Activity_Data

Sara Michele Schaal

6/2/2017

```
##
## 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
```

Data Manipulation

```
# Read in data files
  full.act <- read.csv("data/Activity/20171129_ActivityData.csv")
  black.temps <- read.csv("data/black_exp_temps.csv")
  copper.temps <- read.csv("data/copper_exp_temps.csv")

# Subset data into two experiments
  black.act <- full.act[full.act$Species == "Black", ]
  copper.act <- full.act[full.act$Species == "Copper", ]

# Reorder data to be in numerical ascending order
  # **(only way for loop to work properly)
  black.temps.asc <- black.temps[order(black.temps$Bucket), ]
  copper.temps.asc <- copper.temps[order(copper.temps$Bucket_ID), ]

# Fill in actual mean temperature data from experiment
  #black
  for(i in 1:16){
    for(j in 1:nrow(black.act)){
      if(black.act$Bucket[j] == i){
        black.act$Temperature[j] <- black.temps.asc$mean[i]
      }
    }
  }

  #copper
  for(i in 1:16){
    for(j in 1:nrow(copper.act)){
      if(copper.act$Bucket[j] == i){
        copper.act$Temperature[j] <- copper.temps.asc$mean[i]
```

```
    }
  }
}


# take the mean of the two watches of the same fish per viewer
  black.act$Act_Mean_Julia <- rowMeans(black.act[, 8:9])
  black.act$Act_Mean_Sandi <- rowMeans(black.act[, 10:11])
  copper.act$Act_Mean_Julia <- rowMeans(copper.act[,8:9])
  copper.act$Act_Mean_Sandi <- rowMeans(copper.act[,10:11])
```

Calculate averages per bucket per timepoint

```
### Black Rockfish ###

  # create empty data frames for mean bucket activity values
  Mean.Activity <- matrix(NA, nrow(unique(black.act[c("Date", "AM_PM", "Bucket")])),3)
  bucket.act.average <- unique(black.act[c("Date", "AM_PM", "Bucket", "Temperature")])
  complete.act.data <- cbind(bucket.act.average, Mean.Activity)
  colnames(complete.act.data) <- c("Date", "AM_PM", "Bucket", "Temperature",
                                   "mean.julia", "mean.sandi", "mean.both")

  # initialize the vectors that will be used in the forloop ** NOTE data in these vectors is written
  # over in each loop.. they are TEMPORARY vectors **
  bucket.act.julia <- NULL
  bucket.act.sandi <- NULL
  bucket.act.both <- NULL

  # initialize the counter for stepping through the output matrix
  j <- 1

  # For loop to step through data frame to get activity averages
  for(i in 1:nrow(black.act)){

    # looping through majority of data to grab all those rows that are not the last row
    # but are all those that equal the same bucket from the same day and getting activity averages
    if(i != nrow(black.act) && i == 1 ||
       black.act$Bucket[i] == black.act$Bucket[i-1] && i != nrow(black.act)){

      bucket.act.julia <- c(bucket.act.julia, black.act$Act_Mean_Julia[i])
      bucket.act.sandi <- c(bucket.act.sandi, black.act$Act_Mean_Sandi[i])
      bucket.act.both <- c(bucket.act.both, black.act$Act_Mean_Julia[i], black.act$Act_Mean_Sandi[i])

      # special case for last row
     } else {
       if(i == nrow(black.act)){

        # Remove any NAs before taking mean for a bucket at a timepoint
        bucket.act.both.noNA <- bucket.act.both[!is.na(bucket.act.both)]
        bucket.act.julia.noNA <- bucket.act.julia[!is.na(bucket.act.julia)]
        bucket.act.sandi.noNA <- bucket.act.sandi[!is.na(bucket.act.sandi)]
```

```
          complete.act.data$mean.julia[j] <- sum(bucket.act.julia.noNA)/length(bucket.act.julia.noNA)
          complete.act.data$mean.sandi[j] <- sum(bucket.act.sandi.noNA)/length(bucket.act.sandi.noNA)
          complete.act.data$mean.both[j] <- sum(bucket.act.both.noNA)/length(bucket.act.both.noNA)
          j <- j + 1

        } else {

          # Remove any NAs before taking mean for a bucket at a timepoint
          bucket.act.both.noNA <- bucket.act.both[!is.na(bucket.act.both)]
          bucket.act.julia.noNA <- bucket.act.julia[!is.na(bucket.act.julia)]
          bucket.act.sandi.noNA <- bucket.act.sandi[!is.na(bucket.act.sandi)]

          # Take mean of that bucket for that day and insert into complete.data dataframe
          complete.act.data$mean.julia[j] <- sum(bucket.act.julia.noNA)/length(bucket.act.julia.noNA)
          complete.act.data$mean.sandi[j] <- sum(bucket.act.sandi.noNA)/length(bucket.act.sandi.noNA)
          complete.act.data$mean.both[j] <- sum(bucket.act.both.noNA)/length(bucket.act.both.noNA)
          bucket.act.julia <- black.act$Act_Mean_Julia[i]
          bucket.act.sandi <- black.act$Act_Mean_Sandi[i]
          bucket.act.both <- c(black.act$Act_Mean_Julia[i], black.act$Act_Mean_Sandi[i])
          # counter to step through inputting data into the complete.data dataframe
          j <- j + 1

        } # close else
       } # close if else
      } # close for loop

### Copper Rockfish ###

    # create empty data frames for mean bucket activity values
    Mean.Activity.cp <- matrix(NA, nrow(unique(copper.act[c("Date", "AM_PM", "Bucket")])),3)
    bucket.act.average.cp <- unique(copper.act[c("Date", "AM_PM", "Bucket", "Temperature")])
    complete.act.data.cp <- cbind(bucket.act.average.cp, Mean.Activity.cp)
    colnames(complete.act.data.cp) <- c("Date", "AM_PM", "Bucket", "Temperature",
                                        "mean.julia", "mean.sandi", "mean.both")

    # initialize the vectors that will be used in the for loop ** NOTE data in these vectors is written
    # over in each loop.. they are TEMPORARY vectors **
    bucket.act.julia.cp <- NULL
    bucket.act.sandi.cp <- NULL
    bucket.act.both.cp <- NULL

    # initialize the counter for stepping through the output matrix
    j <- 1

    # For loop to step through data frame to get activity averages
    for(i in 1:nrow(copper.act)){

      # looping through majority of data to grab all those rows that are not the last row
      # but are all those that equal the same bucket from the same day and getting activity averages
      if(i != nrow(copper.act) && i == 1 ||
         copper.act$Bucket[i] == copper.act$Bucket[i-1] && i != nrow(copper.act)){

        bucket.act.julia.cp <- c(bucket.act.julia.cp, copper.act$Act_Mean_Julia[i])
```

