CASE STUDY 4:

Code

This study allows us to revisit/renew

1. Regression modeling
2. Properties of Least Squares/Fitting “a line”
3. Multiple observation

Datasets for this study are

1. The main file: gauge.txt
2. Supplementary large-scale files: download the following folder Full Resolution Data.zip More information about the supplementary file can be found at http://iabp.apl.washington.edu/data.html (http://iabp.apl.washington.edu/data.html) as well as http://nsidc.org/data/G00791 (http://nsidc.org/data/G00791)

Question

The aim of this lab is to provide a simple procedure for converting gain into density when the gauge is in operation. Keep in mind that the experiment was conducted by varying density and measuring the response in gain, but when the gauge is ultimately in use, the snow-pack density is to be estimated from the measured gain.

Setup

Hide

```
df <- read.table('gauge-1wb1wa6-2gpel41.txt', header=TRUE)
df <- df[order(df$density), ]  # Sort from least to greatest density
m <- 9  # Number of distinct block densities
t <- 10  # Number of replicate measurements
#install.packages('L1pack')
#install.packages('quantreg')
#install.packages('ggplot2')
library(L1pack)  # Used for least absolute deviations regression line
library(quantreg)  # Used for quantile regression line
library(ggplot2)
```

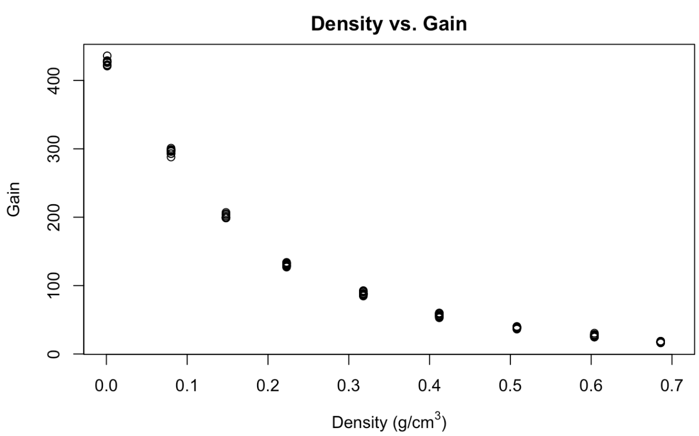
Scenario 1: Fitting

Use the data to fit the gain, or a transformation of gain, to density. Try sketching the least squares line on a scatter plot.

- Do the residuals indicate any problems with the fit?
- If the densities of the polyethylene blocks are not reported exactly, how might this affect the fit?
- What if the blocks of polyethylene were not measured in random order (location)?

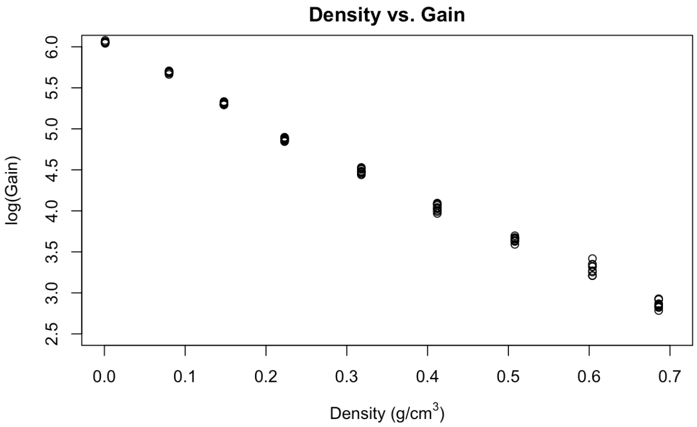
Hide

```
# Plot raw data
title <- 'Density vs. Gain'
x.axis <- expression('Density (g/cm'^3*')')
y.axis <- 'Gain'
x.range <- c(0, .7)
y.range <- c(2.5, 6)
plot(df, main=title, xlab=x.axis, ylab=y.axis, xlim=x.range)
```



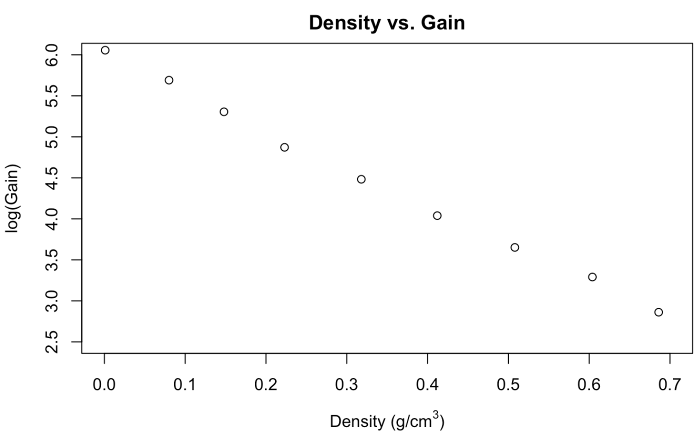
Hide

```
# Take log transformation of response variable (gain)
y.log.axis = 'log(Gain)'
df.log = data.frame(df['density'], log(df['gain']))
plot(df.log, main=title, xlab=x.axis, ylab=y.log.axis, xlim=x.range, ylim=y.range)
```



Hide

```
# Average replicate measurements
df.log.avg = aggregate(list(gain=df.log$gain), by=list(density=df.log$density), FUN=m
ean)
plot(df.log.avg, main=title, xlab=x.axis, ylab=y.log.axis, xlim=x.range, ylim=y.range
)
```



