MP Oyster Depuration Project

Garth C

January 9, 2018

Load the libraries we’ll need. You’ll have to install these if you don’t already have them.

library(plyr)  
library(reshape2)  
library(ggplot2)  
library(glmmTMB)  
library(MuMIn)

## Load, organize, and plot data

Import the oyster depuration data and view it.

oysterdep <-   
 read.csv("~/GRAD school/ACRDP/Depuration/Data/Analysis/R code/oysterdep.csv")  
  
head(oysterdep)

## id sampleday size.cat Table Tank species DOFSample length  
## 1 MIS1 0 twentyto50 n/a n/a C. gigas 11-03-16 93  
## 2 MIS1 0 fiftyto100 n/a n/a C. gigas 11-03-16 93  
## 3 MIS1 0 onehundredto500 n/a n/a C. gigas 11-03-16 93  
## 4 MIS1 0 fivehundredto1000 n/a n/a C. gigas 11-03-16 93  
## 5 MIS1 0 onethousandto5000 n/a n/a C. gigas 11-03-16 93  
## 6 MIS2 0 twentyto50 n/a n/a C. gigas 11-03-16 66  
## width depth cont.weight cont.dryweight dry.weight shell.dry.weight CI  
## 1 43 21 259.28 260.96 1.68 17.65 0.095  
## 2 43 21 259.28 260.96 1.68 17.65 0.095  
## 3 43 21 259.28 260.96 1.68 17.65 0.095  
## 4 43 21 259.28 260.96 1.68 17.65 0.095  
## 5 43 21 259.28 260.96 1.68 17.65 0.095  
## 6 40 21 262.04 263.63 1.59 12.62 0.126  
## time.in.oven time.out.oven dat.filter dat.count observer  
## 1 2016-11-05/ 17:03 2016-11-10/18:00 11-17-2016 11-29-2016 MD  
## 2 2016-11-05/ 17:03 2016-11-10/18:00 11-17-2016 11-29-2016 MD  
## 3 2016-11-05/ 17:03 2016-11-10/18:00 11-17-2016 11-29-2016 MD  
## 4 2016-11-05/ 17:03 2016-11-10/18:00 11-17-2016 11-29-2016 MD  
## 5 2016-11-05/ 17:03 2016-11-10/18:00 11-17-2016 11-29-2016 MD  
## 6 2016-11-05/ 17:03 2016-11-10/18:00 11-17-2016 11-28-2016 GC  
## red.fib yell.fib green.fib turq.fib blu.fib purp.fib gray.fib clear.fib  
## 1 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0  
## 3 0 0 0 0 1 0 0 0  
## 4 0 0 0 0 1 0 0 0  
## 5 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0  
## pink.fib brown.fib orang.fib black.fib red.frag yell.frag green.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 1 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0  
## turq.frag blu.frag purp.frag gray.frag clear.frag pink.frag brown.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0  
## orang.frag black.frag whit.spher clear.spher  
## 1 0 0 0 0  
## 2 0 0 0 0  
## 3 0 0 0 0  
## 4 0 0 0 0  
## 5 0 0 0 0  
## 6 0 0 0 0

Get all of the variables into the right format.

oysterdep$id <- as.factor(oysterdep$id)  
oysterdep$sampleday <- as.factor(oysterdep$sampleday)  
oysterdep$size.cat <- as.factor(oysterdep$size.cat)  
oysterdep$Table <- as.factor(oysterdep$Table)  
oysterdep$Tank <- as.factor(oysterdep$Tank)  
oysterdep$observer <- as.factor(oysterdep$observer)

Make a new dataframe to work with and rename size categories.

oysterdep2 <- oysterdep  
  
oysterdep2$size.cat <- factor(oysterdep2$size.cat,   
 levels=c('twentyto50', 'fiftyto100',  
 'onehundredto500', 'fivehundredto1000',   
 'onethousandto5000')) # reorder  
  
  
oysterdep2$size.cat <- mapvalues(oysterdep2$size.cat,   
 from = c('twentyto50', 'fiftyto100',   
 'onehundredto500',   
 'fivehundredto1000',   
 'onethousandto5000'),   
 to = c('20-50', '50-100', '100-500',  
 '500-1000', '1000-5000')) # rename  
summary(oysterdep2$size.cat) # view

## 20-50 50-100 100-500 500-1000 1000-5000   
## 49 50 50 50 50

We need to melt the dataframe so that the particle types are organized by row instead of by column.

oysterdep3 <- melt(oysterdep2,  
 id.vars = c('id', 'sampleday', 'size.cat', 'Table', 'Tank',  
 'species', 'DOFSample', 'length', 'width',   
 'depth', 'cont.weight', 'cont.dryweight',   
 'dry.weight', 'shell.dry.weight', 'CI',  
 'time.in.oven', 'time.out.oven',  
 'dat.filter', 'dat.count', 'observer'))  
  
## Take out unnecessary columns  
  
oysterdep3 <- oysterdep3[c('id', 'sampleday', 'size.cat', 'Table', 'Tank',   
 'DOFSample', 'length', 'width', 'depth', 'dry.weight',  
 'shell.dry.weight', 'CI', 'dat.filter', 'dat.count',  
 'observer', 'variable', 'value')]  
  
head(oysterdep3) # now 'variable' is particle category and 'value is the count'

## id sampleday size.cat Table Tank DOFSample length width depth  
## 1 MIS1 0 20-50 n/a n/a 11-03-16 93 43 21  
## 2 MIS1 0 50-100 n/a n/a 11-03-16 93 43 21  
## 3 MIS1 0 100-500 n/a n/a 11-03-16 93 43 21  
## 4 MIS1 0 500-1000 n/a n/a 11-03-16 93 43 21  
## 5 MIS1 0 1000-5000 n/a n/a 11-03-16 93 43 21  
## 6 MIS2 0 20-50 n/a n/a 11-03-16 66 40 21  
## dry.weight shell.dry.weight CI dat.filter dat.count observer  
## 1 1.68 17.65 0.095 11-17-2016 11-29-2016 MD  
## 2 1.68 17.65 0.095 11-17-2016 11-29-2016 MD  
## 3 1.68 17.65 0.095 11-17-2016 11-29-2016 MD  
## 4 1.68 17.65 0.095 11-17-2016 11-29-2016 MD  
## 5 1.68 17.65 0.095 11-17-2016 11-29-2016 MD  
## 6 1.59 12.62 0.126 11-17-2016 11-28-2016 GC  
## variable value  
## 1 red.fib 0  
## 2 red.fib 0  
## 3 red.fib 0  
## 4 red.fib 0  
## 5 red.fib 0  
## 6 red.fib 0

Now we want to see what kind of particles we found and remove any categories that we didn’t end up using.

cat.sums <- aggregate(oysterdep3$value ~   
 oysterdep3$variable, FUN = sum) # sums everything  
print(cat.sums)

## oysterdep3$variable oysterdep3$value  
## 1 red.fib 1  
## 2 yell.fib 3  
## 3 green.fib 4  
## 4 turq.fib 1  
## 5 blu.fib 52  
## 6 purp.fib 0  
## 7 gray.fib 0  
## 8 clear.fib 302  
## 9 pink.fib 2  
## 10 brown.fib 9  
## 11 orang.fib 3  
## 12 black.fib 158  
## 13 red.frag 1  
## 14 yell.frag 0  
## 15 green.frag 1  
## 16 turq.frag 0  
## 17 blu.frag 2  
## 18 purp.frag 0  
## 19 gray.frag 0  
## 20 clear.frag 2  
## 21 pink.frag 2  
## 22 brown.frag 0  
## 23 orang.frag 0  
## 24 black.frag 0  
## 25 whit.spher 6  
## 26 clear.spher 1

So we can remove the purple fibre, gray fibre, yellow fragment, purple fragment, gray fragment, brown fragment, orange fragment, and black fragment categories because we didn’t count any of these. Since we’re only going to focus on fibres, we can actually remove all of the fragment categories.

oysterdep4 <- subset(oysterdep3, variable == 'red.fib' |   
 variable == 'yell.fib' | variable == 'green.fib' |  
 variable == 'turq.fib' | variable == 'blu.fib' |   
 variable == 'clear.fib' | variable == 'pink.fib' |  
 variable == 'brown.fib' | variable == 'orang.fib' |  
 variable == 'black.fib') # removes all these rows  
  
oysterdep4$variable <- as.character(oysterdep4$variable)  
oysterdep4$variable <- as.factor(oysterdep4$variable) # gets rid of those names  
  
summary(oysterdep4$variable) # view

## black.fib blu.fib brown.fib clear.fib green.fib orang.fib pink.fib   
## 249 249 249 249 249 249 249   
## red.fib turq.fib yell.fib   
## 249 249 249

Let’s make the category names a bit more useful for plotting

oysterdep4$variable <- mapvalues(oysterdep4$variable,  
 from = c(levels(oysterdep4$variable)),   
 to = c('Black fibres', 'Blue fibres',   
 'Brown fibres', 'Clear fibres',   
 'Green fibres','Orange fibres',  
 'Pink fibres', 'Red fibres',  
 'Turquoise fibres', 'Yellow fibres'))  
  
## And reorder them  
oysterdep4$variable <- factor(oysterdep4$variable,   
 levels = c('Black fibres', 'Blue fibres',  
 'Brown fibres', 'Clear fibres',  
 'Green fibres', 'Orange fibres',  
 'Pink fibres', 'Red fibres',  
 'Turquoise fibres',  
 'Yellow fibres'))  
  
summary(oysterdep4$variable)

## Black fibres Blue fibres Brown fibres Clear fibres   
## 249 249 249 249   
## Green fibres Orange fibres Pink fibres Red fibres   
## 249 249 249 249   
## Turquoise fibres Yellow fibres   
## 249 249

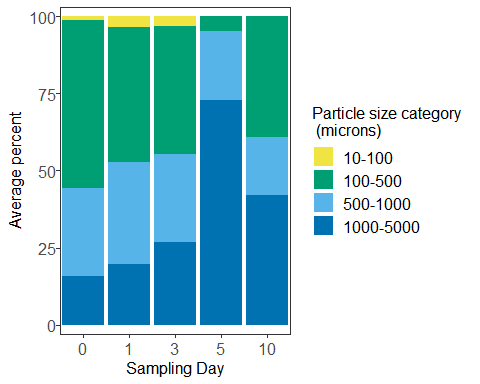
Let’s combine all size categories below 100 microns, since we weren’t very confident in identifying these size classesso it makes more sense to pool them.

oysterdep4$size.cat <-  
 mapvalues(  
 oysterdep4$size.cat,  
 from = c('20-50', '50-100'),  
 to = c('10-100', '10-100')  
 ) ## pool <100 micron particles  
  
oysterdep4 <- ddply(oysterdep4, names(oysterdep4)[-c(17)], summarize,  
 value = sum(value))

Next let’s make plots of the distribution of sizes and particle types and colours.

