FST Prep and Analysis Script

Grace Saville

16/05/2022

1. Prepping the dataframe

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

```
copy <- data # making a copy
t(copy[1,1:20]) # checking column names
```

```
##                     1
## island              "Borneo_002"
## registration.number "NBC.LAB.1968"
## genus               "Rattus"
## species             "exulans"
## sex                 "female"
## country             "Indonesia"
## state_province      "Kalimantan Timur"
## island.1            "Borneo"
## locality            "Badang, Sungai Kajan"
## site                ""
## geo_lat             "-0.5102"
## geo_long            "117.0912"
## collector           "Victor von Plessen"
## collecting.date     "1935"
## field.number        "AMNH.103837"
## Populatie           "1"
## X299_CHR1_114679736 "?"
## X13_CHR1_116614092  "?"
## X14_CHR1_124857905  "?"
## X15_CHR1_134869867  "T:T"
```

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

```
##                     1
## island              "Borneo_002"
## X299_CHR1_114679736 "?"
## X13_CHR1_116614092  "?"
## X14_CHR1_124857905  "?"
## X15_CHR1_134869867  "T:T"
## X16_CHR1_137314938  "A:A"
## X18_CHR1_185979552  "?"
## X19_CHR1_192708191  "C:C"
## X20_CHR1_198383739  "?"
## X21_CHR1_201964872  "?"
## X22_CHR1_209547552  "A:A"
## X23_CHR1_211919559  "?"
## X262_CHR1_212322960 "C:C"
## X24_CHR1_216451585  "G:G"
## X25_CHR1_220057345  "T:T"
## X26_CHR1_231126749  "A:A"
## X27_CHR1_255622475  "?"
## X300_CHR1_262011841 "A:A"
## X301_CHR1_262011844 "C:C"
## X28_CHR1_265508390  "G:G"
```

```
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
```

```
## [1] 370 283
```

```
for (i in 2:283) {
  y <- reshape::colsplit(
          copy[, i], split = ":", names = c(coln[i], paste("blank", i, sep = ".")))
          # splitting each i column and renaming them
  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
```

2. Producing the file necessary for PGDSpider program

```
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"
```

```r
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 because
    # Luzon is part of the mainland cluster in the NeighborNet network)
    "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
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
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 only)
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
```

At this stage PGDSpider program and Arlequin were used to convert the file produced and run tests on the data. The resulting output is used here for analysis.

3. Loading the results files for 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 tri)
x <- t(pv) # transposed copy for p-values
pv[upper.tri(pv, diag = FALSE)] <- x[upper.tri(x, diag = FALSE)]

rm(x)
```

4. Making Geopgraphic distance matrix

```
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
kable(longlat) # checking
```

island.1	geo_lat	geo_long
Aotea (Great Barrier I)	-36.23000	175.4300
Borneo	-0.51020	117.0912
Doubtful Sound	-45.31667	166.9833
Great Mercury Island	-36.58333	175.9167

island.1	geo_lat	geo_long
Halmahera	1.26600	127.8565
Hatutaa	-7.92000	-140.5700
Honuea	-17.00900	-149.5850
Kaikura Island	-36.18000	175.3200
Kamaka	-23.24000	-134.6300
Kayangel	8.07000	134.7000
Late Island	-18.85000	-174.6000
Luzon	15.43469	120.4959
Mainland	15.20989	105.7906
Malenge	-0.26590	122.0439
Mohotani	-10.00000	-138.9300
Motukawanui	-35.00000	173.9400
New Britain	-5.81450	150.0610
New Guinea	-6.22080	147.3689
Normanby Island	-10.05460	150.9625
Rakiura (Stewart Isl)	-46.95000	167.9000
Reiono	-17.04600	-149.5460
Rimatuu (Tetiaroa)	-17.03000	-149.5580
Slipper Island	-37.05000	175.9300
Sulawesi	-1.32520	120.1039
Tahanea	-16.87000	-144.9700
Wake Island	19.30000	166.5800

```r
# 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"
longlat <- longlat[-12,] # removing luzon since fst has it with mainland
row.names(longlat) <- seq(nrow(longlat)) # renaming row numbers to be sequential

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

geo.matrix <- distm(geo.matrix, fun = distGeo) # converting to pairwise distance matrix
dim(geo.matrix) # 25 25
```

```
## [1] 25 25
```

5. FST Mantel test

```
dim(geo.matrix) # 25 25
```

```
## [1] 25 25
```

```
dim(pwd) # 25 25
```

```
## [1] 25 25
```

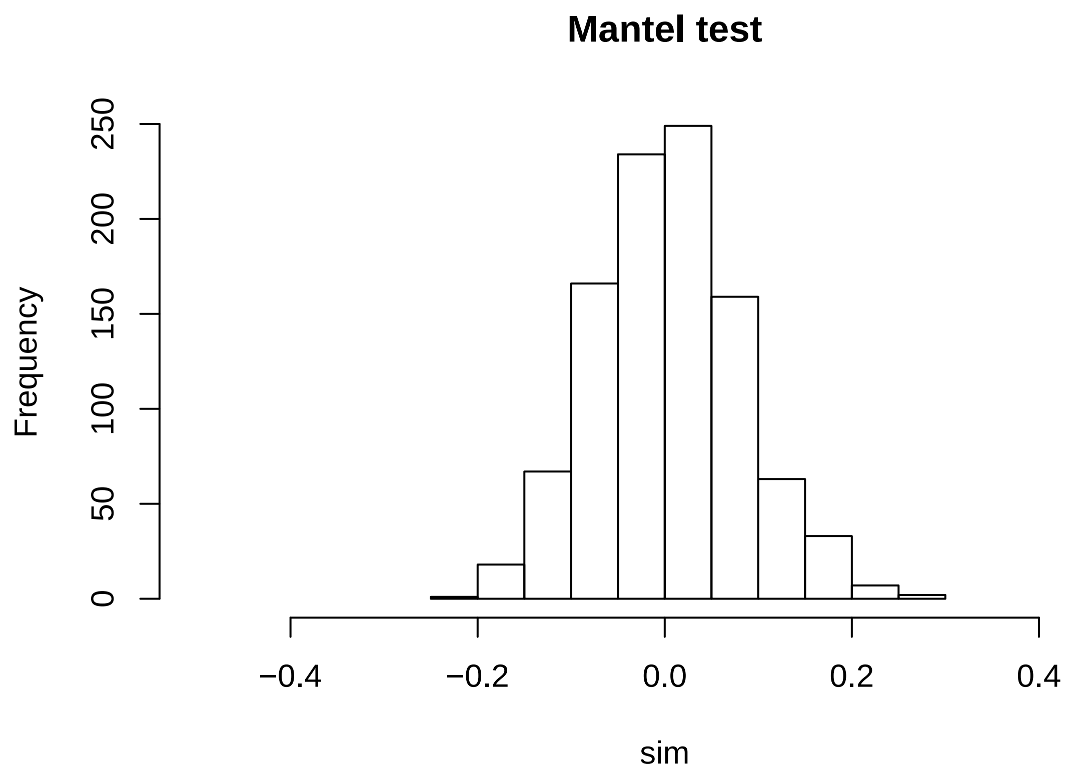
```
geo.dist <- as.dist(geo.matrix, diag = TRUE, upper = TRUE) # converting to dist object
fst.dist <- as.dist(pwd, diag = TRUE, upper = TRUE)

set.seed(4)
r1 <- mantel.rtest(fst.dist, geo.dist, nrepet = 999)
```

```
## Warning in is.euclid(m1): Zero distance(s)
```

```
r1
```

```
## Monte-Carlo test
## Call: mantelnoneuclid(m1 = m1, m2 = m2, nrepet = nrepet)
##
## Observation: 0.376738
##
## Based on 999 replicates
## Simulated p-value: 0.001
## Alternative hypothesis: greater
##
##     Std.Obs Expectation    Variance
## 4.747032343 0.003179756 0.006192589
```

```
plot(r1$plot$hist, main = "Mantel test", xlim = c(-0.5, 0.5))
```



```r
# abline(v = )
```

6. Creating results dataframe on which to base analyses

6a. Converting FST matrix to dataframe

```r
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 (the names of the matrices columns and rows) to give a label
# to each pairwise distance
```

6b. Combining the FST and Geographic dataframes

```
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 (the names of the matrices columns and rows) to give a label
# to each pairwise distance

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

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

rm(geo.df)
```

6c. Saving outcomes

```
write.csv(pwd.df, "./Results/Arlequin_FST/FST_RStudio_outcomes_df.csv", row.names = FALSE)
write.csv(pv, "./Results/Arlequin_FST/FST_RStudio_pvalue_matrix.csv", row.names = FALSE)
```

7. Linear modelling

