BIOL-607 Biological Data Analysis Homework 3: Data Visualization

Andrew Judell-Halfpenny

September 27, 2018

Libraries Used

```r
# load the libraries and the abd dat
library(devtools)
#devtools::install_github(gganimate)
#devtools::install_github(phangorn)
#devtools::install_github(treeio)
#library(ggvis);library(phangorn)
#library(ggtree);library(treeio);

#1. load libraries
library(tidyr);library(tidygraph);library(lubridate)
```

```
##
## Attaching package: 'tidygraph'

## The following object is masked from 'package:stats':
##
##     filter

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```r
library(ggplot2);library(animation);library(readr)
library(abd);library(dplyr);library(magrittr)
```

```
## Loading required package: nlme

## Loading required package: lattice

## Loading required package: grid

## Loading required package: mosaic

## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:nlme':
##
##     collapse
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union

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

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

## Loading required package: ggformula

## Loading required package: ggstance

##
## Attaching package: 'ggstance'

## The following objects are masked from 'package:ggplot2':
##
##     geom_errorbarh, GeomErrorbarh

##
## New to ggformula?  Try the tutorials:
##  learnr::run_tutorial("introduction", package = "ggformula")
##  learnr::run_tutorial("refining", package = "ggformula")

## Loading required package: mosaicData

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':
##
##     expand

##
## The 'mosaic' package masks several functions from core packages in order to add
## additional features.  The original behavior of these functions should not be affected by this.
##
## Note: If you use the Matrix package, be sure to load it BEFORE loading mosaic.

##
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':
##
##     mean

## The following objects are masked from 'package:dplyr':
##
##     count, do, tally

## The following object is masked from 'package:ggplot2':
##
##     stat

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

```
##     binom.test, cor, cor.test, cov, fivenum, IQR, median,
##     prop.test, quantile, sd, t.test, var

## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum

##
## Attaching package: 'magrittr'

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

```
library(forcats)
```

For the first three problems, consider the data from problem 9 on page 109

1) Complete problems 10 and 17-18 on pg. 109-111. Use R where possible. Using an approximate method, provide a rough 95% confidence interval for the population mean.

```
abd.genes<-read.csv(url("http://whitlockschluter.zoology.ubc.ca/wp-content/data/chapter04/chap04e1Human
abd.genes<-as.matrix(abd.genes)

# standard error function
std.er <- function(x){
  output <-sd(x)/sqrt(length(x))
  return(output)}
abd.std.er<-std.er(abd.genes)
abd.mean<-mean(abd.genes)

# This is the standard eror variable
abd.std.er
```

```
## [1] 14.30023
```

```
# This is the sample mean
abd.mean
```

```
## [1] 2622.027
```

```
l.abs.genes=abd.mean - 2 * abd.std.er
h.abs.genes=abd.mean + 2 * abd.std.er

# This variable serves as the lower bound of the 95% confidence interval
l.abs.genes
```

```
## [1] 2593.427
```

```
# This variable serves as the upper bound of the 95% confidence interval
h.abs.genes
```

```
## [1] 2650.628
```

1 b) Provide an interpretation of the interval you calculated.

Answer:

The confidence interval that was just calculated is the range of range of values that we can be 95% certain holds the value of the true mean of the sample populatio.

The following figure is from the website of a U.S. national environmental laboratory.7 It displays sample mean concentrations, with 95% confidence intervals, of three radioactive

substances. The text accompanying the figure explained that "the first plotted mean is 2.0 ± 1.1, so there is a 95% chance that the actual result is between 0.9 and 3.1, a 2.5% chance # it is less than 0.9, and a 2.5% chance it is greater than 3.1." Is this a correct interpretation of a confidence interval? Explain.

Answer:

No this is statement belies a misunderstanding of the nature of a confidence interval. The 95% confidence interval is not integrated over the area of a probability densityy function therefor it is not possible to calculate the probability that the true population parameter mean for a subset or slice of the original field.

18 Amorphophallus johnsonii is a plant growing in West Africa, and it is better known as a "corpse-flower." Its common name comes from the fact that when it flowers, it gives off a "powerful aroma of rotting fish and faeces" (Beath 1996). The flowers smell this way because their principal pollinators are carrion beetles, who are attracted to such a smell. Beath (1996) observed the number of carrion beetles (Phaeochrous amplus) that arrive per night to flowers of this species. The data are as follows:

```r
dung_visits<-c(51, 45, 61, 76, 11, 117, 7, 132, 52, 149)
dung_visits<-as.matrix(dung_visits)
```

18 a) What is the mean and standard deviation of beetles per flower?

```r
mean_dung_vists<-mean(dung_visits)
mean_dung_vists
```

```
## [1] 70.1
```