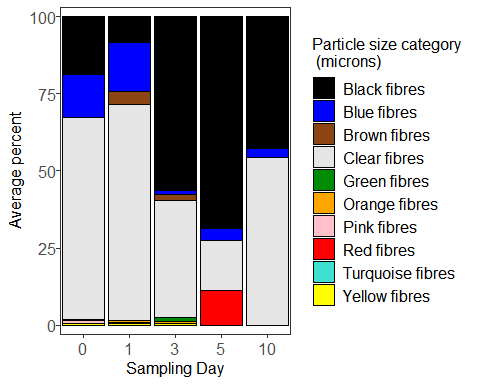
Starting with size

## First we need to make a new data frame of sums according to each size category  
  
oystersizesums <- ddply(oysterdep4, c('id', 'sampleday', 'size.cat', 'Table',   
 'Tank', 'DOFSample', 'length', 'width',  
 'depth', 'dry.weight', 'shell.dry.weight',  
 'CI', 'dat.filter', 'dat.count',   
 'observer'), summarize, sum = sum(value))  
  
  
  
## Now calculate proportions of each particle size category present in each   
## sample and take the means for each sampling day  
  
oysterprop <- ddply(oystersizesums, 'id', transform, prop = sum/sum(sum))  
  
oysterprop2 <- subset(oysterprop, !(is.na(oysterprop["prop"]))) # remove any NAs  
  
oysterprop2$size.cat <- as.character(oysterprop2$size.cat)  
oysterprop2$size.cat <- as.factor(oysterprop2$size.cat)  
  
oysterprop3 <- ddply(oysterprop2, c("sampleday", "size.cat"), summarize,  
 mean = mean(prop))  
  
## Put the size categories in order  
  
oysterprop3$size.cat <- factor(oysterprop3$size.cat,   
 levels = c("10-100",   
 "100-500",   
 "500-1000",   
 "1000-5000"))  
  
## Now plot  
  
ggplot(oysterprop3, aes(x=sampleday, y=100\*mean, fill = size.cat)) +   
 geom\_col(size = 1) +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c("#F0E442", "#009E73",   
 "#56B4E9", "#0072B2", "black")) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))



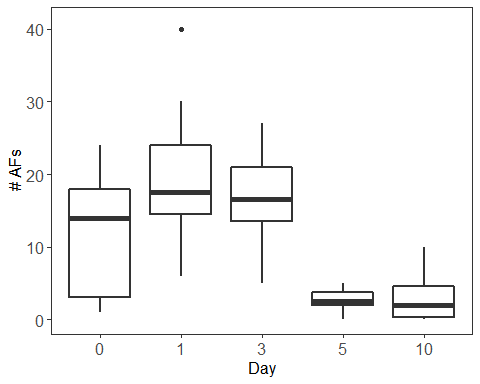
Now with particle type/colour

## First we need to make a new data frame of sums according to each particle  
## type/colour  
  
oystertypesums <- ddply(oysterdep4, c('id', 'sampleday', 'Table',   
 'Tank', 'DOFSample', 'length', 'width',  
 'depth', 'dry.weight', 'shell.dry.weight',  
 'CI', 'dat.filter', 'dat.count',   
 'observer', 'variable'), summarize,   
 sum = sum(value))  
  
  
  
## Now calculate proportions of each particle type/colour present in each   
## sample and take the means for each sampling day  
  
oystertypeprop <- ddply(oystertypesums, 'id', transform, prop = sum/sum(sum))  
  
oystertypeprop2 <- subset(oystertypeprop, !(is.na(oystertypeprop["prop"]))) # remove any NAs  
  
oystertypeprop2$variable <- as.character(oystertypeprop2$variable)  
oystertypeprop2$variable <- as.factor(oystertypeprop2$variable)  
  
oystertypeprop3 <- ddply(oystertypeprop2, c("sampleday", "variable"), summarize,  
 mean = mean(prop))  
  
## Now plot  
  
ggplot(oystertypeprop3, aes(x=sampleday, y=100\*mean, fill = variable)) +   
 geom\_col(size = 0.5, colour = 'black') +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c('black', 'blue', 'saddlebrown', 'grey90',   
 'green4', 'orange', 'pink', 'red', 'turquoise',   
 'yellow')) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))

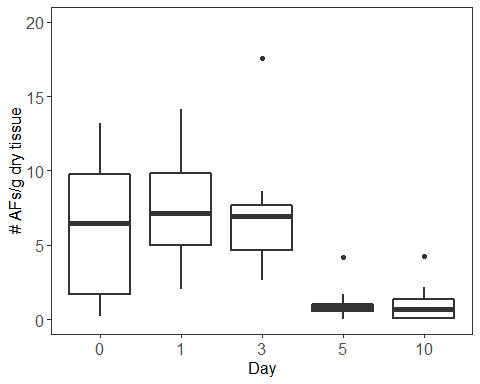


Take total sums overall and plot the results.

totalsums <- ddply(oysterdep4, c('id', 'sampleday', 'Table', 'Tank',   
 'length', 'width', 'depth', 'dry.weight',  
 'shell.dry.weight', 'CI', 'observer'),  
 summarise, sum = sum(value)) # adds all the counts together  
  
## Plot according to count per individual  
  
ggplot(totalsums, aes(x=sampleday , y=sum)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +   
 coord\_cartesian(ylim = c(0, 41)) +  
 ylab('# AFs') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 strip.background = element\_blank(),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())



## Plot according to concentration (particles/dry tissue weight)  
  
ggplot(totalsums, aes(x=sampleday , y=sum/dry.weight)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +   
 coord\_cartesian(ylim = c(0, 20)) +  
 ylab('# AFs/g dry tissue') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 strip.background = element\_blank(),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())



Let’s print out some summary stats.

## Summary stats according to count per individual  
  
with(totalsums, tapply(sum, sampleday, mean)) # takes mean for each day

## 0 1 3 5 10   
## 12.0 19.8 16.2 2.6 2.9

with(totalsums, tapply(sum, sampleday, min)) # takes minimum for each day

## 0 1 3 5 10   
## 1 6 5 0 0

with(totalsums, tapply(sum, sampleday, max)) # takes maximum for each day

## 0 1 3 5 10   
## 24 40 27 5 10

with(totalsums, tapply(sum,   
 sampleday, sd)) # takes standard deviation for each day

## 0 1 3 5 10   
## 8.793937 9.818350 6.746192 1.505545 3.281260

## Summary stats according to concentration  
  
with(totalsums, tapply(sum/dry.weight,   
 sampleday, mean)) # takes mean for each day

## 0 1 3 5 10   
## 6.018839 7.431444 7.128330 1.106082 1.043232

with(totalsums, tapply(sum/dry.weight,   
 sampleday, min)) # takes minimum for each day

## 0 1 3 5 10   
## 0.2386635 2.0270270 2.6415094 0.0000000 0.0000000

with(totalsums, tapply(sum/dry.weight,   
 sampleday, max)) # takes maximum for each day

## 0 1 3 5 10   
## 13.173653 14.150943 17.532468 4.201681 4.237288

with(totalsums, tapply(sum,   
 sampleday, sd)) # takes standard deviation for each day

## 0 1 3 5 10   
## 8.793937 9.818350 6.746192 1.505545 3.281260

## Proportions of particles sizes  
  
sum(oysterdep4$value[oysterdep4$size.cat == '10-100'])/  
 sum(oysterdep4$value)\*100

## [1] 2.242991

sum(oysterdep4$value[oysterdep4$size.cat == '100-500'])/  
 sum(oysterdep4$value)\*100

## [1] 41.86916

sum(oysterdep4$value[oysterdep4$size.cat == '500-1000'])/  
 sum(oysterdep4$value)\*100

## [1] 28.78505

sum(oysterdep4$value[oysterdep4$size.cat == '1000-5000'])/  
 sum(oysterdep4$value)\*100

## [1] 27.1028

## This provides the percentage of total particles found accounted for by each   
## colour/type of particles  
oystersummarytable <- with(oysterdep4, tapply(value, variable, sum))  
oystersummarytable <- oystersummarytable/sum(oysterdep4$value)\*100  
print(oystersummarytable)

## Black fibres Blue fibres Brown fibres Clear fibres   
## 29.5327103 9.7196262 1.6822430 56.4485981   
## Green fibres Orange fibres Pink fibres Red fibres   
## 0.7476636 0.5607477 0.3738318 0.1869159   
## Turquoise fibres Yellow fibres   
## 0.1869159 0.5607477

### Blanks

Now load in the blanks dataset and set it up for plotting, same as with the oyster data set. First we load in the data.

depblanks <-   
 read.csv("~/GRAD school/ACRDP/Depuration/Data/Analysis/depblanks.csv")  
  
head(depblanks)

## id sampleday size.cat DOFSample time.in.oven  
## 1 MControl1 (0,1) 0 twentyto50 11-17-2016 2016-11-05/ 17:03  
## 2 MControl1 (0,1) 0 fiftyto100 11-17-2016 2016-11-05/ 17:03  
## 3 MControl1 (0,1) 0 onehundredto500 11-17-2016 2016-11-05/ 17:03  
## 4 MControl1 (0,1) 0 fivehundredto1000 11-17-2016 2016-11-05/ 17:03  
## 5 MControl1 (0,1) 0 onethousandto5000 11-17-2016 2016-11-05/ 17:03  
## 6 MControl2 (0,1) 0 twentyto50 11-17-2016 2016-11-05/ 17:03  
## time.out.oven dat.filter dat.count observer red.fib yell.fib  
## 1 2016-11-10/18:00 11-17-2016 11-24-2016 GC 0 0  
## 2 2016-11-10/18:00 11-17-2016 11-24-2016 GC 0 0  
## 3 2016-11-10/18:00 11-17-2016 11-24-2016 GC 0 0  
## 4 2016-11-10/18:00 11-17-2016 11-24-2016 GC 0 0  
## 5 2016-11-10/18:00 11-17-2016 11-24-2016 GC 0 0  
## 6 2016-11-10/18:00 11-17-2016 11-24-2016 GC 0 0  
## green.fib turq.fib blu.fib purp.fib gray.fib clear.fib pink.fib  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 1 0  
## 4 0 0 0 0 0 1 0  
## 5 0 0 0 0 0 3 0  
## 6 0 0 0 0 0 0 0  
## brown.fib orang.fib black.fib tot.fib red.frag yell.frag green.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 1 0 0 0  
## 4 0 0 0 1 0 0 0  
## 5 0 0 0 3 0 0 0  
## 6 0 0 0 0 0 0 0  
## turq.frag blu.frag purp.frag gray.frag clear.frag pink.frag brown.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0  
## orang.frag black.frag whit.spher clear.spher tot.frag.spher tot.part  
## 1 0 0 0 0 0 0  
## 2 0 0 0 0 0 0  
## 3 0 0 0 0 0 0  
## 4 0 0 0 0 0 0  
## 5 0 0 0 0 0 0  
## 6 0 0 0 0 0 0

depblanks$id <- as.factor(depblanks$id)  
depblanks$size.cat <- as.factor(depblanks$size.cat)  
depblanks$observer <- as.factor(depblanks$observer)  
depblanks$sampleday <- as.factor(depblanks$sampleday)

