BIO 423 - Lab 1

Benjamin Blonder

Spring 2019

Learning outcomes

- Become familiar with R/Rstudio programming environment
- Create scripts
- Load in data
- Create and manipulate data frames
- Extract summary statistics from data subsets (e.g. means)

Basics of writing scripts

```
# lines with # are 'commented out' and do not run

# we can do arithmetic
3+4
```

```
## [1] 7
```

```
# spaces don't matter
3 +     4
```

```
## [1] 7
```

Using basic functions

```
# we can evaluate simple mathematical functions
sqrt(3^2+4^2)
```

```
## [1] 5
```

```
# evaluation is inside-out, just like in regular mathematics
# all the below statements evaluate to the same value
sqrt(40+(4*6))
```

```
## [1] 8
```

```
sqrt(40+(24))
```

```
## [1] 8
```

```
sqrt(64)
```

```
## [1] 8
```

```
8
```

```
## [1] 8
```

```r
# to get help with any function, use the ? command
?sqrt

# functions have one or more named 'arguments' that take 'values'
# here the two arguments are x and digits, with values 3.982523 and 2 respectively
?round # round a number to a requested precision
round(x=3.982523,digits=2)
```

```
## [1] 3.98
```

Data types

```r
# we can work with 'vectors' of numbers
sqrt(c(4,9,16))
```

```
## [1] 2 3 4
```

```r
# we can 'assign' variable names
x = c(30,57,12)
print(x)
```

```
## [1] 30 57 12
```

```r
z = sum(x)
print(z)
```

```
## [1] 99
```

```r
# we can use 'data frames' to store tables of values
mydataframe = data.frame(firstVariable=c("a","b","c"),secondVariable=c(10,20,30))
print(mydataframe)
```

```
##   firstVariable secondVariable
## 1             a             10
## 2             b             20
## 3             c             30
```

```r
# data frame entries (and matrix entries) are accessed with either $ for columns
# or [,] for rows/columns
# the following two lines are equivalent
mydataframe$secondVariable
```

```
## [1] 10 20 30
```

```r
mydataframe[,"secondVariable"]
```

```
## [1] 10 20 30
```

Accessing and assigning variables

```r
# equals sign 'assigns' the right-side to the left-side
# double equals 'tests'
a = 3
print(a)
```

```
## [1] 3
```

```
a == 4
```

```
## [1] FALSE
```

```
a == 3
```

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

```
# we can 'index' to access elements of a vector with []
myVector = c(100, 64, 39.5, 11.1341)
print(myVector[3])
```

```
## [1] 39.5
```

```
print(myVector[4])
```

```
## [1] 11.1341
```

```
# we can also use these indexed values in other functions
sqrt(myVector[2])
```

```
## [1] 8
```

Order matters

```
# the order that lines are evaluated is very important
# R starts at the top of a file and follows instructions
# sequentially on below lines

# note the difference in output between this block...
x = 3
x = x^2
x = x+10
print(x)
```

```
## [1] 19
```

```
# and this block...
x = 3
x = x+10
x = x^2
print(x)
```

```
## [1] 169
```

Using functions

```
desert_species = data.frame(
  species=c("javelina","coati","rattlesnake","horned lizard","gila monster","black bear"),
  is.mammal=c(TRUE,TRUE,FALSE,FALSE,FALSE,TRUE),
  length.meters=c(0.5,0.4,1,0.1,0.3,2))
print(mydataframe)
```

```
##   firstVariable secondVariable
## 1             a             10
## 2             b             20
## 3             c             30
```

```
# subsetting data is possible using row indexing
desert_species_mammals = desert_species[desert_species$is.mammal==TRUE,]
print(desert_species_mammals)
```

```
##      species is.mammal length.meters
## 1   javelina      TRUE           0.5
## 2      coati      TRUE           0.4
## 6 black bear      TRUE           2.0
```

```
desert_species_nonmammals = desert_species[desert_species$is.mammal==FALSE,]
print(desert_species_nonmammals)
```

```
##         species is.mammal length.meters
## 3   rattlesnake     FALSE           1.0
## 4 horned lizard     FALSE           0.1
## 5  gila monster     FALSE           0.3
```

```
# functions can be used on different parts of a data frame
mean(desert_species_mammals$length.meters)
```

```
## [1] 0.9666667
```

```
mean(desert_species_nonmammals$length.meters)
```

```
## [1] 0.4666667
```

```
# the below block is equivalent to the above block
mean(desert_species_mammals[,"length.meters"])
```

```
## [1] 0.9666667
```

```
mean(desert_species_nonmammals[,"length.meters"])
```

```
## [1] 0.4666667
```

```
# R provides some functionality to apply functions across data subsets
# saving you the trouble of making your own subsets
?tapply
mean.lengths = tapply(X=desert_species$length.meters, INDEX=desert_species$is.mammal, FUN=mean)
print(mean.lengths)
```

```
##     FALSE      TRUE
## 0.4666667 0.9666667
```