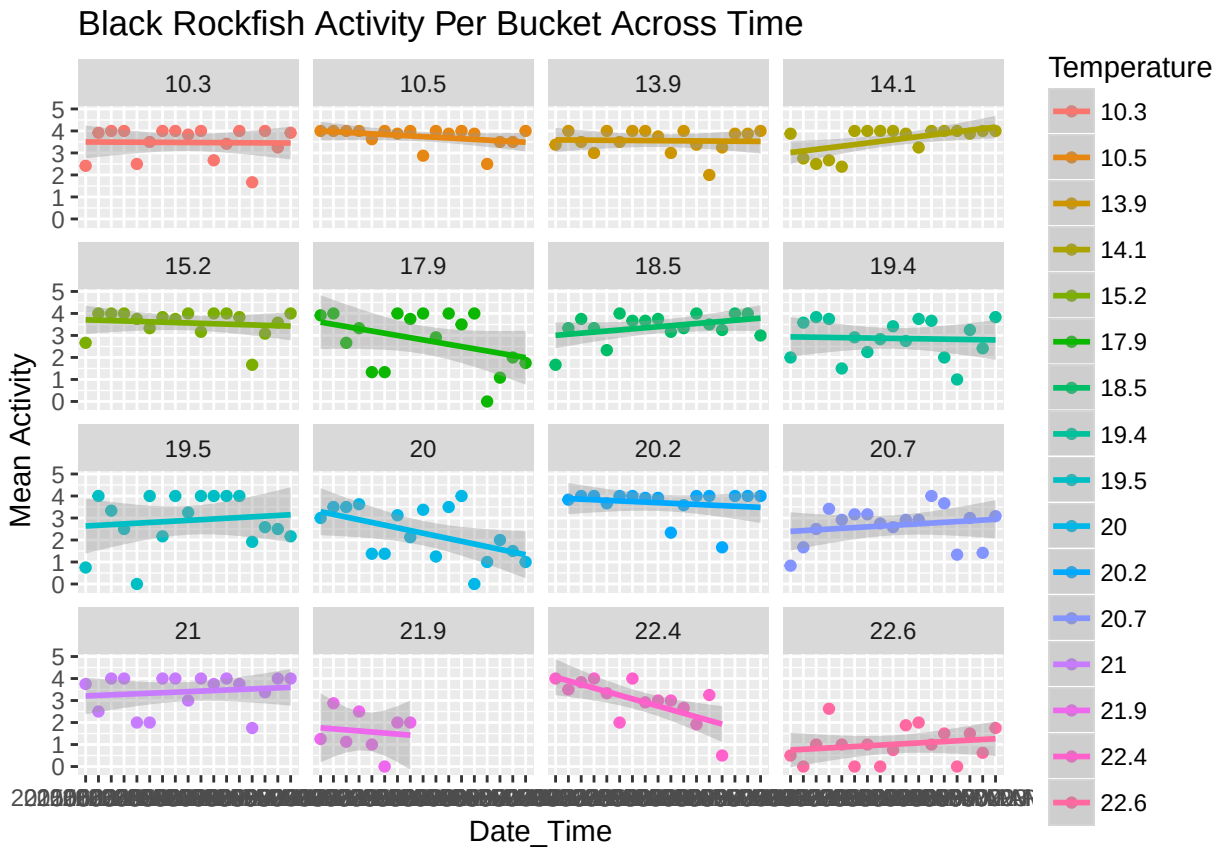
```
        bucket.act.sandi.cp <- c(bucket.act.sandi.cp, copper.act$Act_Mean_Sandi[i])
        bucket.act.both.cp <- c(bucket.act.both.cp, copper.act$Act_Mean_Julia[i], copper.act$Act_Mean_Sa

        # special case for last row
       } else {
         if(i == nrow(copper.act)){

          # Remove any NAs before taking mean for a bucket at a timepoint
          bucket.act.both.noNA.cp <- bucket.act.both.cp[!is.na(bucket.act.both.cp)]
          bucket.act.julia.noNA.cp <- bucket.act.julia.cp[!is.na(bucket.act.julia.cp)]
          bucket.act.sandi.noNA.cp <- bucket.act.sandi.cp[!is.na(bucket.act.sandi.cp)]
          complete.act.data.cp$mean.julia[j] <- sum(bucket.act.julia.noNA.cp)/length(bucket.act.julia.n
          complete.act.data.cp$mean.sandi[j] <- sum(bucket.act.sandi.noNA.cp)/length(bucket.act.sandi.n
          complete.act.data.cp$mean.both[j] <- sum(bucket.act.both.noNA.cp)/length(bucket.act.both.noNA
          j <- j + 1

        } else {

          # Remove any NAs before taking mean for a bucket at a timepoint
          bucket.act.both.noNA.cp <- bucket.act.both.cp[!is.na(bucket.act.both.cp)]
          bucket.act.julia.noNA.cp <- bucket.act.julia.cp[!is.na(bucket.act.julia.cp)]
          bucket.act.sandi.noNA.cp <- bucket.act.sandi.cp[!is.na(bucket.act.sandi.cp)]

          # Take mean of that bucket for that day and insert into complete.data dataframe
          complete.act.data.cp$mean.julia[j] <- sum(bucket.act.julia.noNA.cp)/length(bucket.act.julia.n
          complete.act.data.cp$mean.sandi[j] <- sum(bucket.act.sandi.noNA.cp)/length(bucket.act.sandi.n
          complete.act.data.cp$mean.both[j] <- sum(bucket.act.both.noNA.cp)/length(bucket.act.both.noNA
          bucket.act.julia.cp <- copper.act$Act_Mean_Julia[i]
          bucket.act.sandi.cp <- copper.act$Act_Mean_Sandi[i]
          bucket.act.both.cp <- c(copper.act$Act_Mean_Julia[i], copper.act$Act_Mean_Sandi[i])
          # counter to step through inputting data into the complete.data dataframe
          j <- j + 1

        } # close else
       } # close if else
      } # close for loop
```