Now we make a new dataframe to work with and rename size categories.

depblanks2 <- depblanks  
  
depblanks2$size.cat <- factor(depblanks2$size.cat,   
 levels=c('twentyto50', 'fiftyto100',  
 'onehundredto500', 'fivehundredto1000',   
 'onethousandto5000')) # reorder  
  
  
depblanks2$size.cat <- mapvalues(depblanks2$size.cat,   
 from = c('twentyto50', 'fiftyto100',   
 'onehundredto500',   
 'fivehundredto1000',   
 'onethousandto5000'),   
 to = c('20-50', '50-100', '100-500',  
 '500-1000', '1000-5000')) # rename  
summary(depblanks2$size.cat) # view

## 20-50 50-100 100-500 500-1000 1000-5000   
## 15 15 15 15 15

Again, we melt the dataframe so that the particle types are organized by row instead of by column.

depblanks3 <- melt(depblanks2,  
 id.vars = c('id', 'size.cat', 'sampleday', 'DOFSample',   
 'time.in.oven', 'time.out.oven', 'dat.filter',   
 'dat.count', 'observer'))  
  
## Take out unnecessary columns  
  
depblanks3 <- depblanks3[c('id', 'size.cat', 'sampleday', 'DOFSample',   
 'dat.filter', 'dat.count', 'observer', 'variable',   
 'value')]  
  
head(depblanks3) # now 'variable' is particle category and 'value is the count'

## id size.cat sampleday DOFSample dat.filter dat.count  
## 1 MControl1 (0,1) 20-50 0 11-17-2016 11-17-2016 11-24-2016  
## 2 MControl1 (0,1) 50-100 0 11-17-2016 11-17-2016 11-24-2016  
## 3 MControl1 (0,1) 100-500 0 11-17-2016 11-17-2016 11-24-2016  
## 4 MControl1 (0,1) 500-1000 0 11-17-2016 11-17-2016 11-24-2016  
## 5 MControl1 (0,1) 1000-5000 0 11-17-2016 11-17-2016 11-24-2016  
## 6 MControl2 (0,1) 20-50 0 11-17-2016 11-17-2016 11-24-2016  
## observer variable value  
## 1 GC red.fib 0  
## 2 GC red.fib 0  
## 3 GC red.fib 0  
## 4 GC red.fib 0  
## 5 GC red.fib 0  
## 6 GC red.fib 0

Now sum by totals across all samples and particles sizes to get an idea of what we’re working with.

cat.sums.blanks <- aggregate(depblanks3$value ~   
 depblanks3$variable, FUN = sum) # sums everything  
print(cat.sums.blanks)

## depblanks3$variable depblanks3$value  
## 1 red.fib 0  
## 2 yell.fib 0  
## 3 green.fib 0  
## 4 turq.fib 0  
## 5 blu.fib 4  
## 6 purp.fib 0  
## 7 gray.fib 0  
## 8 clear.fib 43  
## 9 pink.fib 2  
## 10 brown.fib 0  
## 11 orang.fib 0  
## 12 black.fib 0  
## 13 tot.fib 49  
## 14 red.frag 0  
## 15 yell.frag 0  
## 16 green.frag 0  
## 17 turq.frag 0  
## 18 blu.frag 2  
## 19 purp.frag 0  
## 20 gray.frag 0  
## 21 clear.frag 0  
## 22 pink.frag 0  
## 23 brown.frag 0  
## 24 orang.frag 0  
## 25 black.frag 0  
## 26 whit.spher 0  
## 27 clear.spher 0  
## 28 tot.frag.spher 2  
## 29 tot.part 2

So we can remove the red, yellow, green turquoise, purple, gray, brown, orange, and black fibre categories, and all non-fibre categories.

depblanks4 <- subset(depblanks3, c(variable == 'blu.fib' |   
 variable == 'clear.fib' | variable == 'pink.fib')  
 ) # keep only these rows  
  
depblanks4$variable <- as.character(depblanks4$variable)  
depblanks4$variable <- as.factor(depblanks4$variable) # gets rid of those names  
  
summary(depblanks4$variable) # view

## blu.fib clear.fib pink.fib   
## 75 75 75

Let’s make the category names a bit more useful for plotting

depblanks4$variable <- mapvalues(depblanks4$variable,  
 from = c(levels(depblanks4$variable)),   
 to = c('Blue fibres', 'Clear fibres',   
 'Pink fibres'))  
  
## And reorder them  
depblanks4$variable <- factor(depblanks4$variable,   
 levels = c('Blue fibres', 'Clear fibres',  
 'Pink fibres'))  
  
summary(depblanks4$variable)

## Blue fibres Clear fibres Pink fibres   
## 75 75 75

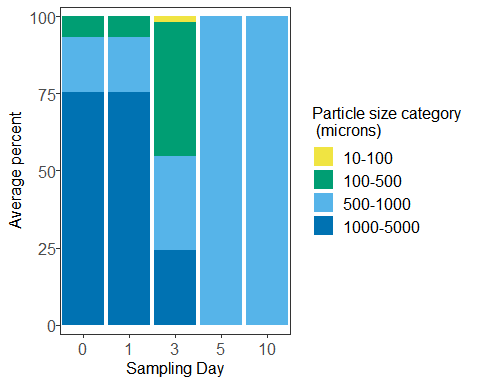
Let’s combine all size categories below 100 microns, since we weren’t very confident in identifying these size classesso it makes more sense to pool them.

depblanks4$size.cat <-  
 mapvalues(  
 depblanks4$size.cat,  
 from = c('20-50', '50-100'),  
 to = c('10-100', '10-100')  
 ) ## pool <100 micron particles  
  
depblanks4 <- ddply(depblanks4, names(depblanks4)[-9], summarize,  
 value = sum(value))

Next let’s make plots of the distribution of sizes and particle types and colours.

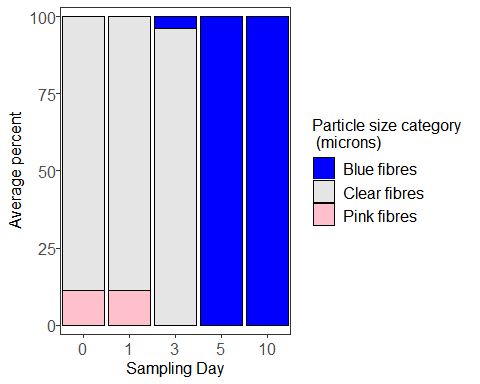
Starting with size

## First we need to make a new data frame of sums according to each size category  
  
blankssizesums <- ddply(depblanks4, c('id', 'size.cat', 'sampleday',   
 'DOFSample', 'dat.filter', 'dat.count',   
 'observer'), summarize, sum = sum(value))  
  
  
  
## Now calculate proportions of each particle size category present in each   
## sample and take the means for each sampling day  
  
blanksprop <- ddply(blankssizesums, c('id', 'sampleday'), transform, prop = sum/sum(sum))  
  
blanksprop2 <- subset(blanksprop, !(is.na(blanksprop["prop"]))) # remove any NAs  
  
blanksprop2$size.cat <- as.character(blanksprop2$size.cat)  
blanksprop2$size.cat <- as.factor(blanksprop2$size.cat)  
  
blanksprop3 <- ddply(blanksprop2, c("DOFSample", "sampleday", "size.cat"), summarize,  
 mean = mean(prop))  
  
## Put the size categories in order  
  
blanksprop3$size.cat <- factor(blanksprop3$size.cat,   
 levels = c("10-100",   
 "100-500", "500-1000",   
 "1000-5000"))  
  
## Now plot  
  
ggplot(blanksprop3, aes(x=sampleday, y=100\*mean, fill = size.cat)) +   
 geom\_col(size = 1) +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c("#F0E442", "#009E73",   
 "#56B4E9", "#0072B2", "black")) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))



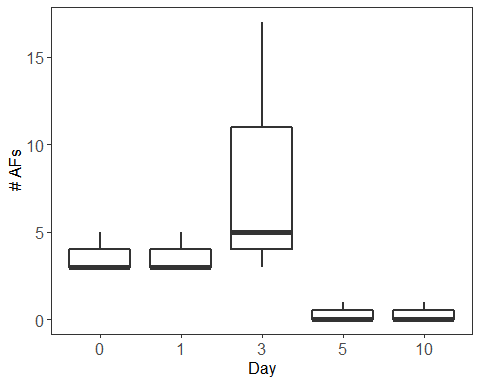
Now with particle type/colour

## First we need to make a new data frame of sums according to each particle  
## type/colour  
  
blankstypesums <- ddply(depblanks4, c('id', 'sampleday', 'DOFSample',   
 'dat.filter', 'dat.count', 'observer',   
 'variable'),   
 summarize, sum = sum(value))  
  
## Now calculate proportions of each particle type/colour present in each   
## sample and take the means for each sampling day  
  
blankstypeprop <- ddply(blankstypesums, c('id', 'sampleday'), transform,   
 prop = sum/sum(sum))  
  
blankstypeprop2 <- subset(blankstypeprop, !(is.na(blankstypeprop["prop"]))) # remove any NAs  
  
blankstypeprop2$variable <- as.character(blankstypeprop2$variable)  
blankstypeprop2$variable <- as.factor(blankstypeprop2$variable)  
  
blankstypeprop3 <- ddply(blankstypeprop2, c("DOFSample", "sampleday",   
 "variable"), summarize,  
 mean = mean(prop))  
  
## Now plot  
  
ggplot(blankstypeprop3, aes(x=sampleday, y=100\*mean, fill = variable)) +   
 geom\_col(size = 0.5, colour = 'black') +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c('blue', 'grey90', 'pink')) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))



Take total sums and plot the results.