Questions

1. Create a data frame storing the name, gender, and height (in cm) of each person in your group.
2. Calculate the mean height of each gender in your group (hint, use the `tapply` function).
3. Calculate the total height of all people in your group (hint, use the `sum` function)

Applying your skills

We will explore how insect populations are changing over time using data from a study that recently got wide media coverage. Our data come from Hallmann et al. PLoS One (2017), who studied insect biodiversity trends in German protected areas over the last several decades.

`https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0185809`

We will estimate the changes in biomass using their raw data.

Loading in the data

```
library(ggplot2) # if you get an error, install this package
# via the Tools > Install Packages menu.

# load in comma-separated-value format data
# you can verify the data format by also opening the file in Excel
data_hallmann = read.csv("journal.pone.0185809.s004.csv")

# inspect the data
str(data_hallmann)
```

```
## 'data.frame':    1512 obs. of  12 variables:
##  $ plot         : Factor w/ 63 levels "BIR1","BIS1",..: 45 45 45 45 41 41 41 41 41 41 ...
##  $ year         : int  2011 2011 2011 2011 2015 2015 2015 2015 2015 2015 ...
##  $ from         : Factor w/ 217 levels "01-05","01-06",..: 32 198 83 184 146 64 1 140 74 11 ...
##  $ to           : Factor w/ 220 levels "01-04","01-05",..: 192 77 178 71 58 210 131 68 4 140 ...
##  $ from.daynr   : int  186 209 224 238 80 100 121 140 162 183 ...
##  $ to.daynr     : int  208 223 237 253 99 120 139 161 182 201 ...
##  $ biomass      : num  80 69.5 96.5 54 2.4 24.3 32.5 59.8 96 65.7 ...
##  $ potID        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ E            : num  6.6 6.6 6.6 6.6 7.24 ...
##  $ N            : num  51.6 51.6 51.6 51.6 50.2 ...
##  $ location.type: int  2 2 2 2 3 3 3 3 3 3 ...
##  $ mean.daynr   : num  197 216 230.5 245.5 89.5 ...
```

Process the data

The data contains measurements of `biomass` in different `years` for multiple `plots`. However, the number of days of sampling at each plot is variable - sampling was carried out across different ranges of day numbers (e.g. 1=January 1st, 365=Dec 31st): from `from.daynr` to `to.daynr`. To estimate biomass trends we therefore need to standardize the biomass values to a per-day value.