18 b) What is the standard error of this estimate of the mean?

```r
sd(dung_visits)
```

```
## [1] 48.50074
```

```r
str(dung_visits)
```

```
##  num [1:10, 1] 51 45 61 76 11 117 7 132 52 149
```

```r
visit.st.er<-std.er(dung_visits)
```

18 c) Give an approximate 95% confidence interval of the mean. Provide lower and upper limits.

```
mean_dung_vists + 2 * visit.st.er
```

```
## [1] 100.7746
```

```
mean_dung_vists - 2 * visit.st.er
```

```
## [1] 39.42544
```

18 d) If you had been given 25 data points instead of 10, would you expect the mean to be greater, less than, or about the same as the mean of this sample?

Answer: I expect that the walue be identified as the mean could potentially change not necessarilly in a particular direction. I assume with greater sample size the calculation of the mean would be ed mean approaches the vb but the measurement closer to the paramter that is the mean of

18 e) If you had been given 25 data points instead of 10, would you have expected the standard deviation to be greater, less than, or about the same as this sample?

Answer: I expect that with aa greater sample size the estimate ofthe standard deviation wouuld be slightly different due to stochasticirty of the valculation but booth the mean and stanard deviation

area independent of the size of a sample.

18 f) If you had been given 25 data points instead of 10, would you have expected the standard error to be greater, less than, or about the same as this sample?

Answer: The standard error can be expected to decrease after resampling with a grater population. This is because with a larger sample the calculation of the true mean is more precice. The true meaan and standard deviation.

2.1) Load the data using readr and make the Month__Names column into a factor whose levels are in order of month using forcats::fct__inorder. Use levels() - a function that takes a factor vector and returns the unique levels - on the column. Are they in the right order?

```
NHice<-readr::read_csv("/home/drew/Downloads/BIOL607-Biological-Data-Analysis/week-03/NH_seaice_extent_
```

```
## Parsed with column specification:
## cols(
##   Year = col_integer(),
##   Month = col_integer(),
##   Day = col_integer(),
##   Extent = col_double(),
##   Missing = col_integer(),
##   Month_Name = col_character()
## )
```

```r
NHice$monthfactor<-factor(NHice$Month_Name,levels=month.abb, ordered=TRUE)
levels(NHice$monthfactor)
```

```
##  [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov"
## [12] "Dec"
```

```r
NHice$inorder<-fct_inorder(NHice$monthfactor, ordered=TRUE)
levels(NHice$inorder)
```

```
##  [1] "Nov" "Dec" "Feb" "Mar" "Jun" "Jul" "Sep" "Oct" "Jan" "Apr" "May"
## [12] "Aug"
```

2.2) Try fct_rev() on ice$Month_Name (I called my data frame ice when I made this). What is the order of factor levels that results? Try out fct_relevel(), and last, fct_recode() as well. Look at the help files to learn more, and in particular try out the examples. Use these to guide how you try each functino out. After trying each of these, mutate month name to get the months in the right order, from January to December. Show that it worked with levels()

```r
levels(fct_rev(NHice$Month_Name))
```

```
##  [1] "Sep" "Oct" "Nov" "May" "Mar" "Jun" "Jul" "Jan" "Feb" "Dec" "Aug"
## [12] "Apr"
```

```r
levels(fct_relevel(NHice$Month_Name))
```

```
##  [1] "Apr" "Aug" "Dec" "Feb" "Jan" "Jul" "Jun" "Mar" "May" "Nov" "Oct"
## [12] "Sep"
```

```r
levels(fct_recode(NHice$Month_Name))
```

```
##  [1] "Apr" "Aug" "Dec" "Feb" "Jan" "Jul" "Jun" "Mar" "May" "Nov" "Oct"
## [12] "Sep"
```

```r
NHice$monthfactor<-factor(NHice$Month_Name,levels=month.abb, ordered=TRUE)
levels(NHice$monthfactor)
```

```
##  [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov"
## [12] "Dec"
```

```r
head(NHice)
```

```
## # A tibble: 6 x 8
##    Year Month   Day Extent Missing Month_Name monthfactor inorder
##   <int> <int> <int>  <dbl>   <int> <chr>      <ord>       <ord>
## 1  1978    11     1   10.7       0 Nov        Nov         Nov
## 2  1978    12     1   12.7       0 Dec        Dec         Dec
## 3  1979     2     1   15.9       0 Feb        Feb         Feb
## 4  1979     3     1   16.6       0 Mar        Mar         Mar
## 5  1979     6     1   13.1       0 Jun        Jun         Jun
## 6  1979     7     1   11.6       0 Jul        Jul         Jul
```

