Plotting Admixture Output

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

Admixture Proportion by Population

Read in .Q file from ADMIXTURE program:

```
#read in .Q file from ADMIXTURE program
admix_prop <- read.table("CV_reseq1kG_GWD_IBS_Hg19_autosomes.2.Q")
```

It's good to check the file and confirm it looks the way we expect:

```
head(admix_prop)
```

```
##         V1       V2
## 1 0.536686 0.463314
## 2 0.562055 0.437945
## 3 0.558277 0.441723
## 4 0.527735 0.472265
## 5 0.519003 0.480997
## 6 0.626970 0.373030
```

We need to label rows by individual ID & population. For this, we can use the fam file which should have individuals in the same order as the .Q file. The column labels come from the PLINK documentation (https://www.cog-genomics.org/plink2/formats#fam).

```
fam <- read.table("CV_reseq1kG_GWD_IBS_Hg19_autosomes.fam",
                  col.names = c("FID", "IID", "FIID", "MIID", "SexCode", "PV"))

head(fam)
```

```
##   FID     IID FIID MIID SexCode PV
## 1   1 6090455    0    0       2 -9
## 2   1 6090510    0    0       2 -9
## 3   2 6090152    0    0       2 -9
## 4   2 6090506    0    0       1 -9
## 5   3 6090468    0    0       1 -9
## 6   4 6090585    0    0       2 -9
```

Add individual ID values to admix_prop table.

```
admix_prop$IID <- fam$IID

head(admix_prop)
```

```
##         V1       V2     IID
## 1 0.536686 0.463314 6090455
## 2 0.562055 0.437945 6090510
## 3 0.558277 0.441723 6090152
## 4 0.527735 0.472265 6090506
## 5 0.519003 0.480997 6090468
## 6 0.626970 0.373030 6090585
```

Now, read in a file containing population labels for each individual ID. Your file may have different column headers than mine, but any dataframe that has a column for ID and a column for population should work.

```
population_IDs <- read.table("cv564_demo.txt", header = TRUE)

head(population_IDs)
```

```
##       IID IslandGroup
## 1 6090455   NWCluster
## 2 6090506   NWCluster
## 3 6090468   NWCluster
## 4 6090608          BV
## 5 6090649    Santiago
## 6 6090735        Fogo
```

Reorder admix_prop dataframe according to order of IIDs in population_IDs. This will exclude any IIDs that are not in the population_IDs table, and it will label rows where population_IDs has an IID not included in admix_prop as NA (i.e. extra IIDs not included in your ADMIXTURE analysis). In that case, you will have to remove any rows labeled as NA.

Then, you can easily add a column of population labels by appending the population label column from the population_IDs dataframe.

```
admix_prop <- admix_prop[match(population_IDs$IID, admix_prop$IID, nomatch = NULL),]
admix_prop$IslandGroup <- population_IDs$IslandGroup
admix_prop <- admix_prop[-which(is.na(admix_prop$IID),]
head(admix_prop)
```

```
##          V1       V2     IID IslandGroup
## 1  0.536686 0.463314 6090455   NWCluster
## 4  0.527735 0.472265 6090506   NWCluster
## 5  0.519003 0.480997 6090468   NWCluster
## 7  0.634628 0.365372 6090608          BV
## 9  0.852594 0.147406 6090649    Santiago
## 12 0.585997 0.414003 6090735        Fogo
```

This is a good point to check that the number of rows in admix_prop still matches the number of rows in fam, to confirm that you didn't lose or add any individuals.

```
nrow(admix_prop)==nrow(fam)
```

```
## [1] TRUE
```

Rename ancestry proportion columns to something more meaningful. If you're not sure which columns are which, you can check the rows corresponding to source population individuals. They should have close to 100% ancestry for one column.

```
tail(admix_prop)
```

```
##           V1       V2     IID IslandGroup
## 772 0.999990 0.000010 HG03049         GWD
## 773 0.999990 0.000010 HG03240         GWD
## 774 0.999990 0.000010 HG03247         GWD
## 775 0.997160 0.002840 HG03259         GWD
## 776 0.999990 0.000010 HG03538         GWD
## 777 0.963099 0.036901 HG03539         GWD
```