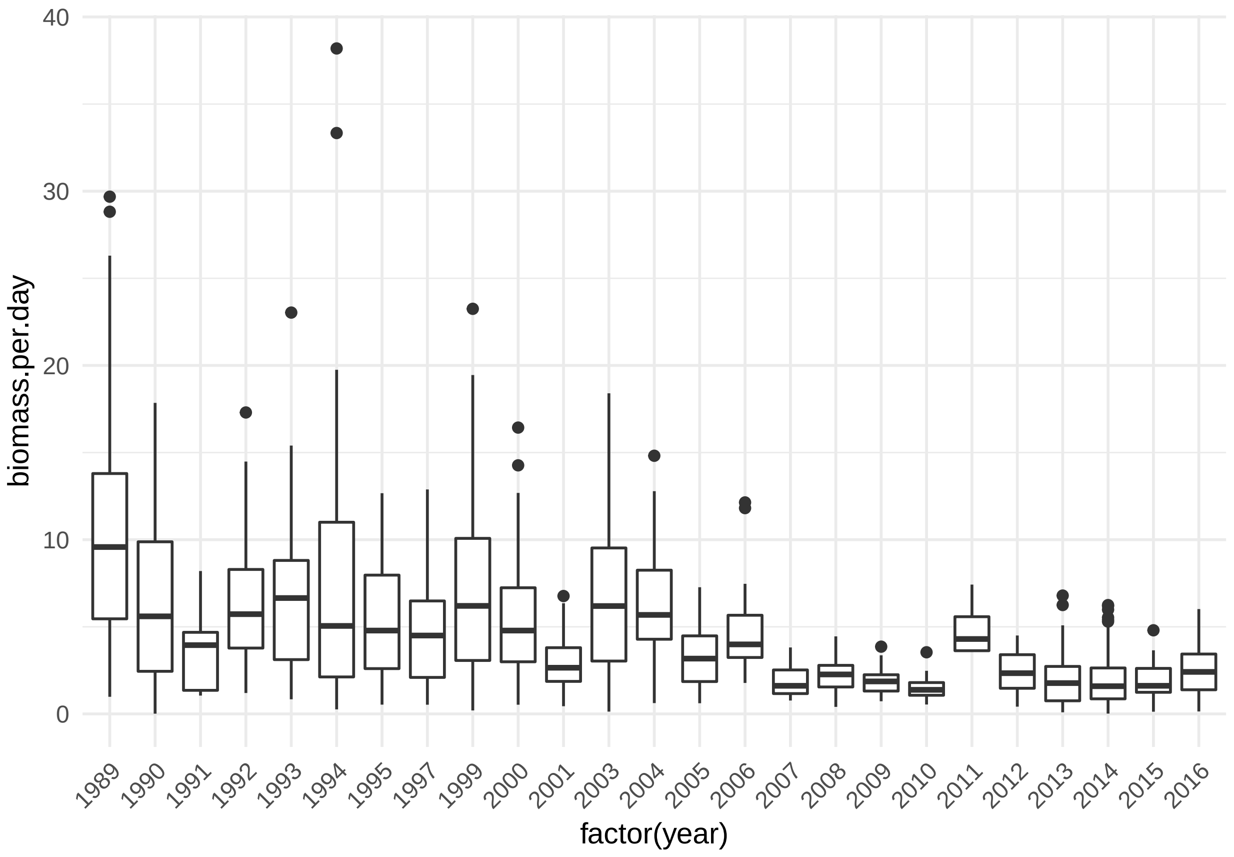
```
# create a column for biomass-standardized values
data_hallmann$biomass.per.day =
  data_hallmann$biomass / (data_hallmann$to.daynr - data_hallmann$from.daynr)
summary(data_hallmann$biomass.per.day)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
##  0.01429  1.51667  2.98333  4.52477  5.96333 38.19231        9
```

We can next visualize the trends in biomass over time. We use boxplots because there are multiple plots in each year.

```r
# set up the plot and define axes (with aes)
ggplot(data=data_hallmann, aes(x=factor(year), y=biomass.per.day)) +
# request a boxplot
  geom_boxplot() +
# set the overall style of the plot
  theme_minimal() +
# rotate the x-axis labels to look nice
  theme(axis.text.x=element_text(angle=45,hjust=1))
```

```
## Warning: Removed 9 rows containing non-finite values (stat_boxplot).
```



Questions

4. Estimate the mean `biomass.per.day` in each year. Hint: if you are getting `NA` values, you may need to tell R to remove them. You can add `, na.rm=TRUE`) to your `tapply` function call to remove the NA values in the data.
5. Calculate the percent difference in `biomass.per.day` in 2016 compared to 1989.
6. Interpret the results: is biomass increasing or decreasing over time?

Optional questions for graduate students

- Consider also fitting a linear model to the data and checking statistical significance of the trend (use `lm()`), or examining how biomass changes at different times of year (use `mean.daynr`) or across space (use latitude/longitudes in `E` and `N`). Consider correcting for spatial autocorrelation with a generalized least squares model.

- How do the results of this analysis contrast with or support those in the published Hallmann paper?

What to hand in

- A Word Document with:
  - written answers (1-2 sentences) to each question above
  - A copy of your R script (the contents of your `.R` file pasted into the Word document)
  - Author contribution statement