NB Population Genomic Analysis on Non-Haplotig Masked Outlier SNPs

Amy Zyck

7/6/2020

Population analyses on Outlier (Non-Haplotig) SNP dataset

I first created a new directory `NB_PopGen_Outlier` within `PATH: /home/azyck/NB_capture/NB_ddocent`. I linked `outlierloci.recode.vcf` to this working directory. The RMarkdown file should be saved in this same working directory.

```
#Loading all the necessary packages
library(adegenet)
```

```
## Loading required package: ade4
```

```
## Registered S3 method overwritten by 'spdep':
##   method   from
##   plot.mst ape
```

```
##
##    /// adegenet 2.1.3 is loaded ////////////
##
##    > overview: '?adegenet'
##    > tutorials/doc/questions: 'adegenetWeb()'
##    > bug reports/feature requests: adegenetIssues()
```

```
library(vcfR)
```

```
##
##    *****       ***   vcfR   ***       *****
##    This is vcfR 1.12.0
##      browseVignettes('vcfR') # Documentation
##      citation('vcfR') # Citation
##    *****       *****      *****       *****
```

```
library("radiator") # Conversion from vcf to a lot of other formats
```

```
## ******************************** IMPORTANT NOTICE ********************************
## radiator v.1.0.0 was modified heavily.
## Read functions documentation and available vignettes.
##
## For reproducibility:
##     radiator version: 1.0.0
##     radiator build date: R 3.5.1; ; 2019-03-20 13:31:04 UTC; unix
##     Keep zenodo DOI.
## **********************************************************************************
```

```
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("hierfstat")
```

```
##
## Attaching package: 'hierfstat'
```

```
## The following object is masked from 'package:adegenet':
##
##     read.fstat
```

```
library("ggplot2") #For plotting
library("reshape2") #For plotting
library("plyr")
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library(PCAviz)  #Visualizing output of PCA
library("stringr")
library("bigsnpr") # package for Linkage Disequilibrium pruning
```

```
## Loading required package: bigstatsr
```

Outlier SNPs

Making files

Make genind object

`outlierloci.recode.vcf` contains outlier SNPs (putatively under selection) for populations PVD, GB, BIS, and NAR. Steps for generating this VCF file are located in EecSeq_Cvirginica_dDocent.md and EecSeq_Cvirginica_Filtering.md. Population NIN was removed from the VCF file following steps in EecSeq_Cvirginica_OutlierDetection.md.

strata contains population, environmental, and library information for each sample - can be accessed here.

```
my_vcf <- read.vcfR("outlierloci.recode.vcf")
```

```
## Scanning file to determine attributes.
## File attributes:
##   meta lines: 63
##   header_line: 64
##   variant count: 882
##   column count: 49
##
Meta line 63 read in.
## All meta lines processed.
## gt matrix initialized.
## Character matrix gt created.
##   Character matrix gt rows: 882
##   Character matrix gt cols: 49
##   skip: 0
##   nrows: 882
##   row_num: 0
##
Processed variant: 882
## All variants processed
```

```
strata <- read.table("strata", header=TRUE)
```

```
rad.filt <- vcfR2genind(my_vcf, strata = strata, pop = c(rep("BIS", 10),rep("GB", 10),rep("NAR", 10), r
```

```
rad.filt
```

```
## /// GENIND OBJECT /////////
##
##  // 40 individuals; 882 loci; 1,764 alleles; size: 847.6 Kb
##
##  // Basic content
##    @tab:  40 x 1764 matrix of allele counts
##    @loc.n.all: number of alleles per locus (range: 2-2)
##    @loc.fac: locus factor for the 1764 columns of @tab
##    @all.names: list of allele names for each locus
##    @ploidy: ploidy of each individual  (range: 2-2)
##    @type:  codom
##    @call: adegenet::df2genind(X = t(x), sep = sep, pop = ..2, strata = ..1)
##
##  // Optional content
##    @pop: population of each individual (group size range: 10-10)
##    @strata: a data frame with 12 columns ( Individual, Population, Latitude, Longitude, Distance, SE
```

```
#Providing population names for plotting
pop_order <- c("BIS","GB","NAR","PVD")
```

Read in the other info from .strata file and extract information such as locality, latitude, and longitude.

```
info <- as.data.frame(read.table("strata",header = T,sep = "\t",stringsAsFactors = F))
mystrats <- as.data.frame(matrix(nrow = length(indNames(rad.filt)),ncol=10))
colnames(mystrats) <- c("Population","Latitude","Longitude","Distance","SE","Temperature","Salinity","pl
just.strats <- select(info,c("Population"))
```

```
stratted.filt <- strata(rad.filt, formula= Population, combine = TRUE,just.strats)
stratted.filt@other <- select(info, Latitude,Longitude,Distance,SE,Temperature,Salinity,pH,Chlorophylla
```

```
stratted.filt
```

```
## /// GENIND OBJECT /////////
##
##  // 40 individuals; 882 loci; 1,764 alleles; size: 837.1 Kb
##
##  // Basic content
##    @tab:  40 x 1764 matrix of allele counts
##    @loc.n.all: number of alleles per locus (range: 2-2)
##    @loc.fac: locus factor for the 1764 columns of @tab
##    @all.names: list of allele names for each locus
##    @ploidy: ploidy of each individual  (range: 2-2)
##    @type:  codom
##    @call: adegenet::df2genind(X = t(x), sep = sep, pop = ..2, strata = ..1)
##
##  // Optional content
##    @pop: population of each individual (group size range: 10-10)
##    @strata: a data frame with 1 columns ( Population )
##    @other: a list containing: Latitude  Longitude  Distance  SE  Temperature  Salinity  pH  Chlorophy
```

Make hierfstat object

```
hf.filt <- genind2hierfstat(rad.filt, pop = c(rep("BIS", 10),rep("GB", 10),rep("NAR", 10), rep("PVD",10
```

Estimating effective migration surfaces (EEMS)

The program `runeems_snps` implements the EEMS method for analyzing spatial population structure. This version uses the pairwise genetic dissimilarity matrix computed from SNP data.

Here is the code used to run `runeems_snps` in both RStudio and command-line. Detailed steps for running this program can be accessed here. Input files are created in RStudio or manually (see below) and saved to the directory where `runeems_snps` will be run. The program is then run in the command-line. The outputs are then plotted in RStudio.

`runeems_snps` requires three data input files that have the same file name but different extension. The description below assumes that datapath is the full path + the file name (but without the extension).

1. datapath.diffs

`datapath.diffs` is the matrix of average pairwise genetic dissimilarities. This can be computed with bed2diffs from genetic data in plink binary format.

The dissimilarity matrix is nonnegative, symmetric, with 0s on the main diagonal. These conditions are necessary but not sufficient for diffs to be a valid dissimilarity matrix. Mathematically, diffs should be conditionally negative definite.

Steps for generating `datapath.diffs` are completed in RStudio.

```
# V1 methtod to get diffs matrix, preferred
bed2diffs_v1 <- function(Geno) {
  nIndiv <- nrow(Geno)
  nSites <- ncol(Geno)
  Diffs <- matrix(0, nIndiv, nIndiv)
```

```
  for (i in seq(nIndiv - 1)) {
    for (j in seq(i + 1, nIndiv)) {
      x <- Geno[i, ]
      y <- Geno[j, ]
      Diffs[i, j] <- mean((x - y)^2, na.rm = TRUE)
      Diffs[j, i] <- Diffs[i, j]
    }
  }
  Diffs
}
```

```
# V2 method to get .diffs matrix, only if V1 doesn't work
bed2diffs_v2 <- function(Geno) {
  nIndiv <- nrow(Geno)
  nSites <- ncol(Geno)
  Miss <- is.na(Geno)
  ## Impute NAs with the column means (= twice the allele frequencies)
  Mean <- matrix(colMeans(Geno, na.rm = TRUE), ## a row of means
                 nrow = nIndiv, ncol = nSites, byrow = TRUE) ## a matrix with nIndiv identical rows of m
  Mean[Miss == 0] <- 0 ## Set the means that correspond to observed genotypes to 0
  Geno[Miss == 1] <- 0 ## Set the missing genotypes to 0 (used to be NA)
  Geno <- Geno + Mean
  ## Compute similarities
  Sim <- Geno %*% t(Geno) / nSites
  SelfSim <- diag(Sim) ## self-similarities
  vector1s <- rep(1, nIndiv) ## vector of 1s
  ## This chunk generates a `diffs` matrix
  Diffs <- SelfSim %*% t(vector1s) + vector1s %*% t(SelfSim) - 2 * Sim
  Diffs
}
```

```
geno <- stratted.filt@tab
```

```
# Get rid of non-biallelic loci
multi.loci <- names(which(stratted.filt@loc.n.all != 2))
multi.cols <- which(grepl(paste0("^", multi.loci, "\\.\\d+$", collapse = "|"), colnames(geno)))
if (length(multi.cols)) geno <- geno[, - multi.cols]
nloci <- dim(geno)[2] / 2
dim(geno)
```

```
## [1]   40 1764
```

```
stopifnot(identical(stratted.filt@type, 'codom'))
```

```
# bed2diffs functions
diffs.v1 <- bed2diffs_v1(geno)
diffs.v2 <- bed2diffs_v2(geno)
# Round to 6 digits
diffs.v1 <- round(diffs.v1, digits = 6)
diffs.v2 <- round(diffs.v2, digits = 6)
```

Check that the dissimilarity matrix has one positive eigenvalue and nIndiv-1 negative eigenvalues, as required by a full-rank Euclidean distance matrix. If the V1 method does not make a Euclidean matrix, you must use V2.

```
tail(sort(round(eigen(diffs.v1)$values, digits = 2)))
```

```
## [1] -0.19 -0.18 -0.16 -0.14 -0.13 31.16
```

```
tail(sort(round(eigen(diffs.v2)$values, digits = 2)))
```

```
## [1] -0.20 -0.19 -0.16 -0.15 -0.14 30.75
```

```
# Set suffix for EEMS input files
suf <- "outlierdata-filt"
```

```
# This saves the file to directory
write.table(diffs.v1, paste(suf,".v1.diffs",sep=""),
            col.names = FALSE, row.names = FALSE, quote = FALSE)
```

2. datapath.coord

`datapath.coord` are the sample coordinates, two coordinates per sample, one sample per line. The sampling locations should be given in the same order as the rows and columns of the dissimilarity matrix.

Steps for generating `datapath.coord` are completed in RStudio.

```
## Get gps coordinates from previously created info matrix
xOR.info <- dplyr::filter(info)
gps_matrix <- select(xOR.info,c("Longitude","Latitude"))
```

```
#write .coord file
write.table(gps_matrix, paste(suf,".v1.coord",sep=""),col.names = FALSE, row.names = FALSE,quote = FALSE
```

3. datapath.outer

`datapath.outer` are the habitat coordinates, as a sequence of vertices that form a closed polygon. The habitat vertices should be listed counterclockwise and the first vertex should also be the last vertex, so that the outline is a closed ring. Otherwise, EEMS attempts to "correct" the polygon and prints a warning message.

`datapath.outer` is created manually in Excel,based on site coordinates gathered from Google Maps, copied into terminal using `nano`, and saved as the file with the appropriate extension.

**`runeems_snps` is then run in command-line following the steps documented in NB_EEMS_OutlierHap.md

Back in RStudio to plot `runeems_snps` outputs

```
# Install rEEMSplots
library(rEEMSplots)
```

```
# Plotting EEMS after running runeems_snps
path = "./NB_EEMS_Outlier/"
dirs = c(paste0(path,"outlierdata-D200-chain1"), paste0(path,"outlierdata-D300-chain1"), paste0(path,"ou
```

```
eems.plots(mcmcpath = c(paste0(path,"./outlierdata-D200-chain1"), paste0(path,"outlierdata-D300-chain1"
           longlat = T,add.grid=F,add.outline = T,add.demes = T,
           projection.in = "+proj=longlat +datum=WGS84",projection.out = "+proj=merc +datum=WGS84",
           add.map = T,add.abline = T, add.r.squared = T)
```

```
## Input projection: +proj=longlat +datum=WGS84
## Output projection: +proj=merc +datum=WGS84
```

```
## Loading rgdal (required by projection.in)
```

```
## Loading rworldmap (required by add.map)
```

```
## Loading rworldxtra (required by add.map)
```

```
## Using the default DarkOrange to Blue color scheme, with 'white' as the midpoint color.
## It combines two color schemes from the 'dichromat' package, which itself is based on
## a collection of color schemes for scientific data graphics:
##  Light A and Bartlein PJ (2004). The End of the Rainbow? Color Schemes for Improved Data
##  Graphics. EOS Transactions of the American Geophysical Union, 85(40), 385.
## See also http://geog.uoregon.edu/datagraphics/color_scales.htm

## Using 'euclidean' distance to assign interpolation points to Voronoi tiles.

## Processing the following EEMS output directories :

## ./NB_EEMS_Outlier/./outlierdata-D200-chain1./NB_EEMS_Outlier/outlierdata-D300-chain1./NB_EEMS_Outlie

## Plotting effective migration surface (posterior mean of m rates)

## ./NB_EEMS_Outlier/./outlierdata-D200-chain1

## ./NB_EEMS_Outlier/outlierdata-D300-chain1

## ./NB_EEMS_Outlier/outlierdata-D600-chain1

## Using the default DarkOrange to Blue color scheme, with 'white' as the midpoint color.
## It combines two color schemes from the 'dichromat' package, which itself is based on
## a collection of color schemes for scientific data graphics:
##  Light A and Bartlein PJ (2004). The End of the Rainbow? Color Schemes for Improved Data
##  Graphics. EOS Transactions of the American Geophysical Union, 85(40), 385.
## See also http://geog.uoregon.edu/datagraphics/color_scales.htm

## Plotting effective diversity surface (posterior mean of q rates)

## ./NB_EEMS_Outlier/./outlierdata-D200-chain1

## ./NB_EEMS_Outlier/outlierdata-D300-chain1

## ./NB_EEMS_Outlier/outlierdata-D600-chain1

## Using the default DarkOrange to Blue color scheme, with 'white' as the midpoint color.
## It combines two color schemes from the 'dichromat' package, which itself is based on
## a collection of color schemes for scientific data graphics:
##  Light A and Bartlein PJ (2004). The End of the Rainbow? Color Schemes for Improved Data
##  Graphics. EOS Transactions of the American Geophysical Union, 85(40), 385.
## See also http://geog.uoregon.edu/datagraphics/color_scales.htm

## Plotting posterior probability trace

## ./NB_EEMS_Outlier/./outlierdata-D200-chain1

## ./NB_EEMS_Outlier/outlierdata-D300-chain1

## ./NB_EEMS_Outlier/outlierdata-D600-chain1

## Plotting average dissimilarities within and between demes

## ./NB_EEMS_Outlier/./outlierdata-D200-chain1

## ./NB_EEMS_Outlier/outlierdata-D300-chain1

## Using 'euclidean' distance to assign interpolation points to Voronoi tiles.
##
##
##
## Using 'euclidean' distance to assign interpolation points to Voronoi tiles.

## EEMS results for at least two different population grids
```

The plots are saved to the same directory where `runeems_snps` was run.