Plotting

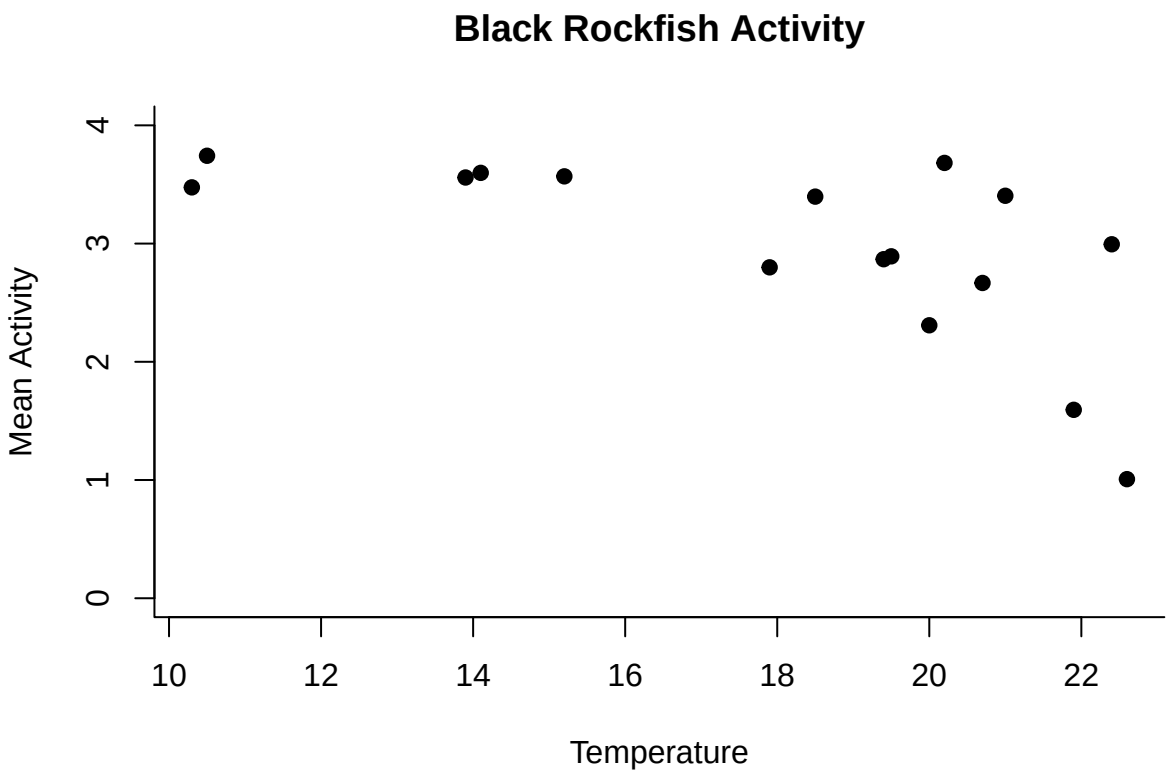
Plot activity across time & as a single value across experiment

```
### Black Rockfish ###

   complete.act.data$Date <- as.factor(complete.act.data$Date)
   complete.act.data$Date_Time <- as.factor(paste(complete.act.data$Date, complete.act.data$AM_PM))

  # Round temperature data to one significant figure after decimal place
   complete.act.data$RTemperature <- round(complete.act.data$Temperature, 1)

  # ggplot to look at activity across time across bucket treatments
   bucket.plot <- ggplot(data=complete.act.data,
```

