Water Chemistry Analysis

Samuel Dumas

Winter Semester 2024

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(lmerTest)
```

```
##
## Attaching package: 'lmerTest'
##
## The following object is masked from 'package:lme4':
##
##     lmer
##
## The following object is masked from 'package:stats':
##
##     step
```

```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.3.2
```

```
library(vegan)
```

```
## Loading required package: permute
## Loading required package: lattice
## This is vegan 2.6-4
```

```
library(gtools)
```

```
## Warning: package 'gtools' was built under R version 4.3.2

##
## Attaching package: 'gtools'
##
## The following object is masked from 'package:permute':
##
##     permute
```

Global

```
PONDS <- c("T106A1", "T106A2", "T106A3", "T106B1", "T106B2", "T106B3",
           "T108A1", "T108A2", "T108A3", "T108B1", "T108B2", "T108B3",
           "T109A1", "T109A2", "T109A3", "T109B1", "T109B2", "T109B3")
```

Water Chemistry

```
verticalData <- read_csv("Urban_Stream_2023_Site_Sample_Data_VP.csv")
```

```
## Rows: 505 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): Pond, SampleID, Date
## dbl  (6): Depth, Temperature, DO, Conductivity, Turbidity, Num
## time (1): Time In
##
## 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.
```

```
head(verticalData)
```

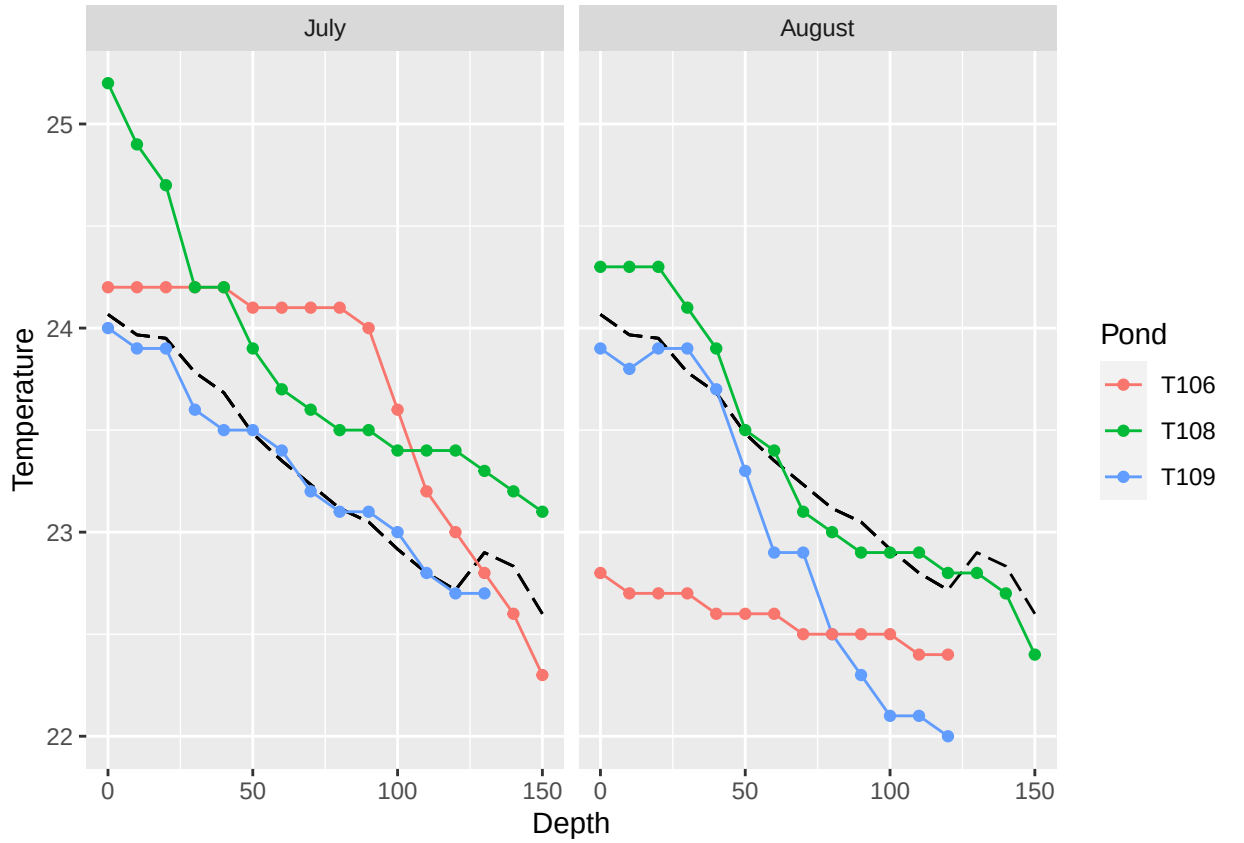
```
## # A tibble: 6 x 10
##   Pond  SampleID Date   `Time In` Depth Temperature    DO Conductivity Turbidity
##   <chr> <chr>    <chr>  <time>    <dbl>       <dbl> <dbl>        <dbl>     <dbl>
## 1 T106  T106_0   7/25/~ 09:15         0        24.2  98.2          513      15.3
```

```
## 2 T106  T106_10  7/25/~ 09:15        10        24.2  98.3          513      15.3
## 3 T106  T106_20  7/25/~ 09:15        20        24.2  98.3          513      15.3
## 4 T106  T106_30  7/25/~ 09:15        30        24.2  98.3          513      15.3
## 5 T106  T106_40  7/25/~ 09:15        40        24.2  98.8          520      16.2
## 6 T106  T106_50  7/25/~ 09:15        50        24.1  99.5          519      16.3
## # i 1 more variable: Num <dbl>
```

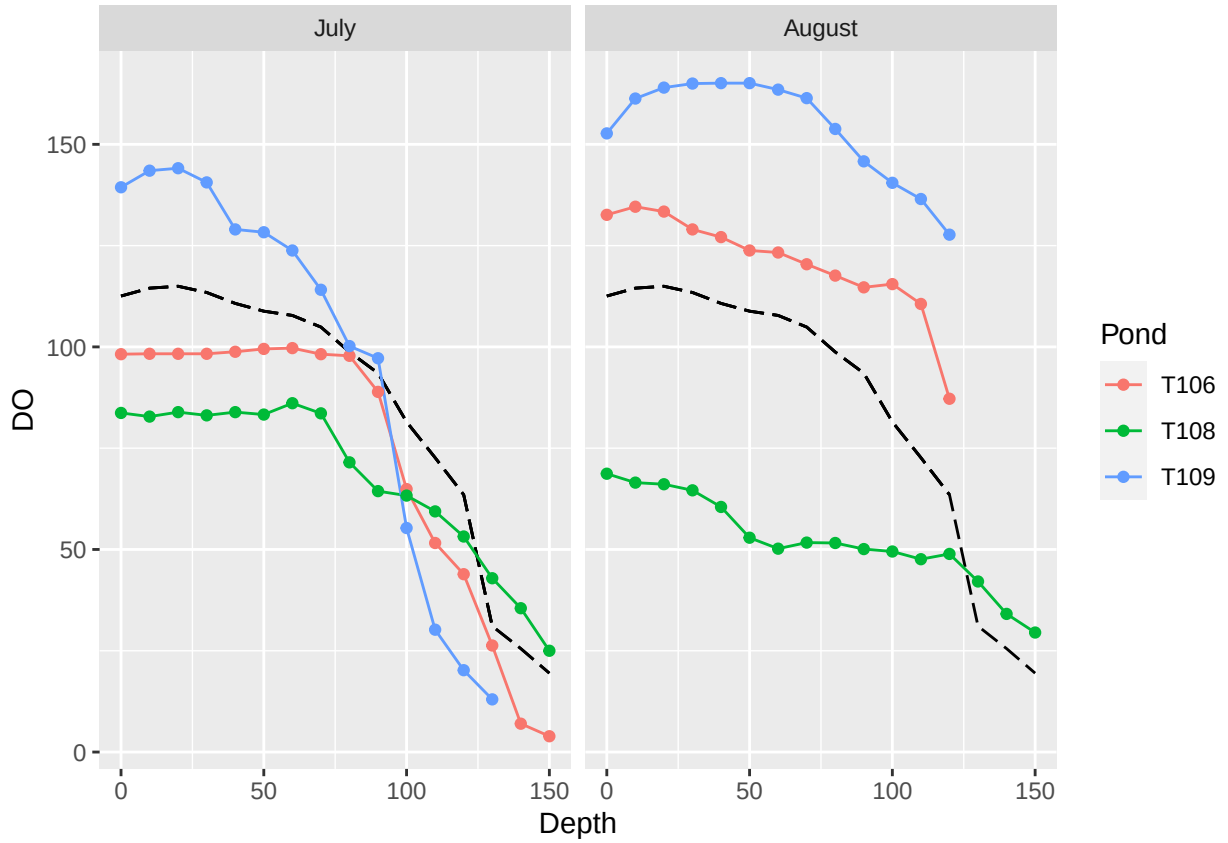
```
pondIDs <- c("T106", "T108", "T109")
verticalSubData <- verticalData %>%
  subset(Pond %in% pondIDs & Depth < 160)

numLabs <- c("July", "August")
names(numLabs) <- c("1", "2")
```

GRAPHS

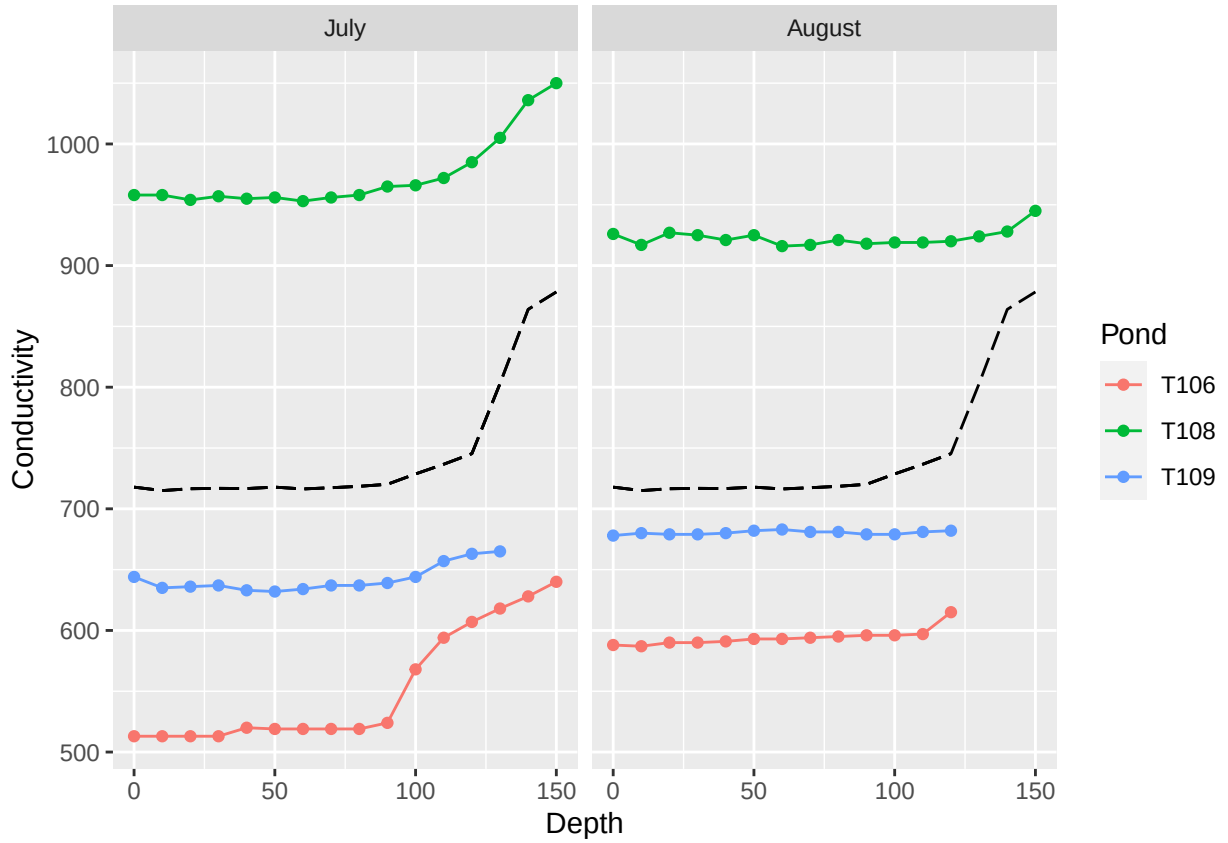
```
meanTempData <- verticalSubData %>%
  group_by(Depth) %>%
  mutate(meanTemp = mean(Temperature))

verticalSubData %>%
  ggplot(aes(x = Depth, y = Temperature, group = Pond, colour = Pond)) +
   geom_line(data = meanTempData, aes(x = Depth, y = meanTemp),
             colour = "black",
             linetype = "longdash") +
  geom_line() +
  geom_point() +
  facet_wrap(vars(Num), labeller = labeller(Num = numLabs))
```



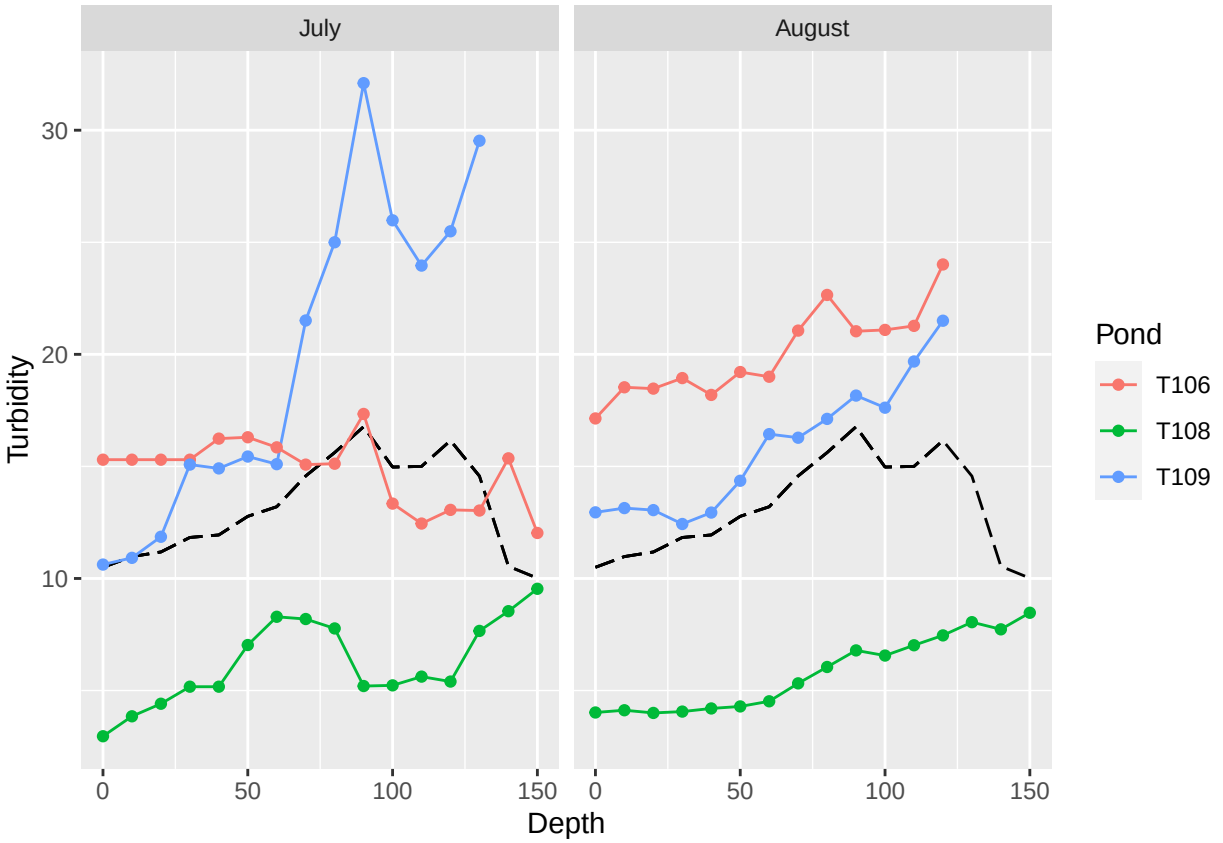