Pairwise Fst

```
fst.mat <- pairwise.WCfst(hf.filt)
```

```
gindF.fst.mat.triN <- as.matrix(fst.mat)
colnames(gindF.fst.mat.triN) <- pop_order
rownames(gindF.fst.mat.triN) <- pop_order
```

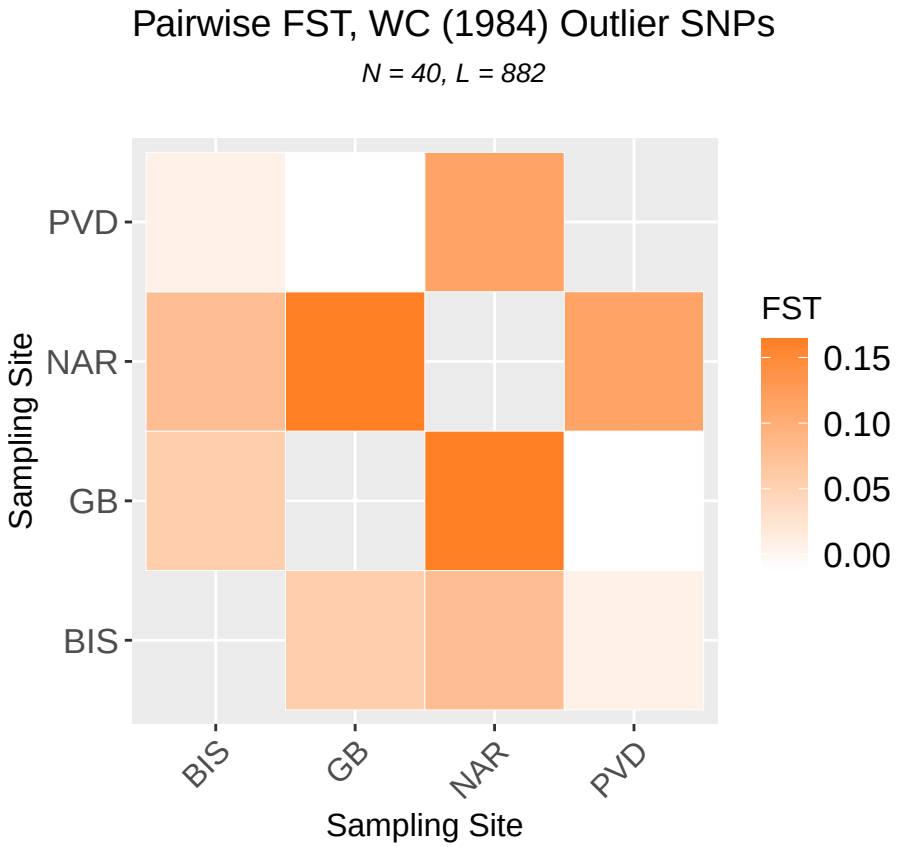
```
meltedN <- melt(gindF.fst.mat.triN, na.rm =TRUE)
round(gindF.fst.mat.triN,4)
```

```
##        BIS      GB    NAR     PVD
## BIS     NA  0.0551 0.0778  0.0076
## GB  0.0551      NA 0.1638 -0.0110
## NAR 0.0778  0.1638     NA  0.1134
## PVD 0.0076 -0.0110 0.1134      NA
```

```
summary(meltedN$value)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.010976  0.007619  0.066479  0.067805  0.113431  0.163796
```

```
#Plotting Pairwise fst
neutral <- ggplot(data = meltedN, aes(Var2, Var1, fill = value))+ geom_tile(color = "white")+
  scale_fill_gradient(low = "white", high = "chocolate1", name="FST")  +
  ggtitle(expression(atop("Pairwise FST, WC (1984) Outlier SNPs", atop(italic("N = 40, L = 882"), ""))))
  labs( x = "Sampling Site", y = "Sampling Site") +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, size = 12, hjust = 1),axis.text.y = element_te
  theme(axis.title = element_text(size = 12),legend.text = element_text(size =13), legend.title = elemer
  theme(plot.title = element_text(size = 14)) +
coord_fixed()
neutral
```

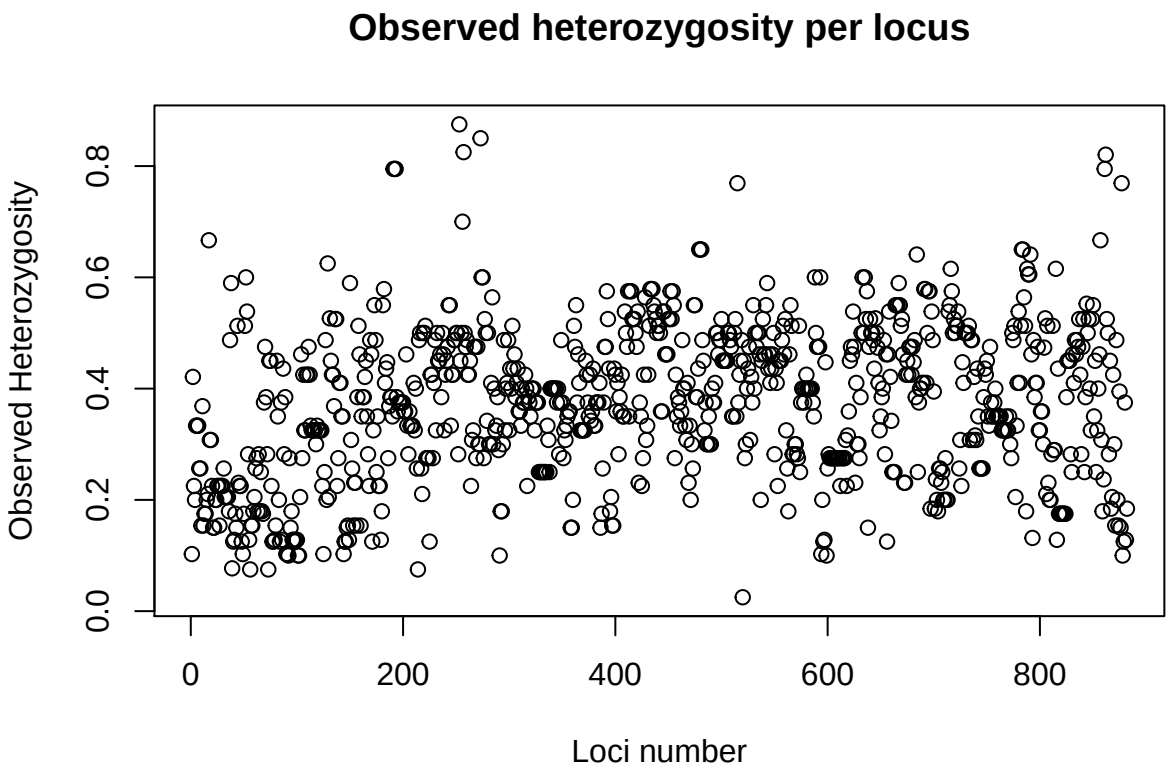


Genetic diversity (observed and expected heterozygosity)

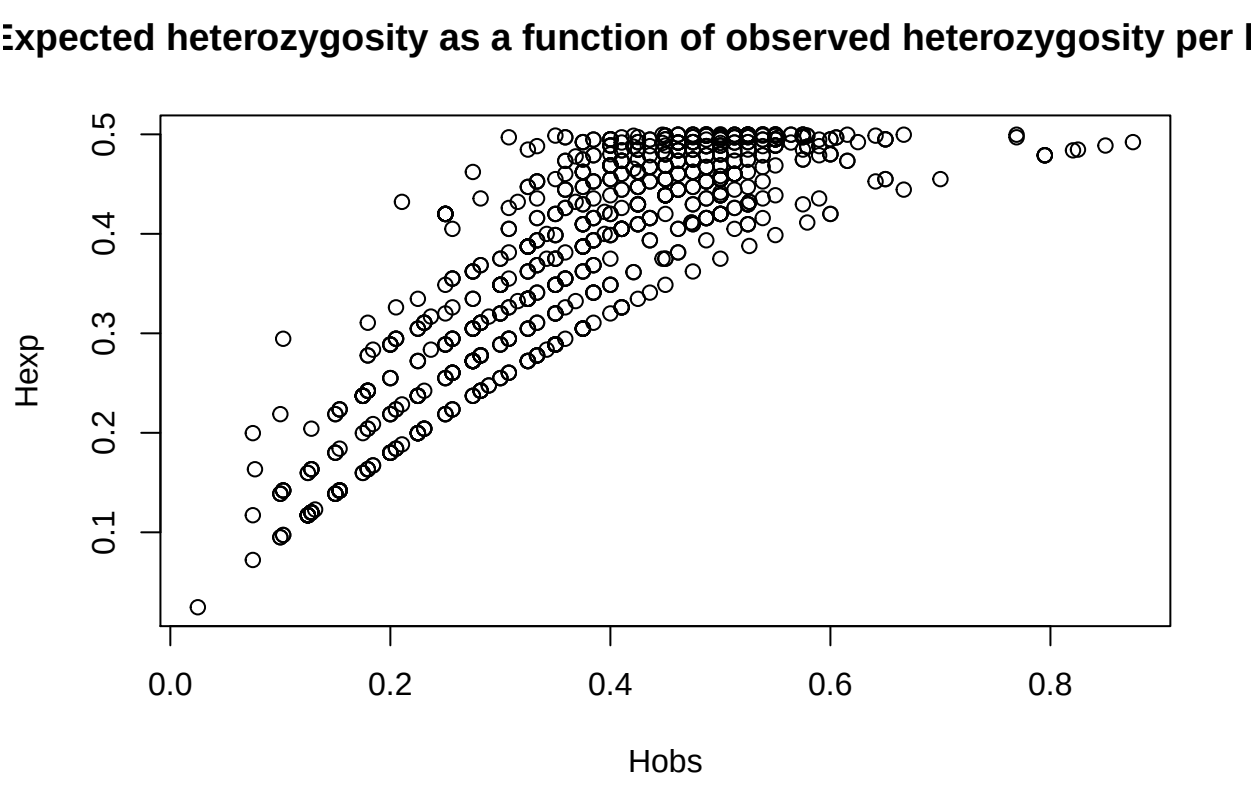
```
comb <- summary(stratted.filt)
names(comb)
```

```
## [1] "n"         "n.by.pop"  "loc.n.all" "pop.n.all" "NA.perc"   "Hobs"
## [7] "Hexp"
```

```
plot(comb$Hobs, xlab="Loci number", ylab="Observed Heterozygosity",
     main="Observed heterozygosity per locus")
```




```
plot(comb$Hobs,comb$Hexp, xlab="Hobs", ylab="Hexp",
     main="Expected heterozygosity as a function of observed heterozygosity per locus")
```




```
bartlett.test(list(comb$Hexp, comb$Hobs)) # a test : HO: Hexp = Hobs
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  list(comb$Hexp, comb$Hobs)
## Bartlett's K-squared = 36.442, df = 1, p-value = 1.573e-09
```

Significant difference between Observed and expected heterzygosity.

```
basicstat <- basic.stats(hf.filt, diploid = TRUE, digits = 3)
```

```
as.data.frame(basicstat$overall)
```

```
##      basicstat$overall
## Ho                0.373
## Hs                0.356
## Ht                0.377
## Dst               0.020
## Htp               0.383
## Dstp              0.027
## Fst               0.054
## Fstp              0.071
## Fis              -0.046
## Dest              0.042
```

```
# get bootstrap confidence values for Fis
boot <- boot.ppfis(hf.filt,nboot = 1000)
```

```
boot5 <- boot.ppfis(hf.filt,nboot = 1000,quant = 0.5)
```

```
# add latitude for each population
latitude = c(41.545, 41.654, 41.505, 41.816)
```

```
# add longitude for each population
longitude = c(-71.431, -71.445, -71.453, -71.391)
```

```
# add distance for each population
distance = c(4.76, 0.47, 15.41, 1.49)
```

```
# add sewage effluent for each population
sewage = c(8.82, 14.60, 2.03, 59.86)
```

```
# add temperature for each population
temperature = c(23, 24, 25, 23)
```

```
# add salinity for each population
salinity = c(30, 28, 18, 25)
```

```
# add pH for each population
pH = c(7.9, 7.4, 7.6, 7.4)
```

```
# add Chlorophylla for each population
Chlor_a = c(4.9, 18.8, 4.6, 8.1)
```

```
# add DO for each population
DO = c(8.2, 5.7, 7, 4.9)
```

```
# combine all pop statistics
colnames(basicstat$Ho) <- pop_order
Ho <- colMeans(basicstat$Ho,na.rm = T)
He <- colMeans(basicstat$Hs,na.rm = T)
Fis<- boot5$fis.ci$ll
y <- cbind(pop_order,Ho,He,Fis,boot$fis.ci,latitude, longitude, distance,sewage, temperature, salinity,
y
```

```
##     pop_order        Ho        He     Fis      ll      hl latitude longitude
## BIS       BIS 0.3946032 0.3363549 -0.1736 -0.1977 -0.1498   41.545   -71.431
## GB         GB 0.3721315 0.3934739  0.0541  0.0335  0.0739   41.654   -71.445
## NAR       NAR 0.3521531 0.3261735 -0.0802 -0.1052 -0.0544   41.505   -71.453
## PVD       PVD 0.3712426 0.3686961 -0.0070 -0.0286  0.0140   41.816   -71.391
##     distance sewage temperature salinity  pH Chlor_a  DO
## BIS     4.76   8.82          23       30 7.9     4.9 8.2
## GB      0.47  14.60          24       28 7.4    18.8 5.7
## NAR    15.41   2.03          25       18 7.6     4.6 7.0
## PVD     1.49  59.86          23       25 7.4     8.1 4.9
```

```
summary(He)
```

