HW4 Code

Group Member:

- Yunlin Tang a14664383
- Yong Liu a15126460
- Jian Jiao a14525939

Set Up

In [ ]:

```
```

In [1]:

```
gauge <- read.table('gauge.txt',header=TRUE)
gain <- gauge$gain
density <- gauge$density
```

In [ ]:

```
```

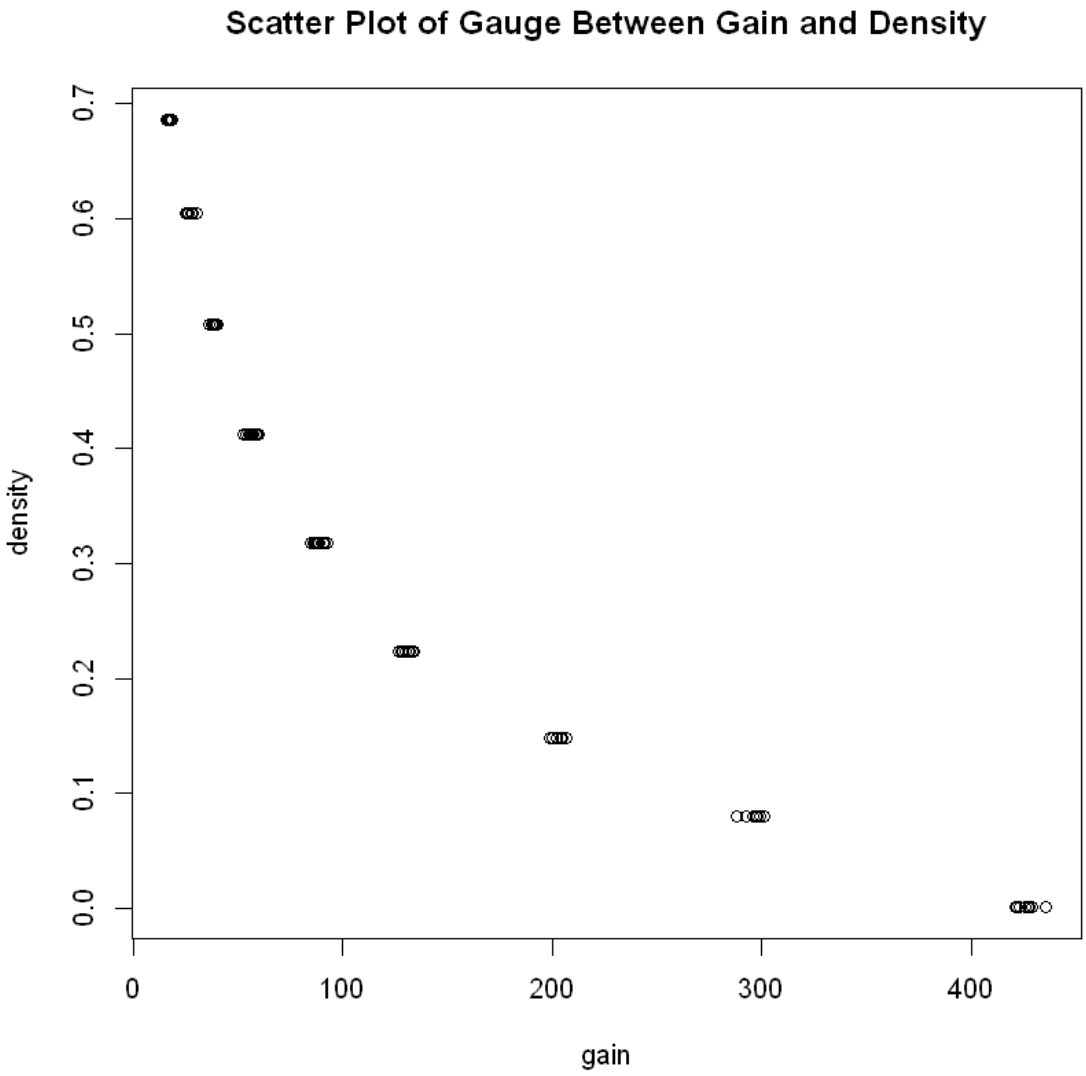
Question 1 (Fitting)

First fit a regression line to your data and plot the fit. Next, examine the residual plot and discuss. If you determine a transformation is necessary, determine the appropriate transformation and fit the model to the transformed data again plotting the fit. Be sure to justify the final model you use and its interpretation in the relevant context. The justification may be based on the theory presented in lecture.

Do not forget to explore how the fit could be affected if there was error in the density measurements.

In [2]:

```
# first plot the scatter plot
plot(x=gain, y=density, xlab='gain', ylab='density',
    main='Scatter Plot of Gauge Between Gain and Density')
```



In [3]:

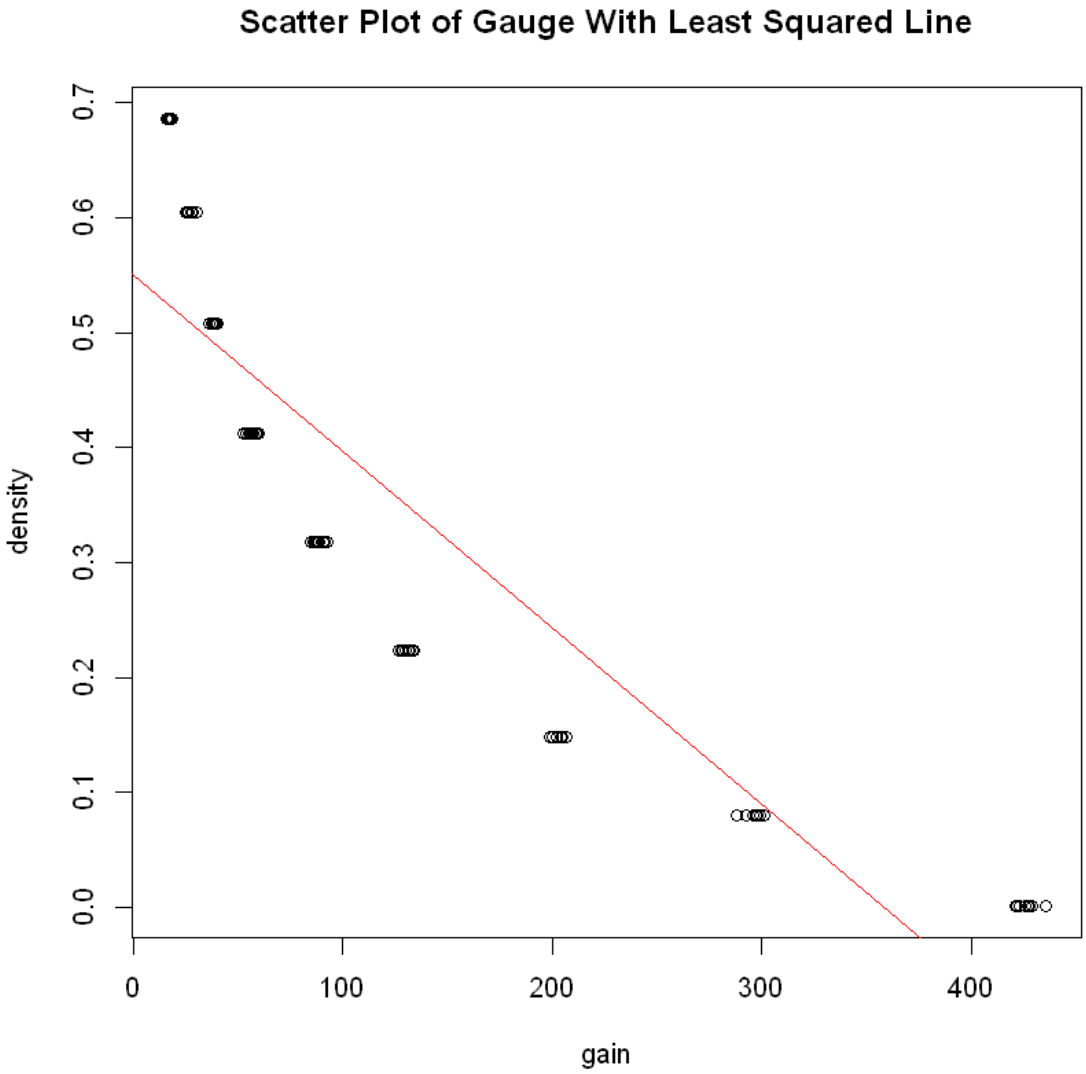
```
# fit the least squares line of the gauge data
fit <- lm(formula=density~gain)
fit
```

```
Call:
lm(formula = density ~ gain)

Coefficients:
(Intercept)         gain
   0.549724    -0.001533
```

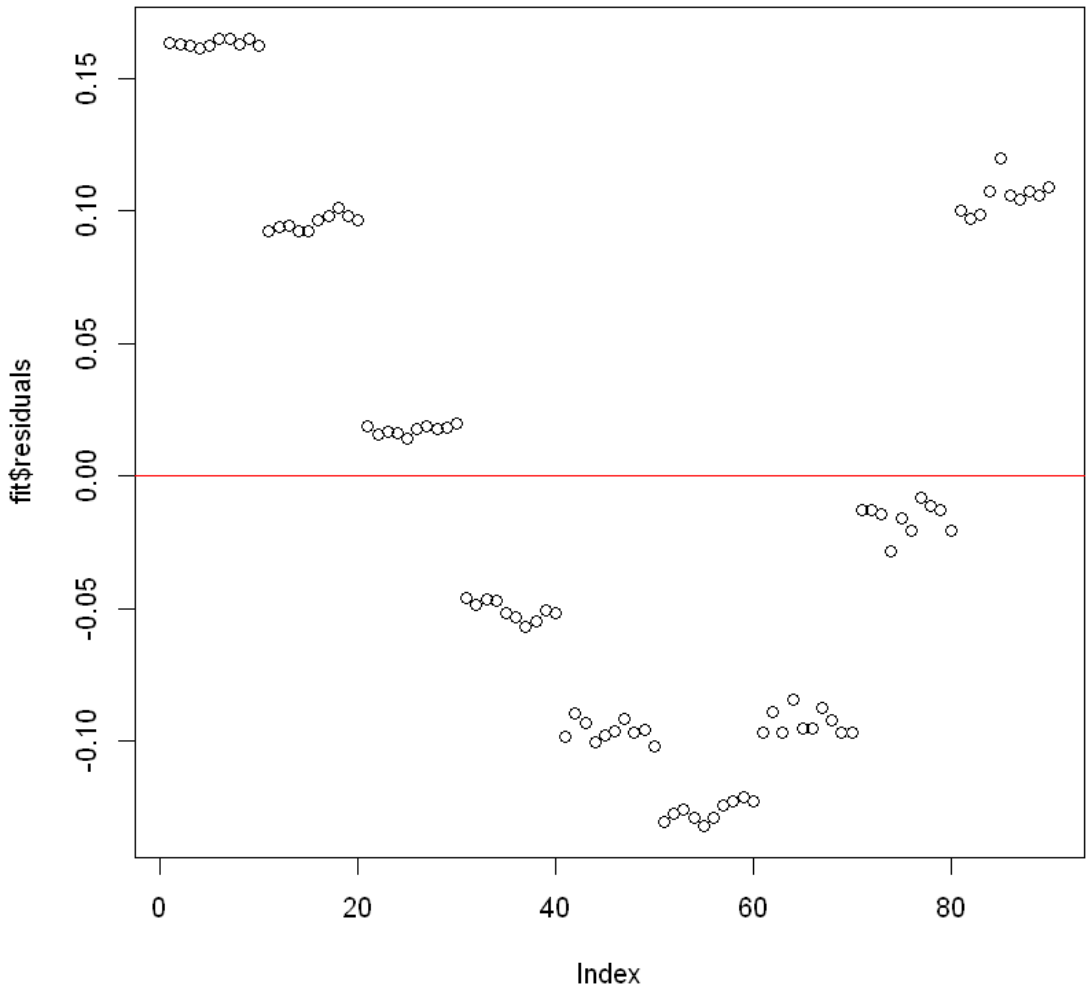
In [4]:

```
# plot the data with the regression line
plot(x=gain, y=density, xlab='gain', ylab='density',
    main='Scatter Plot of Gauge With Least Squared Line')
abline(fit, col='red')
```



