Week 1 Assignment: Basic R

Evgeniya Polezhaeva; Z620: Quantitative Biodiversity, Indiana University

18 January, 2017

OVERVIEW

Week 1 Assignment introduces some of the basic features of the R computing environment (http://www.r-project.org). It is designed to be used along side your Week 1 Handout (hard copy). You will not be able to complete the exercise if you do not have your handout.

Directions:

1. Change "Student Name" on line 3 (above) with your name.
2. Complete as much of the assignment as possible during class; what you do not complete in class will need to be done on your own outside of 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 exercise.
4. Be sure to **answer the questions** in this assignment document. 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.
5. Before you leave the classroom today, it is *imperative* that you **push** this file to your GitHub repo.
6. When you have completed the assignment, **Knit** the text and code into a single PDF file. Basically, just press the `Knit` button in the RStudio scripting panel. This will save the PDF output in your Week1 folder.
7. After Knitting, please submit the completed exercise by making a **push** to your GitHub repo and then create a **pull request** via GitHub. Your pull request should include this file (*Week1_Assignment.Rmd*; with all code blocks filled out and questions answered) and the PDF output of `Knitr` (*Week1_Assignment.pdf*).

The completed exercise is due on **Wednesday, January 18th, 2017 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 assignment.

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 Week1 folder.

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

```
## [1] "D:/Jane/GitHub/QB2017_Polezhaeva/Week1"
```

```
setwd("D:/Jane/GitHub/QB2017_Polezhaeva/Week1")
```

3) USING R AS A CALCULATOR

To follow up on the Week 0 exercises, please calculate the following in the R code chunk below. Feel free to reference the Week 0 handout.

1) the volume of a cube with length, l, = 5.
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. 45Â°) and with hypotenuse length sqrt(2) (remember: sin(theta) = opposite/hypotenuse).
4) the log (base e) of your favorite number.

```
l <- 5
V <- l^3
V
```

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

```
r <- 2
area <- pi * r ^ 2
area
```

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

```
hypotenuse <- sqrt(2)
opposite <- sin(pi/4) * hypotenuse
opposite
```

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

```
log (8)
```

```
## [1] 2.079442
```

4) WORKING WITH VECTORS

To follow up on the Week 0 exercises, please perform the requested operations in the Rcode chunks below. Feel free to reference the Week 0 handout.

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
(x + w) / 15
```

```
## [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`.

```r
k <- c(rnorm(5, mean = 5, sd = 2))
k * x
```

```
## [1]  4.264771  4.277513 14.759574 22.432649 25.385548
```

```r
index1 <- sample(1:length(w), 3, replace=F)
index2 <- sample(1:length(k), 4, replace=F)
d <- c(w[index1[1]], w[index1[2]], w[index1[3]], k[index2[1]], k[index2[2]],k[index2[3]],k[index2[4]])
d
```

```
## [1] 28.000000 70.000000 14.000000  5.077110  4.919858  5.608162  2.138757
```

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.

```r
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)
summary(v)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   10.10   16.50   20.35   20.90   24.95   31.40       1
```

```r
max(na.omit(v))
```

```
## [1] 31.4
```

```r
min(na.omit(v))
```

```
## [1] 10.1
```

```r
sum(na.omit(v))
```

```
## [1] 292.6
```

```r
mean(na.omit(v))
```

```
## [1] 20.9
```

```r
median(na.omit(v))
```

```
## [1] 20.35
```

```r
sd(na.omit(v))
```

```
## [1] 6.280127
```

```r
sem <- function(x){sd(x)/sqrt(length(x))}
sem(na.omit(v))
```

```
## [1] 1.678435
```

5) WORKING WITH MATRICES

In the R code chunk below, do the following: Using a mixture of Approach 1 and 2 from the 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.

```
a1 <- c(rnorm(10, mean = 8, sd = 2))
a2 <- c(rnorm(10, mean = 25, sd = 10))
b <- cbind(a1, a2)
dim(b)
```

```
## [1] 10  2
```

```
c <- matrix(b, nrow = 5, ncol = 2)
c
```

```
##          [,1]     [,2]
## [1,] 4.287796 5.822483
## [2,] 7.976881 6.061583
## [3,] 6.582964 6.227851
## [4,] 9.678723 6.860984
## [5,] 9.306882 6.178622
```

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 function returns N(first argument) random numbers from normal normal distribution with mean value equal to MEAN and standatd deviation equal to SD

In the R code chunk below, do the following: 1) Load `matrix.txt` from the Week1 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))
m1 <- t(m)
dim(m1)
```

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

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

> Answer 2: 10 x 5

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`.

```
m2 <- m[ , c(1:2, 4:5)]
m2
```

```
##       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
```

```
m <- m[-10, ]
m
```

```
##       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
```

Question 3: Describe what we just did in the last series of indexing steps.

> ***Answer 3***:) We selected subset of matrix m (all elements but the third column) 2) We changed matrix m deleting its last row

6) BASIC DATA VISUALIZATION AND STATISTICAL ANALYSIS

Load Zooplankton Dataset

In the R code chunk below, do the following: 1) Load the zooplankton dataset from the Week1 data folder. 2) Display the structure of this data set.

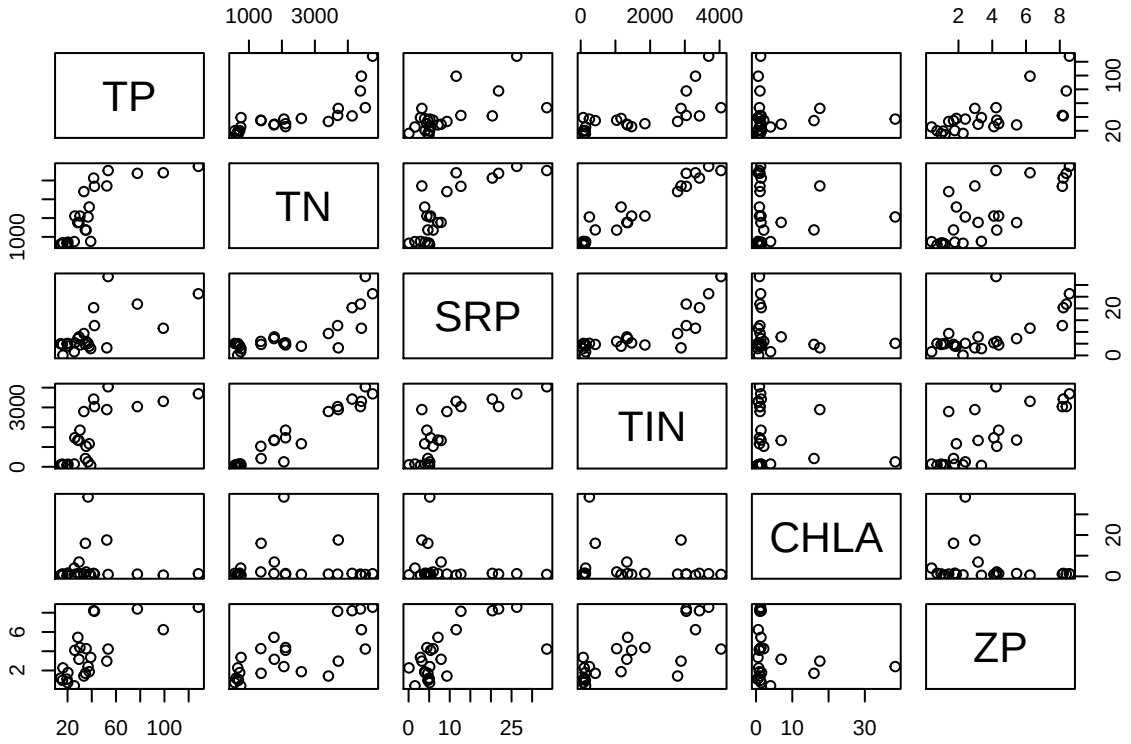
```
meso <- read.table("data/zoop_nuts.txt", sep = "\t", header = TRUE)
str(meso)
```

```
## '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.

```
meso.num <- meso[, 3:8]
pairs(meso.num)
```




```
cor1 <- cor(meso.num)
print(cor1)
```

```
##                TP           TN        SRP        TIN         CHLA
## TP    1.00000000  0.786510407  0.6540957  0.7171143 -0.016659593
## TN    0.78651041  1.000000000  0.7841904  0.9689999 -0.004470263
## SRP   0.65409569  0.784190400  1.0000000  0.8009033 -0.189148017
## TIN   0.71711434  0.968999866  0.8009033  1.0000000 -0.156881463
## CHLA -0.01665959 -0.004470263 -0.1891480 -0.1568815  1.000000000
## ZP    0.69747649  0.756247384  0.6762947  0.7605629 -0.182599904
##               ZP
## TP    0.6974765
## TN    0.7562474
## SRP   0.6762947
## TIN   0.7605629
## CHLA -0.1825999
## ZP    1.0000000
```

Question 4: Describe some of the general features based on the visualization and correlation analysis above?