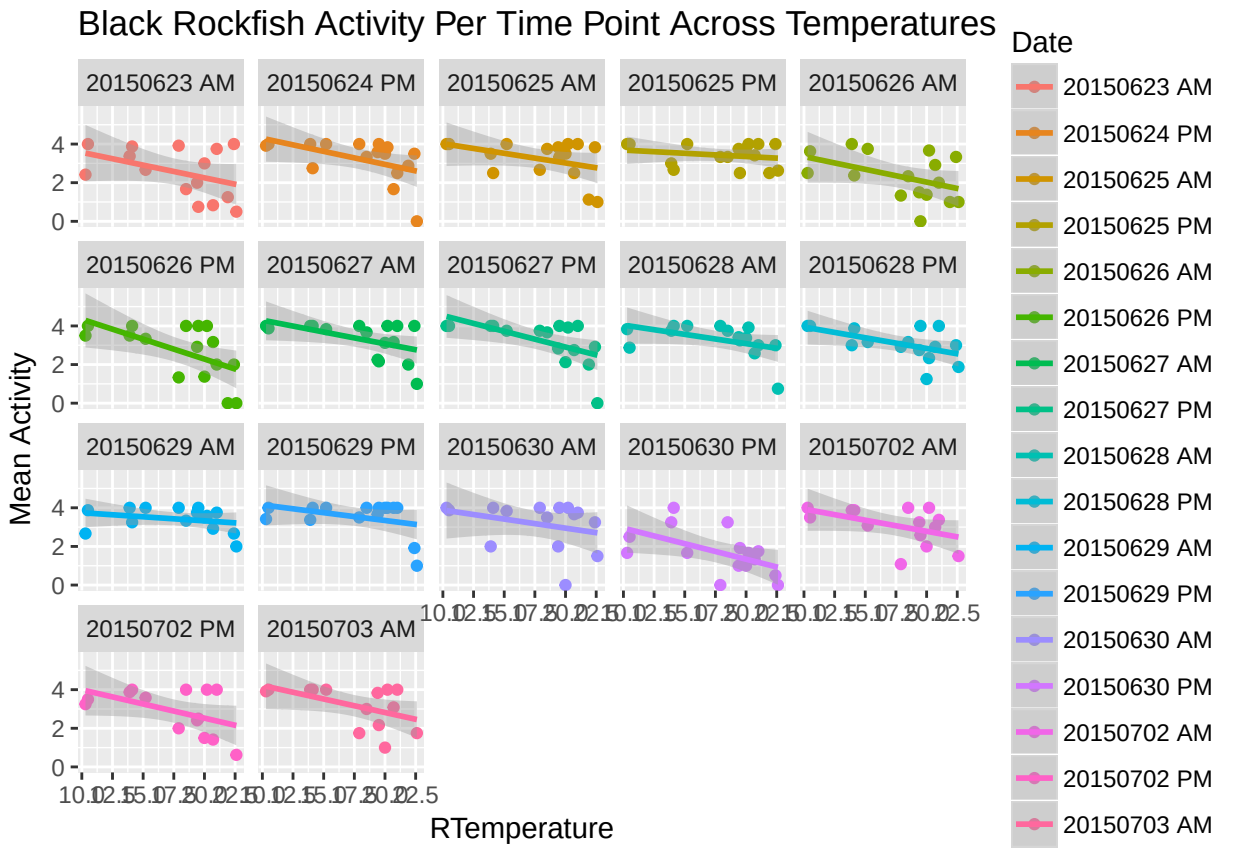
```
                        aes(x=Date_Time, y=mean.both,
                            group=as.factor(RTemperature),
                            color = as.factor(RTemperature))) + geom_point() + geom_smooth(method = "lm
bucket.plot + facet_wrap(~RTemperature) + labs(y = "Mean Activity") + scale_color_discrete(name = "Te
```



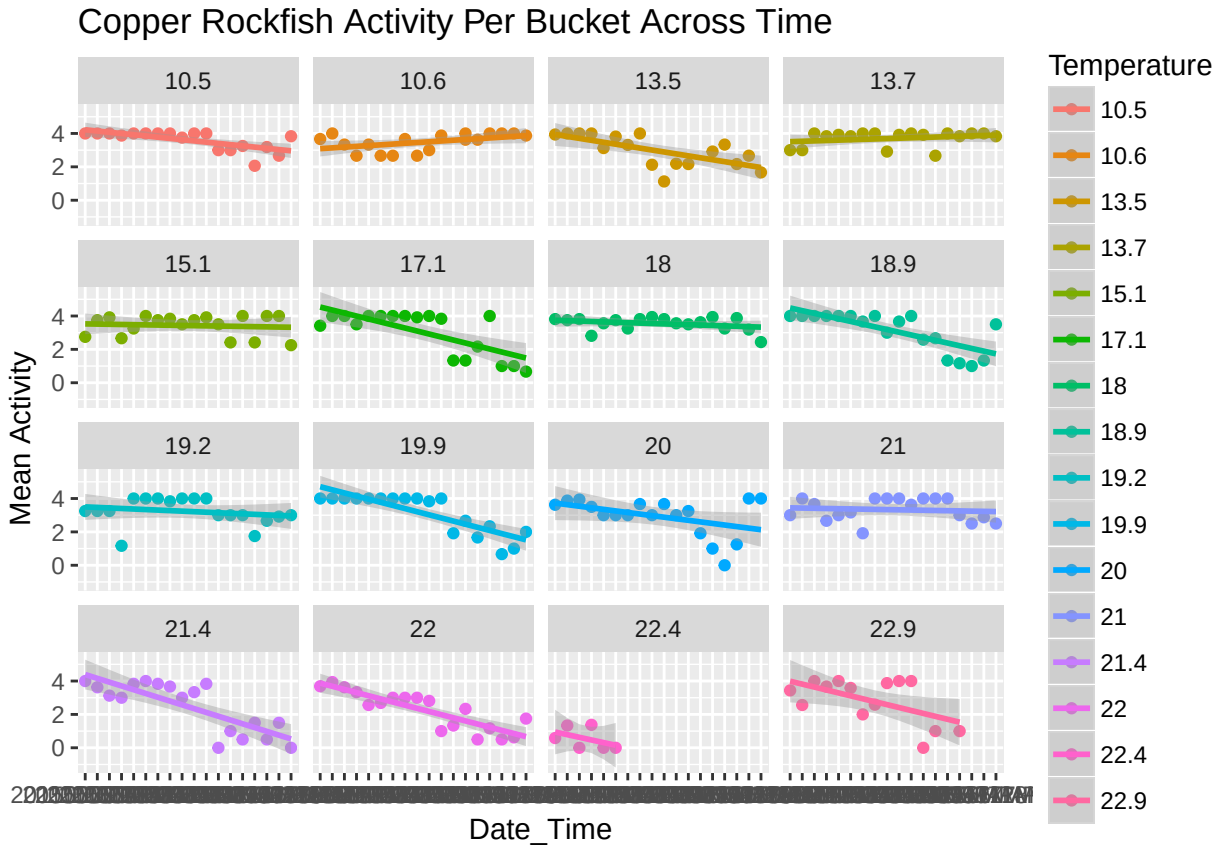

```
# Get bucket averages across entire experiement
 mean.black.activity <- aggregate(complete.act.data$mean.both,
                                  list(complete.act.data$RTemperature), mean)
 colnames(mean.black.activity) <- c("Temperature", "Mean_Activity")

# Plot averages
 plot(mean.black.activity$Temperature, mean.black.activity$Mean_Activity, ylim = c(0,4),
      xlab = "Temperature", ylab = "Mean Activity", pch = 19, bty = "l",
      main = "Black Rockfish Activity")
```