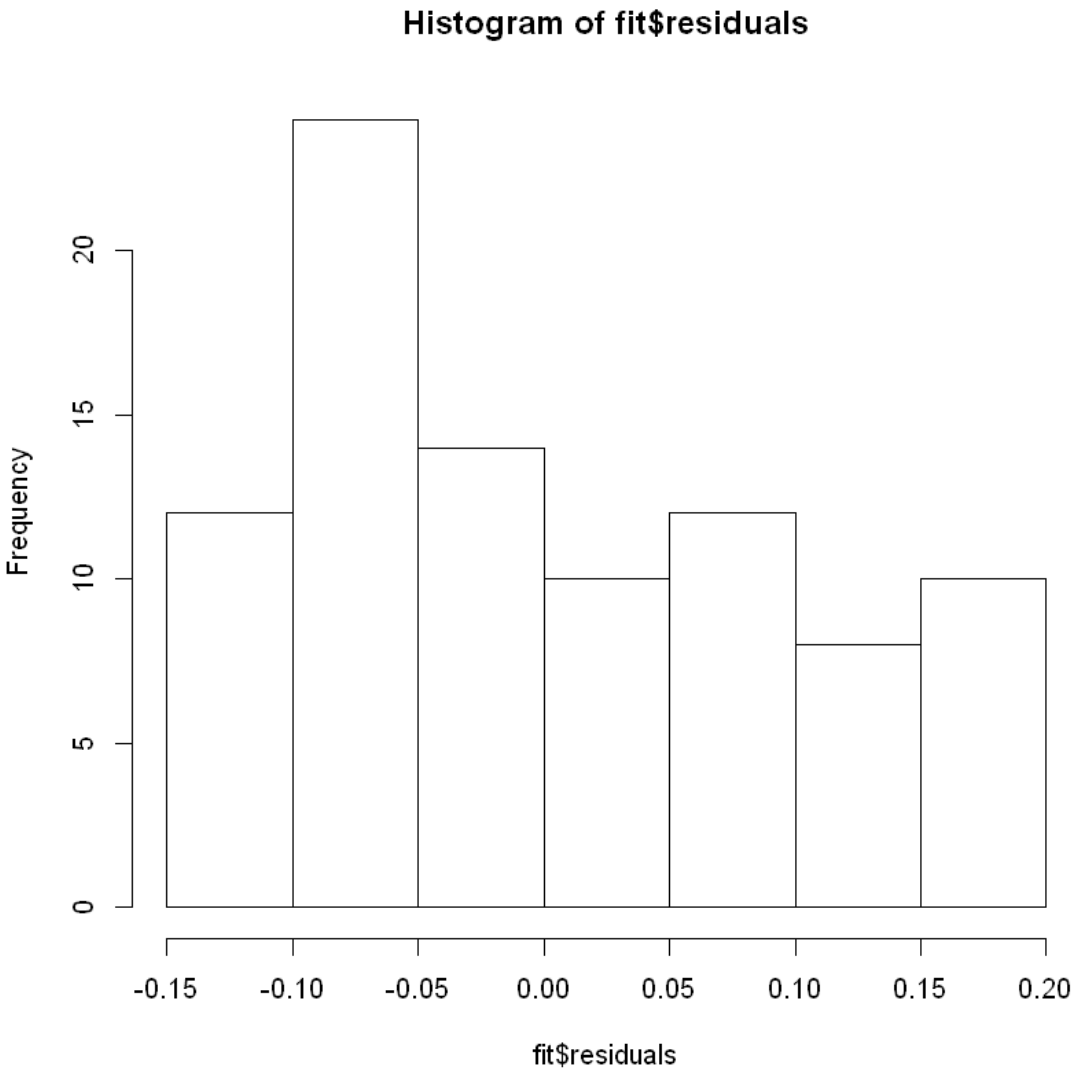
In [5]:

```
# examine the linearity of data and constant variability by the residual plot
plot(fit$residuals)
abline(0, 0, col='red')
```



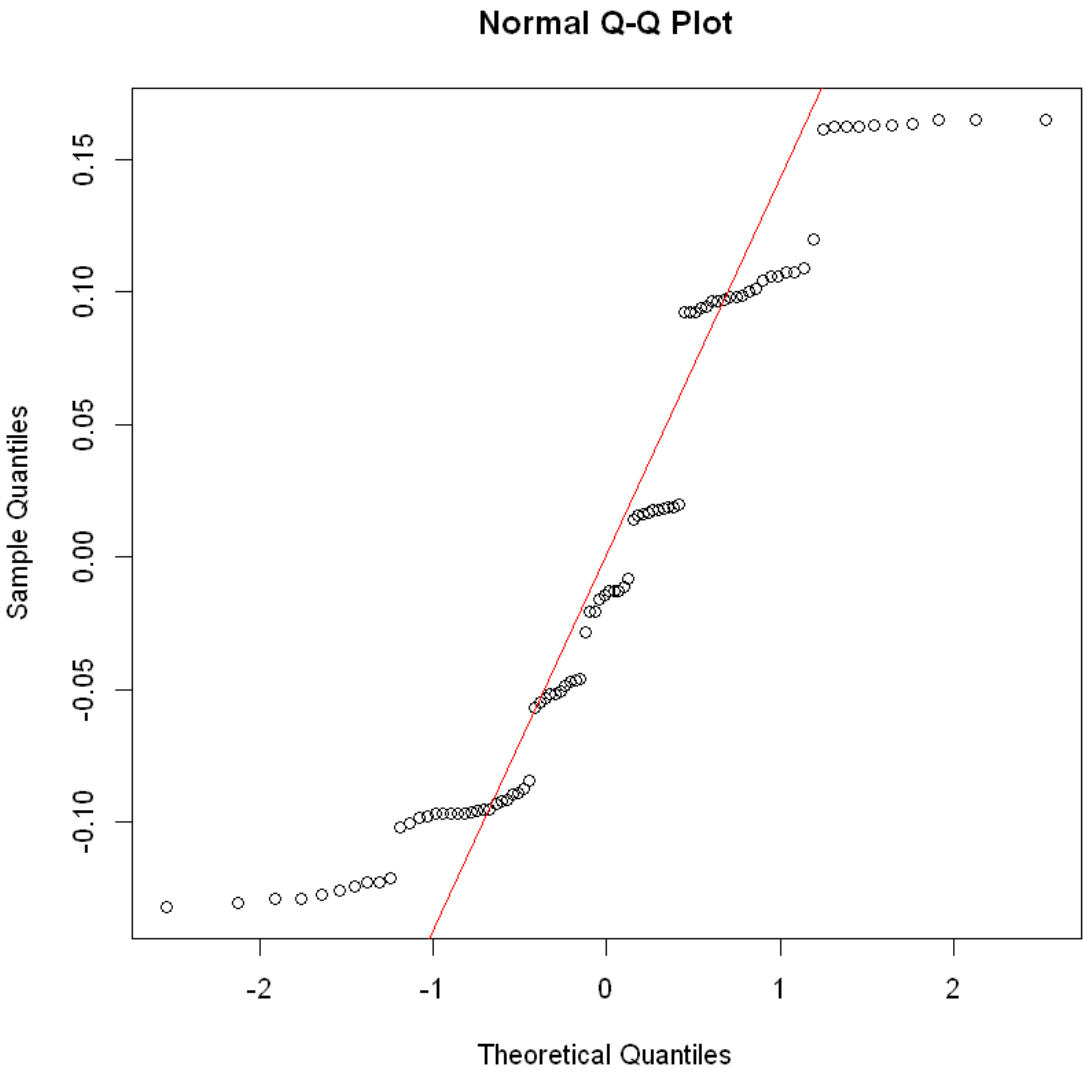
In [6]:

```
# examine the residual normality by histogram
hist(fit$residuals)
```



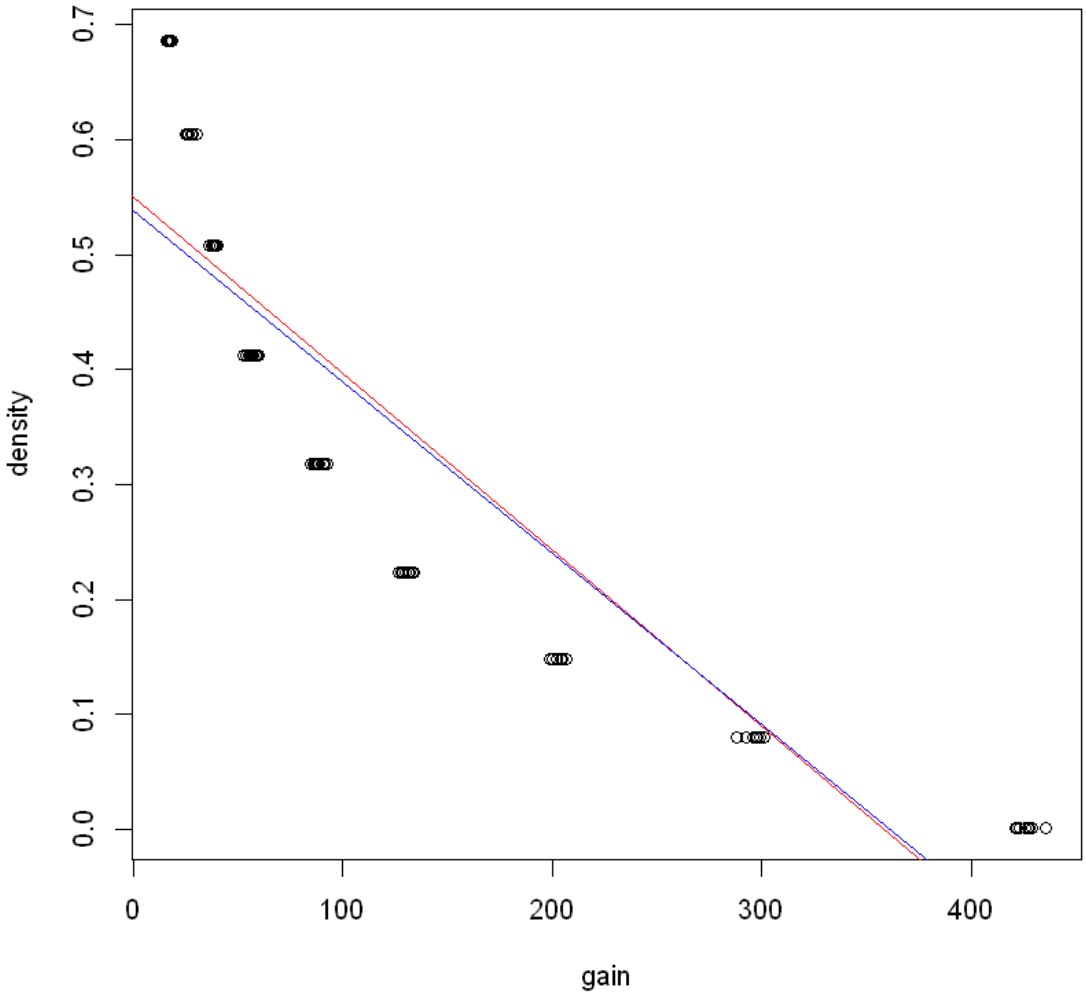
In [7]:

```
# examine the residual normality by Q-Q plot of the residuals
qqnorm(fit$residuals)
qqline(fit$residuals,col='red')
```