totalblanksums <- ddply(depblanks4, c('id', 'sampleday', 'observer',   
 'dat.filter', 'DOFSample'),  
 summarise, sum = sum(value)) # adds all the counts together  
  
## Plot according to count per filtration day  
  
ggplot(totalblanksums, aes(x=sampleday, y=sum)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +   
 ylab('# AFs') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 strip.background = element\_blank(),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())



Now do the same summary stats for the blanks

## Means, range, and SD  
  
with(totalblanksums, tapply(sum, DOFSample, mean)) # takes mean for each day

## 11-17-2016 12-06-16   
## 5.2222222 0.3333333

with(totalblanksums, tapply(sum, DOFSample, min)) # takes minimum for each day

## 11-17-2016 12-06-16   
## 3 0

with(totalblanksums, tapply(sum, DOFSample, max)) # takes maximum for each day

## 11-17-2016 12-06-16   
## 17 1

with(totalblanksums, tapply(sum,   
 DOFSample, sd)) # takes standard deviation for each day

## 11-17-2016 12-06-16   
## 4.5215533 0.5163978

## Proportions of particles sizes  
  
sum(depblanks4$value[depblanks4$size.cat == '10-100'])/  
 sum(depblanks4$value)\*100

## [1] 2.040816

sum(depblanks4$value[depblanks4$size.cat == '100-500'])/  
 sum(depblanks4$value)\*100

## [1] 34.69388

sum(depblanks4$value[depblanks4$size.cat == '500-1000'])/  
 sum(depblanks4$value)\*100

## [1] 24.4898

sum(depblanks4$value[depblanks4$size.cat == '1000-5000'])/  
 sum(depblanks4$value)\*100

## [1] 38.77551

## This provides the percentage of total particles found accounted for by each   
## colour/type of particles  
blanksummarytable <- with(depblanks4, tapply(value, variable, sum))  
blanksummarytable <- blanksummarytable/sum(depblanks4$value)\*100  
print(blanksummarytable)

## Blue fibres Clear fibres Pink fibres   
## 8.163265 87.755102 4.081633

Now subtract the blank counts from the appropriate size and shape categories for the oyster counts.

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

depblanksmean <- ddply(depblanks4, c('size.cat', 'sampleday', 'dat.filter',   
 'variable'),  
 summarize, value.b = mean(value))  
  
depblanksmean$value.b <-  
 ceiling(depblanksmean$value.b) ## Round all blank averages up to the nearest whole number  
  
oysterdep5 <-  
 left\_join(oysterdep4, depblanksmean,   
 by = c('sampleday', 'size.cat', 'variable')) ## Join the 2 dataframes

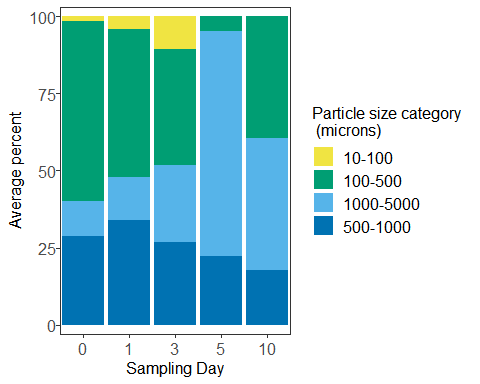
## Warning: Column `variable` joining factors with different levels, coercing  
## to character vector

oysterdep5$value.b <- as.numeric(oysterdep5$value.b)  
oysterdep5$value.b[is.na(oysterdep5$value.b)] <- 0  
  
oysterdep4$variable <- as.factor(oysterdep4$variable)  
  
oysterdep5$finalcount <- (oysterdep5$value - oysterdep5$value.b)  
  
oysterdep5$finalcount[oysterdep5$finalcount < 0] <-  
 0 ## Change any negative values to 0  
  
summary(oysterdep5$finalcount)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 0.000 0.000 0.218 0.000 12.000

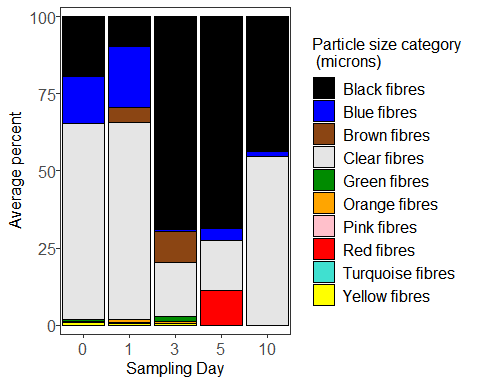
Now run all of the same code again but for the modified oyster counts.

## First we need to make a new data frame of sums according to each size category  
  
oystersizesums2 <- ddply(oysterdep5, c('id', 'sampleday', 'size.cat', 'Table',   
 'Tank', 'DOFSample', 'length', 'width',  
 'depth', 'dry.weight', 'shell.dry.weight',  
 'CI', 'dat.count',   
 'observer'), summarize, sum = sum(finalcount))  
  
  
  
## Now calculate proportions of each particle size category present in each   
## sample and take the means for each sampling day  
  
oysterprop4 <- ddply(oystersizesums2, 'id', transform, prop = sum/sum(sum))  
  
oysterprop5 <- subset(oysterprop4, !(is.na(oysterprop4["prop"]))) # remove any NAs  
  
oysterprop5$size.cat <- as.character(oysterprop5$size.cat)  
oysterprop5$size.cat <- as.factor(oysterprop5$size.cat)  
  
oysterprop6 <- ddply(oysterprop5, c("sampleday", "size.cat"), summarize,  
 mean = mean(prop))  
  
  
## Now plot  
  
ggplot(oysterprop6, aes(x=sampleday, y=100\*mean, fill = size.cat)) +   
 geom\_col(size = 1) +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c("#F0E442", "#009E73",   
 "#56B4E9", "#0072B2", "black")) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))



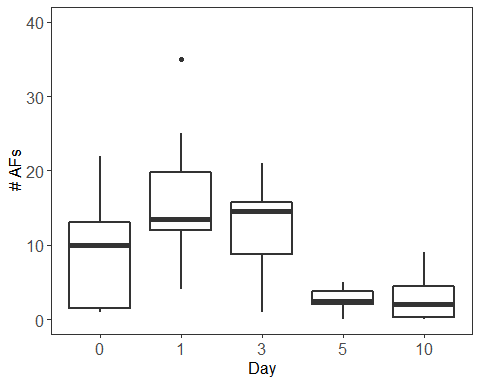
Now with particle type/colour

## First we need to make a new data frame of sums according to each particle  
## type/colour  
  
oystertypesums2 <- ddply(oysterdep5, c('id', 'sampleday', 'Table',   
 'Tank', 'DOFSample', 'length', 'width',  
 'depth', 'dry.weight', 'shell.dry.weight',  
 'CI', 'dat.count',   
 'observer', 'variable'), summarize,   
 sum = sum(finalcount))  
  
  
  
## Now calculate proportions of each particle type/colour present in each   
## sample and take the means for each sampling day  
  
oystertypeprop4 <- ddply(oystertypesums2, 'id', transform, prop = sum/sum(sum))  
  
oystertypeprop5 <- subset(oystertypeprop4, !(is.na(oystertypeprop4["prop"]))) # remove any NAs  
  
oystertypeprop5$variable <- as.character(oystertypeprop5$variable)  
oystertypeprop5$variable <- as.factor(oystertypeprop5$variable)  
  
oystertypeprop6 <- ddply(oystertypeprop5, c("sampleday", "variable"), summarize,  
 mean = mean(prop))  
  
## Now plot  
  
ggplot(oystertypeprop6, aes(x=sampleday, y=100\*mean, fill = variable)) +   
 geom\_col(size = 0.5, colour = 'black') +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c('black', 'blue', 'saddlebrown', 'grey90',   
 'green4', 'orange', 'pink', 'red', 'turquoise',   
 'yellow')) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))

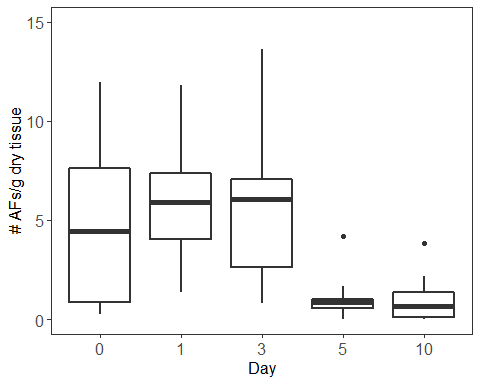


Take total sums overall and plot the results.

totalsums2 <- ddply(oysterdep5, c('id', 'sampleday', 'Table', 'Tank',   
 'length', 'width', 'depth', 'dry.weight',  
 'shell.dry.weight', 'CI', 'observer'),  
 summarise, sum = sum(finalcount)) # adds all the counts together  
  
## Plot according to count per individual  
  
ggplot(totalsums2, aes(x=sampleday , y=sum)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +   
 coord\_cartesian(ylim = c(0, 40)) +  
 ylab('# AFs') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 strip.background = element\_blank(),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())



## Plot according to concentration (particles/dry tissue weight)  
  
ggplot(totalsums2, aes(x=sampleday , y=sum/dry.weight)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +   
 coord\_cartesian(ylim = c(0, 15)) +  
 ylab('# AFs/g dry tissue') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 strip.background = element\_blank(),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())



Let’s print out some summary stats.