> Answer 4: Based on visualization and correlation analysis we can say that there is correlation between some parameters, for example almost linear relationship between TN and TIN(|correlation coefficient| -> 1), non-linear, but still strong correlation ZP~TP,TN,SRP,TIN, TP~TN, TP~SRP, SPR~TIN. It's interesting that only concentration of Chrolophyll negatively correlates with all the considered parameters (nutrients concentration and ZP biomass).

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 <- corr.test(meso.num, method = "pearson", adjust ="BH")
print(cor2, digits = 4, short=FALSE)
```

```
## Call:corr.test(x = meso.num, method = "pearson", adjust = "BH")
## Correlation matrix
##           TP      TN     SRP     TIN    CHLA      ZP
## TP    1.0000  0.7865  0.6541  0.7171 -0.0167  0.6975
## TN    0.7865  1.0000  0.7842  0.9690 -0.0045  0.7562
## SRP   0.6541  0.7842  1.0000  0.8009 -0.1891  0.6763
## TIN   0.7171  0.9690  0.8009  1.0000 -0.1569  0.7606
## CHLA -0.0167 -0.0045 -0.1891 -0.1569  1.0000 -0.1826
## ZP    0.6975  0.7562  0.6763  0.7606 -0.1826  1.0000
## Sample Size
## [1] 24
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##          TP     TN    SRP    TIN   CHLA     ZP
## TP   0.0000 0.0000 0.0008 0.0002 0.9835 0.0003
## TN   0.0000 0.0000 0.0000 0.0000 0.9835 0.0000
## SRP  0.0005 0.0000 0.0000 0.0000 0.4914 0.0005
## TIN  0.0001 0.0000 0.0000 0.0000 0.5355 0.0000
## CHLA 0.9384 0.9835 0.3761 0.4641 0.0000 0.4914
## ZP   0.0002 0.0000 0.0003 0.0000 0.3931 0.0000
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
##
##  Confidence intervals based upon normal theory.  To get bootstrapped values, try cor.ci
##            lower       r  upper      p
## TP-TN     0.5612  0.7865 0.9033 0.0000
## TP-SRP    0.3406  0.6541 0.8367 0.0005
## TP-TIN    0.4414  0.7171 0.8691 0.0001
## TP-CHLA  -0.4173 -0.0167 0.3894 0.9384
## TP-ZP     0.4092  0.6975 0.8591 0.0002
## TN-SRP    0.5570  0.7842 0.9022 0.0000
## TN-TIN    0.9286  0.9690 0.9867 0.0000
## TN-CHLA  -0.4071 -0.0045 0.3996 0.9835
## TN-ZP     0.5077  0.7562 0.8886 0.0000
## SRP-TIN   0.5872  0.8009 0.9102 0.0000
## SRP-CHLA -0.5505 -0.1891 0.2319 0.3761
## SRP-ZP    0.3753  0.6763 0.8483 0.0003
## TIN-CHLA -0.5269 -0.1569 0.2632 0.4641
## TIN-ZP    0.5152  0.7606 0.8907 0.0000
## CHLA-ZP  -0.5458 -0.1826 0.2384 0.3931
```

Question 5: Describe what you learned from `corr.test`. 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 5: corr.test function finds correlation, sample sizes, and probability values between elements of a data.frame.

The results are sensitive weather we use parametric or non-parametric method. Parametric method (Pearson) measures the strength of the LINEAR relationship between normally distributed variable, while relationship between the variables is not linear, it may be more appropriate to use the Spearman rank correlation method.

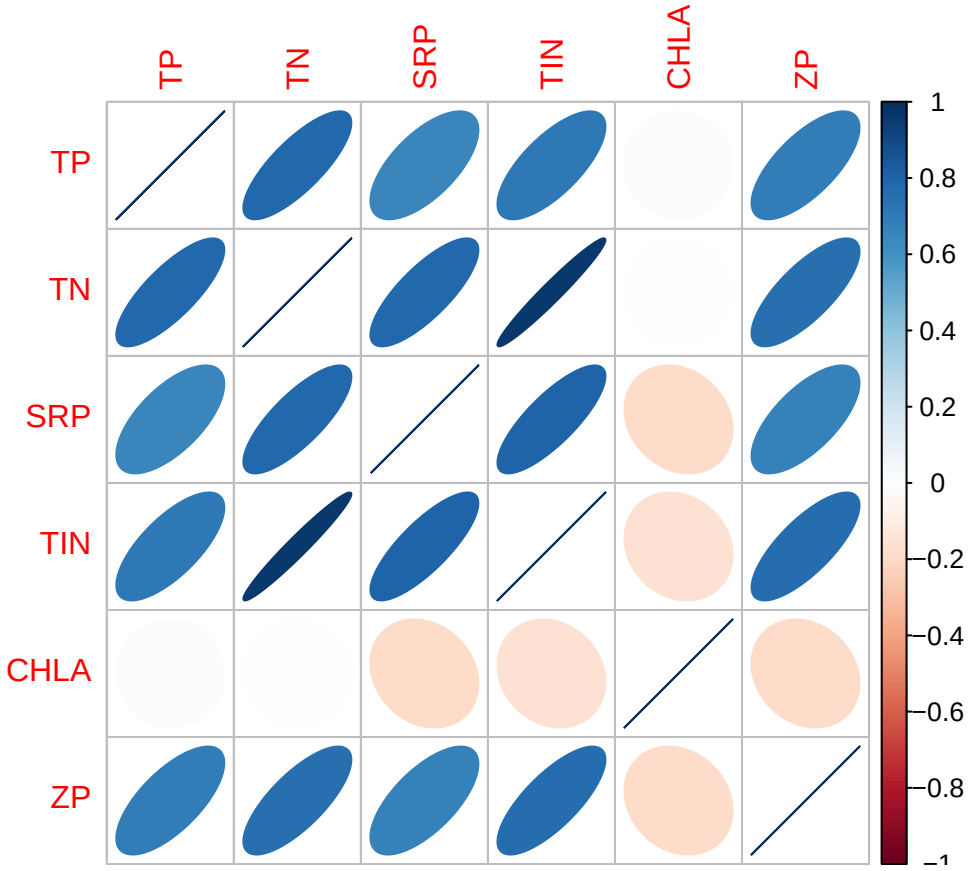
With the Pearson's method, there is evidence for false discovery rate due to multiple comparisons: p-values above the diagonal differ from non-adjusted for multiple tests. FDR is the proportion of discoveries that are false among all discoveries. This approach allows to adjust p-values that could result in reducing of false-positive results.

In the R code chunk below, use the `corrplot` function in the *corrplot* package to produce the ellipse correlation plot in the handout.

```
require(corrplot)
```

```
## Loading required package: corrplot
```

```
corrplot(cor1, method = "ellipse")
```




```
dev.off()
```

```
## null device
##           1
```

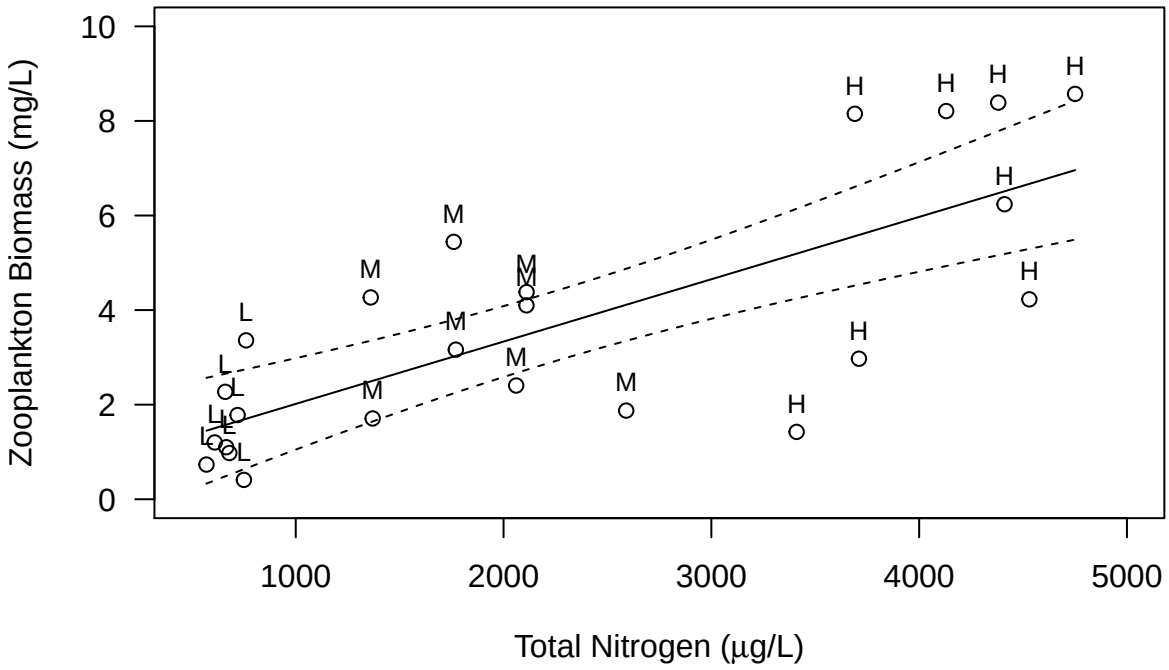
Linear Regression

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.