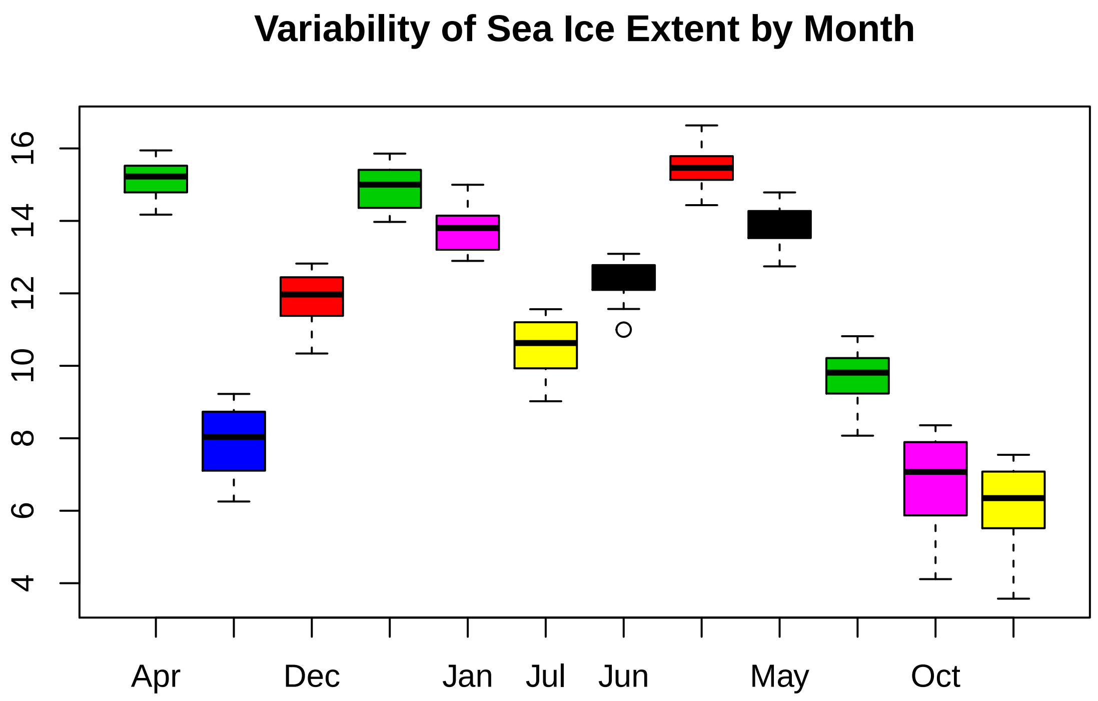
2.3) Now, using what you have just learned about forcats, make a column called Season that is a copy of Month_Name. Use the function fct_recode to turn it into a factor vector with the levels Winter, Spring, Summer, Fall in that order. Use levels() on ice$Season to show that it worked.

```
NHice$season<-fct_collapse(NHice$monthfactor,Winter=c("Dec","Jan", "Feb"),Spring=c("Mar","Apr", "May"),
levels(NHice$season)
```

```
## [1] "Winter" "Spring" "Summer" "Fall"
```

