Quantitative Genetics & QST-FST comparison

Antoine Fraimout

I - DESCRIPTIVE ANALYSES: PHENOTYPES

Nine-spined sticklebacks (Pungitius pungitius) are small teleost fish distributed in the circumpolar regions of Eurasia and North-America. In Northern-Europe they inhabit marine habitats in the Baltic Sea and the White Sea. They can also be found in landlocked freshwater ponds where they have been isolated for thousands of generations. The dataset you are about to analyse contains morphological measurements from both types of habitats (marine and freshwater) and a total of 4 populations (2 marines, 2 freshwater).

I - 1. Data preparation

First, let's load some useful plotting/data management R packages

```r
library(tidyverse)
library(ggfortify)
```

Let's read in the data

```r
data = read.table("phenotypes.txt", header=T, sep="\t")
```

You have now loaded the raw data. In R, it is good to make sure your file is "clean" and that all columns are in the proper format. Let's reshape the "data" object into a cleaner dataset, this will also allow to rename some columns with more meaningful names.

```r
morpho = data.frame(animal = factor(data$id),
                    sex = factor(data$sex),
                    pop = factor(data$pop),
                    habitat = factor(data$habitat),
                    size = data$SL,
                    weight = data$BW,
                    jaw.length = data$JL,
                    head.depth = data$HD,
                    head.length = data$HL)
```

You now have the "morpho" object which has the same structure than the "data" file (92 obs. of 9 variables). You can verify that everything is ok with this object by checking the levels of the different factors in your data. For instance, you know that you have data for 92 individuals from 2 habitats:

```r
table(morpho$habitat)
```

```
##
## marine   pond
##     45     47
```

Indeed both habitats are present (marine & pond) and you have 92 individuals. You can look in details the number of individuals per population also with the "table" function:

```
table(morpho$pop)
```

```
##
## BYN HKI PYO WHS
##  22  21  25  24
```

Each population has a 3 letters code corresponding to the sampling location: BYN = Bynästjärnen (Freshwater pond, Sweden), HKI = Helsinki (Baltic Sea, Finland), PYO = Pyöreälampi (Freshwater pond, Finland), WHS = White Sea (Russia). Everything looks good so you can remove the "data" object using the "rm" function

```
rm(data)
```