In [8]:

```
# remove some suspected and see how the fit will be affected
outliers1<-order(abs(fit$residuals))[88:90]
outliers2<-order(abs(fit$residuals))[1:3]
outliers <- c(outliers1, outliers2)
plot(x=gain,y=density)
abline(fit,col='red')
fit.out <- lm(density[-outliers]~gain[-outliers])
abline(fit.out, col='blue')
```



In [9]:

```
# check the fits
summary(fit)
```

```
Call:
lm(formula = density ~ gain)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.13198 -0.09452 -0.01354  0.09682  0.16495 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.5497239  0.0151243   36.35   <2e-16 ***
gain        -0.0015334  0.0000777  -19.73   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09769 on 88 degrees of freedom
Multiple R-squared:  0.8157,    Adjusted R-squared:  0.8136 
F-statistic: 389.5 on 1 and 88 DF,  p-value: < 2.2e-16
```

In [10]:

```
summary(fit.out)
```

```
Call:
lm(formula = density[-outliers] ~ gain[-outliers])

Residuals:
     Min       1Q   Median       3Q      Max 
-0.12583 -0.08873 -0.02176  0.09905  0.17434 

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)    
(Intercept)      5.379e-01  1.532e-02   35.10   <2e-16 ***
gain[-outliers] -1.488e-03  7.926e-05  -18.78   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09579 on 82 degrees of freedom
Multiple R-squared:  0.8113,    Adjusted R-squared:  0.809 
F-statistic: 352.6 on 1 and 82 DF,  p-value: < 2.2e-16
```

In [11]:

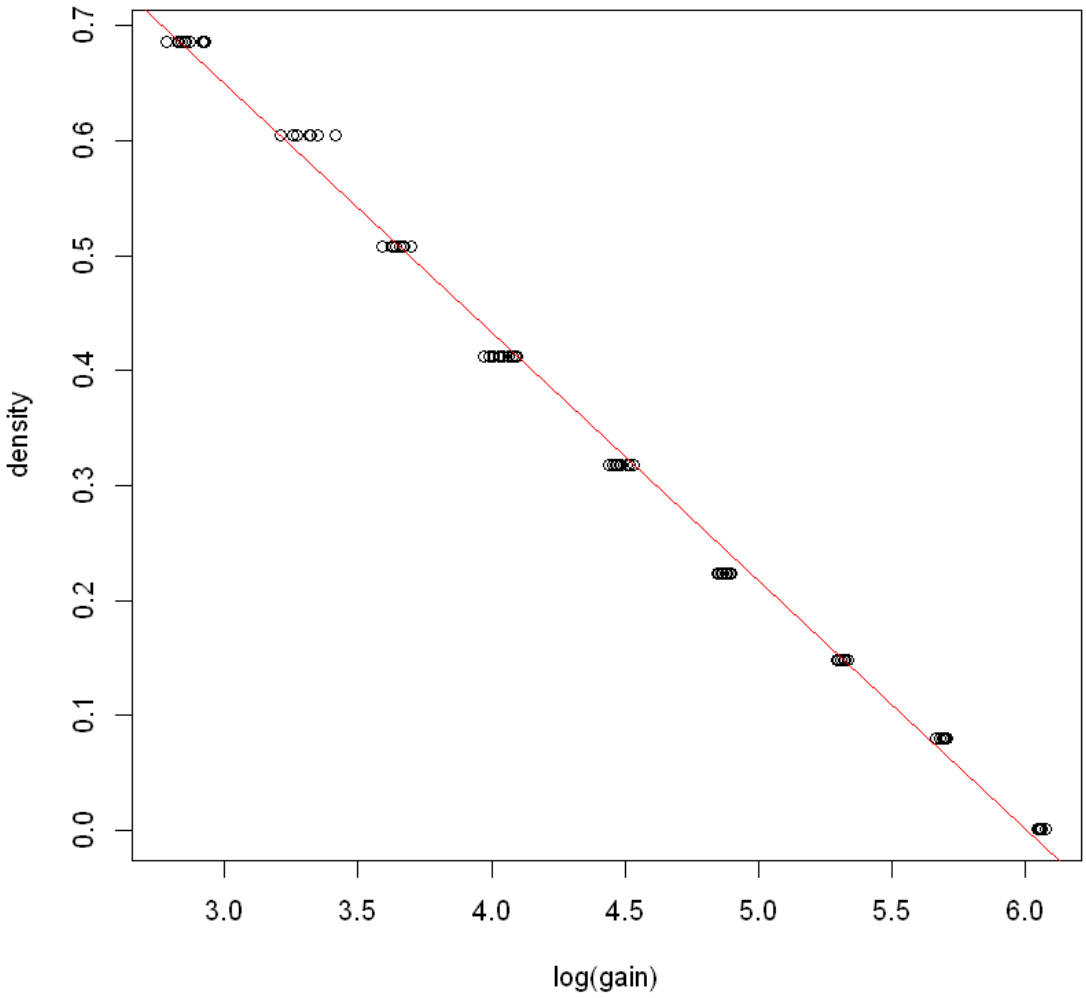
```
# create a new least squared line on the log(gain)
fit_log <- lm(formula=density~log(gain))
fit_log
```

```
Call:
lm(formula = density ~ log(gain))

Coefficients:
(Intercept)    log(gain)  
     1.2980      -0.2162  
```

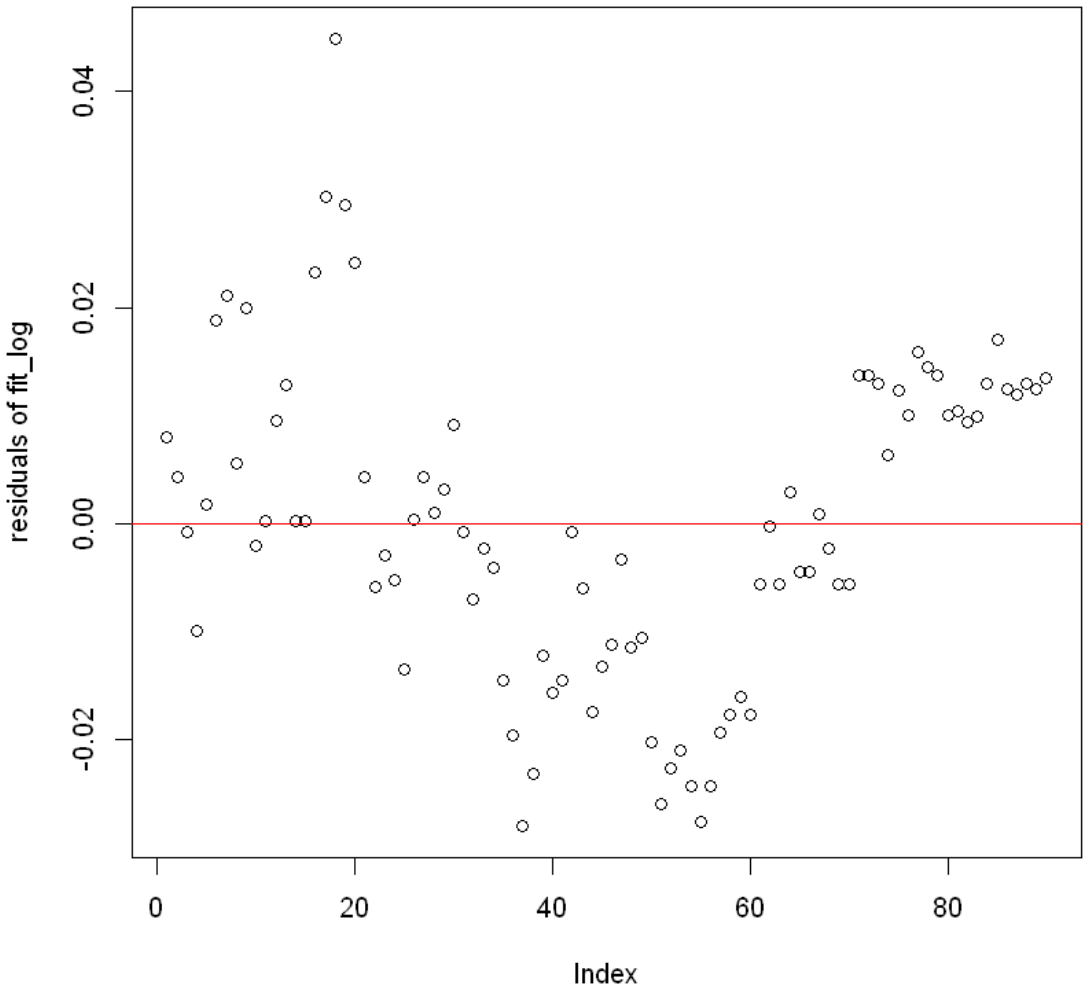
In [12]:

```
# plot the new fit(log) on the data
plot(x=log(gain),y=density, xlab='log(gain)',ylab='density')
abline(fit_log, col='red')
```