```
meanDoData <- verticalSubData %>%
  group_by(Depth) %>%
  mutate(meanDo = mean(DO))

verticalSubData %>%
  ggplot(aes(x = Depth, y = DO, group = Pond, colour = Pond)) +
   geom_line(data = meanDoData, aes(x = Depth, y = meanDo),
             colour = "black",
             linetype = "longdash") +
  geom_line() +
  geom_point() +
  facet_wrap(vars(Num), labeller = labeller(Num = numLabs))
```




```
meanCondData <- verticalSubData %>%
  group_by(Depth) %>%
  mutate(meanCond = mean(Conductivity))

verticalSubData %>%
  ggplot(aes(x = Depth, y = Conductivity, group = Pond, colour = Pond)) +
   geom_line(data = meanCondData, aes(x = Depth, y = meanCond),
             colour = "black",
             linetype = "longdash") +
  geom_line() +
  geom_point() +
  facet_wrap(vars(Num), labeller = labeller(Num = numLabs))
```




```
meanTurbData <- verticalSubData %>%
  group_by(Depth) %>%
  mutate(meanTurb = mean(Turbidity))

verticalSubData %>%
  ggplot(aes(x = Depth, y = Turbidity, group = Pond, colour = Pond)) +
   geom_line(data = meanTurbData, aes(x = Depth, y = meanTurb),
             colour = "black",
             linetype = "longdash") +
  geom_line() +
  geom_point() +
  facet_wrap(vars(Num), labeller = labeller(Num = numLabs))
```



LINEAR MODELS

```
tempDepthModel1 <- lm(Temperature ~ Depth, verticalSubData)
summary(tempDepthModel1)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = verticalSubData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.24804 -0.21066  0.00742  0.36884  1.15196
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.048037   0.108510 221.620  < 2e-16 ***
## Depth       -0.010482   0.001332  -7.869 9.73e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5414 on 86 degrees of freedom
## Multiple R-squared:  0.4186, Adjusted R-squared:  0.4118
## F-statistic: 61.92 on 1 and 86 DF,  p-value: 9.735e-12
```

```r
tempDepthModel <- function(pondID,
                           num){
  tempModel <- lm(Temperature ~ Depth, subset(verticalSubData, Pond == pondID & Num == num))
  return(summary(tempModel))
}

tempDepthModel('T106', 1)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44044 -0.20996 -0.04397  0.23103  0.50691
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.622059   0.148493 165.813  < 2e-16 ***
## Depth       -0.012544   0.001687  -7.437 3.17e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.311 on 14 degrees of freedom
## Multiple R-squared:  0.798,  Adjusted R-squared:  0.7836
## F-statistic: 55.31 on 1 and 14 DF,  p-value: 3.17e-06
```

```r
tempDepthModel('T108', 1)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3125 -0.2000 -0.0250  0.1875  0.4375
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.762500   0.113950 217.311  < 2e-16 ***
## Depth       -0.012500   0.001294  -9.657 1.44e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2387 on 14 degrees of freedom
## Multiple R-squared:  0.8695, Adjusted R-squared:  0.8602
## F-statistic: 93.26 on 1 and 14 DF,  p-value: 1.437e-07
```

```r
tempDepthModel('T109', 1)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.080440 -0.053901  0.008022  0.043791  0.114945
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 23.9942857  0.0318355  753.70  < 2e-16 ***
## Depth       -0.0104615  0.0004162  -25.13 9.54e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06278 on 12 degrees of freedom
## Multiple R-squared:  0.9814, Adjusted R-squared:  0.9798
## F-statistic: 631.7 on 1 and 12 DF,  p-value: 9.544e-12
```

```
tempDepthModel('T106', 2)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04615 -0.02308  0.00000  0.02308  0.04615
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 22.7615385  0.0164001 1387.89  < 2e-16 ***
## Depth       -0.0030769  0.0002319  -13.27 4.12e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03129 on 11 degrees of freedom
## Multiple R-squared:  0.9412, Adjusted R-squared:  0.9358
## F-statistic:   176 on 1 and 11 DF,  p-value: 4.121e-08
```

```
tempDepthModel('T108', 2)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.29677 -0.13555  0.03941  0.13257  0.24809
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.313971   0.088128  275.89  < 2e-16 ***
## Depth       -0.013103   0.001001  -13.09 3.04e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1846 on 14 degrees of freedom
## Multiple R-squared:  0.9245, Adjusted R-squared:  0.9191
## F-statistic: 171.3 on 1 and 14 DF,  p-value: 3.044e-09
```

```
tempDepthModel('T109', 2)
```

```
##
## Call:
## lm(formula = Temperature ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.26703 -0.15110  0.03022  0.11429  0.30495
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.167033   0.100845  239.65  < 2e-16 ***
## Depth       -0.019066   0.001426  -13.37  3.8e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1924 on 11 degrees of freedom
## Multiple R-squared:  0.942,  Adjusted R-squared:  0.9367
## F-statistic: 178.7 on 1 and 11 DF,  p-value: 3.805e-08
```

```
doDepthModel1 <- lm(DO ~ Depth, verticalSubData)
summary(doDepthModel1)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = verticalSubData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -62.648 -25.035  -8.623  24.576  70.573
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 131.34791    7.06591  18.589  < 2e-16 ***
## Depth        -0.57887    0.08674  -6.673 2.32e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.26 on 86 degrees of freedom
## Multiple R-squared:  0.3412, Adjusted R-squared:  0.3335
## F-statistic: 44.53 on 1 and 86 DF,  p-value: 2.32e-09
```

```r
doDepthModel <- function(pondID,
                         num){
  tempModel <- lm(DO ~ Depth, subset(verticalSubData, Pond == pondID & Num == num))
  return(summary(tempModel))
}

doDepthModel('T106', 1)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = subset(verticalSubData, Pond ==
##     pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -24.1076 -12.8029   0.3953  11.5776  27.6994
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 122.09118    8.52081  14.329 9.31e-10 ***
## Depth        -0.64988    0.09679  -6.714 9.87e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.85 on 14 degrees of freedom
## Multiple R-squared:  0.763,  Adjusted R-squared:  0.7461
## F-statistic: 45.08 on 1 and 14 DF,  p-value: 9.873e-06
```

```r
doDepthModel('T108', 1)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = subset(verticalSubData, Pond ==
##     pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.568  -5.477   2.263   5.042  13.865
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 96.13162    4.12437  23.308 1.34e-12 ***
## Depth       -0.37709    0.04685  -8.049 1.28e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.639 on 14 degrees of freedom
## Multiple R-squared:  0.8223, Adjusted R-squared:  0.8096
## F-statistic: 64.78 on 1 and 14 DF,  p-value: 1.276e-06
```

```r
doDepthModel('T109', 1)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = subset(verticalSubData, Pond ==
##     pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.7314 -15.3702   0.5595  16.7674  25.4912
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 168.1314     9.3907  17.904 5.06e-10 ***
## Depth        -1.0714     0.1228  -8.726 1.53e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.52 on 12 degrees of freedom
## Multiple R-squared:  0.8639, Adjusted R-squared:  0.8525
## F-statistic: 76.14 on 1 and 12 DF,  p-value: 1.528e-06
```

```
doDepthModel('T106', 2)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = subset(verticalSubData, Pond ==
##     pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1044  -0.4786   1.0132   2.6626   6.3791
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 138.20330    3.08050  44.864 8.24e-14 ***
## Depth        -0.29082    0.04356  -6.676 3.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.877 on 11 degrees of freedom
## Multiple R-squared:  0.802,  Adjusted R-squared:  0.784
## F-statistic: 44.56 on 1 and 11 DF,  p-value: 3.488e-05
```

```
doDepthModel('T108', 2)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = subset(verticalSubData, Pond ==
##     pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8735 -2.0643  0.5297  2.2781  6.8109
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 68.95147    1.72421   39.99 7.80e-16 ***
## Depth       -0.22385    0.01959  -11.43 1.74e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.611 on 14 degrees of freedom
## Multiple R-squared:  0.9032, Adjusted R-squared:  0.8963
## F-statistic: 130.6 on 1 and 14 DF,  p-value: 1.743e-08
```

```
doDepthModel('T109', 2)
```

```
##
## Call:
## lm(formula = DO ~ Depth, data = subset(verticalSubData, Pond ==
##     pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.2286  -5.1159   0.0374   6.1033   9.8522
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 168.92857    4.44978  37.963 5.12e-13 ***
## Depth        -0.24830    0.06293  -3.946  0.00229 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.49 on 11 degrees of freedom
## Multiple R-squared:  0.586,  Adjusted R-squared:  0.5483
## F-statistic: 15.57 on 1 and 11 DF,  p-value: 0.00229
```

```
turbDepthModel1 <- lm(Turbidity ~ Depth, verticalSubData)
summary(turbDepthModel1)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = verticalSubData)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.279 -7.149  1.196  4.523 18.215
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.50145    1.33960   8.586 3.43e-13 ***
## Depth        0.02648    0.01645   1.610    0.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.684 on 86 degrees of freedom
## Multiple R-squared:  0.02926,    Adjusted R-squared:  0.01797
## F-statistic: 2.592 on 1 and 86 DF,  p-value: 0.1111
```

```r
turbDepthModel <- function(pondID,
                           num){
  tempModel <- lm(Turbidity ~ Depth, subset(verticalSubData, Pond == pondID & Num == num))
  return(summary(tempModel))
}

turbDepthModel('T106', 1)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6547 -0.8678 -0.4325  0.7895  2.8523
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.211250   0.603653  26.855 1.92e-13 ***
## Depth       -0.019150   0.006857  -2.793   0.0144 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.264 on 14 degrees of freedom
## Multiple R-squared:  0.3578, Adjusted R-squared:  0.3119
## F-statistic: 7.799 on 1 and 14 DF,  p-value: 0.01438
```

```r
turbDepthModel('T108', 1)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0560 -1.3271 -0.1045  1.3071  2.4395
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.245074   0.700315   6.062 2.93e-05 ***
## Depth       0.026757   0.007955   3.364  0.00464 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.467 on 14 degrees of freedom
## Multiple R-squared:  0.4469, Adjusted R-squared:  0.4074
## F-statistic: 11.31 on 1 and 14 DF,  p-value: 0.004637
```

```r
turbDepthModel('T109', 1)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9386 -1.7742 -0.3792  0.8639  8.3646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.64514    1.60117   6.024 5.99e-05 ***
## Depth        0.15656    0.02093   7.479 7.44e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.158 on 12 degrees of freedom
## Multiple R-squared:  0.8233, Adjusted R-squared:  0.8086
## F-statistic: 55.93 on 1 and 12 DF,  p-value: 7.445e-06
```

```
turbDepthModel('T106', 2)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0454 -0.7672 -0.1876  0.5616  1.6987
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.327582   0.485142  35.717 9.97e-13 ***
## Depth        0.045297   0.006861   6.602 3.85e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9256 on 11 degrees of freedom
## Multiple R-squared:  0.7985, Adjusted R-squared:  0.7802
## F-statistic: 43.59 on 1 and 11 DF,  p-value: 3.853e-05
```

```
turbDepthModel('T108', 2)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.76950 -0.25262  0.05325  0.23225  0.73750
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.282500   0.209818   15.64 2.91e-10 ***
## Depth       0.033450   0.002383   14.04 1.22e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4395 on 14 degrees of freedom
## Multiple R-squared:  0.9336, Adjusted R-squared:  0.9289
## F-statistic:   197 on 1 and 14 DF,  p-value: 1.223e-09
```

```
turbDepthModel('T109', 2)
```

```
##
## Call:
## lm(formula = Turbidity ~ Depth, data = subset(verticalSubData,
##     Pond == pondID & Num == num))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.45495 -0.74786  0.08088  0.61923  1.40670
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.543297   0.516306  22.357 1.61e-10 ***
## Depth        0.071291   0.007302   9.764 9.38e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.985 on 11 degrees of freedom
## Multiple R-squared:  0.8965, Adjusted R-squared:  0.8871
## F-statistic: 95.33 on 1 and 11 DF,  p-value: 9.384e-07
```

Invertebrates Data

```
invertData <- read.csv("EEB397_InvertData.csv", row.names = 1)
```

```
rank.abund <- function(x, num) {
  tmp <- x[num, -c(1,2)]
  tmp2 <- tmp[tmp > 0] / sum(tmp)
  tmp3 <- tmp2[order(-tmp2)]
cbind(rank = 1:length(tmp3), RA = tmp3) %>%
  as.data.frame() %>%
ggplot(aes(x = rank, y = RA)) +
  geom_col(colour ="blue", fill = "lightblue") +
  labs(y = "Relative Abundance", x = "Rank", title = paste("Pond", PONDS[num])) +
  ylim(0,1)
}
```

