiore-project")
```

Prepping df

```
# load(".RData") # if necessary
data <- read.csv("./data/RStudio/ratsSNPs_clean.csv")
```

```
copy <- data # making a copy
copy[1,1:20] # checking column names
copy <- copy[,-c(2:16)] # removing all but specimen names and SNPs
copy[1,1:20] # checking

copy[copy == "?"] <- "?:?" # replacing single ? with double ? so alleles can be split

x <- data.frame(island = copy$island) # setting up new df for for loop
coln <- as.vector(colnames(copy)) # prepping to paste the column names into the for loop
dim(copy) # 379 rows 283 columns
for (i in 2:283) {
  y <- colsplit(copy[,i], split = ":", names = c(coln[i], paste("blank", i, sep = "."))) # splitting ea
  x <- cbind(x, y) # combining output with current df
  rm(i, y) # removing temp objects
}

# Checking:
# dim(x3) # 379 rows 565 columns
# x2[1:5,1:5]
# x3[1:5,1:5] # comparing the 2 dfs to check the column naming worked correctly

copy <- x
rm(x, coln) # removing excess objects
```

Producing the file necessary for PGDSpider program

```r
copy <- copy[order(copy$island, decreasing = FALSE), ] # ordering df alphabetically by island
print(as.matrix(copy[, 1])) # printing the island names and row numbers

# A=1, T=2, G=3, C=4
copy[copy == "A"] <- "1"
copy[copy == "T"] <- "2"
copy[copy == "G"] <- "3"
copy[copy == "C"] <- "4"

# row numbers in dataset df listed below for each popn.
popnames <- as.character(
  c(
    "pop = Aotea", # 1:10
    "pop = Borneo", # 11:28
    "pop = Doubtful_Sound", # 306
    "pop = Great_Mercury_Island", # 30
    "pop = Halmahera", # 31:42
    "pop = Hatutaa", # 43:63
    "pop = Honuea", # 64:83
    "pop = Kaikura_Island", # 84:103
    "pop = Kamaka", # 104:123
    "pop = Kayangel", # 124:138
    "pop = Late_Island", # 141:161
    "pop = Mainland", # 29, 139, 140, 162, 349, 350 (including Luzon here)
    "pop = Malenge", # 163:174
    "pop = Mohotani", # 175:188
    "pop = Motukawanui", # 189:209
    "pop = New_Britain", # 210:219
    "pop = New_Guinea", # 220:221
    "pop = Normanby_Island", # 223
    "pop = Rakiura", # 224:244
    "pop = Reiono", # 245:265
    "pop = Rimatuu", # 266:284
    "pop = Slipper_Island", # 285:305
    "pop = Sulawesi", # 307:328
    "pop = Tahanea", # 329:348
    "pop = Wake_Island" # 351:370
  )
)

# Creating population dfs
a <- as.data.frame(copy[1:10,]) # Aotea
b <- as.data.frame(copy[11:28,]) # Borneo
c <- as.data.frame(copy[306,]) # Doubtful_Sound
d <- as.data.frame(copy[30,]) # Great_Mercury_Island
e <- as.data.frame(copy[31:42,]) # Halmahera
f <- as.data.frame(copy[43:63,]) # Hatutaa
g <- as.data.frame(copy[64:83,]) # Honuea
h <- as.data.frame(copy[84:103,]) # Kaikura_Island
i <- as.data.frame(copy[104:123,]) # Kamaka
j <- as.data.frame(copy[124:138,])  # Kayangel
```

```r
k <- as.data.frame(copy[141:161,]) # Late_Island
l <- as.data.frame(copy[c(29, 139, 140, 162, 349, 350),]) # Mainland
m <- as.data.frame(copy[163:174,]) # Malenge
n <- as.data.frame(copy[175:188,]) # Mohotani
o <- as.data.frame(copy[189:209,]) #  Motukawanui
p <- as.data.frame(copy[210:219,]) #  New_Britain
q <- as.data.frame(copy[220:221,]) # New_Guinea
r <- as.data.frame(copy[223,]) #  Normanby_Island
s <- as.data.frame(copy[224:244,]) # Rakiura
t <- as.data.frame(copy[245:265,]) # Reiono
u <- as.data.frame(copy[266:284,]) # Rimatuu
v <- as.data.frame(copy[285:305,]) # Slipper_Island
w <- as.data.frame(copy[307:328,]) #  Sulawesi
x <- as.data.frame(copy[329:348,]) # Tahanea
y <- as.data.frame(copy[351:370,]) # Wake_Island

pops <- as.character(c(letters[seq(from = 1, to = 25)])) # list of popn object names
```