From this, we can see that the individuals from GWD population have ~100% ancestry in the first column. This means V1 is GWD ancestry, and (by elimination) V2 is IBS ancestry. But we can double check IBS rows to confirm.

```
head(admix_prop[which(admix_prop$IslandGroup=="IBS"),])
```

```
##           V1       V2     IID IslandGroup
## 564 0.004806 0.995194 HG01500         IBS
## 565 0.011165 0.988835 HG01501         IBS
## 566 0.000010 0.999990 HG01503         IBS
## 567 0.000010 0.999990 HG01504         IBS
## 568 0.000010 0.999990 HG01506         IBS
## 569 0.000010 0.999990 HG01507         IBS
```

As expected, IBS individuals have close to ~100% ancestry in column V2.

```
names(admix_prop)[1:2] <- c("GWD_Ancestry", "IBS_Ancestry")
head(admix_prop)
```

```
##    GWD_Ancestry IBS_Ancestry     IID IslandGroup
## 1      0.536686     0.463314 6090455   NWCluster
## 4      0.527735     0.472265 6090506   NWCluster
## 5      0.519003     0.480997 6090468   NWCluster
## 7      0.634628     0.365372 6090608          BV
## 9      0.852594     0.147406 6090649    Santiago
## 12     0.585997     0.414003 6090735        Fogo
```

For admixture plots, the standard is to sort individuals by ancestry proportion. In this case, I will sort in the following way:

1) Sort by population assignment
2) Sort by *decreasing* GWD ancestry

You will have more than 2 ancestries, so you may want to add a 3rd level for one of the other source population ancestries.

```
admix_prop_reordered <- admix_prop[order(admix_prop$IslandGroup, -admix_prop$GWD_Ancestry),]

head(admix_prop_reordered)
```

```
##     GWD_Ancestry IBS_Ancestry     IID IslandGroup
## 293     0.758267     0.241733 6090533          BV
## 326     0.713644     0.286356 6090589          BV
## 302     0.700034     0.299966 6090656          BV
## 178     0.660838     0.339162 6090562          BV
## 163     0.657326     0.342674 6090645          BV
## 509     0.657322     0.342678 6090650          BV
```

Reshape the dataframe to be in long format for plotting purposes. Now, there are two rows for each inividual. One for IBS ancestry proportion, and one for GWD ancestry proportion.

```
admix_prop_melt <- melt(admix_prop_reordered, id.vars = c("IID", "IslandGroup"), measure.vars = c("IBS_

head(admix_prop_melt)
```

```
##       IID IslandGroup     variable    value
## 1 6090533          BV IBS_Ancestry 0.241733
## 2 6090589          BV IBS_Ancestry 0.286356
## 3 6090656          BV IBS_Ancestry 0.299966
## 4 6090562          BV IBS_Ancestry 0.339162
## 5 6090645          BV IBS_Ancestry 0.342674
## 6 6090650          BV IBS_Ancestry 0.342678
```

