Task 04: Revisiting a Lab

I. General

Type up all of your work in a text editor. Basically, you should NEVER type things directly into the `R` terminal. Type them into a text editor, then either run them or copy/paste them into `R`.

Before you begin, make a new folder in Tasks called Task_04, and save an empty file named task03.r in that folder.

When you're **done** with this assignment, turn it in by (1) saving your text document, (2) opening your Terminal or GitBash, (3) navigating to the appropriate directory using `cd` and (4) typing:

`git add -A` (enter)

`git commit -m "yourname Task 04"` (enter)

`git push -u origin master` (enter)

II. Sampling

A sample is a subset of the total population that you take measurements/data from. So, if I wanted to measure the height of people in West Virginia and compare them to people in Virginia, it'd obviously be impossible for me to measure every single person in both states. So what I would do instead is take measurements from a small, randomly selected group of individuals in both states and compare the heights in my two small groups (samples).

If you've ever had trouble with this concept, come by my office! I have some code below to illustrate the effects of sampling, but this is a concept that's going to be central to almost everything we do for the rest of the semester, so even a little misunderstanding of it (if you allow that misunderstanding to persist) can be a problem.

```
# Make our populations
trueMean1 <- 5
trueSD1 <- 5
population1 <- rnorm(1e6, trueMean1, trueSD1)

trueMean2 <- 4
trueSD2 <- 5
population2 <- rnorm(1e6, trueMean2, trueSD2)

# Now take a sample of each population
Size <- 50
Sample1 <- sample(population1, Size)
Sample2 <- sample(population2, Size)
```

```
# Compare the samples! Are they different? Were the populations
    different?
boxplot(Sample1, Sample2)
```

III. Basic Genetics

Way back in BIOL 111 and 112, you learned about the basic principles undergirding inheritance and genetics in DNA-based life forms of planet Earth. In sexually reproducing organisms (like us), multiple parents (two in mammals) contribute half of their genome to the offspring. The offspring then grows up, and when it's ready, scrambles the DNA it inherited from its parents via a process called recombination, then splits its DNA in half and passes that half off to *its* offspring.

We're going to model this process in `R`. We're going to assume there are four grandparents (paternal & maternal grandma & grandpa) and two parents (Alan and Brenda). We're going to track the genes passed on from each of the four grandparents (in the `makeFounder` function, you can manipulate the number of loci with the `len` argument).

We'll model 10,000 loci from each grandparent, and follow those loci into Alan & Brenda, and then into their children.

```
# Read in the needed functions
source("http://jonsmitchell.com/code/simFxn04.R")

# First, we'll make the grandparents. I would encourage you to use head
    () and nrow() and such to examine these objects so you understand
    them.
MatGrandma <- makeFounder("grandma_mom")
MatGrandpa <- makeFounder("grandpa_mom")
PatGrandma <- makeFounder("grandma_da")
PatGrandpa <- makeFounder("grandpa_da")

# Now, we're going to have the paternal grandparents make Alan. Again,
    examine the object!
Alan <- makeBaby(PatGrandma, PatGrandpa)

# Now I would like for you to write the line of code that will make
    Brenda.

# Once you have Alan and Brenda, they'll have their first child. This
    child will be our focus.
Focus <- makeBaby(Brenda, Alan)

# Because each locus is tagged with which grandparent and parent it
    came from, we can search to see how many genes came from any one
    known ancestor of Focus. We'll use grep(), which is basically like
    Ctrl+F. It will find any matches. So grandpa_mom and grandma_mom
    both include "mom" and thus will both match to the line below. We'll
     find how many loci Focus inherited from Brenda, and then divide
    that by the total number of loci present. This gives us the percent
```

```
   of genes shared between Brenda and Focus. Before you look at the
   ToMom object, what should the number be?
ToMom <- length( grep("mom", Focus ) ) / length( Focus )

# We can also look at how many genes Focus shares with each of fhis
   maternal grandparents. What should these numbers be? Do they match
   your expectation?
ToMomMom <- length( grep( "grandma_mom", Focus ) ) / length( Focus )
ToMomDad <- length( grep( "grandpa_mom", Focus ) ) / length( Focus )

# Is Focus equally related to each maternal grandparent? What about
   each paternal grandparent? Is this what you expected? What is the
   average relatedness of Focus to all four grandparents?

# Let's imagine Focus gets a sibling. Yay, sibling!
Sibling_01 <- makeBaby(Brenda, Alan)

# How much DNA do you expect Focus to share with Sibling_01? Is that
   the amount actually shared?
ToSib <- length( intersect( Focus, Sibling_01 ) ) / length( Focus )

# Now, let's imagine Brenda and Alan are extremely busy, and have 1,000
    more children. How many genes does Focus share with each of the
   1,000 siblings?
ManySiblings <- replicate( 1e3, length( intersect( Focus, makeBaby(
   Brenda, Alan) ) ) / length( Focus ) )

# We can summarize the data using quantiles and the mean
quantile(ManySiblings)
mean(ManySiblings)

# And we can also plot the data
plot(density(ManySiblings), main="", xlab="proportion shared genes")

# Please provide an explanation for why you see a range of values in
   these analyses. Note that you can adjust the number of loci by
   changing the "len" argument in the makeFounder() function calls
   above.
```