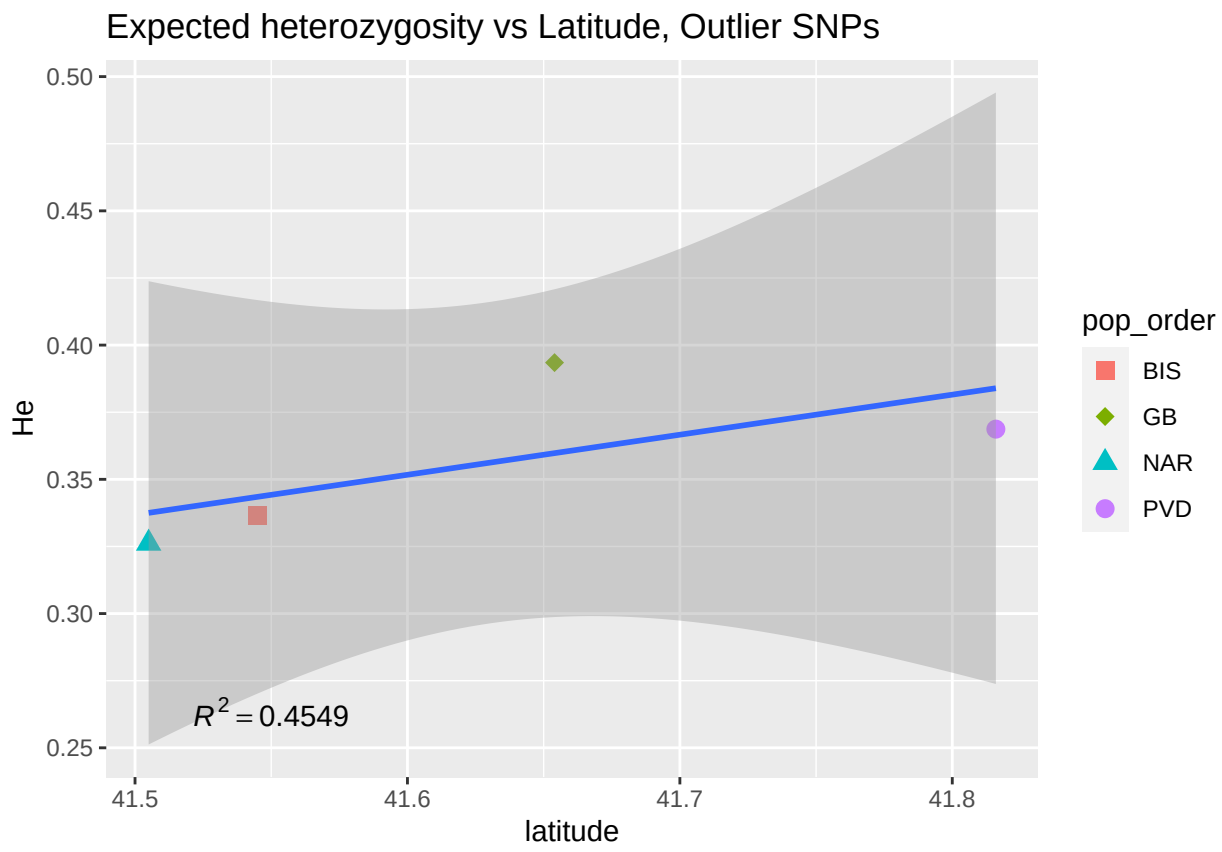
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3262  0.3338  0.3525  0.3562  0.3749  0.3935
```

```
summary(Fis)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.173600 -0.103550 -0.043600 -0.051675  0.008275  0.054100
```

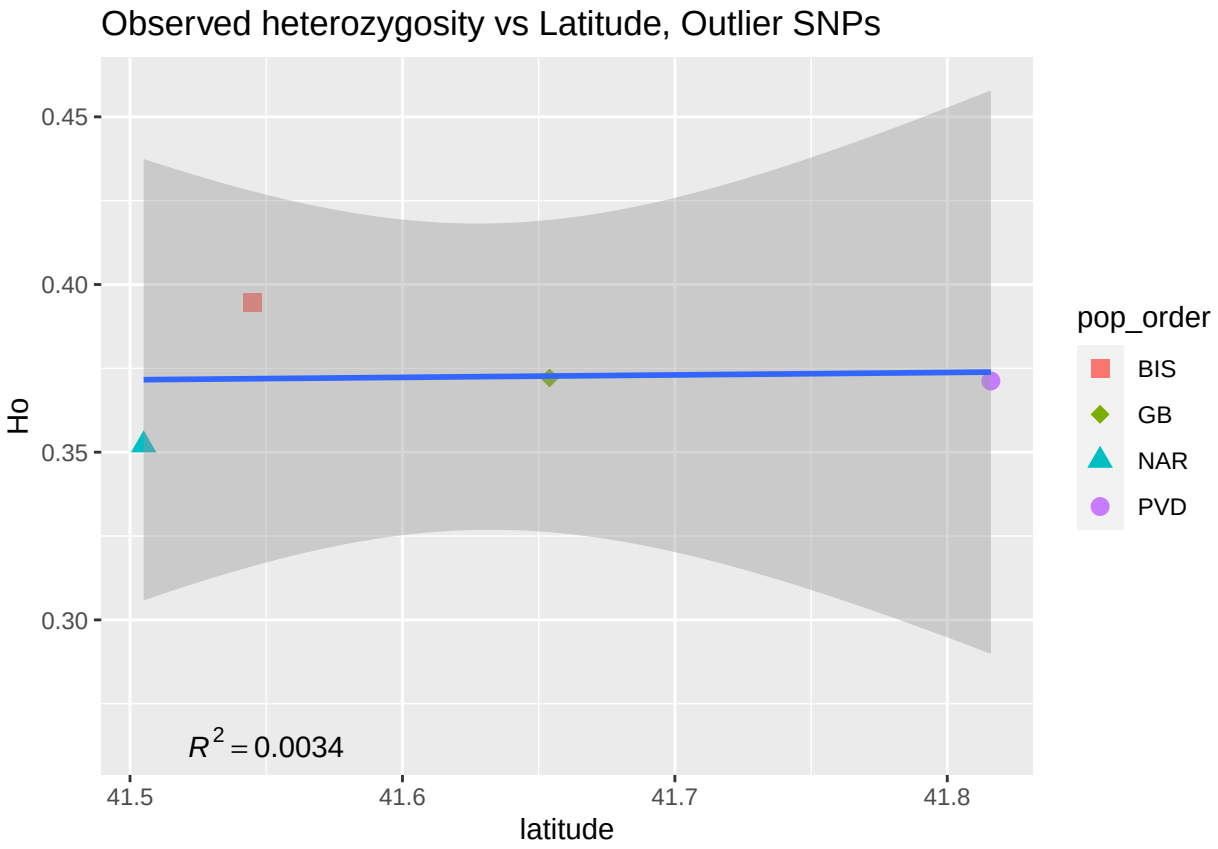
```
# Plot He vs Latitude
R2 = round(summary(lm(y$He ~ y$latitude))$r.squared, 4)
ggplot(y, aes(x = latitude, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Latitude, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R2), x=41.55, y=0.2635, parse=T) +
scale_x_continuous()
```

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



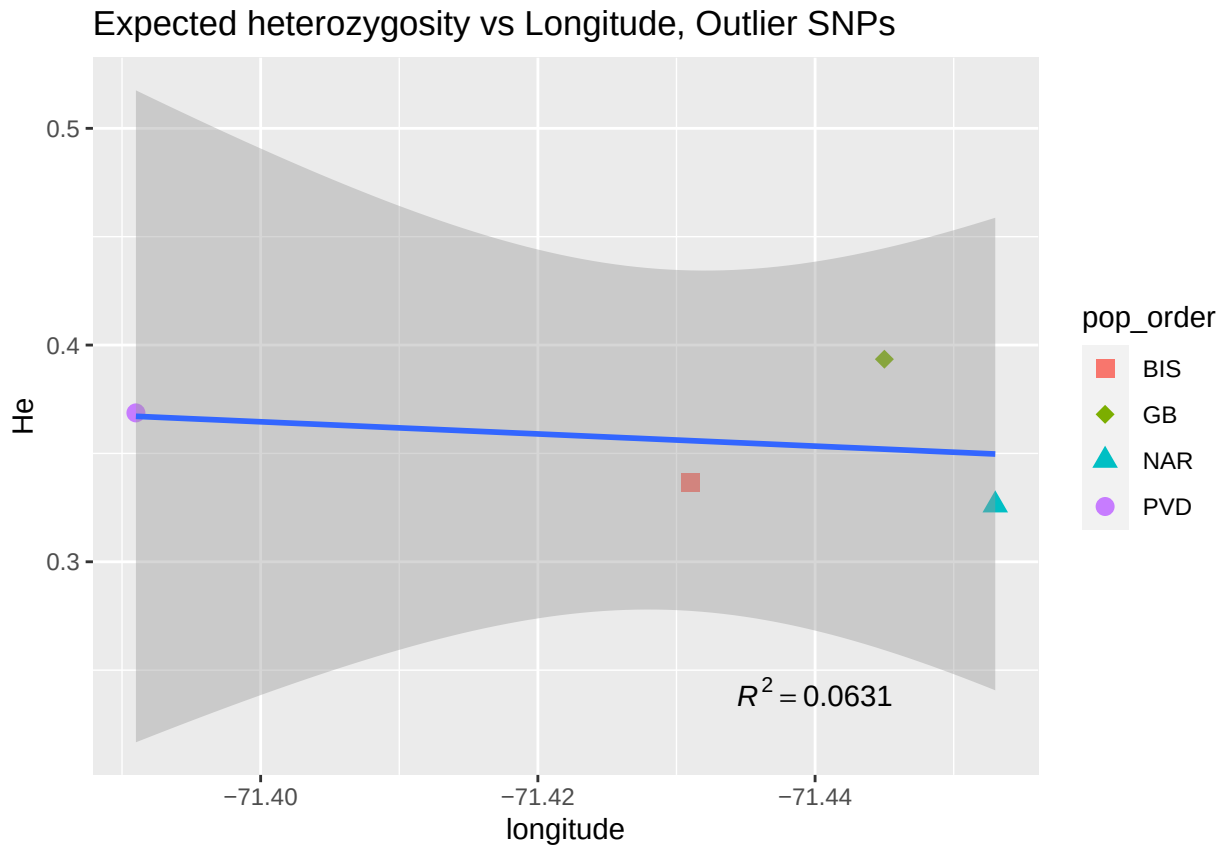

```
# Plot Ho vs Latitude
R2 = round(summary(lm(y$Ho ~ y$latitude))$r.squared, 4)
ggplot(y, aes(x = latitude, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Latitude, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R2), x=41.55, y=0.2635, parse=T) +
scale_x_continuous()
```

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



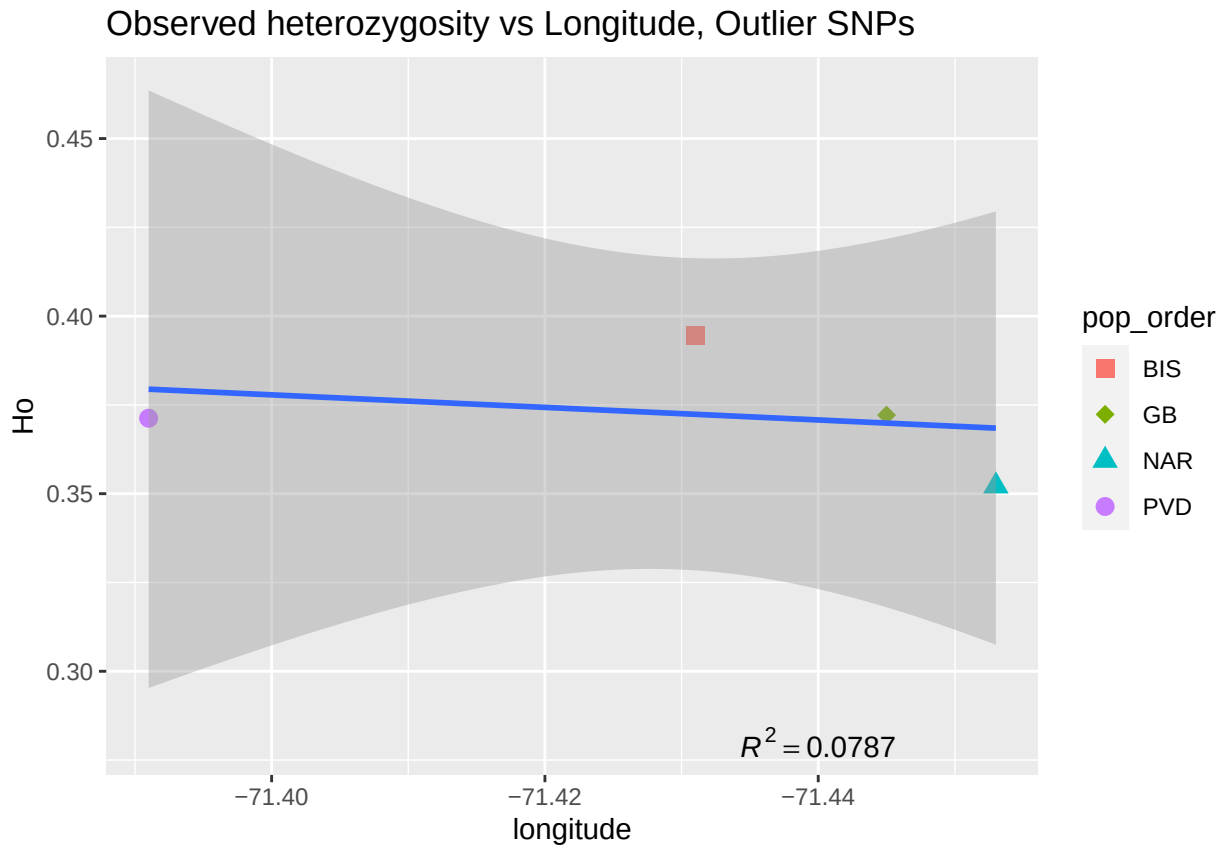