```
fitreg <- lm(ZP ~ TN, data = meso)
summary(fitreg)
```

```
##
## Call:
## lm(formula = ZP ~ TN, data = meso)
##
## 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
```

```
plot(meso$TN, meso$ZP, ylim = c(0, 10), xlim = c(500, 5000), xlab = expression(paste("Total Nitrogen ("

text(meso$TN, meso$ZP, meso$NUTS, pos = 3, cex = 0.8)
newTN <- seq(min(meso$TN), max(meso$TN), 10)
regline <- predict(fitreg, newdata = data.frame(TN = newTN))
lines(newTN, regline)
conf95 <- predict(fitreg, newdata = data.frame(TN = newTN), interval = c("confidence"), level = 0.95, ty
matlines(newTN, conf95[, c("lwr", "upr")], type="l", lty = 2, lwd = 1, col = "black")
```



Question 6: Interpret the results from the regression model

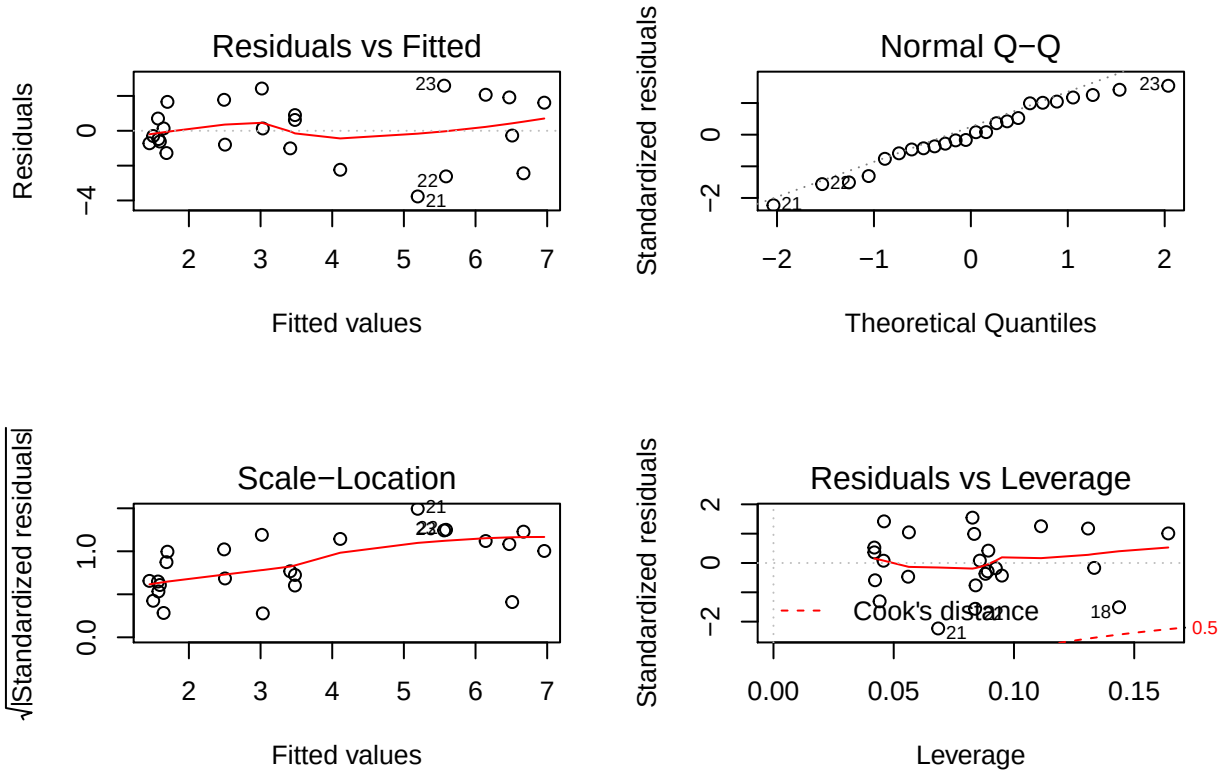
> ***Answer 6***: lm function is used to fit data to linear model. According to regression model this equetion best describes ZP~TN relation: ZP = 0.0013181TN + 0.697771 with adjusted R^2 = 0.5525 The section 'Residuals' gives the difference between the experimental and predicted signals.

Question 7: Explain what the `predict()` function is doing in our analyses.

> ***Answer 7***: function predict() returns a line based on linear regression analysis of ZP~TN data.

Using the R code chunk below, use the code provided in the handout to determine if our data meet the assumptions of the linear regression analysis.

```
par(mfrow = c(2,2), mar = c(5.1, 4.1, 4.1, 2.1))
plot(fitreg)
```

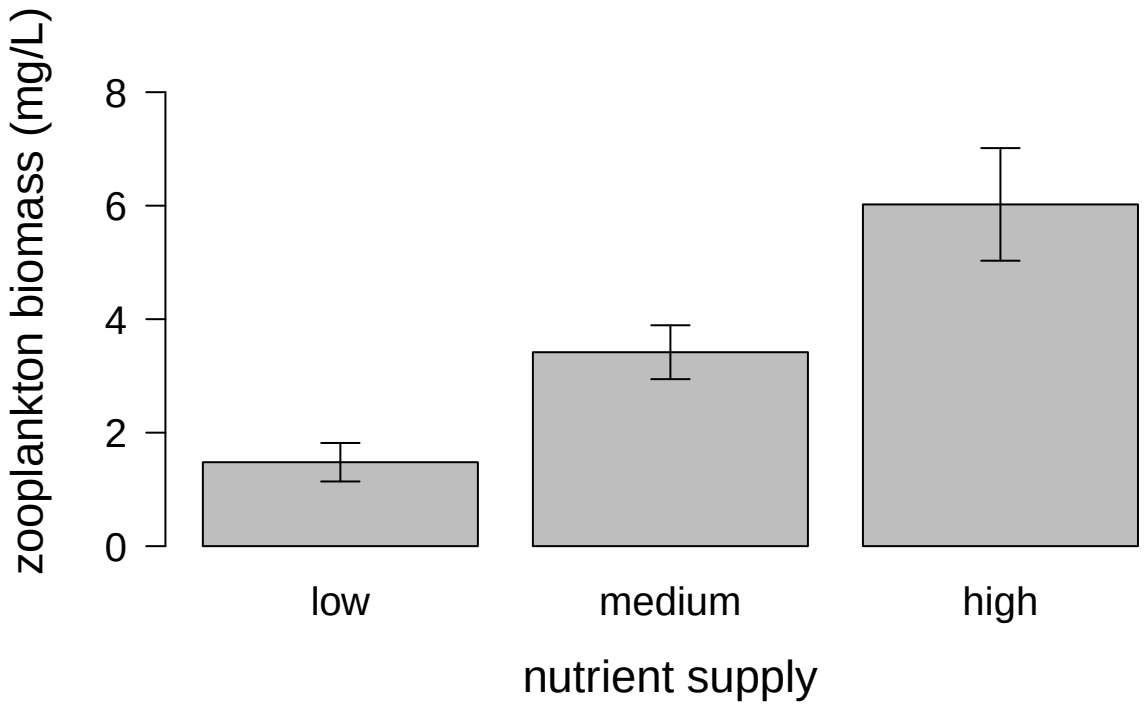


- Upper left: is there a random distribution of the residuals around zero (horizontal line)?
- Upper right: is there a reasonably linear relationship between standardized residuals and theoretical quantiles? Try `help(qqplot)`
- Bottom left: again, looking for a random distribution of sqrt(standardized residuals)
- Bottom right: leverage indicates the influence of points; contours correspond with Cook's distance, where values > |1| are "suspicious"

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. 5) Use a Tukey's HSD to identify which treatments are different.

```
NUTS <- factor(meso$NUTS, levels = c('L', 'M', 'H'))
zp.means <- tapply(meso$ZP, NUTS, mean)
sem <- function(x){sd(na.omit(x))/sqrt(length(na.omit(x)))}
zp.sem <- tapply(meso$ZP, NUTS, sem)

bp <- barplot(zp.means, ylim = c(0, round(max(meso$ZP), digits = 0)), pch = 15, cex = 1.25, las = 1, ce

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, data = meso)
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
```