Hide

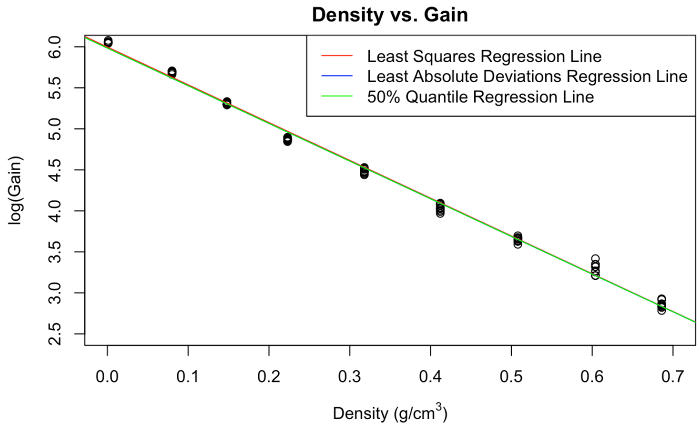
```
# Fit gain to density
least.squares <- lm(gain~density, data=df.log.avg)
lad <- lad(gain~density, data=df.log.avg)
quant <- rq(gain~density, tau=.5, data=df.log.avg)
plot(df.log, main=title, xlab=x.axis, ylab=y.log.axis, xlim=x.range, ylim=y.range)
abline(least.squares, col='red')
```

Hide

```
abline(lad, col='blue')
abline(quant, col='green')
```

Hide

```
legend('topright', legend=c('Least Squares Regression Line', 'Least Absolute Deviatio
ns Regression Line', '50% Quantile Regression Line'), col=c('red', 'blue', 'green'),
lty=1)
```



Hide

```
c(cor(df.log.avg), summary(least.squares)$r.squared)
```

```
[1]  1.0000000 -0.9984469 -0.9984469  1.0000000  0.9968963
```

Hide

```
least.squares
```

```

Call:
lm(formula = gain ~ density, data = df.log.avg)

Coefficients:
(Intercept)      density
      5.997       -4.606
```

Hide

```
lad
```

```
Call:
lad(formula = gain ~ density, data = df.log.avg)
Converged in 4 iterations

Coefficients:
 (Intercept)     density
     5.9850     -4.5935

Degrees of freedom: 9 total; 7 residual
Scale estimate: 0.06926379
```

Hide

```
quant
```

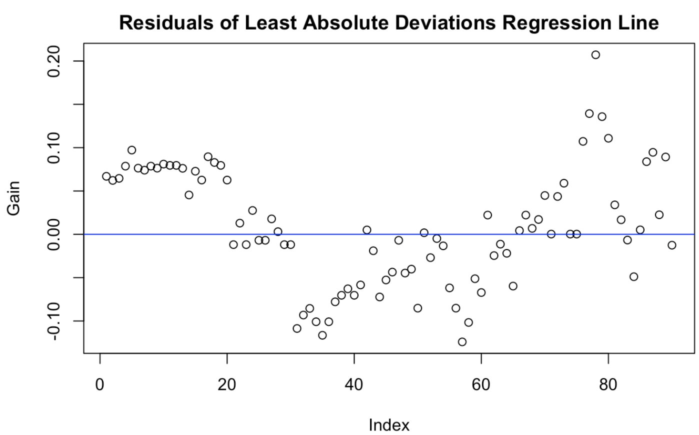
```
Call:
rq(formula = gain ~ density, tau = 0.5, data = df.log.avg)

Coefficients:
(Intercept)     density
   5.985029   -4.593460

Degrees of freedom: 9 total; 7 residual
```

Hide

```
# Check conditions for linear regression: linearity, normality of residuals, and cons
tant variability
least.squares.residuals <- data.frame(df.log['density'], df.log['gain'] - rep(predict
(least.squares), each=10))
lad.residuals <- data.frame(df.log['density'], df.log['gain'] - rep(predict(lad), eac
h=10))
quant.residuals <- data.frame(df.log['density'], df.log['gain'] - rep(predict(quant),
each=10))
title.residuals1 <- 'Residuals of Least Squares Regression Line'
title.residuals2 <- 'Residuals of Least Absolute Deviations Regression Line'
title.residuals3 <- 'Residuals of 50% Quantile Regression Line'
plot(least.squares.residuals$gain, main=title.residuals1, ylab=y.axis)
abline(0, 0, col='red')
```

Hide

```
plot(lad.residuals$gain, main=title.residuals2, ylab=y.axis)
abline(0, 0, col='blue')
```

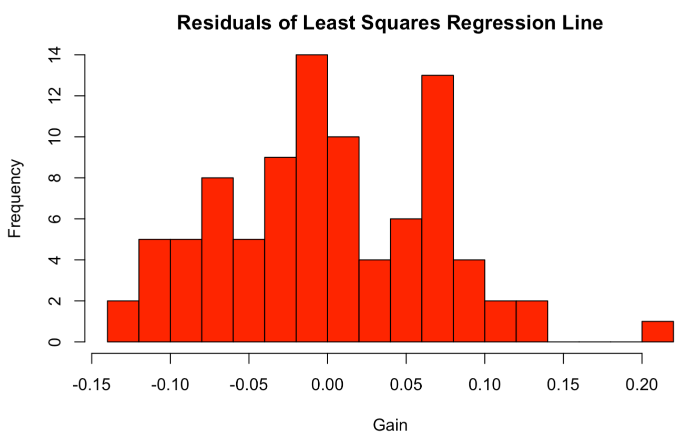


Hide

```
plot(quant.residuals$gain, main=title.residuals3, ylab=y.axis)
abline(0, 0, col='green')
```

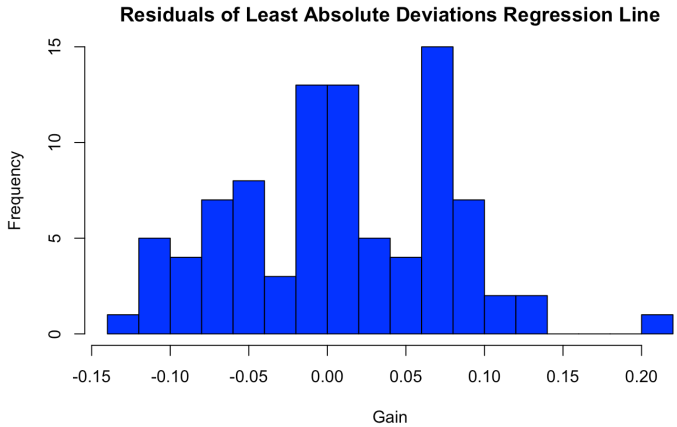
Hide

```
num.bins <- 12
hist(least.squares.residuals$gain, breaks=num.bins, main=title.residuals1, xlab=y.axi
s, col='red')
```



