3. Worksheet: Basic R

Pat Wall; Z620: Quantitative Biodiversity, Indiana University

25 March, 2021

OVERVIEW

This worksheet introduces some of the basic features of the R computing environment (http://www.r-project.org). It is designed to be used along side the **3. RStudio** handout in your binder. You will not be able to complete the exercises without the corresponding handout.

Directions:

1. In the Markdown version of this document in your cloned repo, change "Student Name" on line 3 (above) with your name.
2. Complete as much of the worksheet as possible during class.
3. Use the handout as a guide; it contains a more complete description of data sets along with examples of proper scripting needed to carry out the exercises.
4. Answer questions in the worksheet. Space for your answers is provided in this document and is indicated by the ">" character. If you need a second paragraph be sure to start the first line with ">". You should notice that the answer is highlighted in green by RStudio (color may vary if you changed the editor theme).
5. Before you leave the classroom today, it is *imperative* that you **push** this file to your GitHub repo, at whatever stage you are. Ths will enable you to pull your work onto your own computer.
6. When you have completed the worksheet, **Knit** the text and code into a single PDF file by pressing the `Knit` button in the RStudio scripting panel. This will save the PDF output in your '3.RStudio' folder.
7. After Knitting, please submit the worksheet by making a **push** to your GitHub repo and then create a **pull request** via GitHub. Your pull request should include this file (**3.RStudio_Worksheet.Rmd**) with all code blocks filled out and questions answered) and the PDF output of `Knitr` (**3.RStudio_Worksheet.pdf**).

The completed exercise is due on **Wednesday, March 24th, 2021 before 12:00 PM (noon)**.

1) HOW WE WILL BE USING R AND OTHER TOOLS

You are working in an RMarkdown (.Rmd) file. This allows you to integrate text and R code into a single document. There are two major features to this document: 1) Markdown formatted text and 2) "chunks" of R code. Anything in an R code chunk will be interpreted by R when you *Knit* the document.

When you are done, you will *knit* your document together. However, if there are errors in the R code contained in your Markdown document, you will not be able to knit a PDF file. If this happens, you will need to review your code, locate the source of the error(s), and make the appropriate changes. Even if you are able to knit without issue, you should review the knitted document for correctness and completeness before you submit the Worksheet. Next to the `Knit` button in the RStudio scripting panel there is a spell checker button (`ABC`) button.

2) SETTING YOUR WORKING DIRECTORY

In the R code chunk below, please provide the code to: 1) clear your R environment, 2) print your current working directory, and 3) set your working directory to your '3.RStudio' folder.

```
rm(list = ls())
print(getwd())
```

```
## [1] "/home/patgwall/Classwork/Current/QB/2.Worksheets/3.RStudio"
```

```
setwd('~/Classwork/Current/QB/2.Worksheets/3.RStudio/')
require(extrafont)
```

```
## Loading required package: extrafont
```

```
## Warning: package 'extrafont' was built under R version 3.6.3
```

```
## Registering fonts with R
```

```
my_font = "Arial" # linux problems
```

3) USING R AS A CALCULATOR

To follow up on the pre-class exercises, please calculate the following in the R code chunk below. Feel free to reference the **1. Introduction to version control and computing tools** handout.

1) the volume of a cube with length, l, = 5 (volume = l^3)
2) the area of a circle with radius, r, = 2 (area = pi * r^2).
3) the length of the opposite side of a right-triangle given that the angle, theta, = pi/4. (radians, a.k.a. 45°) and with hypotenuse length sqrt(2) (remember: sin(theta) = opposite/hypotenuse).
4) the log (base e) of your favorite number.

```
# volume
l <- 5
vol <- l^3
print(vol)
```

```
## [1] 125
```

```
# area
r <- 2
area <- pi * r^2
print(area)
```

```
## [1] 12.56637
```

```
# triangle side
theta <- pi / 4
hypotenuse <- sqrt(2)
opp = sin(theta) * hypotenuse
print(opp)
```

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

```
# natural log
fav_num <- 2
log_num <- log(fav_num)
print(log_num)
```

```
## [1] 0.6931472
```

4) WORKING WITH VECTORS

To follow up on the pre-class exercises, please perform the requested operations in the R-code chunks below.