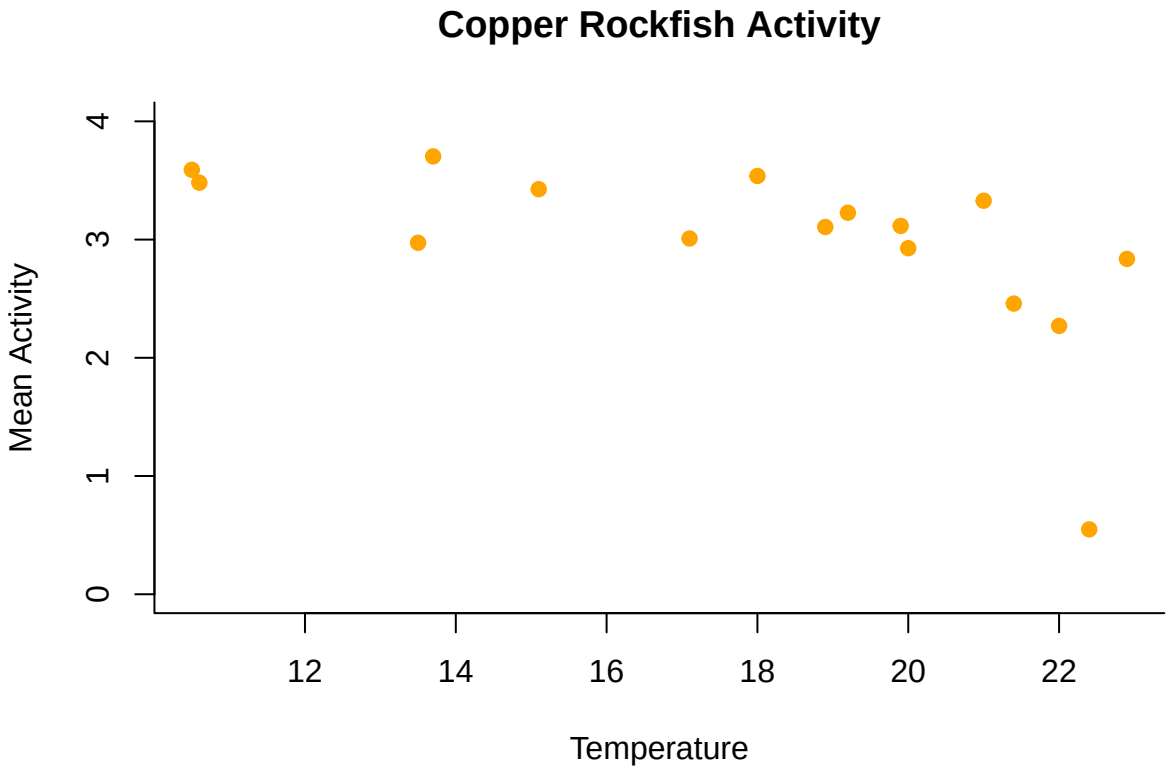

```
# Plot Activity across temperatures for each date
 day.plot <- ggplot(data=complete.act.data,
                        aes(x=RTemperature, y=mean.both,
                           group=as.factor(Date_Time),
                           color = as.factor(Date_Time))) + geom_point() + geom_smooth(method = "lm")

 day.plot + facet_wrap(~Date_Time) + labs(y = "Mean Activity") + scale_color_discrete(name = "Date") +
```