## Summary stats according to count per individual  
  
with(totalsums2, tapply(sum, sampleday, mean)) # takes mean for each day

## 0 1 3 5 10   
## 9.4 16.1 12.7 2.6 2.8

with(totalsums2, tapply(sum, sampleday, min)) # takes minimum for each day

## 0 1 3 5 10   
## 1 4 1 0 0

with(totalsums2, tapply(sum, sampleday, max)) # takes maximum for each day

## 0 1 3 5 10   
## 22 35 21 5 9

with(totalsums2, tapply(sum,   
 sampleday, sd)) # takes standard deviation for each day

## 0 1 3 5 10   
## 7.876830 8.849984 6.600505 1.505545 3.047768

## Summary stats according to concentration  
  
with(totalsums2, tapply(sum/dry.weight,   
 sampleday, mean)) # takes mean for each day

## 0 1 3 5 10   
## 4.760769 5.932840 5.588296 1.106082 1.000859

with(totalsums2, tapply(sum/dry.weight,   
 sampleday, min)) # takes minimum for each day

## 0 1 3 5 10   
## 0.2386635 1.3513514 0.8000000 0.0000000 0.0000000

with(totalsums2, tapply(sum/dry.weight,   
 sampleday, max)) # takes maximum for each day

## 0 1 3 5 10   
## 11.976048 11.792453 13.636364 4.201681 3.813559

with(totalsums2, tapply(sum/dry.weight,   
 sampleday, sd)) # takes standard deviation for each day

## 0 1 3 5 10   
## 4.228159 2.865865 3.821515 1.175746 1.213063

## Proportions of particles sizes  
  
sum(oystersizesums2$sum[oystersizesums2$size.cat == '10-100'])/  
 sum(oystersizesums2$sum)\*100

## [1] 2.522936

sum(oystersizesums2$sum[oystersizesums2$size.cat == '100-500'])/  
 sum(oystersizesums2$sum)\*100

## [1] 43.34862

sum(oystersizesums2$sum[oystersizesums2$size.cat == '500-1000'])/  
 sum(oystersizesums2$sum)\*100

## [1] 29.3578

sum(oystersizesums2$sum[oystersizesums2$size.cat == '1000-5000'])/  
 sum(oystersizesums2$sum)\*100

## [1] 24.77064

## This provides the percentage of total particles found accounted for by each   
## colour/type of particles  
oystersummarytable <- with(oystertypesums2, tapply(sum, variable, sum))  
oystersummarytable <- oystersummarytable/sum(oystertypesums2$sum)\*100  
print(oystersummarytable)

## Black fibres Blue fibres Brown fibres Clear fibres   
## 36.2385321 11.4678899 2.0642202 47.2477064   
## Green fibres Orange fibres Pink fibres Red fibres   
## 0.9174312 0.6880734 0.2293578 0.2293578   
## Turquoise fibres Yellow fibres   
## 0.2293578 0.6880734

Now load in the water sample data.

depwater <-   
 read.csv("~/GRAD school/ACRDP/Depuration/Data/Analysis/depwater.csv")  
  
head(depwater)

## id day size.cat Table Tank DOFSample  
## 1 Water Day 0 A 0 twentyto50 A 4 12-13-2017  
## 2 Water Day 0 A 0 fiftyto100 A 4 12-13-2017  
## 3 Water Day 0 A 0 onehundredto500 A 4 12-13-2017  
## 4 Water Day 0 A 0 fivehundredto1000 A 4 12-13-2017  
## 5 Water Day 0 A 0 onethousandto5000 A 4 12-13-2017  
## 6 Water Day 0 B 0 twentyto50 B 3 12-13-2017  
## time.in.oven time.out.oven dat.filter dat.count observer red.fib  
## 1 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017 MD 0  
## 2 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017 MD 0  
## 3 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017 MD 0  
## 4 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017 MD 0  
## 5 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017 MD 0  
## 6 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-21-2017 MD 0  
## yell.fib green.fib turq.fib blu.fib purp.fib gray.fib clear.fib pink.fib  
## 1 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 1 0  
## 6 0 0 0 0 0 0 0 0  
## brown.fib orang.fib black.fib tot.fib red.frag yell.frag green.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 1 0 0 0  
## 6 0 0 0 0 0 0 0  
## turq.frag blu.frag purp.frag gray.frag clear.frag pink.frag brown.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0  
## orang.frag black.frag whit.spher clear.spher tot.frag.spher tot.part  
## 1 0 0 0 0 0 0  
## 2 0 0 0 0 0 0  
## 3 0 0 0 0 0 0  
## 4 0 0 0 0 0 0  
## 5 0 0 0 0 0 0  
## 6 0 0 0 0 0 0

depwater$id <- as.factor(depwater$id)  
depwater$day <- as.factor(depwater$day)  
depwater$size.cat <- as.factor(depwater$size.cat)  
depwater$Table <- as.factor(depwater$Table)  
depwater$Tank <- as.factor(depwater$Tank)  
depwater$observer <- as.factor(depwater$observer)

Now we make a new dataframe to work with and rename size categories.

depwater2 <- depwater  
  
depwater2$size.cat <- factor(depwater2$size.cat,   
 levels=c('twentyto50', 'fiftyto100',  
 'onehundredto500', 'fivehundredto1000',   
 'onethousandto5000')) # reorder  
  
  
depwater2$size.cat <- mapvalues(depwater2$size.cat,   
 from = c('twentyto50', 'fiftyto100',   
 'onehundredto500',   
 'fivehundredto1000',   
 'onethousandto5000'),   
 to = c('20-50', '50-100', '100-500',  
 '500-1000', '1000-5000')) # rename  
summary(depwater2$size.cat) # view

## 20-50 50-100 100-500 500-1000 1000-5000   
## 20 20 20 20 20

Again, we melt the dataframe so that the particle types are organized by row instead of by column.

depwater3 <- melt(depwater2,  
 id.vars = c('id', 'day', 'size.cat', 'Table', 'Tank',   
 'DOFSample', 'time.in.oven', 'time.out.oven',   
 'dat.filter', 'dat.count', 'observer'))  
  
## Take out unnecessary columns  
  
depwater3 <- depwater3[c('id', 'day', 'size.cat', 'Table', 'Tank', 'DOFSample',   
 'dat.filter', 'dat.count', 'observer', 'variable',   
 'value')]  
  
head(depwater3) # now 'variable' is particle category and 'value is the count'

## id day size.cat Table Tank DOFSample dat.filter dat.count  
## 1 Water Day 0 A 0 20-50 A 4 12-13-2017 12-13-2017 12-20-2017  
## 2 Water Day 0 A 0 50-100 A 4 12-13-2017 12-13-2017 12-20-2017  
## 3 Water Day 0 A 0 100-500 A 4 12-13-2017 12-13-2017 12-20-2017  
## 4 Water Day 0 A 0 500-1000 A 4 12-13-2017 12-13-2017 12-20-2017  
## 5 Water Day 0 A 0 1000-5000 A 4 12-13-2017 12-13-2017 12-20-2017  
## 6 Water Day 0 B 0 20-50 B 3 12-13-2017 12-13-2017 12-21-2017  
## observer variable value  
## 1 MD red.fib 0  
## 2 MD red.fib 0  
## 3 MD red.fib 0  
## 4 MD red.fib 0  
## 5 MD red.fib 0  
## 6 MD red.fib 0

Now sum by totals across all samples and particles sizes to get an idea of what we’re working with.

cat.sums.blanks <- aggregate(depwater3$value ~   
 depwater3$variable, FUN = sum) # sums everything  
print(cat.sums.blanks)

## depwater3$variable depwater3$value  
## 1 red.fib 3  
## 2 yell.fib 0  
## 3 green.fib 0  
## 4 turq.fib 1  
## 5 blu.fib 4  
## 6 purp.fib 0  
## 7 gray.fib 0  
## 8 clear.fib 37  
## 9 pink.fib 0  
## 10 brown.fib 0  
## 11 orang.fib 0  
## 12 black.fib 0  
## 13 tot.fib 45  
## 14 red.frag 0  
## 15 yell.frag 0  
## 16 green.frag 0  
## 17 turq.frag 0  
## 18 blu.frag 0  
## 19 purp.frag 0  
## 20 gray.frag 0  
## 21 clear.frag 0  
## 22 pink.frag 0  
## 23 brown.frag 0  
## 24 orang.frag 0  
## 25 black.frag 0  
## 26 whit.spher 0  
## 27 clear.spher 0  
## 28 tot.frag.spher 0  
## 29 tot.part 0

So we can remove the yellow, green, purple, gray, pink, brown, orange, and black fibre categories, and all other categories as well.

depwater4 <- subset(depwater3, c(variable == 'red.fib' |   
 variable == 'turq.fib' | variable == 'blu.fib' |  
 variable == 'clear.fib')) # keep only these rows  
  
depwater4$variable <- as.character(depwater4$variable)  
depwater4$variable <- as.factor(depwater4$variable) # gets rid of those names  
  
summary(depwater4$variable) # view

## blu.fib clear.fib red.fib turq.fib   
## 100 100 100 100

Let’s make the category names a bit more useful for plotting

depwater4$variable <- mapvalues(depwater4$variable,  
 from = c(levels(depwater4$variable)),   
 to = c('Blue fibres', 'Clear fibres',   
 'Red fibres', 'Turquoise fibres'))  
  
## And reorder them  
depwater4$variable <- factor(depwater4$variable,   
 levels = c('Blue fibres', 'Clear fibres',  
 'Red fibres', 'Turquoise fibres'))  
  
summary(depwater4$variable)

## Blue fibres Clear fibres Red fibres Turquoise fibres   
## 100 100 100 100

Let’s combine all size categories below 100 microns, since we weren’t very confident in identifying these size classesso it makes more sense to pool them.

depwater4$size.cat <-  
 mapvalues(  
 depwater4$size.cat,  
 from = c('20-50', '50-100'),  
 to = c('10-100', '10-100')  
 ) ## pool <100 micron particles  
  
depwater4 <- ddply(depwater4, names(depwater4)[-11], summarize,  
 value = sum(value))

Now do all these same things for the water blanks.

waterblanks <-   
 read.csv("~/GRAD school/ACRDP/Depuration/Data/Analysis/waterblanks.csv")  
  
head(waterblanks)

## id day size.cat Table Tank  
## 1 Water Control Control 1 Control twentyto50 Control n/a  
## 2 Water Control Control 1 Control fiftyto100 Control n/a  
## 3 Water Control Control 1 Control onehundredto500 Control n/a  
## 4 Water Control Control 1 Control fivehundredto1000 Control n/a  
## 5 Water Control Control 1 Control onethousandto5000 Control n/a  
## 6 Water Control Control 2 Control twentyto50 Control n/a  
## DOFSample time.in.oven time.out.oven dat.filter dat.count  
## 1 12-13-2017 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017  
## 2 12-13-2017 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017  
## 3 12-13-2017 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017  
## 4 12-13-2017 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017  
## 5 12-13-2017 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017  
## 6 12-13-2017 2017-01-09/17:00 2017-01-17/09:30 12-13-2017 12-20-2017  
## observer red.fib yell.fib green.fib turq.fib blu.fib purp.fib gray.fib  
## 1 MD 0 0 0 0 0 0 0  
## 2 MD 0 0 0 0 0 0 0  
## 3 MD 0 0 0 0 0 0 0  
## 4 MD 0 0 0 0 0 0 0  
## 5 MD 0 0 0 0 0 0 0  
## 6 MD 0 0 0 0 0 0 0  
## clear.fib pink.fib brown.fib orang.fib black.fib tot.fib red.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0  
## yell.frag green.frag turq.frag blu.frag purp.frag gray.frag clear.frag  
## 1 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0  
## pink.frag brown.frag orang.frag black.frag whit.spher clear.spher  
## 1 0 0 0 0 0 0  
## 2 0 0 0 0 0 0  
## 3 0 0 0 0 0 0  
## 4 0 0 0 0 0 0  
## 5 0 0 0 0 0 0  
## 6 0 0 0 0 0 0  
## tot.frag.spher tot.part  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 0 0

waterblanks$id <- as.factor(waterblanks$id)  
waterblanks$day <- as.factor(waterblanks$day)  
waterblanks$size.cat <- as.factor(waterblanks$size.cat)  
waterblanks$Table <- as.factor(waterblanks$Table)  
waterblanks$Tank <- as.factor(waterblanks$Tank)  
waterblanks$observer <- as.factor(waterblanks$observer)