```
TukeyHSD(fitanova)
```

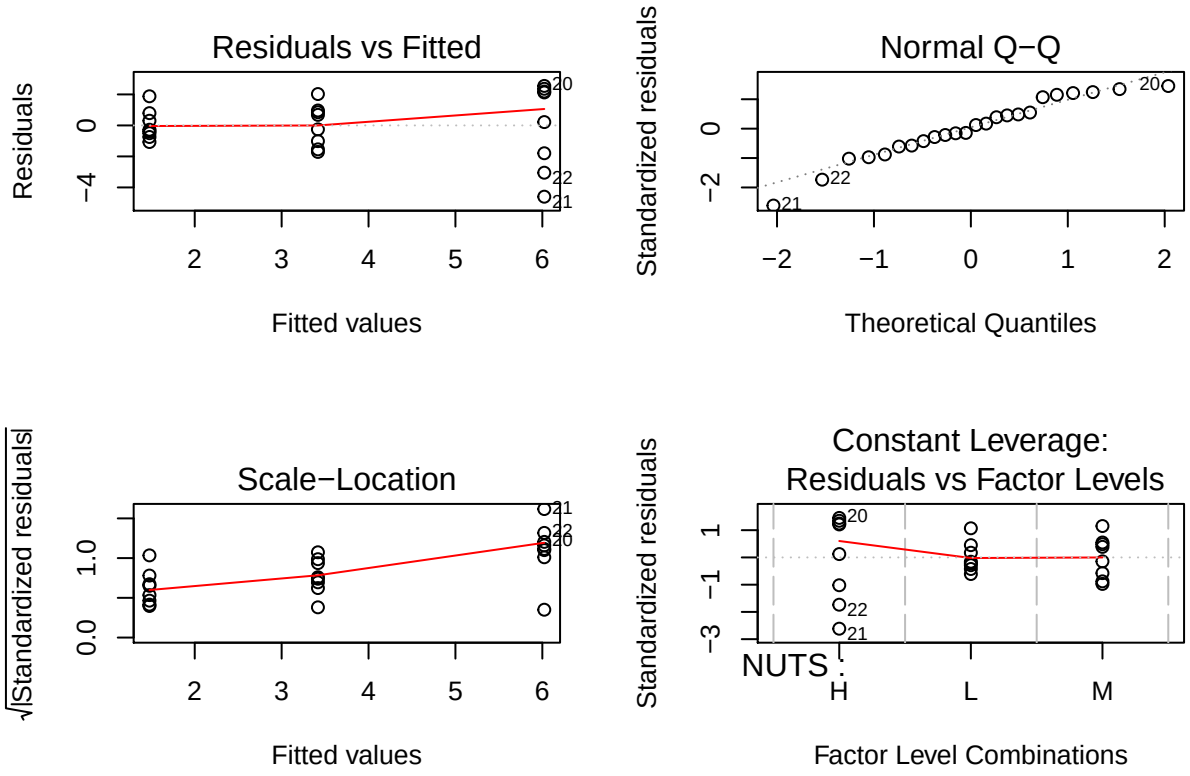
```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = ZP ~ NUTS, data = meso)
##
## $NUTS
##          diff        lwr        upr     p adj
## L-H -4.543175 -6.9115094 -2.1748406 0.0002512
## M-H -2.604550 -4.9728844 -0.2362156 0.0294932
## M-L  1.938625 -0.4297094  4.3069594 0.1220246
```

Question 8: How do you interpret the ANOVA results relative to the regression results? Do you have any concerns about this analysis?

Answer 8:

Using the R code chunk below, use the diagnostic code provided in the handout to determine if our data meet the assumptions of ANVOA (similar to regression).

```
par(mfrow = c(2, 2), mar = c(5.1, 4.1, 4.1, 2.1))
plot(fitanova)
```



SYNTHESIS: SITE-BY-SPECIES MATRIX

In the R code chunk below, load the zoop.txt dataset in your Week1 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 (Âµg/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 9: With the visualization and statistical tools that we learned about in the Week 1 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. ***Answer 9***: According to correlation analysis there is strong positive correlation between ZP biomass and following taxa: Simocephallus and Chydorus; and strong negative correlation:calanoid copepods, Diaphanasoma, naupuli, Daphnia lumholtzi. The most significant contribution in ZP biomass change was made by Chydorus (we can see it on bar plot). Unfortunately the analysis performed does not allow to compare statistical weights.

```
zoop <- read.table("data/zoops.txt", sep = "\t", header = TRUE)
zoop$sum <- zoop$CAL +zoop$DIAP + zoop$CYCL + zoop$BOSM + zoop$SIMO + zoop$CERI + zoop$NAUP + zoop$DLUM
str(zoop)
```

```
## 'data.frame':    24 obs. of  12 variables:
##  $ TANK: int  5 14 16 21 23 25 27 34 12 15 ...
##  $ NUTS: Factor w/ 3 levels "H","L","M": 2 2 2 2 2 2 2 2 3 3 ...
##  $ CAL : num  70.5 27.1 5.3 79.2 31.4 22.7 0 35.7 74.8 5.3 ...
##  $ DIAP: num  0 19.2 8.8 17.9 0 ...
##  $ CYCL: num  66.1 129.6 12.7 141.3 11 ...
##  $ BOSM: num  2.2 0 0 3.4 0 0 0 0 0 0 ...
##  $ SIMO: num  417.8 0 73.1 0 482 ...
##  $ CERI: num  159.8 79.4 107.5 199 101.9 ...
##  $ NAUP: num  0 0 1.2 0 0 1.2 1.6 3.1 0 1.4 ...
##  $ DLUM: num  0 0 0 0 0 6.6 0 0 0 0 ...
##  $ CHYD: num  267 159 3158 298 580 ...
##  $ sum : num  983 414 3367 739 1206 ...
```

```
NUTS.zoop <- factor(zoop$NUTS, levels = c('L', 'M', 'H'))
CAL.means <- tapply(zoop$CAL, NUTS, mean)
DIAP.means <- tapply(zoop$DIAP, NUTS, mean)
CYCL.means <- tapply(zoop$CYCL, NUTS, mean)
BOSM.means <- tapply(zoop$BOSM, NUTS, mean)
SIMO.means <- tapply(zoop$SIMO, NUTS, mean)
CERI.means <- tapply(zoop$CERI, NUTS, mean)
NAUP.means <- tapply(zoop$NAUP, NUTS, mean)
DLUM.means <- tapply(zoop$DLUM, NUTS, mean)
CHYD.means <- tapply(zoop$CHYD, NUTS, mean)
SUM.means <- tapply(zoop$sum, NUTS, mean)
MEANS <- data.frame(CAL.means, DIAP.means, CYCL.means, BOSM.means, SIMO.means, CERI.means, NAUP.means, D

str(MEANS)
```

```
## 'data.frame':    3 obs. of  10 variables:
##  $ CAL.means : num [1:3(1d)] 33.99 58.41 4.29
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ DIAP.means: num [1:3(1d)] 49.9 57.79 5.79
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ CYCL.means: num [1:3(1d)] 78.5 131.7 89.2
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ BOSM.means: num [1:3(1d)] 0.7 1.61 1.04
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ SIMO.means: num [1:3(1d)] 161 465 701
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
```

```
##  $ CERI.means: num [1:3(1d)] 160.8 77.5 136.3
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ NAUP.means: num [1:3(1d)] 0.887 0.625 0.35
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ DLUM.means: num [1:3(1d)] 0.825 0 0
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ CHYD.means: num [1:3(1d)] 999 2632 5089
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
##  $ SUM.means : num [1:3(1d)] 1485 3424 6028
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr  "L" "M" "H"
```

```
cor3 <- cor(MEANS)
print(cor3)
```

```
##              CAL.means DIAP.means  CYCL.means BOSM.means SIMO.means
## CAL.means   1.00000000  0.9472687  0.71667376  0.5782181 -0.4866134
## DIAP.means  0.94726869  1.0000000  0.45540476  0.2862863 -0.7408959
## CYCL.means  0.71667376  0.4554048  1.00000000  0.9833970  0.2605252
## BOSM.means  0.57821811  0.2862863  0.98339701  1.0000000  0.4314002
## SIMO.means -0.48661343 -0.7408959  0.26052520  0.4314002  1.0000000
## CERI.means -0.64462978 -0.3656626 -0.99515455 -0.9964744 -0.3541905
## NAUP.means  0.55904518  0.7952550 -0.17759439 -0.3532283 -0.9963876
## DLUM.means  0.05617945  0.3731513 -0.65604481 -0.7821097 -0.8995754
## CHYD.means -0.64091737 -0.8530941  0.07600906  0.2556892  0.9824764
## SUM.means  -0.61637825 -0.8362067  0.10743242  0.2860656  0.9878682
##            CERI.means NAUP.means  DLUM.means  CHYD.means  SUM.means
## CAL.means  -0.6446298  0.5590452  0.05617945 -0.64091737 -0.6163783
## DIAP.means -0.3656626  0.7952550  0.37315132 -0.85309414 -0.8362067
## CYCL.means -0.9951545 -0.1775944 -0.65604481  0.07600906  0.1074324
## BOSM.means -0.9964744 -0.3532283 -0.78210972  0.25568923  0.2860656
## SIMO.means -0.3541905 -0.9963876 -0.89957536  0.98247635  0.9878682
## CERI.means  1.0000000  0.2734940  0.72707258 -0.17367942 -0.2046659
## NAUP.means  0.2734940  1.0000000  0.85923458 -0.99475567 -0.9974876
## DLUM.means  0.7270726  0.8592346  1.00000000 -0.80240397 -0.8208344
## CHYD.means -0.1736794 -0.9947557 -0.80240397  1.00000000  0.9995021
## SUM.means  -0.2046659 -0.9974876 -0.82083438  0.99950210  1.0000000
```