```
A1 <- rank.abund(invertData, 1)
A2 <- rank.abund(invertData, 2)
A3 <- rank.abund(invertData, 3)
B1 <- rank.abund(invertData, 4)
B2 <- rank.abund(invertData, 5)
B3 <- rank.abund(invertData, 6)
A4 <- rank.abund(invertData, 7)
A5 <- rank.abund(invertData, 8)
A6 <- rank.abund(invertData, 9)
B4 <- rank.abund(invertData, 10)
B5 <- rank.abund(invertData, 11)
B6 <- rank.abund(invertData, 12)
A7 <- rank.abund(invertData, 13)
A8 <- rank.abund(invertData, 14)
A9 <- rank.abund(invertData, 15)
B7 <- rank.abund(invertData, 16)
B8 <- rank.abund(invertData, 17)
B9 <- rank.abund(invertData, 18)
```

```
ggarrange(A1, A2, A3, ncol = 3) +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 2A. Rank-abundance distributions for the pond T106 site A.
")
```

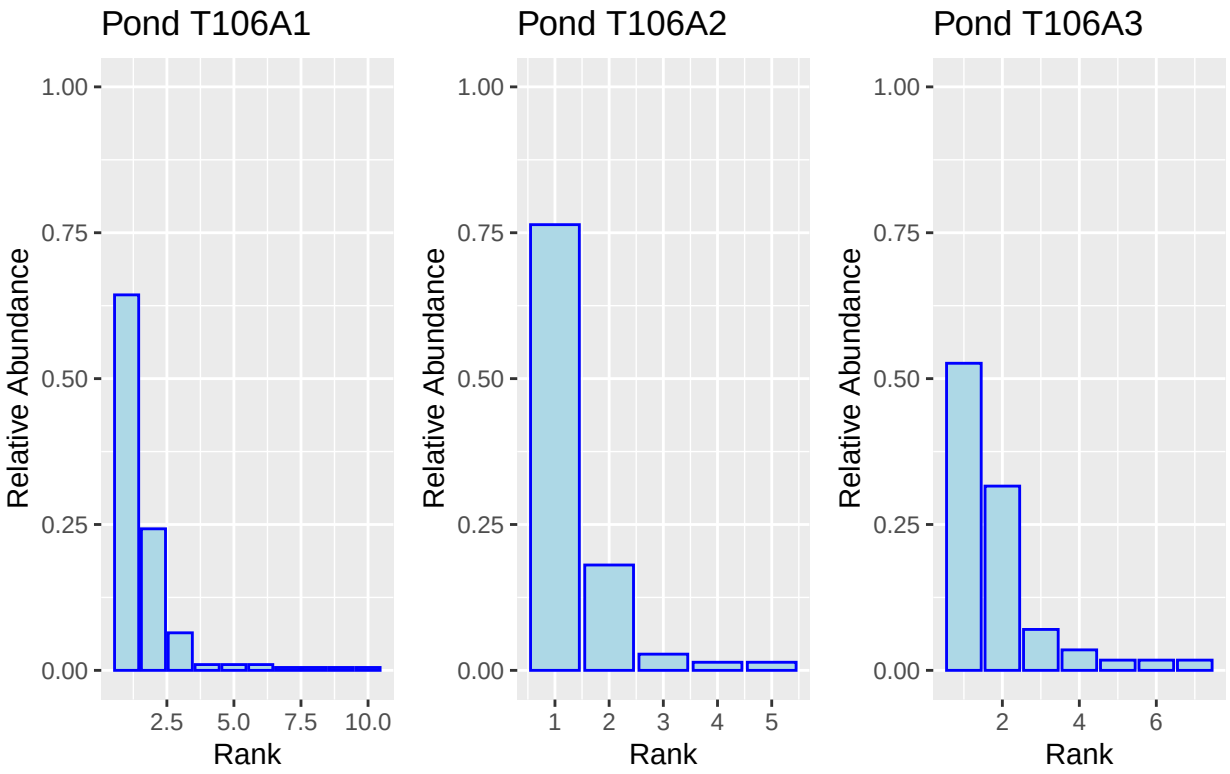


Figure 2A. Rank−abundance distributions for the pond T106 site A.

```
ggsave("Plots/rankdistT106A.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggarrange(A4, A5, A6, ncol = 3) +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 2B. Rank-abundance distributions for the pond T108 site A.
")
```

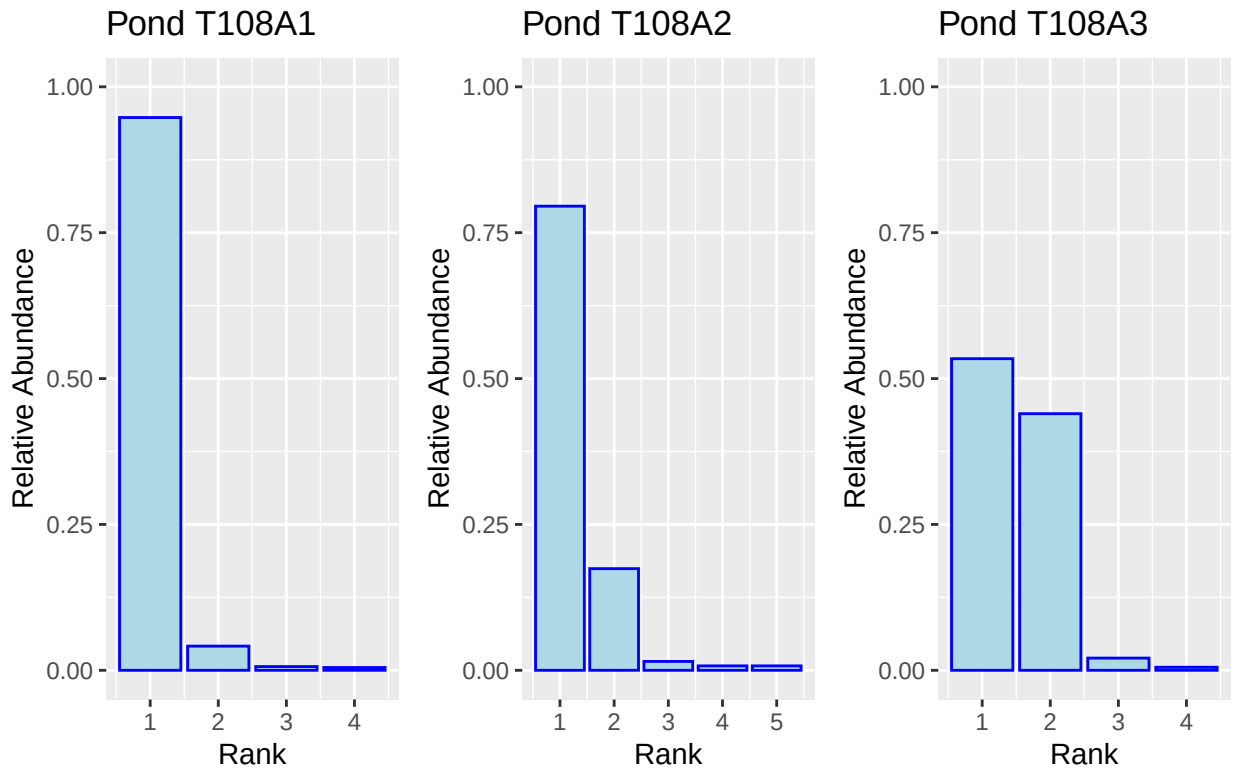


Figure 2B. Rank−abundance distributions for the pond T108 site A.

```
ggsave("Plots/rankdistT108A.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggarrange(A7, A8, A9, ncol = 3) +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 2C. Rank-abundance distributions for the pond T109 site A.
")
```

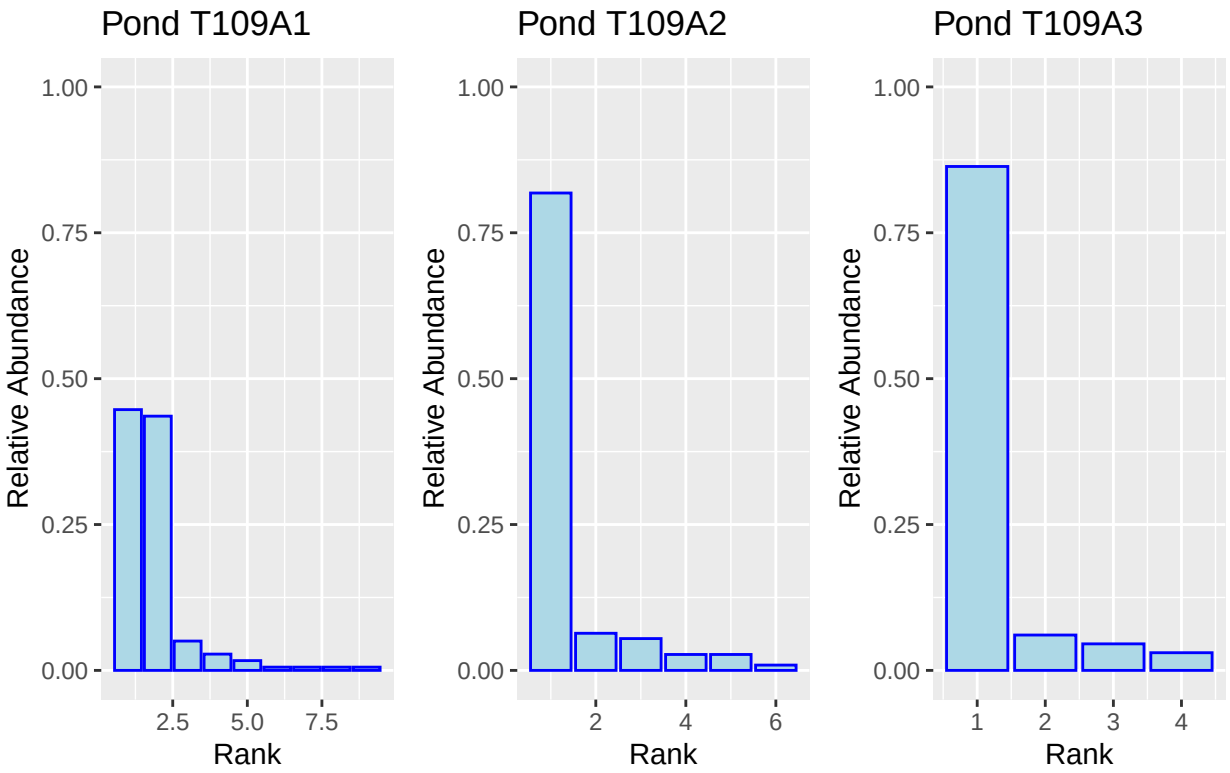


Figure 2C. Rank−abundance distributions for the pond T109 site A.

```
ggsave("Plots/rankdistT109A.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggarrange(B1, B2, B3, ncol = 3) +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 2D. Rank-abundance distributions for the pond T106 site B.
")
```

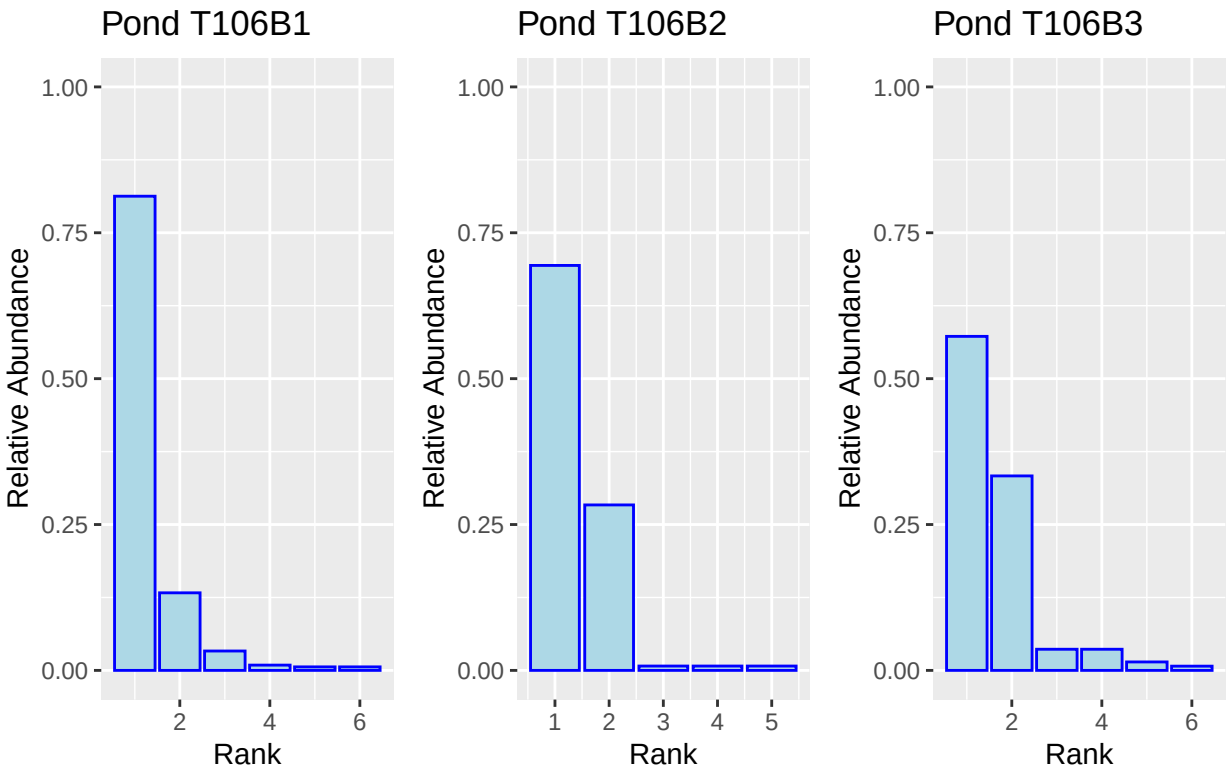


Figure 2D. Rank−abundance distributions for the pond T106 site B.