```
pwd.df <- read.csv("./Results/Arlequin_FST/FST_RStudio_outcomes_df.csv")
```

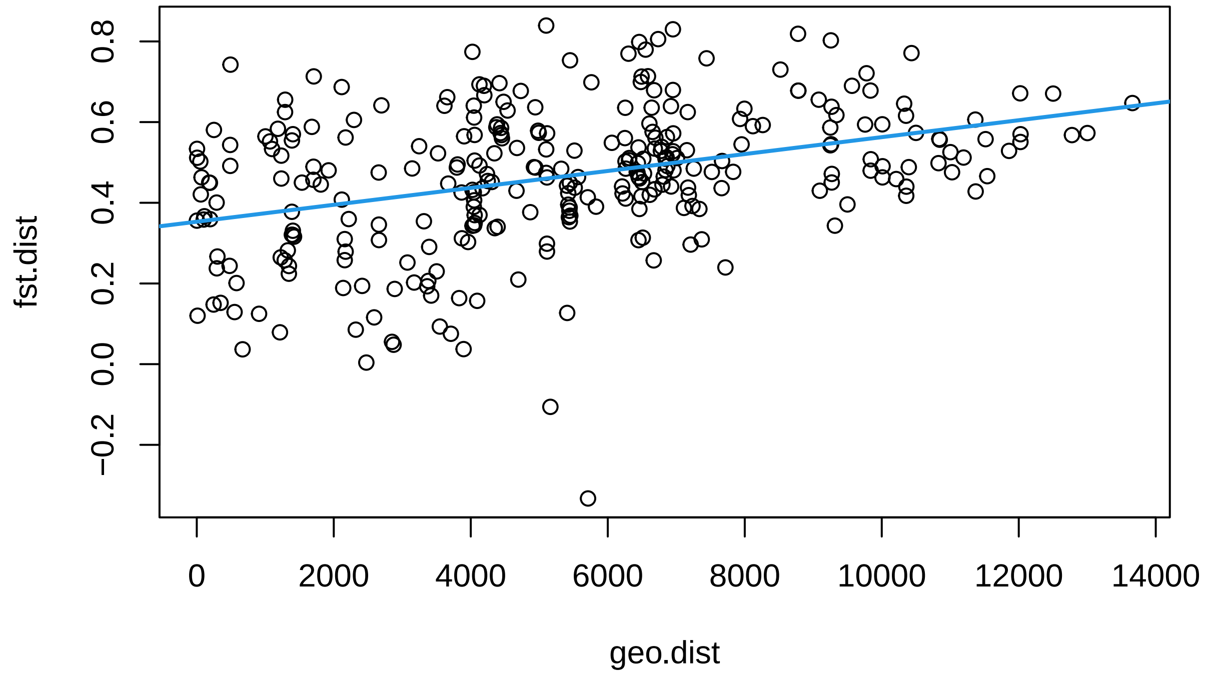
7a. Test model

```
testLM <- lm(fst.dist ~ geo.dist, data = pwd.df) # model
summary(testLM) # model results
```

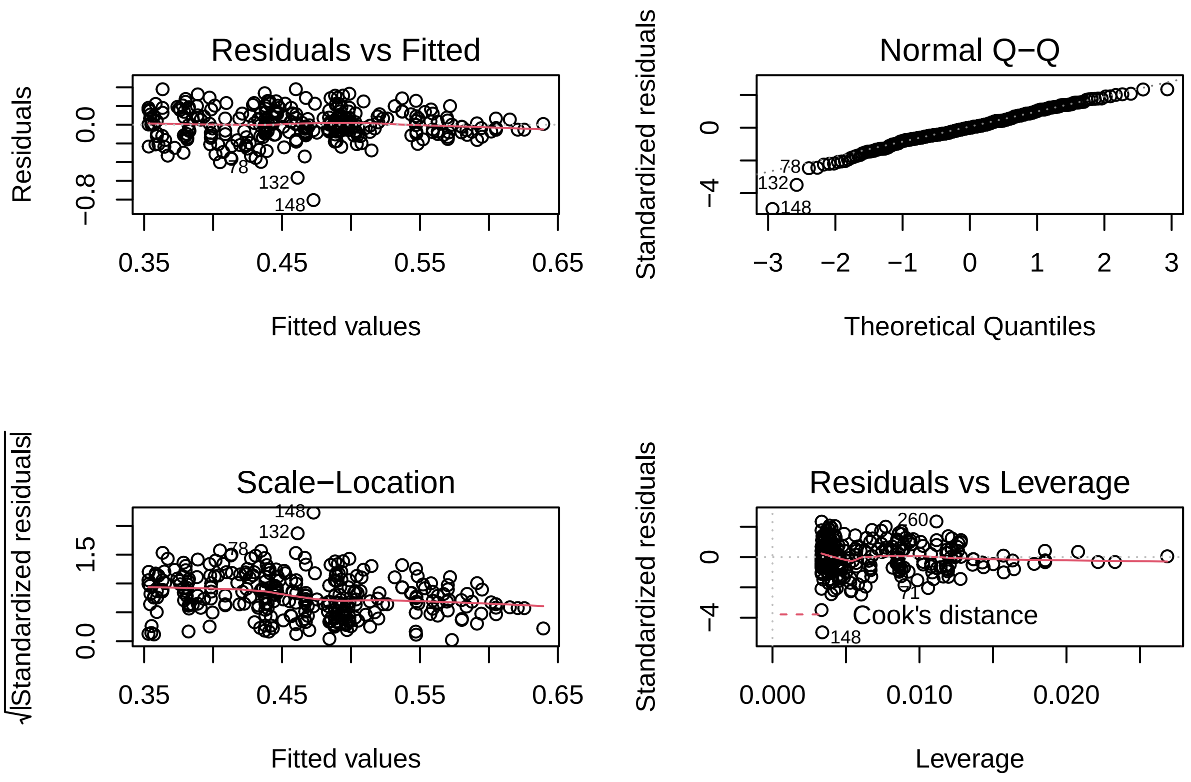
```
##
## Call:
## lm(formula = fst.dist ~ geo.dist, data = pwd.df)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.80573 -0.08944  0.00017  0.11193  0.37928
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (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
```

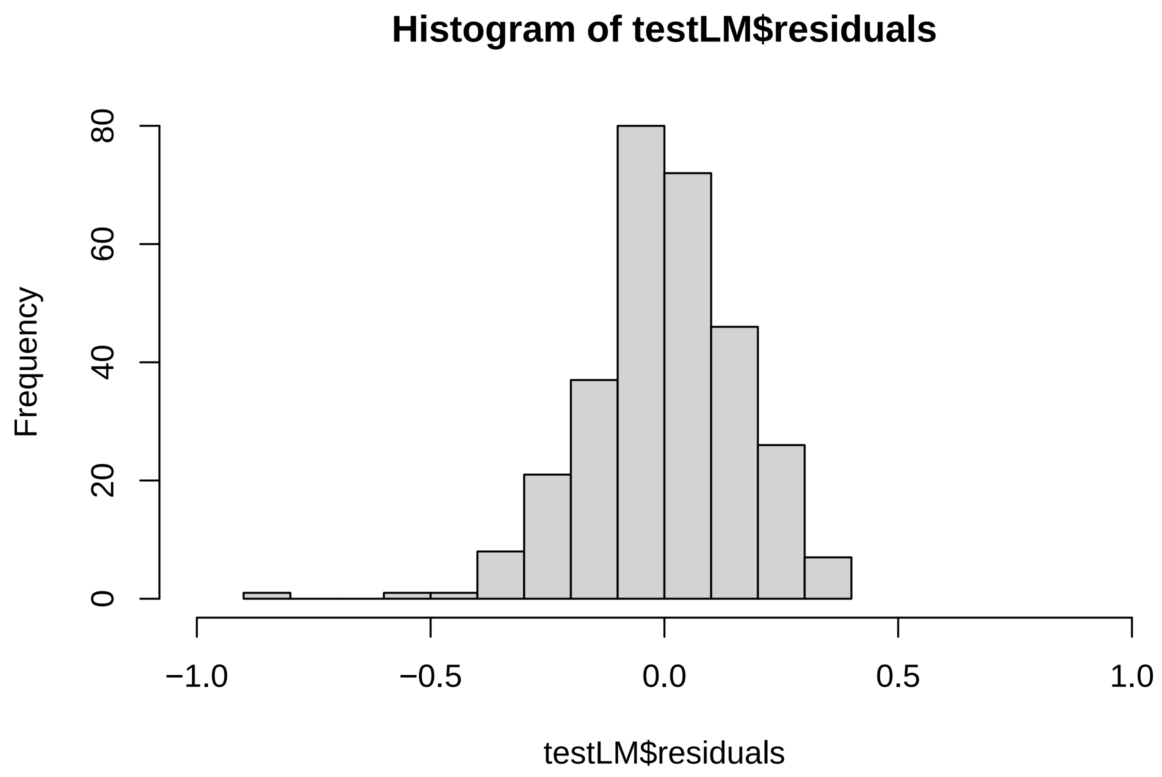
```
plot(fst.dist ~ geo.dist, data = pwd.df)
abline(coef = coef(testLM), col = 4, lwd = 2)
```



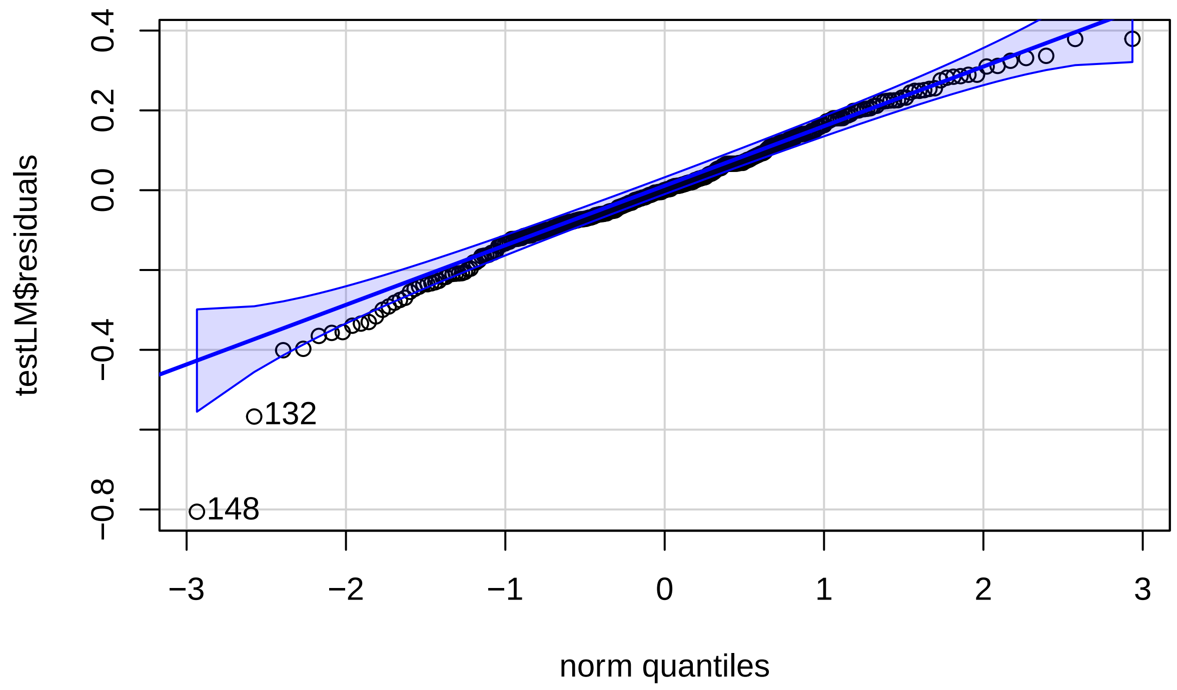
```
par(mfrow = c(2, 2)) # changing the number of plots visible at once
plot(testLM) # diagnostic plots
```



