Data Screening

PAZ

06/04/2017

Introduction

This Data Screenining notebook follows the GUide to STatistical Analysis in Microbial Ecology (GUSTA ME). The purpose is to inspect the variables that we'll be using to test for hypotheses later on, and check whether they follow typical assumptions made in parametric tests such as normality, freedom from heteroskedasticity (difference in variability btw. two+ variables) and outliers.

Reference:

https://sites.google.com/site/mb3gustame/home Buttigieg PL, Ramette A (2014) A Guide to Statistical Analysis in Microbial Ecology: a community-focused, living review of multivariate data analyses. FEMS Microbiol Ecol. 90: 543-550.

Files Used

- **MassBalance_R.csv** (Book 07 - Soils & Discharge Merged)
- **WeeklySoils_Rng.csv** (Book 06 - Mass Soils COmposite)

Files Written

- **OutletData4Lutz_R.csv** (Data to compare against Lutz 2013 article)

Packages

```
library(sm)
library(vioplot)

library(dplyr)
library(tidyr)
library(zoo)
library(reshape)
library(ggplot2)
library("ggrepel")

library("plotly")
library("cowplot")
library("gridExtra")
library("Cairo")
library("GGally")
library("scales")

library("plotKML")
```

```r
# Stats
library("vegan")
library("cluster")

# Saving a xlxs file
# library(xlsx)
```

Missing values

1. Missing chemical and isotope data due to machine failure or automatic sampling servicing program.

These have been considered to be Values Missing Completely at Random (MCAR) as they are associated to the end of the automatic sampler's capacity for a certain number of events where servicing was inadequate for the discharge amounts seen during a sampling week. Here the values' missingess is not related to any other value in the data set.

2. Isotope data for both soil and water samples due to concentration value being below the limit of detection.

These values must be considered to be Missing at Random (MAR) as the missing value has no relation to the value that 'should' be there, but does depend on other variables in the data set. Thus, other variables must be taken into account for MAR data to be considered random (i.e. missing data is "conditioned by" other data in the data set).

Lab parameters

```r
source("global.R")
```

Import soils

Convert to single time observation for merging with water observation.

```r
# Soils
soils = read.csv2("Data/MassBalance_R.csv",
                        na.strings=c('#DIV/0!', '', 'NA'), header = TRUE)
names(soils)
```

```
##  [1] "ti"                  "WeekSubWeek"
##  [3] "Event"               "Duration.Hrs"
##  [5] "timeSinceApp"        "timeSinceApp.NoSo"
##  [7] "timeSinceApp.N"      "timeSinceApp.T"
##  [9] "timeSinceApp.S"      "diss.d13C"
## [11] "SD.d13C"             "CumOutDiss.g"
## [13] "CumOutFilt.g"        "TotSMout.g"
## [15] "TotSMout.g.SD"       "MELsm.g"
## [17] "MELsm.g.SD"          "Appl.Mass.g"
## [19] "Appl.Mass.g.OT"      "CumAppMass.g"
## [21] "CumAppMass.g.N"      "CumAppMass.g.T"
## [23] "CumAppMass.g.S"      "CumAppMass.g.OT"
## [25] "CumAppMass.g.N.OT"   "CumAppMass.g.T.OT"
```

```
## [27] "CumAppMass.g.S.OT"     "iniCo.ug.g.N"
## [29] "iniCo.ug.g.T"          "iniCo.ug.g.S"
## [31] "CumOutSmeto.g"         "CumOutMELsm.g"
## [33] "MassSoil.g.North"      "MassSoil.g.SD.North"
## [35] "Conc.mug.g.dry.soil.N" "comp.d13C.North"
## [37] "comp.d13C.SD.North"    "ID.N"
## [39] "Area.N"                "Area.T"
## [41] "Area.S"                "MassSoil.g.Talweg"
## [43] "MassSoil.g.SD.Talweg"  "Conc.mug.g.dry.soil.T"
## [45] "comp.d13C.Talweg"      "comp.d13C.SD.Talweg"
## [47] "ID.T"                  "MassSoil.g.South"
## [49] "MassSoil.g.SD.South"   "Conc.mug.g.dry.soil.S"
## [51] "comp.d13C.South"       "comp.d13C.SD.South"
## [53] "ID.S"                  "DD13C.North"
## [55] "DD13C.Talweg"          "DD13C.South"
## [57] "CatchMassSoil.g"       "CatchMassSoil.g.SD"
## [59] "BulkCatch.d13"         "BulkCatch.d13.SD"
## [61] "DD13.Bulk"             "Area.Catchment"
## [63] "BulkCatch.Conc"        "iniCo.Bulk"
```

```
colnames(soils)[colnames(soils) == "ti"] <- "Date.ti"
soils$Date.ti <- as.POSIXct(strptime(soils$Date.ti,
                                     "%Y-%m-%d %H:%M", tz="EST")) # csv typos, option 1
sum(is.na(soils$Date.ti)) == 0
```

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

```
initialDelta
```

```
## [1] -32.3
```

```
# Get rid of imputed values to avoid bias
soils$DD13C.North <- (ifelse(!is.na(soils$comp.d13C.SD.North), soils$comp.d13C.North - (initialDelta),
soils$DD13C.Talweg <- (ifelse(!is.na(soils$comp.d13C.SD.Talweg), soils$comp.d13C.Talweg - (initialDelta
soils$DD13C.South <- (ifelse(!is.na(soils$comp.d13C.SD.South), soils$comp.d13C.South - (initialDelta),


dropSoil <- c("WeekSubWeek", # "Event",
              "CumOutDiss.g", "CumOutFilt.g", "CumOutAppMass.g",  "CumOutMELsm.g",
              # "CumAppMass.g",
              # "ID.N",
              "ID.T",  "Area.N", "Area.T", "Area.S",
              "comp.d13C.SE.North", "comp.d13C.SE.Talweg", "comp.d13C.SE.South",
              "ngC.SD","ngC.SE", "N_compsoil" )#, "N_ngC")
soils <- soils[ , !(names(soils) %in% dropSoil)]

soilsCheck <- soils[complete.cases(soils[ , "ID.N"]),]

timeApps <- soils[ , c("Date.ti", "timeSinceApp", "timeSinceApp.NoSo",
                       "timeSinceApp.N", "timeSinceApp.T" , "timeSinceApp.S",
                      "Event")]
```

Soils from Book: 06, to merge with "timeApps"

```
# Quasi-Molten SOILS
soilGroups = read.csv2("Data/WeeklySoils_Rng.csv",
                       na.strings=c('#DIV/0!', '', 'NA'), header = TRUE)
soilGroups$Date.ti <- as.POSIXct(strptime(soilGroups$Date.ti,
                                          "%Y-%m-%d %H:%M", tz="EST")) # csv typos, option 1
sum(is.na(soilGroups$Date.ti)) == 0
```

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

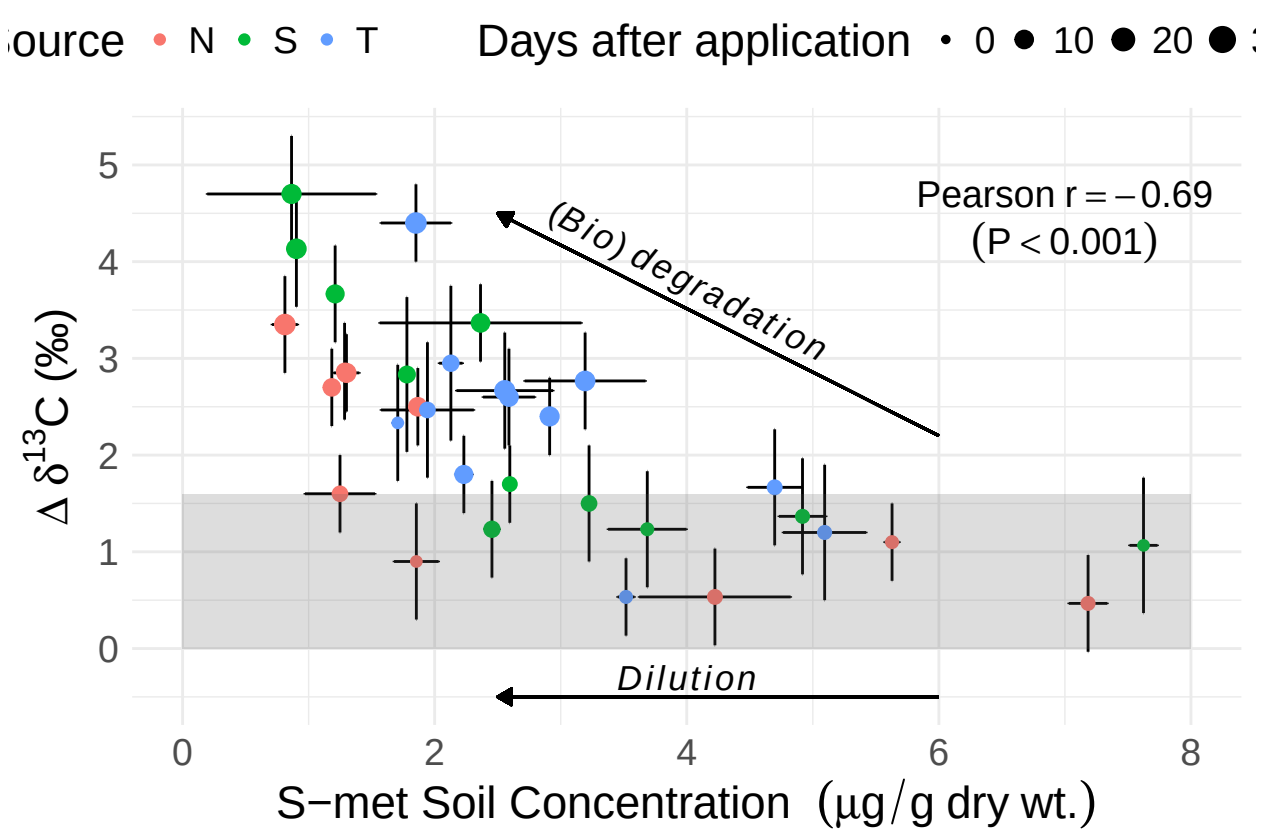
```
soilGroups$comp.d13C <- ifelse(is.na(soilGroups$comp.d13C.SD), NA, soilGroups$comp.d13C)
# soilGroups$ngC.Label <- ifelse(soilGroups$ngC.mean < 10, "< 10 ng", "> 10 ng")

soilGroups <- subset(soilGroups, comp.d13C.SD <= 0.70)

#str(soils)

soilGrApp <- merge(soilGroups, timeApps, by = "Date.ti", all = F)
soilGrApp <- soilGrApp[complete.cases(soilGrApp[ , "timeSinceApp"]),]

soilGrApp$DD13C.comp <- ifelse(is.na(soilGrApp$comp.d13C.SD), NA, soilGrApp$DD13C.comp)
soilGrApp <- subset(soilGrApp, comp.d13C.SD <= 0.70)

# Propagated SD
soilGrApp$prop.d13C.SD = round((soilGrApp$comp.d13C.SD^2 + initialDeltaError^2)^0.5, 1)

cor.test(soilGroups$comp.d13C, soilGroups$Conc.mug.g.dry.soil)
```

```
##
##  Pearson's product-moment correlation
##
## data:  soilGroups$comp.d13C and soilGroups$Conc.mug.g.dry.soil
## t = -5.3104, df = 31, p-value = 8.817e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.8355218 -0.4545936
## sample estimates:
##       cor
## -0.6901877
```

```
pearson_r <- cor.test(soilGroups$comp.d13C, soilGroups$Conc.mug.g.dry.soil)[4]
r_label <- sprintf("Pearson~r == %0.2f", pearson_r)
p_value <- cor.test(soilGroups$comp.d13C, soilGroups$Conc.mug.g.dry.soil)[3]

if (p_value < 0.0001){
  p_label <- "(P < 0.001)"
} else if (p_value < 0.001) {
  p_label <- "(P < 0.001)"
} else if (p_value < 0.015) {
  p_label <- ("P < 0.01")
} else {
  p_label <- "Check significance"
}
```

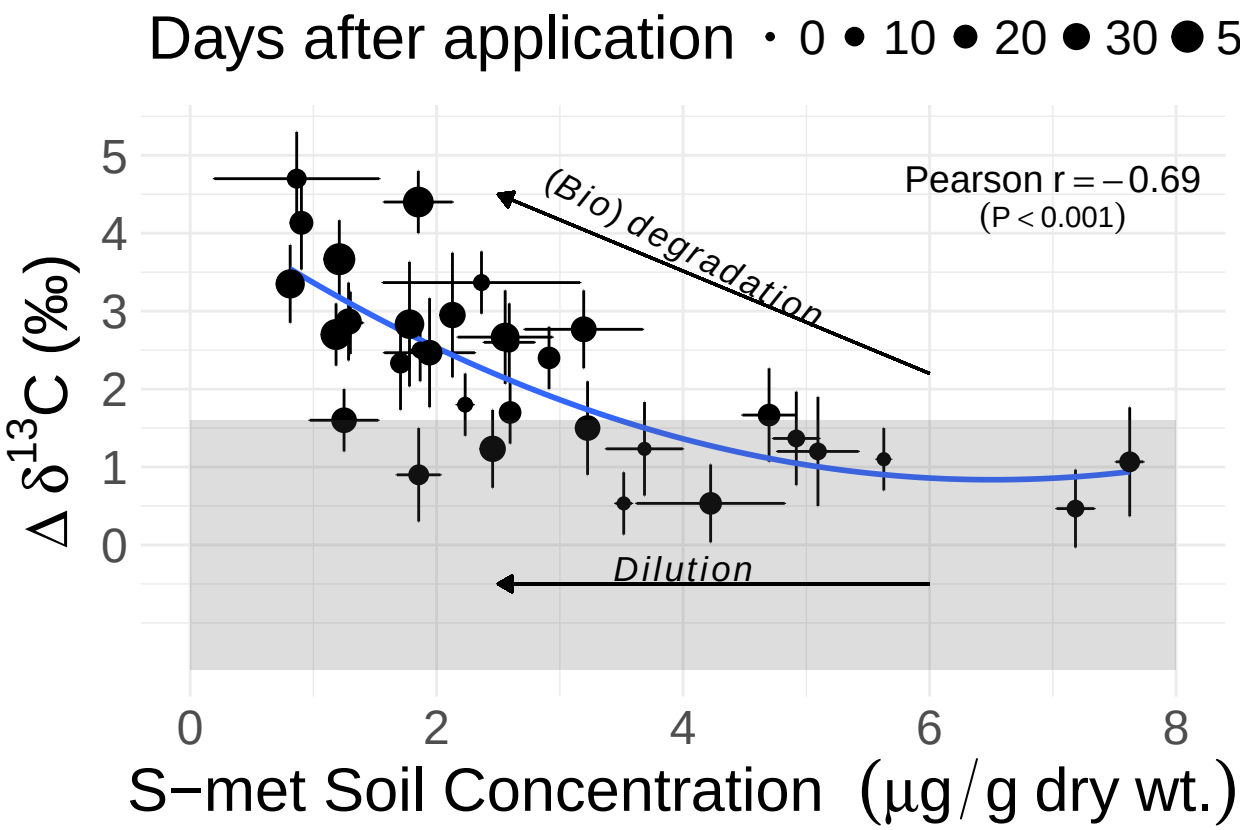
```r
soilGrApp$Source <- ifelse(soilGrApp$Transect == "T", "Valley", "Plateau")
soilGrApp$Source <- as.factor(soilGrApp$Source)

soilGrApp.N <- subset(soilGrApp, soilGrApp$Transect == "N")
soilGrApp.T <- subset(soilGrApp, soilGrApp$Transect == "T")
soilGrApp.S <- subset(soilGrApp, soilGrApp$Transect == "S")

soilGrApp.N$timeSinceApp <- soilGrApp.N$timeSinceApp.N
soilGrApp.T$timeSinceApp <- soilGrApp.T$timeSinceApp.T
soilGrApp.S$timeSinceApp <- soilGrApp.S$timeSinceApp.S

dropAppDates <- c("timeSinceApp.NoSo", "timeSinceApp.N", "timeSinceApp.T", "timeSinceApp.S")
soilGrApp.N <- soilGrApp.N[ , !(names(soilGrApp.N) %in% dropAppDates)]
soilGrApp.T <- soilGrApp.T[ , !(names(soilGrApp.T) %in% dropAppDates)]
soilGrApp.S <- soilGrApp.S[ , !(names(soilGrApp.S) %in% dropAppDates)]

soilGrApp <- rbind(soilGrApp.N, soilGrApp.T)
soilGrApp <- rbind(soilGrApp, soilGrApp.S)

# comp.d13C.SD, replaced by: prop.d13C.SD

p <- ggplot(data = soilGrApp, aes(x=Conc.mug.g.dry.soil, y=DD13C.comp))+
  geom_errorbar(aes(ymin = DD13C.comp - prop.d13C.SD, ymax = DD13C.comp + prop.d13C.SD)) +
  geom_errorbarh(aes(xmin = Conc.mug.g.dry.soil - Conc.ComSoil.SD, xmax = Conc.mug.g.dry.soil + Conc.Co
  #stat_smooth(data = subset(soilGrApp, Conc.mug.g.dry.soil < 8),
  #            aes(x=Conc.mug.g.dry.soil, y=DD13C.comp), method = "lm", formula = y ~ poly(x, 2), se=F)
  # geom_point(aes(group = ID, size = timeSinceApp.NoSo)) + # , colour = Source)) +  # , shape = ngC.La
  geom_point(aes(group = Transect, colour = Transect, size = Wnum)) +
  # theme_bw() +
  theme_minimal() +
  theme(legend.position = "top",
        text = element_text(size=17)) +
  labs(size="   Days after application", colour="Source" ) + #, shape = "Mass Carbon") +
  ylab(expression(paste({Delta~delta}^"13","C", ' (\u2030)'))) +
  xlab(expression(paste("S-met Soil Concentration  ", {({mu}*g / g~dry~wt.)}))) +
  annotate("text", x = 7.0, y = 4.7, label = as.character(r_label), parse = T, size = 5) +
  annotate("text", x = 7.0, y = 4.2, label = p_label, parse = T, size = 5) +
  scale_size_continuous(range = c(1, 5), breaks= c(0, 10, 20, 30, 50), limits = c(0, 50)) +
  scale_y_continuous(breaks=c(0, 1, 2, 3 , 4 ,5) ) +
  # scale_size_continuous(range = c(1, 5)) +
  guides(size=guide_legend(nrow=1)) +
  annotate("text", x = 4, y = -0.3, label= "italic(Dilution)", parse=T, size = 4.5) +
  geom_segment(aes(x = 6, y = -0.5, xend = 2.5, yend = -0.5),
                   arrow = arrow(length = unit(1/2, 'picas'), type = "closed")) +
  annotate("text",
           x = 4.0, y = 3.78,
           label= "paste(\"(\", italic(Bio), \") \", italic(degradation) )", parse=T, size = 4.5, angle=
  geom_segment(aes(x = 6, y = 2.2, xend = 2.5, yend = 4.5),
                   arrow = arrow(length = unit(1/2, 'picas'), type = "closed")) +
  annotate("rect", xmin=0, xmax=8, ymin=0, ymax=propagatedError, alpha=0.2)

  # geom_rect(aes(xmin=0, xmax=8, ymin=-0.5, ymax=0.5), colour = "grey", alpha = 0.5) +
  #geom_hline(yintercept = 0.5, color = "dodgerblue4", linetype = "dotted") +
```

```
  #geom_hline(yintercept = 0, color = "dodgerblue3", linetype = "dotted") +
  #geom_hline(yintercept = -0.5, color = "dodgerblue3", linetype = "dotted")

  #scale_color_hue("Group") +
  #scale_fill_manual(
  #  "CI horizontal line", values=rep(1,4),
  #  guide=guide_legend(override.aes = list(colour=c("orange", "darkred"))),
  #  labels=c("CI of 95%", "CI of 99%")
  #)
  #geom_text_repel(data = subset(soilGrApp, (!is.na(ngC.Label) & Wnum > 10) ), aes(label=Wnum),
  #                arrow = arrow(length = unit(0.005, 'npc'), type = "closed"),
  #                force = 1,
  #                point.padding = unit(1.0, 'lines'),
  #                max.iter = 2e3,
  #                nudge_x = .2)
p
```




```
#
# ggsave(p, filename = "images/DDvsConc_soils_2.pdf", device = "pdf", dpi = 300,  scale = 2)


# Note: SD are error propagated: prop.d13C.SD
SAVE = F
PC = T
if (SAVE){
  if (PC){
```