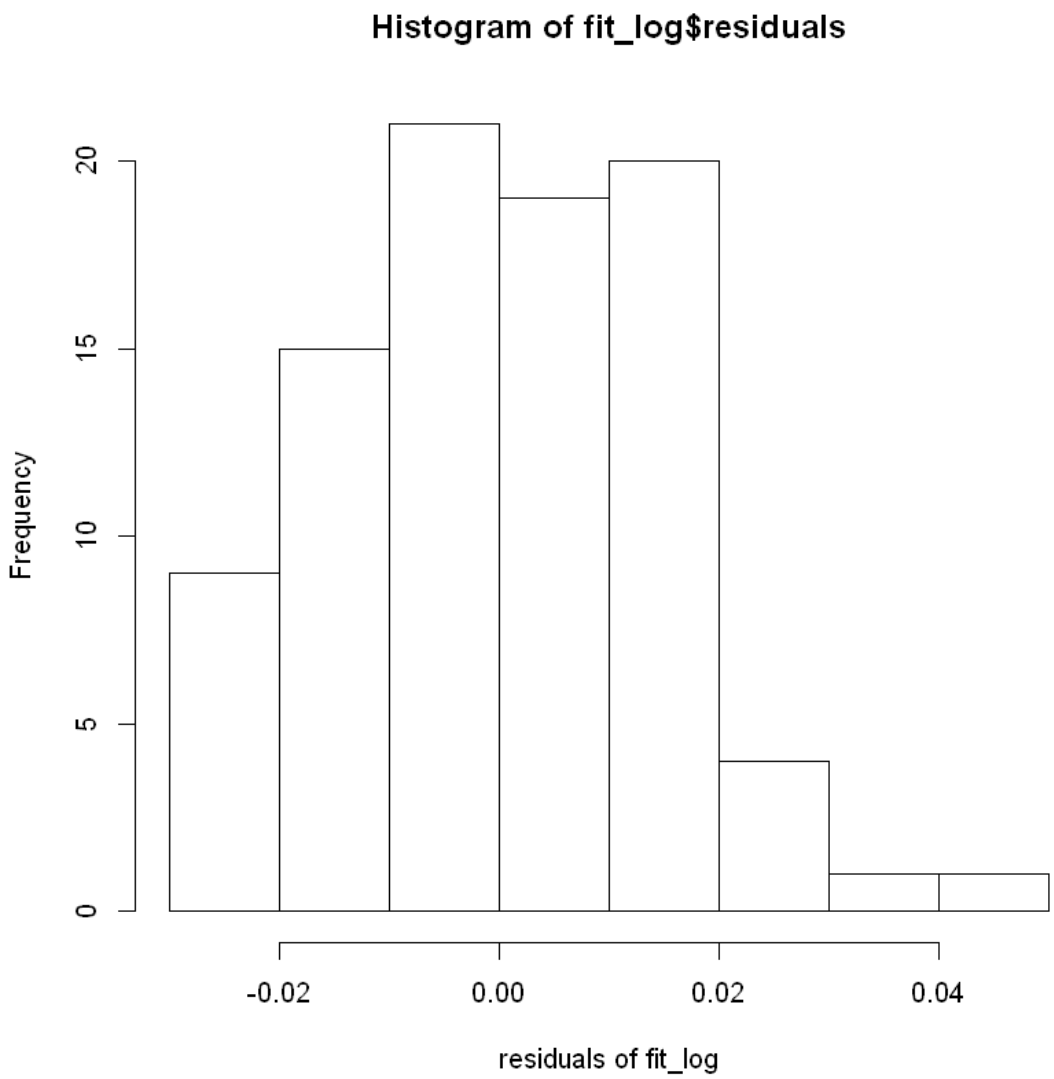
In [13]:

```
# examine the linearity and constant variablity by the residual plot
plot(fit_log$residuals, ylab='residuals of fit_log')
abline(0, 0, col='red')
```



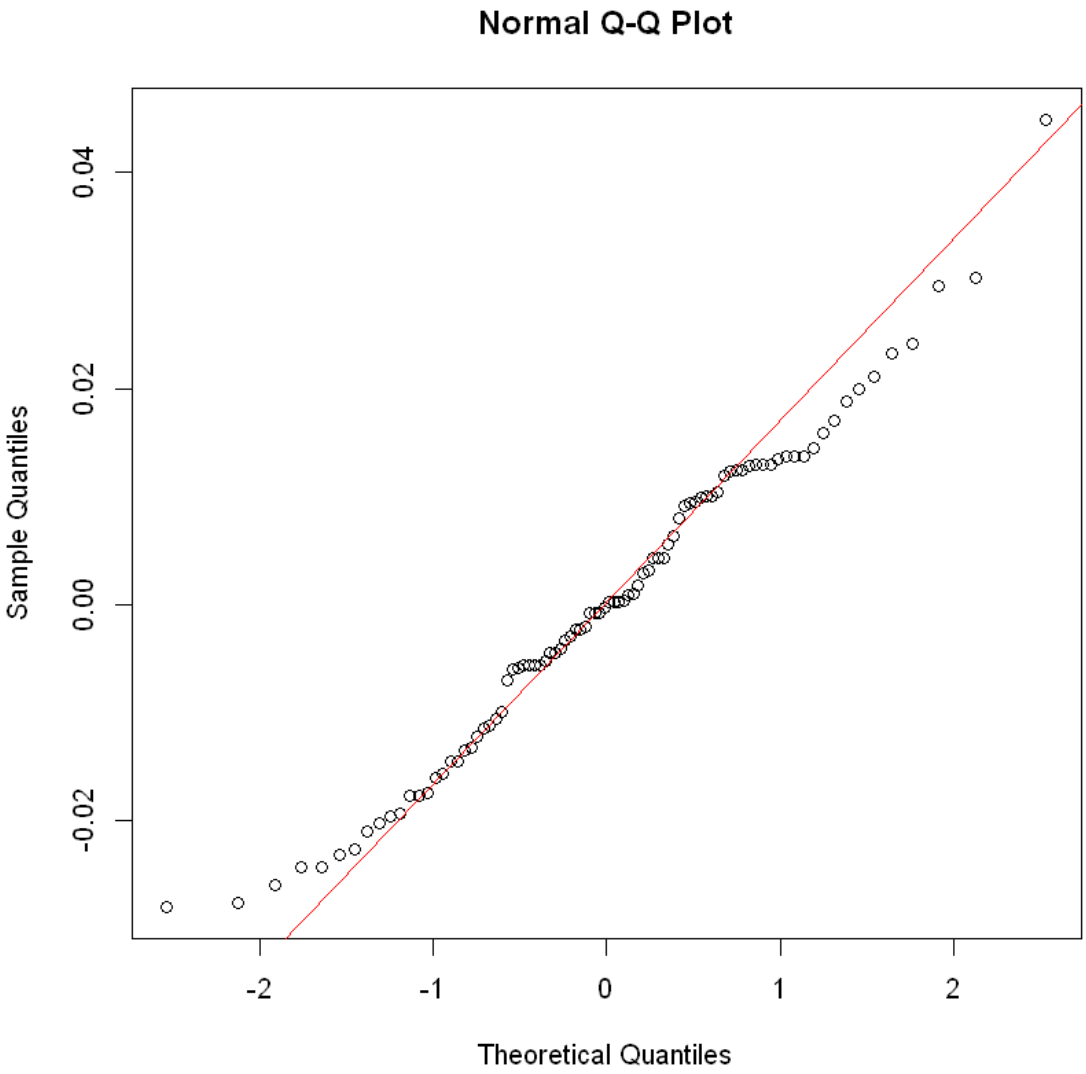
In [14]:

```
# examine the residual normality by histogram
hist(fit_log$residuals,xlab='residuals of fit_log')
```



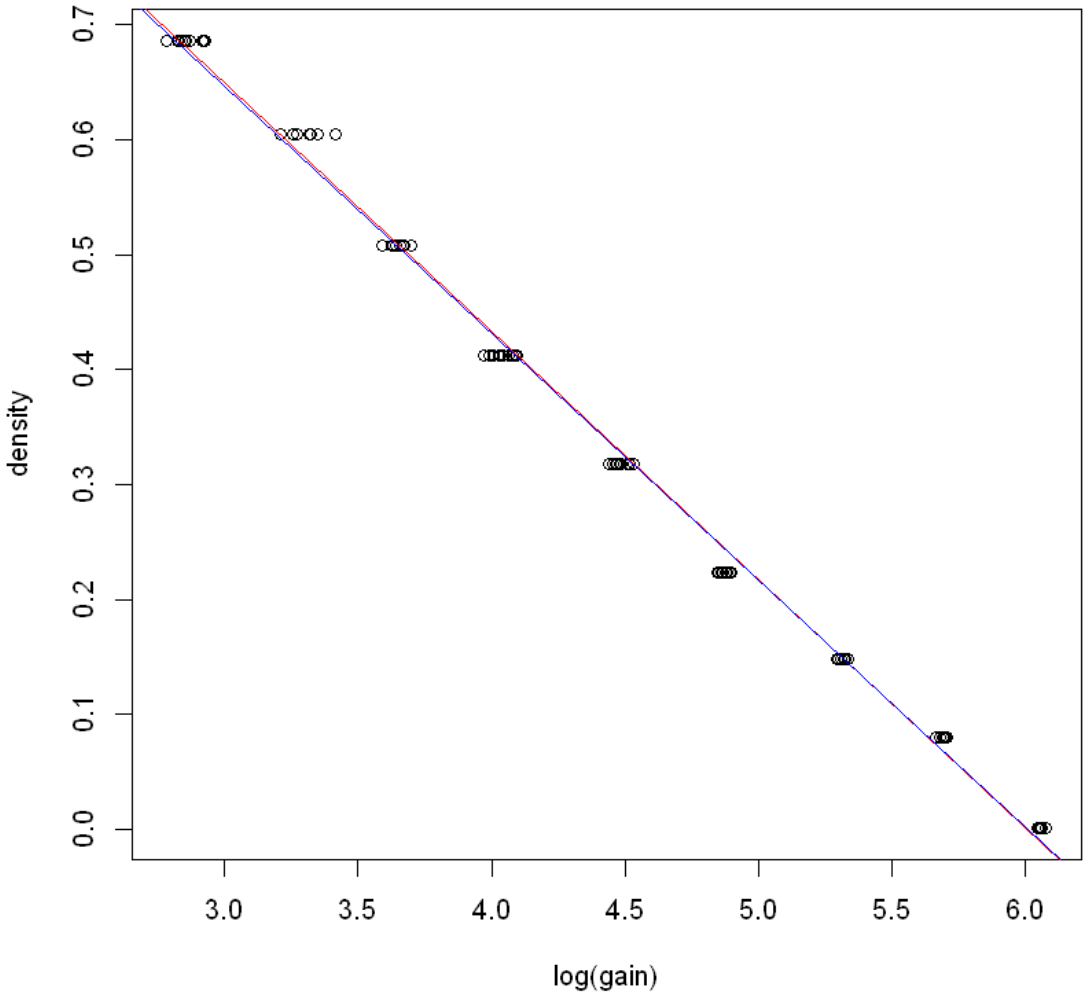
In [15]:

```
# examine the residual normality by Q-Q plot
qqnorm(fit_log$residuals)
qqline(fit_log$residuals, col='red')
```



In [16]:

```
# remove some suspected and see how the fit will be affected
outliers1<-order(abs(fit_log$residuals))[88:90]
outliers2<-order(abs(fit_log$residuals))[1:3]
outliers <- c(outliers1, outliers2)
plot(x=log(gain),y=density)
abline(fit_log,col='red')
fit.out <- lm(density[-outliers]~log(gain)[-outliers])
abline(fit.out, col='blue')
```



In [17]:

```
summary(fit_log)
```

```
Call:
lm(formula = density ~ log(gain))

Residuals:
      Min        1Q    Median        3Q       Max 
-0.028031 -0.011079 -0.000018  0.011595  0.044911 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.298013   0.006857   189.3   <2e-16 ***
log(gain)   -0.216203   0.001494  -144.8   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01471 on 88 degrees of freedom
Multiple R-squared:  0.9958,    Adjusted R-squared:  0.9958 
F-statistic: 2.096e+04 on 1 and 88 DF,  p-value: < 2.2e-16
```

