About Me

Ellen Campbell

Contents

Who I am and where I came from

I grew up in a small farm town near UC Davis. When I was about three I watched a PBS Nature program called "Incredible Suckers" on cephalopods and decided then and there that I wanted to study squid when I grew up. As I learned more, my interests broadened to a general love of math and science (although I still love squid!). I attended UC Santa Cruz, and graduated in 2014 with a B.S in biology and a minor in applied math and statistics. I joined Carlos Garza's lab as one of his three full time lab staff immediately after I graduated and have been working there ever since.

When I'm not working, I love to:

1. Bake whatever interesting new food experiments I can find
2. Brew beers, ciders, and mead with Akshar
3. Go hiking!
4. Knit

Here's a picture of my adorable cat, since apparently the only photos I have on my computer right now are of her

Research Interests

I really enjoy the more trouble-shooting/practical work that we do in the lab. I'm less invested in a particular topic, and generally enjoy the more specific questions we approach in lab, like why a specific protocol has stopped working, or how do we assess the quality of genotypes for types of genotyping data that we haven't generated before. I've really enjoyed being part of the lab's transition towards sequencing-based genotyping. I've loved getting to handle GTseq data and the sneak peaks I got into the development of the snakemake pipeline and am really excited about our current dive into whole genome sequencing runs!

Influential papers

Diving back into more GTseq (Campbell, Harmon, and Narum 2014) stuff in lab work this week. I need to get back into Coho VCF stuff. Diana demonstrated with the work in her paper Baetscher et al. (2017) that microhaplotype loci have higher power for relationship inference. We have an existing VCF that's been working really well for relationship inference (as far as I've hear from Libby), but we're trying to add additional variants so it can serve as species ID between salmonids as well.

The mathematics behind my research

This is a perpetually useful formula:

$$C_1V_1 = C_2V_2$$

Write another one here if time...

My computing experience

I've been using R since my undergrad days, but was only introduced to the tidyverse world of R after I graduated. I find it immensely useful for stripping through data and checking for common issues (assessing sample/genotype quality; assessing assay/reagent success) and less common issues (like the whole 9D debocle).

Here's an ugly loopy chunk of R code that I want to rewrite in a tidier format, but every time I try, it slows things down!

```
#This input has a sample name column with a unique ID--lower the ID, the earlier the sample was run
#I'll be arranging by sample name so that for each rerun, the earlier run comes first.  You could arran
genotypes <- as.data.frame(Together)
#Create a vector containing each of the sample names only once
samples <- as.vector(unique(genotypes[ , 1]))
#Create a matrix to hold all of the consensus genotypes and the sample names
consensus <- matrix(nrow = length(samples), ncol = (length(genotypes[1, ])+1)/2)
#set the first column to be the sample names
consensus[ , 1] <- samples
#Start looping over samples
for (i in c(1:length(samples))){
  #Find the indicies for the two runs of sample i
  indicies <- c(1:length(genotypes[,1]))[genotypes[ , 1] == samples[i]]
  #Start looping over assays (increment by 2 as there are two columns per assay)
  for (j in seq.int(3, length(genotypes[1, ]), 2)){
    #We'll be comparing sums of the two columns to make things easy, so calculate the sums
    a <- genotypes[indicies[1], j]+genotypes[indicies[1], j+1]
    b <- genotypes[indicies[2], j]+genotypes[indicies[2], j+1]
    #If the two runs result in the same genotype note as "S" (same), if mismatched note as
    #"M" (mismatch), if one nocalled but other called note as "L" (loss)
    if (a==b){
      consensus[i,((j/2)+1)] <- "S"
    } else if (a == 0){
      consensus[i,((j/2)+1)] <- "C"
    } else if (b == 0){
      consensus[i,((j/2)+1)] <- "L"
    } else if (genotypes[indicies[1], j]==genotypes[indicies[1], j+1]){
      if (genotypes[indicies[2], j]==genotypes[indicies[2], j+1]){
        consensus[i,((j/2)+1)] <- "XXtoYY"
      } else {
        consensus[i,((j/2)+1)] <- "XXtoXY"
      }
    } else if (genotypes[indicies[2], j]==genotypes[indicies[2], j+1]){
      consensus[i,((j/2)+1)] <- "XYtoXX"
    } else if (sum(genotypes[indicies[1], c(j, j+1)] %in% genotypes[indicies[2], c(j, j+1)]) > 0){
      consensus[i,((j/2)+1)] <- "XYtoXZ"
    } else {
      consensus[i,((j/2)+1)] <- "XYtoWZ"
    }
  }
}
```

```
#Store the consensus score matrix as a data frame
consensus <- as.data.frame(consensus)
#Set the names of the columns to be the same names as the assays
names(consensus) <- names(genotypes)[c(1, seq(3, length(genotypes[1, ]), 2))]
#Write to file
#write.csv(consensus, file = "Differences.csv", row.names = F)
```