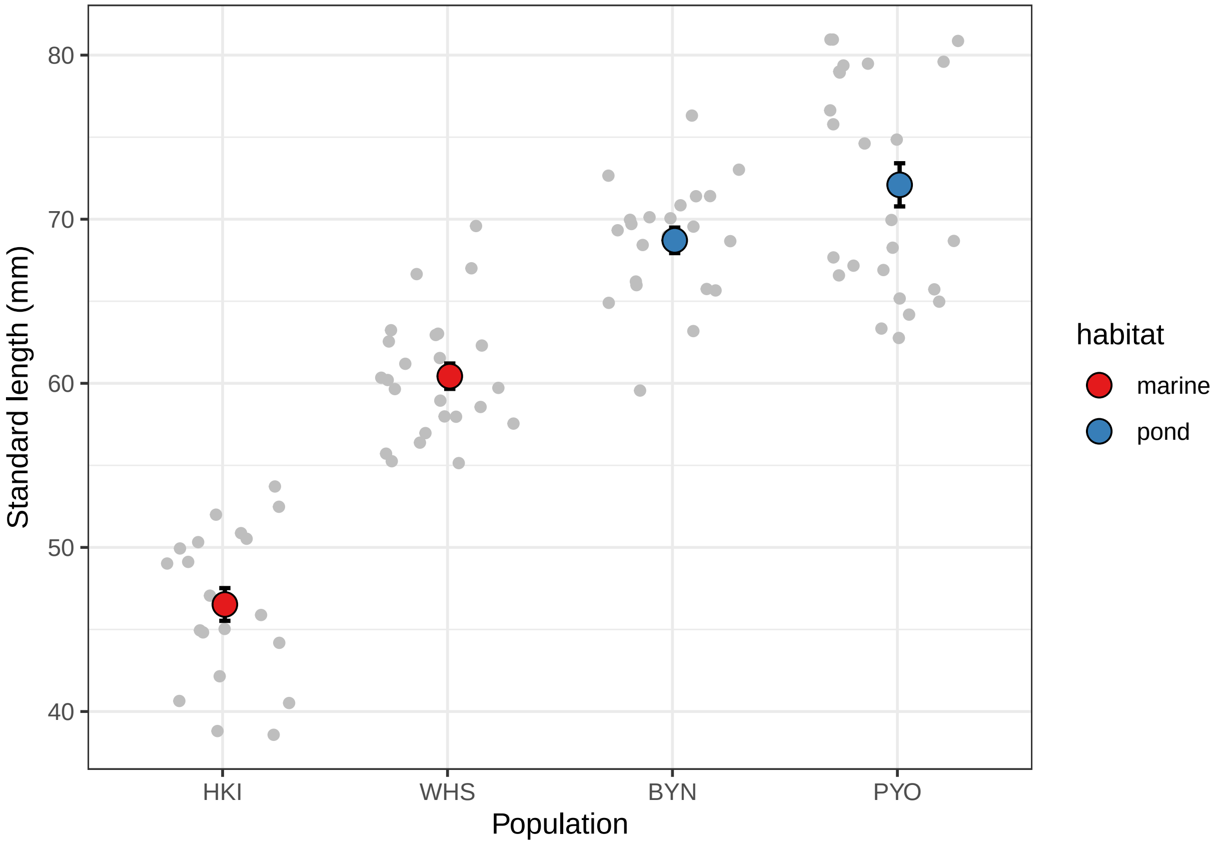
I - 2. Plotting the data

We will use functions from ggplot2 and dplyr packages (included in tidyverse) to plot the data. First let's write a template code for plotting that you will be able to modify:

```
  ggplot(data=morpho, # here you specify the dataset you use
         aes(x = pop, # specify the grouping factor on the x axis
             y = size,# specify the trait to plot on the y axis
             fill = habitat))+# color the points according to habitat of origin

  # Below are the more "aesthetics" arguments that let you custom your plot, they are not
  # very important for today's exercise but feel free to play around with it and try!

  geom_point(position = position_jitter(width = 0.3),
             colour = "grey") + # this adds individual data points around the mean
  stat_summary(fun.data = mean_se, # you indicate here that you are plotting standard
                                   # errors around the mean
               geom = "errorbar", # this adds the "whiskers" around the mean points
               size = 0.75, # size of the whiskers
               width = 0.05, # width of the whiskers
               color = "black", position = position_nudge(x = 0.01))+ # color and placement
  stat_summary(fun = mean, # you indicate here that you are plotting the mean of
                           # each grouping factor
               geom = "point", # means will be represented by points
               size = 4, # size of the points
               color = "black", # stroke color of the points
               position = position_nudge(x = 0.01), pch = 21)+ # position and shape of points
  scale_fill_brewer(palette = "Set1") + # a set of predefined colors (here blue and red)
  theme_bw() + # sets the background to white
  xlab("Population") + # Give a legend to the x axis
  ylab(expression(paste("Standard length (mm)", sep = "")))+ # Gives a legend to the y axis
  scale_x_discrete(limits=c("HKI","WHS","BYN","PYO")) # Lets you decide the order
```



```
                                                         # of the populations on the x axis
```