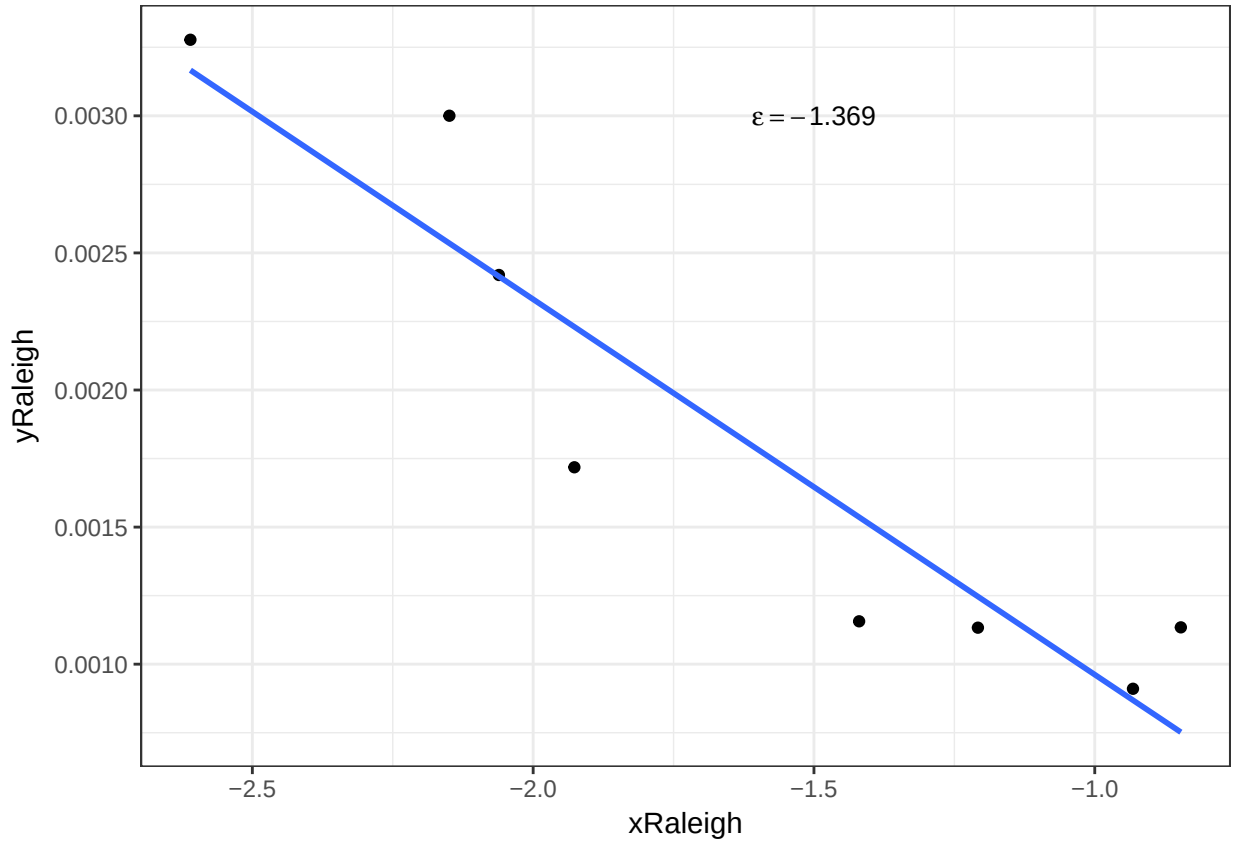
```
    ggsave(p,
       filename = "D:/Documents/these_pablo/WriteUp/Alteck_PNAS_LaTeX/images/DDvsConc_soils.pdf",
       device = "pdf", dpi = 600,  scale = 1, # )# ,
       width = 8.7, height = 6)
  } else {
    ggsave(p,
     filename = "/Users/DayTightChunks/Documents/PhD/Writeups/PNAS/Alteck_PNAS_LaTeX/images/DDvsConc_so
      device=cairo_pdf, dpi = 600,  scale = 1, # )# ,
       width = 8.7, height = 6)
  }
}

#ggplotly(p)
  #stat_smooth(method = "lm", formula = y ~ poly(x, 2)) +
  #stat_smooth(method = "lm", formula = y~x, se=F)
```



Field enrichment derivation (for error estimation)

```
soils$yRaleigh <- log((1000+d13Co+soils$DD13.Bulk)/(1000+d13Co))
soils$xRaleigh <- log(soils$BulkCatch.Conc/soils$iniCo.Bulk)
soilModel<-lm(yRaleigh~xRaleigh, data= soils)

cofsoil <- as.numeric(coef(soilModel)[2]*1000)
minX <- confint(soilModel, "xRaleigh", level = 0.95)[1]*1000
```

```r
maxX <- confint(soilModel, "xRaleigh", level = 0.95)[2]*1000
se <- summary(soilModel)$coef[[4]]*1000

e_label <- sprintf("epsilon == %0.3f", cofsoil)

CI95 = maxX - cofsoil

ggplot(data = subset(soils,  !is.na(yRaleigh) ), aes(x=xRaleigh, y=yRaleigh)) +
  geom_point() +
  stat_smooth(method = "lm", formula = y ~ x, se=F) +
  annotate("text", x = -1.5, y = 0.003, label = as.character(e_label), parse = T, size = 3.5) +
  theme_bw()
```




```r
summary(soilModel)
```

```
##
## Call:
## lm(formula = yRaleigh ~ xRaleigh, data = soils)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -5.122e-04 -1.801e-04  2.341e-05  1.789e-04  4.654e-04
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0004072  0.0003789  -1.075 0.323870
```

```
## xRaleigh    -0.0013690  0.0002167  -6.316 0.000735 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0003643 on 6 degrees of freedom
##   (44 observations deleted due to missingness)
## Multiple R-squared:  0.8693, Adjusted R-squared:  0.8475
## F-statistic:  39.9 on 1 and 6 DF,  p-value: 0.0007353
```

Import waters

Compare mass balance, theoretical and CSIA

```
waters = read.csv2("Data/WeeklyHydroContam_R.csv")
waters$ti <- as.POSIXct(strptime(waters$ti, "%Y-%m-%d %H:%M", tz="EST"))
colnames(waters)[colnames(waters) == "ti"] <- "Date.ti"
waters$Events <- factor(waters$Events, levels = unique(waters$Events))
waters$Event <- factor(waters$Event, levels = unique(waters$Event))

# Concentration ranges not being able to quantify CSIA
low4CSIA <- subset(waters, !is.na(diss.d13C))
min(low4CSIA$Conc.mug.L)
```

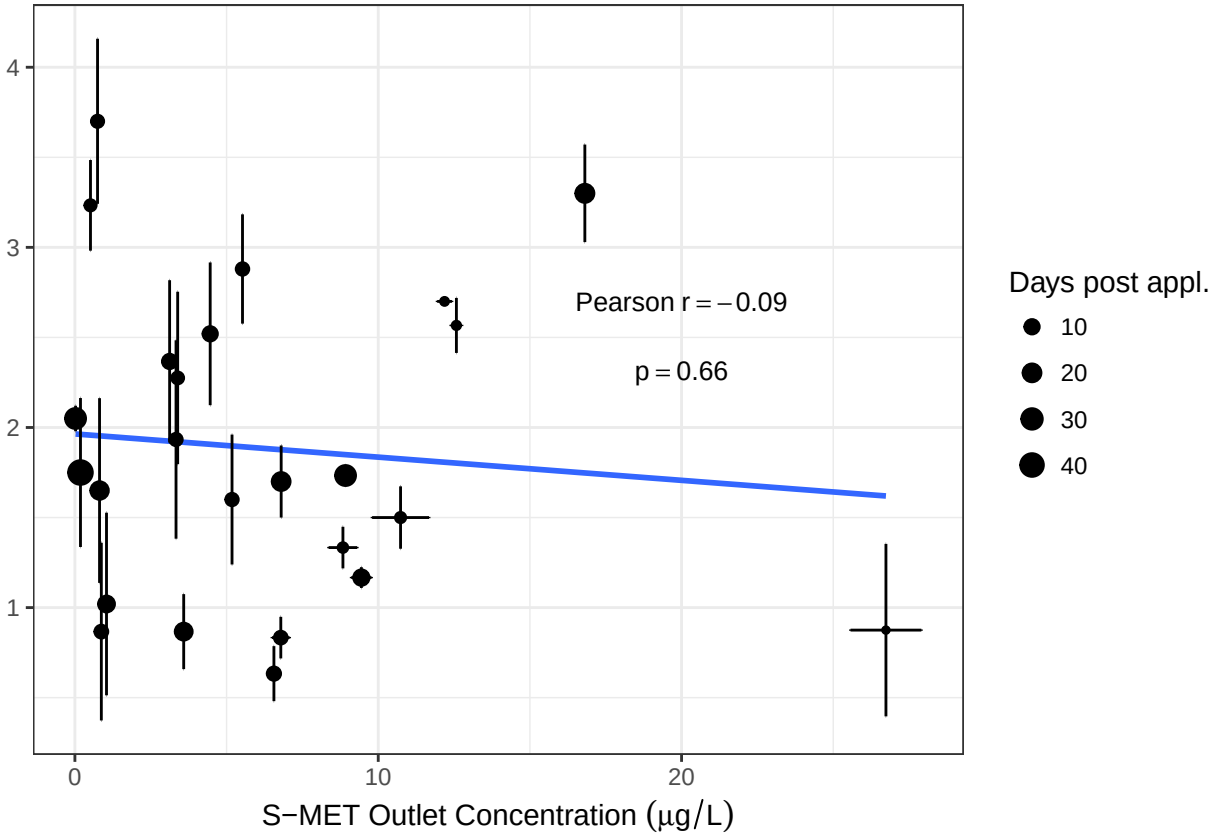
```
## [1] 0.02193412
```

```
#waters$remain_maxHalf
#waters$remain_minHalf

waterCo <- max(waters$Conc.mug.L)
d13Co
```

```
## [1] -32.3
```

```
waters$yRaleigh <- log((1000+d13Co+waters$DD13C.diss)/(1000+d13Co))
waters$xRaleigh <- log(waters$Conc.mug.L/waterCo)
waters$DIa <- waters$maxQ*waters$Volume.m3/waters$Duration.Hrs

# For evidence of desorption effects, Event 7-1 (May 12th) would need to show SD < 0.54 (currently at 0
# Contingent on sample repeats
waterClean <- subset(waters, Sampled == "Sampled" & SD.d13C < 0.64) # | filt.SD.d13C <= 0.75 )

# cor.test(waterClean$TotSMout.g, waterClean$DD13C.diss)

pearson_water_r <- cor.test(waterClean$Conc.mug.L, waterClean$DD13C.diss)[4]
water_r_label <- sprintf("Pearson~r == %0.2f", pearson_water_r)
water_p_value <- cor.test(waterClean$Conc.mug.L, waterClean$DD13C.diss)[3]
water_p_label <- sprintf("p == %0.2f", water_p_value)

waterIsoConc  <- ggplot(data = subset(waterClean), aes(x=Conc.mug.L, y=DD13C.diss))+
  stat_smooth(data = subset(waterClean),
              aes(x=Conc.mug.L, y=DD13C.diss), method = "lm", formula = y~x, se=F) +
  geom_errorbar(aes(ymin = DD13C.diss - SD.d13C, ymax = DD13C.diss + SD.d13C)) +
  geom_errorbarh(aes(xmin = Conc.mug.L - Conc.SD, xmax = Conc.mug.L + Conc.SD)) +
```

```
  geom_point(aes(size = timeSinceApp)) +
  theme_bw() +
  scale_size_continuous(range = c(1, 4)) +
  labs(size="Days post appl.") +
  theme(axis.title.y = element_blank()) +
  #scale_y_continuous(breaks=c(0, 1, 2, 3 , 4 ,5) ) +
  scale_y_continuous(breaks=seq(1,5,1)) +
  #ylab(expression(paste({Delta~delta}^"13","C", ' (\u2030)'))) +
  xlab(expression(paste("S-MET Outlet Concentration ", {({mu}*g / L)}))) +
  annotate("text", x = 20, y = 2.7, label = as.character(water_r_label), parse = T, size = 3.5) +
  annotate("text", x = 20, y = 2.3, label = water_p_label, parse = T, size = 3.5)


waterIsoConc
```




```
#ggsave(waterIsoConc , filename = "DDvsConc_water.png", width = 8, height = 5, units = "in", scale = 1)
```

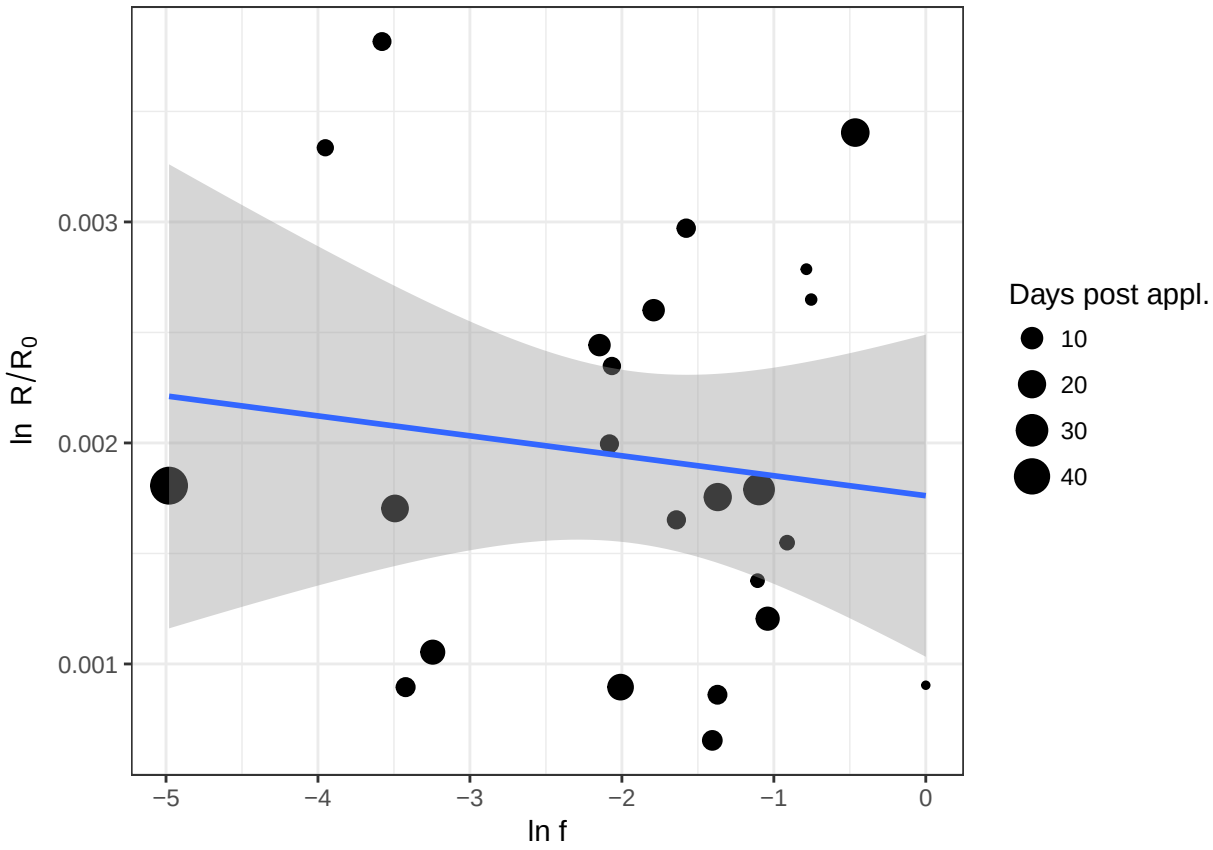
Water Rayleigh

```
waterModel<-lm(yRaleigh~xRaleigh, data= waterClean)
summary(waterModel)
```

```
##
## Call:
## lm(formula = yRaleigh ~ xRaleigh, data = waterClean)
```

```
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0012395 -0.0006644 -0.0001595  0.0006812  0.0017769
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.800e-03  3.043e-04   5.915 4.98e-06 ***
## xRaleigh    -6.701e-05  1.149e-04  -0.583    0.566
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0008992 on 23 degrees of freedom
## Multiple R-squared:  0.01457,    Adjusted R-squared:  -0.02828
## F-statistic:  0.34 on 1 and 23 DF,  p-value: 0.5655
```

```
minX <- confint(waterModel, "xRaleigh", level = 0.95)[1]*1000
maxX <- confint(waterModel, "xRaleigh", level = 0.95)[2]*1000

cofwater <- as.numeric(coef(waterModel)[2]*1000)
se <- summary(waterModel)$coef[[4]]*1000

CI95 = maxX - cofwater


waterRaleigh <- ggplot(data = subset(waterClean, (!is.na(yRaleigh) & xRaleigh > -7)), aes(x=xRaleigh, y=
  geom_point(aes(size = timeSinceApp)) +
  theme_bw() +
  scale_size_continuous(range = c(1, 6)) +
  labs(size="Days post appl.") +
  xlab("ln f") +
  ylab("ln R/Ro") +
  ylab(expression(paste("ln  ",  R / R['0'] ))) +
  stat_smooth(data= subset(waterClean, (!is.na(yRaleigh) & xRaleigh > -7)), method = "lm", formula = y~x

waterRaleigh
```




```
# ggsave(waterRaleigh, filename = "lnDDvslnConc_water.png", width = 8, height = 5, units = "in", scale =
# Date conversion correct:
sum(is.na(waters$Date.ti)) == 0
```