Now we make a new dataframe to work with and rename size categories.

waterblanks2 <- waterblanks  
  
waterblanks2$size.cat <- factor(waterblanks2$size.cat,   
 levels=c('twentyto50', 'fiftyto100',  
 'onehundredto500', 'fivehundredto1000',   
 'onethousandto5000')) # reorder  
  
  
waterblanks2$size.cat <- mapvalues(waterblanks2$size.cat,   
 from = c('twentyto50', 'fiftyto100',   
 'onehundredto500',   
 'fivehundredto1000',   
 'onethousandto5000'),   
 to = c('20-50', '50-100', '100-500',  
 '500-1000', '1000-5000')) # rename  
summary(waterblanks2$size.cat) # view

## 20-50 50-100 100-500 500-1000 1000-5000   
## 3 3 3 3 3

Again, we melt the dataframe so that the particle types are organized by row instead of by column.

waterblanks3 <- melt(waterblanks2,  
 id.vars = c('id', 'day', 'size.cat', 'Table', 'Tank',   
 'DOFSample', 'time.in.oven', 'time.out.oven',   
 'dat.filter', 'dat.count', 'observer'))  
  
## Take out unnecessary columns  
  
waterblanks3 <- waterblanks3[c('id', 'day', 'size.cat', 'Table', 'Tank', 'DOFSample',   
 'dat.filter', 'dat.count', 'observer', 'variable',   
 'value')]  
  
head(waterblanks3) # now 'variable' is particle category and 'value is the count'

## id day size.cat Table Tank DOFSample  
## 1 Water Control Control 1 Control 20-50 Control n/a 12-13-2017  
## 2 Water Control Control 1 Control 50-100 Control n/a 12-13-2017  
## 3 Water Control Control 1 Control 100-500 Control n/a 12-13-2017  
## 4 Water Control Control 1 Control 500-1000 Control n/a 12-13-2017  
## 5 Water Control Control 1 Control 1000-5000 Control n/a 12-13-2017  
## 6 Water Control Control 2 Control 20-50 Control n/a 12-13-2017  
## dat.filter dat.count observer variable value  
## 1 12-13-2017 12-20-2017 MD red.fib 0  
## 2 12-13-2017 12-20-2017 MD red.fib 0  
## 3 12-13-2017 12-20-2017 MD red.fib 0  
## 4 12-13-2017 12-20-2017 MD red.fib 0  
## 5 12-13-2017 12-20-2017 MD red.fib 0  
## 6 12-13-2017 12-20-2017 MD red.fib 0

Now sum by totals across all samples and particles sizes to get an idea of what we’re working with.

cat.sums.blanks <- aggregate(waterblanks3$value ~   
 waterblanks3$variable, FUN = sum) # sums everything  
print(cat.sums.blanks)

## waterblanks3$variable waterblanks3$value  
## 1 red.fib 0  
## 2 yell.fib 0  
## 3 green.fib 0  
## 4 turq.fib 0  
## 5 blu.fib 3  
## 6 purp.fib 0  
## 7 gray.fib 0  
## 8 clear.fib 0  
## 9 pink.fib 0  
## 10 brown.fib 0  
## 11 orang.fib 0  
## 12 black.fib 0  
## 13 tot.fib 3  
## 14 red.frag 0  
## 15 yell.frag 0  
## 16 green.frag 0  
## 17 turq.frag 0  
## 18 blu.frag 0  
## 19 purp.frag 0  
## 20 gray.frag 0  
## 21 clear.frag 0  
## 22 pink.frag 0  
## 23 brown.frag 0  
## 24 orang.frag 0  
## 25 black.frag 0  
## 26 whit.spher 0  
## 27 clear.spher 0  
## 28 tot.frag.spher 0  
## 29 tot.part 0

So we can remove the red, yellow, green, turquoise, purple, gray, clear, pink, brown, orange, and black fibre categories, and all other categories as well.

waterblanks4 <- subset(waterblanks3, variable == 'blu.fib') # keep only these rows  
  
waterblanks4$variable <- as.character(waterblanks4$variable)  
waterblanks4$variable <- as.factor(waterblanks4$variable) # gets rid of those names  
  
summary(waterblanks4$variable) # view

## blu.fib   
## 15

Let’s make the category names a bit more useful for plotting

waterblanks4$variable <- mapvalues(waterblanks4$variable,  
 from = c(levels(waterblanks4$variable)),   
 to = 'Blue fibres')  
  
summary(waterblanks4$variable)

## Blue fibres   
## 15

Let’s combine all size categories below 100 microns, since we weren’t very confident in identifying these size classesso it makes more sense to pool them.

waterblanks4$size.cat <-  
 mapvalues(  
 waterblanks4$size.cat,  
 from = c('20-50', '50-100'),  
 to = c('10-100', '10-100')  
 ) ## pool <100 micron particles  
  
waterblanks4 <- ddply(waterblanks4, names(waterblanks4)[-11], summarize,  
 value = sum(value))

Now subtract the blank counts from the appropriate size and shape categories for the water counts.

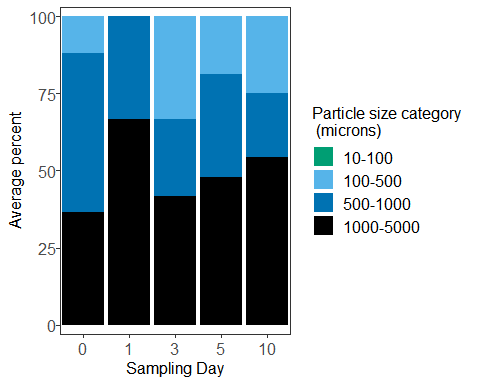
depwater4$dat.filter <- as.factor(depwater4$dat.filter)  
waterblanks4$dat.filter <- as.factor(waterblanks4$dat.filter)  
  
waterblanksmean <- ddply(waterblanks4, c('size.cat', 'dat.filter',   
 'variable'),  
 summarize, value.b = mean(value))  
  
depwater5 <-  
 left\_join(depwater4, waterblanksmean,   
 by = c('dat.filter', 'size.cat', 'variable')) ## Join the 2 dataframes

## Warning: Column `variable` joining factors with different levels, coercing  
## to character vector

depwater5$value.b <- as.numeric(depwater5$value.b)  
depwater5$value.b[is.na(depwater5$value.b)] <- 0  
  
depwater5$variable <- as.factor(depwater5$variable)  
  
depwater5$finalcount <- (depwater5$value - depwater5$value.b)  
  
depwater5$finalcount[depwater5$finalcount < 0] <-  
 0 ## Change any negative values to 0  
  
summary(depwater5$finalcount)

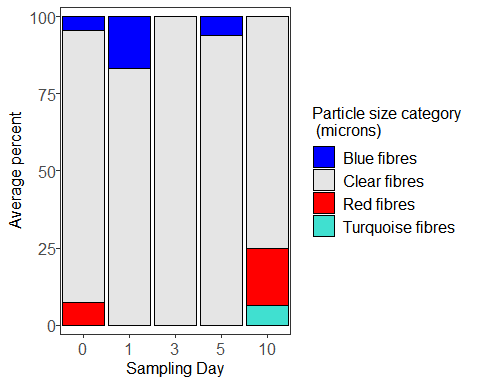
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0000 0.0000 0.0000 0.1344 0.0000 4.0000

## First we need to make a new data frame of sums according to each size category  
  
depwater5$day <- factor(depwater5$day,   
 levels =   
 c('0', '1', '3', '5', '10')) # put in order  
  
watersizesums <- ddply(depwater5, c('id', 'day', 'size.cat', 'Table', 'Tank',   
 'DOFSample', 'dat.filter', 'dat.count',   
 'observer'), summarize, sum = sum(finalcount))  
  
  
  
## Now calculate proportions of each particle size category present in each   
## sample and take the means for each sampling day  
  
waterprop <- ddply(watersizesums, 'id', transform, prop = sum/sum(sum))  
  
waterprop2 <- subset(waterprop, !(is.na(waterprop["prop"]))) # remove any NAs  
  
waterprop2$size.cat <- as.character(waterprop2$size.cat)  
waterprop2$size.cat <- as.factor(waterprop2$size.cat)  
  
waterprop3 <- ddply(waterprop2, c("day", "size.cat"), summarize,  
 mean = mean(prop))  
  
## Put the size categories in order  
  
waterprop3$size.cat <- factor(waterprop3$size.cat,   
 levels = c("10-100", "100-500", "500-1000",   
 "1000-5000"))  
  
## Now plot  
  
ggplot(waterprop3, aes(x=day, y=100\*mean, fill = size.cat)) +   
 geom\_col(size = 1) +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c("#009E73", "#56B4E9", "#0072B2", "black")) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))



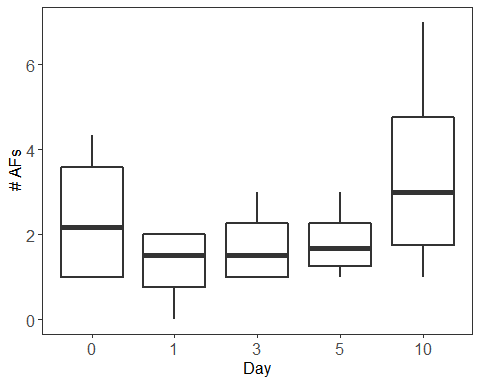
Now with particle type/colour

## First we need to make a new data frame of sums according to each particle  
## type/colour  
  
watertypesums <- ddply(depwater5, c('id', 'day', 'Table', 'Tank',   
 'DOFSample', 'dat.filter', 'dat.count',   
 'observer', 'variable'),   
 summarize, sum = sum(finalcount))  
  
  
  