```
require("psych")
cor4 <- corr.test(MEANS, method = "pearson", adjust = "BH")
```

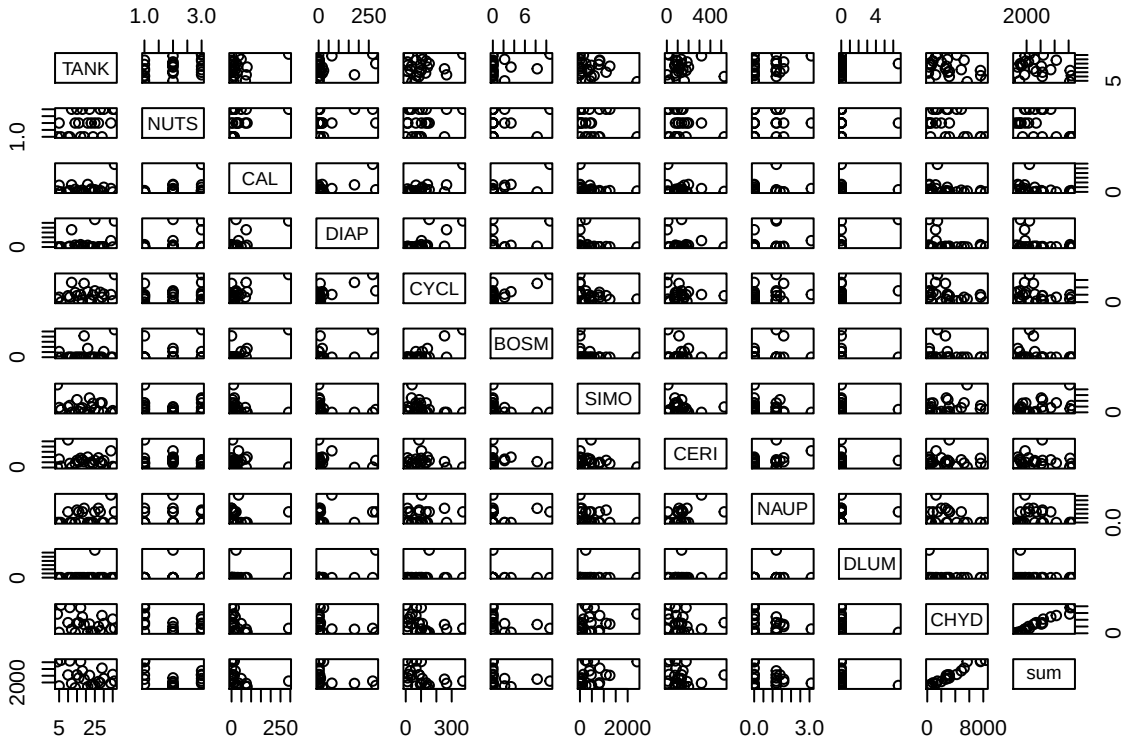
```
## Warning in corr.test(MEANS, method = "pearson", adjust = "BH"): Number of
## subjects must be greater than 3 to find confidence intervals.
```

```
print(cor4, digits = 3)
```

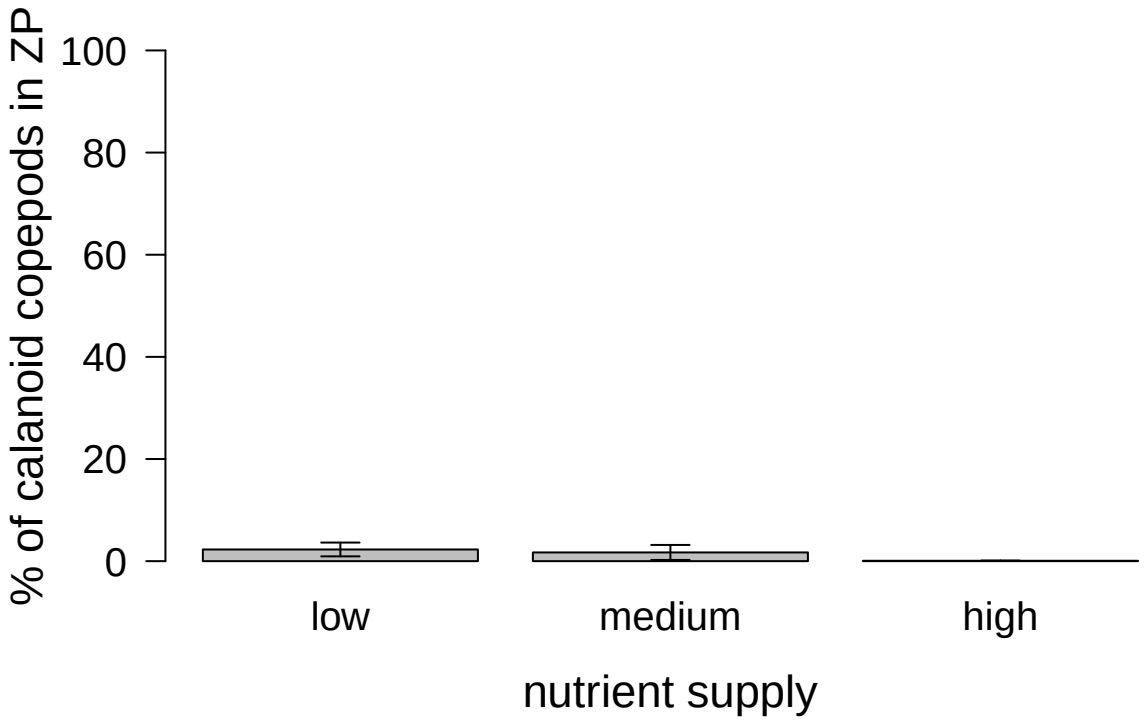
```
## Call:corr.test(x = MEANS, method = "pearson", adjust = "BH")
## Correlation matrix
##            CAL.means DIAP.means CYCL.means BOSM.means SIMO.means
## CAL.means      1.000      0.947      0.717      0.578     -0.487
## DIAP.means     0.947      1.000      0.455      0.286     -0.741
## CYCL.means     0.717      0.455      1.000      0.983      0.261
```

```
## BOSM.means     0.578      0.286      0.983      1.000      0.431
## SIMO.means    -0.487     -0.741      0.261      0.431      1.000
## CERI.means    -0.645     -0.366     -0.995     -0.996     -0.354
## NAUP.means     0.559      0.795     -0.178     -0.353     -0.996
## DLUM.means     0.056      0.373     -0.656     -0.782     -0.900
## CHYD.means    -0.641     -0.853      0.076      0.256      0.982
## SUM.means     -0.616     -0.836      0.107      0.286      0.988
##            CERI.means NAUP.means DLUM.means CHYD.means SUM.means
## CAL.means      -0.645      0.559      0.056     -0.641    -0.616
## DIAP.means     -0.366      0.795      0.373     -0.853    -0.836
## CYCL.means     -0.995     -0.178     -0.656      0.076     0.107
## BOSM.means     -0.996     -0.353     -0.782      0.256     0.286
## SIMO.means     -0.354     -0.996     -0.900      0.982     0.988
## CERI.means      1.000      0.273      0.727     -0.174    -0.205
## NAUP.means      0.273      1.000      0.859     -0.995    -0.997
## DLUM.means      0.727      0.859      1.000     -0.802    -0.821
## CHYD.means     -0.174     -0.995     -0.802      1.000     1.000
## SUM.means      -0.205     -0.997     -0.821      1.000     1.000
## Sample Size
## [1] 3
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##            CAL.means DIAP.means CYCL.means BOSM.means SIMO.means
## CAL.means      0.000      0.934      0.952      0.952      0.952
## DIAP.means     0.208      0.000      0.952      0.952      0.952
## CYCL.means     0.491      0.699      0.000      0.597      0.952
## BOSM.means     0.607      0.815      0.116      0.000      0.952
## SIMO.means     0.676      0.469      0.832      0.716      0.000
## CERI.means     0.554      0.762      0.063      0.053      0.770
## NAUP.means     0.622      0.415      0.886      0.770      0.054
## DLUM.means     0.964      0.757      0.544      0.428      0.288
## CHYD.means     0.557      0.349      0.952      0.835      0.119
## SUM.means      0.577      0.370      0.931      0.815      0.099
##            CERI.means NAUP.means DLUM.means CHYD.means SUM.means
## CAL.means       0.952      0.952      0.964      0.952     0.952
## DIAP.means      0.952      0.952      0.952      0.952     0.952
## CYCL.means      0.489      0.952      0.952      0.964     0.964
## BOSM.means      0.489      0.952      0.952      0.952     0.952
## SIMO.means      0.952      0.489      0.952      0.597     0.597
## CERI.means      0.000      0.952      0.952      0.952     0.952
## NAUP.means      0.824      0.000      0.952      0.489     0.489
## DLUM.means      0.482      0.342      0.000      0.952     0.952
## CHYD.means      0.889      0.065      0.407      0.000     0.489
## SUM.means       0.869      0.045      0.387      0.020     0.000
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