```
#Plot He vs Longitude
R3 = round(summary(lm(y$He ~ y$longitude))$r.squared, 4)
ggplot(y, aes(x = longitude, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Longitude, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R3), x=-71.44, y=0.24, parse=T) +
scale_x_reverse()
```

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



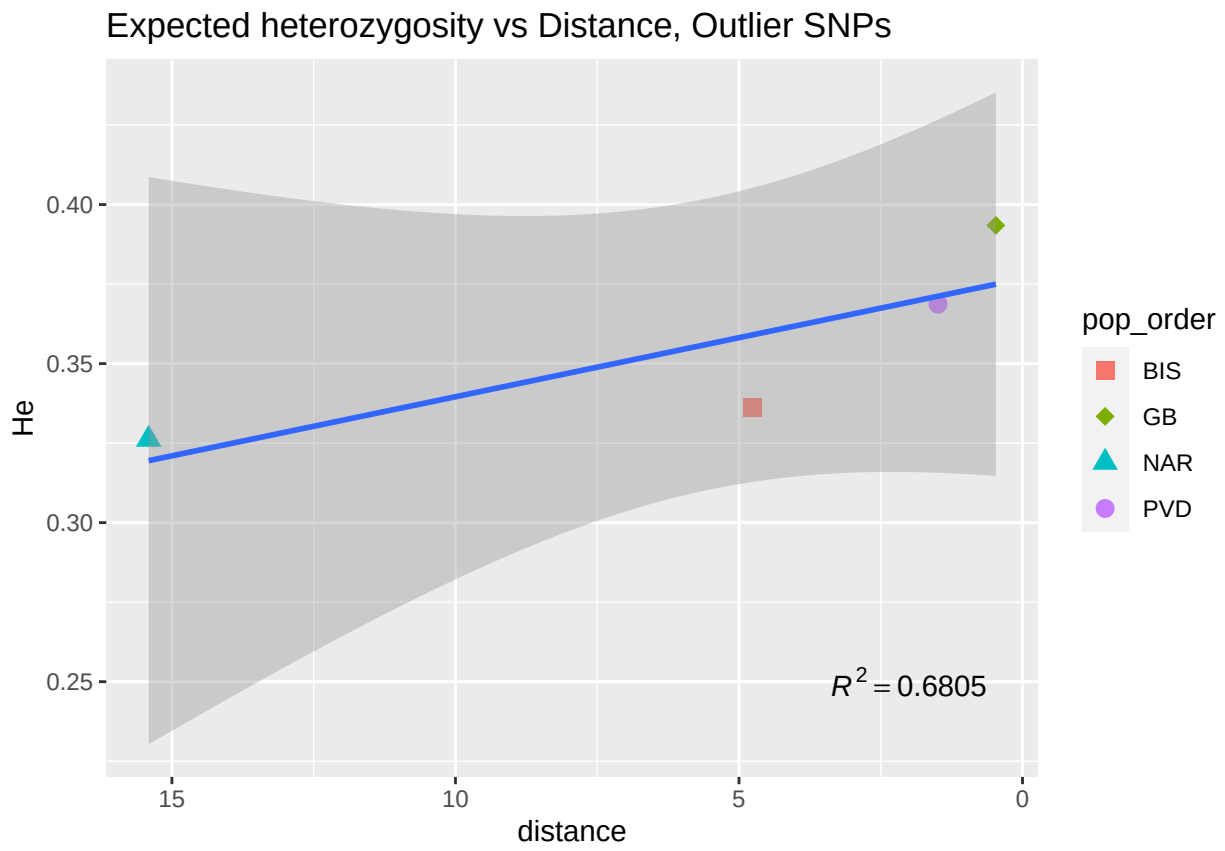

```
#Plot Ho vs Longitude
R3 = round(summary(lm(y$Ho ~ y$longitude))$r.squared, 4)
ggplot(y, aes(x = longitude, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Longitude, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R3), x=-71.44, y=0.28, parse=T) +
scale_x_reverse()
```

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



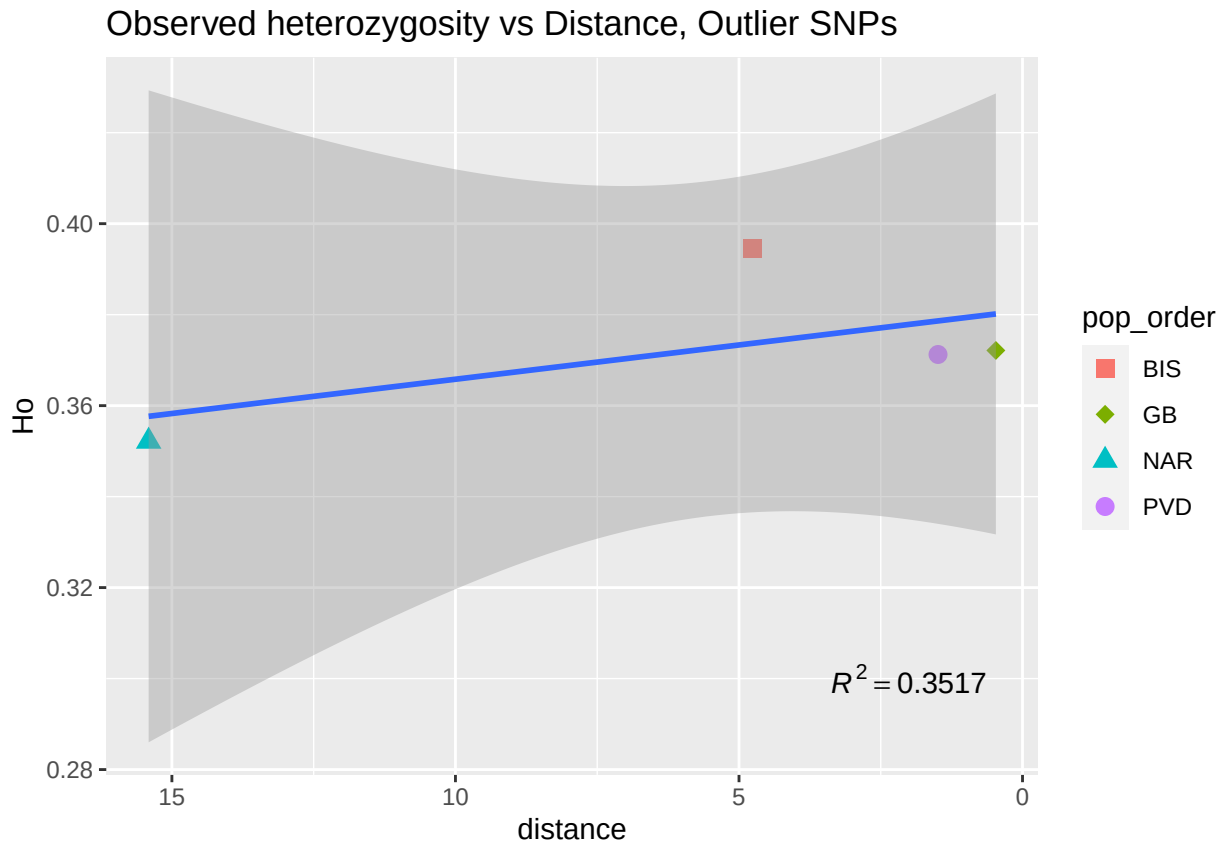

```
#Plot He vs Distance from sewage outflow
R4 = round(summary(lm(y$He ~ y$distance))$r.squared, 4)
ggplot(y, aes(x = distance, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Distance, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=2, y=0.25, parse=T) +
scale_x_reverse()
```

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



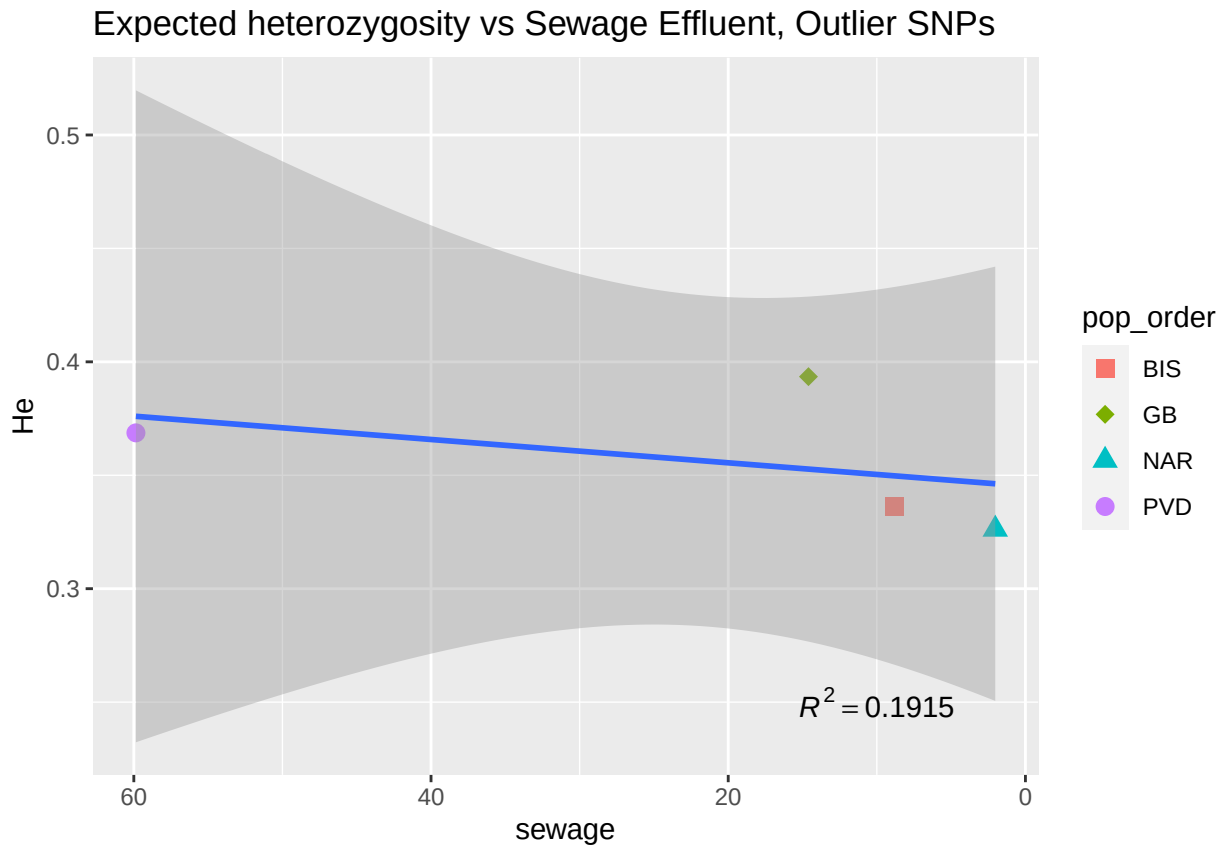

```
#Plot Ho vs Distance from sewage outflow
R4 = round(summary(lm(y$Ho ~ y$distance))$r.squared, 4)
ggplot(y, aes(x = distance, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Distance, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=2, y=0.3, parse=T) +
scale_x_reverse()
```

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



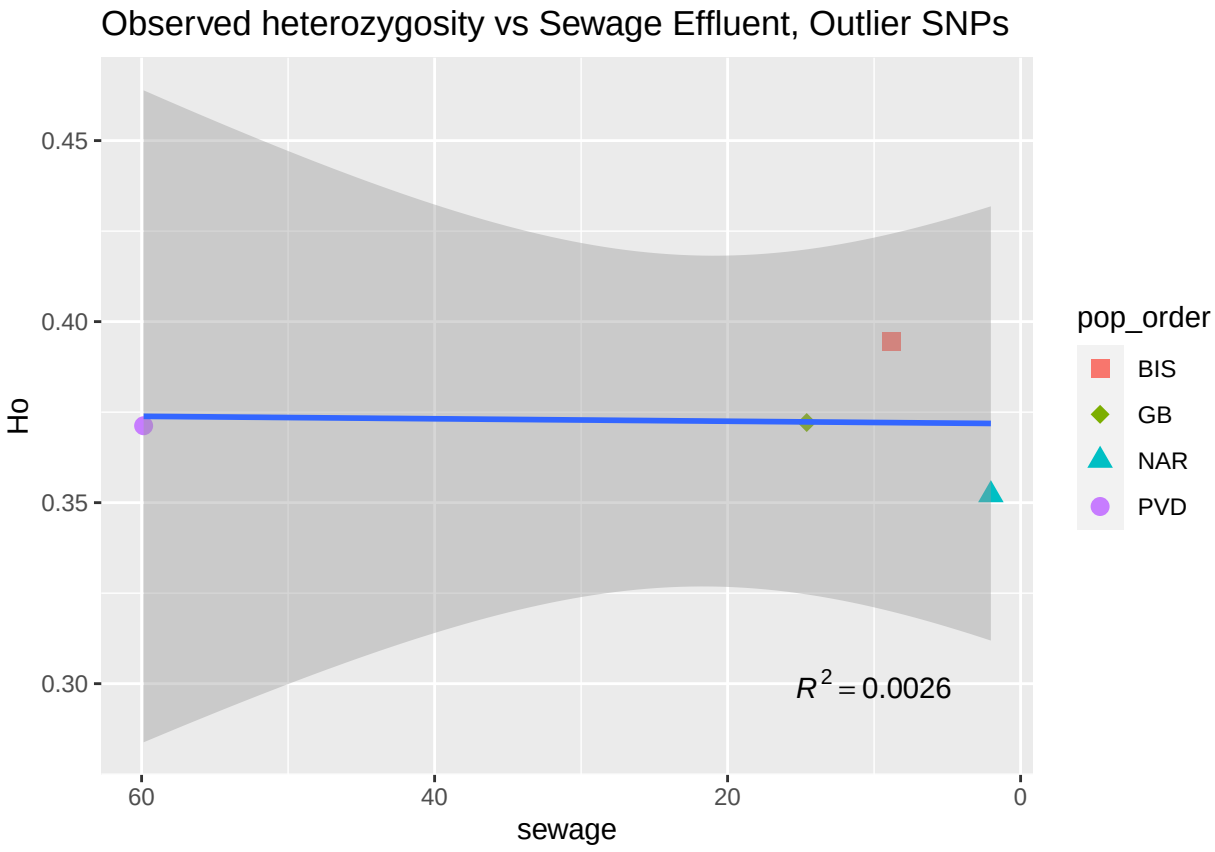