```
ggsave("Plots/rankdistT106B.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggarrange(B4, B5, B6, ncol = 3) +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 2E. Rank-abundance distributions for the pond T108 site B.
")
```

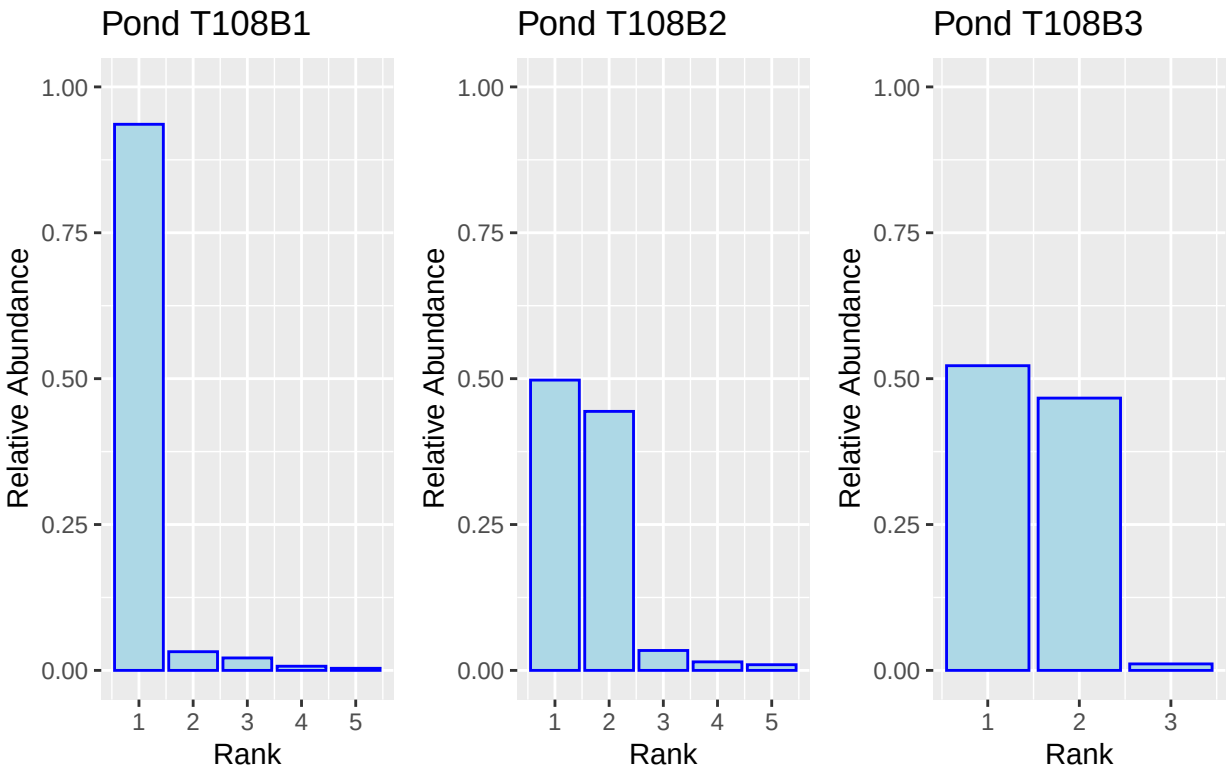


Figure 2E. Rank−abundance distributions for the pond T108 site B.

```
ggsave("Plots/rankdistT108B.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggarrange(B7, B8, B9, ncol = 3) +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 2F. Rank-abundance distributions for the pond T109 site B.
")
```

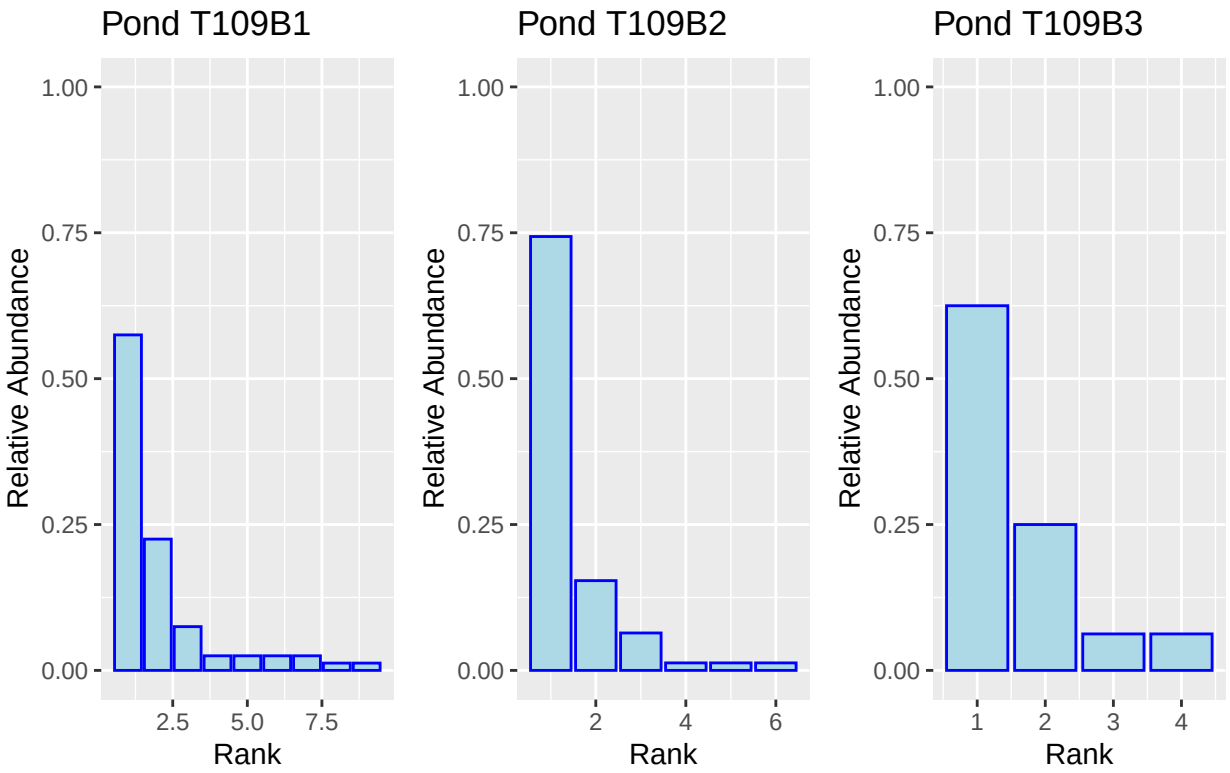


Figure 2F. Rank−abundance distributions for the pond T109 site B.

```
ggsave("Plots/rankdistT109B.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
invertData %>%
  mutate(nSpecies=rowSums(.!=0) - 2) %>%
  ggplot(aes(x = nSpecies)) +
  geom_bar(colour = "blue", fill = "lightblue") +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(x = "Number of Species", y = "Frequency", title = "Frequency versus Number of Species",
       caption = "Figure 3A. Frequency distribution of number of species found in each sample (n = 18).
```

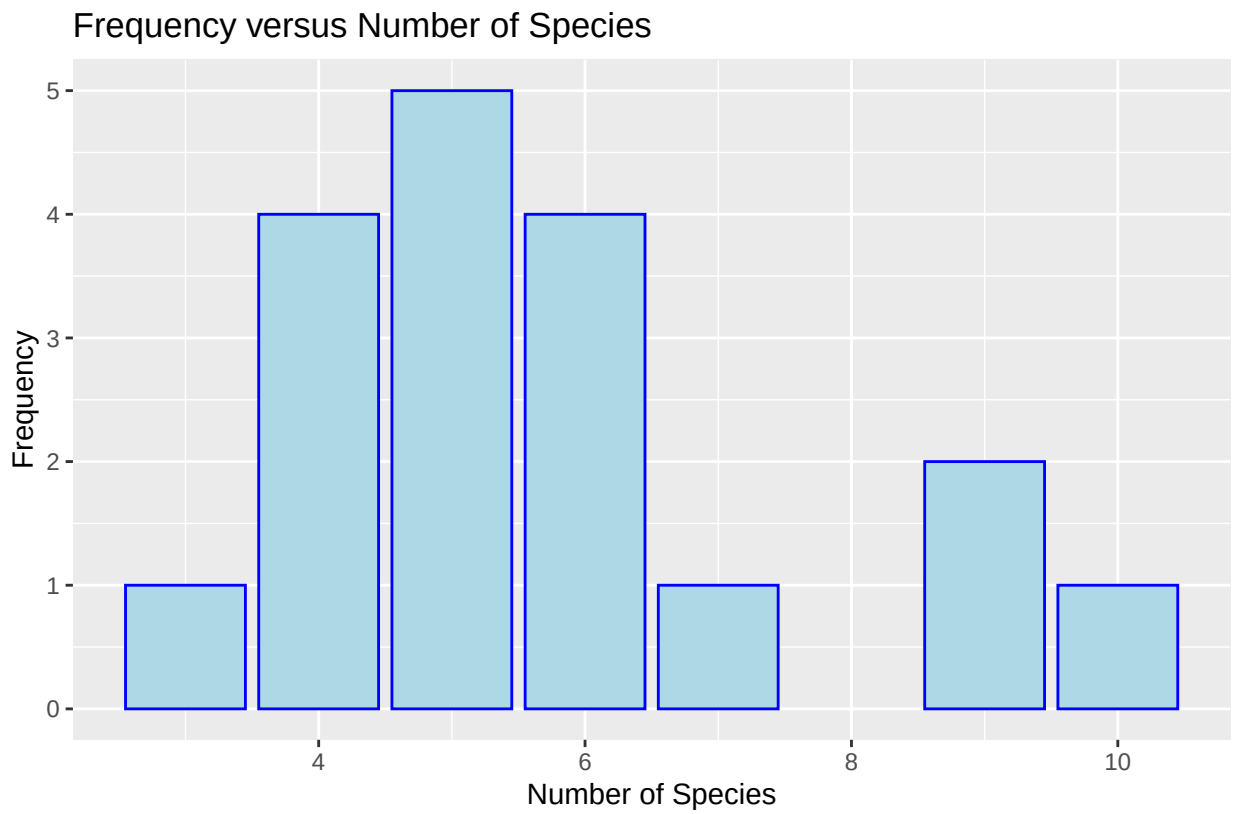


Figure 3A. Frequency distribution of number of species found in each sample (n = 18).

```
ggsave("Plots/speciesdistribution.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
invertData %>%
  mutate(nSpecies=rowSums(.!=0) - 2) %>%
  ggplot(aes(x= nSpecies, fill = factor(depth), group = depth)) +
    geom_bar(colour = "black", position = "stack") +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(x = "Number of Species", y = "Frequency",
       fill = "Depth (cm)",
       title = "Frequency versus Number of Species",
       caption = "Figure 3B. Frequency distribution of number of species found in each sample (n = 18),
```

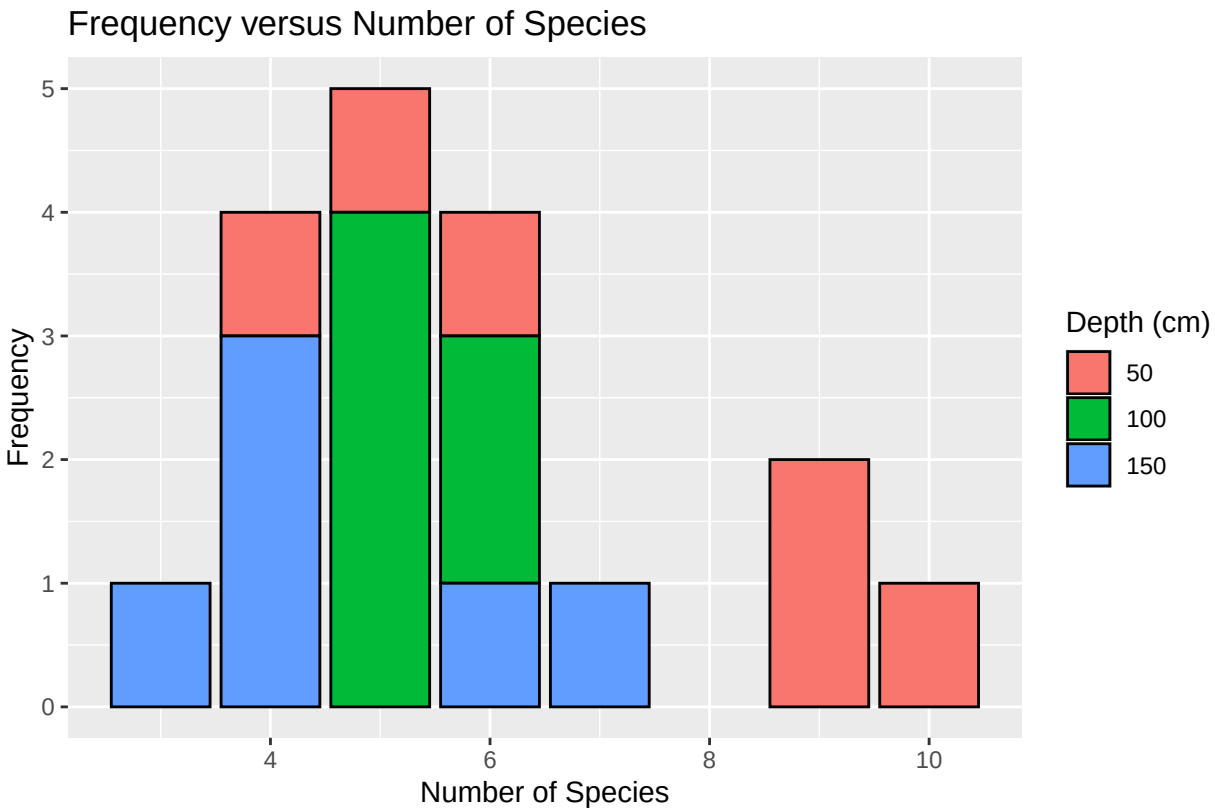


Figure 3B. Frequency distribution of number of species found in each sample (n = 18), separated by depth.