```
par(mfrow = c(1,1))
hist(testLM$residuals, breaks = 10, xlim = c(-1,1))
```



```r
qqPlot(testLM$residuals, line = "quartiles") # normal, possible outliers 132, 148
```



```
## [1] 148 132
```

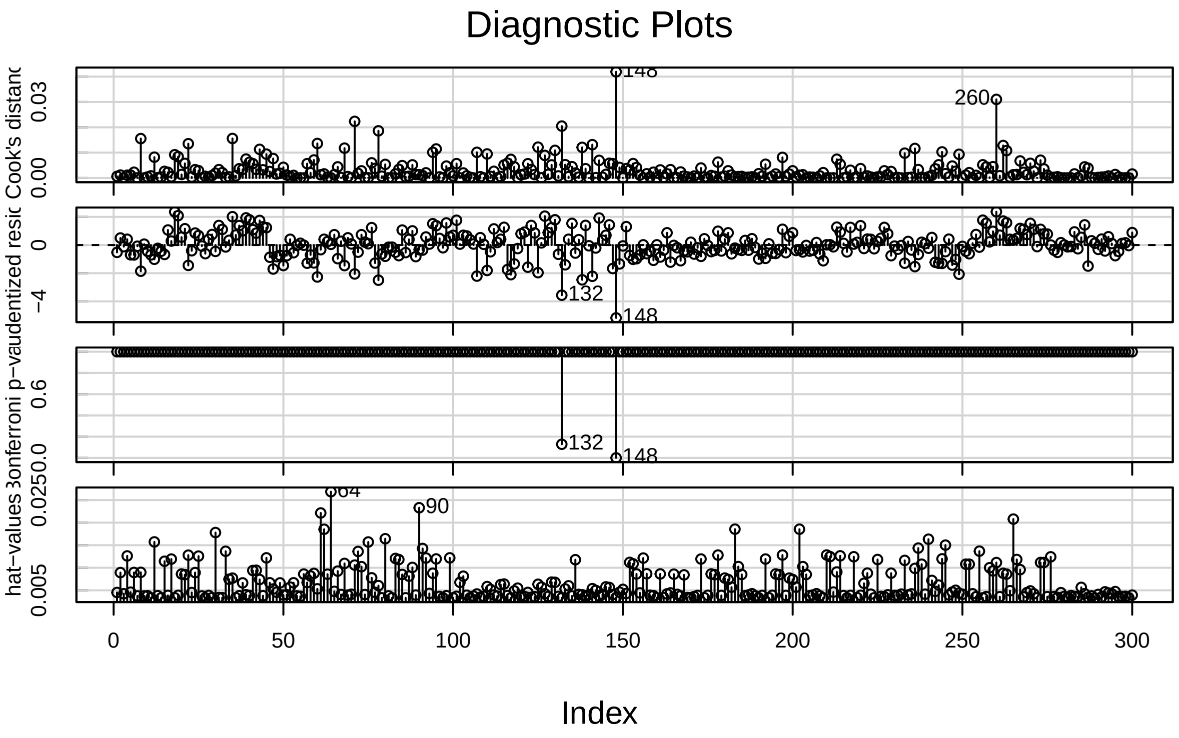
```
shapiro.test(testLM$residuals) # indicates non-normality of residuals but
```

```
##
##  Shapiro-Wilk normality test
##
## data:  testLM$residuals
## W = 0.97774, p-value = 0.0001291
```

```
# likely affected by the 2 outliers mentioned above

ncvTest(testLM) # homoscedasticity test: H0 of constant variance is rejected.
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 12.31918, Df = 1, p = 0.00044833
```

```
# Supported by downward slope in Scale-location plot (plot(testLM))

influenceIndexPlot(testLM) # outliers
```



```
# Cooks distances: none larger than 0.5,
# Studentised residuals: 132 and 148 less than -3
# Bonferroni p-value: 132 and 148 smaller then 0.05,
# Hat-values: none influential, higher than 1
outlierTest(testLM)
```

```
##        rstudent unadjusted p-value Bonferroni p
## 148 -5.174788         4.2083e-07   0.00012625
```

- Diagnostic Plots: indications that the relationship is linear, normal distribution of residuals, down trending scale-location plot and cone-shaped residuals vs. fitted plot therefore non-constant variance, and 132 (New Guinea to Mainland) and 148 (Normanby Island to Mainland) are potential issues.

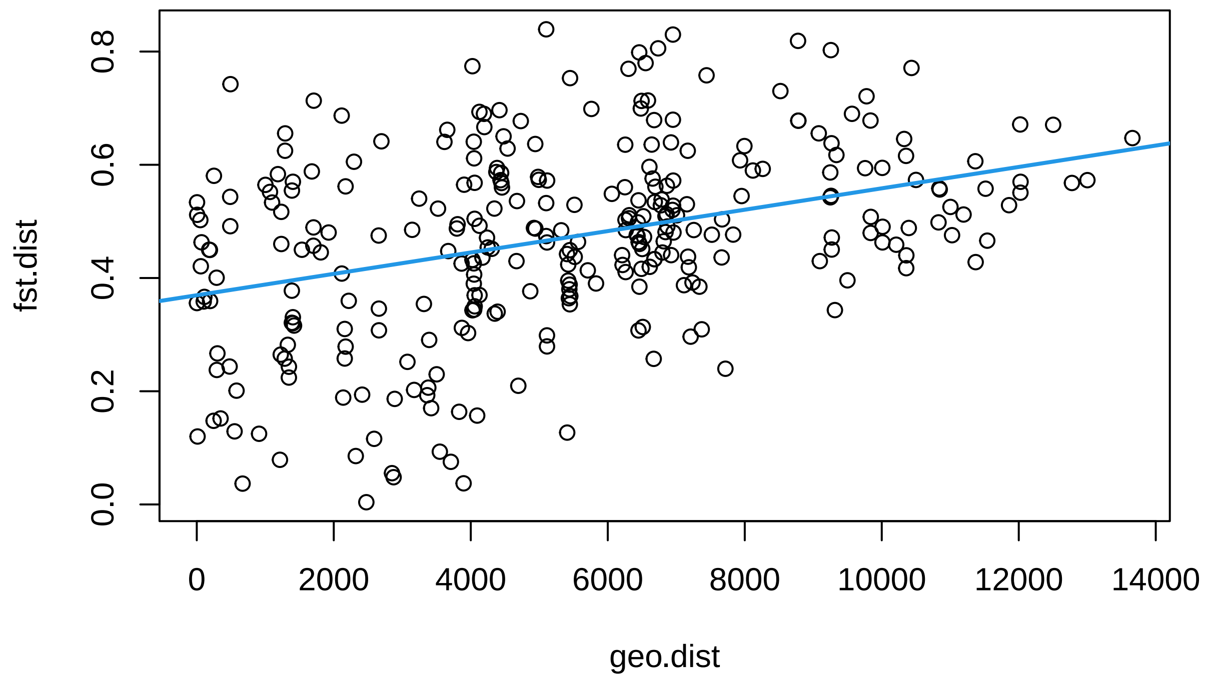
7b. Adjusted Model