```
#Plot He vs Sewage Effluent
R4 = round(summary(lm(y$He ~ y$sewage))$r.squared, 4)
ggplot(y, aes(x = sewage, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Sewage Effluent, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=10, y=0.25, parse=T) +
scale_x_reverse()
```

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



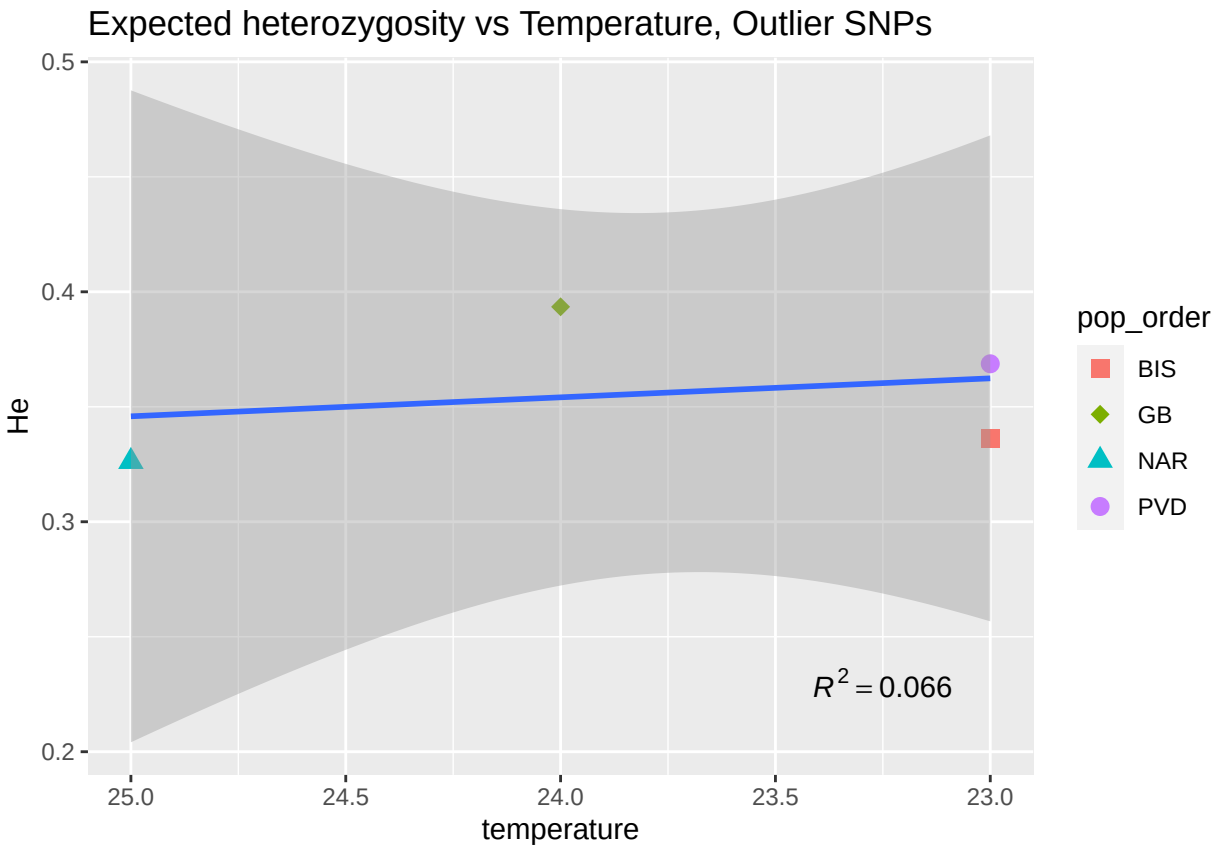

```
#Plot Ho vs Sewage Effluent
R4 = round(summary(lm(y$Ho ~ y$sewage))$r.squared, 4)
ggplot(y, aes(x = sewage, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Sewage Effluent, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=10, y=0.3, parse=T) +
scale_x_reverse()
```

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



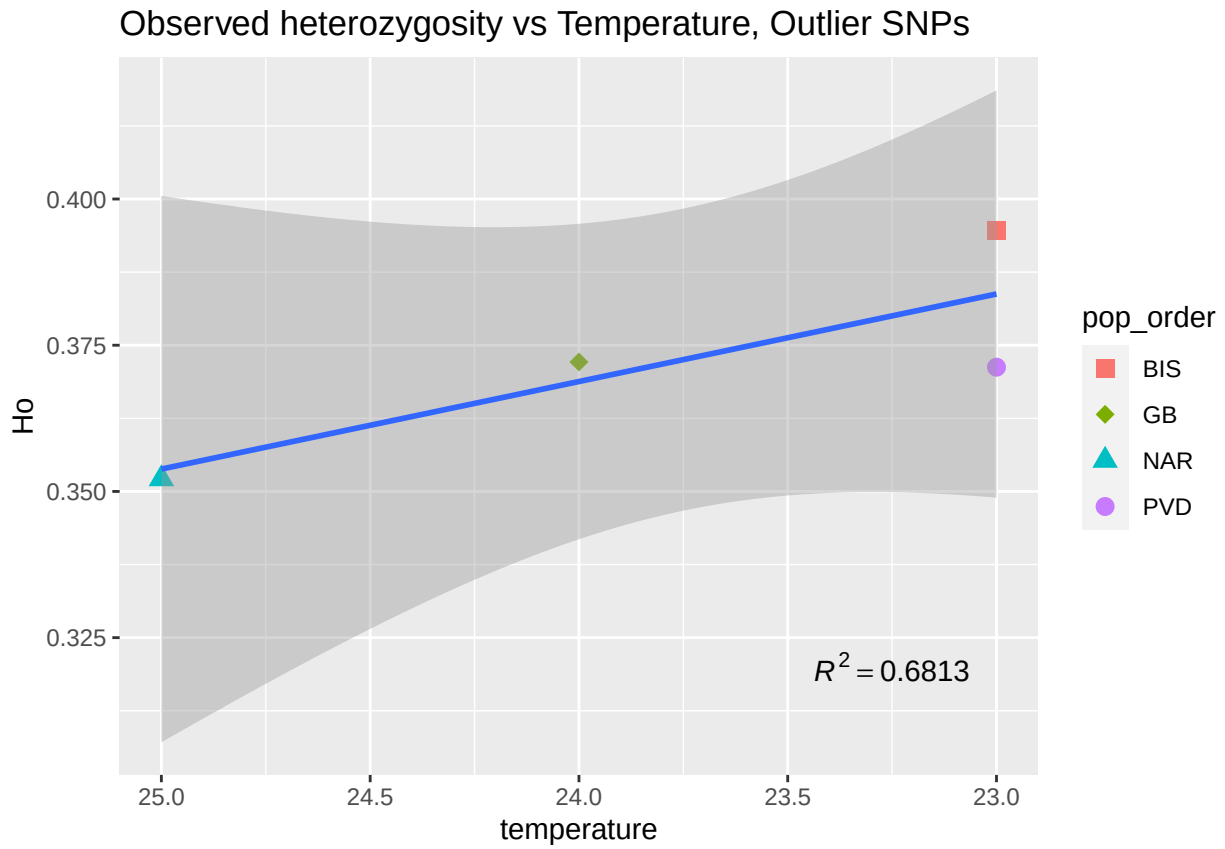

```
#Plot He vs Temperature
R4 = round(summary(lm(y$He ~ y$temperature))$r.squared, 4)
ggplot(y, aes(x = temperature, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Temperature, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=23.25, y=0.23, parse=T) +
scale_x_reverse()
```

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



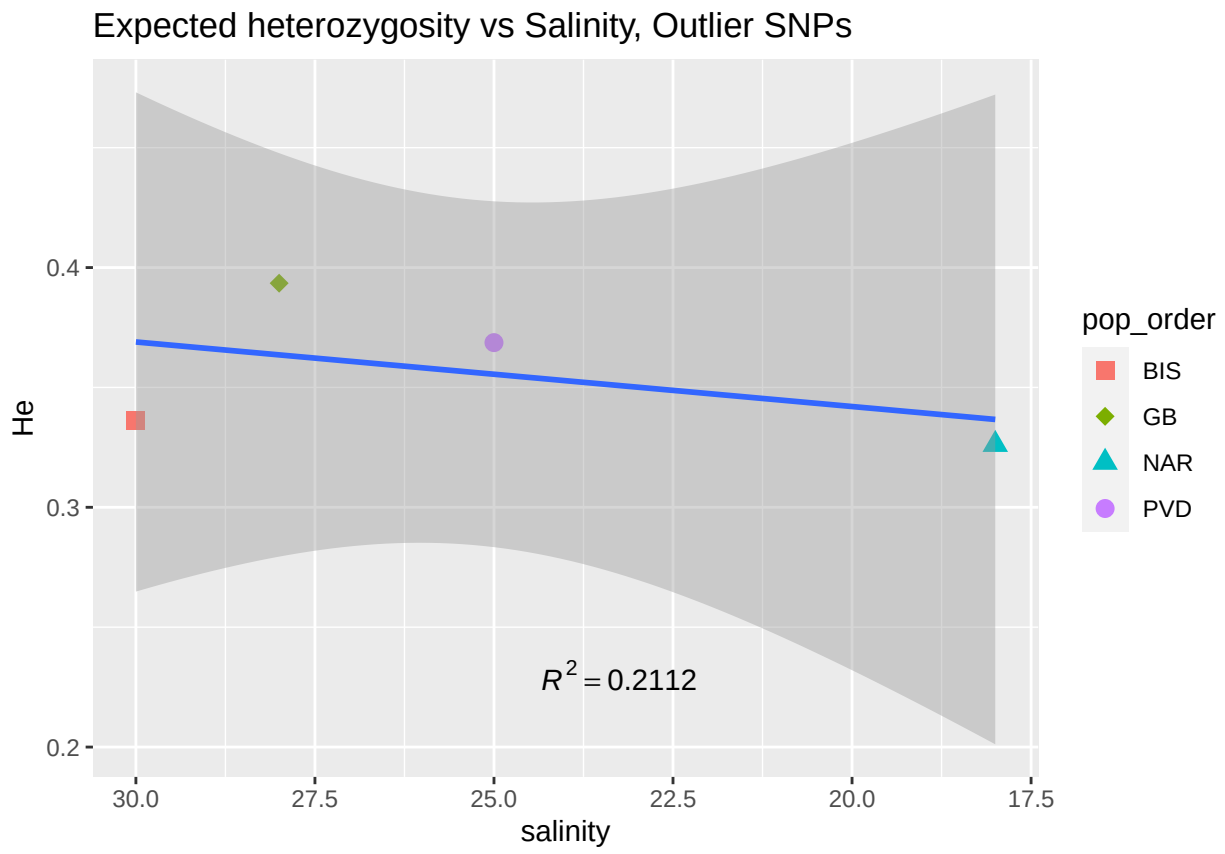

```
#Plot Ho vs Temperature
R4 = round(summary(lm(y$Ho ~ y$temperature))$r.squared, 4)
ggplot(y, aes(x = temperature, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Temperature, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=23.25, y=0.32, parse=T) +
scale_x_reverse()
```

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



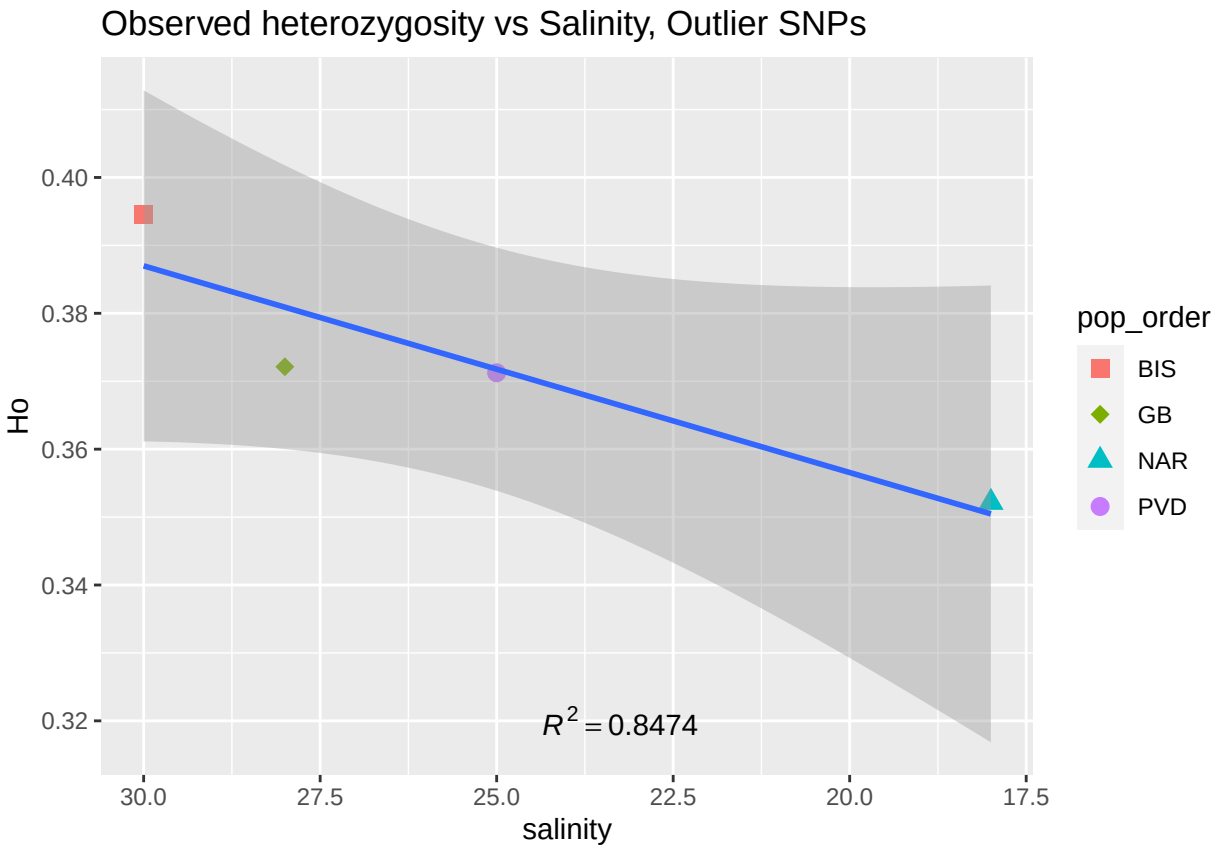