## Now calculate proportions of each particle type/colour present in each   
## sample and take the means for each sampling day  
  
watertypeprop <- ddply(watertypesums, 'id', transform, prop = sum/sum(sum))  
  
watertypeprop2 <- subset(watertypeprop, !(is.na(watertypeprop["prop"]))) # remove any NAs  
  
watertypeprop2$variable <- as.character(watertypeprop2$variable)  
watertypeprop2$variable <- as.factor(watertypeprop2$variable)  
  
watertypeprop3 <- ddply(watertypeprop2, c("day", "variable"), summarize,  
 mean = mean(prop))  
  
## Now plot  
  
ggplot(watertypeprop3, aes(x=day, y=100\*mean, fill = variable)) +   
 geom\_col(size = 0.5, colour = 'black') +   
 xlab('Sampling Day') +   
 ylab('Average percent') +  
 guides(fill=guide\_legend(title="Particle size category \n (microns)")) +  
 theme\_bw() +  
 scale\_fill\_manual(values = c('blue', 'grey90', 'red', 'turquoise')) +  
 theme(legend.text = element\_text(size=12), text = element\_text(size=12),   
 panel.spacing = unit(1, "lines"),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12,   
 margin = margin(t=0, r=0, b=0, l=2)),  
 strip.background = element\_blank(),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 strip.placement = "outside") +  
 scale\_x\_discrete(expand = expand\_scale(0,0.5)) +   
 scale\_y\_continuous(expand = expand\_scale(0.03,0))



Take total sums and plot the results.

totalwatersums <- ddply(depwater5, c('id', 'day', 'Table', 'Tank', 'observer',   
 'dat.filter'),  
 summarise, sum = sum(finalcount)) # adds all the counts together  
  
## Plot according to count per filtration day  
  
ggplot(totalwatersums, aes(x=day, y=sum)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +  
 ylab('# AFs') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 strip.background = element\_blank(),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())



Now run the same summary stats for the water samples.

## Means, range, and SD  
  
with(totalwatersums, tapply(sum, day, mean)) # takes mean for each day

## 0 1 3 5 10   
## 2.416667 1.250000 1.750000 1.833333 3.500000

with(totalwatersums, tapply(sum, day, min)) # takes minimum for each day

## 0 1 3 5 10   
## 1 0 1 1 1

with(totalwatersums, tapply(sum, day, max)) # takes maximum for each day

## 0 1 3 5 10   
## 4.333333 2.000000 3.000000 3.000000 7.000000

with(totalwatersums, tapply(sum,   
 day, sd)) # takes standard deviation for each day

## 0 1 3 5 10   
## 1.6859990 0.9574271 0.9574271 0.8819171 2.6457513

## Proportions of particles sizes  
  
sum(depwater4$value[depwater4$size.cat == '10-100'])/  
 sum(depwater4$value)\*100

## [1] 0

sum(depwater4$value[depwater4$size.cat == '100-500'])/  
 sum(depwater4$value)\*100

## [1] 17.77778

sum(depwater4$value[depwater4$size.cat == '500-1000'])/  
 sum(depwater4$value)\*100

## [1] 31.11111

sum(depwater4$value[depwater4$size.cat == '1000-5000'])/  
 sum(depwater4$value)\*100

## [1] 51.11111

## This provides the percentage of total particles found accounted for by each   
## colour/type of particles  
watersummarytable <- with(depwater4, tapply(value, variable, sum))  
watersummarytable <- watersummarytable/sum(depwater4$value)\*100  
print(watersummarytable)

## Blue fibres Clear fibres Red fibres Turquoise fibres   
## 8.888889 82.222222 6.666667 2.222222

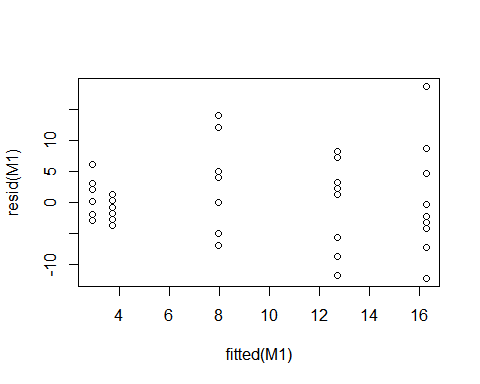
## MODELLING

Now fit a generalized linear mixed model (GLMM) to the oyster data using the package glmmTMB to determine differences over time. We’ll specify AF count as the response variable, sampling day as the predictor, and tank nested within table as the random, or mixed, effects. You’ll notice that sample day is specified twice. The second time is to set it up as the intercept for the random effect, which will allow the linear relationship between AP sum and sampling day to vary in both slope and intercept according separately within each tank, within each table. This accounts for any the variation that may have been caused by any variables that we did not account for and may have varied somehow in different tanks and or/tables.

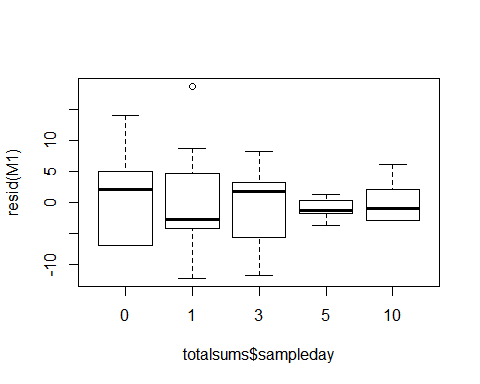
M1 <- glmmTMB(sum ~ sampleday + (1 | Table/Tank),   
 family = genpois(link = 'log'),   
 data = totalsums2) ## Fits a GLMM  
  
summary(M1) # model output

## Family: genpois ( log )  
## Formula: sum ~ sampleday + (1 | Table/Tank)  
## Data: totalsums2  
##   
## AIC BIC logLik deviance df.resid   
## 308.2 323.5 -146.1 292.2 42   
##   
## Random effects:  
##   
## Conditional model:  
## Groups Name Variance Std.Dev.   
## Tank:Table (Intercept) 1.720e-09 4.148e-05  
## Table (Intercept) 1.237e-11 3.517e-06  
## Number of obs: 50, groups: Tank:Table, 11; Table, 5  
##   
## Overdispersion parameter for genpois family (): 4.82   
##   
## Conditional model:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.0734 0.2325 8.919 <2e-16 \*\*\*  
## sampleday1 0.7159 0.2803 2.554 0.0107 \*   
## sampleday3 0.4705 0.2931 1.605 0.1084   
## sampleday5 -0.7537 0.3679 -2.048 0.0405 \*   
## sampleday10 -1.0061 0.4084 -2.464 0.0138 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

plot(resid(M1) ~ fitted(M1)) # plot residuals vs. fitted



plot(resid(M1) ~ totalsums$sampleday) # plot residuals vs. predictor



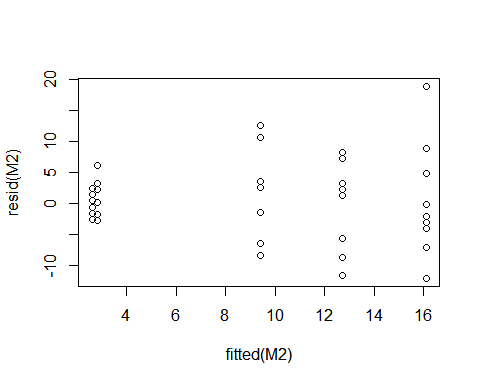
Looking at the plot of residuals vs. fitted and residuals vs. sampe day, we can see that the variance of the residuals varies according to the AP count and the sampling day. This violates the assumption of homogeneity of variance.

Let’s try fitting the same model structure, but assuming that the response variable (AF count) varies at each value of x (sampling day) according to a negative binomial distribution.

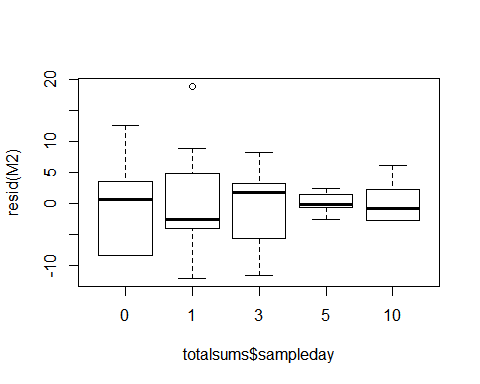
M2 <- glmmTMB(sum ~ sampleday + (1 | Table/Tank),   
 family = nbinom2(link = 'log'),  
 data = totalsums2) ## Fits a NB GLMM  
  
summary(M2) # model output

## Family: nbinom2 ( log )  
## Formula: sum ~ sampleday + (1 | Table/Tank)  
## Data: totalsums2  
##   
## AIC BIC logLik deviance df.resid   
## 306.0 321.3 -145.0 290.0 42   
##   
## Random effects:  
##   
## Conditional model:  
## Groups Name Variance Std.Dev.   
## Tank:Table (Intercept) 2.303e-10 1.517e-05  
## Table (Intercept) 1.398e-10 1.182e-05  
## Number of obs: 50, groups: Tank:Table, 11; Table, 5  
##   
## Overdispersion parameter for nbinom2 family (): 2.51   
##   
## Conditional model:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.2407 0.2246 9.978 < 2e-16 \*\*\*  
## sampleday1 0.5381 0.3105 1.733 0.083124 .   
## sampleday3 0.3009 0.3132 0.961 0.336705   
## sampleday5 -1.2852 0.3587 -3.583 0.000340 \*\*\*  
## sampleday10 -1.2111 0.3549 -3.413 0.000643 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

plot(resid(M2) ~ fitted(M2)) # plot residuals vs. fitted



plot(resid(M2) ~ totalsums$sampleday) # plot residuals vs. predictor



## Can compare M1 and M2 using AICc  
  
AICc(M1, M2)

## df AICc  
## M1 8 311.7047  
## M2 8 309.4901

# the negative binomial model with quadratically increasing variance is a better fit

The negative binomial model, which assumes that the variance of AF concentration increases quadratically with day of sampling, is a better fit.

Now let’s try pulling out the predictor (sampling day) and using the anova() function to compare the first model with the null models to determine whether sampling day significantly predicts AP concentration.

M3 <- glmmTMB(sum ~ (1 | Table/Tank), family = nbinom2(link = 'log'),   
 data = totalsums2) ## Same structure except no predictor  
  
anova(M2, M3) # compare the null model with the previously fitted model

## Data: totalsums2  
## Models:  
## M3: sum ~ (1 | Table/Tank), zi=~0, disp=~1  
## M2: sum ~ sampleday + (1 | Table/Tank), zi=~0, disp=~1  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)   
## M3 4 329.98 337.63 -160.99 321.98   
## M2 8 305.98 321.27 -144.99 289.98 32.006 4 1.908e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Sampling day is significant with a p-value of <0.001. We can now go into the model to determine which days are significantly different from each other. Looking back at the summarry of M1 we can see that the AP concentrations at days 5 and 10 are both significantly different from day 0 (the reference level).