```r
ncol(copy) #565
getwd()

sink("./data/PGDSpider/ratsSNPs_PGDSpider_input_CLEAN.txt") # create empty file
cat("rats_SNPS", "npops = 25", "nloci = 282", fill = 1)
cat("\t", fill = FALSE)
cat(colnames(copy[,c(FALSE,TRUE)]), "\n", sep = "\t\t", fill = FALSE) # column/SNP names (even columns
for (i1 in 1:25) {
  cat(popnames[i1], fill = 1) # island name
  foo <- get(pops[i1]) # calling the island object based on the pops vector
  for (i2 in 1:nrow(foo)) {
    cat(as.character(foo[i2, ]), "\n", fill = FALSE, sep = "\t") # printing the SNP rows
  } # inner loop close
} # outer loop close
sink() # closing the sink connection (do not forget!)

rm(i1, i2, foo, popnames, pops)
rm(list = c(letters[seq(from = 1, to = 25)])) # removing excess objects
```

Analysis

```r
popnames <- as.character(
  c(
    "Aotea",
    "Borneo",
    "Doubtful_Sound",
    "Great_Mercury_Island",
    "Halmahera",
    "Hatutaa",
    "Honuea",
    "Kaikura_Island",
    "Kamaka",
    "Kayangel",
    "Late_Island",
    "Mainland", # (inc Luzon here)
    "Malenge",
    "Mohotani",
    "Motukawanui",
    "New_Britain",
    "New_Guinea",
    "Normanby_Island",
    "Rakiura",
    "Reiono",
    "Rimatuu",
    "Slipper_Island",
    "Sulawesi",
    "Tahanea",
    "Wake_Island"
  )
)

pwd <- read.csv("./results/Arlequin_FST/fst_pairwisedistances_only.csv", header = TRUE)
pv <- read.csv("./results/Arlequin_FST/fst_pairwisedistances_pvalues_only.csv", header = TRUE)
colnames(pwd) <- popnames
rownames(pwd) <- popnames
pwd <- as.matrix(pwd)
colnames(pv) <- popnames
rownames(pv) <- popnames
pv <- as.matrix(pv)

x <- t(pwd) # transposed copy
pwd[upper.tri(pwd, diag = FALSE)] <- x[upper.tri(x, diag = FALSE)] # making full matrix (not just lower
x <- t(pv) # transposed copy for p-values
pv[upper.tri(pv, diag = FALSE)] <- x[upper.tri(x, diag = FALSE)]
```

Loading the results files

```
pwd.df <- pwd
pwd.df[lower.tri(pwd.df, diag = TRUE)] <- NA # keeping only the upper triangle of each matrix

pwd.df <- data.frame(
  col = colnames(pwd.df)[col(pwd.df)],
  row = rownames(pwd.df)[row(pwd.df)],
  fst.dist = c(pwd.df)
) # converting the fst matrix into a df with columns describing which combos result in the distance

pwd.df <- na.omit(pwd.df)

pwd.df <- unite(pwd.df, islands.combo, 1:2, sep = ":", remove = TRUE) # combining the first 2 columns (
longlat <- data[,c(8,11,12)]
longlat <- longlat[!duplicated(longlat$island.1),] # keeping only 1 coordinate for each island
longlat <- longlat[order(longlat$island.1, decreasing = FALSE),] # sorting alphabetically
row.names(longlat) <- seq(nrow(longlat)) # renaming row numbers to be sequential
longlat # checking

# editing the names to match those in the pwd df so I can merge them later
longlat[1,1] <- "Aotea"
longlat[3,1] <- "Doubtful_Sound"
longlat[4,1] <- "Great_Mercury_Island"
longlat[8,1] <- "Kaikura_Island"
longlat[11,1] <- "Late_Island"
longlat[17,1] <- "New_Britain"
longlat[18,1] <- "New_Guinea"
longlat[19,1] <- "Normanby_Island"
longlat[20,1] <- "Rakiura"
longlat[22,1] <- "Rimatuu"
longlat[23,1] <- "Slipper_Island"
longlat[26,1] <- "Wake_Island"


geo.matrix <- as.matrix(longlat[,c(3,2)]) # distGeo function needs a matrix with 2 columns, col 1 longi

geo.matrix <- distm(geo.matrix, fun = distGeo) # converting to pairwise distance matrix
dim(geo.matrix) # 26 26
```

```


colnames(geo.matrix) <- longlat[,1]
rownames(geo.matrix) <- longlat[,1] # naming the rows and columns

geo.matrix[lower.tri(geo.matrix, diag = TRUE)] <- NA # keeping only the upper triangle of matrix

geo.df <- data.frame(
  col = colnames(geo.matrix)[col(geo.matrix)],
  row = rownames(geo.matrix)[row(geo.matrix)],
  geo.dist = c(geo.matrix)
) # converting the genetic matrix into a df with columns describing which combos result in the distance

geo.df <- na.omit(geo.df) # removing NA's left from lower triangle

geo.df <- unite(geo.df, islands.combo, 1:2, sep = ":", remove = TRUE) # combining the first 2 columns (

pwd.df <- merge(pwd.df, geo.df, by = "islands.combo", all = FALSE) # merging distance between islands w

pwd.df$geo.dist <- pwd.df$geo.dist/1000 # going from metres to km

rm(longlat, geo.matrix, geo.df)
```

Creating results df on which to base analyses