```
#Plot He vs Salinity
R4 = round(summary(lm(y$He ~ y$salinity))$r.squared, 4)
ggplot(y, aes(x = salinity, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Salinity, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=23.25, y=0.23, parse=T) +
scale_x_reverse()
```

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



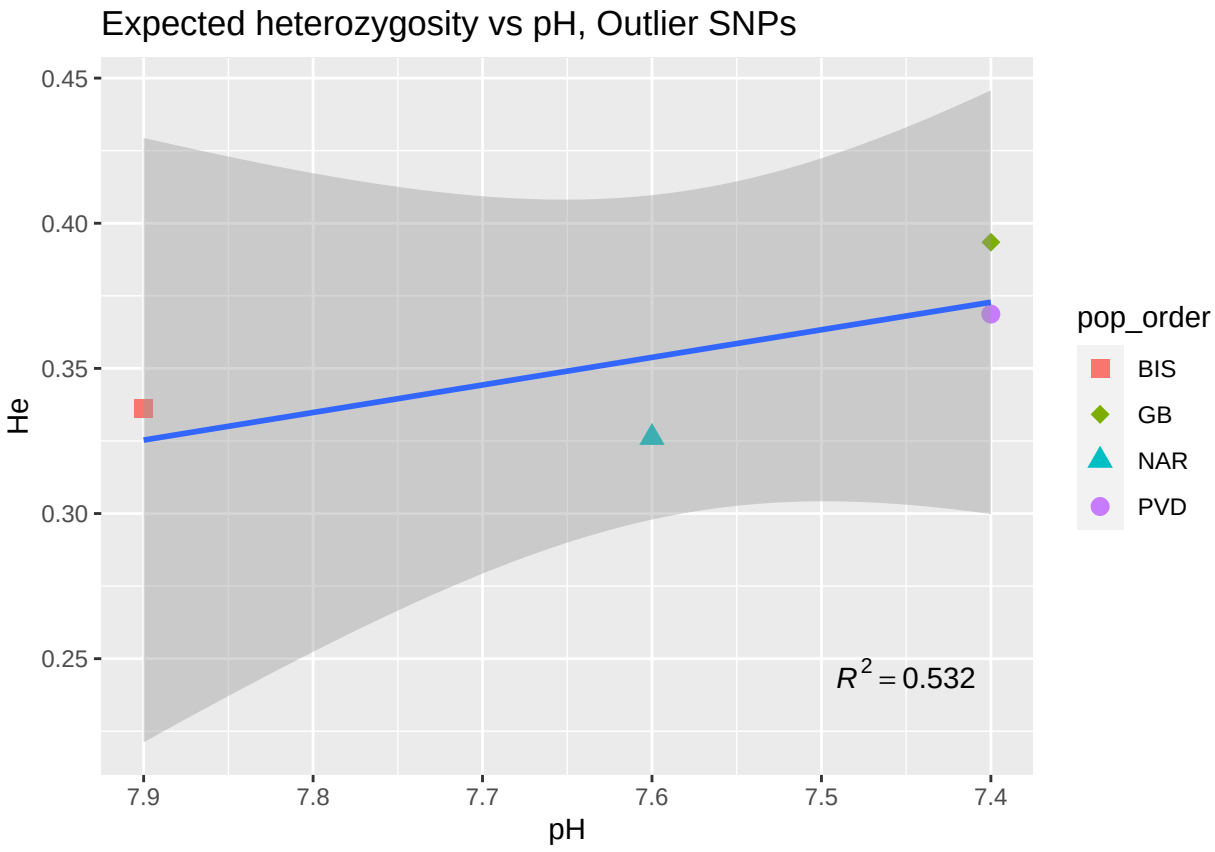

```
#Plot Ho vs Salinity
R4 = round(summary(lm(y$Ho ~ y$salinity))$r.squared, 4)
ggplot(y, aes(x = salinity, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Salinity, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=23.25, y=0.32, parse=T) +
scale_x_reverse()
```

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



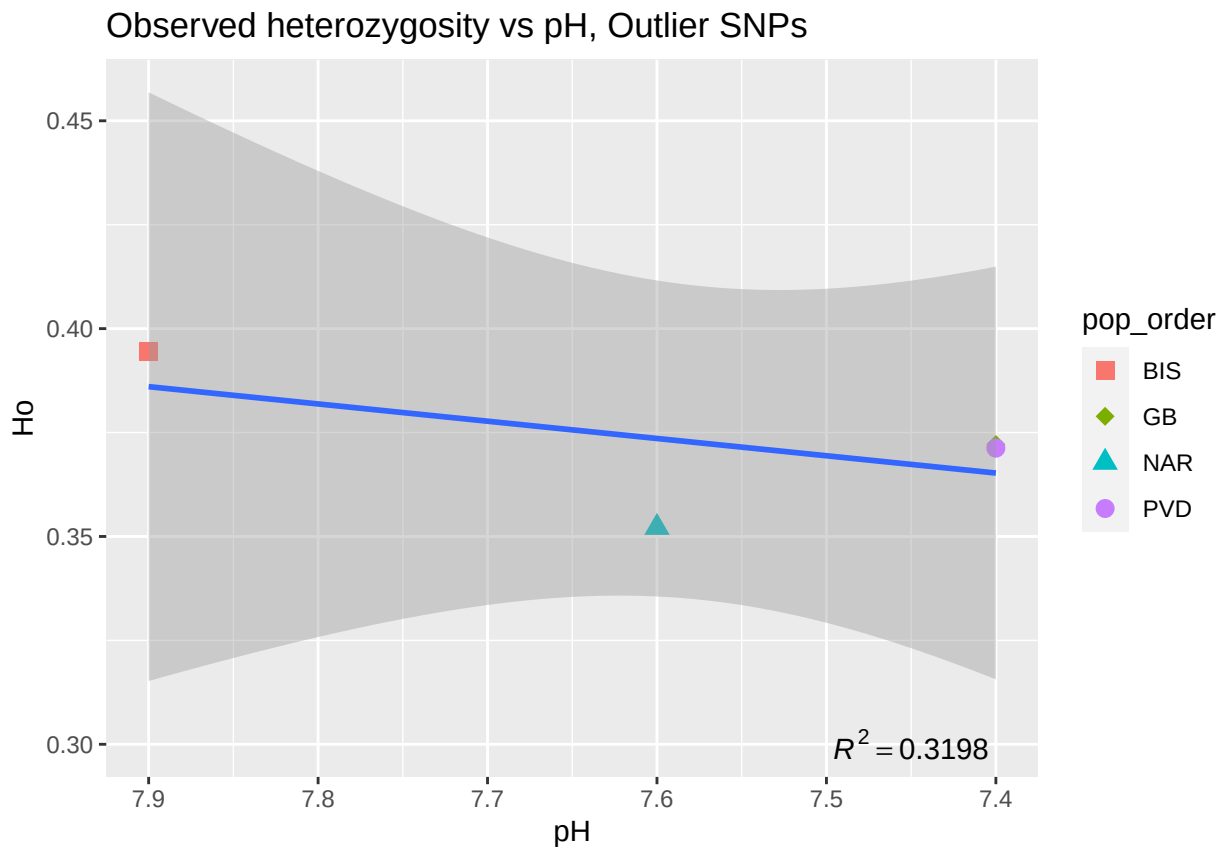

```
#Plot He vs pH
R4 = round(summary(lm(y$He ~ y$pH))$r.squared, 4)
ggplot(y, aes(x = pH, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs pH, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=7.45, y=0.245, parse=T) +
scale_x_reverse()
```

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



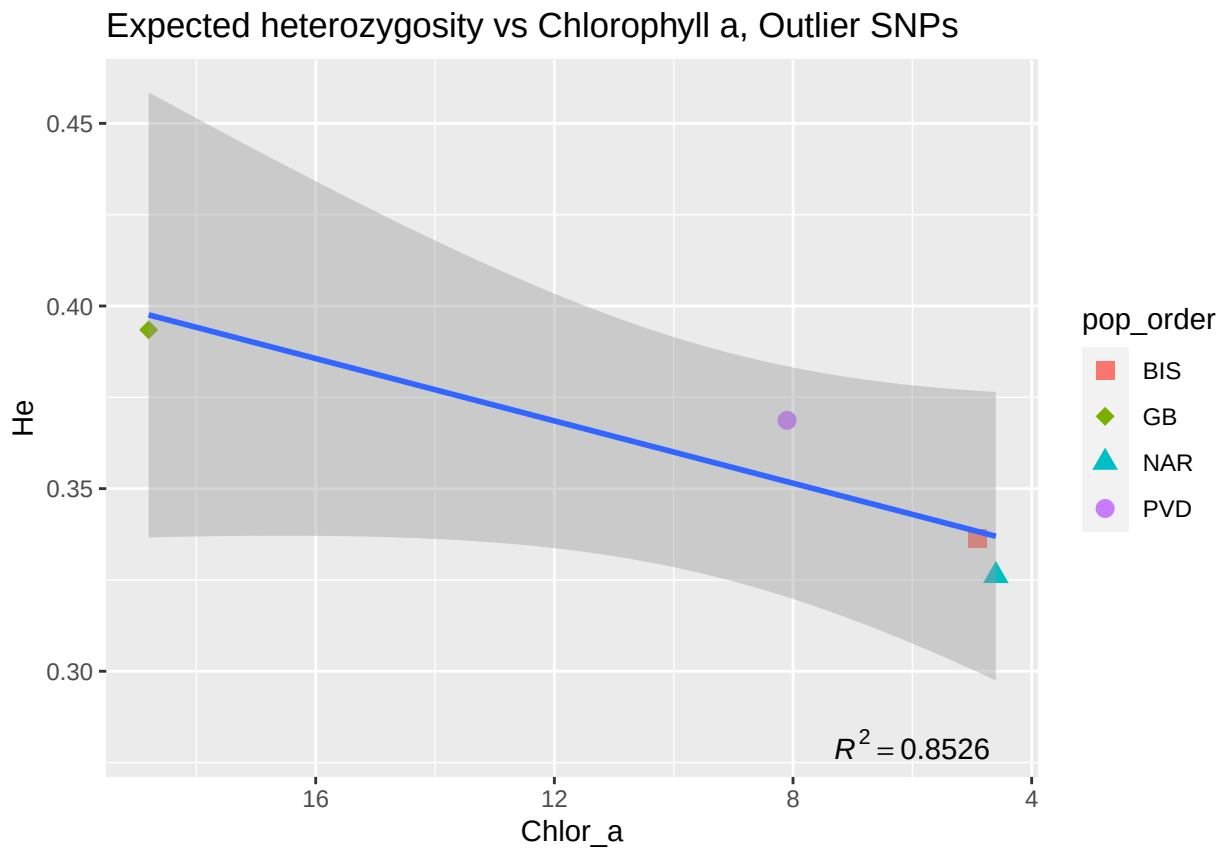

```
#Plot Ho vs pH
R4 = round(summary(lm(y$Ho ~ y$pH))$r.squared, 4)
ggplot(y, aes(x = pH, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs pH, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=7.45, y=0.3, parse=T) +
scale_x_reverse()
```

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



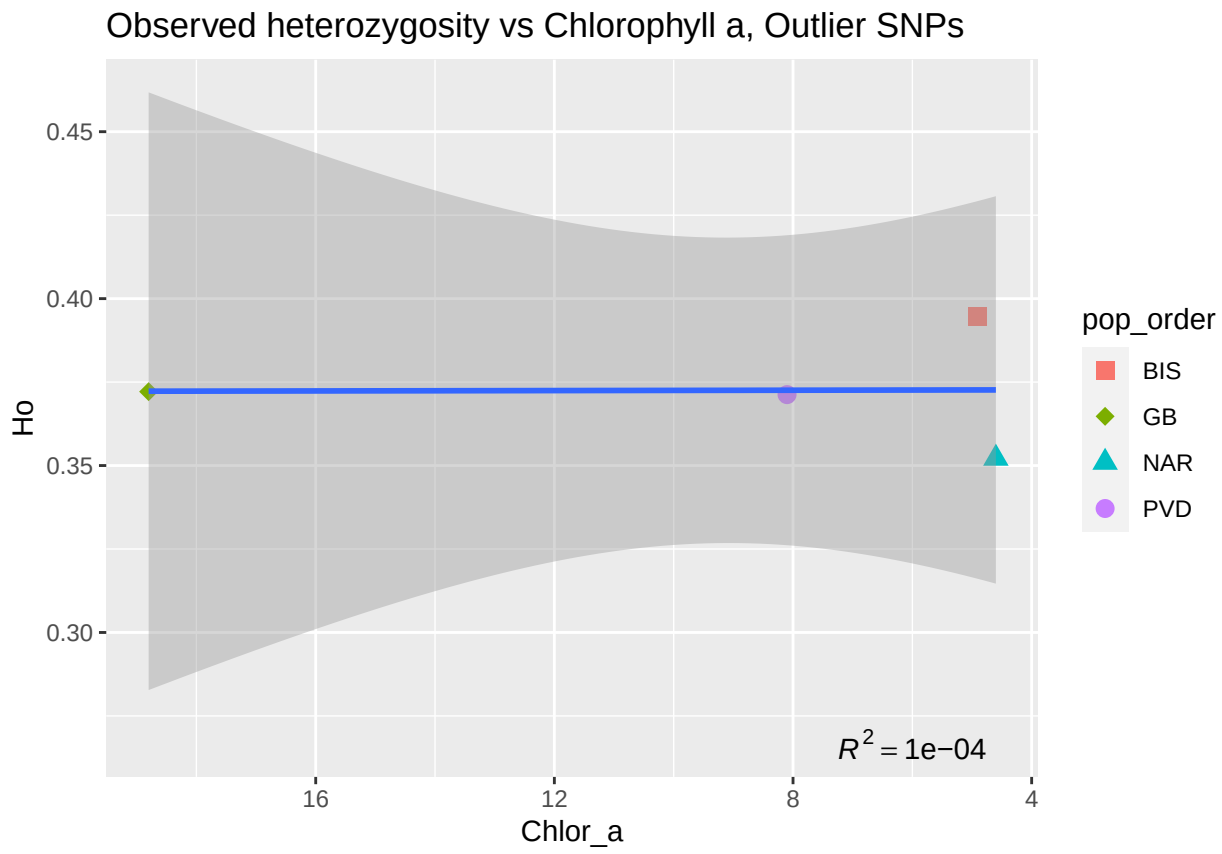