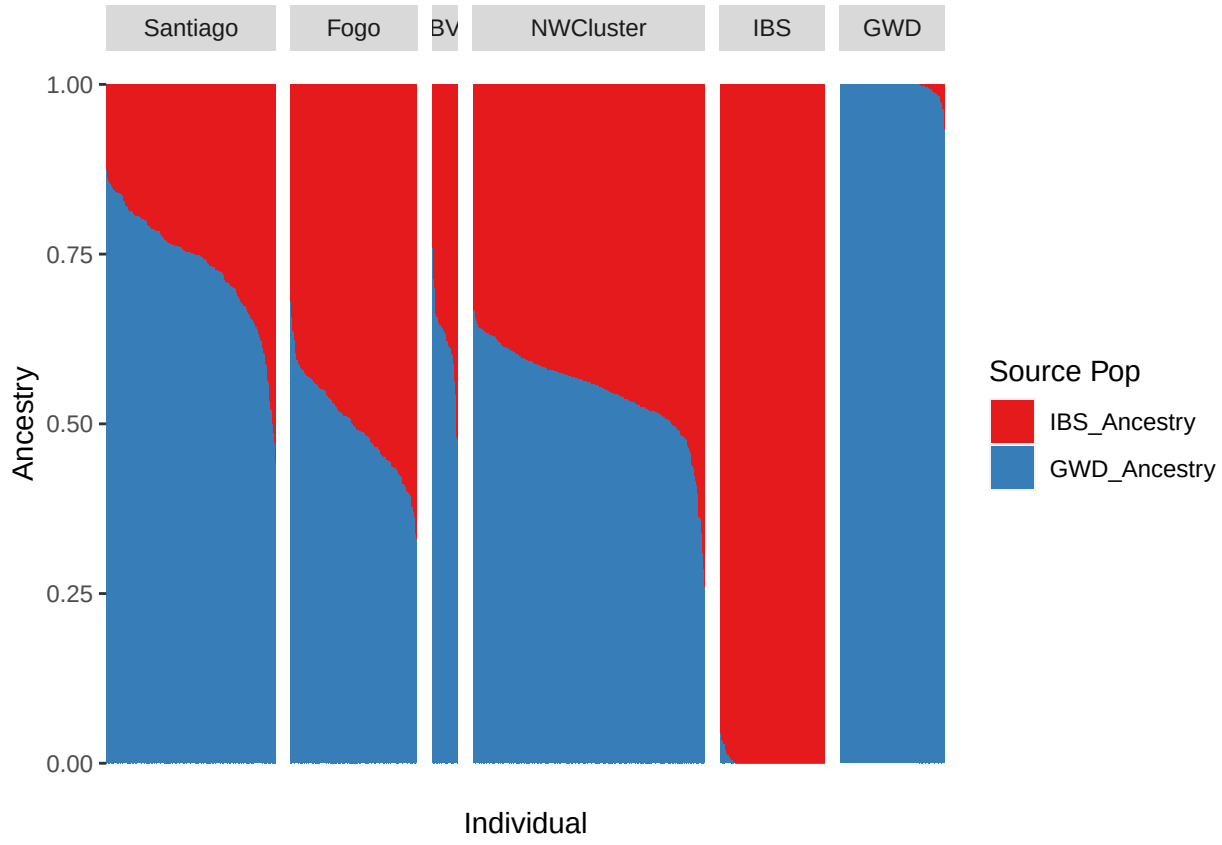
You also may want to reorder the factor levels in the order you want populations and individuals to plot.

```
admix_prop_melt$IslandGroup <- factor(admix_prop_melt$IslandGroup, levels=c('Santiago','Fogo','BV','NWC

admix_prop_melt$IID <- factor(admix_prop_melt$IID, levels=admix_prop_reordered$IID)
```

Now, you're ready to plot! We are plotting individual ancestry proportion values, faceted by island.

```
ggplot(admix_prop_melt, aes(x = IID, y = value, fill = variable)) +
  geom_bar(stat = "identity", position="stack")+
  facet_grid(. ~ IslandGroup, drop=TRUE, space="free", scales="free")+
  scale_fill_brewer(palette="Set1")+
  labs(x="Individual", y="Ancestry", fill = "Source Pop")+
  theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())
```



Calculating expected vs observed allele frequencies

First, we are calculating expected allele frequencies under neutrality (i.e. no selection). Under neutrality, we expect the allele frequency in an admixed population to be approximately equal to the source population allele frequencies weighted by their relative admixture contributions.

We can't actually know the true admixture contributions from each source population, so we use global ancestry as a proxy for this measure.