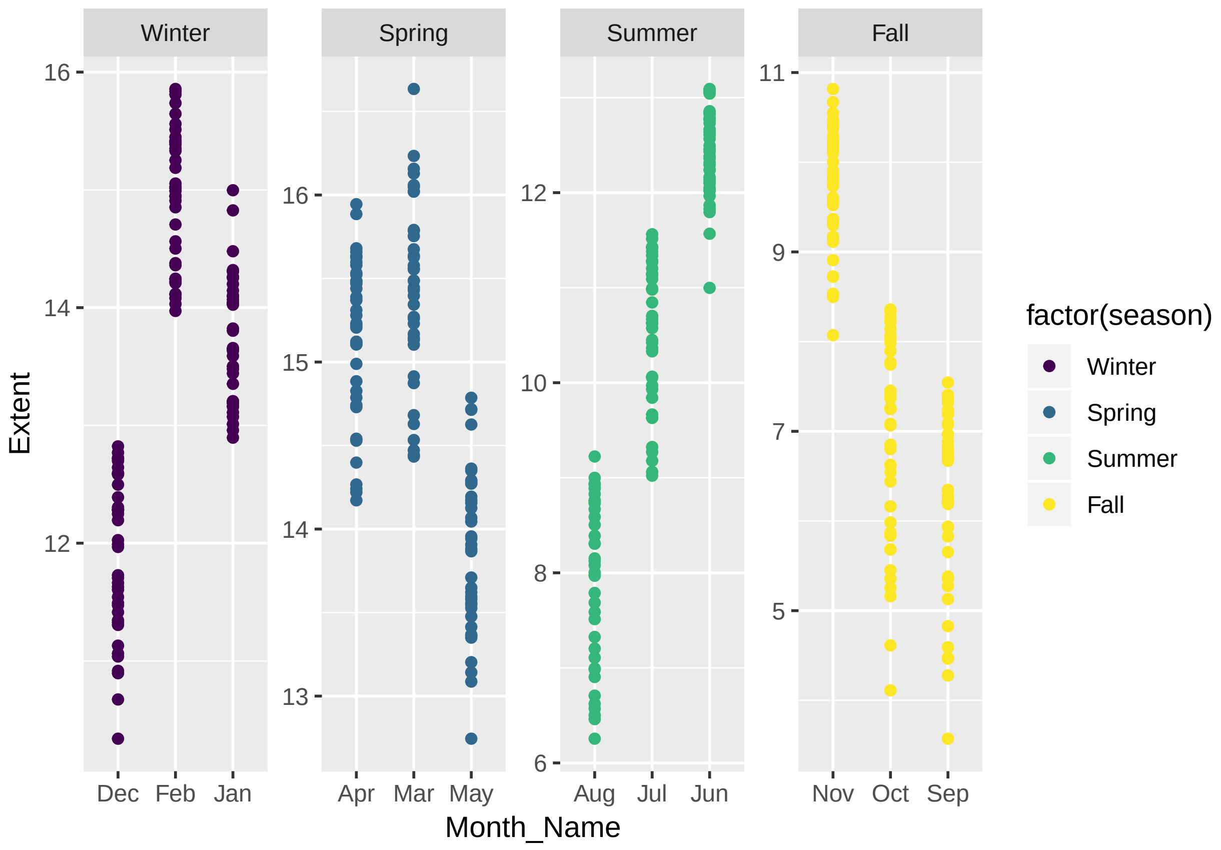
2.4) Make a boxplot showing the variability in sea ice extent every month.

```
boxplot(Extent ~ Month_Name, data = NHice, col = NHice$Month,main="Variability of Sea Ice Extent by Mon
```

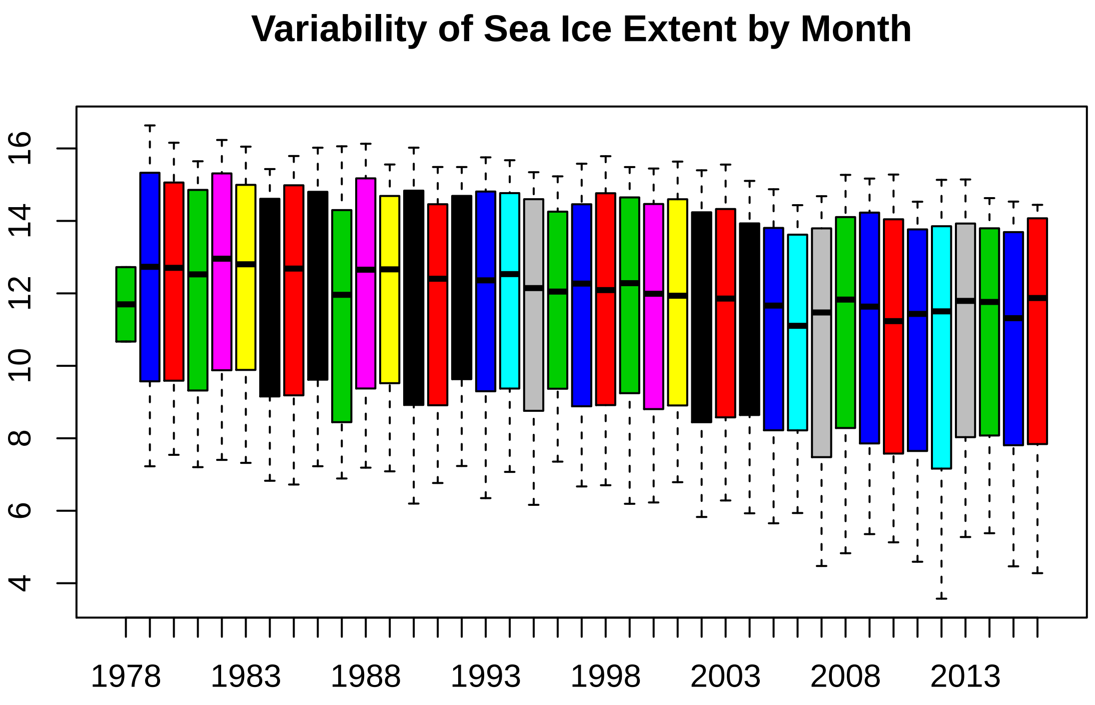


2.5) With the original data, plot sea ice by year, with different lines for different months. Then, use facet_wrap and cut_interval(Month, n=4) to split the plot into seasons.

```
ggplot(NHice, aes(Month_Name, Extent, colour = factor(season))) +
  geom_point() + facet_wrap(~ season, ncol = 4, scales = "free")
```