Basic Features Of Vectors

In the R-code chunk below, do the following: 1) Create a vector `x` consisting of any five numbers. 2) Create a new vector `w` by multiplying `x` by 14 (i.e., "scalar"). 3) Add `x` and `w` and divide by 15.

```
x <- c(1, 2, 3, 4, 5)
w <- x * 14
z <- (x + w) / 15
print(z)
```

```
## [1] 1 2 3 4 5
```

Now, do the following: 1) Create another vector (`k`) that is the same length as `w`. 2) Multiply `k` by `x`. 3) Use the combine function to create one more vector, `d` that consists of any three elements from `w` and any four elements of `k`.

```
k <- c(7, 135, 42, 99, 2)
q <- k * x
# is this right?
any_3_w <- sample(w, 3)
any_4_k <- sample(k, 4)
d <- c(any_3_w, any_4_k)
print(d)
```

```
## [1] 70 56 14  2  7 99 42
```

Summary Statistics of Vectors

In the R-code chunk below, calculate the **summary statistics** (i.e., maximum, minimum, sum, mean, median, variance, standard deviation, and standard error of the mean) for the vector (`v`) provided.

```
v <- c(16.4, 16.0, 10.1, 16.8, 20.5, NA, 20.2, 13.1, 24.8, 20.2, 25.0, 20.5, 30.5, 31.4, 27.1)

# drop na values
v_clean = v[!is.na(v)]

print(paste('Max =', max(v_clean)))
```

```
## [1] "Max = 31.4"
```

```
print(paste('Min =', min(v_clean)))
```

```
## [1] "Min = 10.1"
```

```
print(paste('Sum =', sum(v_clean)))
```

```
## [1] "Sum = 292.6"
```

```
print(paste('Mean =', mean(v_clean)))
```

```
## [1] "Mean = 20.9"
```

```
print(paste('Variance =', var(v_clean)))
```

```
## [1] "Variance = 39.44"
```

```
print(paste('Std. Dev. =', sd(v_clean)))
```

```
## [1] "Std. Dev. = 6.28012738724303"
```

```
print(paste('Std. Err. =', sd(v_clean) / sqrt(length(v_clean))))
```

```
## [1] "Std. Err. = 1.6784346448828"
```

5) WORKING WITH MATRICES

In the R-code chunk below, do the following: Using a mixture of Approach 1 and 2 from the **3. RStudio** handout, create a matrix with two columns and five rows. Both columns should consist of random numbers. Make the mean of the first column equal to 8 with a standard deviation of 2 and the mean of the second column equal to 25 with a standard deviation of 10.

```
mat_vals <- c(rnorm(5, mean = 8, sd = 2), rnorm(5, mean = 25, sd = 10))
my_mat <- matrix(mat_vals, nrow = 5, ncol = 2)
print(my_mat)
```

```
##          [,1]     [,2]
## [1,] 6.138263 16.70596
## [2,] 9.112349 23.71066
## [3,] 7.592954 31.59833
## [4,] 8.899526 29.01758
## [5,] 9.281965 11.78845
```

Question 1: What does the `rnorm` function do? What do the arguments in this function specify? Remember to use `help()` or type `?rnorm`.

> Answer 1: rnorm generates pseudorandom numbers from a normal distribution. It takes three arguments that specify the number of values to generate, the mean of the underlying distribution, and the standard deviation of the underlying standard deviation.

In the R code chunk below, do the following: 1) Load `matrix.txt` from the **3.RStudio** data folder as matrix `m`. 2) Transpose this matrix. 3) Determine the dimensions of the transposed matrix.

```
m <- as.matrix(read.table("data/matrix.txt", sep="\t", header=FALSE))
m_T = t(m)
dim(m_T)
```

```
## [1]  5 10
```

Question 2: What are the dimensions of the matrix you just transposed?

> Answer 2: After transposing the matrix the dimensions are 5×10 meaning 5 rows with 10 columns.