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

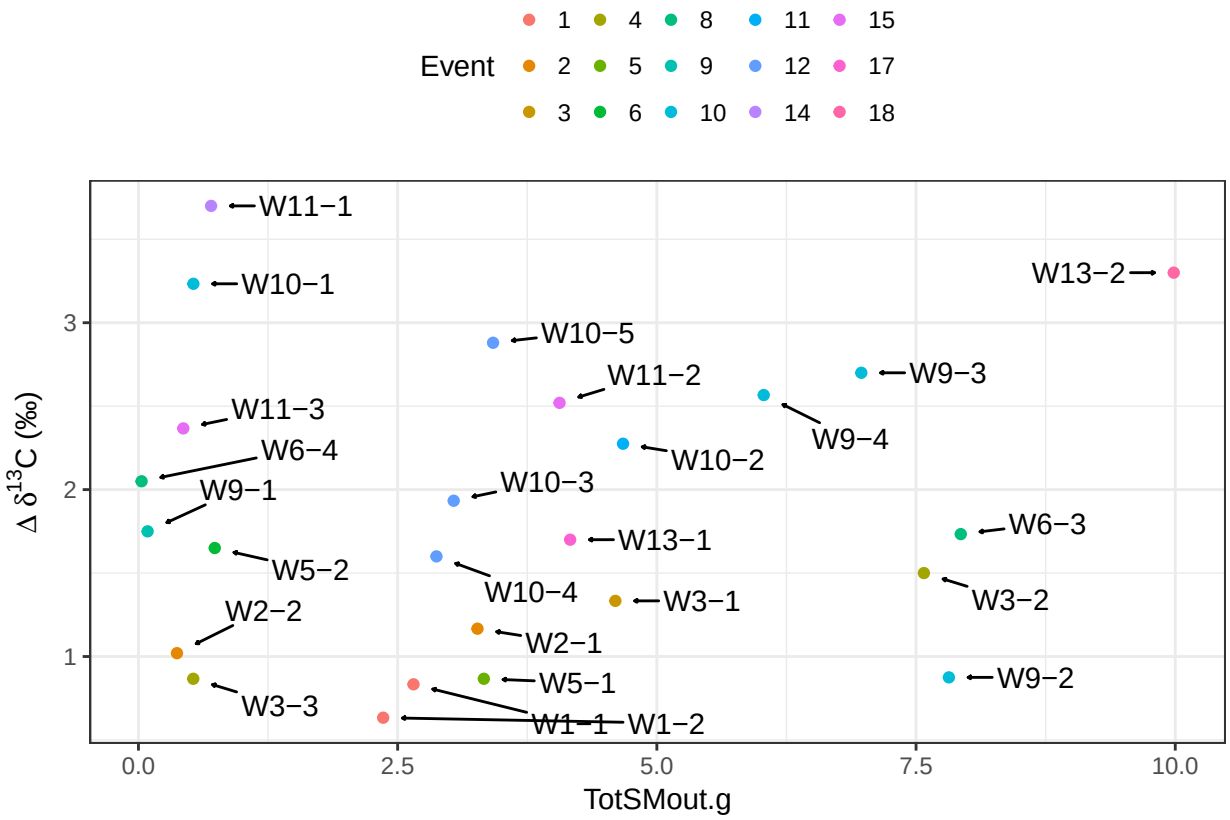
```
str(waters)
```

```
## 'data.frame':    51 obs. of  110 variables:
##  $ Date.ti               : POSIXct, format: "2016-03-25 00:04:00" "2016-03-25 12:04:00" ...
##  $ WeekSubWeek           : Factor w/ 51 levels "W0-0x","W0-1",..: 1 2 3 4 5 6 26 27 28 29 ...
##  $ tf                    : Factor w/ 51 levels "2016-03-25 12:02:00",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ iflux                 : num  1.25 1.12 1.31 1.46 16.33 ...
##  $ fflux                 : num  1.13 1.31 1.46 16.45 15.18 ...
##  $ changeflux            : num  -0.119 0.189 0.148 14.989 -1.15 ...
##  $ maxQ                  : num  1.25 1.38 1.64 38.4 18.67 ...
##  $ minQ                  : num  1.118 1.082 0.929 1.449 13.201 ...
##  $ dryHrsIni             : num  0.0167 0.0333 0.2667 0.1167 4.2167 ...
##  $ dryHrsMax             : num  2.75 24.52 13.32 4.2 5.43 ...
##  $ dryHrsAve             : num  0.745 7.827 4.859 1.289 1.314 ...
##  $ noEventHrsIni         : num  0.0167 6.0167 47.3 66.1333 1.65 ...
##  $ noEventHrsMax         : num  6 47.28 66.12 72.1 6.37 ...
##  $ noEventHrsAve         : num  3.01 26.65 56.71 30.4 3.33 ...
##  $ Duration.Hrs          : num  12 82.5 37.6 27.3 23.1 ...
##  $ chExtreme             : num  -0.13 0.256 0.33 36.944 -3.133 ...
##  $ Peak                  : int  NA NA NA 1 NA NA 2 NA NA 3 ...
##  $ Markers               : num  NA NA NA 16.9 NA ...
```

```
##  $ TimeDiff              : Factor w/ 18 levels "106","136","150",..: NA NA NA 10 NA NA 2 NA NA 11 ..
##  $ AveDischarge.m3.h     : num  1.2 1.21 1.28 14.32 15.53 ...
##  $ Volume.m3             : num  14.4 100.2 48.3 390.4 359.2 ...
##  $ Sampled.Hrs           : num  12 82.5 37.6 27.3 23.1 ...
##  $ Sampled               : Factor w/ 2 levels "Not Sampled",..: 1 2 1 2 2 1 2 2 1 2 ...
##  $ CumRain.mm            : num  2.8 7.6 7.6 16.8 6 9.4 5.4 0.8 5.4 20 ...
##  $ RainInt.mmhr          : num  0.234 0.0921 0.2019 0.6161 0.2594 ...
##  $ Conc.mug.L            : num  0.246 0.246 3.517 6.788 6.561 ...
##  $ Conc.SD               : num  0.0193 0.0193 0.1544 0.2894 0.1906 ...
##  $ Vol.SPE.L             : num  0.57 0.57 NA 1.91 1.91 ...
##  $ Conc.in500uL          : num  NA 140 NA 12966 12532 ...
##  $ OXA_mean              : num  74 74 71.5 69 70 59 48 64 69.5 75 ...
##  $ OXA_SD                : num  39 39 42 45 22 31 40 32 46.5 61 ...
##  $ ESA_mean              : num  45 45 58.5 72 70 76.5 83 76 65.5 55 ...
##  $ ESA_SD                : num  59 59 58 57 25 38.5 52 11 13.5 16 ...
##  $ N.x                   : int  NA NA NA 3 3 NA 3 5 NA 3 ...
##  $ diss.d13C             : num  NA NA NA -31.5 -31.7 ...
##  $ SD.d13C               : num  NA NA NA 0.115 0.153 ...
##  $ N_d13C.diss           : int  NA NA NA 3 3 NA 3 5 NA 3 ...
##  $ MES.mg.L              : num  NA 53.4 NA 62.5 22.5 ...
##  $ MES.sd                : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MO.mg.L               : num  NA 0 NA 0.001 0.0001 NA 0.0001 0.0001 NA 0.0058 ...
##  $ Conc.Solids.mug.gMES  : num  0.645 0.645 0.385 0.126 0.436 ...
##  $ Conc.Solids.ug.gMES.SD: num  0.0232 0.0232 0.0252 0.0271 0.1232 ...
##  $ N.y                   : int  NA NA NA NA NA NA 3 3 NA NA ...
##  $ filt.d13C             : num  NA NA NA NA NA ...
##  $ filt.SD.d13C          : num  NA NA NA NA NA ...
##  $ DD13C.diss            : num  NA NA NA 0.833 0.633 ...
##  $ DD13C.filt            : num  NA NA NA NA NA ...
##  $ ExpMES.Kg             : num  5.35 5.35 14.88 24.4 8.08 ...
##  $ DissSmeto.mg          : num  3.54 24.6 170.04 2649.91 2357 ...
##  $ DissSmeto.mg.SD       : num  0.278 1.934 7.463 112.98 68.486 ...
##  $ DissSmeto.g           : num  0.00354 0.0246 0.17004 2.64991 2.357 ...
##  $ DissSmeto.g.SD        : num  0.000278 0.001934 0.007463 0.11298 0.068486 ...
##  $ DissOXA.mg            : num  1067 7411 3457 26935 25147 ...
##  $ DissOXA.mg.SD         : num  562 3906 2031 17567 7903 ...
##  $ DissOXA.g             : num  1.07 7.41 3.46 26.94 25.15 ...
##  $ DissOXA.g.SD          : num  0.562 3.906 2.031 17.567 7.903 ...
##  $ DissESA.mg            : num  649 4507 2828 28106 25147 ...
##  $ DissESA.mg.SD         : num  851 5909 2804 22251 8981 ...
##  $ DissESA.g             : num  0.649 4.507 2.828 28.106 25.147 ...
##  $ DissESA.g.SD          : num  0.851 5.909 2.804 22.251 8.981 ...
##  $ FiltSmeto.mg          : num  3.45 3.45 5.73 3.07 3.52 ...
##  $ FiltSmeto.mg.SD       : num  0.124 0.124 0.374 0.66 0.996 ...
##  $ FiltSmeto.g           : num  0.00345 0.00345 0.00573 0.00307 0.00352 ...
##  $ FiltSmeto.g.SD        : num  0.000124 0.000124 0.000374 0.00066 0.000996 ...
##  $ TotSMout.mg           : num  6.99 28.06 175.77 2652.98 2360.52 ...
##  $ TotSMout.mg.SD        : num  0.216 1.37 5.284 79.89 48.432 ...
##  $ TotSMout.g            : num  0.00699 0.02806 0.17577 2.65298 2.36052 ...
##  $ TotSMout.g.SD         : num  0.000216 0.00137 0.005284 0.07989 0.048432 ...
##  $ FracDiss              : num  0.506 0.877 0.967 0.999 0.999 ...
##  $ FracFilt              : num  0.49352 0.12301 0.03261 0.00116 0.00149 ...
##  $ MELsm.g               : num  1.65 11.44 6.13 54.26 49.6 ...
##  $ MELsm.g.SD            : num  0.537 3.729 1.835 15.13 6.441 ...
```

```
##  $ CumOutDiss.g          : num  0.00354 0.02815 0.19818 2.84809 5.2051 ...
##  $ CumOutFilt.g          : num  0.00345 0.0069 0.01263 0.01571 0.01923 ...
##  $ CumOutSmeto.g         : num  0.00699 0.03505 0.21082 2.8638 5.22432 ...
##  $ CumOutMELsm.g         : num  1.65 13.09 19.22 73.48 123.07 ...
##  $ Appl.Mass.g           : num  31670 0 0 0 0 ...
##  $ Appl.Mass.g.OT        : num  24477 0 0 0 0 ...
##  $ Appl.Mass.g.N         : num  8429 0 0 0 0 ...
##  $ Appl.Mass.g.T         : num  6904 0 0 0 0 ...
##  $ Appl.Mass.g.S         : num  16337 0 0 0 0 ...
##  $ Appl.Mass.g.N.OT      : num  8429 0 0 0 0 ...
##  $ Appl.Mass.g.T.OT      : num  2727 0 0 0 0 ...
##  $ Appl.Mass.g.S.OT      : num  13321 0 0 0 0 ...
##  $ iniCo.ug.g.N          : num  8.46 8.46 8.46 8.46 8.46 ...
##  $ iniCo.ug.g.T          : num  7.09 7.09 7.09 7.09 7.09 ...
##  $ iniCo.ug.g.S          : num  12.4 12.4 12.4 12.4 12.4 ...
##  $ timeSinceApp          : num  0.5 3.9 5.5 6.6 7.6 11.6 12.6 14 20.6 2.2 ...
##  $ timeSinceApp.N        : num  0.5 3.9 5.5 6.6 7.6 11.6 12.6 14 20.6 2.2 ...
##  $ timeSinceApp.T        : num  0.5 3.9 5.5 6.6 7.6 11.6 12.6 14 20.6 2.2 ...
##  $ timeSinceApp.S        : num  0.5 3.9 5.5 6.6 7.6 11.6 12.6 14 20.6 2.2 ...
##  $ Appl.Mass.g.NoSo      : num  31670 0 0 0 0 ...
##  $ timeSinceApp.NoSo     : num  0.5 3.9 5.5 6.6 7.6 11.6 12.6 14 20.6 2.2 ...
##  $ CumAppMass.g          : num  31670 31670 31670 31670 31670 ...
##  $ CumAppMass.g.OT       : num  24477 24477 24477 24477 24477 ...
##  $ CumAppMass.g.N        : num  8429 8429 8429 8429 8429 ...
##  $ CumAppMass.g.T        : num  6904 6904 6904 6904 6904 ...
##  $ CumAppMass.g.S        : num  16337 16337 16337 16337 16337 ...
##  $ CumAppMass.g.N.OT     : num  8429 8429 8429 8429 8429 ...
##    [list output truncated]
```

```
ggplot(waterClean, aes(x=TotSMout.g, y=DD13C.diss))+
  geom_point(aes(group = Event, colour = Event))+
  theme_bw() +
  theme(legend.position="top"
        # axis.title.x = element_blank(),
        ) +
  guides(col = guide_legend(nrow =  3)) + #,
         # shape = guide_legend(nrow =  3)) +
  ylab(expression(paste({Delta~delta}^"13","C", ' (\u2030)'))) +
  # xlab(expression(paste("Conc. S-Meto.  ", {({mu}*g / L)}))) +
  geom_text_repel(aes(label=WeekSubWeek),
                  arrow = arrow(length = unit(0.005, 'npc'), type = "closed"),
                  force = 1,
                  point.padding = unit(1.0, 'lines'),
                  max.iter = 2e3,
                  nudge_x = .2)
```



Correlations Waters

```
cor.test(waters$Conc.mug.L, waters$diss.d13C)
```

```
##
##  Pearson's product-moment correlation
##
## data:  waters$Conc.mug.L and waters$diss.d13C
## t = -0.44534, df = 23, p-value = 0.6602
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4704067  0.3141462
## sample estimates:
##        cor
## -0.0924619
```

```
#cor.test(waters$TotSMout.g, waters$diss.d13C)

#esaoxa <- waters$MELsm.g-waters$TotSMout.g
# cor.test(esaoxa, waters$diss.d13C)
```

Merge Soil and Water data frames

Objective is to plot both soils and water temporaly

Outlet Isotope Shifts (DD)

In the same plot consider this secondary axis, where the secondary axis is a formulation of the first:

ggplot(mpg, aes(displ, hwy)) + geom_point() + scale_y_continuous("mpg (US)", sec.axis = sec_axis(~ . * 1.20, name = "mpg (UK)"))

The equation for the secondary y-axis will be:

$$B = (1 - (\frac{1000 + \delta^{13}C_0 + \Delta\delta^{13}C}{1000 + \delta^{13}C_0})^{\frac{1000}{\epsilon}}) * 100$$

Or this: https://github.com/tidyverse/ggplot2/wiki/Align-two-plots-on-a-page

```
# SD min. selection line 914 (for dissolved)
WaterSoils <- merge(waterClean, soils, by = "Date.ti", all = T)

# Save for SI Graphs (book 11.2)
write.csv2(WaterSoils, "Data/WaterSoils_R.csv", row.names = F)

# Choose and rearrange variables
# names(WaterSoils)
wsSmall <- WaterSoils[c("Date.ti", "WeekSubWeek", "ID.N", "Event.y", "Events",
                    "maxQ", "AveDischarge.m3.h",
                    "dryHrsIni", "dryHrsMax", "dryHrsAve", "noEventHrsIni", "noEventHrsMax", "noEventHr
                     "CumRain.mm", "RainInt.mmhr", ## Rainfall is per subsample (See Book 3)
                     "DD13C.diss", "SD.d13C.x",
                     "DD13C.filt", "filt.SD.d13C" ,
                     "DD13C.Talweg", "comp.d13C.SD.Talweg",
                     "DD13C.South", "comp.d13C.SD.South",
                     "DD13C.North", "comp.d13C.SD.North",
                     "DD13.Bulk", "BulkCatch.d13.SD")]

names(wsSmall)
```

```
##  [1] "Date.ti"             "WeekSubWeek"         "ID.N"
##  [4] "Event.y"             "Events"              "maxQ"
##  [7] "AveDischarge.m3.h"   "dryHrsIni"           "dryHrsMax"
## [10] "dryHrsAve"           "noEventHrsIni"       "noEventHrsMax"
## [13] "noEventHrsAve"       "CumRain.mm"          "RainInt.mmhr"
## [16] "DD13C.diss"          "SD.d13C.x"           "DD13C.filt"
## [19] "filt.SD.d13C"        "DD13C.Talweg"        "comp.d13C.SD.Talweg"
## [22] "DD13C.South"         "comp.d13C.SD.South"  "DD13C.North"
## [25] "comp.d13C.SD.North"  "DD13.Bulk"           "BulkCatch.d13.SD"
```

```
keepCorrTest <- c("DD13C.diss",
                  "DD13C.Talweg",
                  "DD13C.South",
                  "DD13C.North",
                  "DD13.Bulk")

wsTest <- wsSmall[ , (names(wsSmall) %in% keepCorrTest)]

names(wsSmall) <- c("Date", "Week", "IDSoil", "Event", "Events",
                    "Qmax", "Qmean",
                    "dryHrsIni", "dryHrsMax", "dryHrsAve", "noEventHrsIni", "noEventHrsMax", "noEventHrs
```

```
                    "CumRain", "RainInt", ## Rainfall is per subsample (See Book 3)
                    "diss.measure", "diss.SD",
                    "filt.measure", "filt.SD",
                    "Talweg.measure", "Talweg.SD",
                    "South.measure", "South.SD",
                    "North.measure", "North.SD",
                    "BulkDD.measure", "BulkDD.SD"
                    )

wsTest <- wsTest[7:length(wsTest$DD13C.diss) , ]
wsTest$DD13.Bulk <- na.locf(wsTest$DD13.Bulk)
wsTest$DD13C.Talweg <- na.locf(wsTest$DD13C.Talweg)
wsTest$DD13C.South <- na.locf(wsTest$DD13C.South)
wsTest$DD13C.North <- na.locf(wsTest$DD13C.North)

cor.test(wsTest$DD13.Bulk, wsTest$DD13C.diss, method = "pearson", use = "pairwise.complete.obs")
```

```
##
##  Pearson's product-moment correlation
##
## data:  wsTest$DD13.Bulk and wsTest$DD13C.diss
## t = 4.3954, df = 21, p-value = 0.0002524
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3918106 0.8592469
## sample estimates:
##       cor
## 0.6922163
```

```
cor.test(wsTest$DD13C.Talweg, wsTest$DD13C.diss, method = "pearson", use = "pairwise.complete.obs")
```

```
##
##  Pearson's product-moment correlation
##
## data:  wsTest$DD13C.Talweg and wsTest$DD13C.diss
## t = 1.4919, df = 21, p-value = 0.1506
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1176536  0.6400872
## sample estimates:
##       cor
## 0.3095614
```

```
cor.test(wsTest$DD13C.North, wsTest$DD13C.diss, method = "pearson", use = "pairwise.complete.obs")
```

```
##
##  Pearson's product-moment correlation
##
## data:  wsTest$DD13C.North and wsTest$DD13C.diss
## t = 3.6, df = 21, p-value = 0.001683
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2757861 0.8209219
## sample estimates:
##       cor
```

```
## 0.6177614
```

```
cor.test(wsTest$DD13C.South, wsTest$DD13C.diss, method = "pearson", use = "pairwise.complete.obs")
```

```
##
##  Pearson's product-moment correlation
##
## data:  wsTest$DD13C.South and wsTest$DD13C.diss
## t = 3.8001, df = 21, p-value = 0.001047
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3068618 0.8316920
## sample estimates:
##       cor
## 0.6383231
```

```
# Conventional way of melting won't work if we need SDs.
# ws <- melt(wsSmall, id=c("Date.ti", "timeSinceApp.x", "Events", "Event.x"))

# Need to rename the columns so that I can use separate and spread from the package tidyr
#names(wsSmall)[-1][seq(2, length(names(wsSmall)) - 1, 2)] <-
#  paste0(names(wsSmall)[-1][seq(1, length(names(wsSmall)) - 1, 2)], "-SD")
#names(wsSmall)[-1][seq(1, length(names(wsSmall)) - 1, 2)] <-
#  paste0(names(wsSmall)[-1][seq(1, length(names(wsSmall)) - 1, 2)], "-measure")

wstidier <- wsSmall %>%
  gather(measure, value, -Date, -IDSoil, -Event, -Events, -Week,
         -Qmax, -Qmean,
         -CumRain, -RainInt,
         -dryHrsIni, -dryHrsMax, -dryHrsAve,
         -noEventHrsMax,
         -noEventHrsIni, -noEventHrsAve) %>% # Melts data frame
  separate(measure, into = c("Location", "temporary_var")) %>% # parses the sep = "." into...
  # Location will be first string of variable name
  spread(temporary_var, value)

wstidier$Type <- ifelse(wstidier$Location == "diss", "Dissolved (Outlet)",
                  ifelse(wstidier$Location == "filt", "Sediment",
                         "Top Soil"))

wstidier$Source <- ifelse(wstidier$Location == "diss", "Outlet",
                  ifelse(wstidier$Location == "filt", "Outlet",
                         ifelse(wstidier$Location == "South", "South",
                                ifelse(wstidier$Location == "Talweg", "Valley",
                                       ifelse(wstidier$Location == "BulkDD", "Bulk",
                                       "North"))) ))

wstidier$Source <- as.factor(wstidier$Source)
wstidier$Type <- as.factor(wstidier$Type)
wstidier$IDSoil <- as.factor(wstidier$IDSoil)
wstidier$Event <- as.numeric(wstidier$Event)

# Copy all data
wstidierAll <- wstidier
```

```
levels(wstidier$Source)
```