```
ggsave("Plots/speciesdistributionD.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
invertData %>%
  pivot_longer(!c(pondID, depth), values_to = "total", names_to = "family") %>%
  ggplot(aes(x=family, y = total, fill=family)) +
         geom_bar(stat = "identity", position = "fill") +
    theme(axis.text.x=element_blank(),
          axis.ticks.x=element_blank(),
          axis.text.y=element_blank(),
          axis.ticks.y=element_blank(),
          plot.caption = element_text(hjust = 0)) +
  labs(x = "Present Taxonomic Families", y="", title = "Species Richness across Ponds.",
       fill = "Taxonomic Families",
       caption = "Figure 4A. Taxonomic families present in each sample.") +
  guides(fill=guide_legend(ncol=2)) +
         facet_wrap(vars(pondID), nrow = 6, ncol =3)
```

```
## Warning: Removed 257 rows containing missing values (‘geom_bar()‘).
```

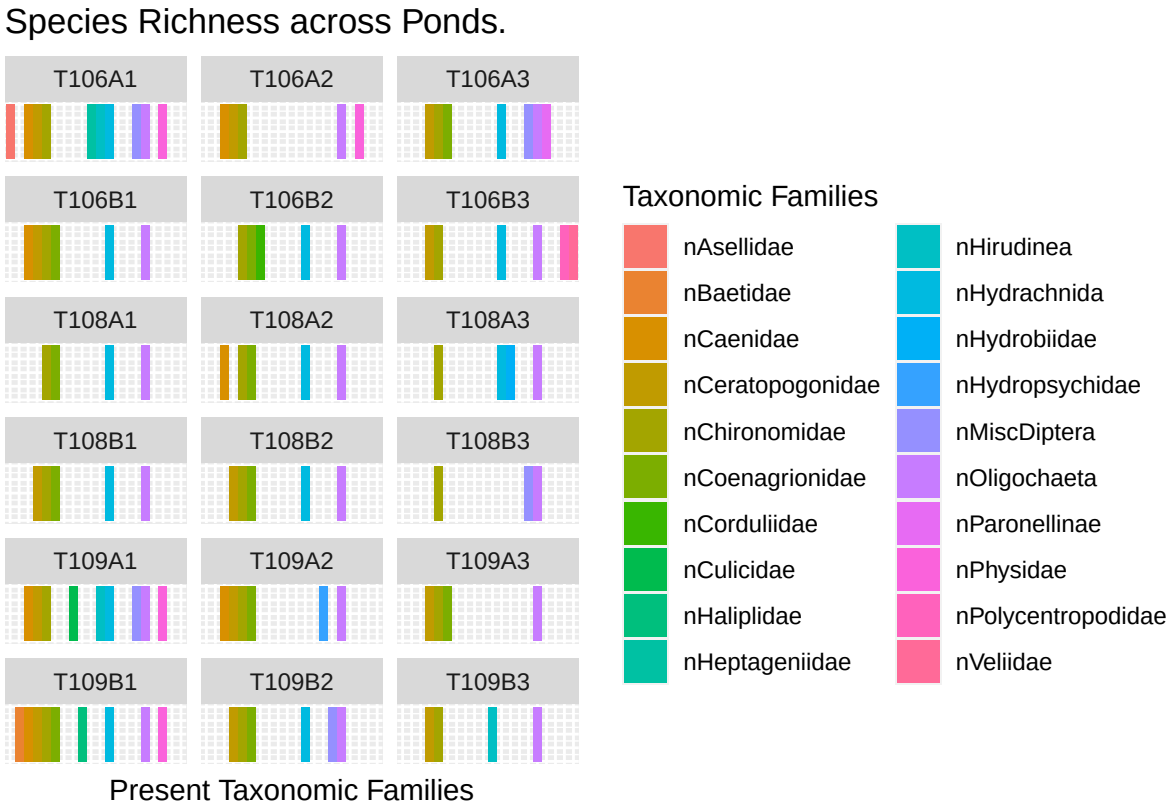


Figure 4A. Taxonomic families present in each sample.

```
ggsave("Plots/familypresence.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 257 rows containing missing values (‘geom_bar()‘).
```

```
invertData %>%
  pivot_longer(!c(pondID, depth), values_to = "total", names_to = "family") %>%
  ggplot(aes(x=pondID, y = total, fill=family)) +
         geom_bar(stat = "identity", position = "fill") +
    theme(axis.text.x=element_text(angle = 45, hjust = 1),
          plot.caption = element_text(hjust = 0)) +
  labs(x = "Present Taxonomic Families", y="Relative Abundance", title = "Species Richness across Ponds
       fill = "Taxonomic Families",
       caption = "Figure 4B. Relative abundance of taxonomic families by sample.") +
  guides(fill=guide_legend(ncol=2))
```

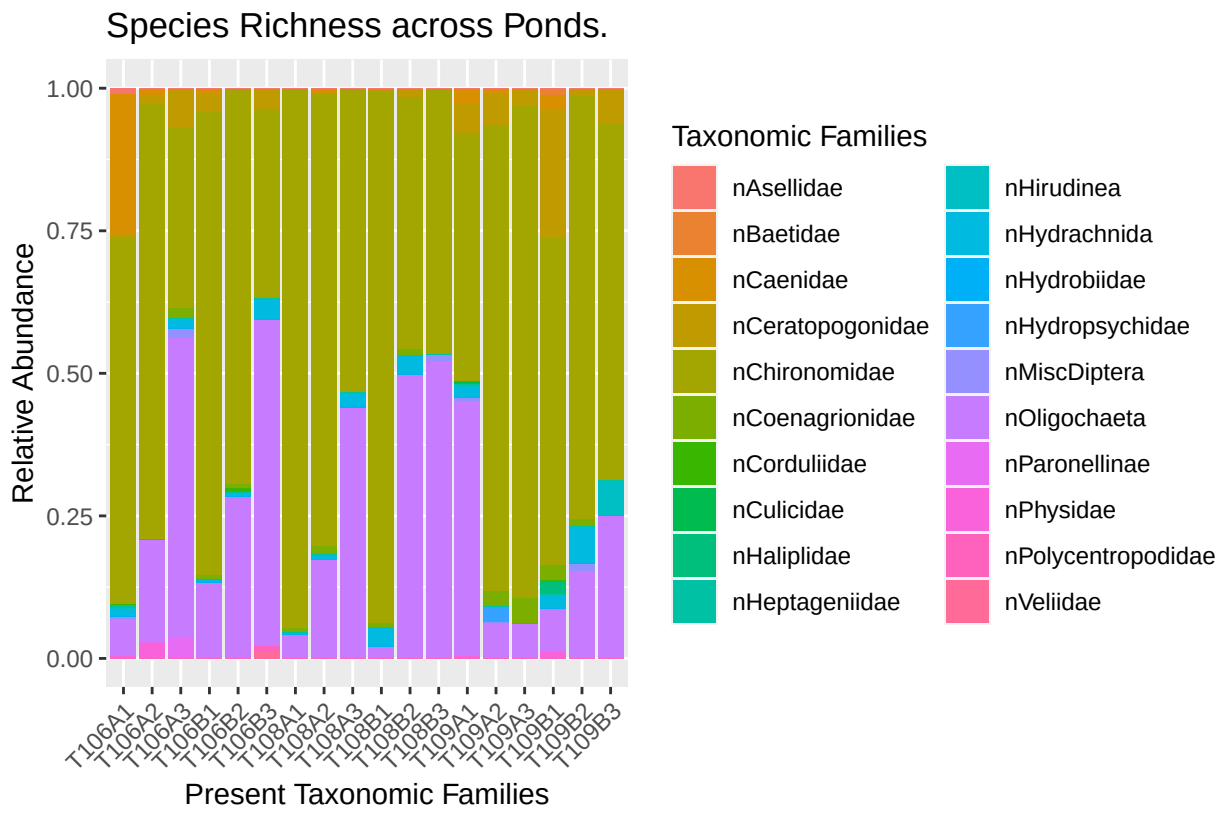


Figure 4B. Relative abundance of taxonomic families by sample.

```
ggsave("Plots/familyrelative.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
SR <- specnumber(invertData[,-c(1,2)])
simpInv <- diversity(invertData[,-c(1,2)], index = "invsimpson")
simpEven <- simpInv / SR
```

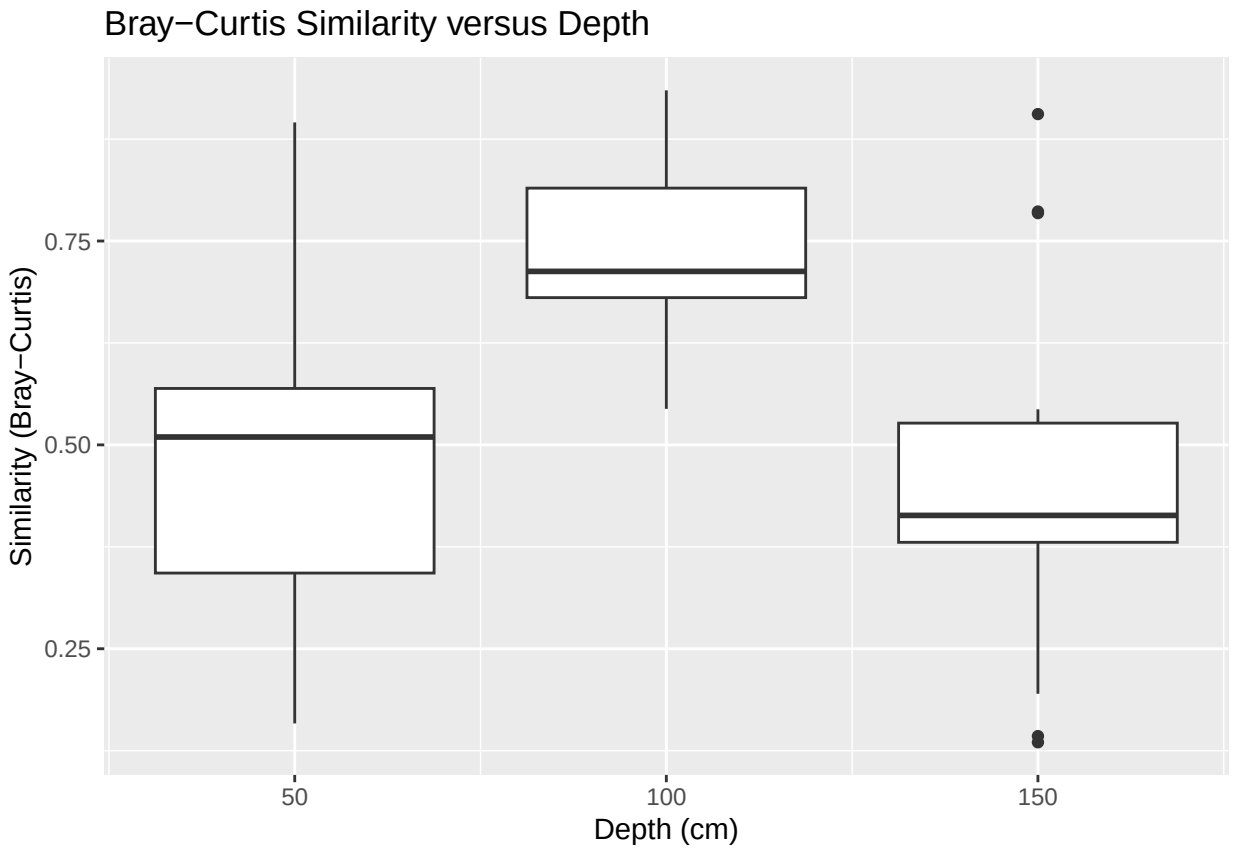
```
brayCurtisInv <- 1 - vegdist(invertData[,-c(1,2)], method = "bray", diag = FALSE) %>%
  as.matrix()
```

Bray Curtis By Depth

```
mod3index1 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(1, 18, 3)) %>%
  as.data.frame()

mod3index2 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(2, 18, 3)) %>%
  as.data.frame()

mod3index3 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(3, 18, 3)) %>%
  as.data.frame()
```

```
brayByDepth <- c()
for (i in 1:length(mod3index1$V1)){
  brayByDepth <- rbind(brayByDepth,
                       c(50, brayCurtisInv[mod3index1$V1[i], mod3index1$V2[i]]))
}


temp2 <- 0
for (i in 1:length(mod3index2$V1)){
  brayByDepth <- rbind(brayByDepth,
                       c(100, brayCurtisInv[mod3index2$V1[i], mod3index2$V2[i]]))
}


temp3 <- 0
for (i in 1:length(mod3index3$V1)){
  brayByDepth <- rbind(brayByDepth,
                       c(150, brayCurtisInv[mod3index3$V1[i], mod3index3$V2[i]]))
}

brayByDepth <- as.data.frame(brayByDepth)
colnames(brayByDepth) <- c("depth", "bcSim")
```

```
brayByDepth %>%
  ggplot(aes(x = depth, y = bcSim, group = depth)) +
  geom_boxplot() +
  labs(x = "Depth (cm)", y = "Similarity (Bray-Curtis)", title = "Bray-Curtis Similarity versus Depth")
```



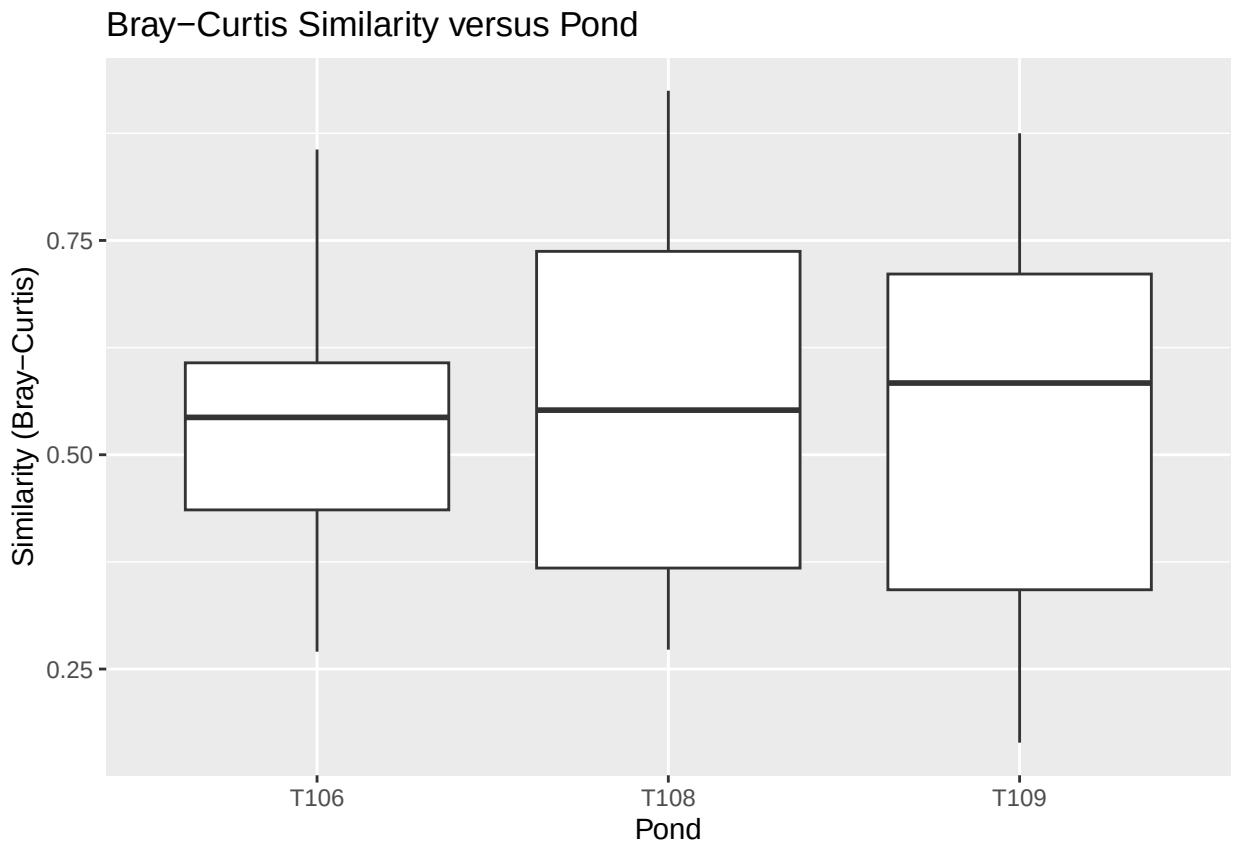
Bray-Curtis By Pond