In [18]:

```
summary(fit.out)
```

```
Call:
lm(formula = density[-outliers] ~ log(gain)[-outliers])

Residuals:
       Min         1Q     Median         3Q        Max 
-0.0268368 -0.0101833  0.0007768  0.0116165  0.0270840 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)           1.290341   0.006719   192.0   <2e-16 ***
log(gain)[-outliers] -0.214794   0.001439  -149.3   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0135 on 82 degrees of freedom
Multiple R-squared:  0.9963,    Adjusted R-squared:  0.9963 
F-statistic: 2.229e+04 on 1 and 82 DF,  p-value: < 2.2e-16
```

In [19]:

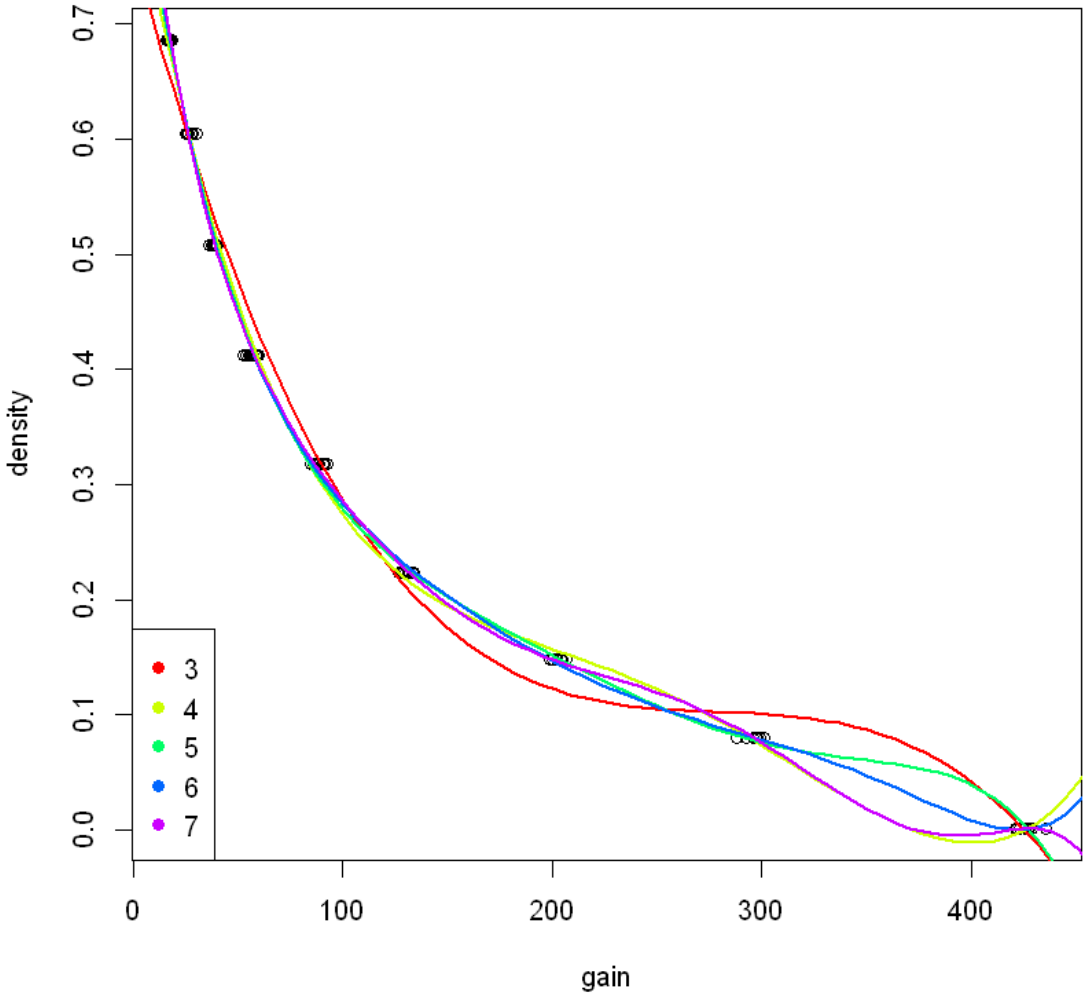
```
rainbow(5)
```

```
'#FF0000FF' '#CCFF00FF' '#00FF66FF' '#0066FFFF' '#CC00FFFF'
```

In [20]:

```
# see the performance of poly fit
plot(gain, density)
legend('bottomleft',
       legend=c('3','4','5','6','7'),
       col=rainbow(5),pch=c(19))
pts <- seq(0.001,600, length.out=90)

# run 5 times
for (d in 3:7){
    fit_test <- lm(density~poly(gain, d, raw=TRUE))
    val <- predict(fit_test, data.frame(gain=pts))
    lines(pts,val, col=rainbow(5)[d-2],lwd=2)
}
```



In [21]:

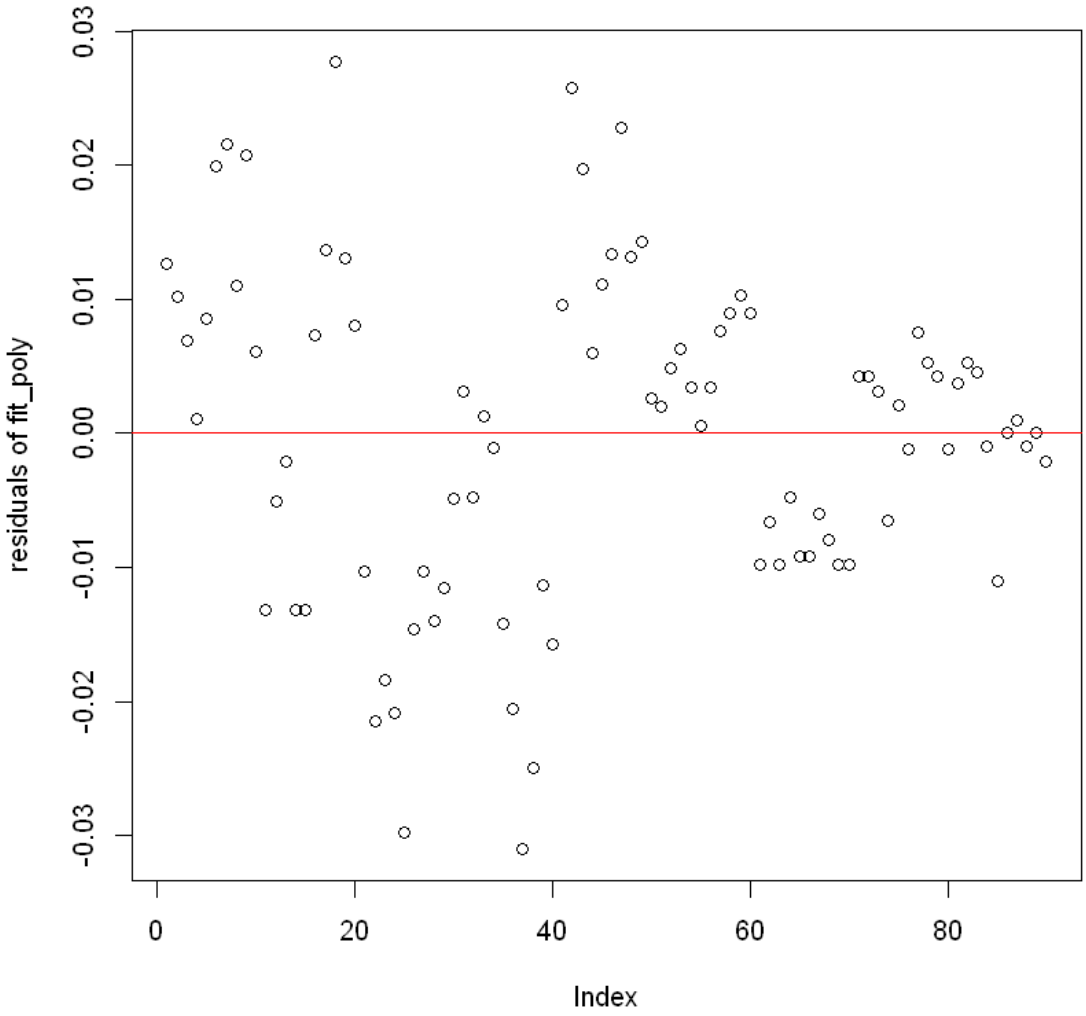
```
# create a new polynomial fit on log(gain)
fit_poly <- lm(density~poly(gain,degree=4,raw=TRUE))
fit_poly
```

```
Call:
lm(formula = density ~ poly(gain, degree = 4, raw = TRUE))

Coefficients:
                          (Intercept)  poly(gain, degree = 4, raw = TRUE)1
                            8.340e-01                            -1.013e-02
poly(gain, degree = 4, raw = TRUE)2  poly(gain, degree = 4, raw = TRUE)3
                            5.998e-05                            -1.626e-07
poly(gain, degree = 4, raw = TRUE)4
                            1.568e-10
```

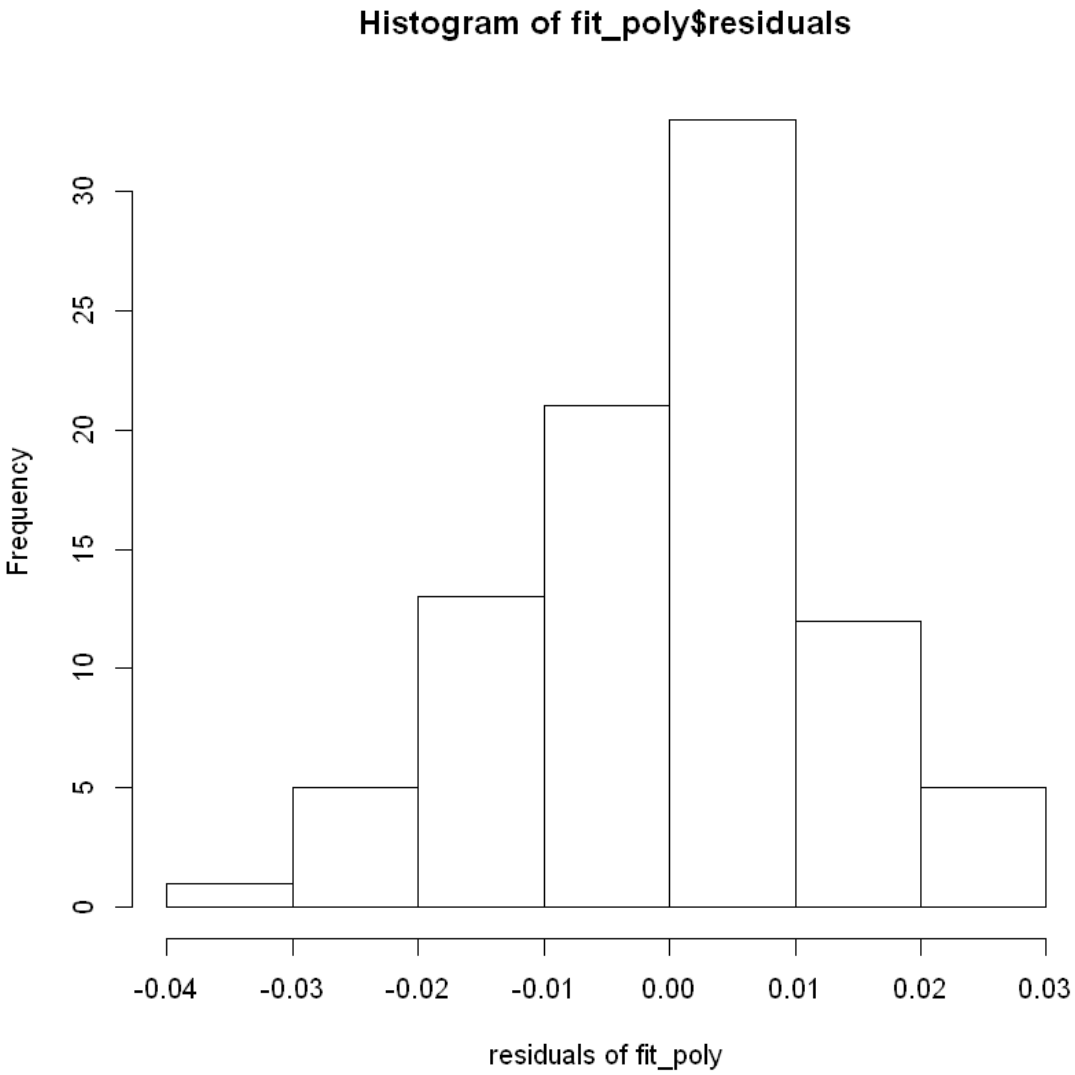
In [22]:

```
# examine the linearity and constant variability by residual plot
plot(fit_poly$residuals,ylab='residuals of fit_poly')
abline(0, 0, col='red')
```