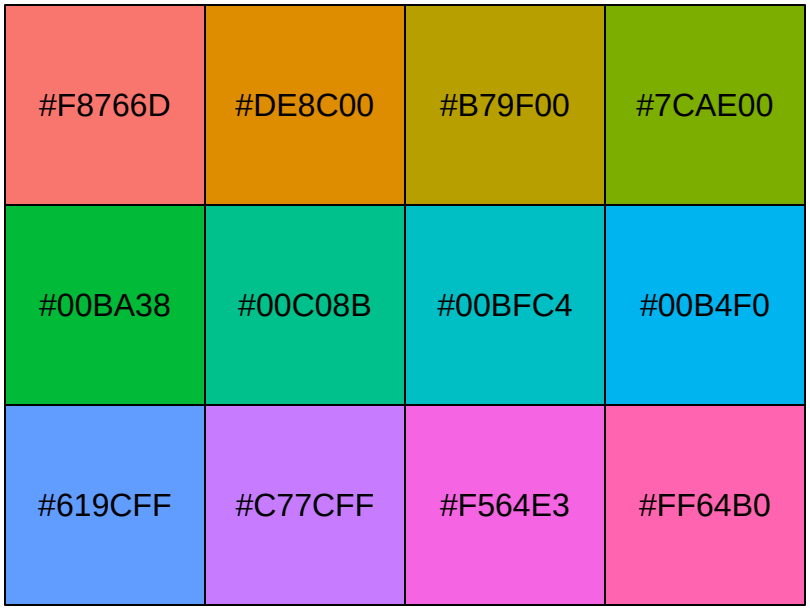
```
## [1] "Bulk"   "North"  "Outlet" "South"  "Valley"
```

```
levels(wstidier$Type)
```

```
## [1] "Dissolved (Outlet)" "Sediment"           "Top Soil"
```

```
#wstidier$Source <- factor(wstidier$Source, levels = c("Bulk", "Plateau","Valley", "Outlet"))
wstidier$Source <- factor(wstidier$Source, levels = c("Bulk", "North", "Valley", "South", "Outlet"))

wstidier$Type <- factor(wstidier$Type, levels = c("Top Soil", "Dissolved (Outlet)", "Sediment" ))


# epsilon
#epsilon_field
#initialDelta

#wstidier$DegField <- (1-((1000 + d13Co + wstidier$measure)/(1000+d13Co))^(1000/epsilon_field))*100
#wstidier$DegLab <- (1-((1000 + d13Co + wstidier$measure)/(1000+d13Co))^(1000/epsilon_lab))*100

#wstidier$DegDiff <- (wstidier$DegField - wstidier$DegLab)

wstidier$Location <- as.factor(wstidier$Location)
#wstidier$Week <- as.factor(wstidier$Week)

#library(dplyr)
#detach("package:plyr")
#sumary <- na.omit(wstidier) %>%
#  group_by(Type) %>%
#  summarise(mean = mean(DegDiff))
```

```
# library(scales)
## Color palette
#
show_col(hue_pal()(12))
```



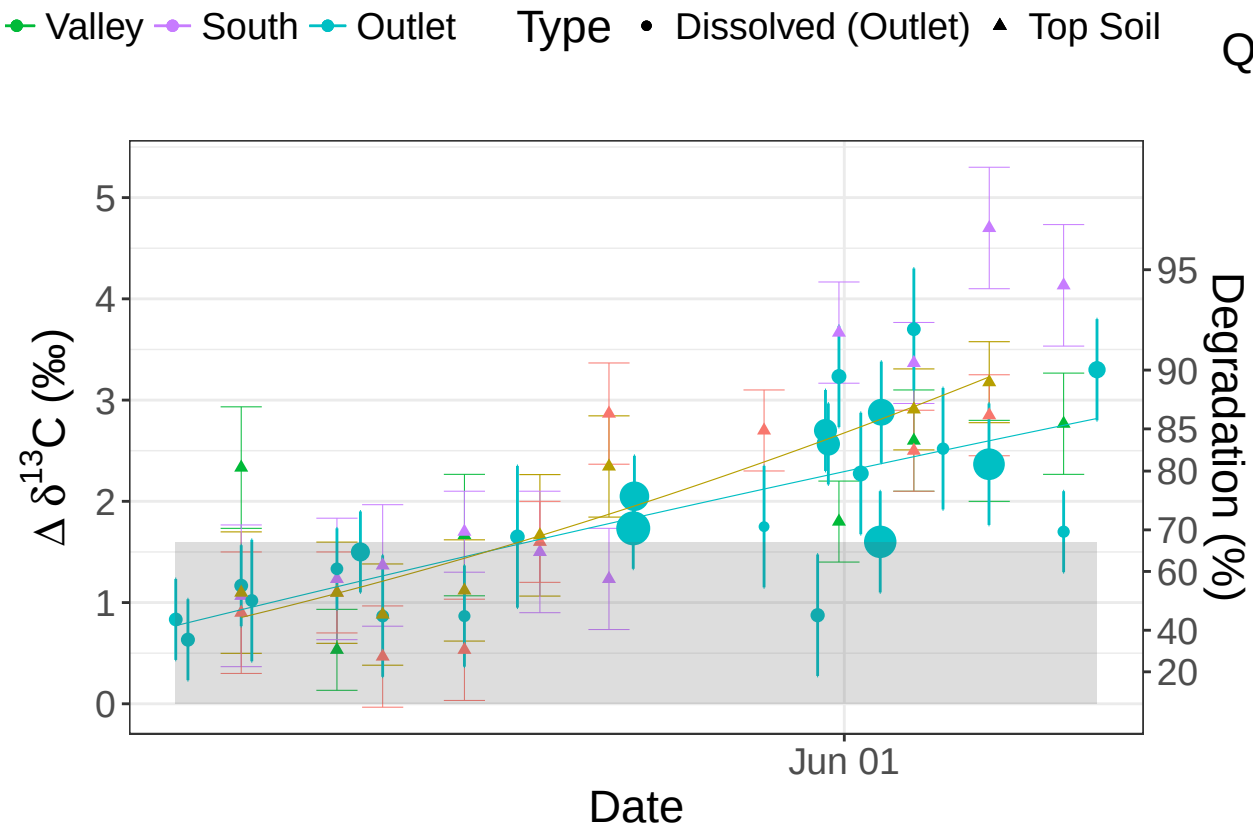

```
# Bulk, North, Valley, South, Outlet
# "black", "#F8766D", "#00BA38", "#DE8C00", ("#619CFF" / "#00B4F0" / "#00BFC4")
```

Lab Enrichment plot

```
# Dissolved has been selected, but not soils or filters
wstidier2 = subset(wstidier, SD <= 0.54 & Date <= as.POSIXct('2016-06-24 14:52:00', tz = "EST"))

# Propagating error
wstidier2$SD = round((wstidier2$SD^2 + initialDeltaError^2)^(1/2), 1)

write.csv2(wstidier2, "Data/OutletSoilsPlot_R.csv", row.names = F)

mindate = min(wstidier2$Date)
maxdate = max(wstidier2$Date)

pd <- position_dodge(width = 0.5)
limits_DdC <- aes(ymin=measure-SD, ymax=measure+SD, colour = Source)

wsALL_lab <- ggplot(data = wstidier2, aes(x = Date, y = measure, group = Source) )+
  geom_errorbar(data=subset(wstidier2, Source == 'Bulk'), limits_DdC, size=0.2) +
  geom_errorbar(data=subset(wstidier2, Source == 'South'
                            | Source == 'North'
                            | Source == 'Valley'
                            ), limits_DdC, size=0.1) +
```

```
geom_errorbar(data=subset(wstidier2, Source == 'Outlet'), limits_DdC) +
geom_point(data=subset(wstidier2, (Source == 'South'
                                   | Source == 'North'
                                   | Source == 'Valley'
                                   )
                       # & Date > as.POSIXct('2016-05-14 08:04:00')
                       ),
           aes(shape = Type,
               colour = Source)) +
geom_point(data=subset(wstidier2, Source == 'Outlet'), aes(shape = Type, colour = Source, size = Qmean
geom_point(data=subset(wstidier2, Source == 'Bulk'), aes(shape = Type, colour = Source)) +

# Water
stat_smooth(data=subset(wstidier2,
                        (Source == "Outlet"
                         # & Event > 1
                         & Type == "Dissolved (Outlet)")),
            method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = 'Outlet'), alpha = 0.2, size
# North
stat_smooth(data=subset(wstidier2,
                        (Source == "Bulk" )), #| Source == "South" )),
            method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = 'Bulk'), alpha = 0.2, size=0
#stat_smooth(data=subset(wstidier2,
#                         (Source == "South")),
#            method = "lm", formula = y ~ poly(x, 2), se = T, aes(colour = 'South'), alpha = 0.2, size
theme_bw() +

scale_x_datetime(breaks = date_breaks("3 months"), labels = date_format("%b %d")) +
#scale_x_datetime(breaks = date_breaks("2 weeks"), labels = date_format("%b %d")) +
theme(text = element_text(size=17),
      legend.position="top"
      #axis.title.x = element_blank()
      # axis.text.x=element_text(angle = 45, hjust = 1)
      ) +
# geom_smooth(data=subset(ws, Source != "Outlet"), method = "lm", formula = y ~ poly(x, 2)) +
xlab("Date") +
#ylab(expression(paste({Delta~delta}^"13","C", ' (\u2030)'))) +
scale_y_continuous(
  breaks=c(0, 1, 2, 3 , 4 ,5),
  expression(paste({Delta~delta}^"13","C", ' (\u2030)')),
  sec.axis = sec_axis(trans = ~ (1-((1000 + d13Co + .)/(1000+d13Co))^(1000/epsilon_lab))*100 ,
                      name = "Degradation (%)",
                      #name = element_blank(),
                      breaks=c(20, 40, 60, 70, 80, 85, 90, 95) )# breaks=seq(20, 120, 15))
)  +
scale_color_manual(name= "Source",
                   # Actual order:
                   # Bulk, North, Outlet, South, Valley
                    values = c("#B79F00", "#F8766D", "#00BFC4", "#C77CFF", "#00BA38"
                               # working solution:
                               #c("black", "#F8766D", "#00BFC4", "#DE8C00", "#00BA38"
                               #"black", "#D55E00",  "#00BFC4",  "#B79F00", "#00BA38"
                               # Bulk, North, outlet, South, Valley
```

```
                                  #"#D55E00", "darkgreen", "dodgerblue"
                                  ),
                       breaks=c("Bulk", "North" , "Valley" ,"South",  "Outlet"),
                       labels=c("Bulk", "North" , "Valley" ,"South",  "Outlet")
                       ) +

  scale_size_continuous(range = c(1, 6), breaks= c(0, 50, 100, 150, 200, 300), limits = c(0, 300))+
  annotate("rect", xmin=mindate, xmax=maxdate, ymin=0, ymax=propagatedError, alpha=0.2)
  # scale_size_continuous(range = c(1, 3))

#
# Reds
# gold = "#B79F00"
# red-pink  = "#F8766D"
# "firebrick1",
# 'yellow', "orange1","red",
# pink = "#F564E3"

# Mono
# "gray35", "ghostwhite", 'gray99'

# Greens
# 'darkgreen','darkolivegreen3','darkseagreen3','darkseagreen1'
# dark green = "chartreuse4"
# darkish freen = "#00BA38"

# Blues
# purple = "blueviolet"
# "dodgerblue", "#00BFC4" (light blue), "#619CFF" (sharp blue),
#  "deepskyblue"

wsALL_lab
```



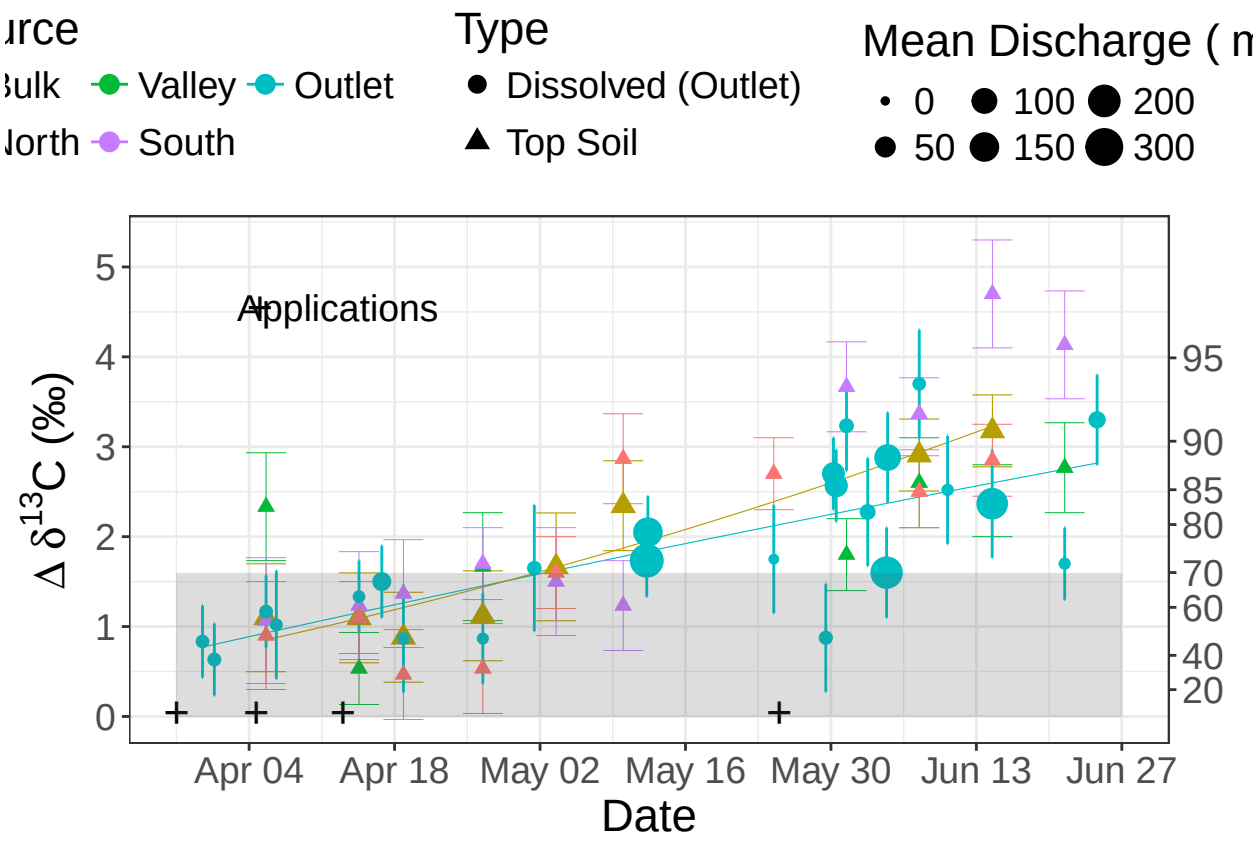

```
# ggplotly(wsALL_lab)
```

Field Enrichment Plot

```
mindate = as.POSIXct("2016-03-28 00:04:00" , tz = "EST") # min(wstidier2$Date)
maxdate = as.POSIXct("2016-06-27 00:01:00", tz = "EST")

wsALL_field <- ggplot(data = wstidier2, aes(x = Date, y = measure, group = Source) )+
  # Dissolved (Outlet) trend
  stat_smooth(data=subset(wstidier2,
                          (Source == "Outlet"
                           # & Event > 1
                           & Type == "Dissolved (Outlet)")),
              method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = 'Outlet'), alpha = 0.9, siz
  # Bulk trend
  stat_smooth(data=subset(wstidier2,
                          (Source == "Bulk" )), #| Source == "South" )),
              method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = 'Bulk'), alpha = 0.9, size=
  # Error bars
  geom_errorbar(data=subset(wstidier2, Source == 'Bulk'), limits_DdC, size=0.2) +
  geom_errorbar(data=subset(wstidier2, Source == 'South'
                            | Source == 'North'
                            | Source == 'Valley'
                            ), limits_DdC, size=0.1) +
```

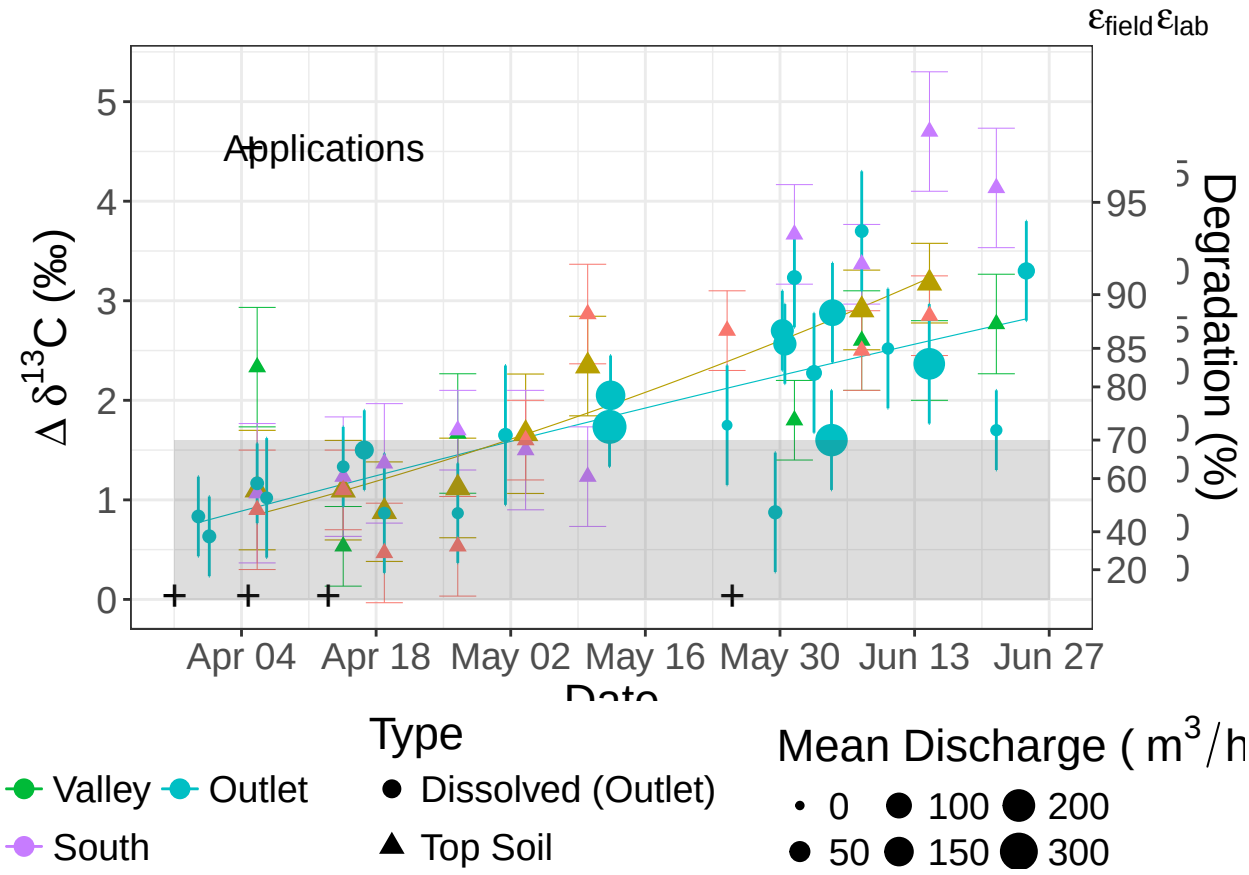
```
geom_errorbar(data=subset(wstidier2, Source == 'Outlet'), limits_DdC) +
# Data points
geom_point(data=subset(wstidier2, Source == 'Bulk'), aes(shape = Type, colour = Source), size=3) +
geom_point(data=subset(wstidier2, (Source == 'South'
                                   | Source == 'North'
                                   | Source == 'Valley'
                                   )), aes(shape = Type, colour = Source), size=2) +
geom_point(data=subset(wstidier2, Source == 'Outlet'), aes(shape = Type, colour = Source, size = Qmean
theme_bw() +
# Applications
annotate("text", x = as.POSIXct('2016-03-28 08:04:00'), y = 0,
         label = as.character(expression(paste( "+"))), parse = T, size = 6.0) +
annotate("text", x = as.POSIXct('2016-04-05 00:04:00'), y = 0,
         label = as.character(expression(paste( "+"))), parse = T, size = 6.0) +
annotate("text", x = as.POSIXct('2016-04-13 08:04:00'), y = 0,
         label = as.character(expression(paste( "+"))), parse = T, size = 6.0) +
annotate("text", x = as.POSIXct('2016-05-25 08:04:00'), y = 0,
         label = as.character(expression(paste( "+"))), parse = T, size = 6.0) +
# Title applics
annotate("text", x = as.POSIXct('2016-04-05 08:04:00'), y = 4.5,
         label = as.character(expression(paste( "+"))), parse = T, size = 6.0) +
 annotate("text", x = as.POSIXct('2016-04-12 08:04:00'), y = 4.5,
         label = as.character(expression(paste(" Applications"))), parse = T, size = 5.0) +
scale_x_datetime(breaks = date_breaks("2 weeks"), labels = date_format("%b %d")) +
#scale_x_datetime(breaks = date_breaks("1 month"), labels = date_format("%b %d")) +
theme(text = element_text(size=17),
      legend.position="top"
      # axis.title.x = element_blank()
      # axis.text.x=element_text(angle = 45, hjust = 1)
      ) +
xlab("Date") +
#ylab(expression(paste({Delta~delta}^"13","C", ' (\u2030)'))) +
scale_y_continuous(
  breaks=c(0, 1, 2, 3 , 4 ,5),
  expression(paste({Delta~delta}^"13","C", ' (\u2030)')),
  sec.axis = sec_axis(trans = ~ (1-((1000 + d13Co + .)/(1000+d13Co))^(1000/epsilon_field))*100 ,
                      name = element_blank(),
                      #name = "Degradation (%)",
                      breaks=c(20, 40, 60, 70, 80, 85, 90, 95) )# breaks=seq(20, 120, 15))
)  +
scale_color_manual(name= "Source",
                   values = c("#B79F00", "#F8766D", "#00BFC4", "#C77CFF", "#00BA38"
                              # c("black", "#F8766D", "#00BFC4", "#DE8C00", "#00BA38"
                              #  "black", "#D55E00",  "#00BFC4",  "#B79F00", "#00BA38"
                               # Bulk, North, outlet, South, Valley
                              ),
                   breaks=c("Bulk", "North" , "Valley" ,"South",  "Outlet"),
                   labels=c("Bulk", "North" , "Valley" ,"South",  "Outlet")
                  ) +
scale_size_continuous(range = c(1, 6), breaks= c(0, 50, 100, 150, 200, 300), limits = c(0, 300)) +
# scale_size_continuous(range = c(1, 3)) +
guides(col = guide_legend(order = 1,
                          #title=expression("Source"),
```

```
                            #title.vjust = -1,
                            nrow = 2,
                            keyheight = 1.5,
                            title.position = "top"
                            ),
         shape=guide_legend(title=("Type"),
                            order = 2,
                            nrow=2,
                            title.position = "top",
                            keyheight = 1.5,
                            title.vjust = NULL, label.vjust = NULL
                            ),
         size = guide_legend(order = 3,
                             #title=expression("Mean Discharge"),
                             title=expression("Mean Discharge (" ~m^3 / h~")" ),
                             nrow=2,
                             title.position = "top"
                             # title.vjust = .26
                             #keyheight = 0,
                             #label.vjust = 0
                              )) +
  annotate("rect", xmin=mindate, xmax=maxdate, ymin=0, ymax=propagatedError, alpha=0.2)

#ggplotly(wsALL_field)
wsALL_field
```