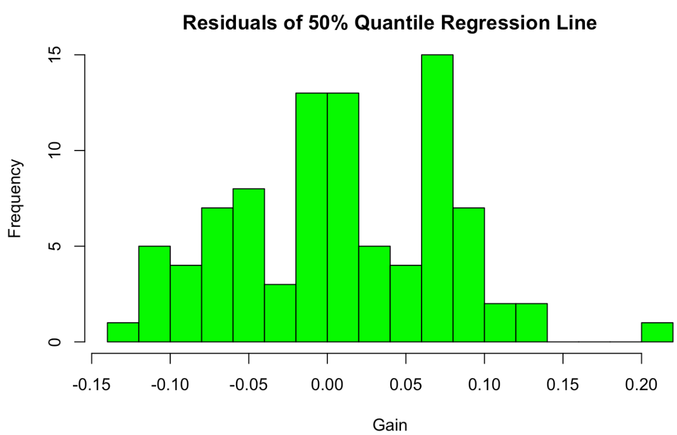
Hide

```
hist(lad.residuals$gain, breaks=num.bins, main=title.residuals2, xlab=y.axis, col='bl
ue')
```



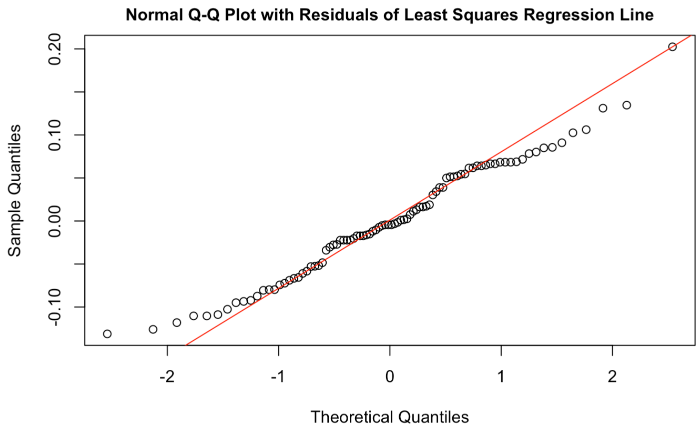
Hide

```
hist(quant.residuals$gain, breaks=num.bins, main=title.residuals3, xlab=y.axis, col='
green')
```



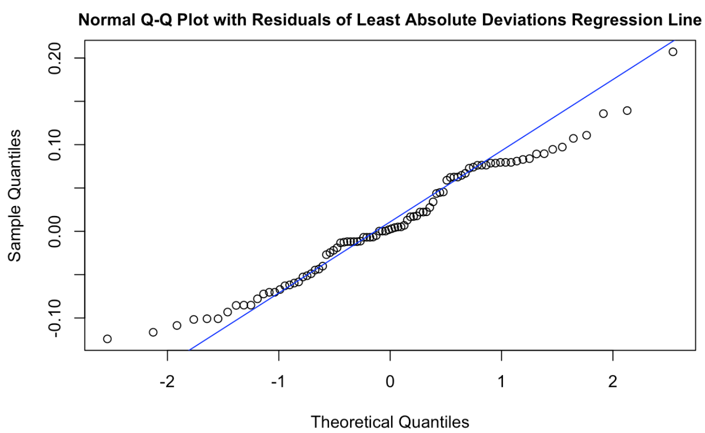
Hide

```
qqnorm(least.squares.residuals$gain, main=paste('Normal Q-Q Plot with', title.residua
ls1), cex.main=1)
qqline(least.squares.residuals$gain, col='red')
```



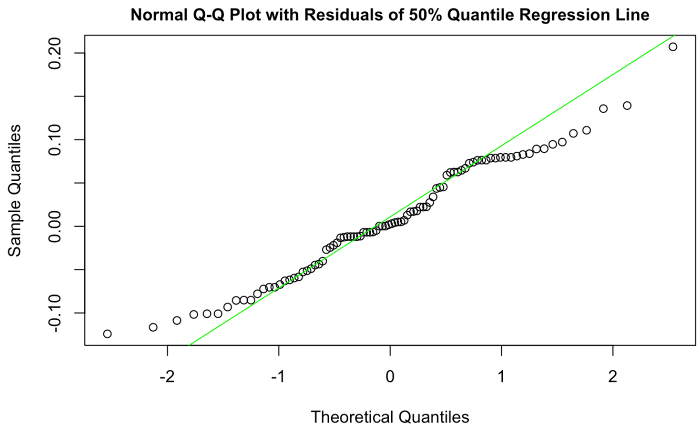
Hide

```
qqnorm(lad.residuals$gain, main=paste('Normal Q-Q Plot with', title.residuals2), cex.
main=1)
qqline(lad.residuals$gain, col='blue')
```



Hide

```
qqnorm(quant.residuals$gain, main=paste('Normal Q-Q Plot with', title.residuals3), ce
x.main=1)
qqline(quant.residuals$gain, col='green')
```



Scenario 2: Predicting

Ultimately we are interested in answering questions such as: Given a gain reading of 38.6, what is the density of the snow-pack? Given a gain reading of 426.7, what is the density of the snow-pack? These two numeric values, 38.6 and 426.7, were chosen because they are the average gains for the 0.508 and 0.001 densities, respectively.

- Develop a procedure for adding bands around your least squares line that can be used to make interval estimates for the snow-pack density from gain measurements. Keep in mind how the data were collected: several measurements of gain were taken for polyenythylene blocks of known density.