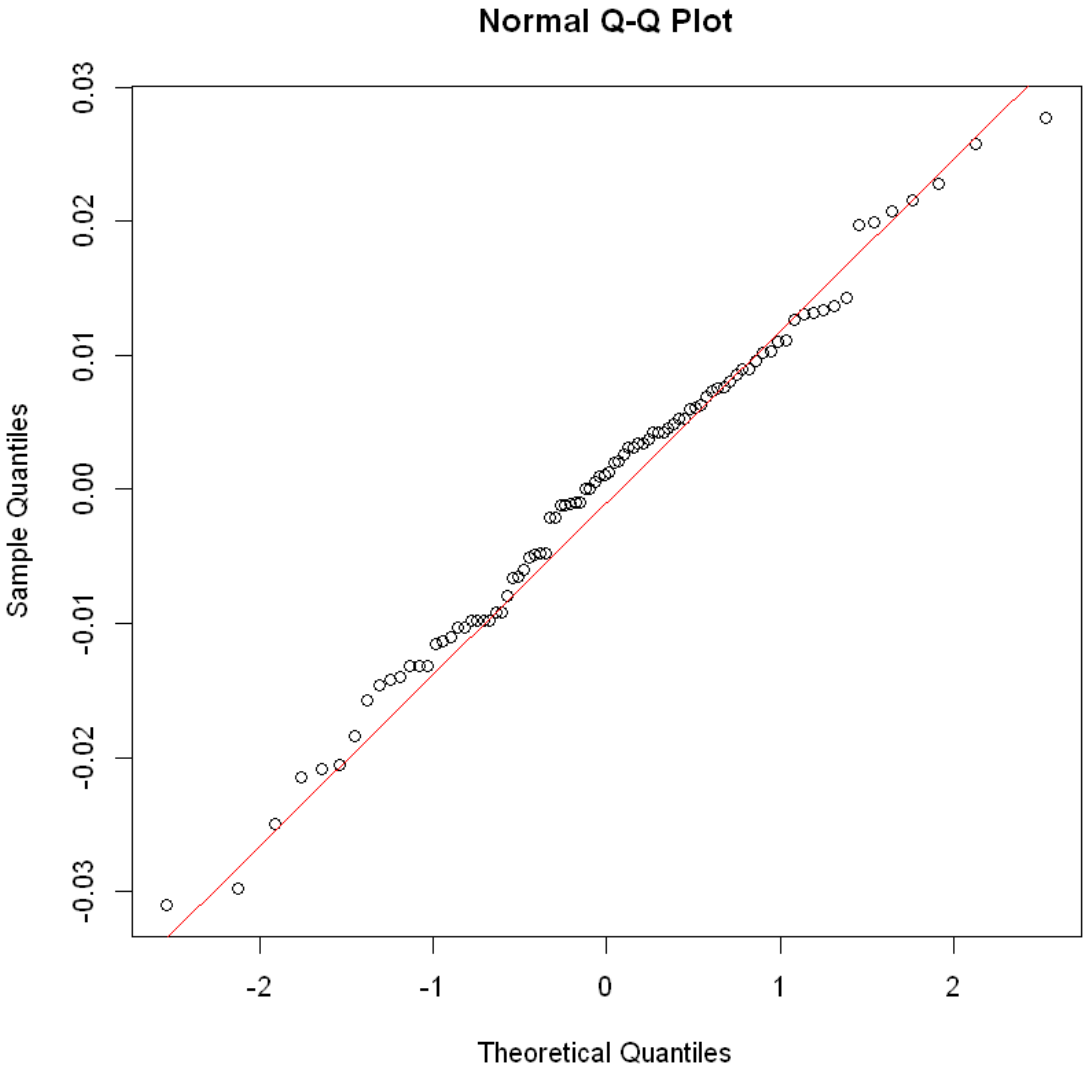
In [23]:

```
# examine residual normality by histogram
hist(fit_poly$residuals,xlab='residuals of fit_poly')
```



In [24]:

```
# examine residual normality by Q-Q plot
qqnorm(fit_poly$residuals)
qqline(fit_poly$residuals,col='red')
```



In [25]:

```
# remove some suspected and see how the fit will be affected
outliers1<-order(abs(fit_poly$residuals))[88:90]
outliers2<-order(abs(fit_poly$residuals))[1:3]
outliers <- c(outliers1, outliers2)
fit.out <- lm(density[-outliers]~poly(log(gain[-outliers]),degree=4,raw=TRUE))
```

In [26]:

```
summary(fit_poly)
```

```
Call:
lm(formula = density ~ poly(gain, degree = 4, raw = TRUE))

Residuals:
      Min        1Q    Median        3Q       Max
-0.031008 -0.009623  0.001183  0.007614  0.027727

Coefficients:
                                      Estimate Std. Error t value Pr(>|
t|)
(Intercept)                          8.340e-01  5.982e-03  139.42   <2e
-16 ***
poly(gain, degree = 4, raw = TRUE)1 -1.013e-02  2.359e-04  -42.95   <2e
-16 ***
poly(gain, degree = 4, raw = TRUE)2  5.998e-05  2.470e-06   24.28   <2e
-16 ***
poly(gain, degree = 4, raw = TRUE)3 -1.626e-07  9.132e-09  -17.80   <2e
-16 ***
poly(gain, degree = 4, raw = TRUE)4  1.569e-10  1.073e-11   14.62   <2e
-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01239 on 85 degrees of freedom
Multiple R-squared:  0.9971,    Adjusted R-squared:  0.997
F-statistic:  7397 on 4 and 85 DF,  p-value: < 2.2e-16
```

In [27]:

```
summary(fit.out)
```

```
Call:
lm(formula = density[-outliers] ~ poly(log(gain[-outliers]), 
    degree = 4, raw = TRUE))

Residuals:
       Min         1Q     Median         3Q        Max 
-0.0156163 -0.0037648 -0.0003798  0.0040146  0.0175902 

Coefficients:
                                                    Estimate Std. Error
(Intercept)                                        -1.818904   0.425828
poly(log(gain[-outliers]), degree = 4, raw = TRUE)1  2.800310   0.407743
poly(log(gain[-outliers]), degree = 4, raw = TRUE)2 -1.056375   0.143135
poly(log(gain[-outliers]), degree = 4, raw = TRUE)3  0.158892   0.021859
poly(log(gain[-outliers]), degree = 4, raw = TRUE)4 -0.008688   0.001227
                                                   t value Pr(>|t|)    
(Intercept)                                         -4.271 5.37e-05 ***
poly(log(gain[-outliers]), degree = 4, raw = TRUE)1   6.868 1.34e-09 ***
poly(log(gain[-outliers]), degree = 4, raw = TRUE)2  -7.380 1.39e-10 ***
poly(log(gain[-outliers]), degree = 4, raw = TRUE)3   7.269 2.28e-10 ***
poly(log(gain[-outliers]), degree = 4, raw = TRUE)4  -7.080 5.26e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.007486 on 79 degrees of freedom
Multiple R-squared:  0.9989,    Adjusted R-squared:  0.9989 
F-statistic: 1.878e+04 on 4 and 79 DF,  p-value: < 2.2e-16
```

In [28]:

```
# create a error density dataframe
gauge_error <- data.frame(gauge)
gauge_error[3,1] <- 0.8
gauge_error[63,1] <- 0.4
density_error <- gauge_error$density
```

In [29]:

```
fit
```

```
Call:
lm(formula = density ~ gain)

Coefficients:
(Intercept)         gain  
   0.549724    -0.001533  
```

In [30]:

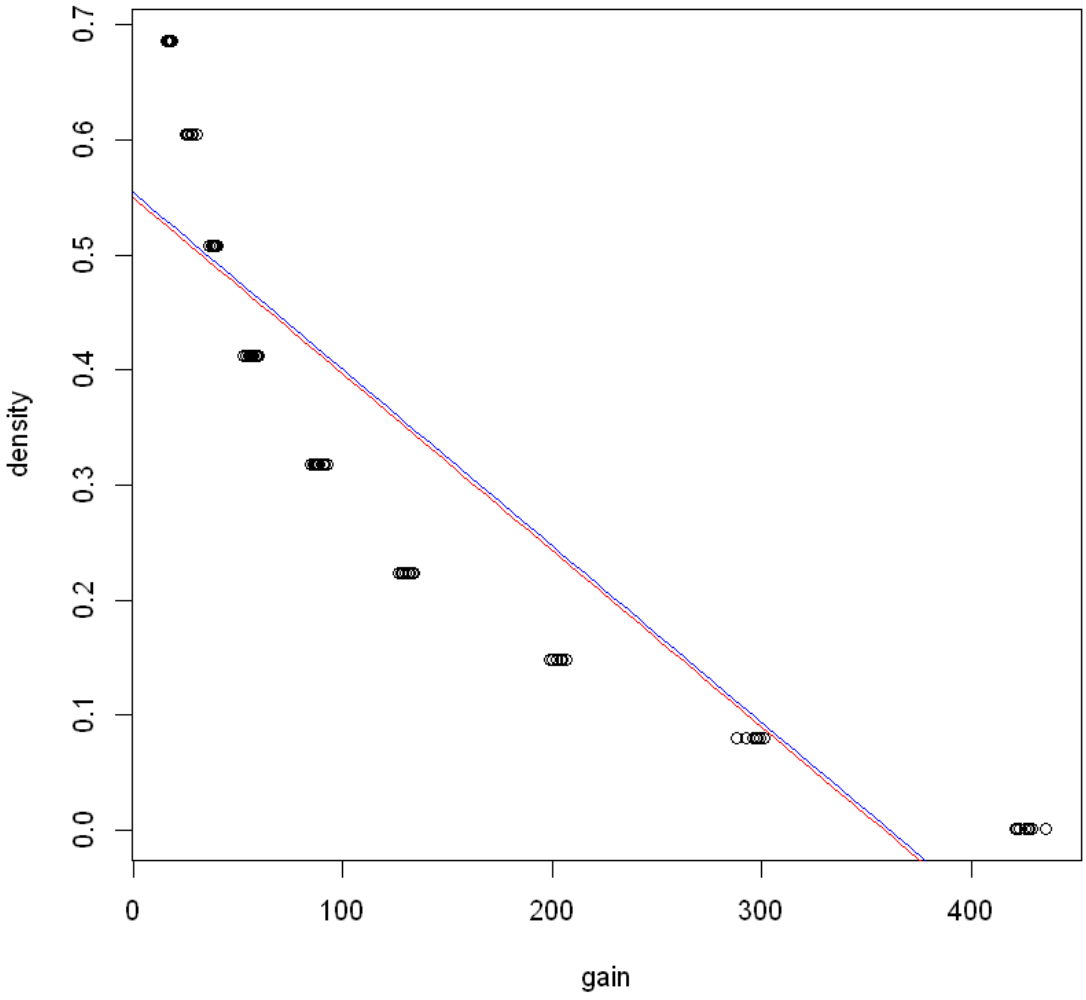
```
fit_error <- lm(formula=density_error~gain)
fit_error
```

```
Call:
lm(formula = density_error ~ gain)

Coefficients:
(Intercept)         gain
   0.553800    -0.001533
```