Now we reorder the levels in sampleday to vary the reference level and figure out which days are significantly different from which other days.

totalsums2$sampleday <- relevel(totalsums2$sampleday, '1')  
M2 <- glmmTMB(sum ~ sampleday + (1 | Table/Tank), family = nbinom2(),   
 data = totalsums2) ## Fits a NB GLMM  
  
summary(M2) # model output

## Family: nbinom2 ( log )  
## Formula: sum ~ sampleday + (1 | Table/Tank)  
## Data: totalsums2  
##   
## AIC BIC logLik deviance df.resid   
## 306.0 321.3 -145.0 290.0 42   
##   
## Random effects:  
##   
## Conditional model:  
## Groups Name Variance Std.Dev.   
## Tank:Table (Intercept) 3.714e-10 1.927e-05  
## Table (Intercept) 1.703e-10 1.305e-05  
## Number of obs: 50, groups: Tank:Table, 11; Table, 5  
##   
## Overdispersion parameter for nbinom2 family (): 2.51   
##   
## Conditional model:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.7788 0.2145 12.956 < 2e-16 \*\*\*  
## sampleday0 -0.5381 0.3105 -1.733 0.0831 .   
## sampleday3 -0.2372 0.3061 -0.775 0.4383   
## sampleday5 -1.8233 0.3525 -5.173 2.31e-07 \*\*\*  
## sampleday10 -1.7492 0.3486 -5.018 5.22e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Similarly, days 5 and 10 are significantly different from day 1.

totalsums2$sampleday <- relevel(totalsums2$sampleday, '3')  
M2 <- glmmTMB(sum ~ sampleday + (1 | Table/Tank), family = nbinom2(),   
 data = totalsums2) ## Fits a NB GLMM  
  
summary(M2) # model output

## Family: nbinom2 ( log )  
## Formula: sum ~ sampleday + (1 | Table/Tank)  
## Data: totalsums2  
##   
## AIC BIC logLik deviance df.resid   
## 306.0 321.3 -145.0 290.0 42   
##   
## Random effects:  
##   
## Conditional model:  
## Groups Name Variance Std.Dev.   
## Tank:Table (Intercept) 3.329e-10 1.825e-05  
## Table (Intercept) 2.503e-10 1.582e-05  
## Number of obs: 50, groups: Tank:Table, 11; Table, 5  
##   
## Overdispersion parameter for nbinom2 family (): 2.51   
##   
## Conditional model:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.5416 0.2183 11.641 < 2e-16 \*\*\*  
## sampleday1 0.2372 0.3061 0.775 0.438   
## sampleday0 -0.3009 0.3132 -0.961 0.337   
## sampleday5 -1.5861 0.3549 -4.470 7.83e-06 \*\*\*  
## sampleday10 -1.5120 0.3510 -4.308 1.65e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Days 5 and 10 are significantly different from day 3.

totalsums2$sampleday <- relevel(totalsums2$sampleday, '5')  
M2 <- glmmTMB(sum ~ sampleday + (1 | Table/Tank), family = nbinom2(),   
 data = totalsums2) ## Fits a NB GLMM  
  
summary(M2) # model output

## Family: nbinom2 ( log )  
## Formula: sum ~ sampleday + (1 | Table/Tank)  
## Data: totalsums2  
##   
## AIC BIC logLik deviance df.resid   
## 306.0 321.3 -145.0 290.0 42   
##   
## Random effects:  
##   
## Conditional model:  
## Groups Name Variance Std.Dev.   
## Tank:Table (Intercept) 3.018e-10 1.737e-05  
## Table (Intercept) 2.530e-10 1.590e-05  
## Number of obs: 50, groups: Tank:Table, 11; Table, 5  
##   
## Overdispersion parameter for nbinom2 family (): 2.51   
##   
## Conditional model:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.95551 0.27974 3.416 0.000636 \*\*\*  
## sampleday3 1.58609 0.35485 4.470 7.83e-06 \*\*\*  
## sampleday1 1.82331 0.35250 5.173 2.31e-07 \*\*\*  
## sampleday0 1.28520 0.35872 3.583 0.000340 \*\*\*  
## sampleday10 0.07411 0.39212 0.189 0.850099   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

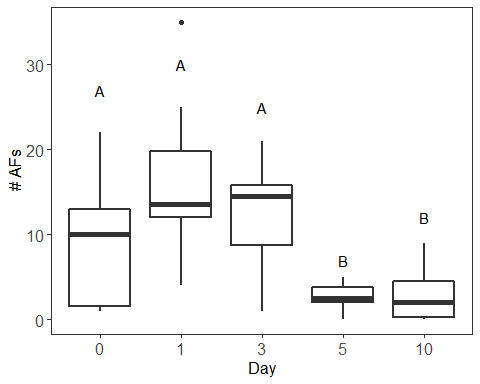
Day 10 is the only day not significantly different from day 5.

So in summary:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Day | 0 | 1 | 3 | 5 | 10 |
| 0 | x | - | - | sig | sig |
| —- | — | — | — | — | —– |
| 1 | x | x | - | sig | sig |
| —- | — | — | — | — | —– |
| 3 | x | x | x | sig | sig |
| —- | — | — | — | — | —– |
| 5 | x | x | x | x | - |
| —- | — | — | — | — | —– |
| 10 | x | x | x | x | x |

Now let’s replot the data with letters used to indicate which days are significantly different from one another. Those with different letters are statistically different.

## First we have to get the days back in the right order  
  
totalsums2$sampleday <- factor(totalsums2$sampleday,   
 levels = c("0","1","3","5", "10"))  
  
## Now plot  
  
ggplot(totalsums2, aes(x=sampleday , y=sum)) +   
 geom\_boxplot(position=position\_dodge(), size=1) +   
 xlab('Day') +   
 coord\_cartesian(ylim = c(0, 35)) +  
 ylab('# AFs') +  
 guides(fill=FALSE) +  
 theme\_bw() +  
 theme(text = element\_text(size=12),   
 axis.text.x = element\_text(size = 12),  
 axis.text.y = element\_text(size = 12),  
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank()) +  
 annotate("text", x=1, y=27, label = "A") +   
 annotate("text", x=2, y=30, label = "A") +  
 annotate("text", x=3, y=25, label = "A") +  
 annotate("text", x=4, y=7, label = "B") +  
 annotate("text", x=5, y=12, label = "B")



Run tests to determine whether there were significant differences in AP size according to day.

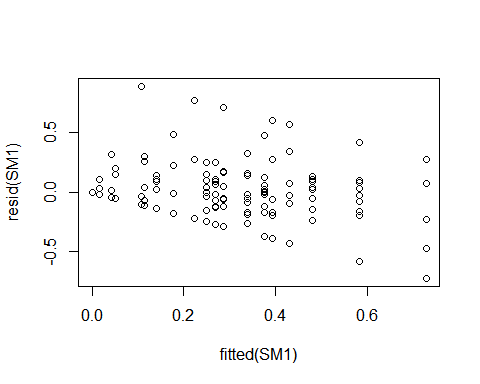
SM1 <- glmmTMB(prop ~ size.cat\*sampleday + (1 | Table/Tank), family = binomial,   
 data = oysterprop5)

## Warning in eval(family$initialize): non-integer #successes in a binomial  
## glm!

summary(SM1)

## Family: binomial ( logit )  
## Formula: prop ~ size.cat \* sampleday + (1 | Table/Tank)  
## Data: oysterprop5  
##   
## AIC BIC logLik deviance df.resid   
## 208.7 279.4 -82.3 164.7 162   
##   
## Random effects:  
##   
## Conditional model:  
## Groups Name Variance Std.Dev.   
## Tank:Table (Intercept) 1.069e-15 3.270e-08  
## Table (Intercept) 8.578e-14 2.929e-07  
## Number of obs: 184, groups: Tank:Table, 11; Table, 5  
##   
## Conditional model:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -4.0546 2.4430 -1.660 0.0970 .  
## size.cat100-500 4.3836 2.5257 1.736 0.0826 .  
## size.cat1000-5000 2.0089 2.6372 0.762 0.4462   
## size.cat500-1000 3.1444 2.5410 1.238 0.2159   
## sampleday1 0.9132 2.9131 0.314 0.7539   
## sampleday3 1.9294 2.6491 0.728 0.4664   
## sampleday5 -17.3395 14742.2469 -0.001 0.9991   
## sampleday10 -17.3531 16829.8201 -0.001 0.9992   
## size.cat100-500:sampleday1 -1.3262 3.0492 -0.435 0.6636   
## size.cat1000-5000:sampleday1 -0.6785 3.2095 -0.211 0.8326   
## size.cat500-1000:sampleday1 -0.6705 3.0694 -0.218 0.8271   
## size.cat100-500:sampleday3 -2.7697 2.8027 -0.988 0.3230   
## size.cat1000-5000:sampleday3 -0.9848 2.9220 -0.337 0.7361   
## size.cat500-1000:sampleday3 -2.0192 2.8311 -0.713 0.4757   
## size.cat100-500:sampleday5 14.0661 14742.2470 0.001 0.9992   
## size.cat1000-5000:sampleday5 20.3687 14742.2469 0.001 0.9989   
## size.cat500-1000:sampleday5 16.9970 14742.2469 0.001 0.9991   
## size.cat100-500:sampleday10 16.5921 16829.8201 0.001 0.9992   
## size.cat1000-5000:sampleday10 19.1176 16829.8201 0.001 0.9991   
## size.cat500-1000:sampleday10 16.7209 16829.8201 0.001 0.9992   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

plot(resid(SM1) ~ fitted(SM1))



SM2 <- glmmTMB(prop ~ size.cat + sampleday + (1 | Table/Tank), family = binomial,   
 data = oysterprop5)

## Warning in eval(family$initialize): non-integer #successes in a binomial  
## glm!

anova(SM1, SM2) # interaction is signficant at p = 0.04

## Data: oysterprop5  
## Models:  
## SM2: prop ~ size.cat + sampleday + (1 | Table/Tank), zi=~0, disp=~1  
## SM1: prop ~ size.cat \* sampleday + (1 | Table/Tank), zi=~0, disp=~1  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)   
## SM2 10 206.08 238.23 -93.042 186.08   
## SM1 22 208.66 279.39 -82.331 164.66 21.421 12 0.04454 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

ggplot(oysterprop5) +  
 geom\_violin(aes(x = sampleday, y = prop)) +  
 facet\_wrap(~ size.cat)