Hide

```
# Predictions
PredictLogGain <- function(density)
  predict(least.squares, data.frame(density=density))  # Predict log(gain) using dens
ity
PredictDensityLeastSquares <- function(gain) {
  intercept <- coef(least.squares)[[1]]
  slope <- coef(least.squares)[[2]]
  (log(gain) - intercept) / slope  # Predict density using gain
}
PredictDensityLad <- function(gain) {
  intercept <- coef(lad)[[1]]
  slope <- coef(lad)[[2]]

  (log(gain) - intercept) / slope  # Predict density using gain
}
PredictDensityQuant <- function(gain) {
  intercept <- coef(quant)[[1]]
  slope <- coef(quant)[[2]]
  (log(gain) - intercept) / slope  # Predict density using gain
}
# 95% prediction and confidence intervals of log(gain) using density
t <- qt(.975, df=m-2)
mean.density <- mean(df.log.avg$density)
summation <- sum((df.log.avg$density - mean.density) ^ 2)
s2 <- aggregate(list(variance=least.squares.residuals$gain), by=list(density=least.sq
uares.residuals$density), FUN=var)
s.pooled <- sqrt(mean(s2$variance))
center.expr <- quote(center <- PredictLogGain(density))
ci.width.expr <- quote(width <- t * s.pooled * sqrt(1/m + (density-mean.density)^2 /
summation))
pi.width.expr <- quote(width <- t * s.pooled * sqrt(1 + 1/m + (density-mean.density)^
2 / summation))
LogGainCiLower <- function(density) {
  eval(center.expr)
  eval(ci.width.expr)
  center - width
}
LogGainCiUpper <- function(density) {
  eval(center.expr)
  eval(ci.width.expr)
  center + width
}
LogGainPiLower <- function(density) {
  eval(center.expr)
  eval(pi.width.expr)
  center - width
}
LogGainPiUpper <- function(density) {
  eval(center.expr)
  eval(pi.width.expr)
  center + width
}
# Add bands around least squares line
plot(df.log, main=title, xlab=x.axis, ylab=y.log.axis, xlim=x.range, ylim=y.range)
abline(least.squares, col='red')
```

Hide

```
ci.col <- 'purple'
pi.col <- 'blue'
symbol <- '-'
size <- 1.5
line.type <- 3
line.width <- 0.7
confidence.intervals <- data.frame(density=df.log.avg$density, lower=LogGainCiLower(d
f.log.avg$density), upper=LogGainCiUpper(df.log.avg$density))
points(x=confidence.intervals$density, y=confidence.intervals$lower, col=ci.col, pch=
symbol, cex=size)
points(x=confidence.intervals$density, y=confidence.intervals$upper, col=ci.col, pch=
symbol, cex=size)
```

Hide

```
lines(x=confidence.intervals$density, y=confidence.intervals$lower, col=ci.col, lty=l
ine.type, lwd=line.width)
lines(x=confidence.intervals$density, y=confidence.intervals$upper, col=ci.col, lty=l
ine.type, lwd=line.width)
```

Hide

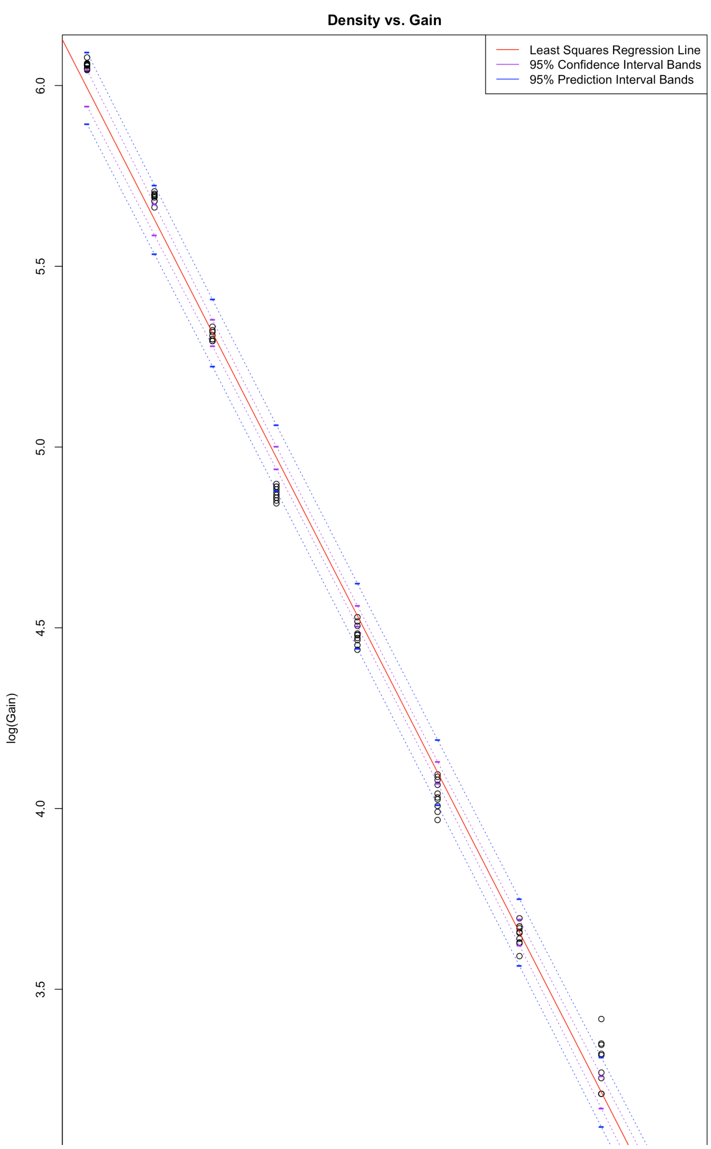
```
prediction.intervals <- data.frame(density=df.log.avg$density, lower=LogGainPiLower(d
f.log.avg$density), upper=LogGainPiUpper(df.log.avg$density))
points(x=prediction.intervals$density, y=prediction.intervals$lower, col=pi.col, pch=
symbol, cex=size)
points(x=prediction.intervals$density, y=prediction.intervals$upper, col=pi.col, pch=
symbol, cex=size)
```