###Indexing a Matrix

In the R code chunk below, do the following: 1) Index matrix `m` by selecting all but the third column. 2) Remove the last row of matrix `m`.

```
print(m[, -3])
```

```
##       V1 V2 V4 V5
##  [1,]  8  1  6  1
##  [2,]  5  5  4  1
##  [3,]  2  5  3  3
##  [4,]  3  2  1  4
##  [5,]  9  9  1  2
##  [6,] 11  8  8  8
```

```
##  [7,]  2  2  8  5
##  [8,]  3  3  7  6
##  [9,]  5  5  3  6
## [10,]  6  5  2  2
```

```
print(m[1:dim(m)[1] - 1,])
```

```
##       V1 V2 V3 V4 V5
##  [1,]  8  1  7  6  1
##  [2,]  5  5  2  4  1
##  [3,]  2  5  4  3  3
##  [4,]  3  2  5  1  4
##  [5,]  9  9  1  1  2
##  [6,] 11  8  1  8  8
##  [7,]  2  2  5  8  5
##  [8,]  3  3  6  7  6
##  [9,]  5  5  1  3  6
```

6) BASIC DATA VISUALIZATION AND STATISTICAL ANALYSIS

Load Zooplankton Data Set

In the R code chunk below, do the following: 1) Load the zooplankton data set from the **3.RStudio** data folder. 2) Display the structure of this data set.

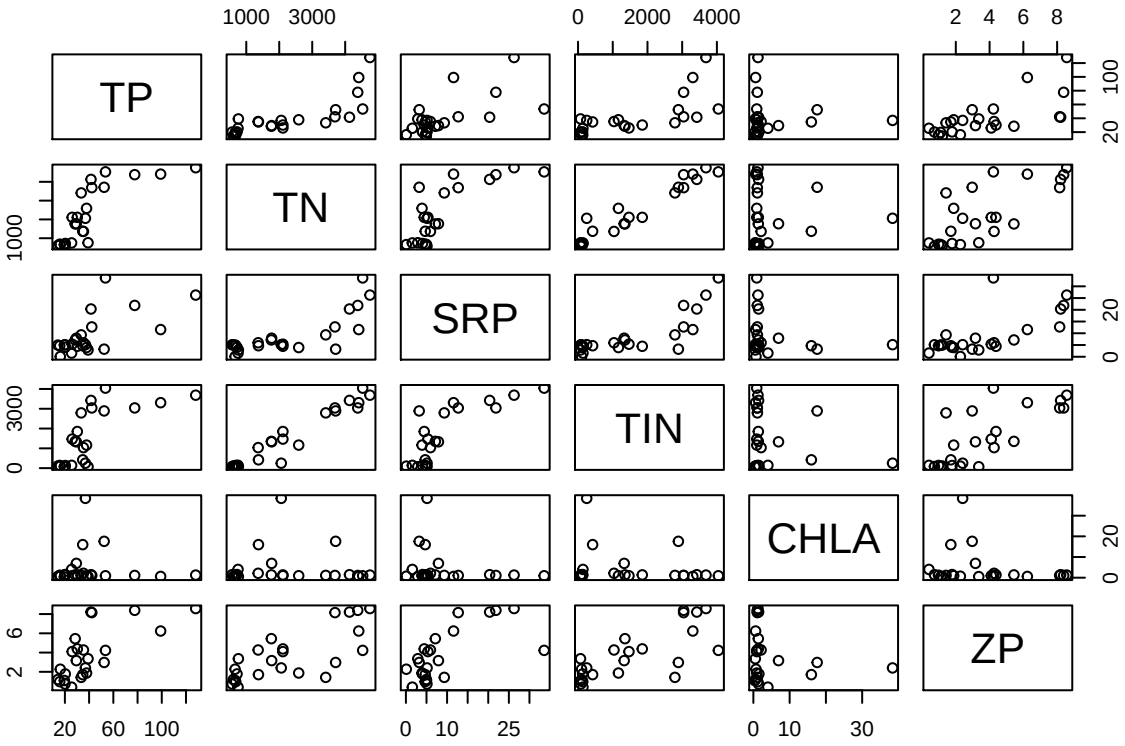
```
df <- read.table('data/zoop_nuts.txt', sep='\t', header=TRUE)
str(df)
```

```
## 'data.frame':    24 obs. of  8 variables:
##  $ TANK: int  34 14 23 16 21 5 25 27 30 28 ...
##  $ NUTS: Factor w/ 3 levels "H","L","M": 2 2 2 2 2 2 2 2 3 3 ...
##  $ TP  : num  20.3 25.6 14.2 39.1 20.1 ...
##  $ TN  : num  720 750 610 761 570 ...
##  $ SRP : num  4.02 1.56 4.97 2.89 5.11 4.68 5 0.1 7.9 3.92 ...
##  $ TIN : num  131.6 141.1 107.7 71.3 80.4 ...
##  $ CHLA: num  1.52 4 0.61 0.53 1.44 1.19 0.37 0.72 6.93 0.94 ...
##  $ ZP  : num  1.781 0.409 1.201 3.36 0.733 ...
```

Correlation

In the R-code chunk below, do the following: 1) Create a matrix with the numerical data in the `meso` dataframe. 2) Visualize the pairwise **bi-plots** of the six numerical variables. 3) Conduct a simple **Pearson's correlation** analysis.

```
df_num <- df[,3:8]
pairs(df_num, family = my_font, lab=colnames(df_num))
```




```
cor1 <- cor(df_num)
```

Question 3: Describe some of the general features based on the visualization and correlation analysis above?