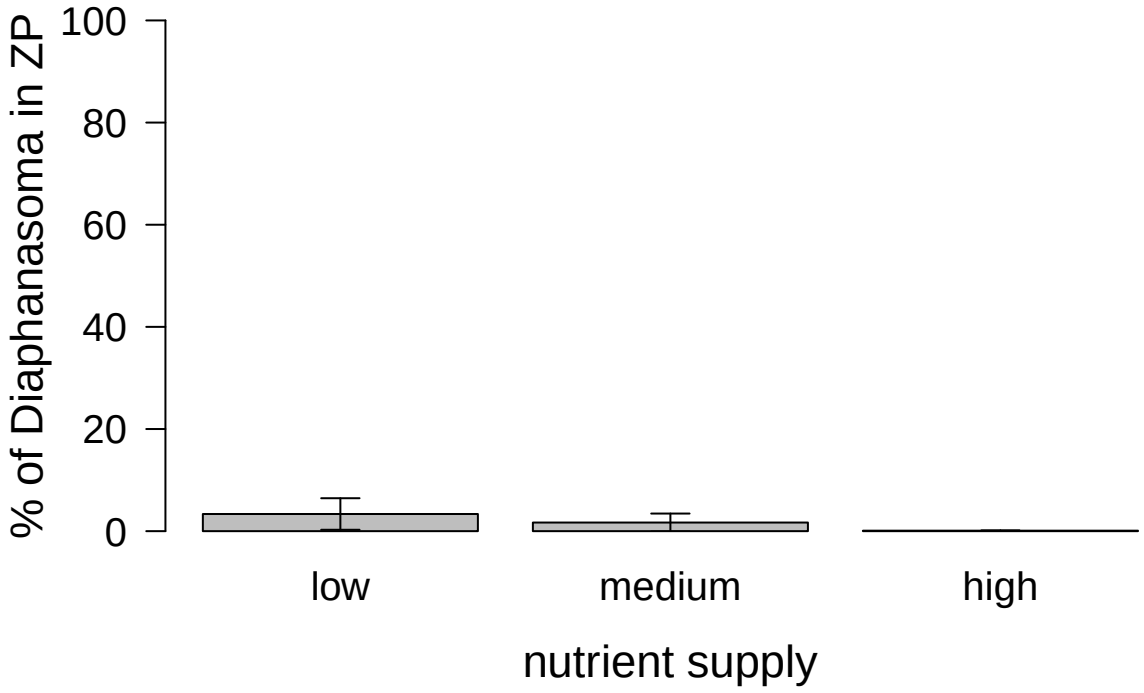
```
pairs(zoop)
```



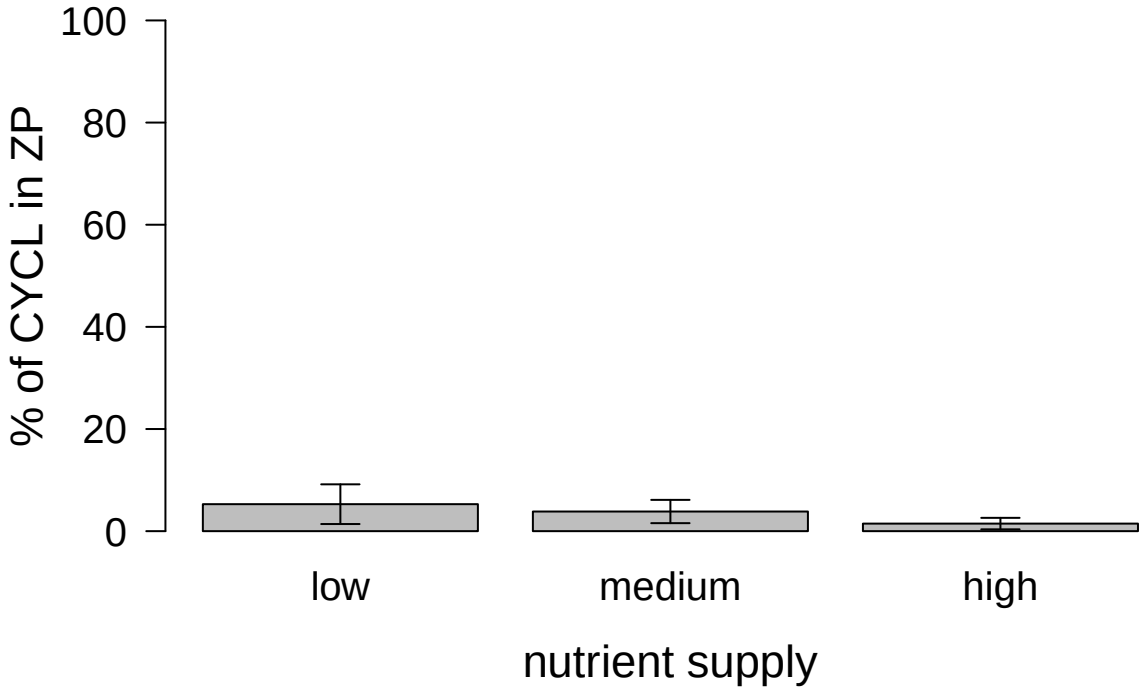

```
#Bar Plots. Everything below this line may be ignored
sem <- function(x){sd(na.omit(x))/sqrt(length(na.omit(x)))}
CAL.sem <- tapply((zoop$CAL)/(zoop$sum), NUTS, sem)
DIAP.sem <- tapply(zoop$DIAP/(zoop$sum), NUTS, sem)
CYCL.sem <- tapply(zoop$CYCL/(zoop$sum), NUTS, sem)
BOSM.sem <- tapply(zoop$BOSM/(zoop$sum), NUTS, sem)
SIMO.sem <- tapply(zoop$SIMO/(zoop$sum), NUTS, sem)
CERI.sem <- tapply(zoop$CERI/(zoop$sum), NUTS, sem)
NAUP.sem <- tapply(zoop$NAUP/(zoop$sum), NUTS, sem)
DLUM.sem <- tapply(zoop$DLUM/(zoop$sum), NUTS, sem)
CHYD.sem <- tapply(zoop$CHYD/(zoop$sum), NUTS, sem)

bp.CAL <- barplot(CAL.means / SUM.means * 100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab =

arrows(x0 = bp.CAL, y0 = CAL.means/SUM.means*100, y1 = (CAL.means/SUM.means - CAL.sem)*100, angle = 90,
arrows(x0 = bp.CAL, y0 = CAL.means/SUM.means*100, y1 = (CAL.means/SUM.means + CAL.sem)*100, angle = 90,
```




```
bp.DIAP <- barplot(DIAP.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab =

arrows(x0 = bp.DIAP, y0 = DIAP.means/SUM.means*100, y1 = (DIAP.means/SUM.means-DIAP.sem)*100, angle = 9
arrows(x0 = bp.DIAP, y0 = DIAP.means/SUM.means*100, y1 = (DIAP.means/SUM.means + DIAP.sem)*100, angle =
```




```
bp.CYCL <- barplot(CYCL.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab = 

arrows(x0 = bp.CYCL, y0 = CYCL.means/SUM.means*100, y1 = (CYCL.means/SUM.means-CYCL.sem)*100, angle = 9
arrows(x0 = bp.CYCL, y0 = CYCL.means/SUM.means*100, y1 = (CYCL.means/SUM.means + CYCL.sem)*100, angle =
```

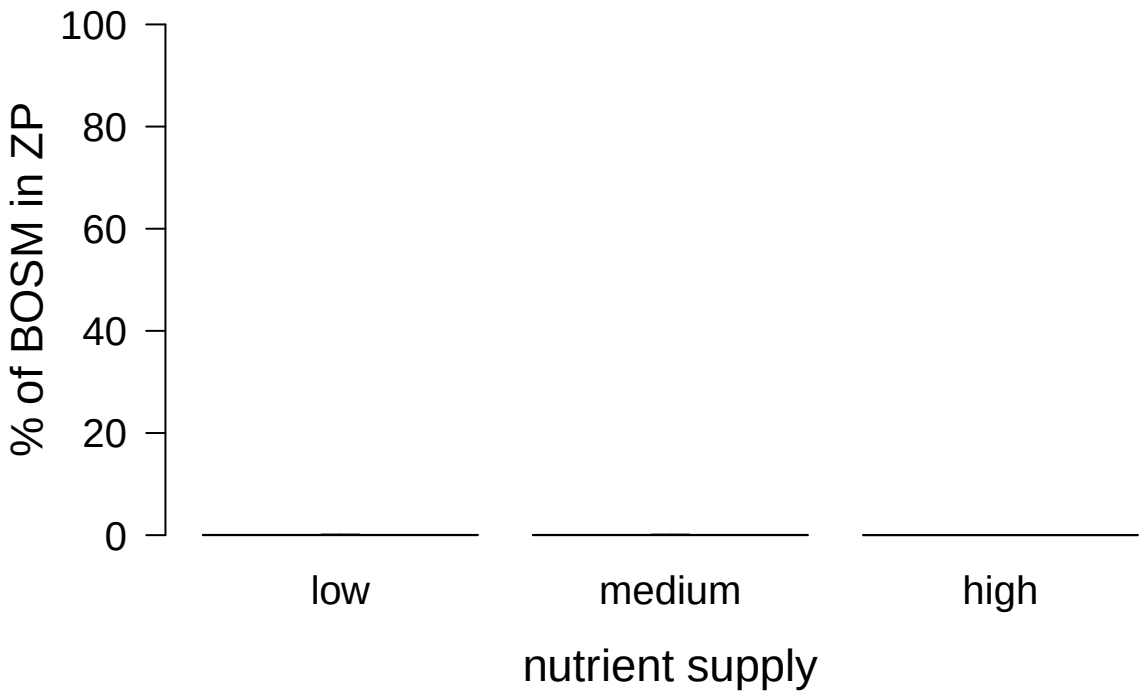



```
bp.BOSM <- barplot(BOSM.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab =

arrows(x0 = bp.BOSM, y0 = BOSM.means/SUM.means*100, y1 = (BOSM.means/SUM.means-BOSM.sem)*100, angle = 9
```