```
pwd.df2 <- pwd.df[pwd.df$fst.dist >= 0,] # removing 132 (New_Guinea:Mainland) and
# 148 (Normanby_Island:Mainland) which are both negative FST values

x <- lm(fst.dist ~ geo.dist, data = pwd.df2) # creating model to take weights from
wt <- 1 / lm(abs(x$residuals) ~ x$fitted.values)$fitted.values^2 # weighting
# residuals by how large they are
rm(x)
```

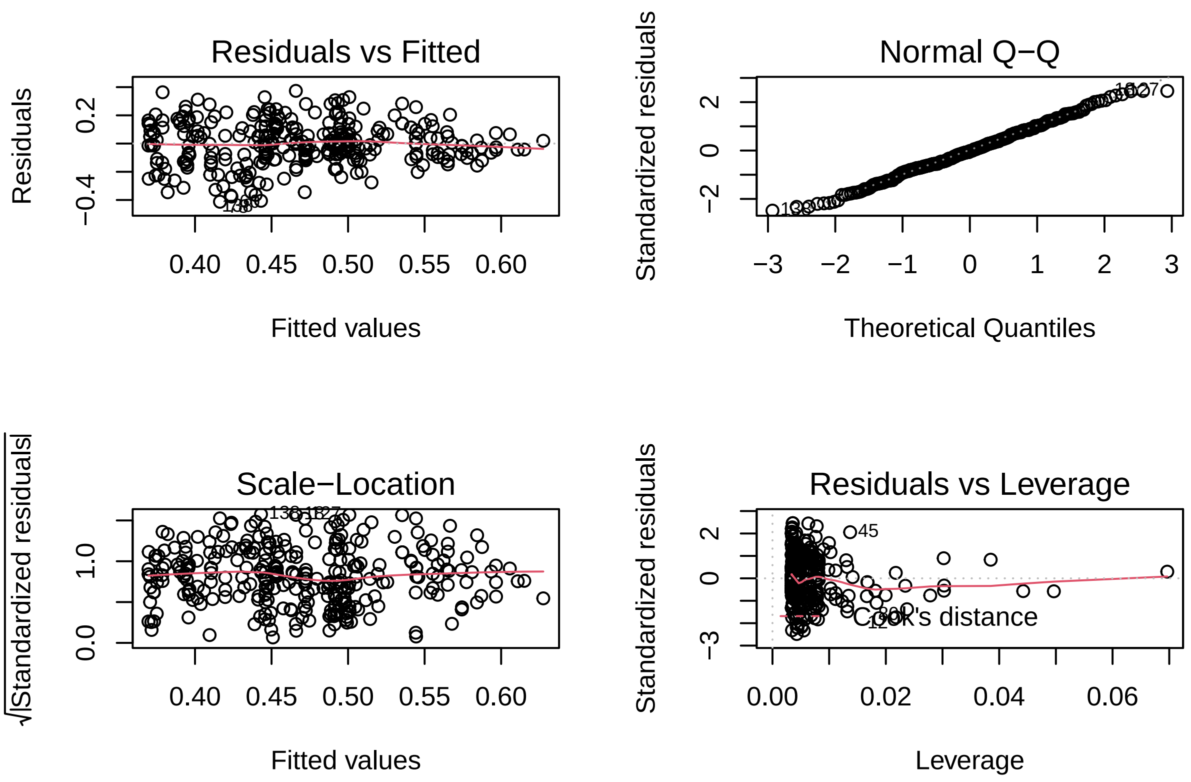
```r
LM <- lm(fst.dist ~ geo.dist, data = pwd.df2, weights = wt) # weighted residual model
summary(LM) # model results
```

```
##
## Call:
## lm(formula = fst.dist ~ geo.dist, data = pwd.df2, weights = wt)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -3.08685 -0.81585 -0.02973  0.88374  3.06125
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.696e-01  1.825e-02  20.251  < 2e-16 ***
## geo.dist    1.889e-05  2.386e-06   7.916 4.96e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.246 on 296 degrees of freedom
## Multiple R-squared:  0.1747, Adjusted R-squared:  0.1719
## F-statistic: 62.66 on 1 and 296 DF,  p-value: 4.959e-14
```

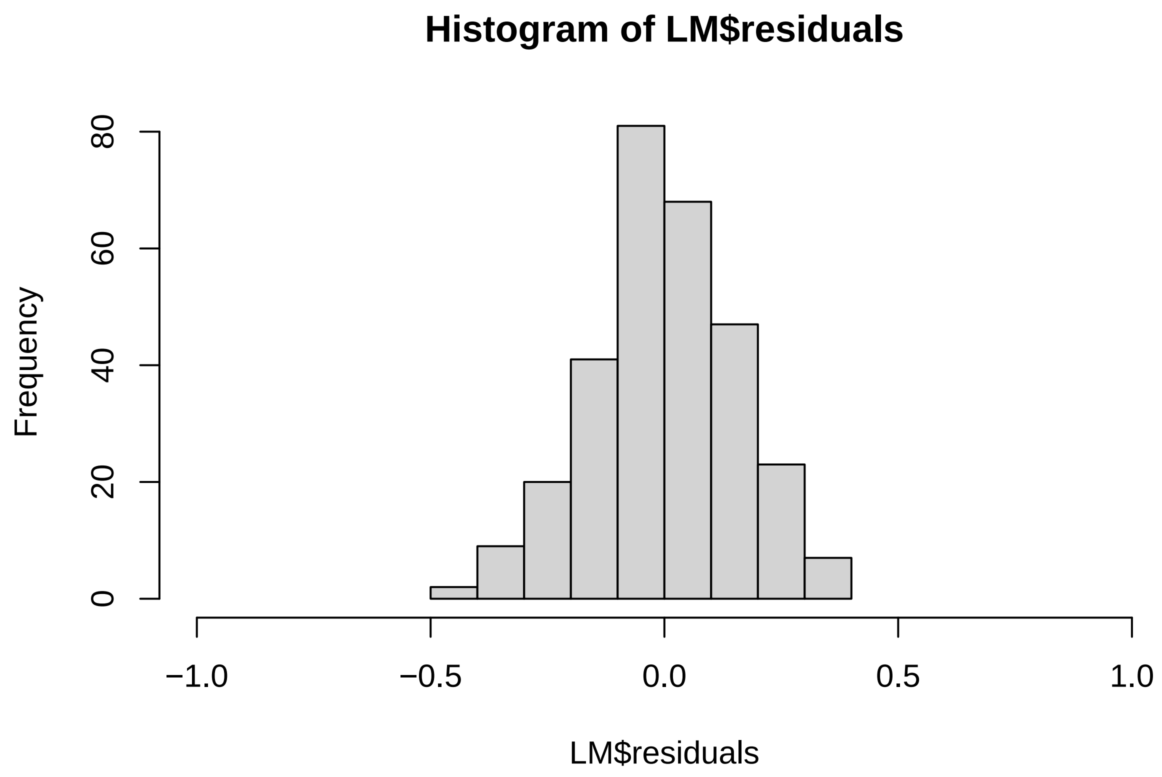
```r
plot(fst.dist ~ geo.dist, data = pwd.df2)
abline(coef = coef(LM), col = 4, lwd = 2)
```



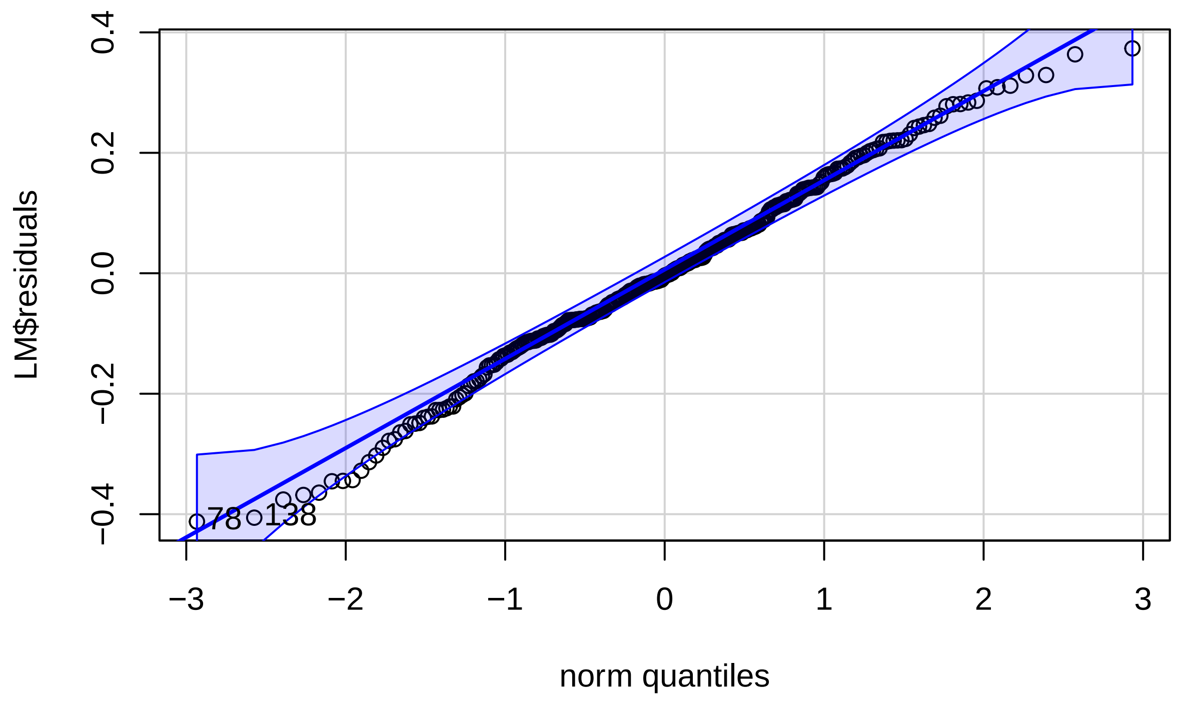
```
par(mfrow = c(2, 2)) # changing the number of plots visible at once
plot(LM) # diagnostic plots: non-constant variance!
```