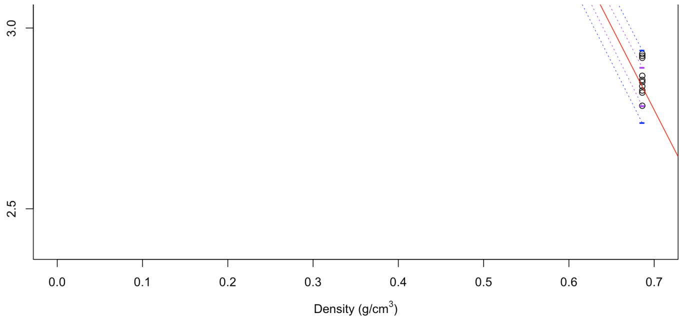
Hide

```
lines(x=prediction.intervals$density, y=prediction.intervals$lower, col=pi.col, lty=l
ine.type, lwd=line.width)
lines(x=prediction.intervals$density, y=prediction.intervals$upper, col=pi.col, lty=l
ine.type, lwd=line.width)
```

Hide

```
legend('topright', legend=c('Least Squares Regression Line', '95% Confidence Interval
Bands', '95% Prediction Interval Bands'), col=c('red', ci.col, pi.col), lty=1)
```





Hide

```
# 95% prediction and confidence intervals of density using gain
end.points <- c(-1, 3)  # Interval to search the root in
DensityCi <- function(gain) {
  lower <- uniroot(function(density) log(gain) - LogGainCiLower(density), interval=en
d.points)[[1]]
  upper <- uniroot(function(density) log(gain) - LogGainCiUpper(density), interval=en
d.points)[[1]]
  c(lower, upper)
}
DensityPi <- function(gain) {
  lower <- uniroot(function(density) log(gain) - LogGainPiLower(density), end.points)
[[1]]
  upper <- uniroot(function(density) log(gain) - LogGainPiUpper(density), end.points)
[[1]]
  c(lower, upper)
}
# Point and interval estimates for example gain readings
PredictDensityLeastSquares(38.6)  # 38.6 is the average gain for 0.508 density
```

```
[1] 0.5089113
```

Hide

```
PredictDensityLad(38.6)
```

```
[1] 0.5076298
```

Hide

```
PredictDensityQuant(38.6)
```

```
[1] 0.5076298
```

Hide

```
DensityCi(38.6)
```

```
[1] 0.5011879 0.5169000
```

Hide

```
DensityPi(38.6)
```

```
[1] 0.4889568 0.5291323
```

Hide

```
PredictDensityLeastSquares(426.7)  # 426.7 is the average gain for 0.001 density
```

```
[1] -0.01276954
```

Hide

```
PredictDensityLad(426.7)
```

```
[1] -0.01546807
```

Hide

```
PredictDensityQuant(426.7)
```

```
[1] -0.01546811
```

Hide

```
DensityCi(426.7)
```

```
[1] -0.024285866 -0.001769193
```

Hide

```
DensityPi(426.7)
```

```
[1] -0.034644666  0.008618629
```

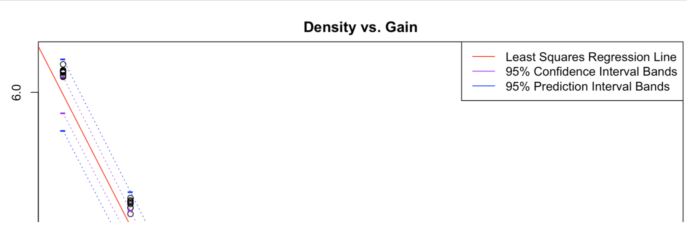
Scenario 3: Cross-Validation

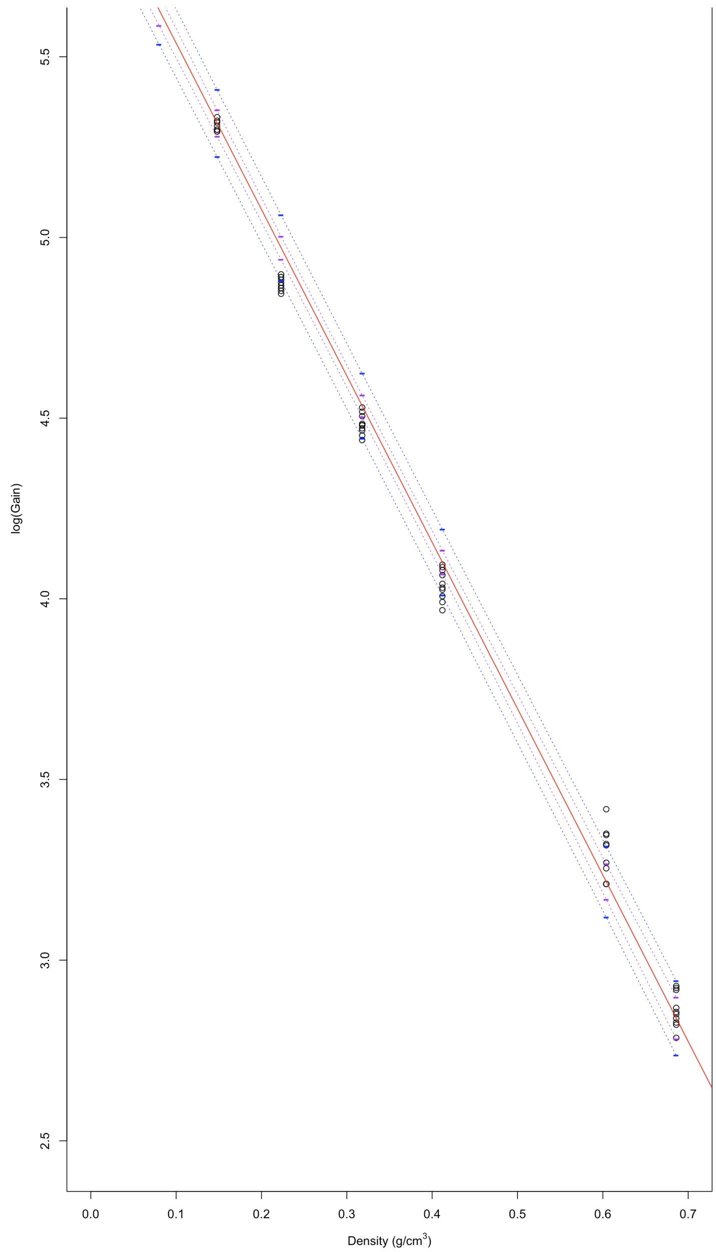
To check how well your procedure works, omit the set of measurements corresponding to the block of density 0.508, apply your “estimation”/calibration procedure to the remaining data, and provide an interval estimate for the density of a block with an average reading of 38.6. Where does the actual density fall in the interval? Try the same test, for the set of measurements at the 0.001 density.