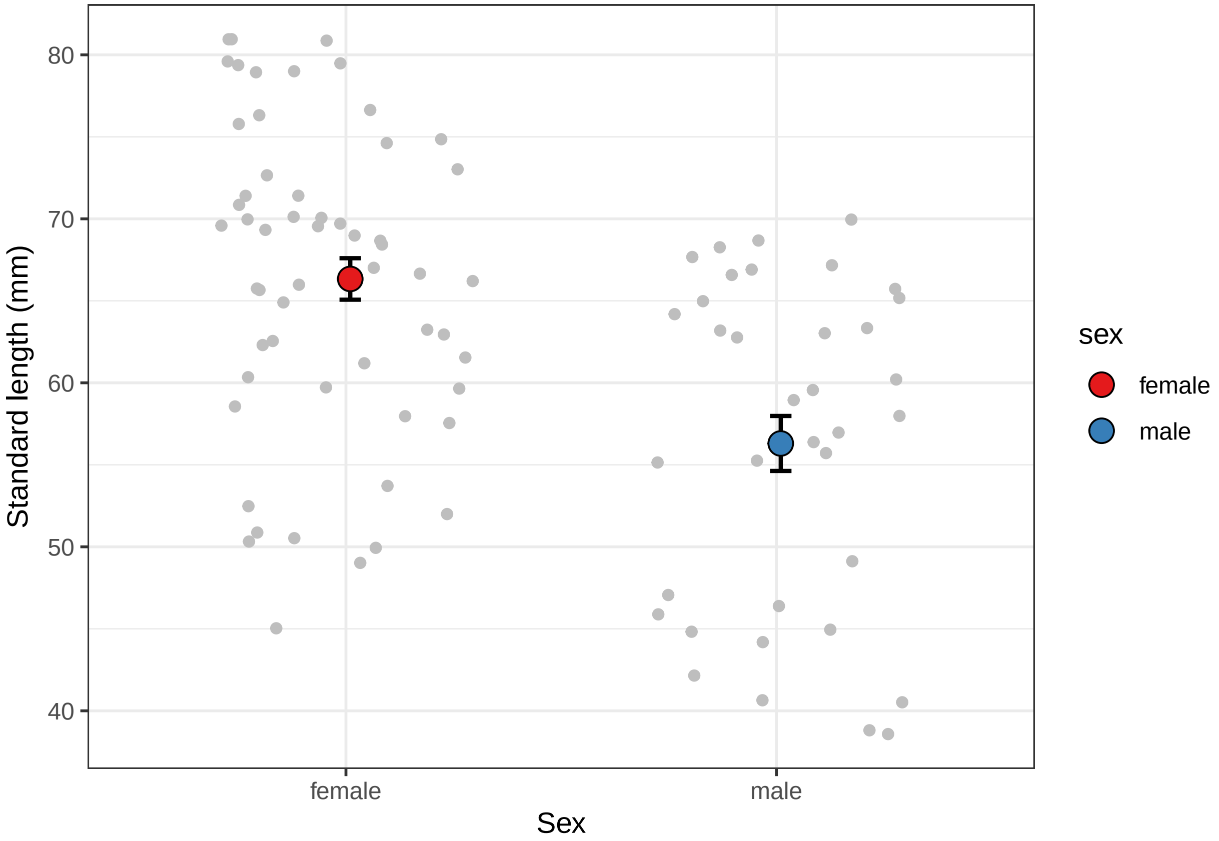
Now let's keep on looking at body size variation but with another grouping factor. Let's look at the differences in size between males and females.

```
ggplot(data=morpho,
       aes(x = sex, # grouping factor is now sex
           y = size,
           fill = sex))+

# Below are the more "aesthetics" arguments that let you custom your plot, they are not
# very important for today's exercise but feel free to play around with it and try!

geom_point(position = position_jitter(width = 0.3),
           colour = "grey") +
stat_summary(fun.data = mean_se,
             geom = "errorbar",
             size = 0.75,
             width = 0.05,
             color = "black", position = position_nudge(x = 0.01))+
stat_summary(fun = mean,
             geom = "point",
             size = 4,
             color = "black",
             position = position_nudge(x = 0.01), pch = 21)+
scale_fill_brewer(palette = "Set1") +
theme_bw() +
xlab("Sex") + # Remember to change the legend
```

```
  ylab(expression(paste("Standard length (mm)", sep = "")))+
  scale_x_discrete(limits=c("female","male")) # names must be the same as in the dataset
```



Exercise 1:

Investigate the data by plotting different traits and using different grouping factors. Describe the data: can you summarize the main phenotypic differentiation between marine and pond sticklebacks? Bonus question: What would you do to confirm the differences you observe in the plots?

II - DESCRIPTIVE ANALYSES: GENOTYPES

All the individuals from the dataset have been genotyped at 12 microsatellite loci. This allows to estimate the neutral genetic differentiation between them, as measured by FST. We will use the familiar hierfstat and adegenet package to perform this set of descriptive analyses. Let's load the packages first:

```
library(hierfstat)
library(adegenet)
```

Then, read in the genotype data:

```
geno = read.genepop("genotypes.gen", ncode=3L)
```

```
##
##  Converting data from a Genepop .gen file to a genind object...
##
```

```
##
## File description:  Title: "9spines microsat"
##
## ...done.
```

We have now two datasets in our R environment: the "morpho" object corresponding to the phenotype data and the "geno" object corresponding to the genotype data. You can get information about the new "geno" dataset by using the "summary()" function:

```
summary(geno)
```

```
##
## // Number of individuals: 183
## // Group sizes: 40 40 40 63
## // Number of alleles per locus: 7 51 2 7 14 7 15 6 6 5 6 15
## // Number of alleles per group: 91 114 21 14
## // Percentage of missing data: 1.05 %
## // Observed heterozygosity: 0.36 0.52 0.01 0.18 0.25 0.24 0.35 0.16 0.25 0.22 0.3 0.46
## // Expected heterozygosity: 0.58 0.83 0.01 0.45 0.3 0.38 0.59 0.25 0.59 0.51 0.45 0.78
```

You will notice here that we have more genotype data than phenotype data (183 vs. 92), but we still have our 4 populations of interest.