I've been slowly dipping my toes into python programming over the past couple years. I was briefly introduced in my undergrad years, but only formally started learning in the past two years. I've found it's useful for automating bits of our lab process (like merging metadata files with the code I've written below to replace our RDBMerge excel plugin which, sadly, has ceased working)

```
#Let's see if we can write a script that pulls the metadata we need out of
#metadata excel files

#Pull in libraries needed
import pandas as pd
import os

#Avoid truncating long numbers by setting float format to display 12 digits
pd.options.display.float_format = "{:.12f}".format

#Find all the files in the current directory
files = sorted([file for file in os.listdir("./") if file.endswith(('.xlsm', '.xlsx', '.xls'))])

#Read in files!!

#Initialize data frame to hold merged metadata
Repository = pd.DataFrame()
Freshwater = pd.DataFrame()
Marine = pd.DataFrame()


#Loop over files
for file in files:
    #Read in repo data
    AllTabs = pd.read_excel(file, sheet_name = ["Repository","Freshwater", "Marine"], index_col = 0)
    Repository = pd.concat([Repository,AllTabs["Repository"]])
    Freshwater = pd.concat([Freshwater,AllTabs["Freshwater"]])
    Marine = pd.concat([Marine,AllTabs["Marine"]])


writer = pd.ExcelWriter("Merged.xlsx", engine = 'openpyxl')
Repository.to_excel(writer, sheet_name = 'Repository', index = True, header = True)
Freshwater.to_excel(writer, sheet_name = 'Freshwater', index = True, header = True)
Marine.to_excel(writer, sheet_name = 'Marine', index = True, header = True)
writer.save()
writer.close()
```

I've been intermittently working on a python script that I'm hoping will eventually replace the microsatellite toolkit plugin in excel, which I'm worried may not work on our computers for much longer. I've currently stalled on reformatting genotype data into genepop format since apparently the original mstoolkit did not do this entirely correctly, but I'll try to pick this up again the next time I have a lull in lab work.

What I hope to get out of this class

Three things I'd like to get out of this class

- Hone my R and UNIX skills. I've used both enough now that I feel comfortable working in both settings, but I know there's so much more they can do that I just don't know of yet so excited to learn more things about what I can do with them!
- Become more comfortable on computing clusters and with SLURM. This was probably the part of the last class that I struggled with the most, and I haven't used it much since then, so hoping I can become more comfortable just working in that environment.
- Get motivated to tackle new problems! Last class definitely motivated me to find new ways of tackling new problems (like how do we generate a reference VCF for coho in an organized/thoughtful fashion; what bits of excel plugins are breaking and how can we rewrite them in R (which ultimately drove me to learn how to try to start fixing them with python))

Evaluating some R code

Here's some R code:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

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

```
silly <- read_delim("references.bib", delim = "\n", col_names = FALSE)
```

