Bivariate Regression Analysis

Yalin Yang

2020-04-27

Contents

Quantile-Normal Plot

Initialize data

```
## Data vector with ties in the last four elements
x <- c(4.0,4.4,3.8,2.5,5.1,4.5,3.8,4.8,4.4,4.1)

## Sorting
( xSort <- sort(x) )   # works only on vectors
```

```
##  [1] 2.5 3.8 3.8 4.0 4.1 4.4 4.4 4.5 4.8 5.1
```

Re-ordering works on matrices and data-frames

```
shuffle <- order(x)      # Order generates a shuffle index
xOrdered <- x[shuffle]   # Shuffle data positions in vector
(cbind(x, shuffle, xOrdered))
```

```
##          x shuffle xOrdered
##  [1,] 4.0       4      2.5
##  [2,] 4.4       3      3.8
##  [3,] 3.8       7      3.8
##  [4,] 2.5       1      4.0
##  [5,] 5.1      10      4.1
##  [6,] 4.5       2      4.4
##  [7,] 3.8       9      4.4
##  [8,] 4.8       6      4.5
##  [9,] 4.4       8      4.8
## [10,] 4.1       5      5.1
```

```
## Ranking data
(xRank <- rank(x, ties.method="random"))      # Explore other methods
```

```
##  [1]  4  7  3  1 10  8  2  9  6  5
```

Quantiles

```
## quantiles Q[i](p) = (1 - z)*x[j] + z*x[j+1] with 0 <= z <= 1
( quantile(xOrdered,prob=seq(0.1,0.9,by=0.1)) )
```

```
##  10%  20%  30%  40%  50%  60%  70%  80%  90%
## 3.67 3.80 3.94 4.06 4.25 4.40 4.43 4.56 4.83
```

```
( quantile(xOrdered,prob=c(0.25,0.5,0.75)) )    # Quartiles
```

```
##   25%   50%   75%
## 3.850 4.250 4.475
```

Quantile-Normal Plot

Percentage (probability) points - for tied data use the larger percentile

```
xPercent <- cbind("X-value"=xOrdered, "Percentile a=0.0"=ppoints(xOrdered,a=0.0),
                  "Percentile a=0.5"=ppoints(xOrdered,a=0.5),
                  "Percentile a=1.0"=ppoints(xOrdered,a=1.0))
round(xPercent, 2)
```

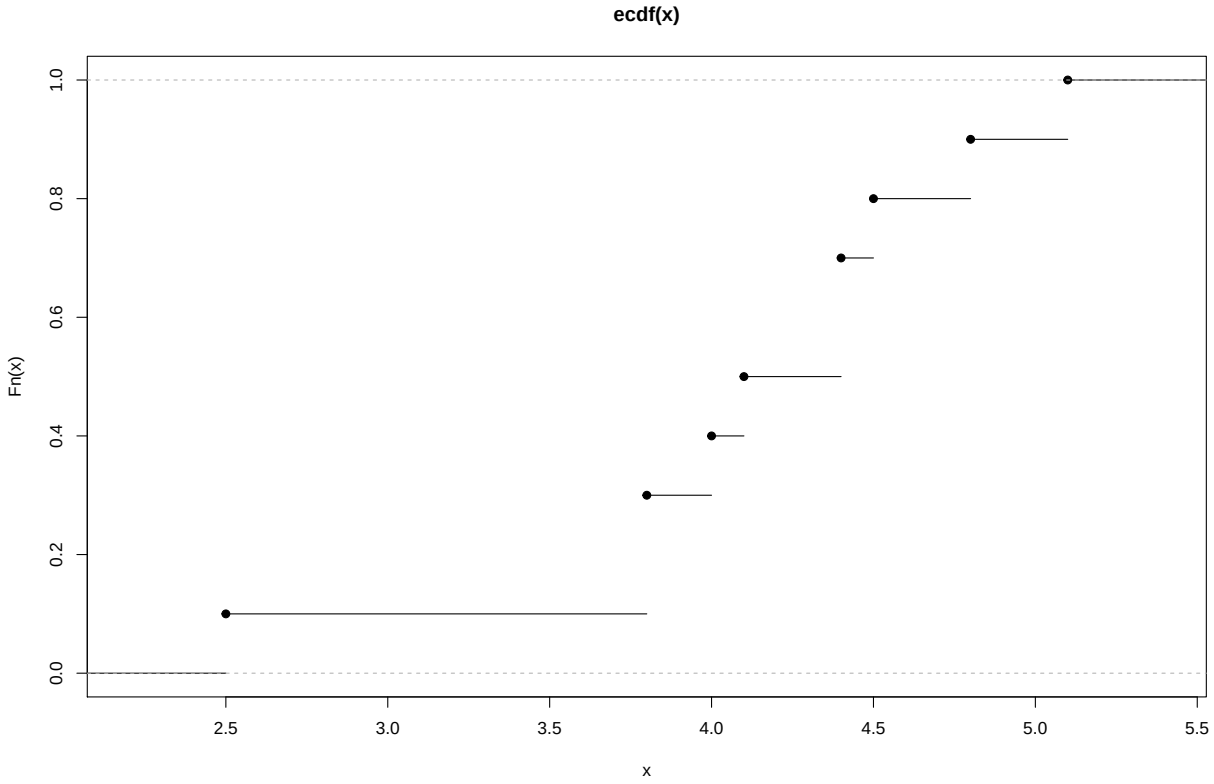
```
##       X-value Percentile a=0.0 Percentile a=0.5 Percentile a=1.0
##  [1,]     2.5             0.09             0.05             0.00
##  [2,]     3.8             0.18             0.15             0.11
##  [3,]     3.8             0.27             0.25             0.22
##  [4,]     4.0             0.36             0.35             0.33
##  [5,]     4.1             0.45             0.45             0.44
##  [6,]     4.4             0.55             0.55             0.56
##  [7,]     4.4             0.64             0.65             0.67
##  [8,]     4.5             0.73             0.75             0.78
##  [9,]     4.8             0.82             0.85             0.89
## [10,]     5.1             0.91             0.95             1.00
```