Similarly as before, let's perform descriptive analyses to observe the genetic variation in our data. We can look at the global FST first:

```
basic.stats(geno)
```

```
## $perloc
##            Ho     Hs     Ht    Dst    Htp   Dstp    Fst   Fstp     Fis   Dest
## X1125P 0.4099 0.4189 0.6343 0.2154 0.7061 0.2872 0.3396 0.4068  0.0213 0.4942
## X4174P 0.5850 0.6115 0.8756 0.2641 0.9636 0.3521 0.3016 0.3654  0.0433 0.9063
## X7033P 0.0125 0.0123 0.0125 0.0001 0.0125 0.0002 0.0104 0.0139 -0.0141 0.0002
## Stn49  0.2074 0.2012 0.4962 0.2951 0.5946 0.3934 0.5946 0.6617 -0.0309 0.4925
## Stn96  0.2837 0.2622 0.3334 0.0711 0.3571 0.0948 0.2133 0.2655 -0.0816 0.1285
## Stn100 0.2733 0.2973 0.4192 0.1219 0.4599 0.1626 0.2908 0.3535  0.0807 0.2314
## Stn130 0.3987 0.3557 0.6268 0.2711 0.7172 0.3615 0.4325 0.5040 -0.1209 0.5611
## Stn163 0.1812 0.2053 0.2836 0.0783 0.3097 0.1044 0.2761 0.3371  0.1171 0.1313
## Stn173 0.2849 0.3061 0.5851 0.2789 0.6780 0.3719 0.4767 0.5485  0.0693 0.5360
## Stn196 0.2535 0.2887 0.5540 0.2653 0.6425 0.3538 0.4789 0.5506  0.1218 0.4974
## Stn198 0.3415 0.3354 0.5039 0.1686 0.5601 0.2248 0.3345 0.4013 -0.0184 0.3382
## Stn380 0.5204 0.4890 0.8136 0.3246 0.9218 0.4328 0.3989 0.4695 -0.0640 0.8470
##
## $overall
##     Ho     Hs     Ht    Dst    Htp   Dstp    Fst   Fstp    Fis   Dest
## 0.3127 0.3153 0.5115 0.1962 0.5769 0.2616 0.3836 0.4535 0.0083 0.3821
```

Then let's look at the pairwise FST values:

```
fst <- genet.dist(geno, diploid=TRUE, method='Nei87')
fst
```

```
##              Baltic Whitesea Bynastjarnen
## Whitesea     0.1099
## Bynastjarnen 0.4440   0.4389
## Pyorealampi  0.4935   0.4731       0.8572
```

Finally, let's graphically represent the genetic variation in the data by performing a Principal Component Analysis (PCA). For this we will use the "dudi.pca" function from the adegenet package. However, this function does not allow for missing genotypes in the dataset and we know from the summary of the "geno" data that there is 1.05% of missing data. To circumvent this problem we will use the "tab()" function from the adegenet package to create a second dataset named "geno2" where missing data will be replaced by the mean genotype value.