In [31]:

```
plot(gain, density)
pts <- seq(0.001,600, length.out=90)
val <- predict(fit, data.frame(gain=pts))
val_error <- predict(fit_error, data.frame(gain=pts))
lines(pts,val, col='red')
lines(pts,val_error, col='blue')
```



In [32]:

```
fit_log
```

```
Call:
lm(formula = density ~ log(gain))

Coefficients:
(Intercept)    log(gain)
     1.2980      -0.2162
```

In [33]:

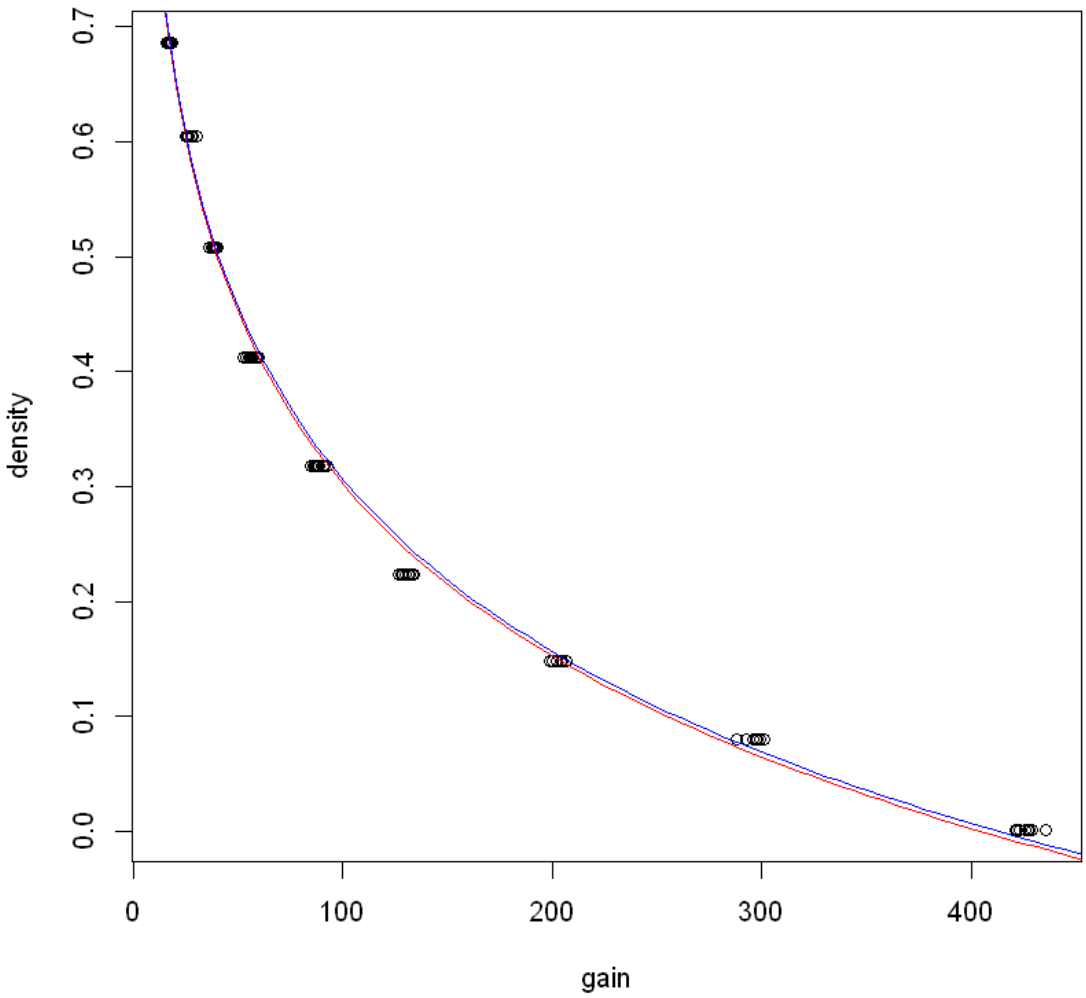
```
fit_log_error <- lm(formula=density_error~log(gain))
fit_log_error
```

```
Call:
lm(formula = density_error ~ log(gain))

Coefficients:
(Intercept)    log(gain)
      1.301       -0.216
```

In [34]:

```
plot(gain, density)
pts <- seq(0.001,600, length.out=90)
val <- predict(fit_log, data.frame(gain=pts))
val_error <- predict(fit_log_error, data.frame(gain=pts))
lines(pts,val, col='red')
lines(pts,val_error, col='blue')
```


density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
100
200
300
400
gain

In [35]:

```
fit_poly
```

```
Call:
lm(formula = density ~ poly(gain, degree = 4, raw = TRUE))

Coefficients:
                         (Intercept)  poly(gain, degree = 4, raw = TRUE)1
                           8.340e-01                           -1.013e-02
poly(gain, degree = 4, raw = TRUE)2  poly(gain, degree = 4, raw = TRUE)3
                           5.998e-05                           -1.626e-07
poly(gain, degree = 4, raw = TRUE)4
                           1.568e-10
```

In [36]:

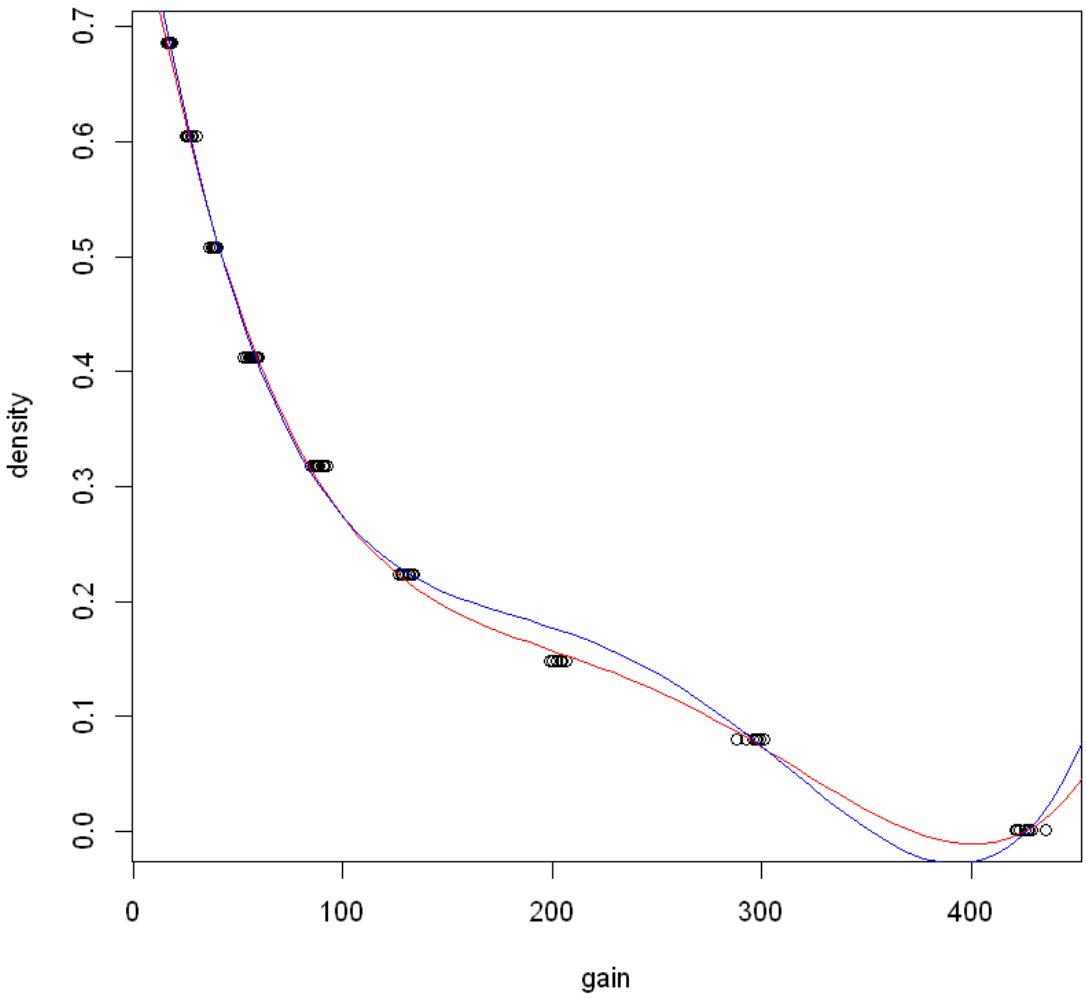
```
fit_poly_error <- lm(density_error~poly(gain,degree=4,raw=TRUE))
fit_poly_error
```

```
Call:
lm(formula = density_error ~ poly(gain, degree = 4, raw = TRUE))

Coefficients:
                         (Intercept)  poly(gain, degree = 4, raw = TRUE)1
                           8.595e-01                           -1.122e-02
poly(gain, degree = 4, raw = TRUE)2  poly(gain, degree = 4, raw = TRUE)3
                           7.242e-05                           -2.090e-07
poly(gain, degree = 4, raw = TRUE)4
                           2.105e-10
```