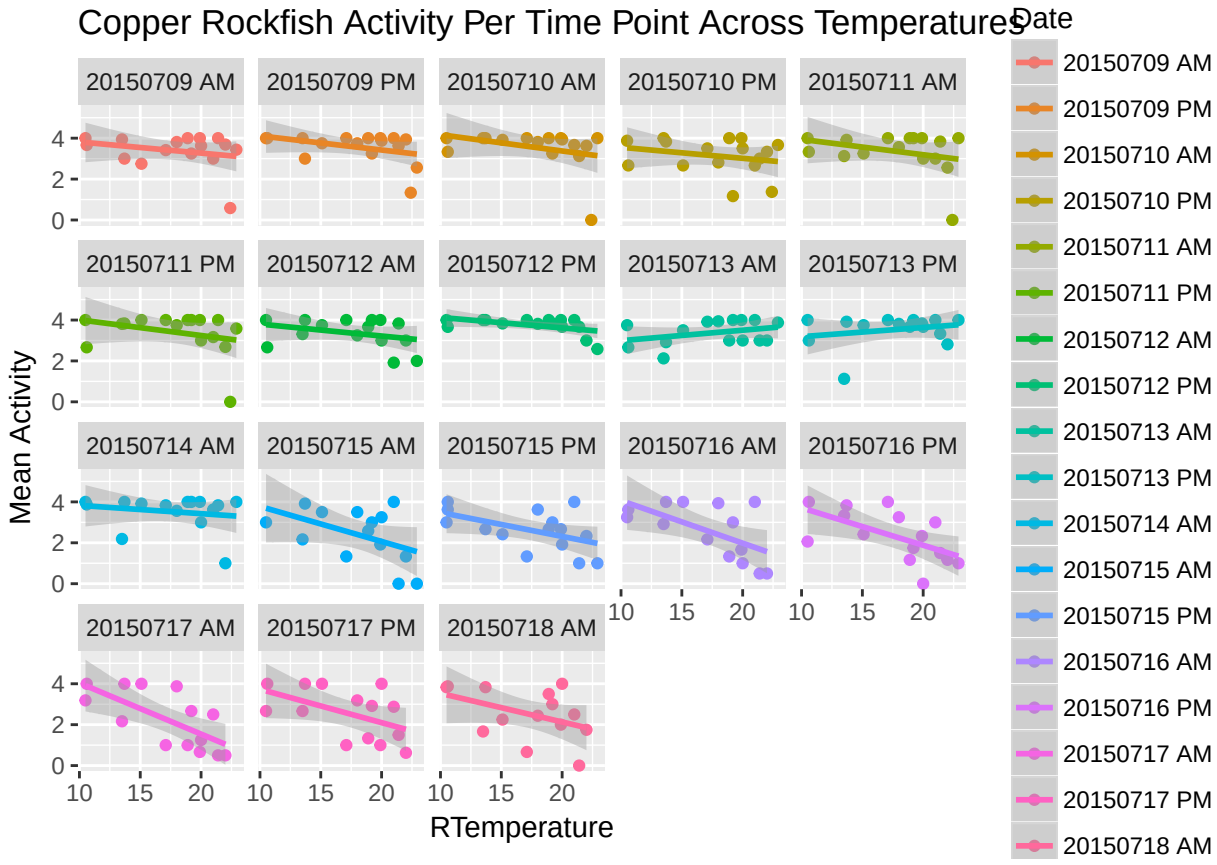

```
########################
### Copper Rockfish ###
########################

   complete.act.data.cp$Date <- as.factor(complete.act.data.cp$Date)
   complete.act.data.cp$Date_Time <- as.factor(paste(complete.act.data.cp$Date, complete.act.data$AM_PM
   complete.act.data.cp[157,8] <- "20150714 AM" #I am not sure why this single value is going in wrong

  # Round temperature data to one significant figure after decimal place
   complete.act.data.cp$RTemperature <- round(complete.act.data.cp$Temperature, 1)

  # ggplot to look at activity across time across bucket treatments
   bucket.plot.cp <- ggplot(data=complete.act.data.cp,
                            aes(x=Date_Time, y=mean.both,
                               group=as.factor(RTemperature),
                               color = as.factor(RTemperature))) + geom_point() + geom_smooth(method = "lm
  # Create Plot
   bucket.plot.cp + facet_wrap(~RTemperature) + labs(y = "Mean Activity") + scale_color_discrete(name =
```




```r
# Get bucket averages across entire experiement
 mean.copper.activity <- aggregate(complete.act.data.cp$mean.both,
                                   list(complete.act.data.cp$RTemperature), mean)
 colnames(mean.copper.activity) <- c("Temperature", "Mean_Activity")

# Plot averages
 plot(mean.copper.activity$Temperature, mean.copper.activity$Mean_Activity,
      ylim = c(0,4), xlab = "Temperature", ylab = "Mean Activity",
      pch = 19, bty = "l", col = "orange",
      main = "Copper Rockfish Activity")
```




```
# Plot Activity across temperatures for each date
 day.plot.cp <- ggplot(data=complete.act.data.cp,
                       aes(x=RTemperature, y=mean.both,
                          group=as.factor(Date_Time),
                          color = as.factor(Date_Time))) + geom_point() + geom_smooth(method = "lm")

 day.plot.cp + facet_wrap(~Date_Time) + labs(y = "Mean Activity") + scale_color_discrete(name = "Date
```