> Answer 3: Theres a lot going on in these correlations. Many pairs are highly correlated in ways we should expect. In particular, each specific inorganic nutrient concentration is correlated with the total inorganic nutrient conecntration. Nothing correlates particularly well with the chloropyll *a* concentration but all of the inorganics concentrations correlate with the zooplankton biomass. The strongest correlation is with the total inogranic nutrient concentration.

In the R code chunk below, do the following: 1) Redo the correlation analysis using the `corr.test()` function in the `psych` package with the following options: method = "pearson", adjust = "BH". 2) Now, redo this correlation analysis using a non-parametric method. 3) Use the print command from the handout to see the results of each correlation analysis.

```
require('psych')
```

```
## Loading required package: psych
```

```
cor2 <- psych::corr.test(df_num, method='pearson', adjust='BH')
print(cor2, digits=3)
```

```
## Call:psych::corr.test(x = df_num, method = "pearson", adjust = "BH")
## Correlation matrix
##           TP     TN    SRP    TIN   CHLA     ZP
## TP    1.000  0.787  0.654  0.717 -0.017  0.697
## TN    0.787  1.000  0.784  0.969 -0.004  0.756
## SRP   0.654  0.784  1.000  0.801 -0.189  0.676
## TIN   0.717  0.969  0.801  1.000 -0.157  0.761
## CHLA -0.017 -0.004 -0.189 -0.157  1.000 -0.183
## ZP    0.697  0.756  0.676  0.761 -0.183  1.000
## Sample Size
## [1] 24
```

```
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##         TP    TN   SRP   TIN  CHLA    ZP
## TP   0.000 0.000 0.001 0.000 0.983 0.000
## TN   0.000 0.000 0.000 0.000 0.983 0.000
## SRP  0.001 0.000 0.000 0.000 0.491 0.000
## TIN  0.000 0.000 0.000 0.000 0.536 0.000
## CHLA 0.938 0.983 0.376 0.464 0.000 0.491
## ZP   0.000 0.000 0.000 0.000 0.393 0.000
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

Question 4: Describe what you learned from `corr.test`. Specifically, are the results sensitive to whether you use parametric (i.e., Pearson's) or non-parametric methods? When should one use non-parametric methods instead of parametric methods? With the Pearson's method, is there evidence for false discovery rate due to multiple comparisons? Why is false discovery rate important?

> Answer 4: Our results are mostly robust to the correlation method used. Pearson is good at identifying linear relationships between two continuous variables. If that is what we are interested in detecting, or if that is what our data show than Pearson's method is the best choice. If we are less interested in linearity or our interested in using discrete ordinal data we should use Spearman's or Kendall's method. Both can recover general monotonic relationships between variables. Some of our pairwise scatterplots look linear while others look like nonlinear but monotonic functions. I would be inclined to use Spearman's correlation. I don't see evidence of false discovery. I don't see any cases where the corrected p values alter the significance of relationships. It is important that we do some kind of correction however so that we may detect any false discovery issues. With a large number of statistical tests we should expect a fraction of false positives equal to our significance level. We can apply corrections, such as the Benjamini and Hochberg correction used above to keep the false discovery rate at an acceptable level.

Linear Regression

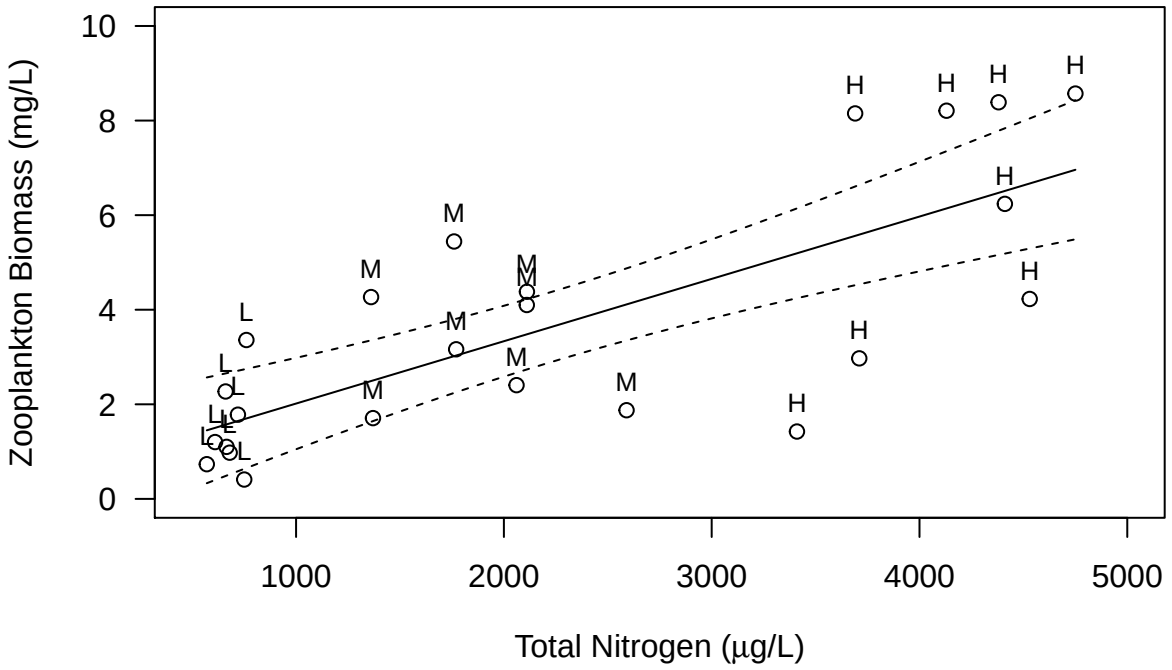
In the R code chunk below, do the following: 1) Conduct a linear regression analysis to test the relationship between total nitrogen (TN) and zooplankton biomass (ZP). 2) Examine the output of the regression analysis. 3) Produce a plot of this regression analysis including the following: categorically labeled points, the predicted regression line with 95% confidence intervals, and the appropriate axis labels.