Join all figures

```
#wsALL_lab
#wsALL_field
#wsPlot
# ggsave(wsALL, filename = "WaterSoilvsTime.png", width = 8, height = 5, units = "in", scale = 1)
# ggsave(wsALL, filename = "WaterBulkvsTime.png", width = 8, height = 5, units = "in", scale = 1)

wsALL_field_noLeg <- wsALL_field + theme(legend.position='none')
wsALL_lab_noLeg <- wsALL_lab + theme(legend.position='none')
wsAll_field_Leg  <- get_legend(wsALL_field)

labely1 = expression(epsilon ["field"])
labely2 = expression(epsilon ["lab"])

label <- substitute(paste(epsilon, " = ", epsilon_f, ", Field", epsilon, " = " , epsilon_l),
                    list(epsilon_f = signif(epsilon_field, 2), epsilon_l = signif(epsilon_lab, 2) ))

label2 <- substitute(paste(epsilon ["field"] , " = ", epsilon_f, " \u00B1 ", "0.53" ,"\u2030"),
                     list(epsilon_f = signif(epsilon_field, 3)))

label3 <- substitute(paste(epsilon ["lab"] , " = ", epsilon_l, " \u00B1 ", "0.47" ,"\u2030"),
                     list(epsilon_l = signif(epsilon_lab, 3)))
# adding label via ggdraw, in the ggdraw coordinates


wsALL <- ggdraw() +
  draw_plot(wsALL_lab_noLeg, x=0, y = 0.15, width = 1, height = 0.82) + # bottom
  draw_plot(wsALL_field_noLeg, x=0, y=.15, width = 0.945, height = .82) + # top
  draw_label(label2, x= .886, y = .10, size = 15) + # Epsilon field (bottom)
  draw_label(label3, x= .89, y = .05, size = 15) + # Epsilon lab (bottom)
  draw_label(labely1 , x= .90, y = .98, size = 14) + # Epsilon field (top)
  draw_label(labely2 , x= .95, y = .98, size = 14) + # Epsilon lab (top)
  draw_plot(wsAll_field_Leg, x=0.2, y=0.0, width = 0.50, height = 0.15)

wsALL
```



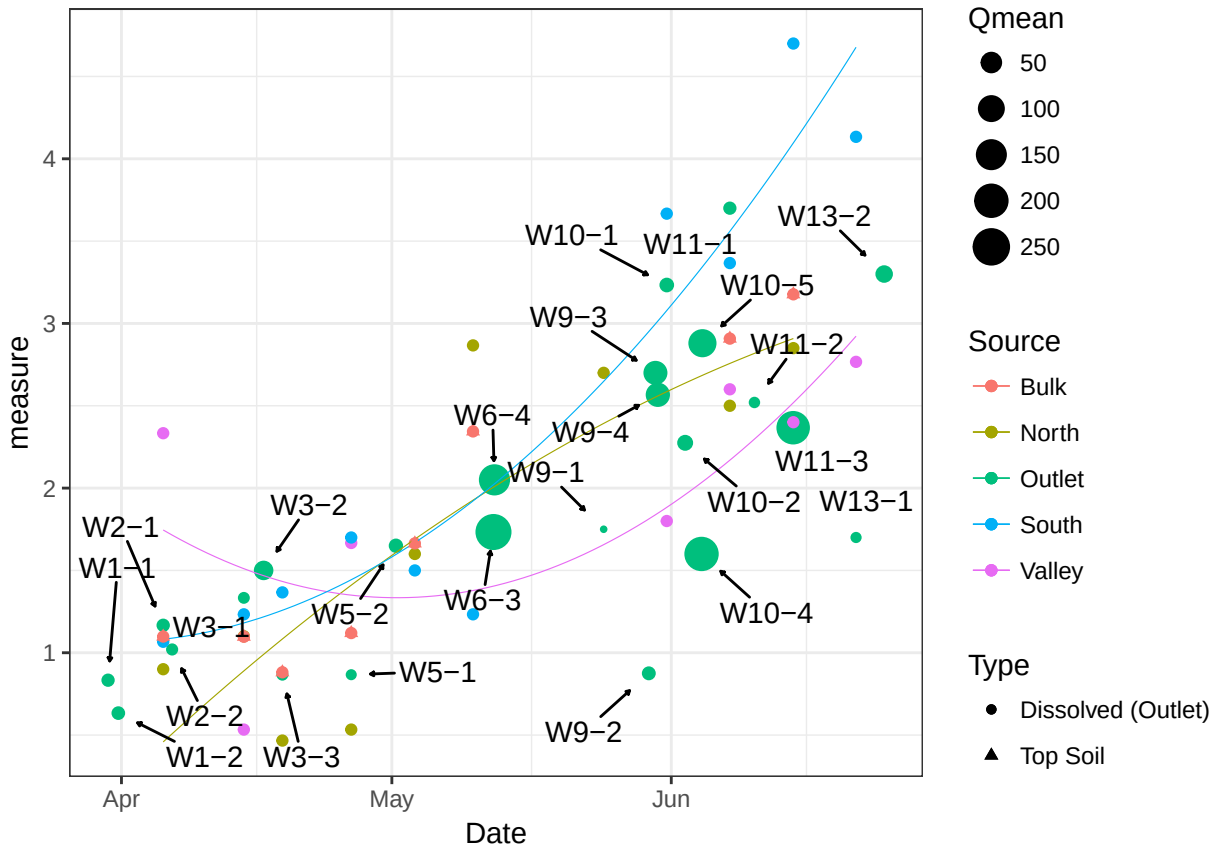

```
# ggsave(wsALL, filename = "images/WaterSoilvsTime.png", width = 8, height = 5, units = "in", scale = 1

SAVE = F
PC = T
if (SAVE){
  if (PC) {
    # cairo and ggdraw having issues.. works fine on MAC though
    ggsave(wsALL,
           filename = "D:/Documents/these_pablo/WriteUp/Alteck_PNAS_LaTeX/images/CatchOutlet.pdf",
           device= pdf, dpi = 600,  scale = 1, # )# ,
           width = 11.4, height = 6.4)
  } else {
    ggsave(wsALL,
     filename = "/Users/DayTightChunks/Documents/PhD/Writeups/PNAS/Alteck_PNAS_LaTeX/images/CatchOutlet
      device=cairo_pdf, dpi = 600,  scale = 1, # )# ,
       width = 11.4, height = 6.4)
  }
}


#install.packages("extrafont")
#library(extrafont)
```

Encodings

```
#pdf('test.pdf',encoding="MacRoman")
#plot.new()
#text(0,labels="\u2030")
#dev.off()
```

Check Soils

```
wstidier2$IDSoil <- as.character(wstidier2$IDSoil)
split <- strsplit(wstidier2$IDSoil, "-", fixed = TRUE)
wstidier2$Soil.ID <- sapply(split, "[", 3)
wstidier2$Soil.ID <- as.factor(wstidier2$Soil.ID)

ggplot(data = wstidier2, aes(x = Date, y = measure, group = Source) )+
  theme_bw() +
  #geom_errorbar(data=subset(wstidier2, Type == 'Top Soil'), limits_DdC, size=0.2) +

  #geom_errorbar(data=subset(wstidier2, Source == 'Valley' &
  #                            Date > as.POSIXct('2016-05-14 08:04:00')), limits_DdC, size=0.2) +
  #geom_errorbar(data=subset(wstidier2, Source == 'Outlet'), limits_DdC) +
  #geom_point(data=subset(wstidier2, Source == 'Outlet'), aes(shape = Type, colour = Source, size = Qma
  geom_point(data=subset(wstidier2, Type == 'Dissolved (Outlet)'), aes(shape = Type, colour = Source, s
  geom_point(data=subset(wstidier2, Type == 'Top Soil'), aes(colour = Source)) +

  stat_smooth(data=subset(wstidier2,
                          (Source == "North" )), #| Source == "South" )),
              method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = Source), alpha = 0.2, size=
  stat_smooth(data=subset(wstidier2,
                          (Source == "Valley" )), #| Source == "South" )),
              method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = Source), alpha = 0.2, size=
  stat_smooth(data=subset(wstidier2,
                          (Source == "South" )), #| Source == "South" )),
              method = "lm", formula = y ~ poly(x, 2), se = F, aes(colour = Source), alpha = 0.2, size=
  geom_point(data=subset(wstidier2, Source == 'Bulk'), aes(shape = Type, colour = Source)) +
  #geom_point(data=subset(wstidier2, Source == 'Valley' &
  #                         Date > as.POSIXct('2016-05-14 08:04:00')), aes(shape = Type, colour = Sourc
  #geom_text_repel(data=subset(wstidier2, Source == 'Bulk'), aes(label=Soil.ID),
  #                arrow = arrow(length = unit(0.005, 'npc'), type = "closed"),
  #                force = 1,
  #                point.padding = unit(1.0, 'lines'),
  #                max.iter = 2e3,
  #               nudge_x = .2) +
  #geom_text_repel(data=subset(wstidier2, Source != 'Outlet'), aes(label=Soil.ID),
  geom_text_repel(data=subset(wstidier2, Source == 'Outlet'), aes(label=Week),
                  arrow = arrow(length = unit(0.005, 'npc'), type = "closed"),
                  force = 1,
                  point.padding = unit(1.0, 'lines'),
                  max.iter = 2e3,
                  nudge_x = .2)
```




```
sum(wstidier2$Location == "North")  # 10
```

```
## [1] 9
```

```
sum(wstidier2$Location == "South")  # 12
```

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

```
sum(wstidier2$Location == "Talweg")  # 12
```

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

```
sum(wstidier2$Source == "Bulk")  # 9
```

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

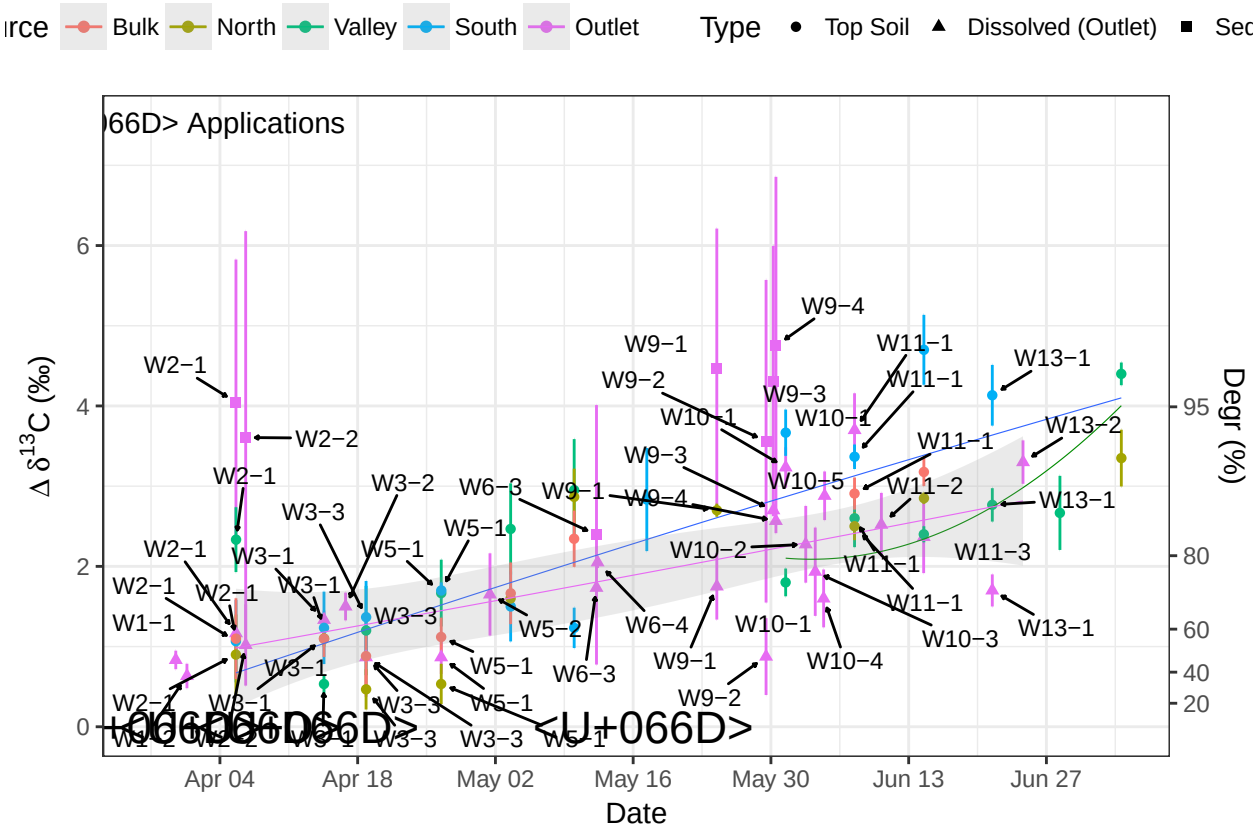
```
write.csv2(wstidier2,
           'Data/OutletData4Lutz_R.csv', row.names = F)
```

Soils and Water with labels (inspection)

```
# Data without the Plateau
#wsNoPlat <- subset(wstidierAll, Source != "Plateau")
wsNoPlat <- subset(wstidier, SD < 4)
#wsNoPlat$Source <- factor(wsNoPlat$Source, levels = c("Bulk", "Valley", "Outlet"))
#levels(wsNoPlat$Source)
```

```
# Subset the data to values with SD < 1
#wsNoPlat2 = subset(wsNoPlat, SD < 1.50)


limits_DdC <- aes(ymin=measure-SD, ymax=measure+SD, colour = Source)

wsPlot <- ggplot(data = wsNoPlat, aes(x = Date, y = measure)) +
  geom_errorbar(limits_DdC) +
  geom_jitter(aes(shape = Type, colour = Source)) +
  stat_smooth(data=subset(wsNoPlat,
                          (Source == "Valley" & Event > 8 )),
              method = "lm", formula = y ~ poly(x, 2), se = F, colour = 'green4' , alpha = 0.1, size=0.1
  stat_smooth(data=subset(wsNoPlat,
                          (Source != "Outlet" & Source != "Valley" & Event < 20 )),
              method = "lm", formula = y ~  poly(x, 2), se = F, alpha = 0.1, size=0.2) +
  stat_smooth(data=subset(wsNoPlat,
                          (Source == "Outlet" & Event > 1 & Type == "Dissolved (Outlet)")),
              method = "lm", formula = y ~ poly(x, 2), se = T, aes(colour = 'Outlet'), alpha = 0.2, size
  #stat_smooth(data=subset(wsNoPlat,
  #                         (Source == "Bulk")),
  #             method = "lm", formula = y ~ poly(x, 2), se = T, aes(colour = 'Bulk'), alpha = 0.2, size=
  theme_bw() +
  # Applics
  annotate("text", x = as.POSIXct('2016-03-28 08:04:00'), y = 0,
           label = as.character(expression(paste( "\u066D"))), parse = T, size = 6.0) +
  annotate("text", x = as.POSIXct('2016-04-05 08:04:00'), y = 0,
           label = as.character(expression(paste( "\u066D"))), parse = T, size = 6.0) +
  annotate("text", x = as.POSIXct('2016-04-13 08:04:00'), y = 0,
           label = as.character(expression(paste( "\u066D"))), parse = T, size = 6.0) +
  annotate("text", x = as.POSIXct('2016-05-17 08:04:00'), y = 0,
           label = as.character(expression(paste( "\u066D"))), parse = T, size = 6.0) +
  # Title applics
  annotate("text", x = as.POSIXct('2016-04-01 08:04:00'), y = 7.5,
           label = as.character(expression(paste( "\u066D", " Applications"))), parse = T, size = 4.0) +

  scale_x_datetime(breaks = date_breaks("2 weeks"), labels = date_format("%b %d")) +
  theme(legend.position="top"
        # axis.title.x = element_blank(),
        # axis.text.x=element_text(angle = 45, hjust = 1)
        ) +
  # geom_smooth(data=subset(ws, Source != "Outlet"), method = "lm", formula = y ~ poly(x, 2)) +
  xlab("Date") +
  #ylab(expression(paste({Delta~delta}^"13","C", ' (\u2030)'))) +
  scale_y_continuous(
    expression(paste({Delta~delta}^"13","C", ' (\u2030)')),
    sec.axis = sec_axis(trans = ~ (1-((1000 + d13Co + .)/(1000+d13Co))^(1000/epsilon_field))*100 ,
                        name = "Degr (%)", breaks=c(20, 40, 60, 80, 95) )# breaks=seq(20, 120, 15))
  ) +
  geom_text_repel(aes(label=as.factor(Week)),
                  size = 3,
                   arrow = arrow(length = unit(0.005, 'npc'), type = "closed"),
                   force = 0.5,
                   point.padding = unit(0.5, 'lines'),
```

```
                 max.iter = 2e3,
                nudge_x = .05)

# +
  #scale_color_manual(name= "Source",
  #                    values = c("black", "dodgerblue", "green", "red")
  #                   ) +
  # scale_shape_manual(name= )

wsPlot
```



Testing difference in $\Delta\delta$ between groups

Based on ANOVA tests, there is:

- No significant difference between soils and water

```
names(WaterSoils)
```

```
##   [1] "Date.ti"                "WeekSubWeek"
##   [3] "tf"                     "iflux"
##   [5] "fflux"                  "changeflux"
##   [7] "maxQ"                   "minQ"
##   [9] "dryHrsIni"              "dryHrsMax"
##  [11] "dryHrsAve"              "noEventHrsIni"
##  [13] "noEventHrsMax"          "noEventHrsAve"
```

```
##  [15] "Duration.Hrs.x"         "chExtreme"
##  [17] "Peak"                   "Markers"
##  [19] "TimeDiff"               "AveDischarge.m3.h"
##  [21] "Volume.m3"              "Sampled.Hrs"
##  [23] "Sampled"                "CumRain.mm"
##  [25] "RainInt.mmhr"           "Conc.mug.L"
##  [27] "Conc.SD"                "Vol.SPE.L"
##  [29] "Conc.in500uL"           "OXA_mean"
##  [31] "OXA_SD"                 "ESA_mean"
##  [33] "ESA_SD"                 "N.x"
##  [35] "diss.d13C.x"            "SD.d13C.x"
##  [37] "N_d13C.diss"            "MES.mg.L"
##  [39] "MES.sd"                 "MO.mg.L"
##  [41] "Conc.Solids.mug.gMES"   "Conc.Solids.ug.gMES.SD"
##  [43] "N.y"                    "filt.d13C"
##  [45] "filt.SD.d13C"           "DD13C.diss"
##  [47] "DD13C.filt"             "ExpMES.Kg"
##  [49] "DissSmeto.mg"           "DissSmeto.mg.SD"
##  [51] "DissSmeto.g"            "DissSmeto.g.SD"
##  [53] "DissOXA.mg"             "DissOXA.mg.SD"
##  [55] "DissOXA.g"              "DissOXA.g.SD"
##  [57] "DissESA.mg"             "DissESA.mg.SD"
##  [59] "DissESA.g"              "DissESA.g.SD"
##  [61] "FiltSmeto.mg"           "FiltSmeto.mg.SD"
##  [63] "FiltSmeto.g"            "FiltSmeto.g.SD"
##  [65] "TotSMout.mg"            "TotSMout.mg.SD"
##  [67] "TotSMout.g.x"           "TotSMout.g.SD.x"
##  [69] "FracDiss"               "FracFilt"
##  [71] "MELsm.g.x"              "MELsm.g.SD.x"
##  [73] "CumOutDiss.g"           "CumOutFilt.g"
##  [75] "CumOutSmeto.g.x"        "CumOutMELsm.g"
##  [77] "Appl.Mass.g.x"          "Appl.Mass.g.OT.x"
##  [79] "Appl.Mass.g.N"          "Appl.Mass.g.T"
##  [81] "Appl.Mass.g.S"          "Appl.Mass.g.N.OT"
##  [83] "Appl.Mass.g.T.OT"       "Appl.Mass.g.S.OT"
##  [85] "iniCo.ug.g.N.x"         "iniCo.ug.g.T.x"
##  [87] "iniCo.ug.g.S.x"         "timeSinceApp.x"
##  [89] "timeSinceApp.N.x"       "timeSinceApp.T.x"
##  [91] "timeSinceApp.S.x"       "Appl.Mass.g.NoSo"
##  [93] "timeSinceApp.NoSo.x"    "CumAppMass.g.x"
##  [95] "CumAppMass.g.OT.x"      "CumAppMass.g.N.x"
##  [97] "CumAppMass.g.T.x"       "CumAppMass.g.S.x"
##  [99] "CumAppMass.g.N.OT.x"    "CumAppMass.g.T.OT.x"
## [101] "CumAppMass.g.S.OT.x"    "BalMassDisch.g"
## [103] "prctMassOut"            "FracDeltaOut"
## [105] "Events"                 "Weeks"
## [107] "Event.x"                "yRaleigh.x"
## [109] "xRaleigh.x"             "DIa"
## [111] "Event.y"                "Duration.Hrs.y"
## [113] "timeSinceApp.y"         "timeSinceApp.NoSo.y"
## [115] "timeSinceApp.N.y"       "timeSinceApp.T.y"
## [117] "timeSinceApp.S.y"       "diss.d13C.y"
## [119] "SD.d13C.y"              "TotSMout.g.y"
## [121] "TotSMout.g.SD.y"        "MELsm.g.y"
```

```
## [123] "MELsm.g.SD.y"            "Appl.Mass.g.y"
## [125] "Appl.Mass.g.OT.y"        "CumAppMass.g.y"
## [127] "CumAppMass.g.N.y"        "CumAppMass.g.T.y"
## [129] "CumAppMass.g.S.y"        "CumAppMass.g.OT.y"
## [131] "CumAppMass.g.N.OT.y"     "CumAppMass.g.T.OT.y"
## [133] "CumAppMass.g.S.OT.y"     "iniCo.ug.g.N.y"
## [135] "iniCo.ug.g.T.y"          "iniCo.ug.g.S.y"
## [137] "CumOutSmeto.g.y"         "MassSoil.g.North"
## [139] "MassSoil.g.SD.North"     "Conc.mug.g.dry.soil.N"
## [141] "comp.d13C.North"         "comp.d13C.SD.North"
## [143] "ID.N"                    "MassSoil.g.Talweg"
## [145] "MassSoil.g.SD.Talweg"    "Conc.mug.g.dry.soil.T"
## [147] "comp.d13C.Talweg"        "comp.d13C.SD.Talweg"
## [149] "MassSoil.g.South"        "MassSoil.g.SD.South"
## [151] "Conc.mug.g.dry.soil.S"   "comp.d13C.South"
## [153] "comp.d13C.SD.South"      "ID.S"
## [155] "DD13C.North"             "DD13C.Talweg"
## [157] "DD13C.South"             "CatchMassSoil.g"
## [159] "CatchMassSoil.g.SD"      "BulkCatch.d13"
## [161] "BulkCatch.d13.SD"        "DD13.Bulk"
## [163] "Area.Catchment"          "BulkCatch.Conc"
## [165] "iniCo.Bulk"              "yRaleigh.y"
## [167] "xRaleigh.y"
```

```
# ISotope values to test
keepDDtest <- c(
  "Date.ti",
  "diss.d13C.x", # "DD13C.diss",
  "comp.d13C.North", "comp.d13C.Talweg", "comp.d13C.South" #,
  #"DD13C.North", "DD13C.Talweg", "DD13C.South"
)

# Concnetrations to plot as box plot
keepConctest <- c( "Date.ti",
                   "Conc.mug.L" ,
                   "Conc.mug.g.dry.soil.S" , "Conc.mug.g.dry.soil.T", "Conc.mug.g.dry.soil.N")

wsStatTest <- WaterSoils[, colnames(WaterSoils) %in% keepDDtest]
wsStatTestConc <- WaterSoils[, colnames(WaterSoils) %in% keepConctest]

mwsStatTest <- melt(wsStatTest, id="Date.ti")
mwsStatTestConc <- melt(wsStatTestConc, id="Date.ti")

mwsStatTest$Group1 <- ifelse(mwsStatTest$variable == "diss.d13C.x", "Outlet", "Soil")
mwsStatTest$Group2 <- ifelse(mwsStatTest$variable == "diss.d13C.x", "Outlet",
                             ifelse(mwsStatTest$variable == "comp.d13C.Talweg", "Valley", "Plateau"))
mwsStatTest$Group3 <- ifelse(mwsStatTest$variable == "diss.d13C.x" &
                               mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST'), "Ou
                       ifelse(mwsStatTest$variable == "diss.d13C.x" &
                               mwsStatTest$Date.ti <= as.POSIXct('2016-06-05 00:06:00', tz = 'EST'), "O
                       ifelse(mwsStatTest$variable == "comp.d13C.Talweg" &
                               mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST'), "Va
                       ifelse(mwsStatTest$variable == "comp.d13C.Talweg" &
                               mwsStatTest$Date.ti <= as.POSIXct('2016-06-05 00:06:00', tz = 'EST'), "V
```

```
                        ifelse( (mwsStatTest$variable == "comp.d13C.North" | mwsStatTest$variable == "comp
                                 mwsStatTest$Date.ti <= as.POSIXct('2016-06-05 00:06:00', tz = 'EST'), "Pl
                        ifelse( (mwsStatTest$variable == "comp.d13C.North" | mwsStatTest$variable == "comp
                                 mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST'), "Pla
                               ))))))

Gr1 <- na.omit(mwsStatTest[, colnames(mwsStatTest) %in% c("value", "Group1")])
Gr2 <- na.omit(mwsStatTest[, colnames(mwsStatTest) %in% c("value", "Group2")])
Gr3 <- na.omit(mwsStatTest[, colnames(mwsStatTest) %in% c("value", "Group3")])


# Test for homogeneity of variance
# Large p-value means no confirmation of homogeneity of variance
bartlett.test(value ~ as.factor(Group3), data = Gr3)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  value by as.factor(Group3)
## Bartlett's K-squared = 0.77938, df = 5, p-value = 0.9783
```

```
# Non-parameteric
# Reject Ho that pop. means are the same if low p-value
res.krs.Grp3 <- kruskal.test(value ~ as.factor(Group3), data = Gr3)
res.krs.Grp3
```

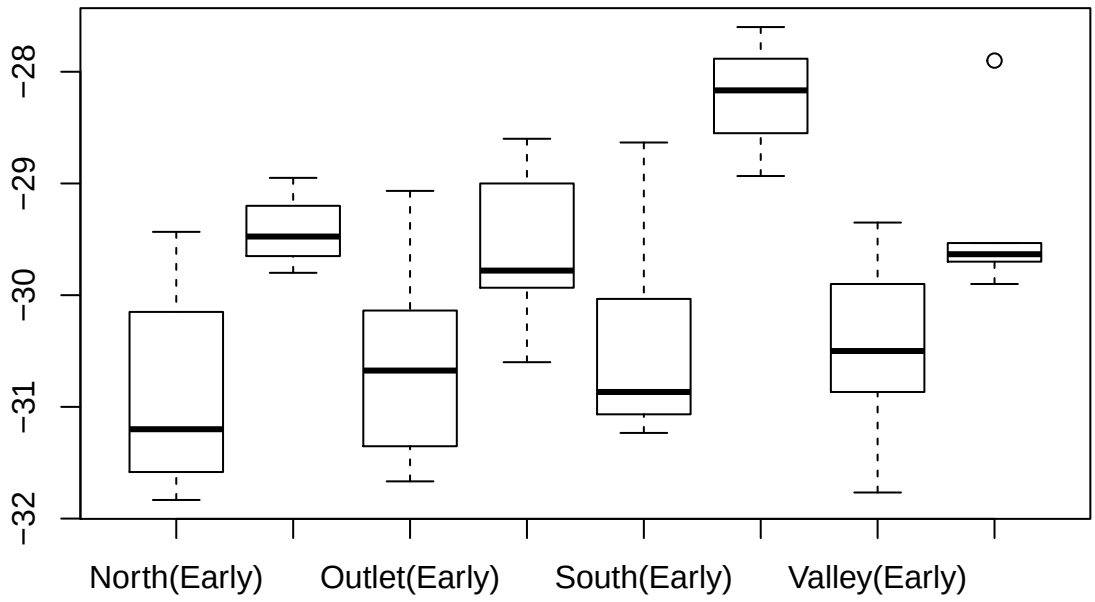
```
##
##  Kruskal-Wallis rank sum test
##
## data:  value by as.factor(Group3)
## Kruskal-Wallis chi-squared = 20.986, df = 5, p-value = 0.0008151
```

```
# Want a TukeyHSD function, but this only works with
# parametric data. So, will pass the ranks of the data instead of the actual values
Gr3.ranks <- rank( Gr3$value )
Gr3.groups <- Gr3$Group3
group3.aov <- aov(Gr3.ranks ~ Gr3.groups)
res.grp3 <- TukeyHSD(group3.aov, ordered = T)
aov.res.grp3.df <- as.data.frame(res.grp3$Gr3.groups)
aov.res.grp3.df$P <- round(aov.res.grp3.df$`p adj`, 3)
# High p-value indicates no significant difference
write.csv(aov.res.grp3.df, "aovResISOs_ranked.csv", row.names = T)
```

Box plot outlet vs soils ()

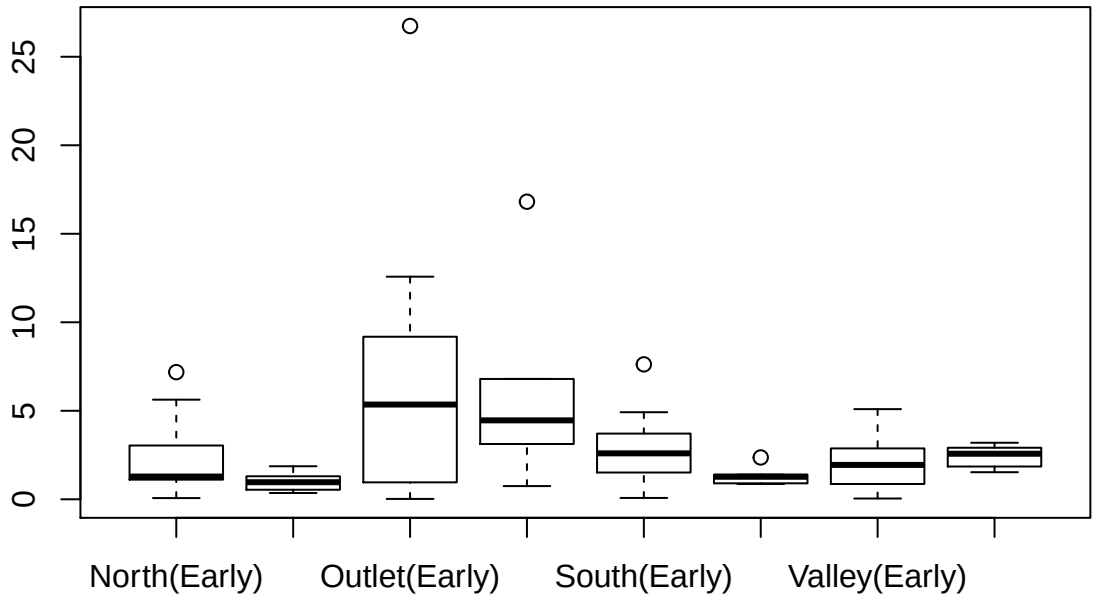
```
mwsStatTest$Group4 <- ifelse(mwsStatTest$variable == "diss.d13C.x" & # Outlet
                               mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST'),
                             ifelse(mwsStatTest$variable == "diss.d13C.x" &
                               mwsStatTest$Date.ti <= as.POSIXct('2016-06-05 00:06:00', tz = 'EST')
                             # South
                             ifelse(mwsStatTest$variable == "comp.d13C.South" &
                               mwsStatTest$Date.ti < as.POSIXct('2016-06-05 00:06:00', tz = 'EST'),
                               ifelse(mwsStatTest$variable == "comp.d13C.South" &
```

```
                                    mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST')
                                # Valley
                                    ifelse(mwsStatTest$variable == "comp.d13C.Talweg" &
                                   mwsStatTest$Date.ti < as.POSIXct('2016-06-05 00:06:00', tz = 'EST'),
                                   ifelse(mwsStatTest$variable == "comp.d13C.Talweg" &
                                    mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST')
                                 # North
                                    ifelse(mwsStatTest$variable == "comp.d13C.North" &
                                   mwsStatTest$Date.ti < as.POSIXct('2016-06-05 00:06:00', tz = 'EST'),
                                   ifelse(mwsStatTest$variable == "comp.d13C.North" &
                                    mwsStatTest$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST')
                                  ))))))))


boxplot(mwsStatTest$value ~ mwsStatTest$Group4)
```




```
mwsStatTestConc$Group4 <- ifelse(mwsStatTestConc$variable == "Conc.mug.L" & # Outlet
                                   mwsStatTestConc$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = 'EST
                                 ifelse(mwsStatTestConc$variable == "Conc.mug.L" &
                                   mwsStatTestConc$Date.ti <= as.POSIXct('2016-06-05 00:06:00', tz = 'ES
                                 # South
                                 ifelse(mwsStatTestConc$variable == "Conc.mug.g.dry.soil.S" &
                                   mwsStatTestConc$Date.ti < as.POSIXct('2016-06-05 00:06:00', tz = 'EST
                                   ifelse(mwsStatTestConc$variable == "Conc.mug.g.dry.soil.S" &
                                     mwsStatTestConc$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = '
```

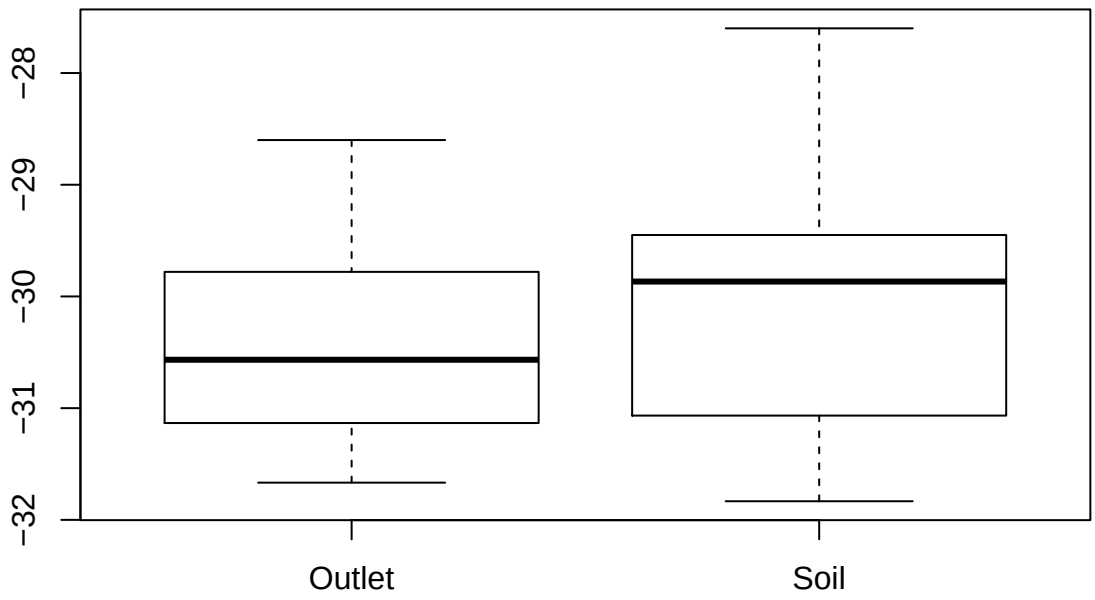
```
                                # Valley
                                    ifelse(mwsStatTestConc$variable == "Conc.mug.g.dry.soil.T" &
                                  mwsStatTestConc$Date.ti < as.POSIXct('2016-06-05 00:06:00', tz = 'EST
                                  ifelse(mwsStatTestConc$variable == "Conc.mug.g.dry.soil.T" &
                                    mwsStatTestConc$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = '
                                 # North
                                    ifelse(mwsStatTestConc$variable == "Conc.mug.g.dry.soil.N" &
                                  mwsStatTestConc$Date.ti < as.POSIXct('2016-06-05 00:06:00', tz = 'EST
                                  ifelse(mwsStatTestConc$variable == "Conc.mug.g.dry.soil.N" &
                                    mwsStatTestConc$Date.ti > as.POSIXct('2016-06-05 00:06:00', tz = '
                                 ))))))))
boxplot(mwsStatTestConc$value ~ mwsStatTestConc$Group4)
```



ANOVA and ANOSIM

Not actually used, as Grouping 3 does not have homegenity of variance

```
# Simple ANOVA tests
# (high p-value indicates lack of difference)
# Big P-value no significant difference
boxplot(Gr1$value ~ Gr1$Group1)
```