```
ord3index1 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 1:6) %>%
  as.data.frame()

ord3index2 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 7:12) %>%
  as.data.frame()

ord3index3 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 13:18) %>%
  as.data.frame()
```

```
brayByPond <- c()
for (i in 1:length(ord3index1$V1)){
  brayByPond <- rbind(brayByPond,
                      c("T106", brayCurtisInv[ord3index1$V1[i], ord3index1$V2[i]]))
}


for (i in 1:length(ord3index2$V1)){
  brayByPond <- rbind(brayByPond,
                      c("T108", brayCurtisInv[ord3index2$V1[i], ord3index2$V2[i]]))
}
```

```
for (i in 1:length(ord3index3$V1)){
  brayByPond <- rbind(brayByPond,
                      c("T109", brayCurtisInv[ord3index3$V1[i], ord3index3$V2[i]]))
}

brayByPond <- as.data.frame(brayByPond)
colnames(brayByPond) <- c("pond", "bcSim")
brayByPond$bcSim <- as.numeric(brayByPond$bcSim)
```

```
brayByPond %>%
  ggplot(aes(x = pond, y = bcSim, group = pond)) +
  geom_boxplot() +
  labs(x = "Pond", y = "Similarity (Bray-Curtis)", title = "Bray-Curtis Similarity versus Pond")
```



For real

Water Chemistry

```
tempModel <- lmer(Temperature ~ Depth + Num + (1|Pond), data = verticalSubData, REML = FALSE)
summary(tempModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
```

```
##   method [lmerModLmerTest]
## Formula: Temperature ~ Depth + Num + (1 | Pond)
##    Data: verticalSubData
##
##      AIC      BIC   logLik deviance df.resid
##     72.3     84.7    -31.1     62.3       83
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.44639 -0.67329 -0.08201  0.67076  2.26160
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Pond     (Intercept) 0.06059  0.2462
##  Residual             0.10776  0.3283
## Number of obs: 88, groups:  Pond, 3
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept) 25.1573248  0.1902827  8.4452081  132.21 2.65e-15 ***
## Depth       -0.0116803  0.0008152 85.1458297  -14.33  < 2e-16 ***
## Num         -0.7057149  0.0703192 85.0305220  -10.04 4.34e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##       (Intr) Depth
## Depth -0.335
## Num   -0.568  0.075
```

```
doModel <- lmer(DO ~ Depth + Num + (1|Pond), data = verticalSubData, REML = FALSE)
summary(doModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: DO ~ Depth + Num + (1 | Pond)
##    Data: verticalSubData
##
##      AIC      BIC   logLik deviance df.resid
##    820.8    833.2   -405.4    810.8       83
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.9139 -0.4827  0.2140  0.7720  1.4060
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Pond     (Intercept) 574.6    23.97
##  Residual             521.3    22.83
## Number of obs: 88, groups:  Pond, 3
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept) 92.27598   16.40070  5.53775   5.626  0.00176 **
```

```
## Depth       -0.49511    0.05671 85.07137  -8.730 1.89e-13 ***
## Num         23.65512    4.89092 85.01029   4.837 5.82e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##       (Intr) Depth
## Depth -0.270
## Num   -0.458  0.075
```

```
condModel <- lmer(Conductivity ~ Depth + Num + (1|Pond), data = verticalSubData, REML = FALSE)
summary(condModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: Conductivity ~ Depth + Num + (1 | Pond)
##    Data: verticalSubData
##
##      AIC      BIC   logLik deviance df.resid
##    879.8    892.2   -434.9    869.8       83
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.7087 -0.8106  0.2058  0.7481  2.7194
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Pond     (Intercept) 25478.5  159.62
##  Residual               913.8   30.23
## Number of obs: 88, groups:  Pond, 3
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept) 691.2274    92.8903   3.0894   7.441  0.00454 **
## Depth         0.3210     0.0751  85.0032   4.274 4.99e-05 ***
## Num           9.0647     6.4757  85.0007   1.400  0.16521
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##       (Intr) Depth
## Depth -0.063
## Num   -0.107  0.075
```

```
turbModel <- lmer(Turbidity ~ Depth + Num + (1|Pond), data = verticalSubData, REML = FALSE)
summary(turbModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: Turbidity ~ Depth + Num + (1 | Pond)
##    Data: verticalSubData
##
##      AIC      BIC   logLik deviance df.resid
```

```
##    488.7    501.1   -239.3    478.7       83
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.3949 -0.5042 -0.1218  0.5652  3.9380
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Pond     (Intercept) 31.31    5.595
##  Residual             11.61    3.408
## Number of obs: 88, groups:  Pond, 3
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)  9.990951   3.487363  3.972117   2.865   0.0461 *
## Depth        0.041130   0.008466 85.032275   4.858 5.34e-06 ***
## Num          0.587043   0.730051 85.006776   0.804   0.4236
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##       (Intr) Depth
## Depth -0.190
## Num   -0.322  0.075
```

```
temp <- verticalSubData %>%
  group_by(Pond, Depth) %>%
  filter(Depth %in% c(50, 100, 150)) %>%
  select(Temperature, DO, Conductivity, Turbidity) %>%
  summarize(meanTemp = mean(Temperature), meanDO = mean(DO), meanCond = mean(Conductivity),
            meanTurb = mean(Turbidity))
```

```
## Adding missing grouping variables: `Pond`, `Depth`
## `summarise()` has grouped output by 'Pond'. You can override using the
## `.groups` argument.
```

```
temp1 <- rbind(temp, data.frame(Pond = "T109", Depth = 150,
                    meanTemp = NA,
                    meanDO = NA,
                    meanCond = NA,
                    meanTurb = NA))

temp2 <- rbind(temp, data.frame(Pond = "T109", Depth = 150,
                    meanTemp = NA,
                    meanDO = NA,
                    meanCond = NA,
                    meanTurb = NA))

sites <- c(rep("A", 9), rep("B", 9))

dfMeanWater <- cbind(rbind(temp1, temp2), sites = sites)
dfMeanWater
```

```
## # A tibble: 18 x 7
```

```
## # Groups:   Pond [3]
##    Pond  Depth meanTemp meanDO meanCond meanTurb sites
##    <chr> <dbl>    <dbl>  <dbl>    <dbl>    <dbl> <chr>
##  1 T106     50     23.4  112.      556     17.8  A
##  2 T106    100     23.0   90.2     582     17.2  A
##  3 T106    150     22.3    3.9     640     12.0  A
##  4 T108     50     23.7   68.1     940.     5.66 A
##  5 T108    100     23.2   56.4     942.     5.90 A
##  6 T108    150     22.8   27.2     998.     9.00 A
##  7 T109     50     23.4  147.      657     14.9  A
##  8 T109    100     22.6   97.9     662.    21.8  A
##  9 T109    150     NA     NA        NA     NA    A
## 10 T106     50     23.4  112.      556     17.8  B
## 11 T106    100     23.0   90.2     582     17.2  B
## 12 T106    150     22.3    3.9     640     12.0  B
## 13 T108     50     23.7   68.1     940.     5.66 B
## 14 T108    100     23.2   56.4     942.     5.90 B
## 15 T108    150     22.8   27.2     998.     9.00 B
## 16 T109     50     23.4  147.      657     14.9  B
## 17 T109    100     22.6   97.9     662.    21.8  B
## 18 T109    150     NA     NA        NA     NA    B
```

Diversity Measures

```
numericalInvData <- invertData[,-c(1,2)]
SR <- specnumber(numericalInvData)
```

```
simp.inv <- diversity(numericalInvData, index = "invsimpson")
simp.even <- simp.inv / SR
simp.even
```

```
##    T106A1    T106A2    T106A3    T106B1    T106B2    T106B3    T108A1    T108A2
## 0.2094016 0.3240000 0.3722076 0.2453169 0.3557052 0.3773181 0.2779971 0.3014533
##    T108A3    T108B1    T108B2    T108B3    T109A1    T109A2    T109A3    T109B1
## 0.5218430 0.2279113 0.4483384 0.6794162 0.2824812 0.2458150 0.3322148 0.2851288
##    T109B2    T109B3
## 0.2867647 0.5423729
```

```
shan <- diversity(numericalInvData, index = "shannon")
exp.shan <- exp(shan)
shan.even <- shan / (log(SR))
dfDivFull <- cbind(dfMeanWater, data.frame(SR, simp.inv, simp.even, shan, shan.even, exp.shan))
dfDiv <- dfDivFull[-c(9, 18),]
```

```
dfDivFull %>%
  group_by(Pond) %>%
  summarize(meanSR = mean(SR),
            meanSimp = mean(simp.inv),
            meanSimpEv = mean(simp.even),
            meanShan = mean(shan),
            meanShanEv = mean(shan.even),
            meanShanEx = mean(exp.shan))
```

```
## # A tibble: 3 x 7
##   Pond  meanSR meanSimp meanSimpEv meanShan meanShanEv meanShanEx
##   <chr>  <dbl>    <dbl>      <dbl>    <dbl>      <dbl>      <dbl>
## 1 T106    5.83     1.96      0.377    0.830      0.497       2.39
## 2 T108    6        1.81      0.306    0.806      0.452       2.29
## 3 T109    5.33     1.86      0.369    0.806      0.489       2.36
```

```
dfDivFull %>%
  group_by(Depth) %>%
  summarize(meanSR = mean(SR),
            meanSimp = mean(simp.inv),
            meanSimpEv = mean(simp.even),
            meanShan = mean(shan),
            meanShanEv = mean(shan.even),
            meanShanEx = mean(exp.shan))
```

```
## # A tibble: 3 x 7
##   Depth meanSR meanSimp meanSimpEv meanShan meanShanEv meanShanEx
##   <dbl>  <dbl>    <dbl>      <dbl>    <dbl>      <dbl>      <dbl>
## 1    50   7.17     1.82      0.255    0.789      0.386       2.39
## 2   100   5.33     1.72      0.327    0.766      0.459       2.16
## 3   150   4.67     2.08      0.471    0.887      0.593       2.49
```

```
mean(dfDivFull$SR)
```

```
## [1] 5.722222
```

```
mean(dfDivFull$simp.inv)
```

```
## [1] 1.87568
```

```
mean(dfDivFull$simp.even)
```

```
## [1] 0.3508715
```

```
mean(dfDivFull$shan)
```

```
## [1] 0.8140112
```

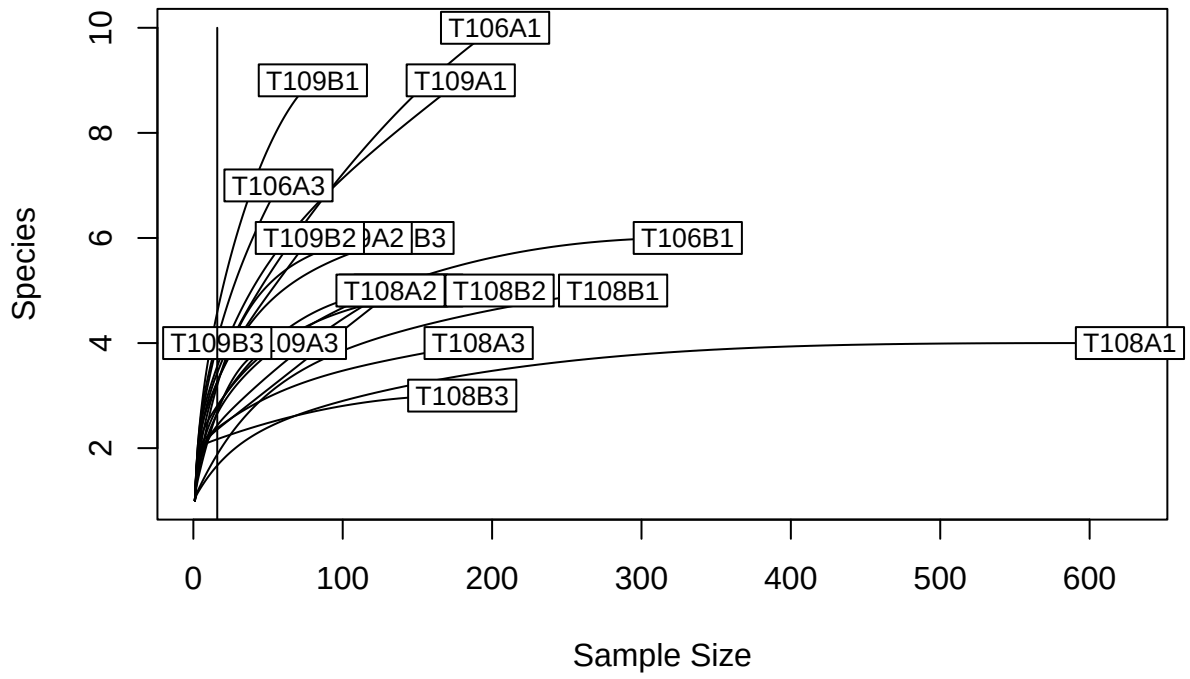
```
mean(dfDivFull$shan.even)
```