```
reg <- lm(ZP ~ TN, df)
summary(reg)
```

```
##
## Call:
## lm(formula = ZP ~ TN, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7690 -0.8491 -0.0709  1.6238  2.5888
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.6977712  0.6496312   1.074    0.294
## TN          0.0013181  0.0002431   5.421 1.91e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.75 on 22 degrees of freedom
## Multiple R-squared:  0.5719, Adjusted R-squared:  0.5525
```

```
## F-statistic: 29.39 on 1 and 22 DF,  p-value: 1.911e-05
```

```
# scatter
plot(df$TN, df$ZP, ylim=c(0, 10), xlim=c(500, 5000),
     xlab=expression(paste("Total Nitrogen (", mu, "g/L)")),
     ylab="Zooplankton Biomass (mg/L)", las=1, family = my_font)
# labels
text(df$TN, df$ZP, df$NUTS, pos=3, cex=0.8, family = my_font)
# regression line
reg_line_x = seq(min(df$TN), max(df$TN), 10)
reg_line_y = predict(reg, newdata = data.frame(TN=reg_line_x))
lines(reg_line_x, reg_line_y)
# 95% CI
conf95 <- predict(reg, newdata=data.frame(TN=reg_line_x),
                  interval=c('confidence'), level=0.95, type='response')
matlines(reg_line_x, conf95[, c('lwr', 'upr')], lty=2, lwd=1, col='black')
```



Question 5: Interpret the results from the regression model

> Answer 5: The linear regression quantifies the increase in biomass in $fracmgL$ corresponding to an increase in total nitrogen concentration in $\frac{\mu g}{L}$. Assuming that the underlying relationship really is linear, which is well supported by the regression, than we should expect an increase of 0.001 $fracmgL$ of zooplankton biomass per $\frac{\mu g}{L}$ of nitrogen.

Analysis of Variance (ANOVA)