```
#Plot He vs Chlorophyll a
R4 = round(summary(lm(y$He ~ y$Chlor_a))$r.squared, 4)
ggplot(y, aes(x = Chlor_a, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Chlorophyll a, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=6, y=0.28, parse=T) +
scale_x_reverse()
```

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



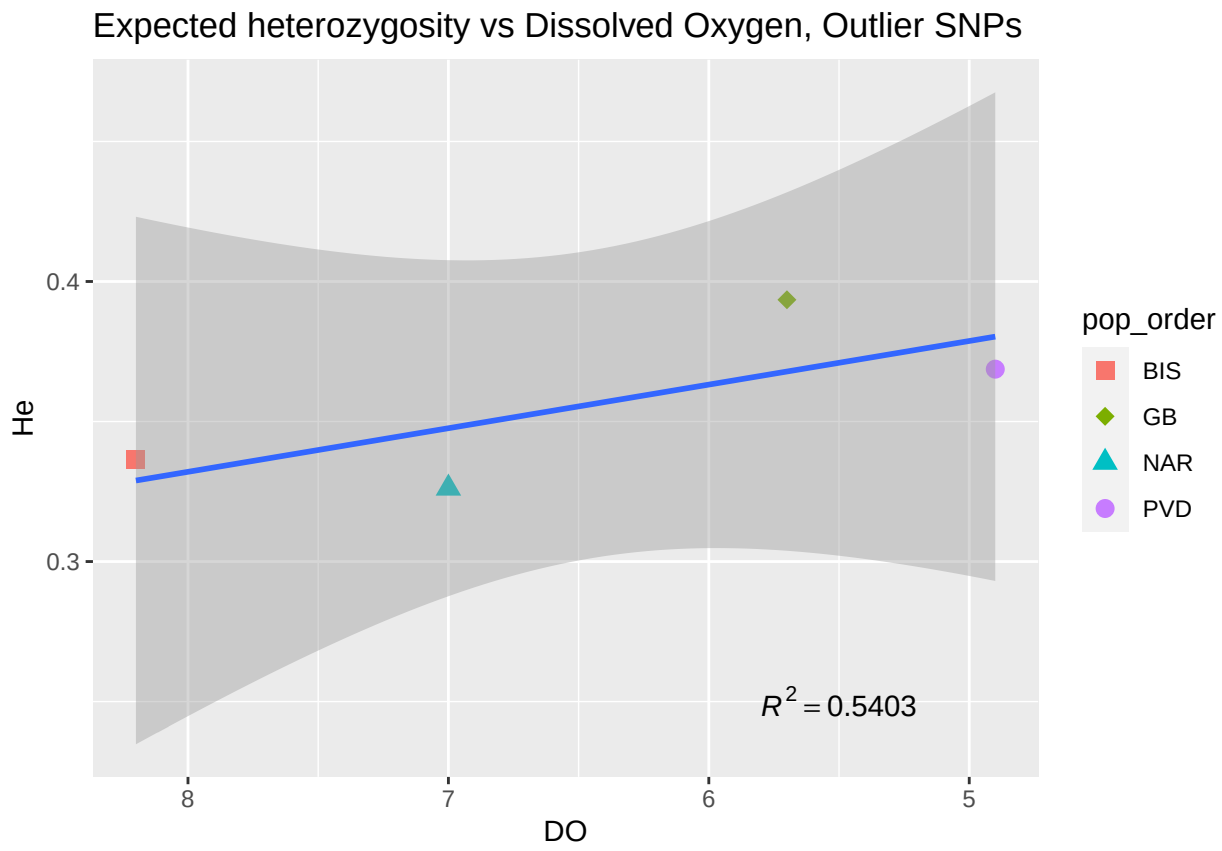

```
#Plot Ho vs Chlorophyll a
R4 = round(summary(lm(y$Ho ~ y$Chlor_a))$r.squared, 4)
ggplot(y, aes(x = Chlor_a, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Chlorophyll a, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=6, y=0.2665, parse=T) +
scale_x_reverse()
```

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



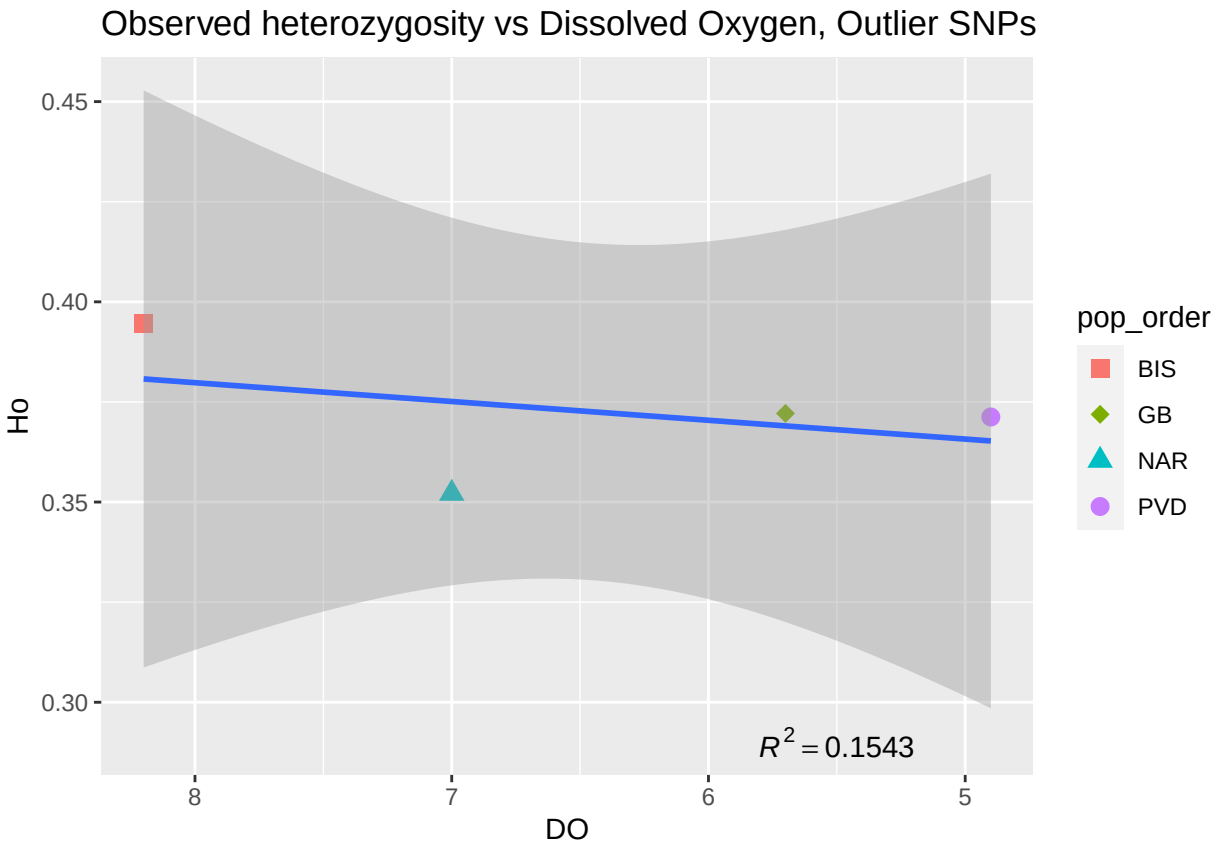

```
#Plot He vs Dissolved Oxygen
R4 = round(summary(lm(y$He ~ y$DO))$r.squared, 4)
ggplot(y, aes(x = DO, y = He)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Expected heterozygosity vs Dissolved Oxygen, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=5.5, y=0.25, parse=T) +
scale_x_reverse()
```

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



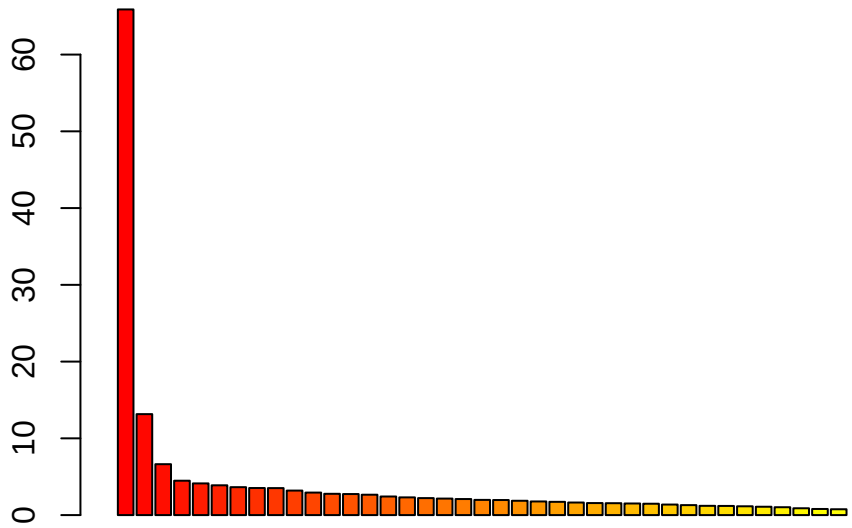

```
#Plot Ho vs Dissolved Oxygen
R4 = round(summary(lm(y$Ho ~ y$DO))$r.squared, 4)
ggplot(y, aes(x = DO, y = Ho)) + geom_point(aes(shape=pop_order, color=pop_order), size = 3) +
  scale_shape_manual(values = c(15,18,17,16)) +
geom_smooth(method=lm) +
ggtitle("Observed heterozygosity vs Dissolved Oxygen, Outlier SNPs") +
annotate(geom = "text", label=paste("italic(R^2)==",R4), x=5.5, y=0.29, parse=T) +
scale_x_reverse()
```

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

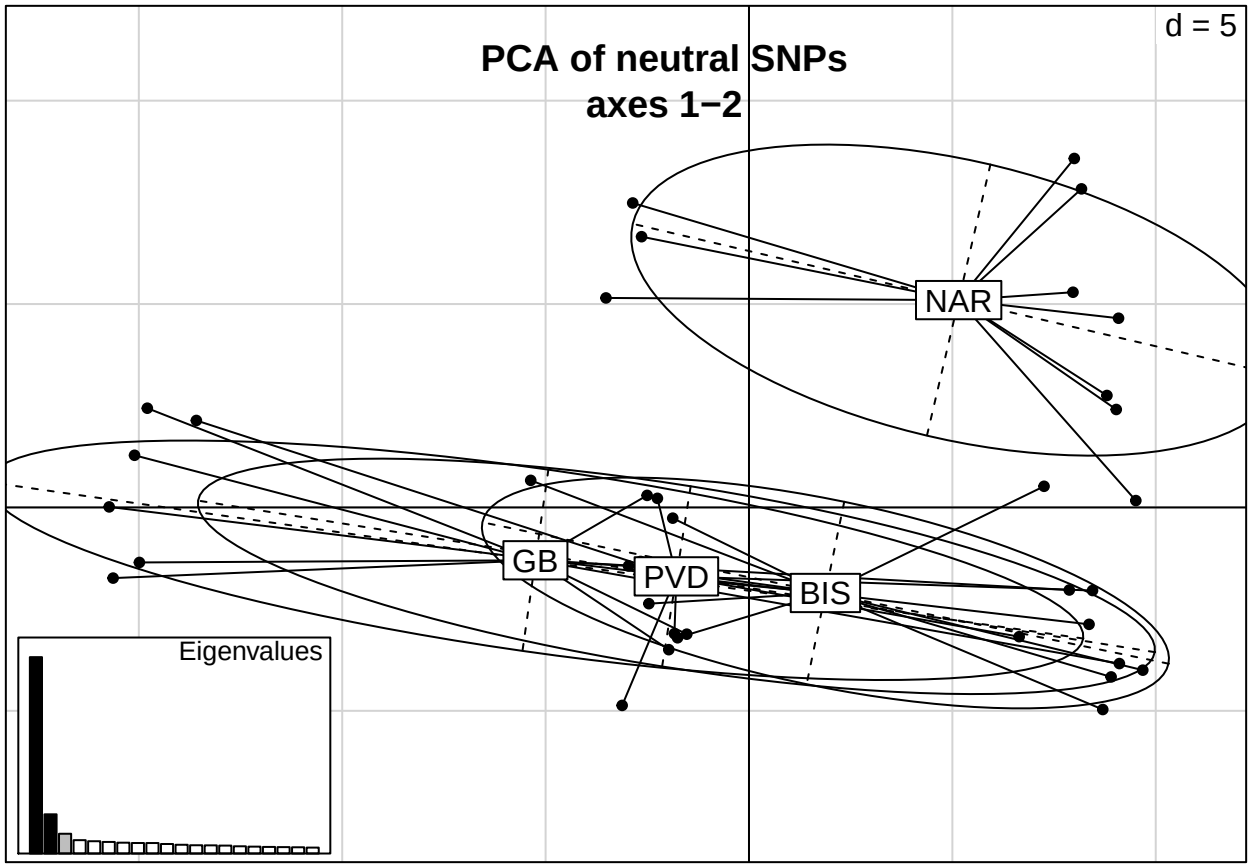


PCA

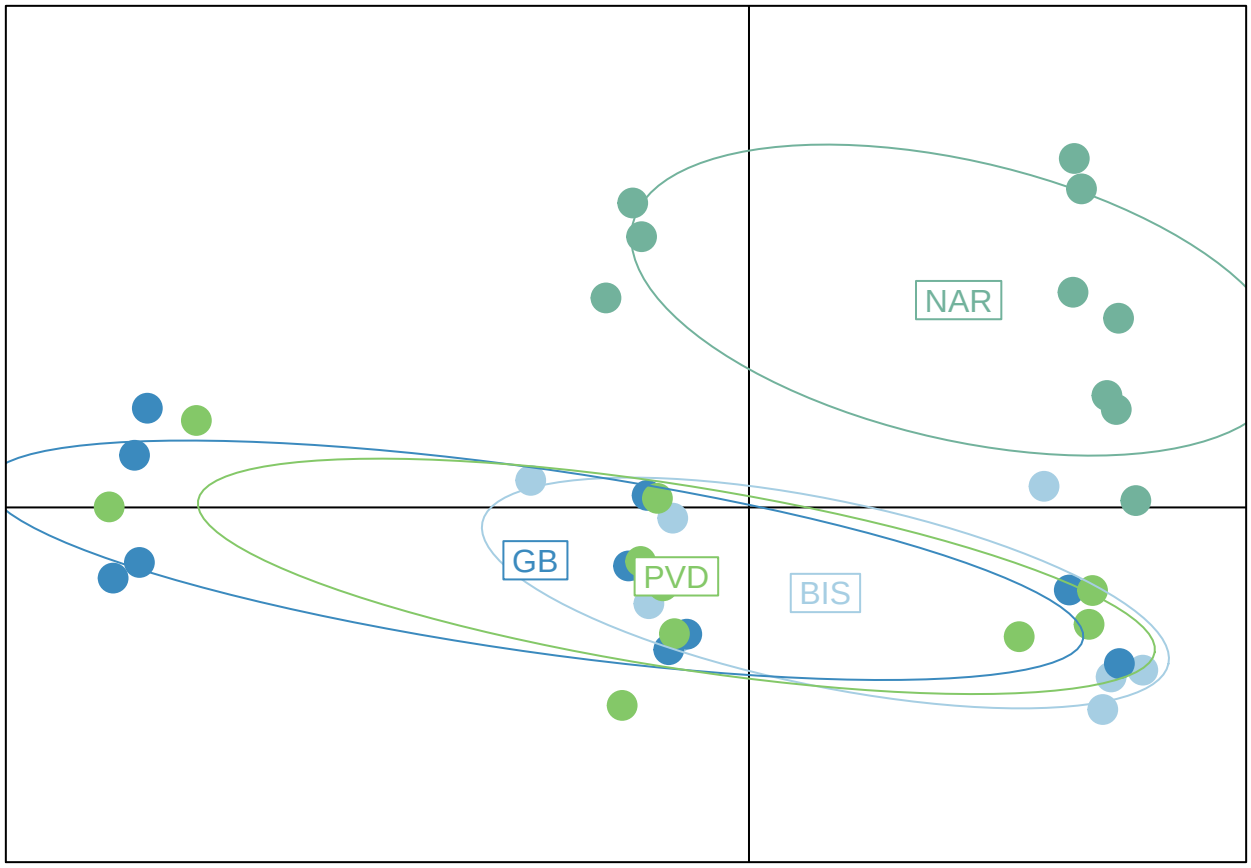
```
X <- tab(stratted.filt, freq = TRUE, NA.method = "mean")
pca1 <- dudi.pca(X, scale = FALSE, scannf = FALSE, nf = 3)
barplot(pca1$eig[1:50], main = "PCA eigenvalues", col = heat.colors(50))
```