```
par(mfrow = c(1, 1))
hist(LM$residuals, breaks = 10, xlim = c(-1,1))
```



```r
qqPlot(LM$residuals, line = "quartiles") # normal
```



```
##  78 138
##  78 137
```

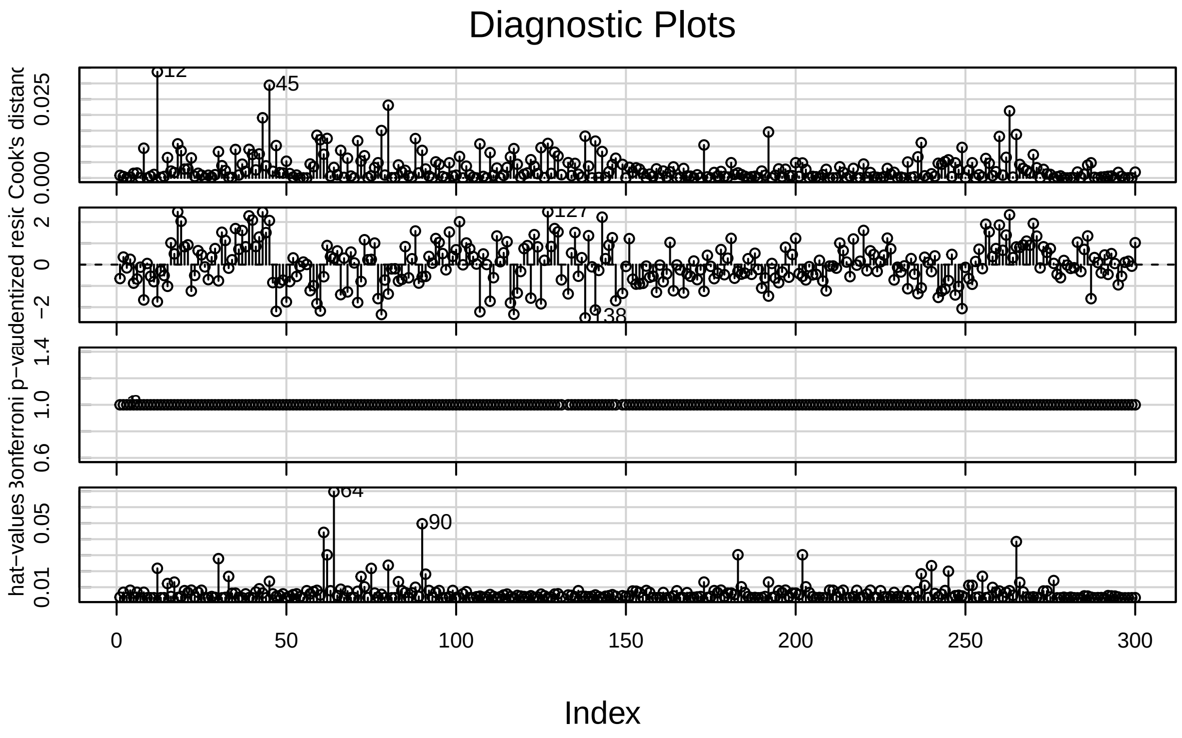
```
shapiro.test(LM$residuals) # indicates normality of residuals
```

```
##
##  Shapiro-Wilk normality test
##
## data:  LM$residuals
## W = 0.99402, p-value = 0.2898
```

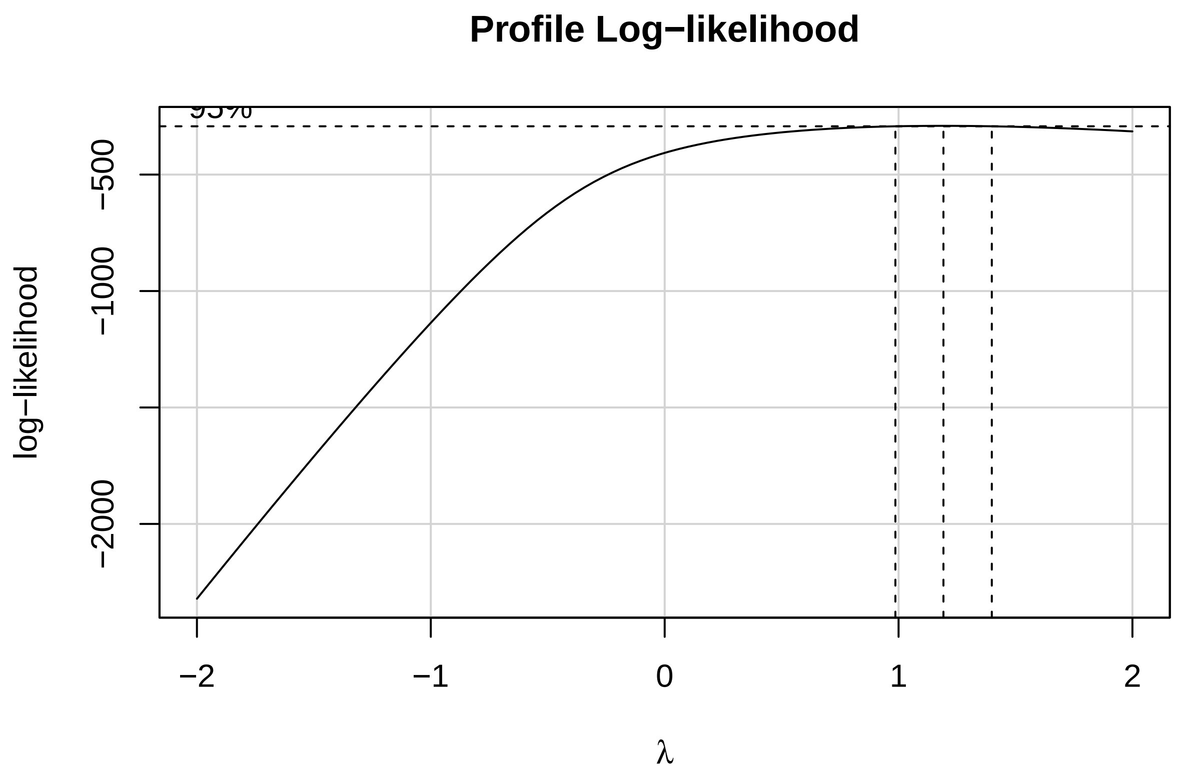
```
ncvTest(LM) # homoscedasticity test: H0 of constant variance is not rejected.
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 2.234525e-07, Df = 1, p = 0.99962
```

```
# Supported by flat-ish slope in Scale-location plot (plot(testLM))

influenceIndexPlot(LM) # outliers test, nothing concerning
```



```
boxCox(LM) # recommended to stay as is
```



- Even spread of residuals around 0
- t-values are far from 1 and both are significant
- Residual standard error is very high compared to the estimate
- Only approx. 17% of the variance of fst.dist can de explained by geo.dist!

8. Examining values below the regression line