Specifically, we calculate the expected allele frequency in the admixed population a_{adm}:

$$a_{adm} = a_1 m_1 + a_2 m_2 + \cdots + a_n m_n$$

where a_n is the allele frequency in source population n, and m_n is the population n global ancestry proportion in the admixed population.

I am going to work through an example with the Duffy neg allele (rs2814778, "C"). Your results will be more complicated because you have 3 source populations to consider, so you will have to include those ancestries and allele frequencies in the calculation.

You can repeat this analysis for each allele you're interested in.

Observed frequencies

You should have this information already from your tables of allele frequencies. You should have something that looks like the following table:

```
Duffy_frequencies <- read_table("Duffy_observed_freq.txt", header=TRUE)

Duffy_frequencies
```

```
##   IslandGroup      rsID   n observed_freq
## 1          BV rs2814778  52    0.55769231
## 2        Fogo rs2814778 258    0.53875969
## 3   NWCluster rs2814778 472    0.55720339
## 4    Santiago rs2814778 344    0.83430233
## 5         IBS rs2814778 214    0.01869159
## 6         GWD rs2814778 214    1.00000000
```

Duffy_frequencies$n refers to the number of individuals from each population from which these frequencies are calculated. Duffy_frequencies$observed_freq refers to the frequency of your allele of interest in the population. For Duffy, individuals who have a "C" at the rs2814778 SNP position are protected against malaria infection. So, this is the frequency of "C" at that position in each population.

Expected frequencies

```
#group by island and calculate mean african ancestry for each island
Duffy_mean_ancestries <- admix_prop %>% group_by(IslandGroup) %>% summarise(GWD_mean_ancestry = mean(GW
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
Duffy_mean_ancestries
```

```
## # A tibble: 6 x 3
##   IslandGroup GWD_mean_ancestry IBS_mean_ancestry
##   <chr>                   <dbl>             <dbl>
## 1 BV                      0.623             0.377
## 2 Fogo                    0.498             0.502
## 3 GWD                     0.997             0.00340
## 4 IBS                     0.00240           0.998
## 5 NWCluster               0.552             0.448
## 6 Santiago                0.737             0.263
```

We want to combine this information with our observed allele frequencies.

```
Duffy_expected_vs_observed <- merge(x = Duffy_frequencies, y=Duffy_mean_ancestries, by = "IslandGroup",

Duffy_expected_vs_observed
```

```
##   IslandGroup      rsID   n observed_freq GWD_mean_ancestry IBS_mean_ancestry
## 1          BV rs2814778  52    0.55769231       0.623152308       0.376847692
## 2        Fogo rs2814778 258    0.53875969       0.497582760       0.502417240
## 3         GWD rs2814778 214    1.00000000       0.996596617       0.003403383
## 4         IBS rs2814778 214    0.01869159       0.002398664       0.997601336
## 5   NWCluster rs2814778 472    0.55720339       0.551648809       0.448351191
## 6    Santiago rs2814778 344    0.83430233       0.736623721       0.263376279
```

Next, we calculate expected allele frequencies. First, for so it's clear what we're doing, we can pull out the observed allele frequencies from the source populations.

```
GWD_Duffy_freq <- Duffy_expected_vs_observed[which(Duffy_expected_vs_observed$IslandGroup=="GWD"), "obs
IBS_Duffy_freq <- Duffy_expected_vs_observed[which(Duffy_expected_vs_observed$IslandGroup=="IBS"), "obs

#confirm these are what we expect
cat("GWD=",GWD_Duffy_freq, " ", "IBS=", IBS_Duffy_freq, sep = "")
```

```
## GWD=1 IBS=0.01869159
```