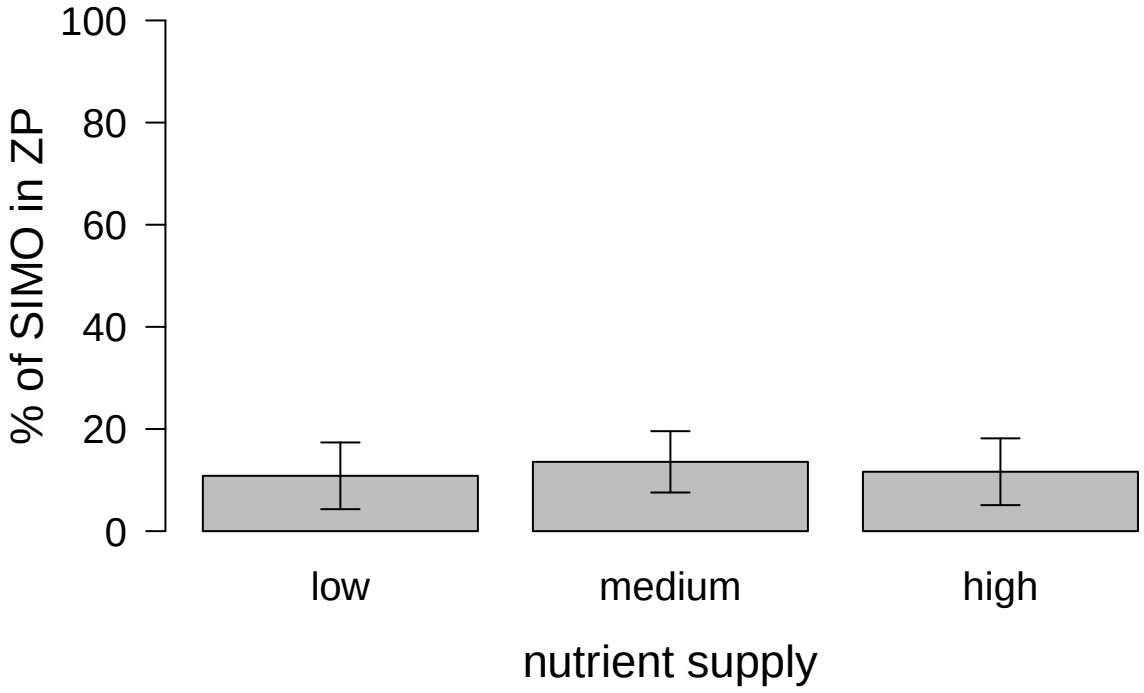
```
## Warning in arrows(x0 = bp.BOSM, y0 = BOSM.means/SUM.means * 100, y1 =
## (BOSM.means/SUM.means - : zero-length arrow is of indeterminate angle and
## so skipped
```

```
arrows(x0 = bp.BOSM, y0 = BOSM.means/SUM.means*100, y1 = (BOSM.means/SUM.means + BOSM.sem)*100, angle =
```

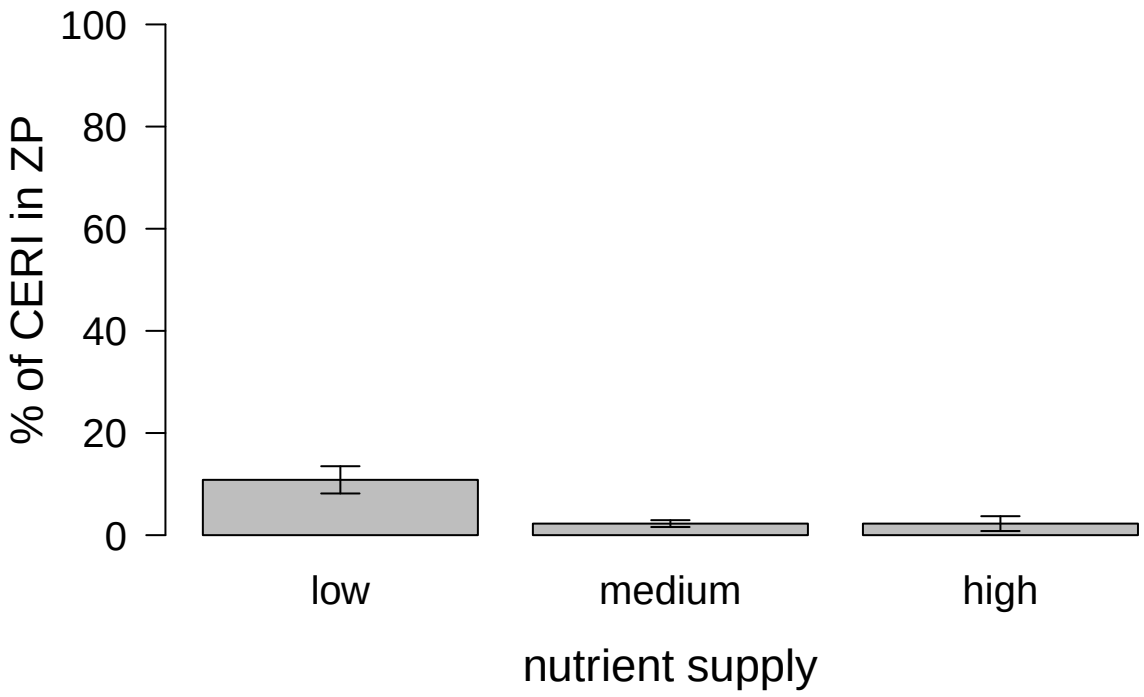
```
## Warning in arrows(x0 = bp.BOSM, y0 = BOSM.means/SUM.means * 100, y1 =
## (BOSM.means/SUM.means + : zero-length arrow is of indeterminate angle and
## so skipped
```




```
bp.SIMO <- barplot(SIMO.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab = 

arrows(x0 = bp.SIMO, y0 = SIMO.means/SUM.means*100, y1 = (SIMO.means/SUM.means-SIMO.sem)*100, angle = 9
arrows(x0 = bp.SIMO, y0 = SIMO.means/SUM.means*100, y1 = (SIMO.means/SUM.means + SIMO.sem)*100, angle = 
```




```
bp.CERI <- barplot(CERI.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab = 
arrows(x0 = bp.CERI, y0 = CERI.means/SUM.means*100, y1 = (CERI.means/SUM.means-CERI.sem)*100, angle = 9
arrows(x0 = bp.CERI, y0 = CERI.means/SUM.means*100, y1 = (CERI.means/SUM.means + CERI.sem)*100, angle = 
```




```
bp.NAUP <- barplot(NAUP.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab =

arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means*100, y1 = (NAUP.means/SUM.means-NAUP.sem)*100, angle = 9
```

```
## Warning in arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means * 100, y1 =
## (NAUP.means/SUM.means - : zero-length arrow is of indeterminate angle and
## so skipped
```

```
## Warning in arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means * 100, y1 =
## (NAUP.means/SUM.means - : zero-length arrow is of indeterminate angle and
## so skipped
```

```
## Warning in arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means * 100, y1 =
## (NAUP.means/SUM.means - : zero-length arrow is of indeterminate angle and
## so skipped
```