Hide

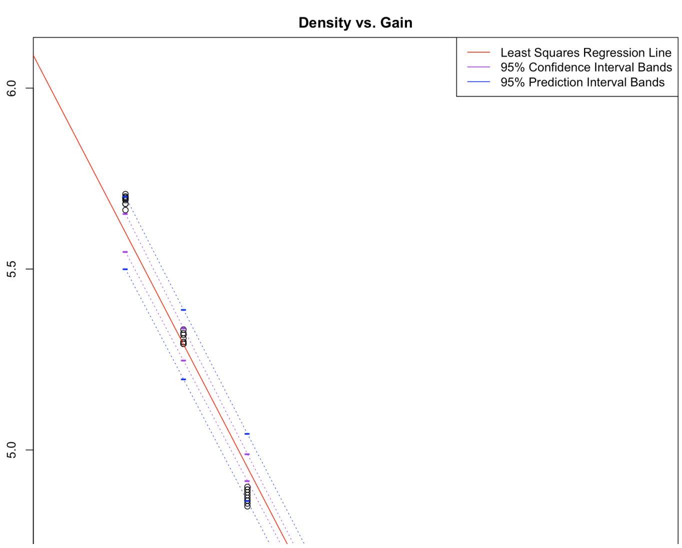
```
for (omitted in c(0.508, 0.001)) {
  # Omit measurements corresponding to the specified density
  df.log.omitted = df.log[which(df.log['density'] != omitted), ]
  df.log.avg.omitted <- df.log.avg[which(df.log.avg['density'] != omitted), ]


  # Redo calculations using modified dataset
  least.squares <- lm(gain~density, data=df.log.avg.omitted)

  mean.density <- mean(df.log.avg.omitted$density)
  summation <- sum((df.log.avg.omitted$density - mean.density) ^ 2)

  s2 <- aggregate(list(variance=least.squares.residuals$gain), by=list(density=least.
squares.residuals$density), FUN=var)
  s.pooled <- sqrt(mean(s2$variance))

  ci.width.expr <- quote(width <- t * s.pooled * sqrt(1/(m-1) + (density-mean.density
)^2 / summation))
  pi.width.expr <- quote(width <- t * s.pooled * sqrt(1 + 1/(m-1) + (density-mean.den
sity)^2 / summation))

  plot(df.log.omitted, main=title, xlab=x.axis, ylab=y.log.axis, xlim=x.range, ylim=y
.range)
  abline(least.squares, col='red')

  ci.col <- 'purple'
  pi.col <- 'blue'
  symbol <- '-'
  size <- 1.5
  line.type <- 3
  line.width <- 0.7

  confidence.intervals <- data.frame(density=df.log.avg.omitted$density, lower=LogGai
nCiLower(df.log.avg.omitted$density), upper=LogGainCiUpper(df.log.avg.omitted$density
))
  points(x=confidence.intervals$density, y=confidence.intervals$lower, col=ci.col, pc
h=symbol, cex=size)
  points(x=confidence.intervals$density, y=confidence.intervals$upper, col=ci.col, pc
h=symbol, cex=size)
  lines(x=confidence.intervals$density, y=confidence.intervals$lower, col=ci.col, lty
=line.type, lwd=line.width)
  lines(x=confidence.intervals$density, y=confidence.intervals$upper, col=ci.col, lty
=line.type, lwd=line.width)

  prediction.intervals <- data.frame(density=df.log.avg.omitted$density, lower=LogGai
nPiLower(df.log.avg.omitted$density), upper=LogGainPiUpper(df.log.avg.omitted$density
))
  points(x=prediction.intervals$density, y=prediction.intervals$lower, col=pi.col, pc
h=symbol, cex=size)
  points(x=prediction.intervals$density, y=prediction.intervals$upper, col=pi.col, pc
h=symbol, cex=size)
  lines(x=prediction.intervals$density, y=prediction.intervals$lower, col=pi.col, lty
=line.type, lwd=line.width)
  lines(x=prediction.intervals$density, y=prediction.intervals$upper, col=pi.col, lty
=line.type, lwd=line.width)

  legend('topright', legend=c('Least Squares Regression Line', '95% Confidence Interv
al Bands', '95% Prediction Interval Bands'), col=c('red', ci.col, pi.col), lty=1)

  print(PredictDensityLeastSquares(38.6))  # 38.6 is the average gain for 0.508 densi
ty
  print(DensityCi(38.6))
  print(DensityPi(38.6))

  print(PredictDensityLeastSquares(426.7))  # 426.7 is the average gain for 0.001 den
sity
  print(DensityCi(426.7))
  print(DensityPi(426.7))
}
```