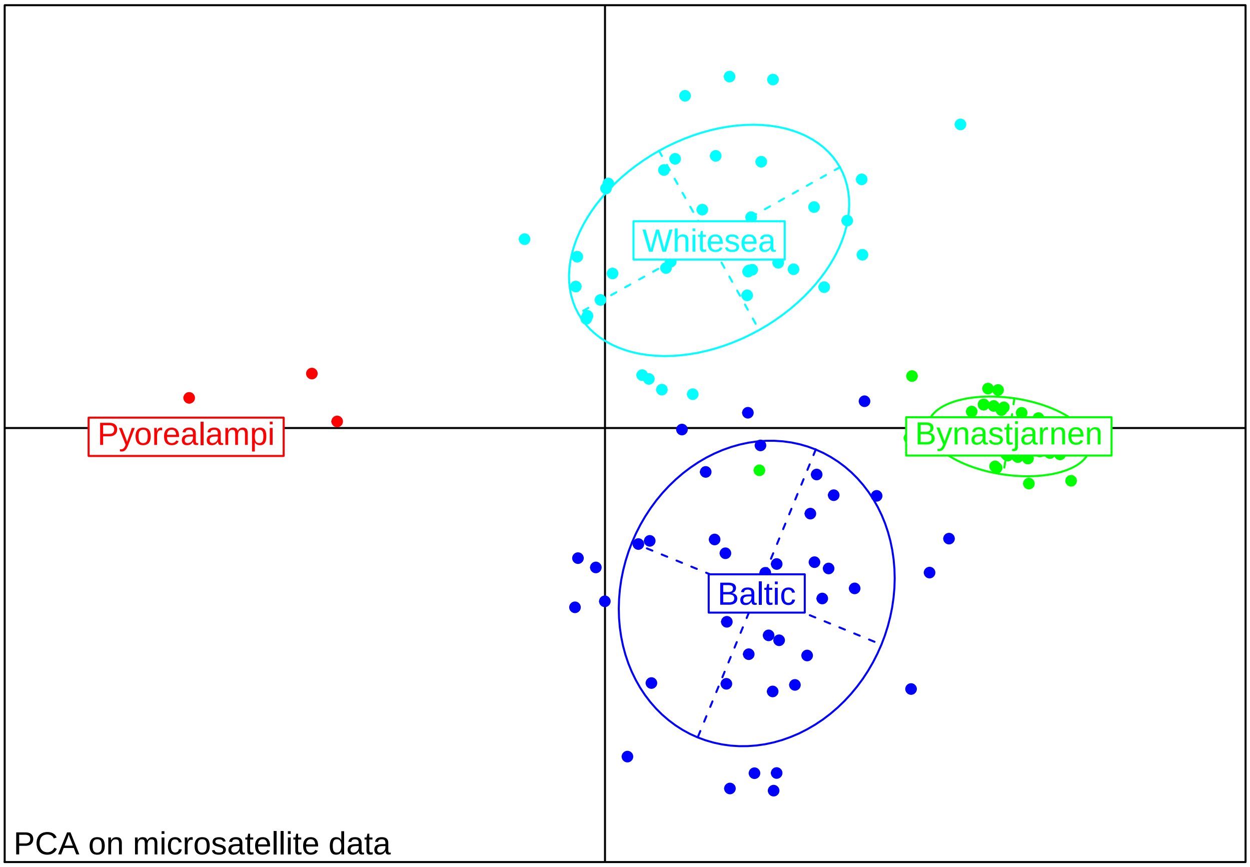
```
geno2 = tab(geno, freq = TRUE, NA.method = "mean")
```

Now we can perform the PCA with the "dudi.pca()" function:

```
pca <- dudi.pca(geno2, scale = FALSE, scannf = FALSE, nf = 3)
```

We can now plot the results of the PCA:

```
s.class(pca$li, pop(geno),col = c("blue","cyan","green","red"),xax=1,yax=3,
        cstar=0, cpoint=1, grid=FALSE,sub = "PCA on microsatellite data")
```



Exercise 2

Describe the pattern of pairwise FST: which populations are the most differentiated? which are the least differentiated? Does the differentiation between populations fits with the previous results from phenotype data? Bonus question: From the PCA plot what can you deduce from the size of the ellipses?

III - QUANTITATIVE GENETICS ANALYSES

III - 1. QST estimation

In the previous section we have estimated the global FST value which corresponds to the expected neutral differentiation among our population samples. We want to compare this value to the global QST value obtained for different traits. By doing that we will ask the question: "Does the quantitative genetic differentiation exceeds the neutral genetic differentiation?". This is the QST-FST comparison. To estimate QST we need to partition the phenotypic variance in our data into between and within-population variance as in the formula:

$Q_{ST} = \frac{\sigma^2_{GB}}{\sigma^2_{GB}+2*\sigma^2_{GW}}$

For this, we will use the MCMCglmm package:

```r
library(MCMCglmm)
```

More specifically, we will use the "animal model" approach implemented in MCMCglmm. The animal model is a class of linear mixed model that is used to decompose phenotypic variance into genetic (VG) and non-genetic components (e.g. VE), given knowledge of the relatedness among the study individuals. This will allow us to estimate the variance components we need to estimate QST. We will use the "morpho" dataset we generated earlier and we need to provide the model with information about the relatedness of our individuals. Each population sample is constituted of five families i.e. five pairs of unrelated parents and their full-sibling offspring. Information about the relatedness is stored in what is called a pedigree. Let's read in the pedigree and look at the first lines:

```r
ped = read.table("pedigree.txt", header=T, sep="\t")
head(ped)
```

```
##   animal dam sire pop
## 1      1  NA   NA HKI
## 2      2  NA   NA HKI
## 3      3  NA   NA HKI
## 4      4  NA   NA HKI
## 5      5  NA   NA HKI
## 6      6  NA   NA HKI
```