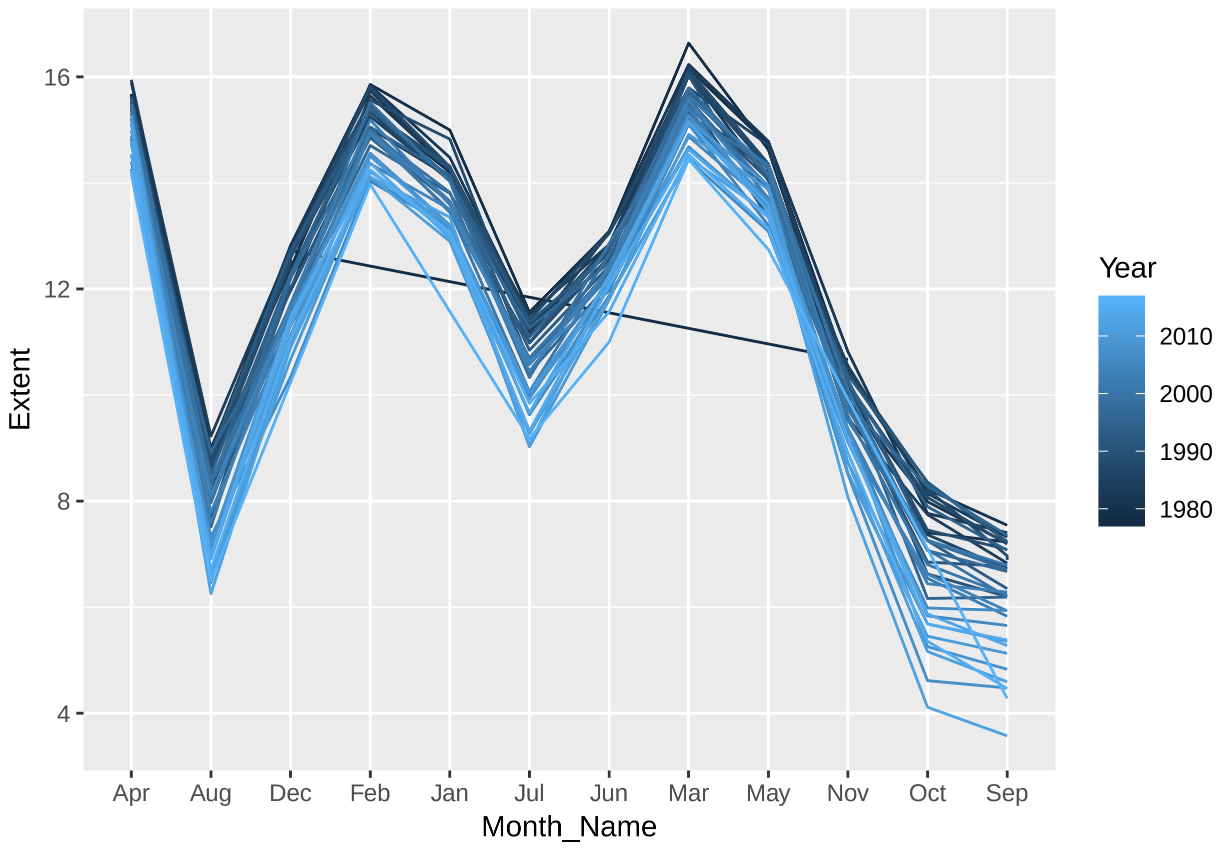


```
bp<-boxplot(Extent ~ Year, data = NHice, col = NHice$Month,main="Variability of Sea Ice Extent by Month
```



2.6) Last, make a line plot of sea ice by month with different lines as different years. Gussy it up with colors by year, a different theme, and whatever other annotations, changes to axes, etc., you think best show the story of this data. For ideas, see the lab.

```r
ggplot(data = NHice,mapping = aes(x= Month_Name, y = Extent, group = Year, col = Year)) + geom_line()
```


Extent
Apr
Aug
Dec
Feb
Jan
Jul
Jun
Mar
May
Nov
Oct
Sep
Month_Name
Year