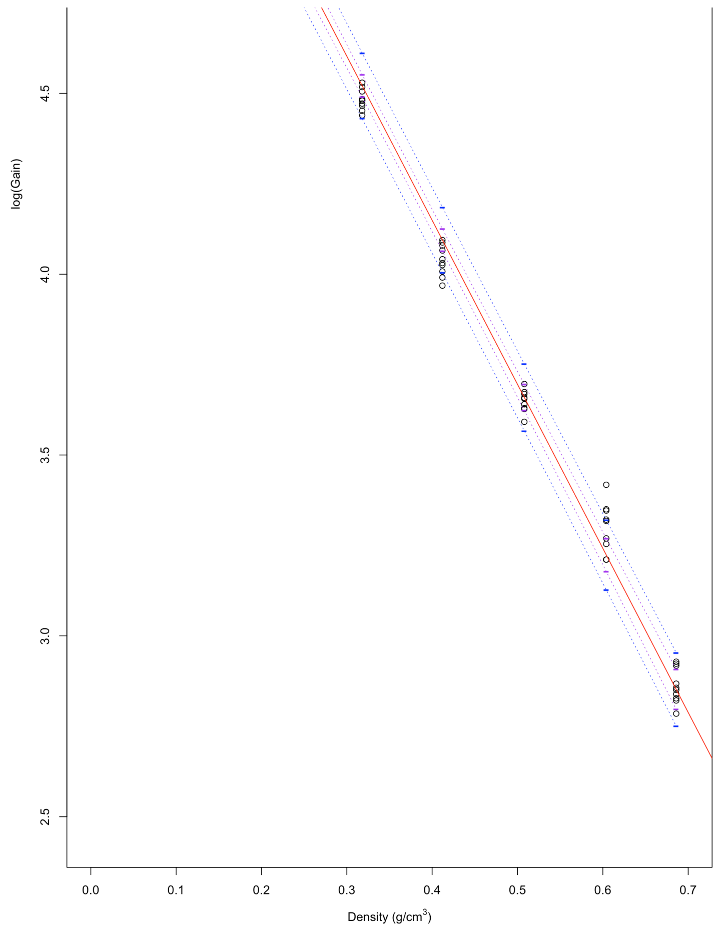
```
[1] 0.5091927
[1] 0.5006695 0.5180406
[1] 0.4889184 0.5297925
[1] -0.0128045
[1] -0.024342164 -0.001790802
[1] -0.034760736  0.008598777
```






```
[1] 0.5092919
[1] 0.5014370 0.5174323
[1] 0.4890257 0.5298473
[1] -0.02051733
[1] -0.035344991 -0.006521825
[1] -0.044604850  0.002738127
```





Additional Scenario: Temperature, DOY, and Latitude.

Use the additional dataset to construct a model fitting temperature with DOY, latitude, and other reasonable features. Try sketching the least squares line on a scatter plot. We aim to investigate the relationship between temperature and the DOY, and its latitude.

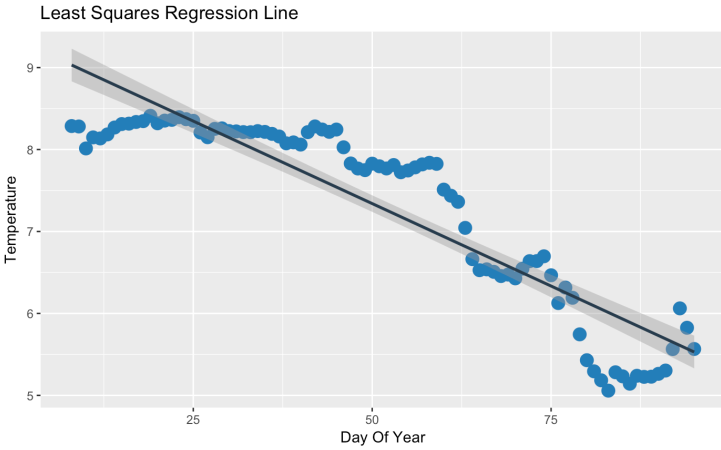
Hide

```
# Check the correlation
data <- read.csv('Full Resolution Data/64506420.csv', header=TRUE)
data <- data[,c('Hour','DOY','POS_DOY','Lat','Lon','Ts','BP')]
# Drop the extreme outlier case
#data <- data[which(data$Ts>-200),]
data_matrix <- as.matrix(data)
# Correlation Matrix
corr_matrix <- cor(data_matrix)
corr_matrix
```

```
                Hour          DOY       POS_DOY          Lat          Lon           Ts
BP
Hour     1.000000000 -0.006779489 -0.006775002 -0.007511024  0.01217860  0.008592576
0.02505819
DOY     -0.006779489  1.000000000  0.999999958  0.903704617 -0.68012490 -0.906918658
-0.17232397
POS_DOY -0.006775002  0.999999958  1.000000000  0.903696909 -0.68013445 -0.906916964
-0.17230481
Lat     -0.007511024  0.903704617  0.903696909  1.000000000 -0.59748962 -0.958835395
-0.29666821
Lon      0.012178596 -0.680124899 -0.680134450 -0.597489620  1.00000000  0.564136723
-0.02981279
Ts       0.008592576 -0.906918658 -0.906916964 -0.958835395  0.56413672  1.000000000
0.20223136
BP       0.025058188 -0.172323969 -0.172304805 -0.296668215 -0.02981279  0.202231363
1.00000000
```

Hide

```
# Group by DOY and average replicated measurements
data$DOY <- round(data$DOY,0)
data.avg = aggregate(list(data=data[,c('Ts','Lat')]), by=list(DOY=data$DOY), FUN=mean
)
# least squares line
ggplot(data.avg,aes(x=data.avg$DOY, y=data.avg$data.Ts)) +
  geom_point(color='#2980B9', size = 4) +
  geom_smooth(method=lm, color='#2C3E50') +ggtitle(label ="Least Squares Regression L
ine") + xlab("Day Of Year") +
  ylab("Temperature")
```