```
mean(pwd.df$fst.dist)
```

```
## [1] 0.4643348
```

```
median(pwd.df$fst.dist)
```

```
## [1] 0.480365
```

```
range(pwd.df$fst.dist)
```

```
## [1] -0.33292  0.83929
```

```r
# without new guinea and norm. isl.:
mean(pwd.df2$fst.dist)
```

```
## [1] 0.4689237
```

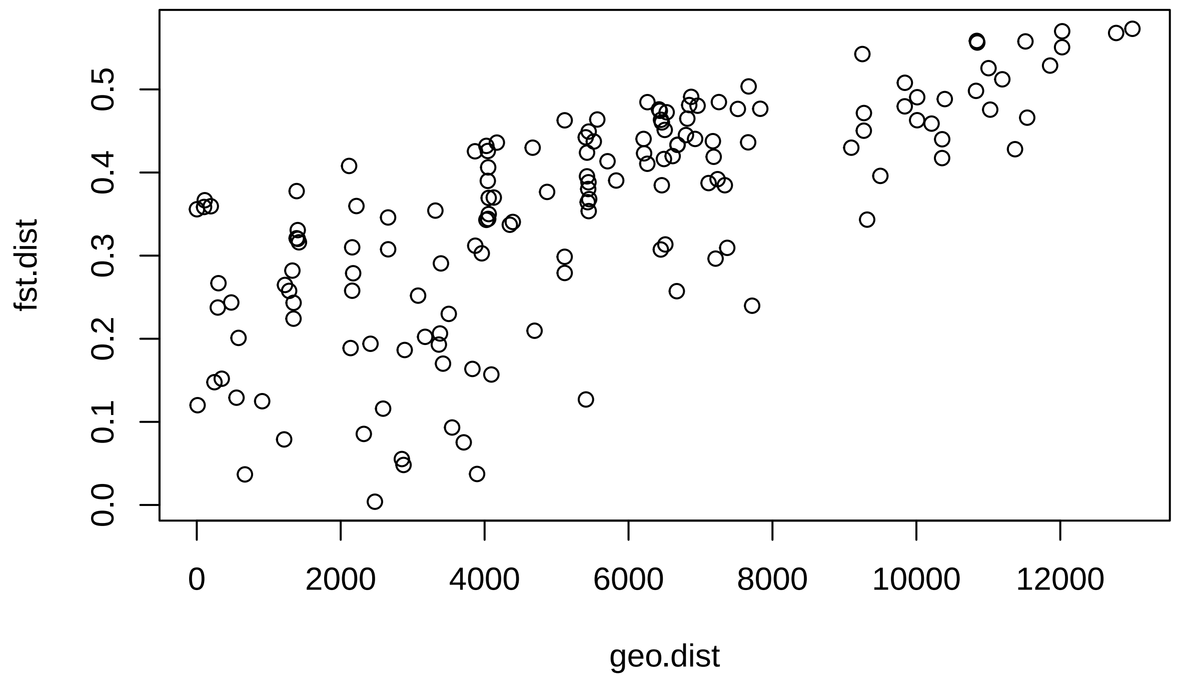
```r
median(pwd.df2$fst.dist)
```

```
## [1] 0.48077
```

```r
range(pwd.df2$fst.dist)
```

```
## [1] 0.00394 0.83929
```

```r
fv <- as.vector(LM$fitted.values)
under.fv <- cbind(pwd.df2, fv)

under.fv <- under.fv[under.fv$fst.dist < under.fv$fv,] # keeping only fst.dist
# values less than fitted values

plot(fst.dist ~ geo.dist, data = under.fv)
```



```
# should be all values up to the regression line
```

```
under.fv <- under.fv[order(under.fv$geo.dist, decreasing = FALSE),] # sorting by geo.dist
under.fv <- under.fv[order(under.fv$fst.dist, decreasing = FALSE),] # sorting by fst.dist
under.fv <- under.fv[order(under.fv$islands.combo, decreasing = FALSE),] # sorting alphabetically
# kable(under.fv)

rownames(under.fv) <- seq(1:nrow(under.fv))
under.fv.copy <- tidyr::separate(under.fv, sep = ":", col = islands.combo, into = c("isl1", "isl2"))
```

```
# this chunk is dedicated to creating a df like this for a plot with connecting points:
# island  lat long  group
# aotea   123 321   aotea:borneo
# borneo  345 543   aotea:borneo
# aotea   123 321   aotea:kaikura
# kaikura 567 765   aotea:kaikura

under.fv.copy$islands.combo <- under.fv$islands.combo # adding island.combo to define where the line on
under.fv.copy <- rbind(under.fv.copy, under.fv.copy) # making 2 of each island combo
nrow(under.fv.copy)
```

```
## [1] 306
```

```
under.fv.copy[153:306,1] <- under.fv.copy[153:306,2] # doing my own merge of the first 2 columns
under.fv.copy <- under.fv.copy[,-2] # now removing the 2nd column now that I fixed the 1st column
```

```
head(longlat) # checking correct columns are used
```

```
##                island.1   geo_lat   geo_long
## 1                 Aotea -36.23000  175.4300
## 2                Borneo  -0.51020  117.0912
## 3         Doubtful_Sound -45.31667  166.9833
## 4 Great_Mercury_Island -36.58333  175.9167
## 5             Halmahera   1.26600  127.8565
## 6               Hatutaa  -7.92000 -140.5700
```

```
dim(longlat)
```

```
## [1] 25  3
```

```
longlat[longlat$geo_long < 0,] # only longitude values up to 180 are plotting, rest
```

```
##       island.1 geo_lat geo_long
## 6      Hatutaa  -7.920 -140.570
## 7       Honuea -17.009 -149.585
## 9       Kamaka -23.240 -134.630
## 11 Late_Island -18.850 -174.600
## 14    Mohotani -10.000 -138.930
## 20      Reiono -17.046 -149.546
## 21     Rimatuu -17.030 -149.558
## 24     Tahanea -16.870 -144.970
```

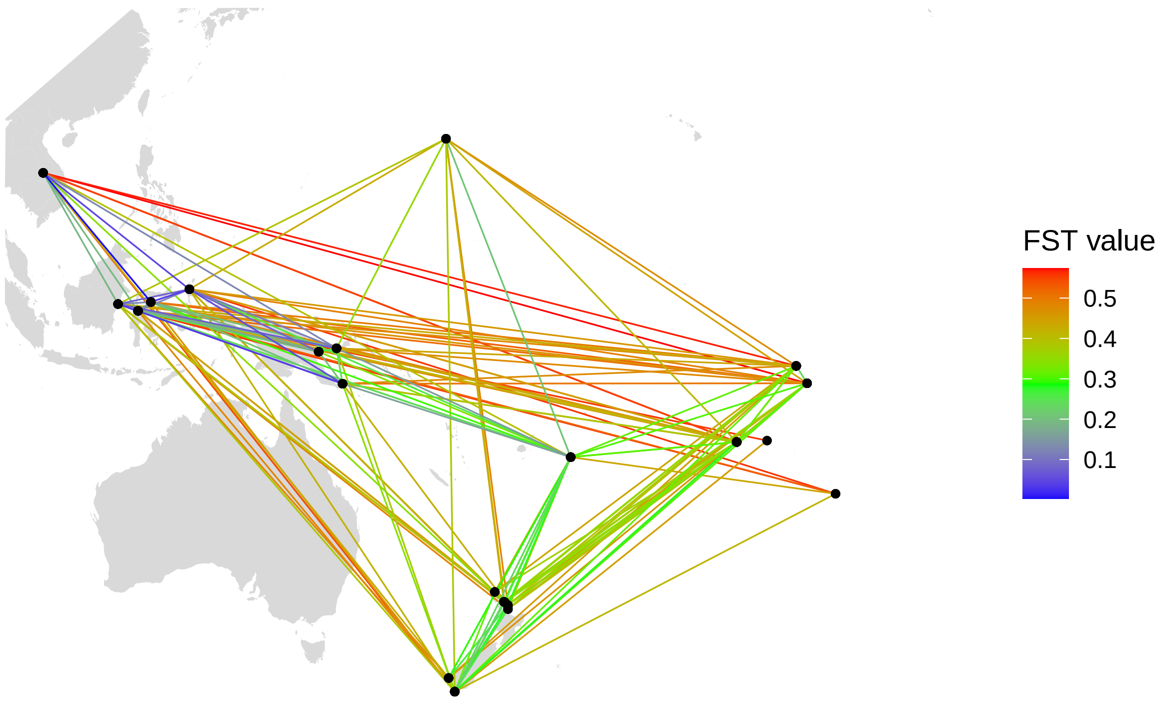
```r
# is blank. Can try fixing this by adding 360 to all the negative longitude points
longlat[6,3] <- longlat[6,3] + 360
longlat[7,3] <- longlat[7,3] + 360
longlat[9,3] <- longlat[9,3] + 360
longlat[11,3] <- longlat[11,3] + 360
longlat[14,3] <- longlat[14,3] + 360
longlat[20,3] <- longlat[20,3] + 360
longlat[21,3] <- longlat[21,3] + 360
longlat[24,3] <- longlat[24,3] + 360

# adding lat and long to df
under.fv.copy <- merge(under.fv.copy, longlat, by.x = "isl1", by.y = "island.1")
# sorting by fst
under.fv.copy <- under.fv.copy[order(under.fv.copy$fst.dist, decreasing = TRUE),]

rownames(under.fv.copy) <- seq(1:nrow(under.fv.copy))
# z$islands.combo
under.fv.copy <- dplyr::mutate(under.fv.copy, group.id = match(islands.combo, unique(islands.combo)))
```

```r
# library(ggrepel)
range(under.fv.copy$fst.dist) # checking what the midpoint would be for coloured legend
```