The pedigree has 4 columns: "animal" refers to the individuals (i.e. the fish), "dam" means mother and "sire" means father, "pop" refers to the population of origin. Each line of the pedigree represents an individual. In the header we can see that the first 6 lines of the pedigree have "NA" (=non availabe) in the dam and sire columns. This is means that we don't know who the parents of these individuals are. This is because they are the "founders" of our families, in other words, they are the parents of all the families. Because these individuals were sampled in the wild, we don't know who are their parents so we have to put "NA" in the dam and sire colums. But let's look now at the offpsring pedigree some lines later (e.g. from line 50 to 55)

```r
ped[50:55,]
```

```
##    animal dam sire pop
## 50   1010   3    4 HKI
## 51   1011   3    4 HKI
## 52   1012   5    6 HKI
## 53   1013   5    6 HKI
## 54   1014   5    6 HKI
## 55   1015   7    8 HKI
```

We see here that the fish called "1010" has individual "3" as dam (=as mother) and individual "4" as sire (=as father). One line below, the individual 1011 also has indivudal "3" and "4" as parents, therefore, it is a full sibling of individual 1010.

Now that we have both phenotype and relatedness data, we are almost ready to run the model. MCMCglmm uses a Bayesian statistical framework and requires the definition of so-called "prior" distribution. We will not be covering this today for the sake of simplicity, just know that it is obligatory to define these "priors" to run the model.

```
priors<-list(G=list(G1=list(V=1,n=1), G2=list(V=1,n=1)),R=list(V=1,n=1))
```

We also need to remove the "pop" column of the pedigree, otherwise MCMCglmm will provide an error message. Let's create another pedigree without the pop column.

```
ped2 = data.frame(animal = factor(ped$animal),
                  dam = factor(ped$dam),
                  sire = factor(ped$sire))
```

Now we are ready to run the model, let's investigate variance of body size and store the results in the "model.size" object (it will take about 1min30 run)

```
model.size <- MCMCglmm(size ~ sex, # size is the trait, sex is set as fixed effect
                       random = ~animal+pop, # random effects
                       nitt=1500000, burnin=500000, thin=1000, # model parameters
                       data=morpho, # the dataset
                       ped = ped2, # the pedigree
                       prior=priors, # the priors
                       verbose=FALSE)
```

This model allows us to decompose the phenotypic variance in body size into the components were are interested: "animal" is the within-population variance, and "pop" is the between-population variance. Hence, to estimate the QST value for body size, we just need to apply the formula and store the results in an object called "qst.size".

```
qst.size<-model.size$VCV[,"pop"]/(model.size$VCV[,"pop"] + 2*model.size$VCV[,"animal"])
```

If you look in the upper right panel of Rstudio you will see that the "qst.size" object is not a single value but a vector of 1000 values. This is because MCMCglmm actually estimates variance components multiple times (based on the model parameters setup in the MCMCglmm function). Therefore, instead of having a single point estimate, you get a distribution of values (called the posterior distribution). Hence, "qst.size" is now a posterior distribution of 1000 QST values. From this distribution we want to get the "average" value which we will compare to the global FST value. We also want to get the "confidence interval" which tells us about the accuracy of our QST estimation.

Let's get the point estimate value:

```
posterior.mode(qst.size)
```

```
##      var1
## 0.9777138
```

And now the confidence interval:

```
HPDinterval(qst.size)
```

```
##         lower     upper
## var1 0.655822 0.9983107
## attr(,"Probability")
## [1] 0.95
```

From this we estimate QST for body size = 0.953 and we can "be confident" that the estimated value lies between 0.681 and 0.999.

Exercise 3

Compare the QST value for body size to the global FST value estimated earlier. What do you interpret from this comparison? Optional: try running a model with a different trait, for this, just replace “size” in the model by any other traits from the dataset. Then estimate QST and compare it to the global FST value, what do you interpret?