Recall the formula:

$$a_{adm} = a_1 m_1 + a_2 m_2 + \cdots + a_n m_n$$

```
#calculate expected allele freq
Duffy_expected_vs_observed$expected_freq <- (Duffy_expected_vs_observed$GWD_mean_ancestry*GWD_Duffy_fre

Duffy_expected_vs_observed
```

```
##   IslandGroup      rsID   n observed_freq GWD_mean_ancestry IBS_mean_ancestry
## 1          BV rs2814778  52    0.55769231       0.623152308       0.376847692
## 2        Fogo rs2814778 258    0.53875969       0.497582760       0.502417240
## 3         GWD rs2814778 214    1.00000000       0.996596617       0.003403383
## 4         IBS rs2814778 214    0.01869159       0.002398664       0.997601336
## 5   NWCluster rs2814778 472    0.55720339       0.551648809       0.448351191
## 6    Santiago rs2814778 344    0.83430233       0.736623721       0.263376279
##   expected_freq
## 1    0.63019619
## 2    0.50697374
## 3    0.99666023
## 4    0.02104542
## 5    0.56002921
## 6    0.74154664
```

In this case, expected frequency essentially matches the GWD mean global ancestry, but **that will likely not be the case for you.**

Binomial Test

We can do an exact binomial test to see whether our observed frequencies **in the admixed populations** are significantly different from our expected frequencies. We are not interested in the source populations expected frequencies for this analysis, because they are not admixed.

First, we need a count of how many individuals from each population carry the allele we are interested in. We can get that from the allele frequencies and the sample sizes.

```
Duffy_expected_vs_observed$count <- Duffy_expected_vs_observed$n * Duffy_expected_vs_observed$observed_

#just viewing a subset of columns to confirm they look right
Duffy_expected_vs_observed[, c("IslandGroup", "n", "observed_freq", "count")]
```

```
##   IslandGroup   n observed_freq count
## 1          BV  52    0.55769231    29
## 2        Fogo 258    0.53875969   139
## 3         GWD 214    1.00000000   214
## 4         IBS 214    0.01869159     4
## 5   NWCluster 472    0.55720339   263
## 6    Santiago 344    0.83430233   287
```

Now, we're ready to do an exact binomial test for each admixed population!

I've written a short function to make it easy to repeat this over multiple populations:

```
population_binom <- function(population, alternative) {
  population_row <- Duffy_expected_vs_observed[which(Duffy_expected_vs_observed$IslandGroup==population
  observed_count <- population_row$count
  expected_freq <- population_row$expected_freq
  n <- population_row$n
  binom.test(x=observed_count, n=n, p=expected_freq, alternative=alternative)
}
```

We can change the values of population and alternative to match our needs.

```
#population of Santiago:
population <- "Santiago"
alternative <- "greater"

population_binom(population, alternative)
```

```
##
##  Exact binomial test
##
## data:  observed_count and n
## number of successes = 287, number of trials = 344, p-value = 2.672e-05
## alternative hypothesis: true probability of success is greater than 0.7415466
## 95 percent confidence interval:
##  0.7977488 1.0000000
## sample estimates:
## probability of success
##              0.8343023
```

```r
#population of Fogo:
population <- "Fogo"
alternative <-"greater"

population_binom(population, alternative)
```

```
##
##  Exact binomial test
##
## data:  observed_count and n
## number of successes = 139, number of trials = 258, p-value = 0.1688
## alternative hypothesis: true probability of success is greater than 0.5069737
## 95 percent confidence interval:
##  0.4856373 1.0000000
## sample estimates:
## probability of success
##              0.5387597
```

```r
#population of Fogo:
population <- "NWCluster"
alternative <- "greater"

population_binom(population, alternative)
```