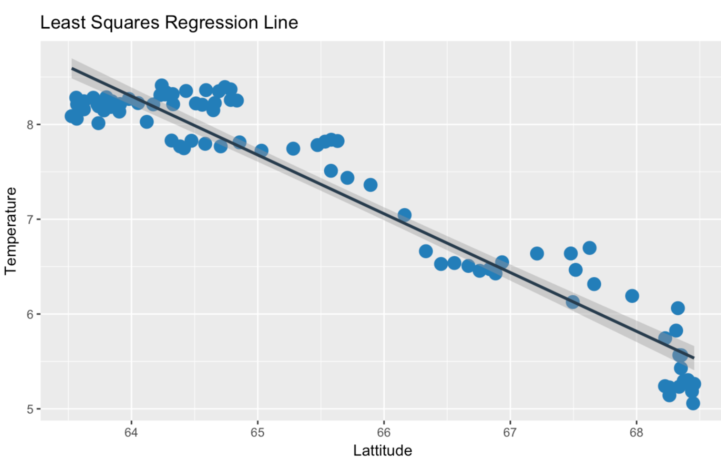
Hide

```
fit1<-lm(formula = data.Ts ~ DOY, data = data.avg)
summary(fit1)
```

```

Call:
lm(formula = data.Ts ~ DOY, data = data.avg)

Residuals:
     Min       1Q   Median       3Q      Max
-0.95439 -0.34633  0.03139  0.35247  0.84693

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.353218   0.114242   81.87   <2e-16 ***
DOY         -0.040253   0.001989  -20.23   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4741 on 86 degrees of freedom
Multiple R-squared:  0.8264,    Adjusted R-squared:  0.8244
F-statistic: 409.4 on 1 and 86 DF,  p-value: < 2.2e-16
```

Hide

```
ggplot(data.avg,aes(x=data.avg$data.Lat, y=data.avg$data.Ts)) +
  geom_point(color='#2980B9', size = 4) +
  geom_smooth(method=lm, color='#2C3E50') +ggtitle(label ="Least Squares Regression L
ine")+ xlab("Lattitude") +
  ylab("Temperature")
```



Hide

```
fit2<-lm(formula = data.Ts ~ data.Lat, data = data.avg)
summary(fit2)
```

```
Call:
lm(formula = data.Ts ~ data.Lat, data = data.avg)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.51377 -0.26924 -0.04201  0.24403  0.64902 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  47.9901     1.2672   37.87   <2e-16 ***
data.Lat     -0.6202     0.0193  -32.14   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3155 on 86 degrees of freedom
Multiple R-squared:  0.9231,	Adjusted R-squared:  0.9222 
F-statistic:  1033 on 1 and 86 DF,  p-value: < 2.2e-16
```

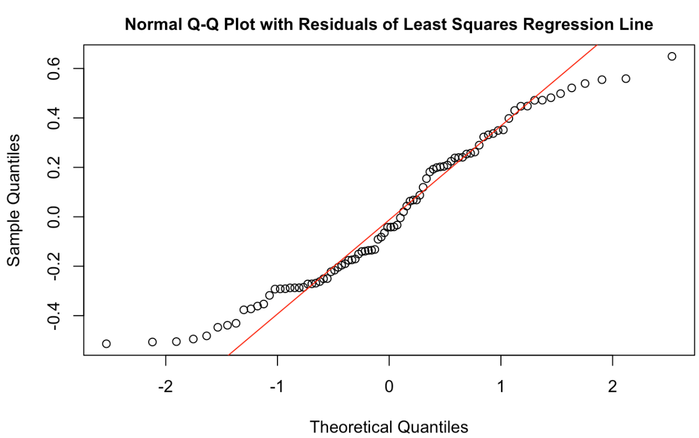
Hide

```
# Polynomial Regression Line
fit3<-lm(formula = data.Ts ~ DOY + data.Lat, data = data.avg)
summary(fit3)
```

```
Call:
lm(formula = data.Ts ~ DOY + data.Lat, data = data.avg)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.60721 -0.22496 -0.00537  0.25822  0.61356 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 40.049638   2.673979  14.978  < 2e-16 ***
DOY         -0.009756   0.002936  -3.322  0.00132 ** 
data.Lat    -0.491566   0.042805 -11.484  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2985 on 85 degrees of freedom
Multiple R-squared:  0.932,	Adjusted R-squared:  0.9304 
F-statistic: 582.1 on 2 and 85 DF,  p-value: < 2.2e-16
```

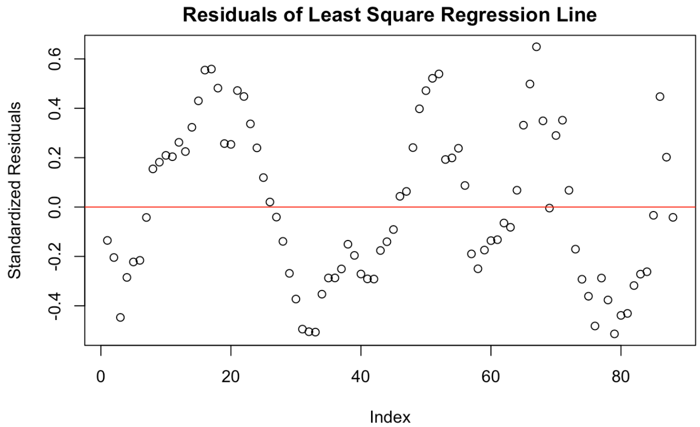
Hide

```
qqnorm(fit2$residuals, main=paste('Normal Q-Q Plot with', title.residuals1), cex.main
=1)
qqline(fit2$residuals, col='red')
```



Hide

```
title.residuals1 <- 'Residuals of Least Square Regression Line'
plot(fit2$residuals, main=title.residuals1, ylab = "Standardized Residuals")
abline(0, 0, col='red')
```



Residuals of Least Square Regression Line
Standardized Residuals
-0.4
-0.2
0.0
0.2
0.4
0.6
0
20
40
60
80
Index