Using the R code chunk below, do the following: 1) Order the nutrient treatments from low to high (see handout). 2) Produce a barplot to visualize zooplankton biomass in each nutrient treatment. 3) Include error bars (+/- 1 sem) on your plot and label the axes appropriately. 4) Use a one-way analysis of variance (ANOVA) to test the null hypothesis that zooplankton biomass is affected by the nutrient treatment.

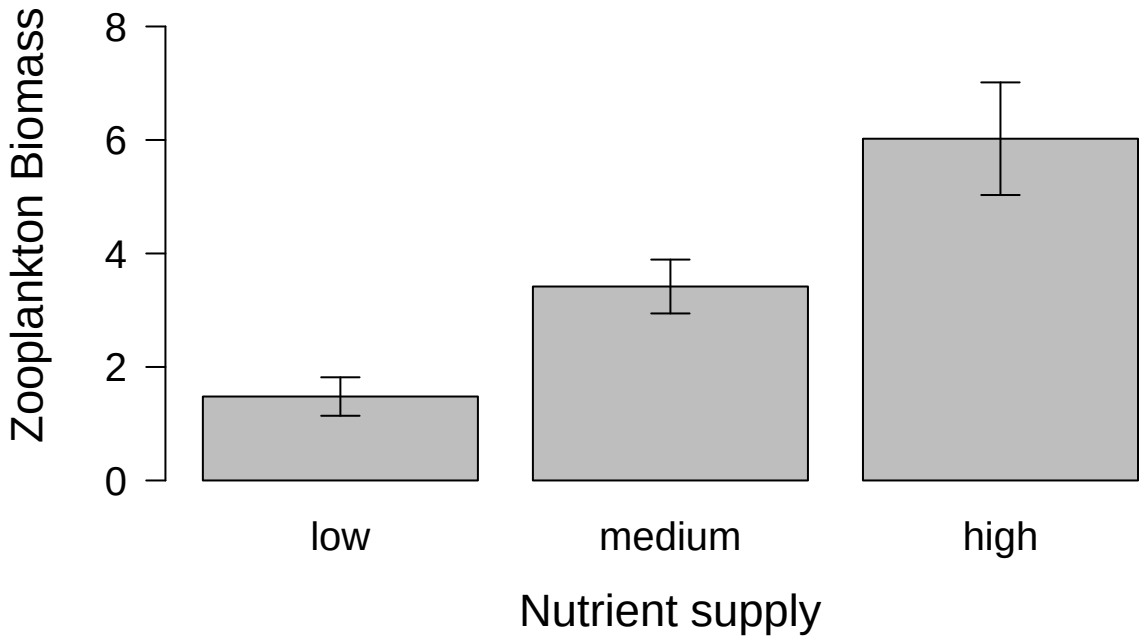
```
nuts <- factor(df$NUTS, levels=c('L', 'M', 'H'))
zp_means <- tapply(df$ZP, nuts, mean)

sem <- function(x) {
```

```
  x_cl <- na.omit(x)
  sd(x_cl) / sqrt(length(x_cl))
}

zp_sem <- tapply(df$ZP, nuts, sem)
bp <- barplot(zp_means, ylim=c(0, round(max(df$ZP), digits=0)),
              pch=15, cex=1.25, las=1, cex.lab=1.4, cex.axis=1.25,
              xlab = "Nutrient supply",
              ylab = "Zooplankton Biomass",
              names.arg = c("low", "medium", "high"),
              family = my_font)

arrows(x0 = bp, y0 = zp_means, y1 = zp_means - zp_sem, angle=90, length = 0.1, lwd=1)
arrows(x0 = bp, y0 = zp_means, y1 = zp_means + zp_sem, angle=90, length = 0.1, lwd=1)
```




```
fitanova <- aov(ZP ~ NUTS, df)
summary(fitanova)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## NUTS         2  83.15   41.58   11.77 0.000372 ***
## Residuals   21  74.16    3.53
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

SYNTHESIS: SITE-BY-SPECIES MATRIX

In the R code chunk below, load the zoops.txt data set in your **3.RStudio** data folder. Create a site-by-species matrix (or dataframe) that does *not* include TANK or NUTS. The remaining columns of data refer to the biomass (µg/L) of different zooplankton taxa:

- CAL = calanoid copepods
- DIAP = *Diaphanasoma* sp.
- CYL = cyclopoid copepods

- BOSM = *Bosmina* sp.
- SIMO = *Simocephallus* sp.
- CERI = *Ceriodaphnia* sp.
- NAUP = naupuli (immature copepod)
- DLUM = *Daphnia lumholtzi*
- CHYD = *Chydorus* sp.