```
## [1] 0.00394 0.57288
```

```r
ggplot(under.fv.copy, aes(x = geo_long, y = geo_lat, group = group.id, colour = fst.dist)) +
  borders("world2", colour = NA, fill = "grey85")  +
  geom_path(size = 0.3) +
  scale_color_gradient2(
    low = "blue",
    mid = "green",
    high = "red",
    midpoint = 0.28644,
    name = "FST value"
  ) +
  geom_point(color = "black", size = 1) +
  scale_x_continuous(limits = c(100, 240)) +
  scale_y_continuous(limits = c(-50, 35)) +
  theme(panel.background = element_rect(fill = "white", colour = "white"),
        axis.text = element_blank(),
        axis.title = element_blank(),
        axis.ticks = element_blank()) +
  # geom_text_repel(size = 3, point.size = 1, min.segment.length = 0.25) +
  ggtitle(
    "Map of the Island Pairs with FST Values lower than expected given Geographic Distance")
```

Map of the Island Pairs with FST Values lower than expected given Geographic I



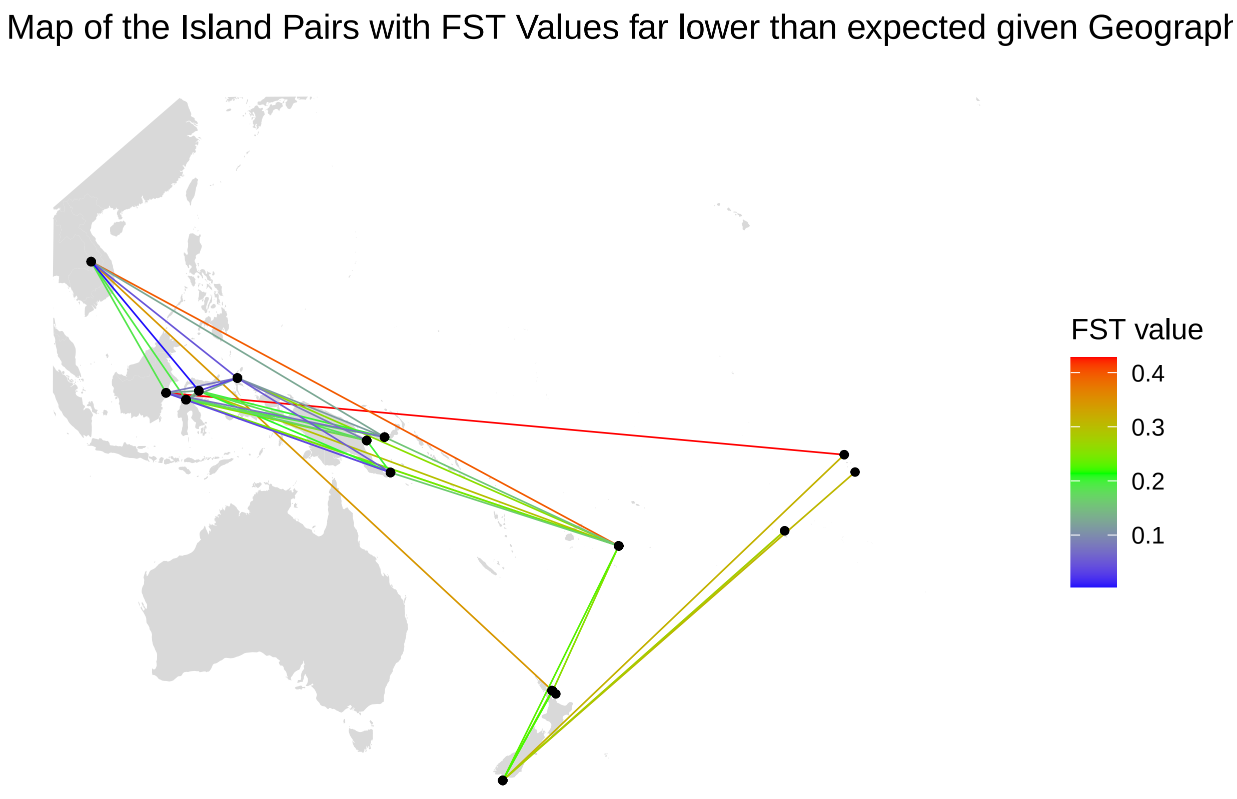
```
# map of lines that are more than 0.15 lower than the regression line
x <- under.fv[under.fv$fst.dist < under.fv$fv - 0.15,] # making df with lowest fst values
foo <- x$islands.combo
x <- tidyr::separate(x, sep = ":", col = islands.combo, into = c("isl1", "isl2"))
x$islands.combo <- foo # adding island.combo to define where the line on the plot needs to go between (
rm(foo)
x <- rbind(x, x) # making 2 of each island combo
rownames(x) <- seq(1:nrow(x))
nrow(x)
```

```
## [1] 86
```

```
x[43:86,1] <- x[43:86,2] # doing my own merge of the first 2 columns
x <- x[,-2] # now removing the 2nd column now that I fixed the 1st column
x <- merge(x, longlat, by.x = "isl1", by.y = "island.1") # adding lat and long to df
x <- x[order(x$fst.dist, decreasing = TRUE),] # sorting by fst
rownames(x) <- seq(1:nrow(x))
# x$islands.combo
x <- dplyr::mutate(x, group.id = match(islands.combo, unique(islands.combo))) # making
# group id number column

range(x$fst.dist)
```

```
## [1] 0.00394 0.42805
```

```
ggplot(x, aes(x = geo_long, y = geo_lat, group = group.id, colour = fst.dist)) +
  borders("world2", colour = NA, fill = "grey85")  +
  geom_path(size = 0.3) +
  scale_color_gradient2(
    low = "blue",
    mid = "green",
    high = "red",
    midpoint = 0.214025,
    name = "FST value"
  ) +
  geom_point(color = "black", size = 1) +
  scale_x_continuous(limits = c(100, 240)) +
  scale_y_continuous(limits = c(-50, 35)) +
  theme(panel.background = element_rect(fill = "white", colour = "white"),
        axis.text = element_blank(),
        axis.title = element_blank(),
        axis.ticks = element_blank()) +
  # geom_text_repel(size = 3, point.size = 1, min.segment.length = 0.25) +
  ggtitle(
    "Map of the Island Pairs with FST Values far lower than expected given Geographic Distance")
```


Map of the Island Pairs with FST Values far lower than expected given Geograph
FST value
0.4
0.3
0.2
0.1

```
# map of all connecting points
foo <- pwd.df2$islands.combo
y <- tidyr::separate(pwd.df2, sep = ":", col = islands.combo, into = c("isl1", "isl2"))
y$islands.combo <- foo # adding island.combo to define where the line on the plot
```

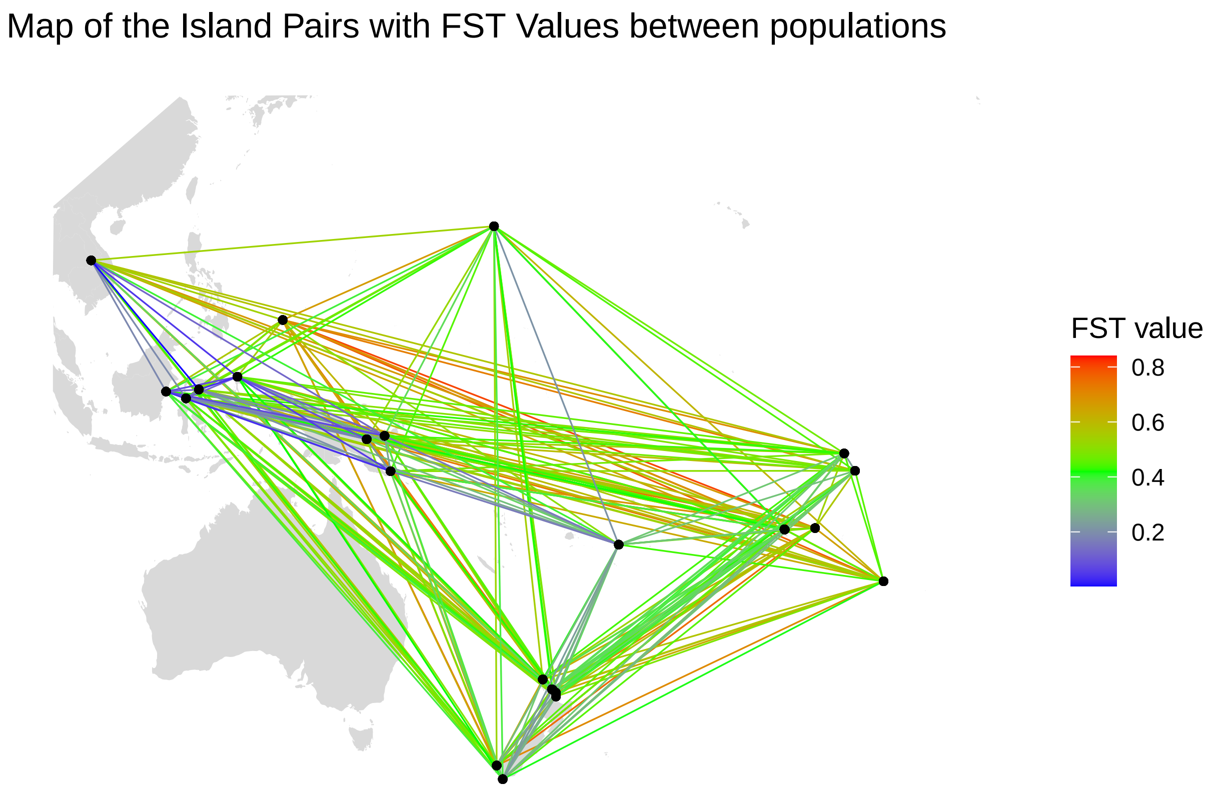
```r
# needs to go between (the "group")
rm(foo)
y <- rbind(y, y) # making 2 of each island combo
rownames(y) <- seq(1:nrow(y))
nrow(y)
```