```
## [1] 0.4794945
```

```
mean(dfDivFull$exp.shan)
```

```
## [1] 2.34643
```

```
rarecurve(numericalInvData)
community.N <- rowSums(numericalInvData)
smallest.N <- min(community.N)
lines(x = c(smallest.N, smallest.N), y = c(0, max(SR)))
```




```
srModel <- lmer(SR ~ Depth + meanTemp + meanDO + meanCond + meanTurb + (1|Pond) + (sites|Pond),
                data = dfDiv,
                REML = FALSE)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(srModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: SR ~ Depth + meanTemp + meanDO + meanCond + meanTurb + (1 | Pond) +
##     (sites | Pond)
##    Data: dfDiv
##
##      AIC      BIC   logLik deviance df.resid
##     78.4     86.9    -28.2     56.4        5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -1.4130 -0.7272 -0.2612  0.7522  1.3890
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  Pond     (Intercept) 0.000    0.000
##  Pond.1   (Intercept) 1.281    1.132
##           sitesB      4.892    2.212    -1.00
##  Residual             1.449    1.204
## Number of obs: 16, groups:  Pond, 3
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 12.615469  61.424415 12.940378   0.205    0.840
## Depth       -0.042866   0.027600 12.939833  -1.553    0.145
## meanTemp    -0.130251   2.550698 12.940203  -0.051    0.960
## meanDO      -0.020562   0.020700 12.940794  -0.993    0.339
## meanCond     0.001346   0.003534 12.938691   0.381    0.709
## meanTurb     0.067559   0.174439 12.938955   0.387    0.705
##
## Correlation of Fixed Effects:
##          (Intr) Depth  menTmp meanDO menCnd
## Depth    -0.743
## meanTemp -0.998  0.735
## meanDO    0.335  0.319 -0.349
## meanCond  0.010 -0.308 -0.057 -0.297
## meanTurb -0.620  0.079  0.599 -0.732  0.606
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
simpModel <- lmer(simp.inv ~ Depth + meanTemp + meanDO + meanCond + meanTurb + (1|Pond) + (sites|Pond),
                  data = dfDiv,
                  REML = FALSE)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(simpModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: simp.inv ~ Depth + meanTemp + meanDO + meanCond + meanTurb +
##     (1 | Pond) + (sites | Pond)
##    Data: dfDiv
##
##      AIC      BIC   logLik deviance df.resid
##     42.0     50.5    -10.0     20.0        5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.4508 -0.8625 -0.1195  0.7609  1.7330
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  Pond     (Intercept) 0.000000 0.0000
```

```
##  Pond.1   (Intercept) 0.007551 0.0869
##           sitesB      0.051388 0.2267   -1.00
##  Residual             0.192087 0.4383
## Number of obs: 16, groups:  Pond, 3
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)  3.5597110 22.4051179 11.7657852   0.159    0.876
## Depth        0.0004669  0.0100553 11.4568131   0.046    0.964
## meanTemp     0.0030679  0.9306290 11.8185906   0.003    0.997
## meanDO      -0.0027155  0.0075508 11.7708354  -0.360    0.725
## meanCond    -0.0015044  0.0012912 11.9871350  -1.165    0.267
## meanTurb    -0.0385452  0.0635303 11.3624657  -0.607    0.556
##
## Correlation of Fixed Effects:
##          (Intr) Depth  menTmp meanDO menCnd
## Depth    -0.743
## meanTemp -0.998  0.735
## meanDO    0.337  0.316 -0.351
## meanCond  0.013 -0.308 -0.060 -0.293
## meanTurb -0.620  0.079  0.599 -0.732  0.603
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
simpEvModel <- lmer(simp.even ~ Depth + meanTemp + meanDO + meanCond + meanTurb + (1|Pond) + (sites|Pon
                data = dfDiv,
                REML = FALSE)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(simpEvModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: simp.even ~ Depth + meanTemp + meanDO + meanCond + meanTurb +
##     (1 | Pond) + (sites | Pond)
##    Data: dfDiv
##
##      AIC      BIC   logLik deviance df.resid
##    -21.7    -13.2     21.8    -43.7        5
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.82090 -0.09868  0.12652  0.37943  1.92627
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  Pond     (Intercept) 0.000000 0.00000
##  Pond.1   (Intercept) 0.002738 0.05232
##           sitesB      0.006499 0.08062  -1.00
##  Residual             0.002916 0.05400
## Number of obs: 16, groups:  Pond, 3
##
```

```
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)  1.352e+00  2.781e+00  1.321e+01   0.486  0.63477
## Depth        1.910e-03  1.242e-03  1.297e+01   1.538  0.14805
## meanTemp    -2.539e-02  1.156e-01  1.317e+01  -0.220  0.82954
## meanDO       9.986e-06  9.372e-04  1.321e+01   0.011  0.99166
## meanCond    -5.755e-04  1.610e-04  1.275e+01  -3.576  0.00348 **
## meanTurb    -1.413e-02  7.835e-03  1.280e+01  -1.804  0.09481 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) Depth  menTmp meanDO menCnd
## Depth    -0.744
## meanTemp -0.998  0.736
## meanDO    0.347  0.305 -0.361
## meanCond  0.022 -0.311 -0.070 -0.276
## meanTurb -0.621  0.082  0.599 -0.731  0.592
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
shanModel <- lmer(exp.shan ~ Depth + meanTemp + meanDO + meanCond + meanTurb + (1|Pond) + (sites|Pond),
                  data = dfDiv,
                  REML = FALSE)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
## Warning: Model failed to converge with 1 negative eigenvalue: -1.5e-01
```

```
summary(shanModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: exp.shan ~ Depth + meanTemp + meanDO + meanCond + meanTurb +
##     (1 | Pond) + (sites | Pond)
##    Data: dfDiv
##
##      AIC      BIC   logLik deviance df.resid
##     52.9     61.4    -15.5     30.9        5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.1527 -0.9332 -0.3505  0.7842  1.5304
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  Pond     (Intercept) 0.0000   0.0000
##  Pond.1   (Intercept) 0.0000   0.0000
##           sitesB      0.5913   0.7690    NaN
##  Residual             0.2855   0.5343
## Number of obs: 16, groups:  Pond, 3
##
```

```
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept) -21.916648  29.759289  13.428759  -0.736   0.4741
## Depth         0.006530   0.012607  12.239604   0.518   0.6137
## meanTemp      1.180739   1.249679  13.615356   0.945   0.3612
## meanDO       -0.011392   0.010038  13.432545  -1.135   0.2763
## meanCond     -0.003231   0.001812  14.015711  -1.783   0.0962 .
## meanTurb     -0.023932   0.078379  11.889735  -0.305   0.7654
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) Depth  menTmp meanDO menCnd
## Depth    -0.754
## meanTemp -0.998  0.746
## meanDO    0.442  0.190 -0.460
## meanCond  0.117 -0.340 -0.168 -0.116
## meanTurb -0.623  0.111  0.601 -0.724  0.480
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
shanEvModel <- lmer(shan.even ~ Depth + meanTemp + meanDO + meanCond + meanTurb + (1|Pond) + (sites|Pon
                data = dfDiv,
                REML = FALSE)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(shanEvModel)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: shan.even ~ Depth + meanTemp + meanDO + meanCond + meanTurb +
##     (1 | Pond) + (sites | Pond)
##    Data: dfDiv
##
##      AIC      BIC   logLik deviance df.resid
##     -3.7      4.8     12.9    -25.7        5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.9664 -0.4161 -0.1816  0.5243  1.5685
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  Pond     (Intercept) 0.000000 0.00000
##  Pond.1   (Intercept) 0.001362 0.03690
##           sitesB      0.011710 0.10821  -1.00
##  Residual             0.009717 0.09857
## Number of obs: 16, groups:  Pond, 3
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)  0.6483533  5.0719050 12.2682215   0.128    0.900
```

```
## Depth        0.0013112  0.0022661 11.8294491   0.579    0.574
## meanTemp     0.0084856  0.2108587 12.2958499   0.040    0.969
## meanDO      -0.0009586  0.0017094 12.2691394  -0.561    0.585
## meanCond    -0.0004212  0.0002936 12.1250217  -1.435    0.177
## meanTurb    -0.0095915  0.0143005 11.6458064  -0.671    0.515
##
## Correlation of Fixed Effects:
##          (Intr) Depth  menTmp meanDO menCnd
## Depth    -0.744
## meanTemp -0.998  0.736
## meanDO    0.346  0.307 -0.360
## meanCond  0.021 -0.311 -0.069 -0.278
## meanTurb -0.621  0.082  0.599 -0.731  0.593
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

Similarity

```
brayCurtisInv <- 1 - vegdist(numericalInvData, method = "bray", diag = FALSE) %>%
  as.matrix()
```

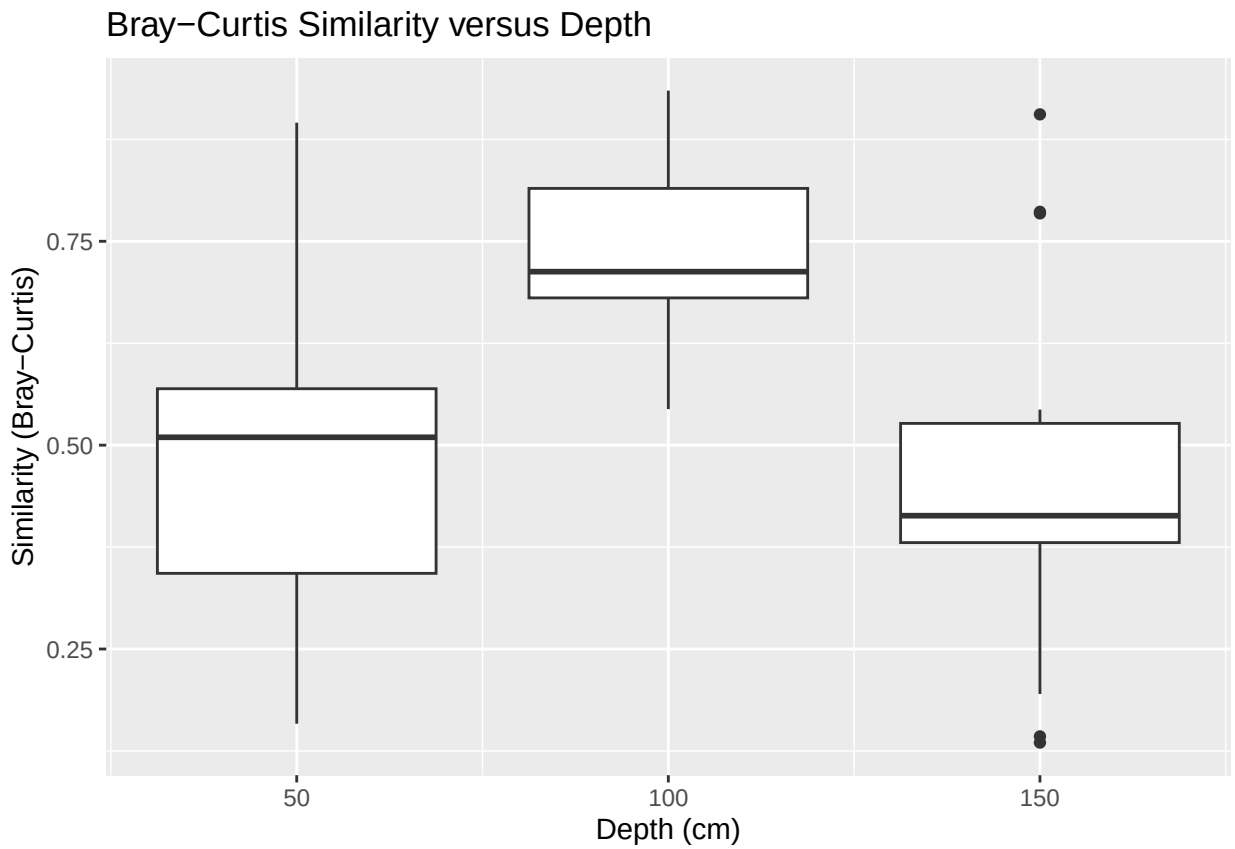
Bray Curtis By Depth

```
mod3index1 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(1, 18, 3)) %>%
  as.data.frame()

mod3index2 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(2, 18, 3)) %>%
  as.data.frame()

mod3index3 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(3, 18, 3)) %>%
  as.data.frame()
```

```
brayByDepth <- c()
for (i in 1:length(mod3index1$V1)){
  brayByDepth <- rbind(brayByDepth,
                       c(50, brayCurtisInv[mod3index1$V1[i], mod3index1$V2[i]]))
}


temp2 <- 0
for (i in 1:length(mod3index2$V1)){
  brayByDepth <- rbind(brayByDepth,
                       c(100, brayCurtisInv[mod3index2$V1[i], mod3index2$V2[i]]))
}


temp3 <- 0
for (i in 1:length(mod3index3$V1)){
  brayByDepth <- rbind(brayByDepth,
                       c(150, brayCurtisInv[mod3index3$V1[i], mod3index3$V2[i]]))
```

```
}

brayByDepth <- as.data.frame(brayByDepth)
colnames(brayByDepth) <- c("depth", "bcSim")
```

```
brayByDepth %>%
  ggplot(aes(x = depth, y = bcSim, group = depth)) +
  geom_boxplot() +
  labs(x = "Depth (cm)", y = "Similarity (Bray-Curtis)", title = "Bray-Curtis Similarity versus Depth")
```