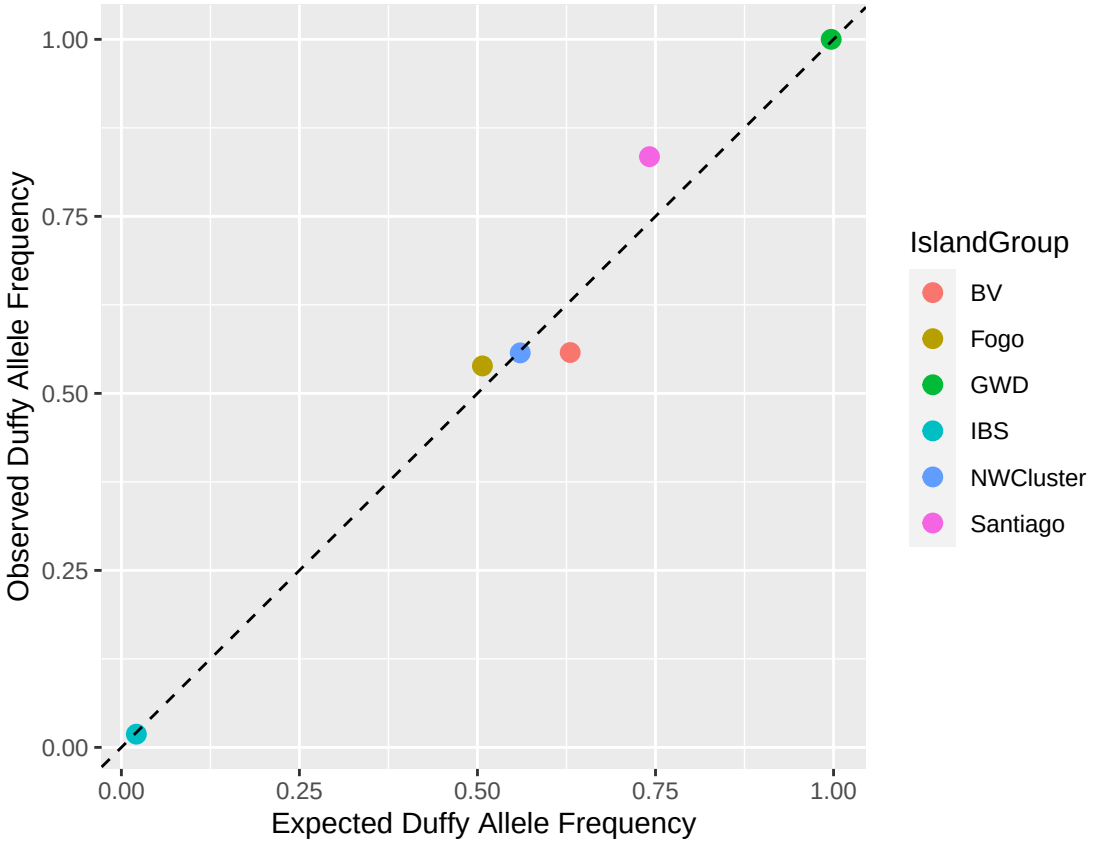
```
##
##  Exact binomial test
##
## data:  observed_count and n
## number of successes = 263, number of trials = 472, p-value = 0.5682
## alternative hypothesis: true probability of success is greater than 0.5600292
## 95 percent confidence interval:
##  0.5183478 1.0000000
## sample estimates:
## probability of success
##              0.5572034
```

You can change alternative to "two.sided", "greater", or "less" depending on what you want to test. Are you asking if the observed allele frequency is greater than (alternative="greater"), less than (alternative="less"), or simply not equal to (alternative="two.sided") the expected frequency?

Plotting Results

Next, we will want to plot the expected vs observed frequencies.

```r
ggplot(Duffy_expected_vs_observed, aes(x = expected_freq, y = observed_freq)) +
  geom_point(aes(color = IslandGroup), size = 3)+
  geom_abline(slope=1, linetype = "dashed")+
  labs(x="Expected Duffy Allele Frequency", y="Observed Duffy Allele Frequency") +
  theme(aspect.ratio = 1)
```


1.00
0.75
0.50
0.25
0.00
Observed Duffy Allele Frequency
0.00
0.25
0.50
0.75
1.00
Expected Duffy Allele Frequency
IslandGroup
BV
Fogo
GWD
IBS
NWCluster
Santiago