```
s.class(pca1$li, pop(stratted.filt))
title("PCA of neutral SNPs\naxes 1-2")
add.scatter.eig(pca1$eig[1:20], 3,1,2)
```




```
col <- funky(15)
s.class(pca1$li, pop(stratted.filt),xax=1,yax=2, col=col, axesell=FALSE, cstar=0, cpoint=3, grid=FALSE)
```



Seascape Redundancy Analysis

This code follows that documented by Tom Jenkins.

Prepare genetic data for redundancy analysis.

Notes before execution:

1. Make sure all required R packages are installed.
2. Set working directory to the location of this R script.

```
# Load packages
library(adegenet)
library(poppr)
```

```
## Registered S3 method overwritten by 'pegas':
##   method      from
##   print.amova ade4
```

```
## This is poppr version 2.8.5. To get started, type package?poppr
## OMP parallel support: available
```

```
##
## Attaching package: 'poppr'
```

```
## The following object is masked from 'package:radiator':
```

```
##
##     private_alleles
```

```
library(dplyr)
library(reshape2)
library(ggplot2)
library(vcfR)
```

```
# Explore data
rad.filt
```

```
## /// GENIND OBJECT /////////
##
##  // 40 individuals; 882 loci; 1,764 alleles; size: 847.6 Kb
##
##  // Basic content
##    @tab:  40 x 1764 matrix of allele counts
##    @loc.n.all: number of alleles per locus (range: 2-2)
##    @loc.fac: locus factor for the 1764 columns of @tab
##    @all.names: list of allele names for each locus
##    @ploidy: ploidy of each individual  (range: 2-2)
##    @type:  codom
##    @call: adegenet::df2genind(X = t(x), sep = sep, pop = ..2, strata = ..1)
##
##  // Optional content
##    @pop: population of each individual (group size range: 10-10)
##    @strata: a data frame with 12 columns ( Individual, Population, Latitude, Longitude, Distance, SE
```

```
nLoc(rad.filt) # number of loci
```

```
## [1] 882
```

```
nPop(rad.filt) # number of sites
```

```
## [1] 4
```

```
nInd(rad.filt) # number of individuals
```

```
## [1] 40
```

```
summary(rad.filt$pop) # sample size
```

```
## BIS  GB NAR PVD
##  10  10  10  10
```

```
# Calculate allele frequencies for each site
allele_freqs = data.frame(rraf(rad.filt, by_pop = TRUE, correction = FALSE), check.names = FALSE)
```

```
# Keep only the first of the two alleles for each SNP (since p=1-q).
allele_freqs = allele_freqs[, seq(1, dim(allele_freqs)[2], 2)]
```

```
# Export allele frequencies
write.csv(allele_freqs, file = "all_allele_freqs.csv", row.names = TRUE)
```

Calculate minor allele frequencies

```
# Separate genind object by site
site_list = seppop(rad.filt)
names(site_list)
```

```
## [1] "BIS" "GB"  "NAR" "PVD"
```

```
# Calculate the minor allele frequency for each site
maf_list = lapply(site_list, FUN = minorAllele)
```

```
# Convert list to dataframe
maf = as.data.frame(maf_list) %>% t() %>% as.data.frame()
```

```
# Export minor allele frequencies
write.csv(maf, file = "minor_allele_freqs.csv", row.names = TRUE)
```

Visualise allele frequencies

```
# Add site labels
allele_freqs$site = rownames(allele_freqs)
```

```
# Function to add regional labels to dataframe
addregion = function(x){
  # If pop label is present function will output the region
  if(x=="BIS") y = " Bissel Cove "
  if(x=="GB") y = " Greenwich Bay "
  if(x=="NAR") y = " Narrow River "
  if(x=="PVD") y = " Bold Point Park "
```

```
  return(y)
}
```

```
# Add regional labels
allele_freqs$region = sapply(rownames(allele_freqs), addregion)
```

```
# Convert dataframe to long format
allele_freqs.long = melt(allele_freqs, id.vars=c("site","region"))
```

```
# Define order of facets using the levels argument in factor
unique(allele_freqs.long$site)
```

```
## [1] "BIS" "GB"  "NAR" "PVD"
```

```
site_order =  c("BIS","GB","NAR","PVD")
allele_freqs.long$site_ord = factor(allele_freqs.long$site, levels = site_order)
```

```
# Define region order
region_order = c(" Bissel Cove "," Greenwich Bay "," Narrow River ", " Bold Point Park ")
allele_freqs.long$region = factor(allele_freqs.long$region, levels = region_order)
```

```
# Create colour scheme
# blue=#377EB8, green=#7FC97F, orange=#FDB462, red=#E31A1C
col_scheme = c("#7FC97F","#377EB8","#FDB462","#E31A1C")
```

A subset of the putatively outlier loci was selected, spanning all 4 outlier detection programs.

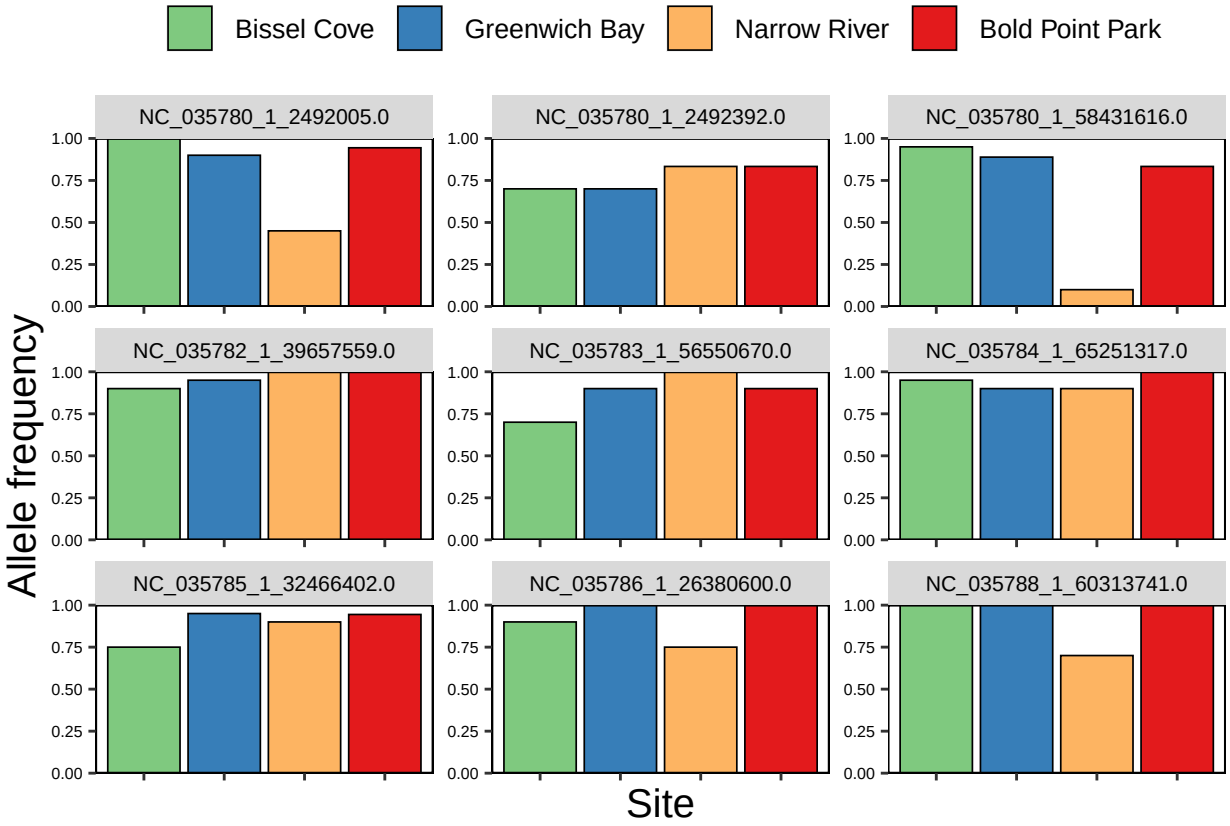
```
# Vector of outlier SNP loci to subset
desired_loci = c("NC_035780.1_2492392","NC_035780.1_58431616","NC_035780.1_2492005","NC_035782.1_396575
desired_loci_ID = sapply(paste(desired_loci, "..", sep = ""),
                         grep,
                         levels(allele_freqs.long$variable),
                         value = TRUE) %>% as.vector()
```

```
# Subset dataset to plot desired SNP loci
allele_freqs.sub = allele_freqs.long %>% filter(variable %in% desired_loci_ID)
```

```
# ggplot2 theme
ggtheme = theme(
  axis.text.x = element_blank(),
  axis.text.y = element_text(colour="black", size=6),
  axis.title = element_text(colour="black", size=15),
  panel.background = element_rect(fill="white"),
  panel.grid.minor = element_blank(),
  panel.grid.major = element_blank(),
  panel.border = element_rect(colour="black", fill=NA, size=0.5),
  plot.title = element_text(hjust = 0.5, size=18),
  legend.title = element_blank(),
  legend.text = element_text(size=10),
  legend.position = "top",
  legend.justification = "centre",
  # facet labels
  strip.text = element_text(colour="black", size=8)
)
```

```
# Plot barplot
ggplot(data = allele_freqs.sub, aes(x = site_ord, y = value, fill = region))+
  geom_bar(stat = "identity", colour = "black", size = 0.3)+
```

```
  facet_wrap(~variable, scales = "free")+
  scale_y_continuous(limits = c(0,1), expand = c(0,0))+
  scale_fill_manual(values = col_scheme)+
  ylab("Allele frequency")+
  xlab("Site")+
  ggtheme
```




```
ggsave("allele_freq.png", width=10, height=8, dpi=300)
ggsave("allele_freq.pdf", width=10, height=8)
```

Prepare environmental data for redundancy analysis.

Environmental variables:

- Distance from sewage effluent source (km)
- Sewage Effluent (PW stats)
- Mean temperature (deg C)
- Mean Salinity (psu)
- Mean pH
- Mean Chlorophyll-a (ug/L)
- Mean Dissolved Oxygen (mg/L)

Environmental data for each population is saved in a `strata_pop` file that can be accessed here

```
# All environmental data was previously saved in strata file
strata_pop <- read.table("strata_pop", header=TRUE)
```

```
strata_pop
```

```
##   Population Latitude Longitude Distance     SE Temperature Salinity  pH
## 1        BIS   41.545   -71.431     4.76  8.825          23       30 7.9
## 2         GB   41.654   -71.445     0.47 14.596          24       28 7.4
## 3        NAR   41.505   -71.453    15.41  2.027          25       18 7.6
## 4        PVD   41.816   -71.391     1.49 59.860          23       25 7.4
##   Chlorophylla  DO
## 1          4.9 8.2
## 2         18.8 5.7
## 3          4.6 7.0
## 4          8.1 4.9
```

```
# Export data as a csv file
write.csv(strata_pop, file="environmental_data.csv", row.names = FALSE)
```

I also prepared spatial data for the redundancy analysis which is documented here.

Allele frequency, environmental, and spatial csv files are saved to your working directory and must be imported into the Rscript to run the redundancy analysis. Documentation of the RDA can be accessed here.

RDA and pRDA output plots can be viewed here.