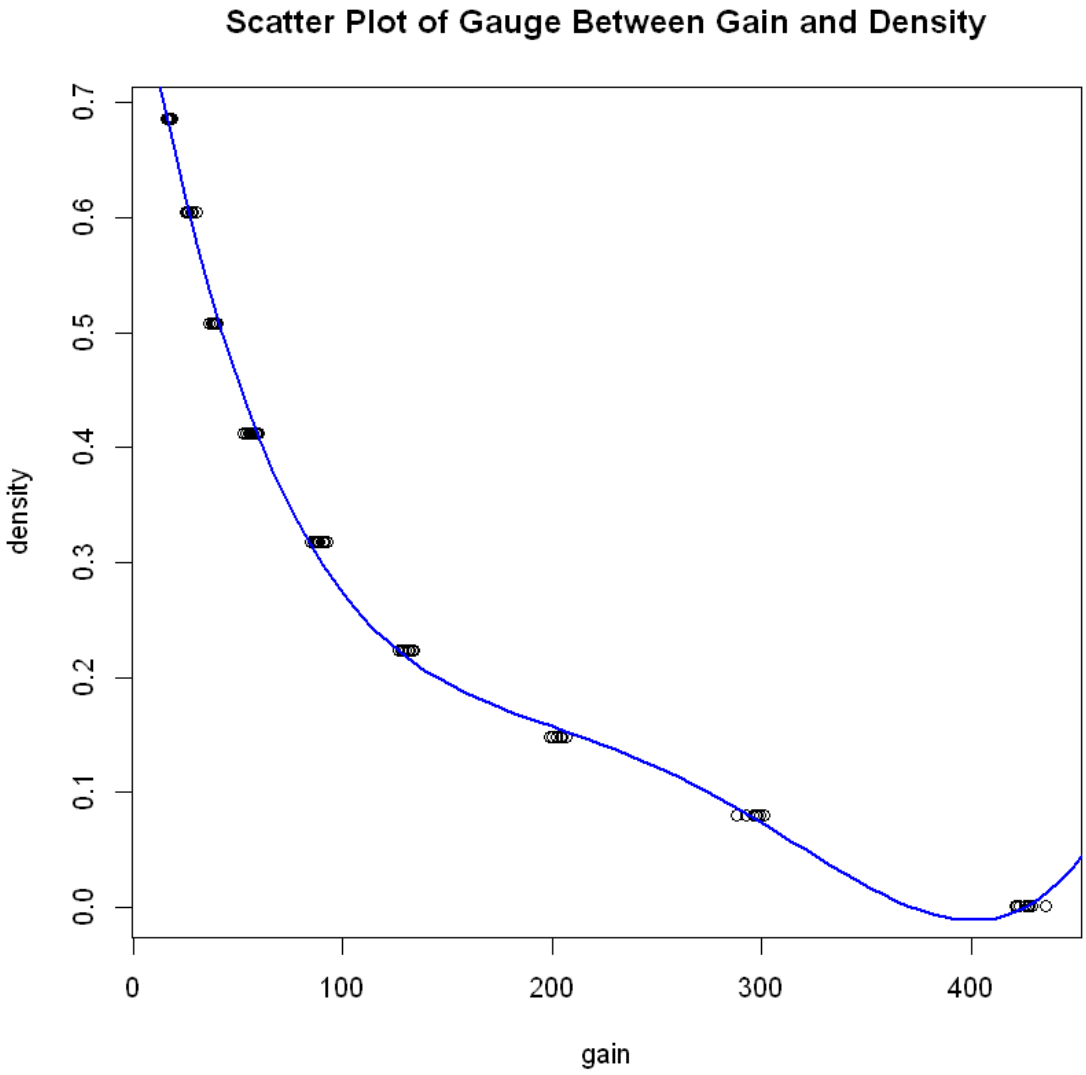
In [37]:

```
# see the performance of poly fit and ploy_error fit
plot(gain, density)
pts <- seq(0.001,600, length.out=90)
val <- predict(fit_poly, data.frame(gain=pts))
val_error <- predict(fit_poly_error, data.frame(gain=pts))
lines(pts,val, col='red')
lines(pts,val_error, col='blue')
```


density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
100
200
300
400
gain

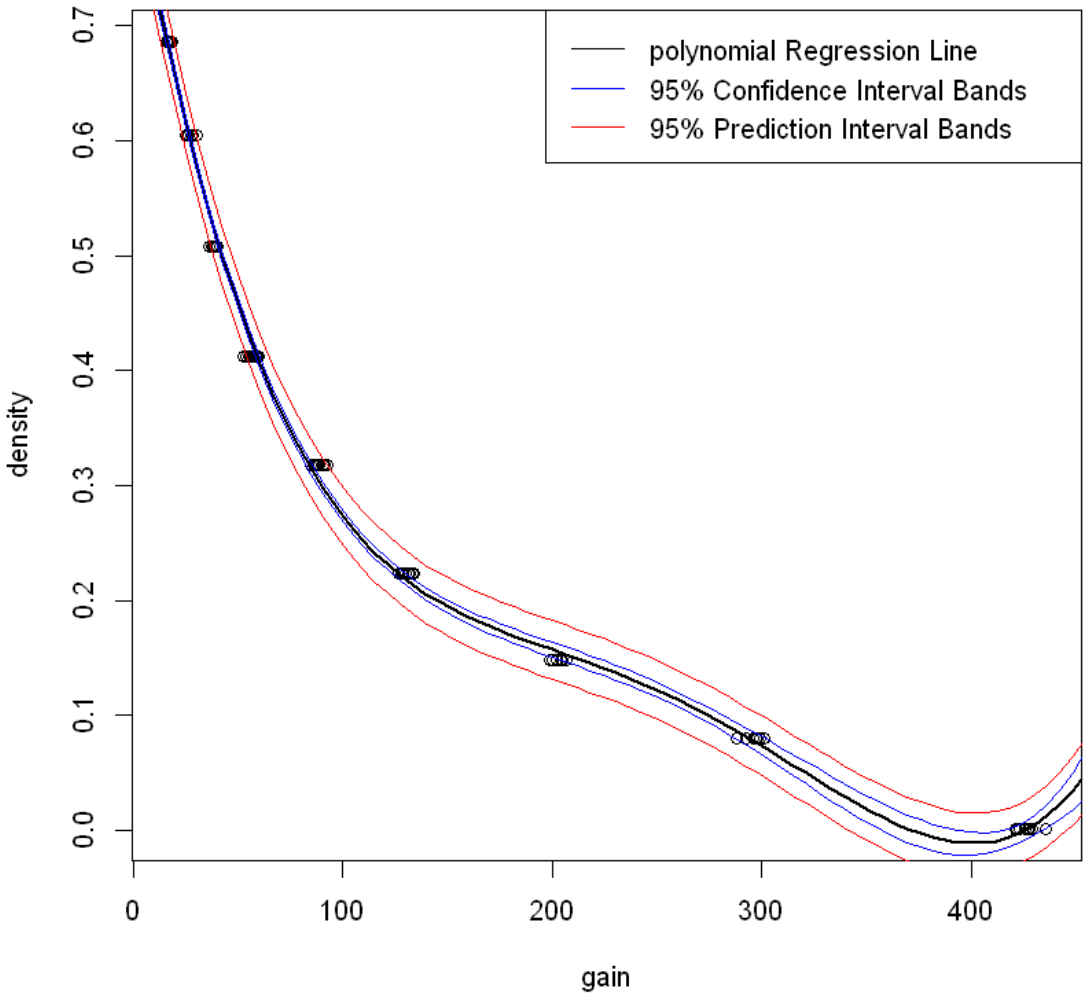
In [38]:

```
# use fit_poly above to predict
pts <- seq(0, 600, length.out=100)
val_1<- predict(fit_poly, data.frame(gain=pts))
plot(x=gain, y=density, xlab='gain', ylab='density',
    main='Scatter Plot of Gauge Between Gain and Density', col = 'black')
lines(pts, val_1, col="blue", lwd=2)
```



In [39]:

```
CI.conf <- predict(fit_poly, data.frame(gain=pts), interval = "confidence") #confidence
interval
CI.pred <- predict(fit_poly, data.frame(gain=pts), interval = "predict") #prediction in
terval
plot(gain, density)
lines(pts, CI.conf[,"fit"], col="black", lwd=2)
lines(pts, CI.conf[,"lwr"], col="blue", lwd=1)
lines(pts, CI.conf[,"upr"], col="blue", lwd=1)
lines(pts, CI.pred[,"lwr"], col="red", lwd=1)
lines(pts, CI.pred[,"upr"], col="red", lwd=1)
legend('topright', legend=c('polynomial Regression Line', '95% Confidence Interval Band
s', '95% Prediction Interval Bands'), col=c('black', 'blue', 'red'), lty=1)
```



In [40]:

```
val1=predict(fit_poly,data.frame(gain=38.6))
val2=predict(fit_poly,data.frame(gain=426.7))
print(paste("the predicted density of gain=38.6 is:",val1))
print(paste("the predicted density of gain=426.7 is:",val2))
```

```
[1] "the predicted density of gain=38.6 is: 0.523250800770545"
[1] "the predicted density of gain=426.7 is: 0.0006650839718718l6"
```

Adcanced

In [41]:

```
library(e1071)
Gain=gauge$gain
Dense=gauge$density
```

```
Error in library(e1071): there is no package called 'e1071'
Traceback:

1. library(e1071)
```

In [ ]:

```
cv.degree.d <- function(k, n, d){
  val.size <- floor(n/k)
  folds_i <- sample(rep(1:k, length = n))
  cv.mse <- rep(0, k)
  ind.remain=1:n
  for (round in 1:k){
    #val.ind <- sample(ind.remain, val.size, replace = FALSE)
    val.ind <- which(folds_i == round)
    y <- Dense[-val.ind]
    x <- Gain[-val.ind]
    fit <- lm(y ~ poly(x, d, raw=TRUE))
    y.hat <- predict(fit, data.frame(x = Gain[val.ind]))
    cv.mse[round] <- mean((Dense[val.ind] - y.hat)^2)
  }
  return (mean(cv.mse))
}
```

In [ ]:

```
set.seed(2020)
k <- 10
n <- 90
d.max <- 9
mse <- rep(0, d.max)
for (d in 1:d.max){
  mse[d] <- cv.degree.d(k, n, d)
}
plot(1:d.max, mse, xlab = "Degree", ylab = "MSE", lwd = 2, col = "blue", pch = 5)
lines(1:d.max, mse, type='l', lwd = 2, col = "blue")
```

In [ ]:

```
model <- svm(Gain, Dense, kernel = 'polynomial', degree = 3, coef0 = 1)
new <- predict(model, data.frame(x = Gain))
plot(Gain, Dense, xlab='gain', ylab='density',
    main='Scatter Plot of Gauge Between Gain and Density')
points(Gain, new, col = 4)
```

In [ ]:

```
cv.degree.d <- function(k, n, d){
  val.size <- floor(n/k)
  folds_i <- sample(rep(1:k, length = n))
  cv.mse <- rep(0, k)
  ind.remain=1:n
  for (round in 1:k){
    #val.ind <- sample(ind.remain, val.size, replace = FALSE)
    val.ind <- which(folds_i == round)
    y <- Dense[-val.ind]
    x <- Gain[-val.ind]
    fit <- svm(x, y, kernel = 'polynomial', degree = d, coef0 = 1)
    y.hat <- predict(fit, data.frame(x = Gain[val.ind]))
    cv.mse[round] <- mean((Dense[val.ind] - y.hat)^2)
  }
  return (mean(cv.mse))
}
```

In [ ]:

```
set.seed(2020)
k <- 10
n <- 90
d.max <- 9
mse <- rep(0, d.max)
for (d in 1:d.max){
  mse[d] <- cv.degree.d(k, n, d)
}
plot(1:d.max, mse, xlab = "Degree", ylab = "MSE", lwd = 2, col = "blue", pch = 5)
lines(1:d.max, mse, type='l', lwd = 2, col = "blue")
```

In [ ]:

```
model <- svm(Gain, Dense, kernel = 'polynomial', degree = 4, coef0 = 1)
new <- predict(model, data.frame(x = Gain))
plot(Gain, Dense, xlab='gain', ylab='density',
    main='Scatter Plot of Gauge Between Gain and Density')
points(Gain, new, col = 4)
```