Question 6: With the visualization and statistical tools that we learned about in the **3. RStudio** handout, use the site-by-species matrix to assess whether and how different zooplankton taxa were responsible for the total biomass (ZP) response to nutrient enrichment. Describe what you learned below in the "Answer" section and include appropriate code in the R chunk.

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
## Registered S3 method overwritten by 'rvest':
##   method            from
##   read_xml.response xml2
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.1.1       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x ggplot2::%+%()   masks psych::%+%()
## x ggplot2::alpha() masks psych::alpha()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
zoops <- readr::read_tsv('data/zoops.txt')
```

```
## Parsed with column specification:
## cols(
##   TANK = col_double(),
##   NUTS = col_character(),
##   CAL = col_double(),
##   DIAP = col_double(),
##   CYCL = col_double(),
##   BOSM = col_double(),
##   SIMO = col_double(),
##   CERI = col_double(),
##   NAUP = col_double(),
##   DLUM = col_double(),
##   CHYD = col_double()
## )
```

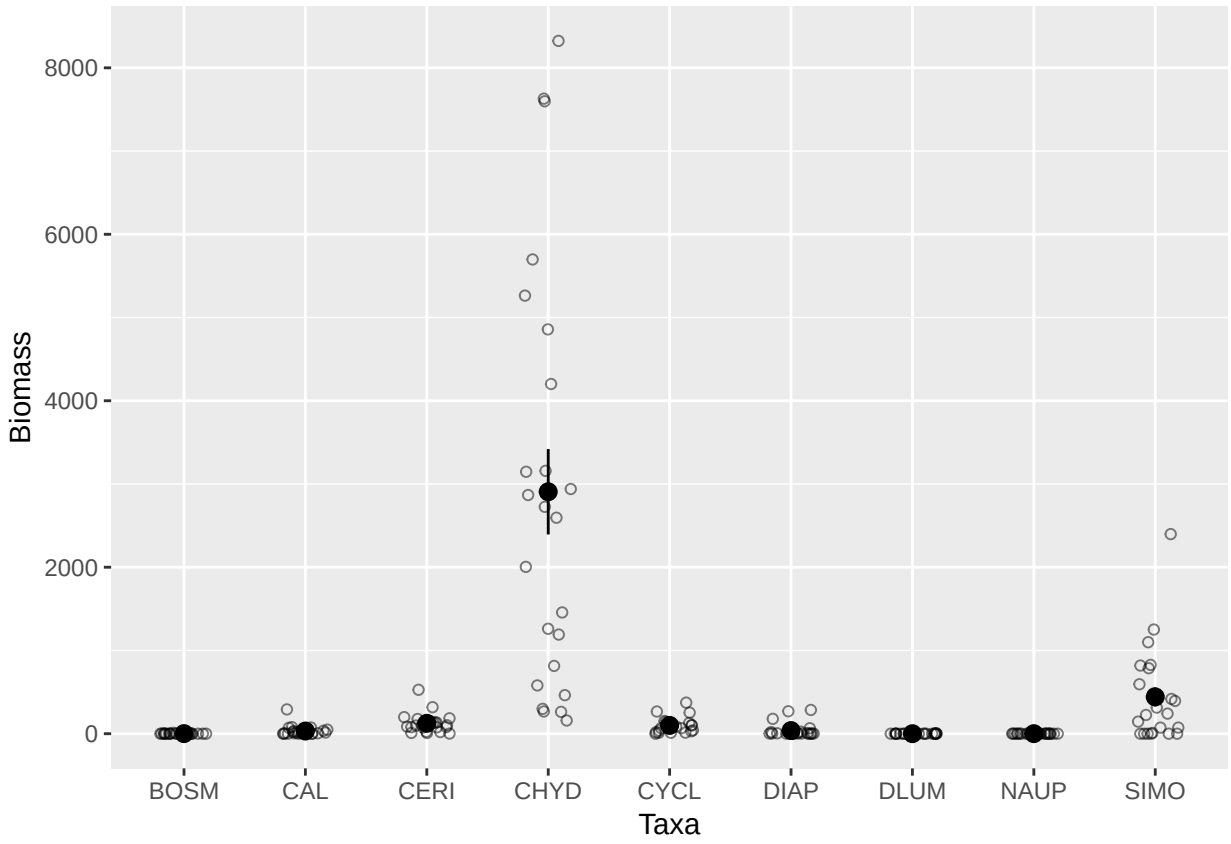
```r
# this calculates the sum
zoops_mat <- zoops %>% select(CAL:CHYD) %>% mutate(ZP = rowSums(.))
# now where going to get the correlation matrix and plot it
cor_mat <- zoops_mat %>%
  as.matrix %>%
  psych::corr.test(., method = 'pearson', adjust = 'BH')
print(cor_mat)
```