ANALYSIS

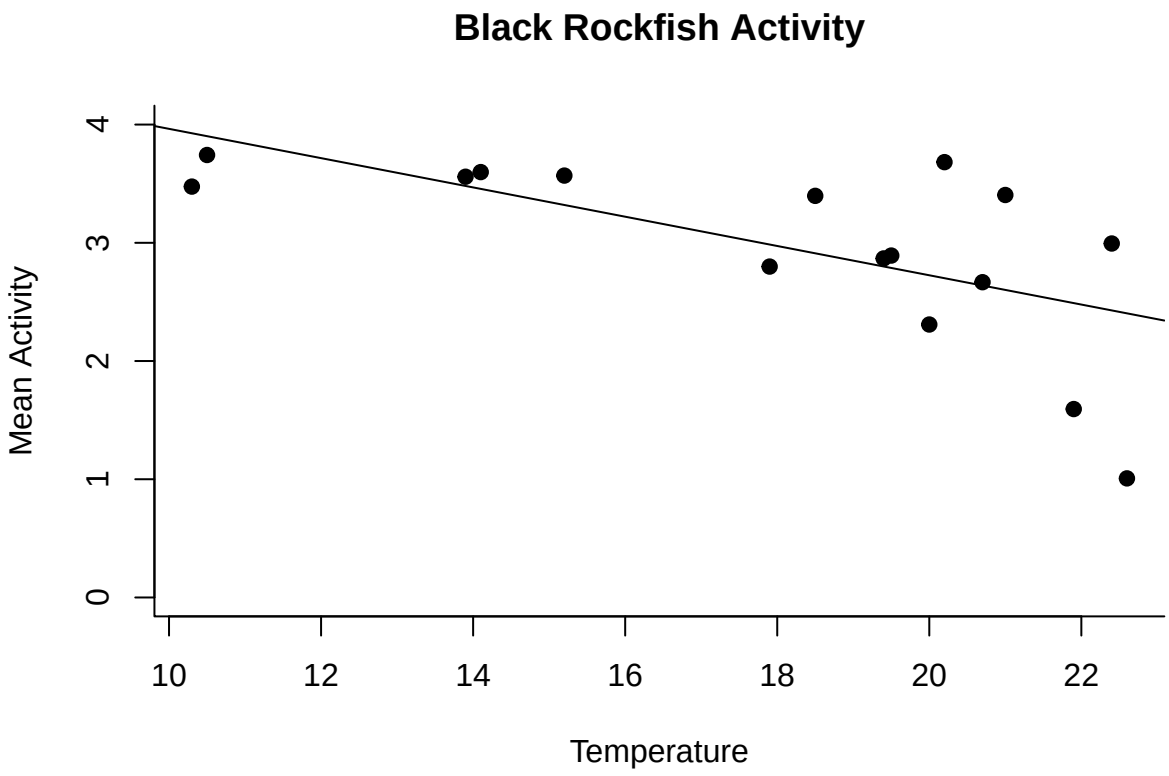
Linear

```
plot(mean.black.activity$Temperature, mean.black.activity$Mean_Activity, ylim = c(0,4),
        xlab = "Temperature", ylab = "Mean Activity", pch = 19, bty = "l",
        main = "Black Rockfish Activity")

black.act.mod1 <- lm(mean.black.activity$Mean_Activity~mean.black.activity$Temperature)
summary(black.act.mod1)
```

```
##
## Call:
## lm(formula = mean.black.activity$Mean_Activity ~ mean.black.activity$Temperature)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.39596 -0.24397  0.07294  0.30803  0.98171
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       5.20260    0.74129   7.018 6.07e-06 ***
## mean.black.activity$Temperature -0.12386    0.04025  -3.078  0.00819 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6244 on 14 degrees of freedom
## Multiple R-squared:  0.4035, Adjusted R-squared:  0.3609
## F-statistic: 9.472 on 1 and 14 DF,  p-value: 0.008187
```

```
abline(black.act.mod1)
```



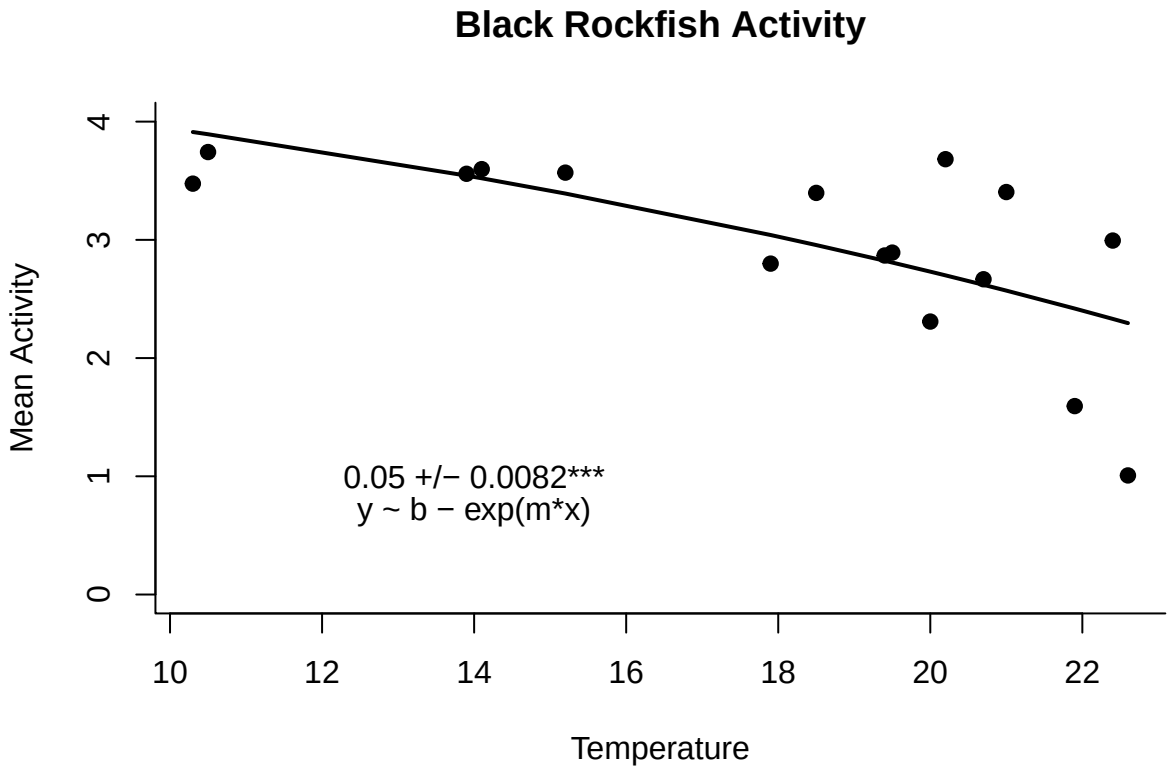
Negative Exponential

```
# Set Variables
    x <- mean.black.activity$Temperature
    y <- mean.black.activity$Mean_Activity
    b <- 4
    m <- 0.01
    #y = b - exp(m*x)

# Build Model
    m1 <- nls(y~b-exp(m*x), start = c(b = b, m = m))

# Goodness of fit
    cor(y,predict(m1))
```

```
## [1] 0.6627362
```

```
# Plot data with nls predicted fit
    plot(mean.black.activity$Temperature, mean.black.activity$Mean_Activity, ylim = c(0,4),
          xlab = "Temperature", ylab = "Mean Activity", pch = 19, bty = "l",
          main = "Black Rockfish Activity")
```

```
    lines(x,predict(m1),col="black",lty=1,lwd=2)
    text(14, 1, labels = "0.05 +/- 0.0082***")
    text(14, 0.7, labels = "y ~ b - exp(m*x)")
```




```
    summary(m1)
```