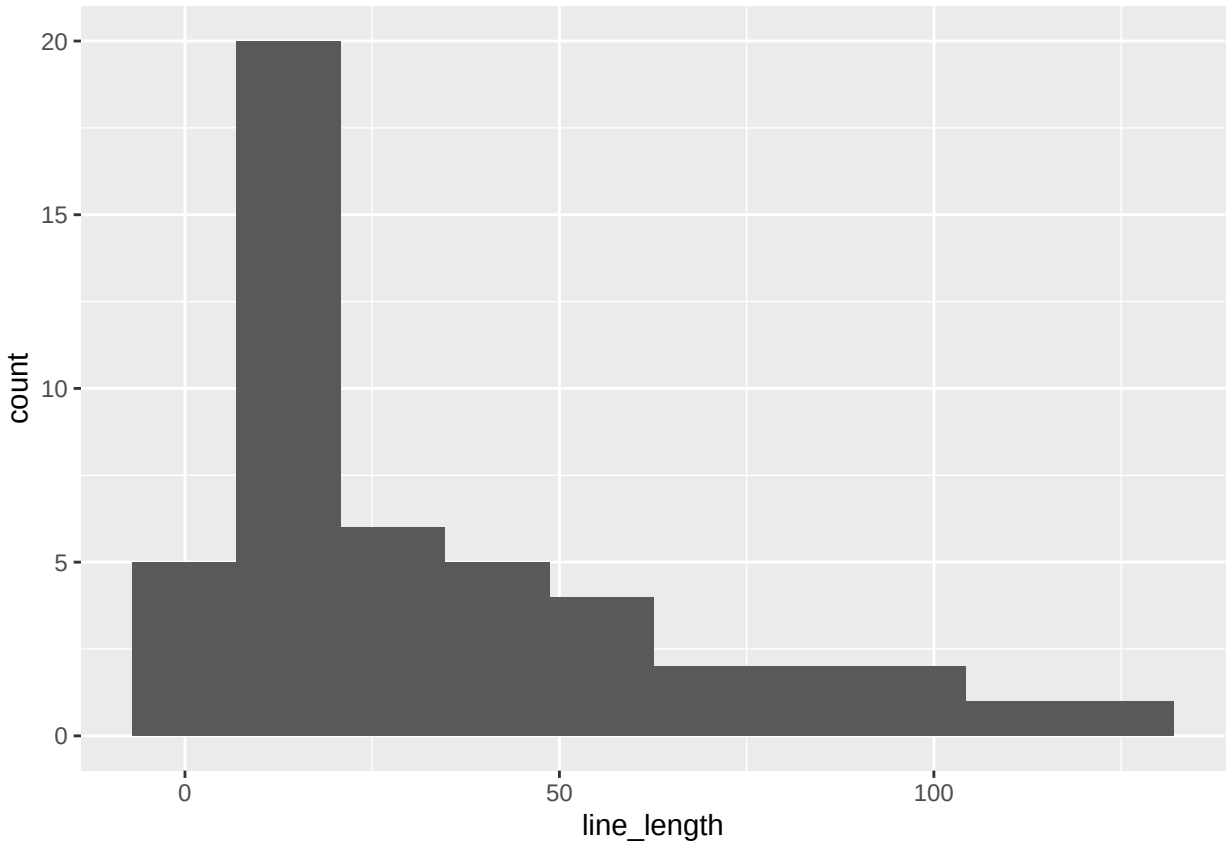
```
## Rows: 48 Columns: 1
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "\001"
## chr (1): X1
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
silly_CountByLine <- mutate(silly, "line_length" = nchar(X1))

ggplot(silly_CountByLine, aes(x = line_length)) + geom_histogram(bins = 10)
```



Citations

Baetscher, D. S., A. J. Clemento, T. C. Ng, E. C. Anderson, and John C. Garza. 2017. "Microhaplotypes Provide Increased Power from Short-read DNA Sequences for Relationship Inference." *Molecular Ecology Resources* 18 (2): 296–305.

Campbell, Nathan R, Stephanie A Harmon, and Shawn R Narum. 2014. "Genotyping-in-Thousands by Sequencing (GT-Seq): A Cost Effective SNP Genotyping Method Based on Custom Amplicon Sequencing." *Molecular Ecology Resources* 15 (4): 855–67.