```
## Call:psych::corr.test(x = ., method = "pearson", adjust = "BH")
## Correlation matrix
##        CAL  DIAP  CYCL  BOSM  SIMO  CERI  NAUP  DLUM  CHYD    ZP
## CAL   1.00  0.64  0.71  0.73 -0.27 -0.19  0.06 -0.03 -0.32 -0.31
## DIAP  0.64  1.00  0.69  0.38 -0.29 -0.17  0.22  0.64 -0.31 -0.30
## CYCL  0.71  0.69  1.00  0.75 -0.32 -0.13  0.19  0.13 -0.37 -0.36
## BOSM  0.73  0.38  0.75  1.00 -0.31 -0.14  0.18 -0.09 -0.21 -0.21
## SIMO -0.27 -0.29 -0.32 -0.31  1.00 -0.18 -0.24 -0.08  0.26  0.43
## CERI -0.19 -0.17 -0.13 -0.14 -0.18  1.00  0.47  0.02 -0.14 -0.14
## NAUP  0.06  0.22  0.19  0.18 -0.24  0.47  1.00  0.15 -0.24 -0.24
## DLUM -0.03  0.64  0.13 -0.09 -0.08  0.02  0.15  1.00 -0.22 -0.21
## CHYD -0.32 -0.31 -0.37 -0.21  0.26 -0.14 -0.24 -0.22  1.00  0.98
## ZP   -0.31 -0.30 -0.36 -0.21  0.43 -0.14 -0.24 -0.21  0.98  1.00
## Sample Size
## [1] 24
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##       CAL DIAP CYCL BOSM SIMO CERI NAUP DLUM CHYD   ZP
## CAL  0.00 0.01 0.00 0.00 0.45 0.55 0.83 0.90 0.38 0.38
## DIAP 0.00 0.00 0.00 0.30 0.41 0.56 0.52 0.01 0.38 0.39
## CYCL 0.00 0.00 0.00 0.00 0.38 0.62 0.55 0.63 0.31 0.33
## BOSM 0.00 0.07 0.00 0.00 0.38 0.62 0.55 0.76 0.52 0.52
## SIMO 0.20 0.17 0.12 0.14 0.00 0.55 0.50 0.77 0.46 0.18
## CERI 0.37 0.42 0.54 0.51 0.39 0.00 0.11 0.93 0.62 0.62
## NAUP 0.79 0.31 0.39 0.40 0.27 0.02 0.00 0.62 0.50 0.50
## DLUM 0.88 0.00 0.56 0.69 0.72 0.93 0.49 0.00 0.52 0.52
## CHYD 0.13 0.14 0.08 0.33 0.22 0.53 0.26 0.29 0.00 0.00
## ZP   0.15 0.16 0.09 0.31 0.04 0.51 0.25 0.33 0.00 0.00
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

```r
# only CHYD has significant correlation with the total biomass. why?
zoops_long <- zoops_mat %>%
  gather(key = 'taxa', value = 'biomass', CAL:CHYD)

gg <- ggplot(zoops_long, aes(x=taxa, y=biomass)) +
  geom_jitter(width=0.2, alpha=0.5, shape=1) +
  stat_summary() +
  # scale_y_continuous(trans='log10') +
  theme(text=element_text(family=my_font)) +
  xlab('Taxa') +
  ylab('Biomass')
gg
```

```
## No summary function supplied, defaulting to `mean_se()
```




```
# turns out that all taxa is just puny compared to Chyrodus
```

Answer 6: The biomasses of most individual taxa do not correlate with total biomass. In fact, only *Chydorus* correlates signifigantly. The reason for this is that the *Chydorus* biomass completely dominates over the others on average.

SUBMITTING YOUR WORKSHEET

Use Knitr to create a PDF of your completed **3.RStudio_Worksheet.Rmd** document, push the repo to GitHub, and create a pull request. Please make sure your updated repo include both the PDF and RMarkdown files.

This assignment is due on **Wednesday, January 24th, 2021 at 12:00 PM (noon)**.