```
LM <- lm(fst.dist ~ geo.dist, data = pwd.df) # model
summary(LM) # model results
par(mfrow = c(2, 2)) # changing the number of plots visible at once
plot(LM) # diagnostic plots

car::outlierTest(LM) # no. 148 most extreme
pwd.df[148,] # Normandy Is:Mainland, FST -0.33292

car::ncvTest(LM) # homoscedasticity test
# Chisquare = 12.31918, Df = 1, p = 0.00044833, H0 of constant variance is rejected

par(mfrow = c(1, 1))
plot(fst.dist ~ geo.dist, data = pwd.df)
abline(coef = coef(LM), col = 4, lwd = 2)
```

Linear modelling Results:

Call: lm(formula = fst.dist ~ geo.dist, data = pwd.df)

Residuals:

Min	1Q	Median	3Q	Max
-0.80573	-0.08944	0.00017	0.11193	0.37928

Coefficients:

	Estimate	Std. Error	t value	p-value
(Intercept)	3.531e-01	1.842e-02	19.172	< 2e-16 ***
geo.dist	2.096e-05	2.985e-06	7.021	1.5e-11 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1626 on 298 degrees of freedom
Multiple R-squared: 0.1419
Adjusted R-squared: 0.1391
F-statistic: 49.29 on 1 and 298 DF, p-value: 1.496e-11

- Diagnostic Plots: indications that the relationship is linear, normal distribution of residuals, down trending scale-location plot therefore non-constant variance, and although 148 is far apart it is not above cook's distance and not high leverage.

```
x <- pwd.df[pwd.df$fst.dist >= 0,]

LM2 <- lm(fst.dist ~ geo.dist, data = x) # model
summary(LM2) # model results
par(mfrow = c(2, 2)) # changing the number of plots visible at once
```

```
plot(LM2) # diagnostic plots: non-constant variance!

par(mfrow = c(1, 1))
plot(fst.dist ~ geo.dist, data = x)
abline(coef = coef(LM2), col = 4, lwd = 2)

car::ncvTest(LM2) # homoscedasticity test
# Chisquare = 16.78224, Df = 1, p = 0.000041924, H0 of constant variance is rejected
MASS::boxcox(LM2) # original values likely produce the best fit
```

Results:

Call: lm(formula = fst.dist ~ geo.dist, data = x)

Residuals:

Min	1Q	Median	3Q	Max
-0.40541	-0.09238	-0.00324	0.10726	0.37482

Coefficients:

	Estimate	Std. Error	t value	p-value
(Intercept)	3.573e-01	1.731e-02	20.641	< 2e-16 ***
geo.dist	2.104e-05	2.804e-06	7.506	7.22e-13 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1527 on 296 degrees of freedom
Multiple R-squared: 0.1599
Adjusted R-squared: 0.157
F-statistic: 56.33 on 1 and 296 DF, p-value: 7.222e-13

- Diagnostic Plots: indicates linear relationship, normal distribution of residuals, non-constant variance (more visible than LM 1), and nosignificant (or high leverage) outliers.

Plots

```
library(corrplot)
par(mfrow = c(1, 1))
corrplot(pwd,
         method = "color",
         type = "lower", # which triangle
         tl.col = "black", # text colour
         order = "FPC",
         diag = FALSE,
         p.mat = pv, # links to p-value matrix
         sig.level = 0.05,
         insig = "pch", # what to do with insignificant p-values
         col.lim = c(-0.4, 1),
         col = RColorBrewer::brewer.pal(n = 10, name = "Spectral") # colour palette
         )
mtext("Correlogram of FST values between Islands", at = 10, line = -5, cex = 1.3) # at = horizontal, li
```

From 0 to 1: 0 implying free interbreeding, 1 means the popn.s don't interbreed "Values for mammal populations between subspecies, or closely related species, typical values are of the order of 5% to 20%"

```
ggplot(data = x, aes(x = geo.dist, y = fst.dist)) +
  geom_point(shape = 1, colour = "black") +
  geom_smooth(method = "lm", se = TRUE) +
  ggtitle("Correlation between FST and Geographic Distance between Islands") +
  xlab("Geographic Distance between 2 islands (km)") + ylab("FST Distance between 2 islands") +
  theme_light()
```

The two negative values may be due to low sample size (could be outliers). FST is usually between 0 and 1. The negative values are also identified and non-significant.