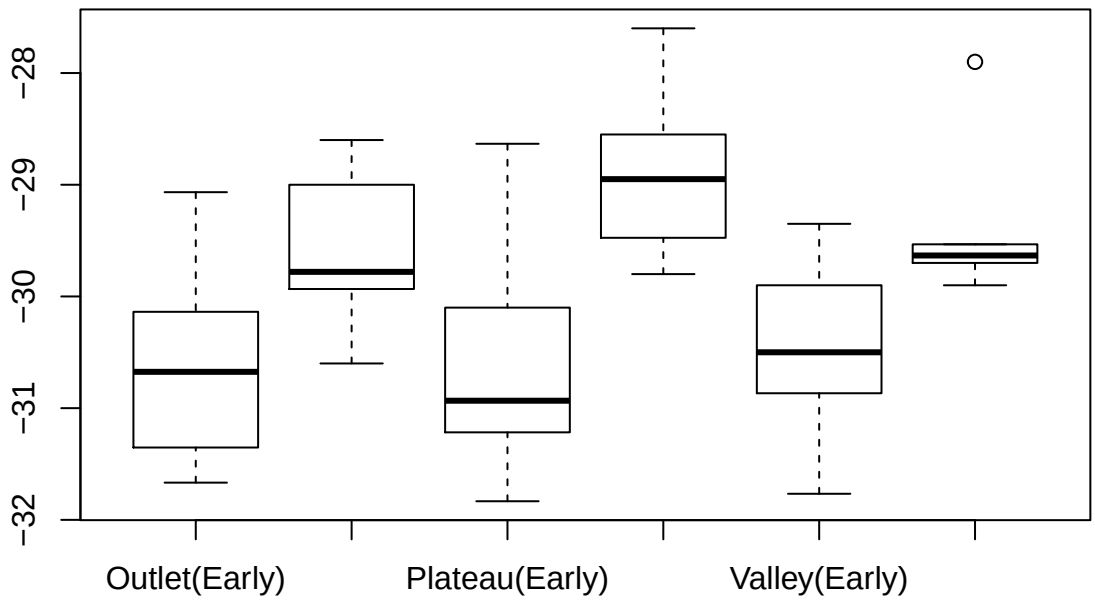

```
summary(aov(Gr1$value ~ Gr1$Group1))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Gr1$Group1   1   1.87   1.873   1.823  0.182
## Residuals   57  58.56   1.027
```

```
TukeyHSD(aov(Gr1$value ~ Gr1$Group1))
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Gr1$value ~ Gr1$Group1)
##
## $`Gr1$Group1`
##                  diff        lwr       upr     p adj
## Soil-Outlet 0.3605137 -0.1742139 0.8952413 0.1823324
```

```
boxplot(Gr3$value ~ Gr3$Group3)
```




```
group3.aov <- aov(Gr3$value ~ Gr3$Group3)
summary(group3.aov)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Gr3$Group3   5  24.33   4.865   7.143 3.59e-05 ***
## Residuals   53  36.10   0.681
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Transform data and compute dissimilarity
Gr1.hell <- decostand(Gr1[ , 1], "hellinger", na.rm=T, MARGIN = 2) # Transform/Standardize
Gr1.hell.daisy = daisy(Gr1.hell, "euclidean") # Dissimilarity
attach(Gr1)
anosim.group1 <- anosim(Gr1.hell.daisy, grouping = Group1)
summary(anosim.group1)
```

```
##
## Call:
## anosim(dat = Gr1.hell.daisy, grouping = Group1)
## Dissimilarity:
##
## ANOSIM statistic R: -0.01812
##       Significance: 0.696
##
## Permutation: free
## Number of permutations: 999
##
```

```
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0378 0.0590 0.0748 0.1116
##
## Dissimilarity ranks between and within classes:
##         0%    25%   50%      75% 100%   N
## Between  4 420.25 835.5 1282.875 1708 850
## Outlet   4 401.25 742.5 1154.500 1652 300
## Soil     4 467.00 967.5 1354.500 1711 561
```

```
Gr2.hell <- decostand(Gr2[ , 1], "hellinger", na.rm=T, MARGIN = 2) # Transform/Standardize
Gr2.hell.daisy = daisy(Gr2.hell, "euclidean") # Dissimilarity
attach(Gr2)
anosim.group2 <- anosim(Gr2.hell.daisy, grouping = Group2)
summary(anosim.group2)
```

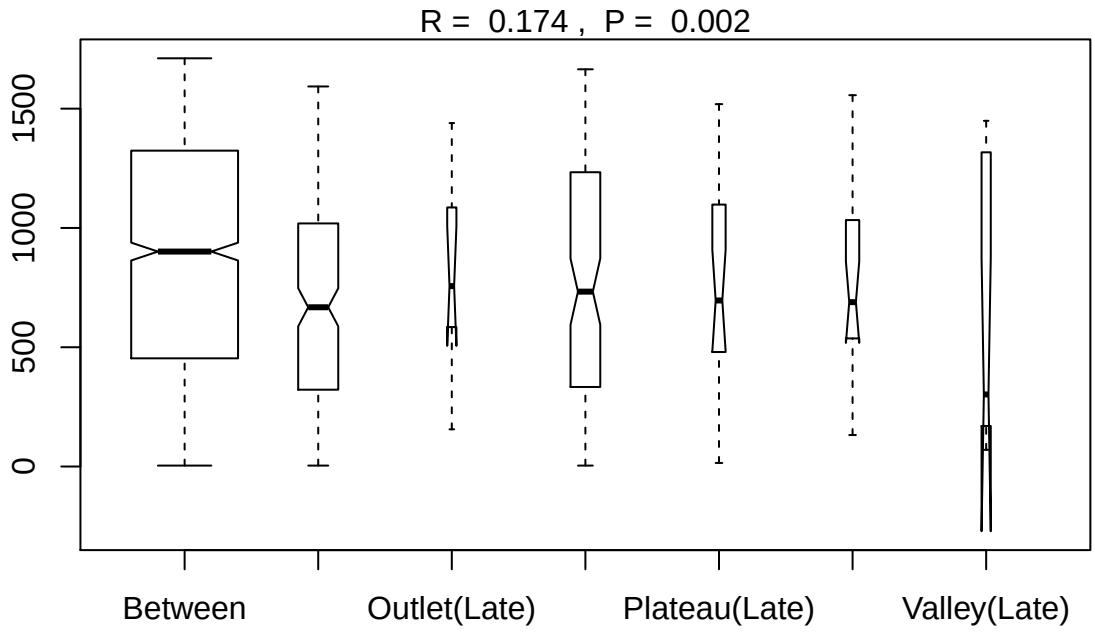
```
##
## Call:
## anosim(dat = Gr2.hell.daisy, grouping = Group2)
## Dissimilarity:
##
## ANOSIM statistic R: 0.008077
##       Significance: 0.335
##
## Permutation: free
## Number of permutations: 999
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0403 0.0631 0.0778 0.0989
##
## Dissimilarity ranks between and within classes:
##         0%    25%    50%      75%   100%    N
## Between  4 430.25  865.5 1283.125 1709.5 1114
## Outlet   4 401.25  742.5 1154.500 1652.0  300
## Plateau  4 480.75 1064.0 1444.750 1711.0  231
## Valley  67 335.50  778.0 1296.500 1704.5   66
```

```
Gr3.hell <- decostand(Gr3[ , 1], "hellinger", na.rm=T, MARGIN = 2) # Transform/Standardize
Gr3.hell.daisy = daisy(Gr3.hell, "euclidean") # Dissimilarity
attach(Gr3)
anosim.group3 <- anosim(Gr3.hell.daisy, grouping = Group3)
summary(anosim.group3)
```

```
##
## Call:
## anosim(dat = Gr3.hell.daisy, grouping = Group3)
## Dissimilarity:
##
## ANOSIM statistic R: 0.1736
##       Significance: 0.002
##
## Permutation: free
## Number of permutations: 999
##
```

```
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0623 0.0823 0.0988 0.1168
##
## Dissimilarity ranks between and within classes:
##                  0%     25%    50%      75%   100%    N
## Between           4 453.500 901.00 1323.625 1711.0 1354
## Outlet(Early)     4 325.500 667.50 1018.500 1593.0  190
## Outlet(Late)    156 609.625 756.25 1063.750 1439.5   10
## Plateau(Early)    4 333.500 733.00 1233.500 1665.0  105
## Plateau(Late)    15 480.000 696.00 1098.000 1519.0   21
## Valley(Early)   132 537.000 689.00 1033.000 1556.5   21
## Valley(Late)     70 177.250 302.00 1304.750 1449.0   10
```

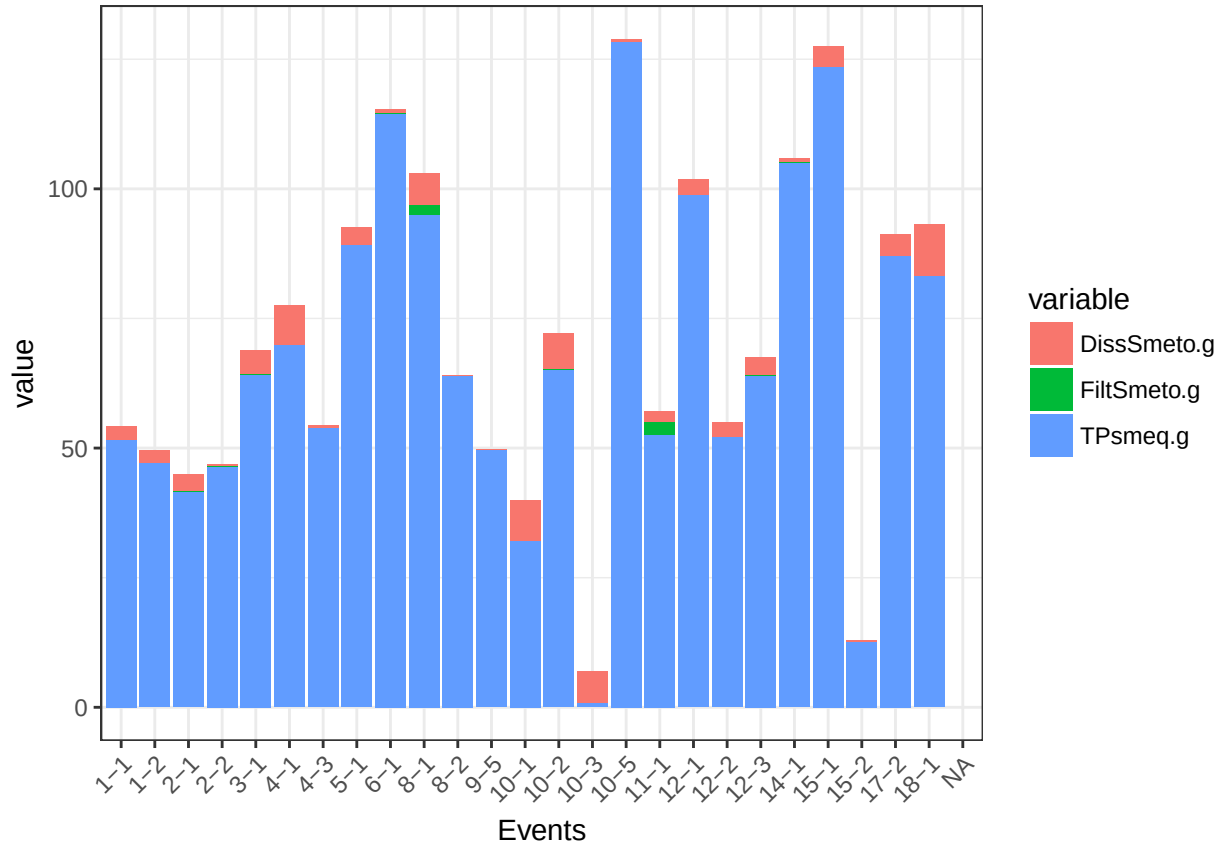
```
plot(anosim.group3)
```



Loadings

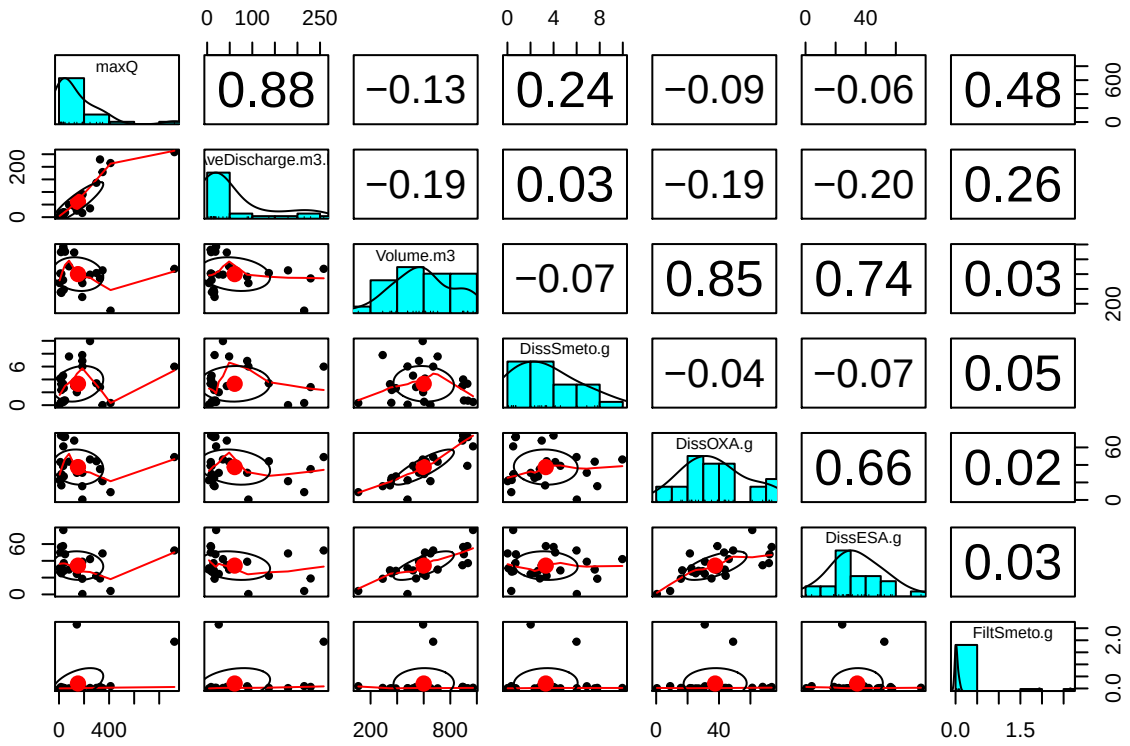
```
keepLoads <- c("Date.ti",
               "DissOXA.g", "DissESA.g", "DissSmeto.g", "FiltSmeto.g",
               "Event.x", "Events")
wsLoads <- WaterSoils[ , (names(WaterSoils) %in% keepLoads)]

mw.SM <- 283.796 # g/mol
mw.MOXA <- 279.33 # g/ml
```

```
mw.MESA <- 329.1 # g/mol
wsLoads$TPsmeq.g <-
  wsLoads$DissOXA.g * (mw.SM/mw.MOXA) +
  wsLoads$DissESA.g * (mw.SM/mw.MESA)

wsLoads <- wsLoads[ , !(names(wsLoads) %in% c("DissOXA.g", "DissESA.g"))]

loads <- melt(wsLoads, id=c("Date.ti", "Events", "Event.x"))

ggplot(data = loads , aes(x=Events, y=value, fill = variable))+
  theme_bw() +
  geom_bar(stat = "identity") +
  theme(# legend.position="top"
        # axis.title.x = element_blank(),
        axis.text.x=element_text(angle = 45, hjust = 1)
        )
```

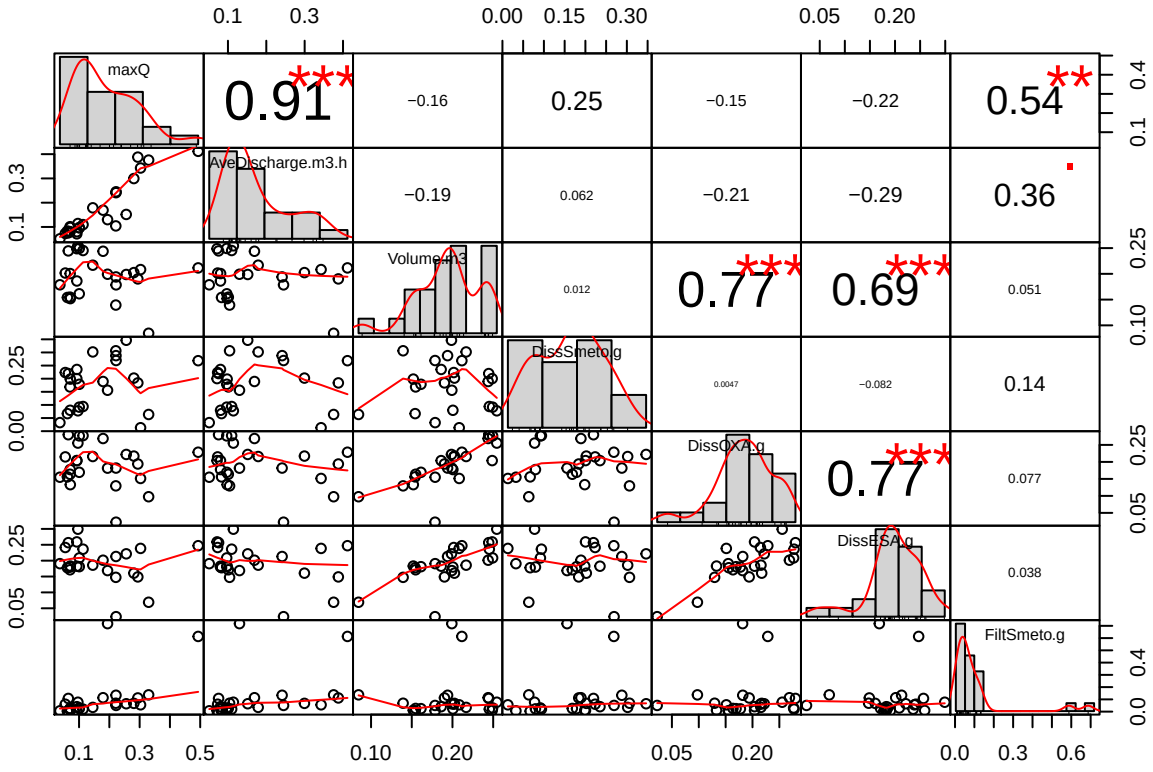



```
  # geom_bar(stat = "identity", position = position_dodge())

# WaterSoils$DIE <- WaterSoils$maxQ*WaterSoils$Volume.m3/WaterSoils$Duration.Hrs

keepCor <- c("maxQ", "Duration.Hrs", "AveDischarge.m3.h", "Volume.m3", # "DIE",
             "DissOXA.g", "DissESA.g", "DissSmeto.g", "FiltSmeto.g" #,
             #"NH4.mM", "TIC.ppm.filt", "Cl.mM", "NO3..mM", "PO4..mM", "NPOC.ppm",
             #"TIC.ppm.unfilt", "TOC.ppm.unfilt"
             )
```

```
corData <- WaterSoils[ , (names(WaterSoils) %in% keepCor)]

# Transform / normalize
corData.hell <- decostand(corData, "hellinger", na.rm=T, MARGIN = 2)

library(psych)
pairs.panels(corData)
```




```
library(PerformanceAnalytics)
chart.Correlation(corData.hell)
```




```
keepLoads <- c("Date.ti",
               "DissOXA.g", "DissESA.g", "DissSmeto.g", "FiltSmeto.g",
               "Event.x", "Events")
wsLoads <- WaterSoils[ , (names(WaterSoils) %in% keepLoads)]
```

Outliers

```
# Test function
g_param = 1.5
# g_param = 2.2  #  (Hoaglin et al.,1986; Hoaglin & Iglewicz, 1987)
is_outlier <- function(x) {
  return(x < quantile(x, 0.25) - g_param * IQR(x) | x > quantile(x, 0.75) + g_param * IQR(x))
}
```

Soil concentrations

Correlation will be made after variable transformation. Options tested:

a) Z-scoring transformation by translation and expansion is done to create unit-free variables with means of zero and standard deviations of one. Standardised values differ from one another in units of standard deviation. The mean of each variable is subtracted from the original values and the difference divided by the variable's standard deviation and is given by:

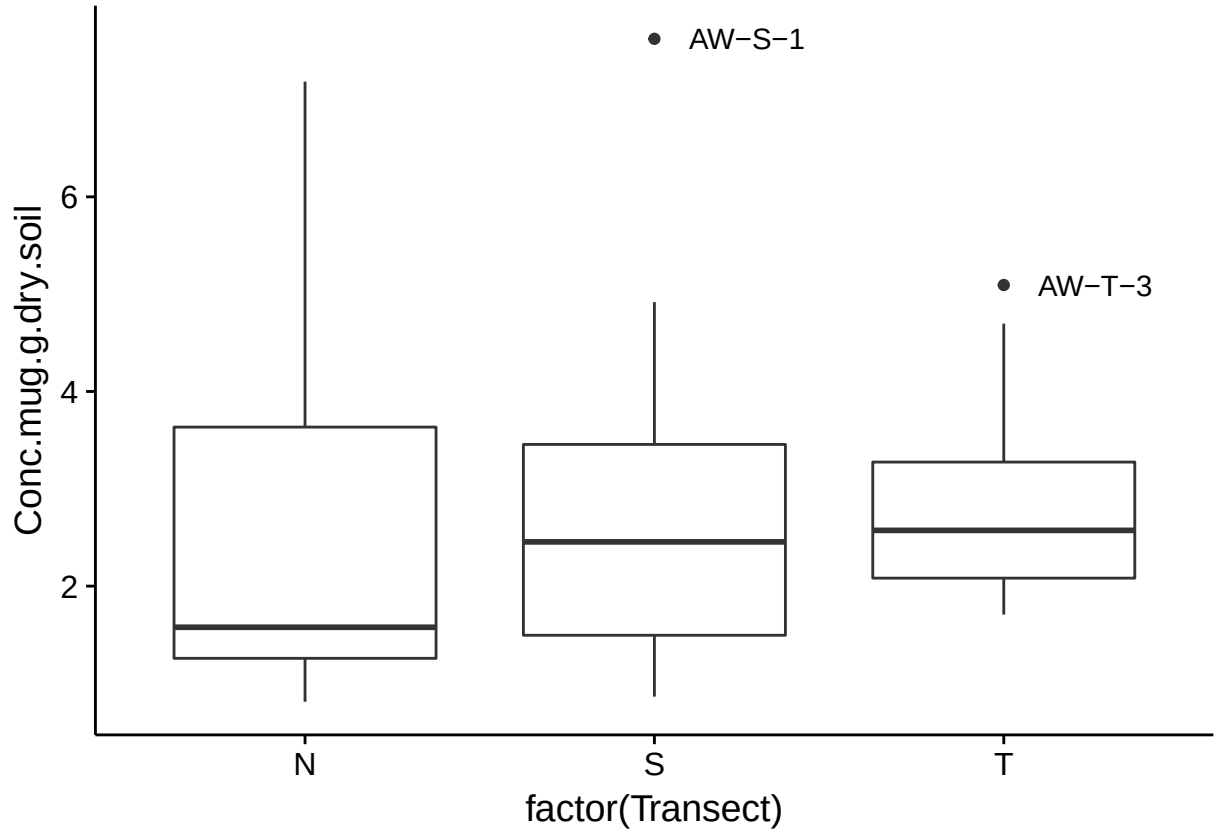
$$z_i = \frac{y_i - \bar{y}}{s_y}$$

Z-scoring did not change correlation results, nor outlier reduction.

b) Scaling by expansion where all values are divided by the maximum observation.