IV. Hardy-Weinberg Equilibrium

```
# Given an allele frequency, p, we can calculate the expected genotype
   frequencies
HWE <- function(p)  {
  aa <- p^2
  ab <- 2 * p * (1 - p)
  bb <- (1 - p)^2
```

```
  return(c(aa=aa, ab=ab, bb=bb))
}
HWE(0.5)

# We can make a blank plot...
plot(1, 1, type="n", xlim=c(0, 1), ylim=c(0, 1), xlab="freq. allele a",
    ylab="geno. freq")

# ...then calculate genotype frequencies for a bunch of allele
   frequencies
p <- seq(from = 0, to = 1, by = 0.01)
GenoFreq <- t(sapply(p, HWE))

# Now we can plot our known allele frequency (p) against our expected
   genotype frequencies (GenoFreq)
lines(p, GenoFreq[,"aa"], lwd=2, col="red")

# Before moving on, can you read and understand this plot? What happens
    to the frequency of aa individuals as the frequency of the a allele
    increases in the population? What happens to the aa frequency as a
   decreases? Is time shown on this plot? Is geographic space? If you
   want to do well in this class it is vital that you know exactly what
    is (and is not) shown on a graph like this.

# Now, let's add the other genotypes
lines(p, GenoFreq[,"ab"], lwd=2, col="purple")
lines(p, GenoFreq[,"bb"], lwd=2, col="blue")
legend("top", legend=c("aa","ab","bb"), col=c("red","purple","blue"),
   lty=1, lwd=2, bty="n")

# Let's use this now! Let's simulate a population, and look at how its
   allele and genotype frequencies vary.
Pop <- simPop(500)

# Now let's add these points to the HWE plot you made above
points(Pop[,"freqa"], Pop[,"Genotypes.aa"]/500, pch=21, bg="red")

# Does the frequency of the aa genotype in your population match the
   expectation from Hardy-Weinberg?

# What if we do another simulation with a much smaller population
Pop <- simPop(50)
points(Pop[,"freqa"], Pop[,"Genotypes.aa"]/50, pch=22, bg="red")

# What's changed and why?
```

V. Two-Allele Drift

```
# install the learnPopGen package
library(learnPopGen)

# Run this over and over again, watching the lines.
# Ne is how many individuals there are in each population
# nrep is how many populations you are simulating at once
# pause is just how fast the lines grow
x <- genetic.drift(Ne=200, nrep=5, pause=0.01)

# Run and rerun the line above, changing Ne, and noting the patterns
    you see

# Let's look at how population sizes effect the time to extinction for
    one allele
# First, we'll make a bunch of populations of different sizes, from 5
    to 50 individuals
PopSizes <- 5:50

# Next, we'll say that there are 5 populations with each given size
Samples <- rep(PopSizes, 5)

# Now, we'll simulate all 230 of those populations & get the time one
    of the two alleles went extinct
# Note: this will take some time (~20 sec on my computer, maybe longer
    or shorter depending on your machine)
tExt <- sapply(Samples, function(x) nrow(simPop(x, 500)))

# To fit a line (Linear Model) to data in R you use the lm() function.
Line <- lm(tExt ~ Samples)

# To see the fit, use summary()
summary(Line)

# To extract the coefficients, use $coef
Line$coef

# To add it to the plot, use abline()
plot(Samples, tExt)
abline(Line)

# You may consider comparing the above to Line2 <- lm( tExt~Samples + 0
    ), and ask yourself which one makes more sense given these data (
    note: run them both and look at the output to figure out what the +0
     does!)

# Linear models assume that all points are roughly equally far from the
     line, and that there are as many above the line as below. What do
    you notice, looking at this graph, about the distance between the
```

```
    points and the line as the population size increases? What does that
     mean to you?
```

VI. Extra Credit

The issue identified in the last question has a name: heteroskedasticity. Use your Google skills to find a way to fit a linear model that is *robust* to heteroskedasticity and compare the estimate of the slope.