```
## [1] 596
```

```r
y[298:596,1] <- y[298:596,2] # doing my own merge of the first 2 columns
y <- y[,-2] # now removing the 2nd column now that I fixed the 1st column
y <- merge(y, longlat, by.x = "isl1", by.y = "island.1") # adding lat and long to df
y <- y[order(y$fst.dist, decreasing = TRUE),] # sorting by fst
rownames(y) <- seq(1:nrow(y))
# x$islands.combo
y <- dplyr::mutate(y, group.id = match(islands.combo, unique(islands.combo))) # making
# group id number column

range(y$fst.dist)
```

```
## [1] 0.00394 0.83929
```

```r
ggplot(y, aes(x = geo_long, y = geo_lat, group = group.id, colour = fst.dist)) +
  borders("world2", colour = NA, fill = "grey85")  +
  geom_path(size = 0.3) +
  scale_color_gradient2(
    low = "blue",
    mid = "green",
    high = "red",
    midpoint = 0.419645,
    name = "FST value"
  ) +
  geom_point(color = "black", size = 1) +
  scale_x_continuous(limits = c(100, 240)) +
  scale_y_continuous(limits = c(-50, 35)) +
  theme(panel.background = element_rect(fill = "white", colour = "white"),
        axis.text = element_blank(),
        axis.title = element_blank(),
        axis.ticks = element_blank()) +
  ggtitle(
    "Map of the Island Pairs with FST Values between populations")
```



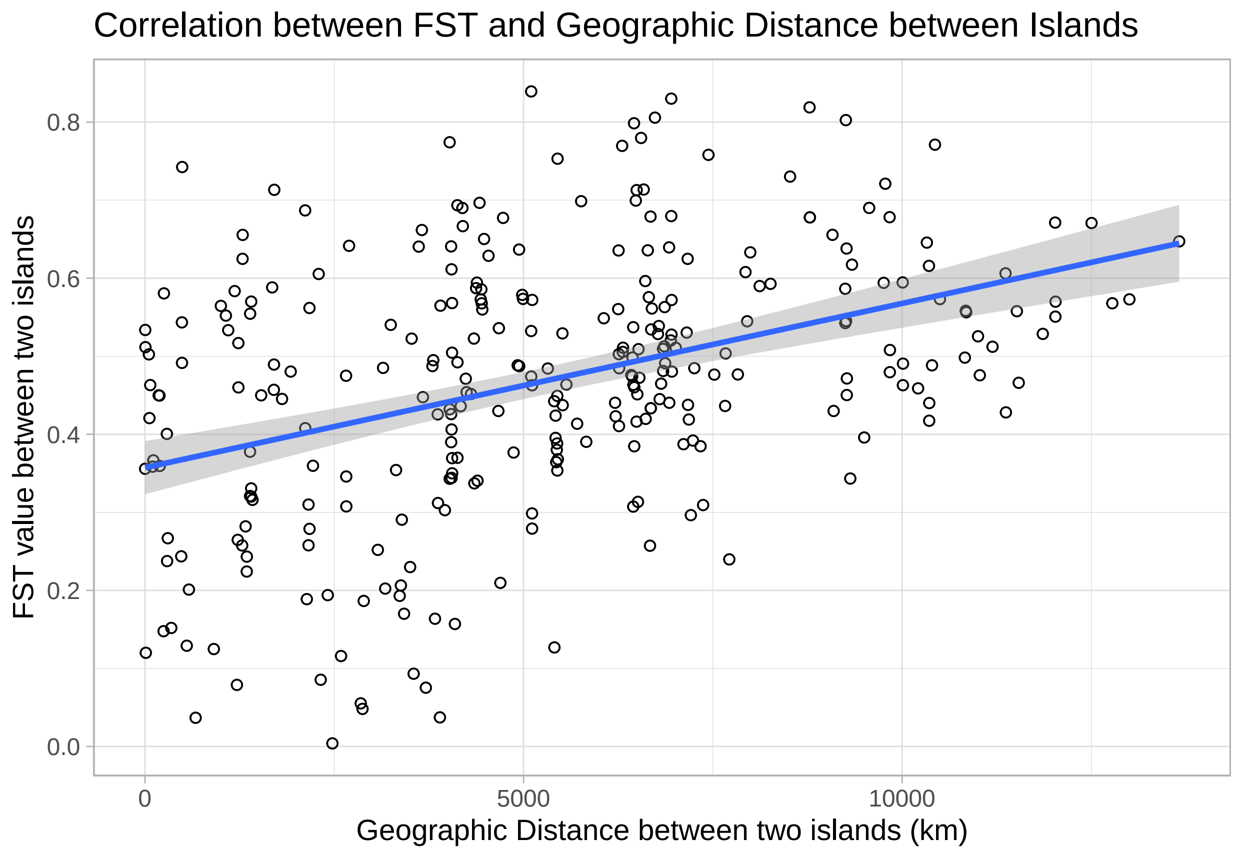
9. Plots

```r
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, line = height, cex = size

# this plot is best seen in the viewer rather than in the knitted pdf.
# Has also been saved as a separate file.
```

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%"

```r
ggplot(data = pwd.df2, 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 two islands (km)") +
  ylab("FST value between two islands") +
  theme_light()
```

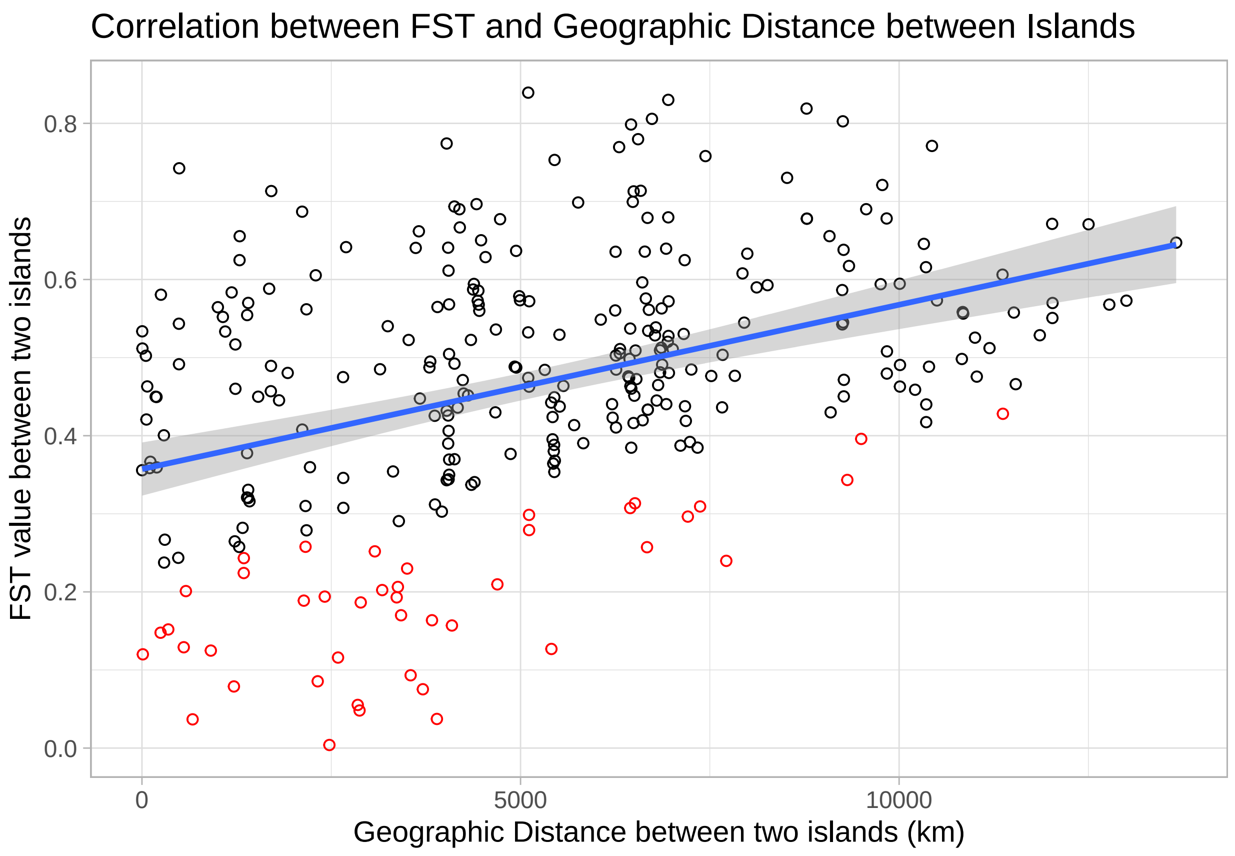
```
## `geom_smooth()` using formula 'y ~ x'
```



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.

```r
fv <- as.vector(LM$fitted.values)
pwd.df2 <- cbind(pwd.df2, fv)
z <- dplyr::mutate(pwd.df2, colour = ifelse(fst.dist < (fv - 0.15), "red", "black"))

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

```
## `geom_smooth()` using formula 'y ~ x'
```


Correlation between FST and Geographic Distance between Islands
FST value between two islands
Geographic Distance between two islands (km)