Outliers before transformation

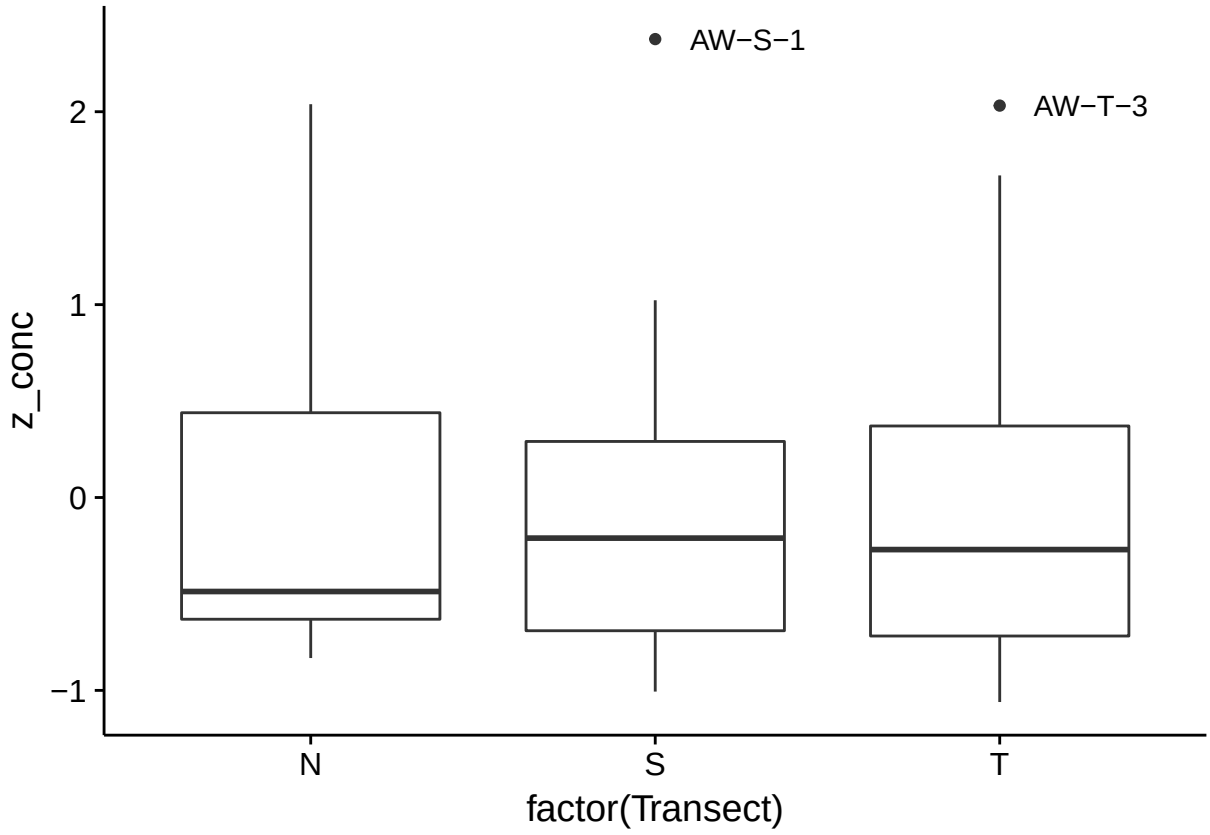
```
# Concentrations
soilGroups %>%
  group_by(Transect) %>%
  mutate(outlier = ifelse(is_outlier(Conc.mug.g.dry.soil), as.character(ID), NA)) %>%
  ggplot(., aes(x = factor(Transect), y = Conc.mug.g.dry.soil)) +
    geom_boxplot() +
    geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3)
```



Outliers after transformation

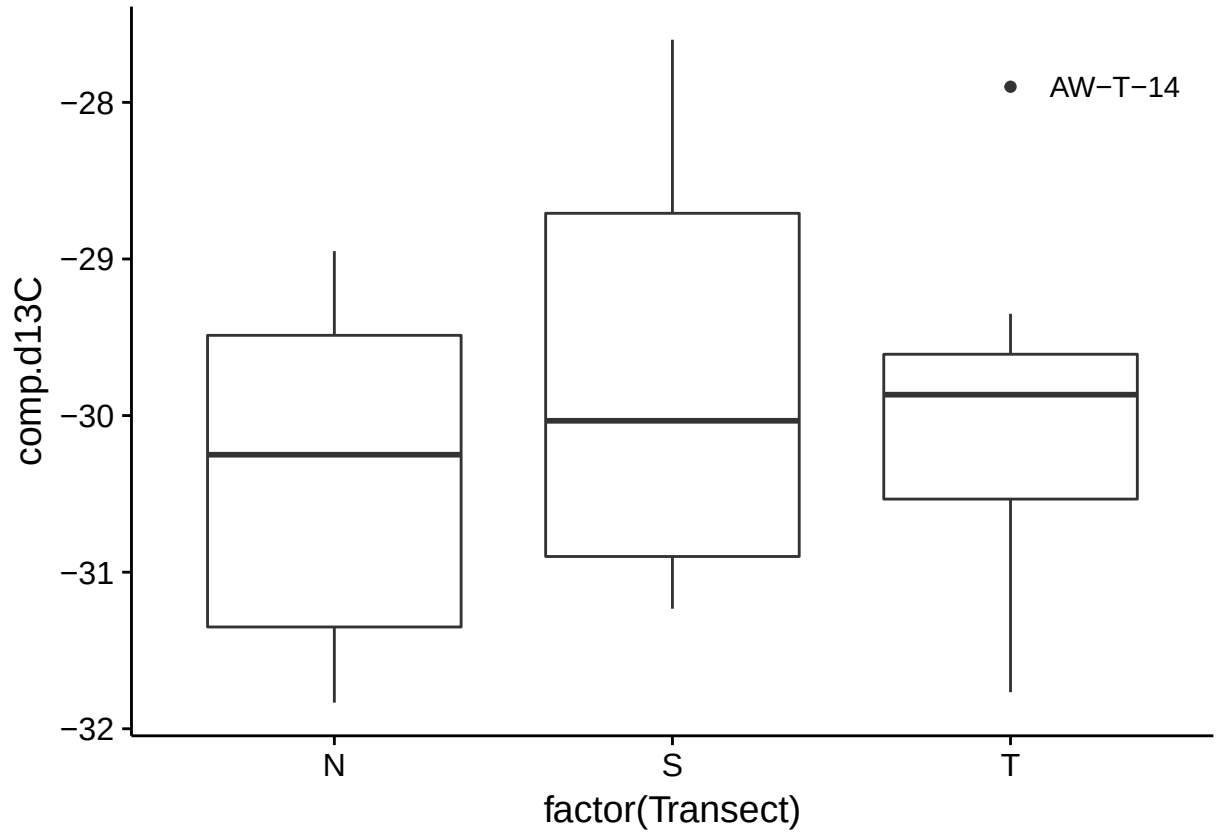
```
soilGroups <- soilGroups %>%
  group_by(Transect) %>%
  mutate(z_conc = (Conc.mug.g.dry.soil-mean(Conc.mug.g.dry.soil))/sd(Conc.mug.g.dry.soil))
```

```
soilGroups %>%
  group_by(Transect) %>%
  mutate(outlier = ifelse(is_outlier(z_conc), as.character(ID), NA)) %>%
  ggplot(., aes(x = factor(Transect), y = z_conc)) +
    geom_boxplot() +
    geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3)
```



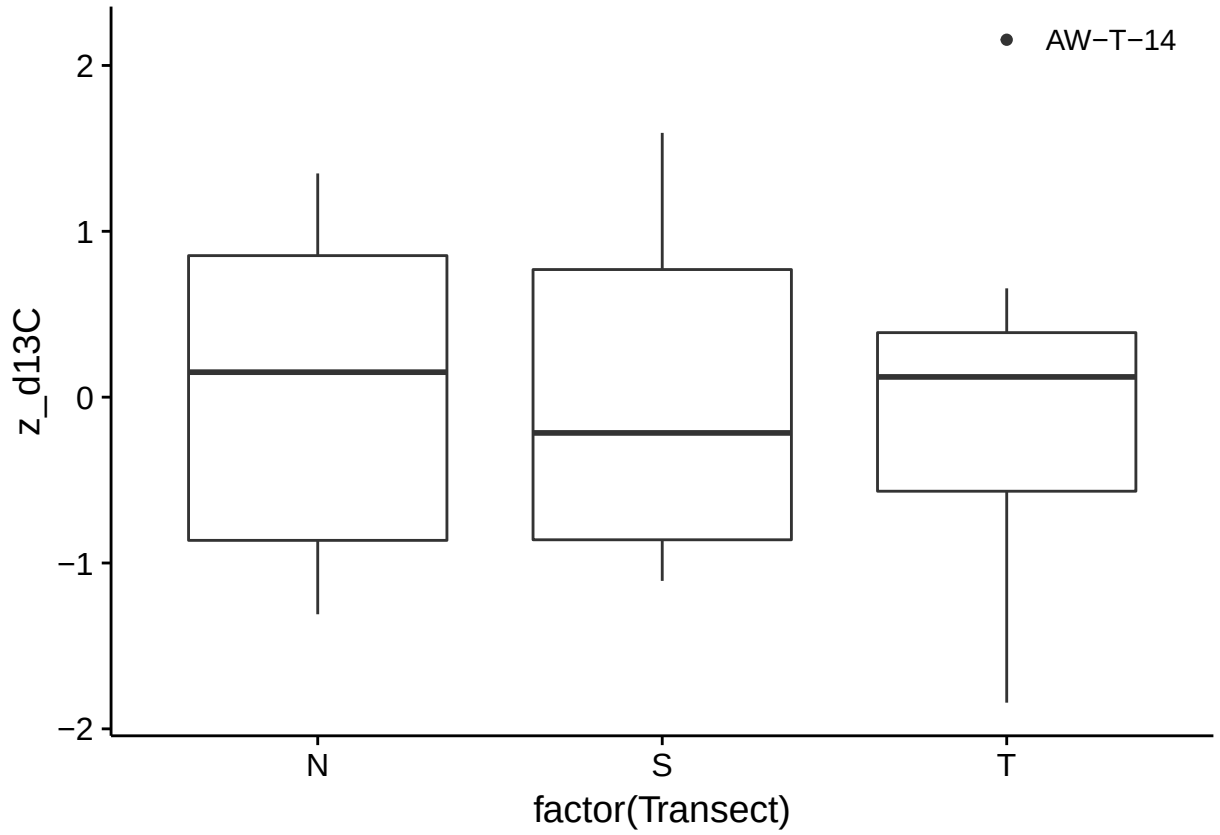
Soil Isotopes

```
# Isotopes

temp <- na.omit(soilGroups)

temp %>%
  group_by(Transect) %>%
  mutate(outlier = ifelse(is_outlier(comp.d13C), as.character(ID), NA)) %>%
  ggplot(., aes(x = factor(Transect), y = comp.d13C)) +
    geom_boxplot() +
    geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3)
```



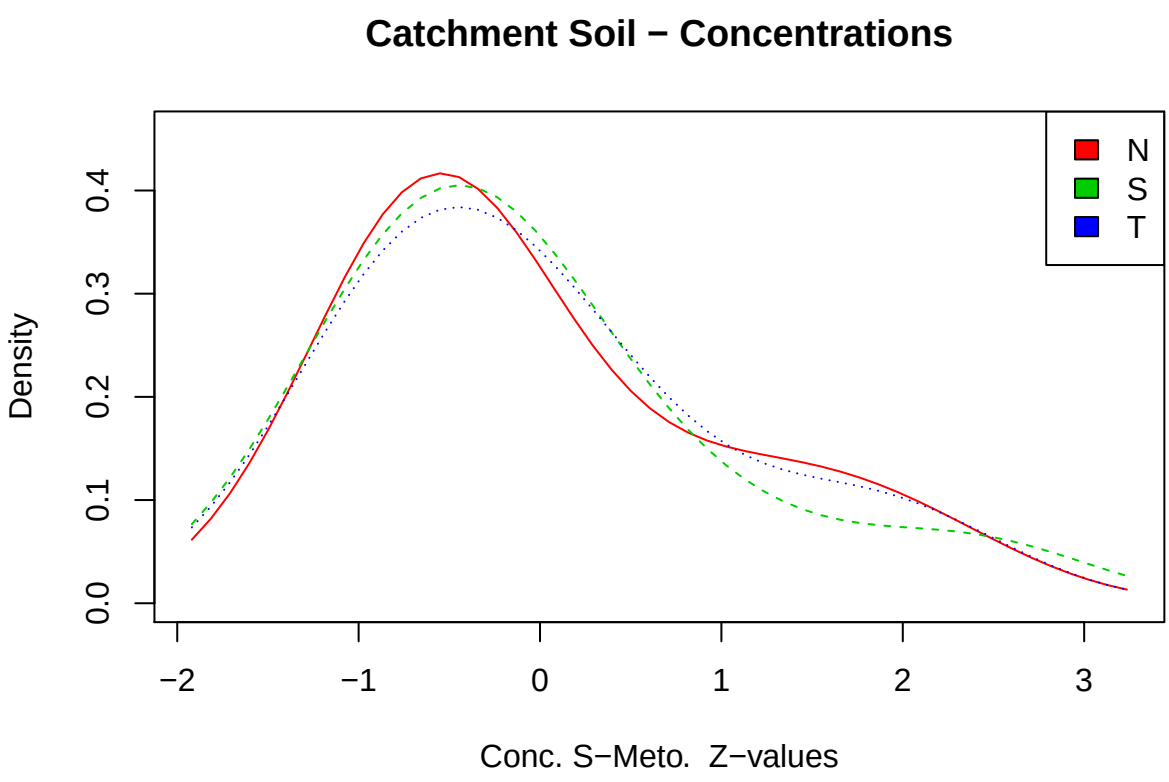
Looks like 7 potential outliers in concentrations and 1 for isotopes. Removing NA's for isotopes and re-computing outliers, reduces the number of outliers to 2 in concentrations and 1 for isotopes.

```
temp <- temp %>%
  group_by(Transect) %>%
  mutate(z_d13C = (comp.d13C-mean(comp.d13C))/sd(comp.d13C))

temp %>%
  group_by(Transect) %>%
  mutate(outlier = ifelse(is_outlier(z_d13C), as.character(ID), NA)) %>%
  ggplot(., aes(x = factor(Transect), y = z_d13C)) +
    geom_boxplot() +
    geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3)
```



Distribution of z values (same as non-transformed)

```
# plot densities
#sm.density.compare(temp$z_conc, temp$Transect, xlab=expression(paste("Conc. S-Meto.  ", {{mu}*g / g.s
sm.density.compare(temp$z_conc, temp$Transect, xlab=expression(paste("Conc. S-Meto.  Z-values")))
title(main="Catchment Soil - Concentrations")
legend("topright", levels( soilGroups$Transect), fill=2+(0:nlevels(soilGroups$Transect)))
```

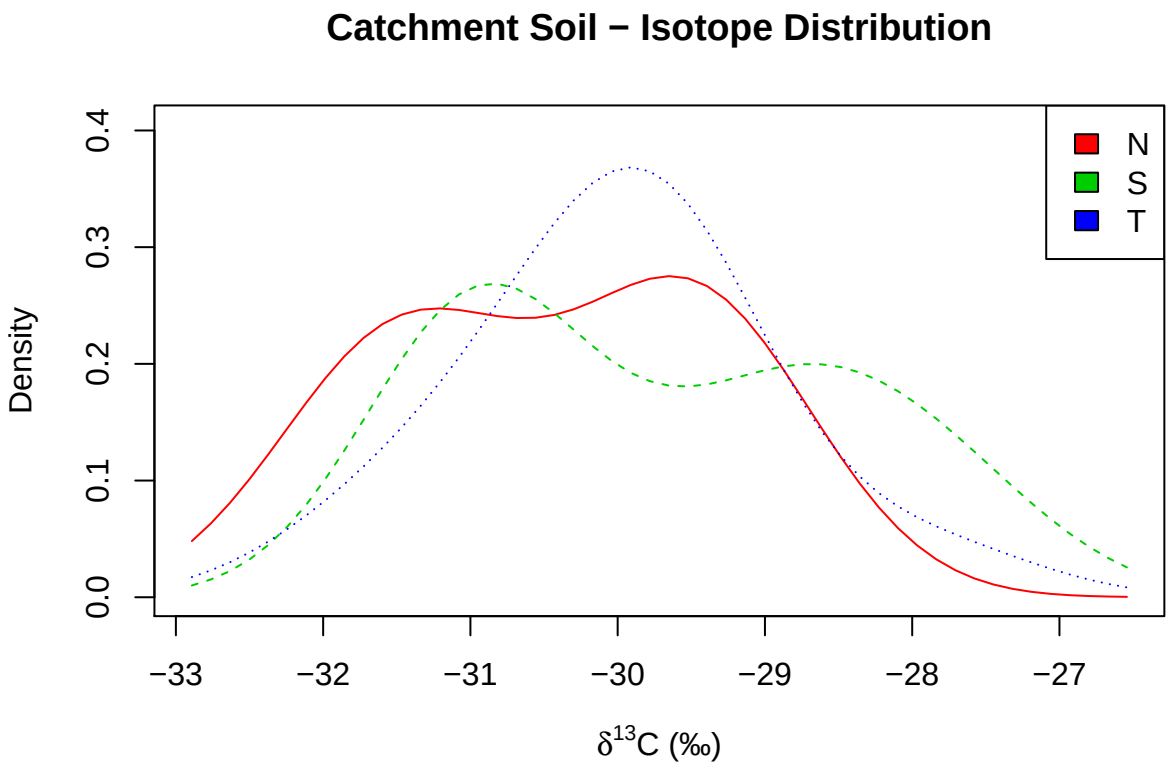



```
#vioplot(soilGroups$Conc.mug.g.dry.soil,  names = "Catchment")
#title(expression(paste("Conc. S-Meto.  ", {({mu}*g / g.soil.dry)})))
```

Soil Isotopes

```
#vioplot(na.omit(soilGroups$comp.d13C),  names = "Catchment")
#title(expression(paste({delta}^"13","C", ' (\u2030)')))
```

```
temp <- na.omit(soilGroups)
sm.density.compare(temp$comp.d13C, temp$Transect,
                   xlab=expression(paste({delta}^"13","C", ' (\u2030)')))
title(main="Catchment Soil - Isotope Distribution")
legend("topright", levels( soilGroups$Transect), fill=2+(0:nlevels(soilGroups$Transect)))
```


Catchment Soil − Isotope Distribution
Density
0.0
0.1
0.2
0.3
0.4
−33
−32
−31
−30
−29
−28
−27
δ13C (‰)
N
S
T