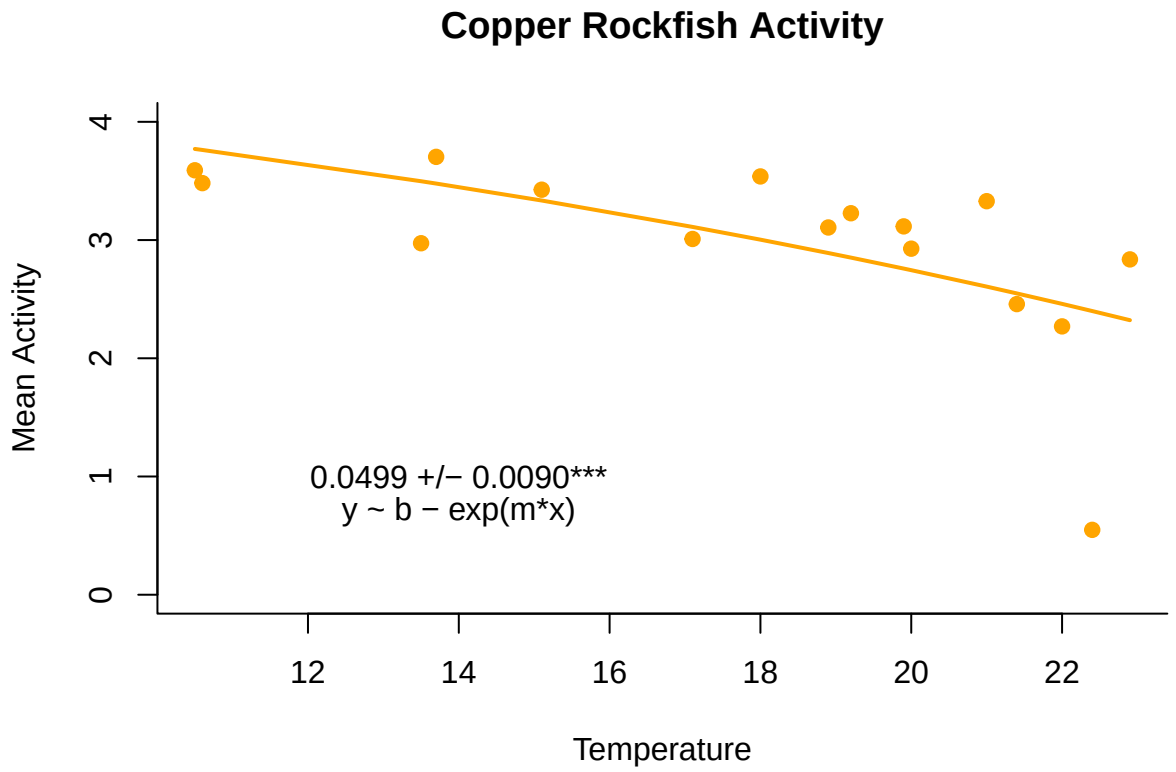
```
##
## Formula: y ~ b - exp(m * x)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## b 5.647556   0.440573  12.819 3.99e-09 ***
## m 0.053519   0.008251   6.487 1.43e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6056 on 14 degrees of freedom
##
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 3.694e-07
```

```
########################
### Copper Rockfish ###
########################

 # Set Variables
    xx <- mean.copper.activity$Temperature
    yy <- mean.copper.activity$Mean_Activity
    bb <- 3.5
    mm <- 0.01
```

```
    #y = b - exp(m*x)

 # Build Model
    m1.cp <- nls(yy~bb-exp(mm*xx), start = c(bb = bb, mm = mm))

 # Goodness of fit
    cor(yy,predict(m1.cp))
```

```
## [1] 0.6143543
```

```
 # Plot data with nls Predicted fit
     plot(mean.copper.activity$Temperature, mean.copper.activity$Mean_Activity,
        ylim = c(0,4), xlab = "Temperature", ylab = "Mean Activity",
        pch = 19, bty = "l", col = "orange",
        main = "Copper Rockfish Activity")
    lines(xx,predict(m1.cp),col="orange",lty=1,lwd=2)
    text(14, 1, labels = "0.0499 +/- 0.0090***")
    text(14, 0.7, labels = "y ~ b - exp(m*x)")
```




```
    summary(m1.cp)
```

```
##
## Formula: yy ~ bb - exp(mm * xx)
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## bb  5.45990    0.44376   12.30 6.78e-09 ***
## mm  0.04993    0.00898    5.56 7.02e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.618 on 14 degrees of freedom
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 2.449e-06
```

Quadratic