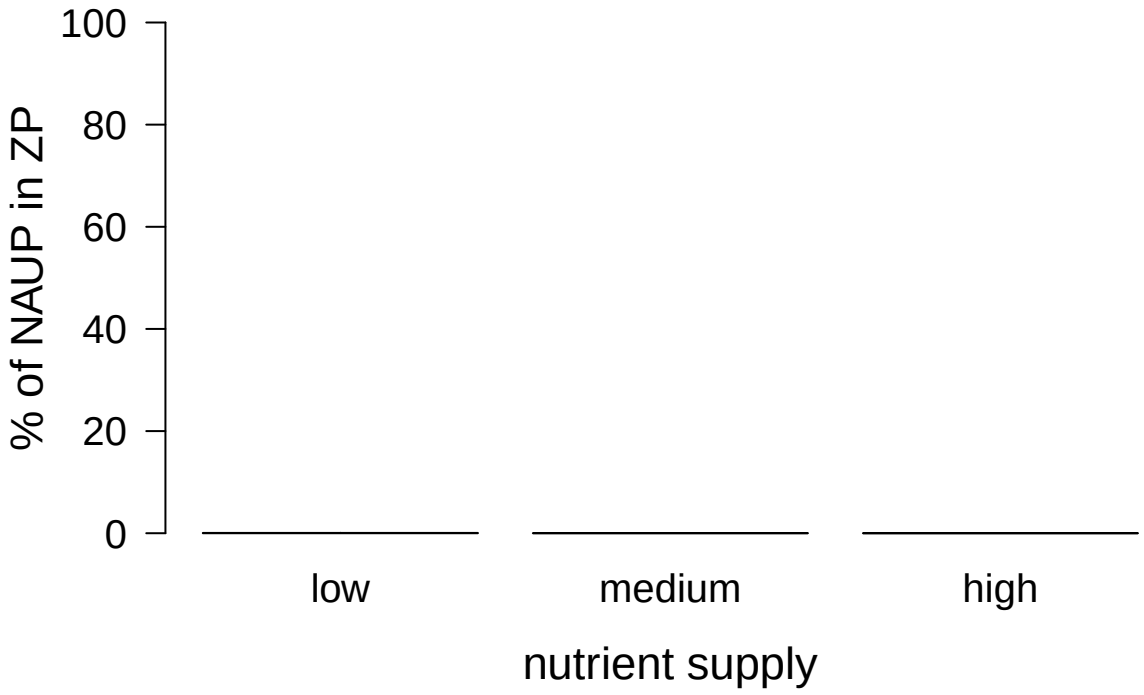
```
arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means*100, y1 = (NAUP.means/SUM.means + NAUP.sem)*100, angle =
```

```
## Warning in arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means * 100, y1 =
## (NAUP.means/SUM.means + : zero-length arrow is of indeterminate angle and
## so skipped
```

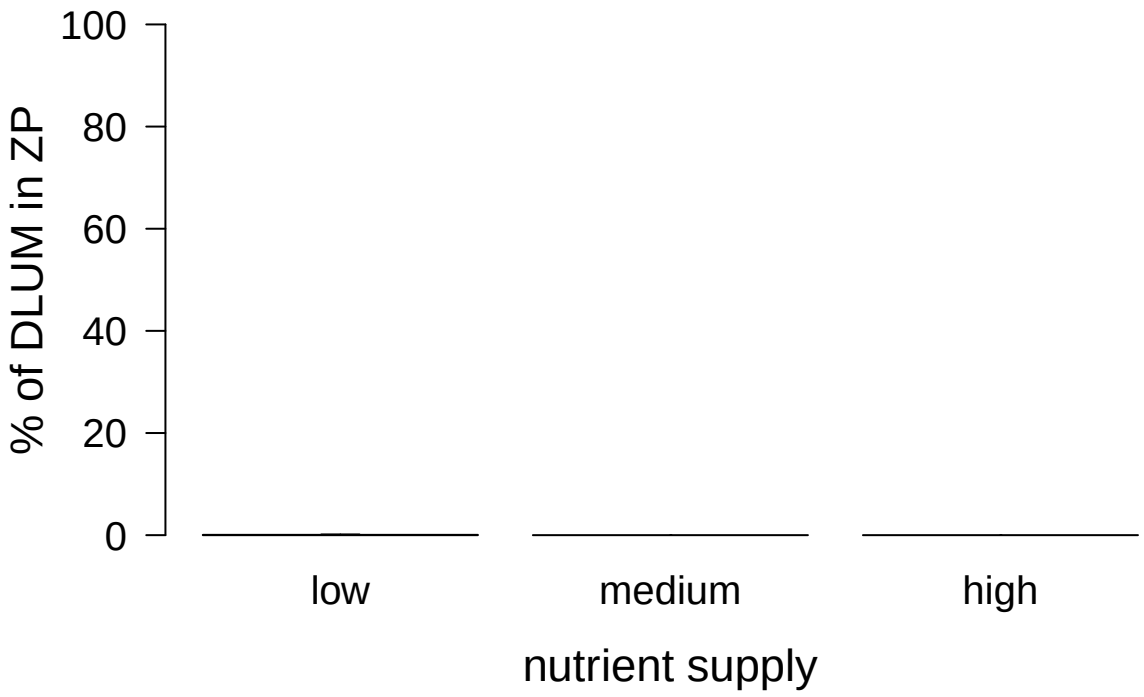
```
## Warning in arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means * 100, y1 =
## (NAUP.means/SUM.means + : zero-length arrow is of indeterminate angle and
## so skipped
```

```
## Warning in arrows(x0 = bp.NAUP, y0 = NAUP.means/SUM.means * 100, y1 =
## (NAUP.means/SUM.means + : zero-length arrow is of indeterminate angle and
```

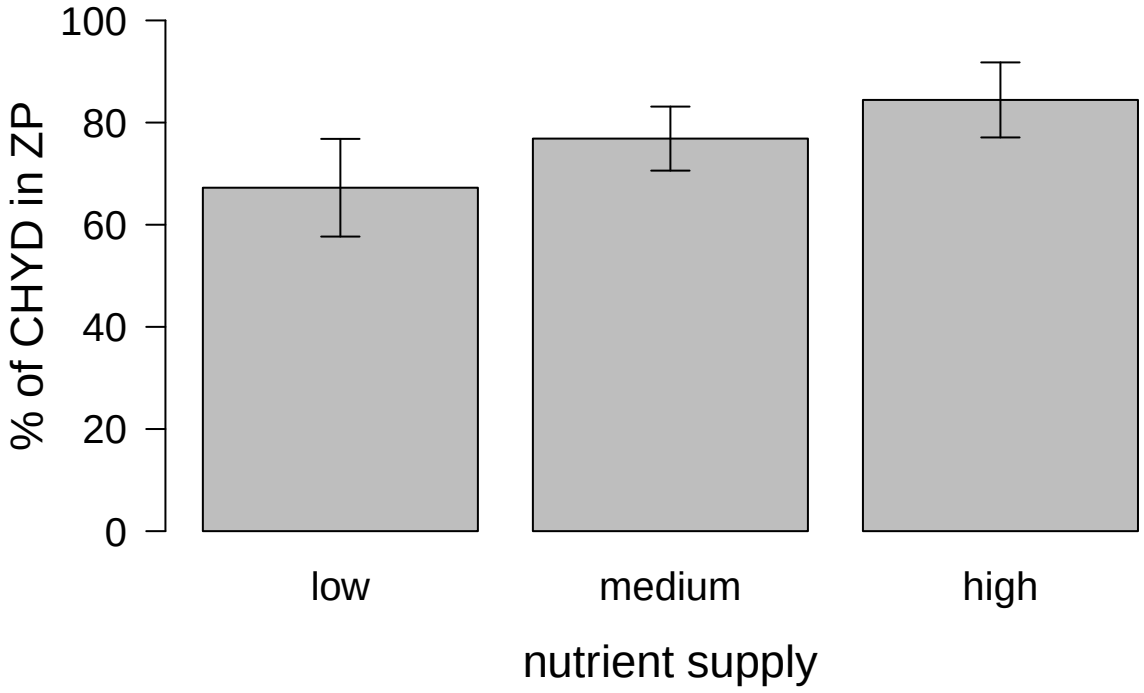
```
## so skipped
```




```
bp.DLUM <- barplot(DLUM.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab =

arrows(x0 = bp.DLUM, y0 = DLUM.means/SUM.means*100, y1 = (DLUM.means/SUM.means-DLUM.sem)*100, angle = 9

## Warning in arrows(x0 = bp.DLUM, y0 = DLUM.means/SUM.means * 100, y1 =
## (DLUM.means/SUM.means - : zero-length arrow is of indeterminate angle and
## so skipped

## Warning in arrows(x0 = bp.DLUM, y0 = DLUM.means/SUM.means * 100, y1 =
## (DLUM.means/SUM.means - : zero-length arrow is of indeterminate angle and
## so skipped

arrows(x0 = bp.DLUM, y0 = DLUM.means/SUM.means*100, y1 = (DLUM.means/SUM.means +DLUM.sem)*100, angle = 9

## Warning in arrows(x0 = bp.DLUM, y0 = DLUM.means/SUM.means * 100, y1 =
## (DLUM.means/SUM.means + : zero-length arrow is of indeterminate angle and
## so skipped

## Warning in arrows(x0 = bp.DLUM, y0 = DLUM.means/SUM.means * 100, y1 =
## (DLUM.means/SUM.means + : zero-length arrow is of indeterminate angle and
## so skipped
```




```
bp.CHYD <- barplot(CHYD.means/SUM.means*100, ylim = c(0,100), pch = 15, cex = 1.25, las = 1, cex.lab = 

arrows(x0 = bp.CHYD, y0 = CHYD.means/SUM.means*100, y1 = (CHYD.means/SUM.means-CHYD.sem)*100, angle = 90
arrows(x0 = bp.CHYD, y0 = CHYD.means/SUM.means*100, y1 = (CHYD.means/SUM.means +CHYD.sem)*100, angle = 9
```



SUBMITTING YOUR ASSIGNMENT

Use Knitr to create a PDF of your completed Week1_Assignment.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.

Unless otherwise noted, this assignment is due on **Wednesday, January 18th, 2015 at 12:00 PM (noon)**.