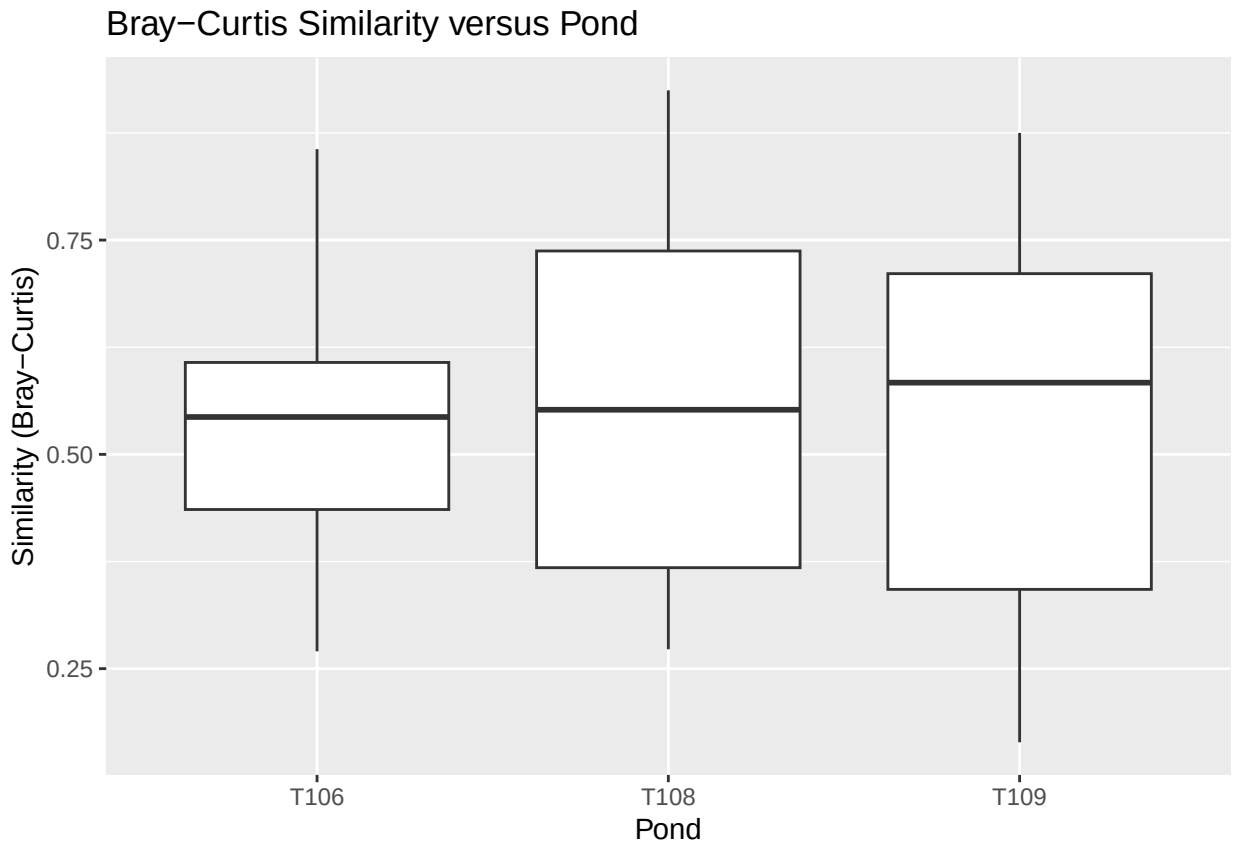


Bray-Curtis By Pond

```
ord3index1 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 1:6) %>%
  as.data.frame()

ord3index2 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 7:12) %>%
  as.data.frame()

ord3index3 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 13:18) %>%
  as.data.frame()
```

```
brayByPond <- c()
for (i in 1:length(ord3index1$V1)){
  brayByPond <- rbind(brayByPond,
                      c("T106", brayCurtisInv[ord3index1$V1[i], ord3index1$V2[i]]))
}


for (i in 1:length(ord3index2$V1)){
  brayByPond <- rbind(brayByPond,
                      c("T108", brayCurtisInv[ord3index2$V1[i], ord3index2$V2[i]]))
}


for (i in 1:length(ord3index3$V1)){
  brayByPond <- rbind(brayByPond,
                      c("T109", brayCurtisInv[ord3index3$V1[i], ord3index3$V2[i]]))
}

brayByPond <- as.data.frame(brayByPond)
colnames(brayByPond) <- c("pond", "bcSim")
brayByPond$bcSim <- as.numeric(brayByPond$bcSim)
```

```
brayByPond %>%
  ggplot(aes(x = pond, y = bcSim, group = pond)) +
  geom_boxplot() +
  labs(x = "Pond", y = "Similarity (Bray-Curtis)", title = "Bray-Curtis Similarity versus Pond")
```




```
paInvData <- numericalInvData
paInvData[paInvData > 0] <- 1

sorensonInv <- 1 - vegdist(paInvData, method = "bray", diag = FALSE) %>%
  as.matrix()
```

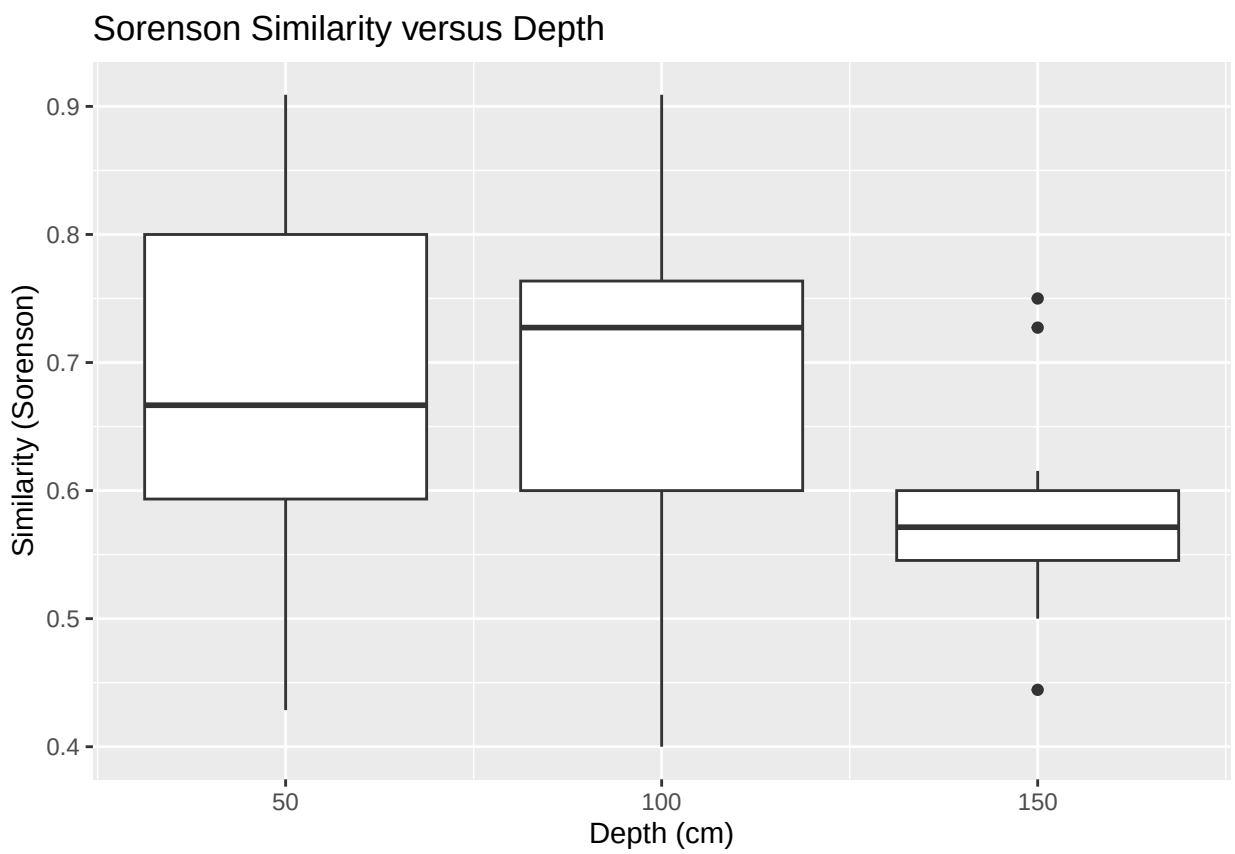
Jaccard By Depth

```
mod3index1 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(1, 18, 3)) %>%
  as.data.frame()

mod3index2 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(2, 18, 3)) %>%
  as.data.frame()

mod3index3 <- combinations(n = 6, r = 2, repeats.allowed = F, v = seq(3, 18, 3)) %>%
  as.data.frame()
```

```
sorenByDepth <- c()
for (i in 1:length(mod3index1$V1)){
  sorenByDepth <- rbind(sorenByDepth,
                        c(50, sorensonInv[mod3index1$V1[i], mod3index1$V2[i]]))
}
```

```
temp2 <- 0
for (i in 1:length(mod3index2$V1)){
  sorenByDepth <- rbind(sorenByDepth,
                        c(100, sorensonInv[mod3index2$V1[i], mod3index2$V2[i]]))
}


temp3 <- 0
for (i in 1:length(mod3index3$V1)){
  sorenByDepth <- rbind(sorenByDepth,
                        c(150, sorensonInv[mod3index3$V1[i], mod3index3$V2[i]]))
}

sorenByDepth <- as.data.frame(sorenByDepth)
colnames(sorenByDepth) <- c("depth", "sorSim")
```

```
sorenByDepth %>%
  ggplot(aes(x = depth, y = sorSim, group = depth)) +
  geom_boxplot() +
  labs(x = "Depth (cm)", y = "Similarity (Sorenson)", title = "Sorenson Similarity versus Depth")
```

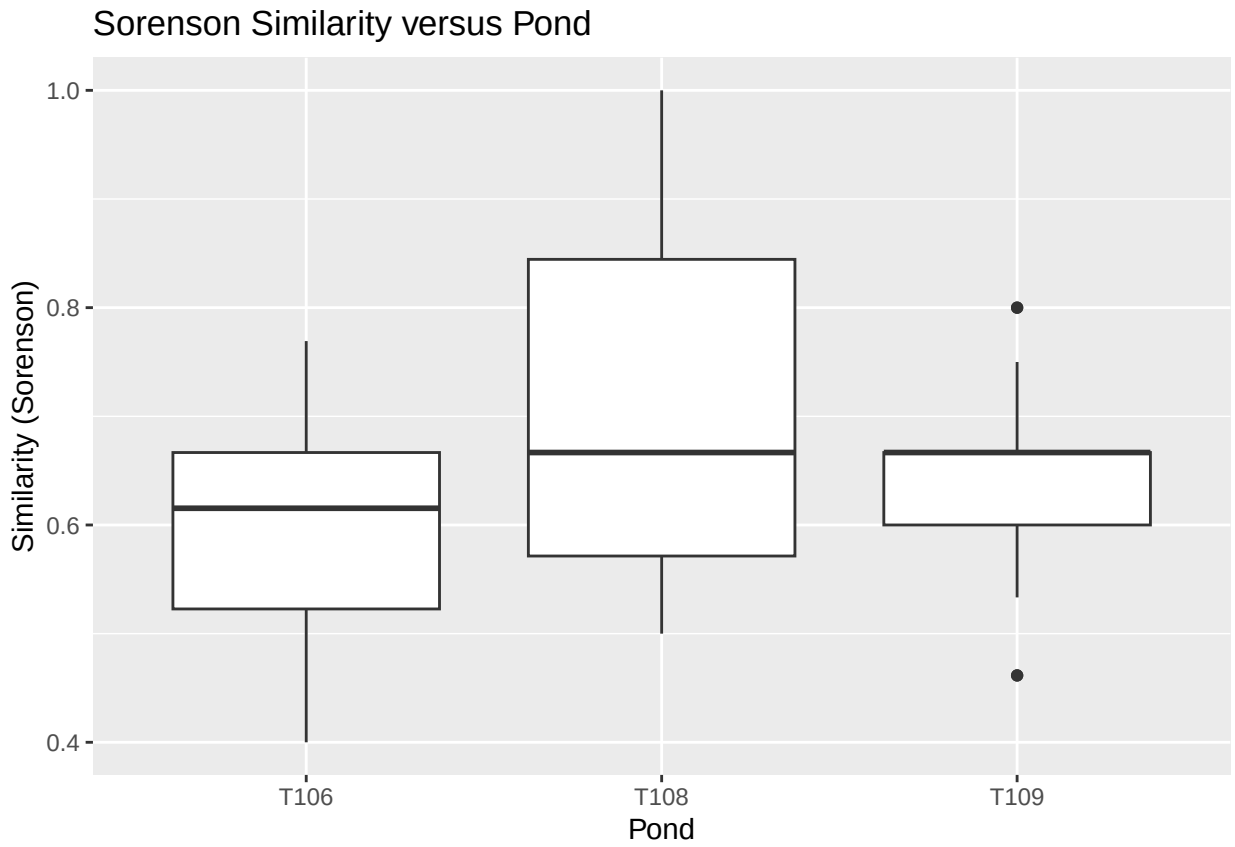


Sorenson By Pond

```
ord3index1 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 1:6) %>%
  as.data.frame()

ord3index2 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 7:12) %>%
  as.data.frame()

ord3index3 <- combinations(n = 6, r = 2, repeats.allowed = F, v = 13:18) %>%
  as.data.frame()
```

```
sorenByPond <- c()
for (i in 1:length(ord3index1$V1)){
  sorenByPond <- rbind(sorenByPond,
                       c("T106", sorensonInv[ord3index1$V1[i], ord3index1$V2[i]]))
}


for (i in 1:length(ord3index2$V1)){
  sorenByPond <- rbind(sorenByPond,
                       c("T108", sorensonInv[ord3index2$V1[i], ord3index2$V2[i]]))
}


for (i in 1:length(ord3index3$V1)){
  sorenByPond <- rbind(sorenByPond,
                       c("T109", sorensonInv[ord3index3$V1[i], ord3index3$V2[i]]))
}

sorenByPond <- as.data.frame(sorenByPond)
colnames(sorenByPond) <- c("pond", "sorSim")
sorenByPond$sorSim <- as.numeric(sorenByPond$sorSim)
```

```
sorenByPond %>%
  ggplot(aes(x = pond, y = sorSim, group = pond)) +
  geom_boxplot() +
  labs(x = "Pond", y = "Similarity (Sorenson)", title = "Sorenson Similarity versus Pond")
```




```
brayCurtisDis <- vegdist(numericalInvData[-c(9,18),], method = "bray")

adonis2(brayCurtisDis ~ Pond + Depth + meanTemp + meanDO + meanCond + meanTurb,
        data = dfDiv)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = brayCurtisDis ~ Pond + Depth + meanTemp + meanDO + meanCond + meanTurb, data = dfDi
##          Df SumOfSqs      R2      F Pr(>F)
## Pond      2  0.15038 0.08188 0.6236  0.769
## Depth     1  0.40640 0.22127 3.3707  0.015 *
## meanTemp  1  0.05580 0.03038 0.4628  0.813
## meanDO    1  0.14840 0.08080 1.2309  0.314
## meanCond  1  0.07955 0.04331 0.6598  0.630
## meanTurb  1  0.03160 0.01720 0.2621  0.941
## Residual  8  0.96454 0.52516
## Total    15  1.83668 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```