Empirical Distribution Function

```
summary(ecdf(x))
```

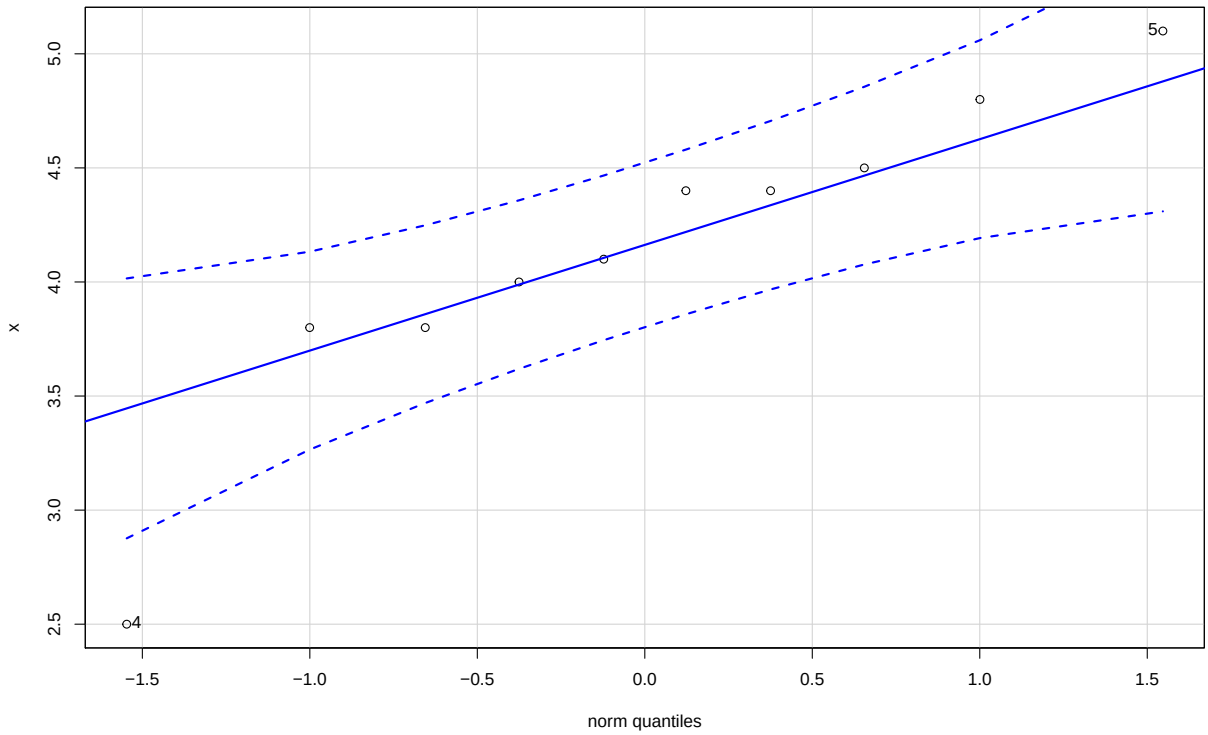
```
## Empirical CDF:     8 unique values with summary
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.500   3.950   4.250   4.150   4.575   5.100
```

```
plot(ecdf(x))
```



QQ plot

```
car::qqPlot(x)
```




```
## [1] 4 5
```

Box Cox Transformation

Initialize data

```
library(car)
```

```
## Loading required package: carData
```

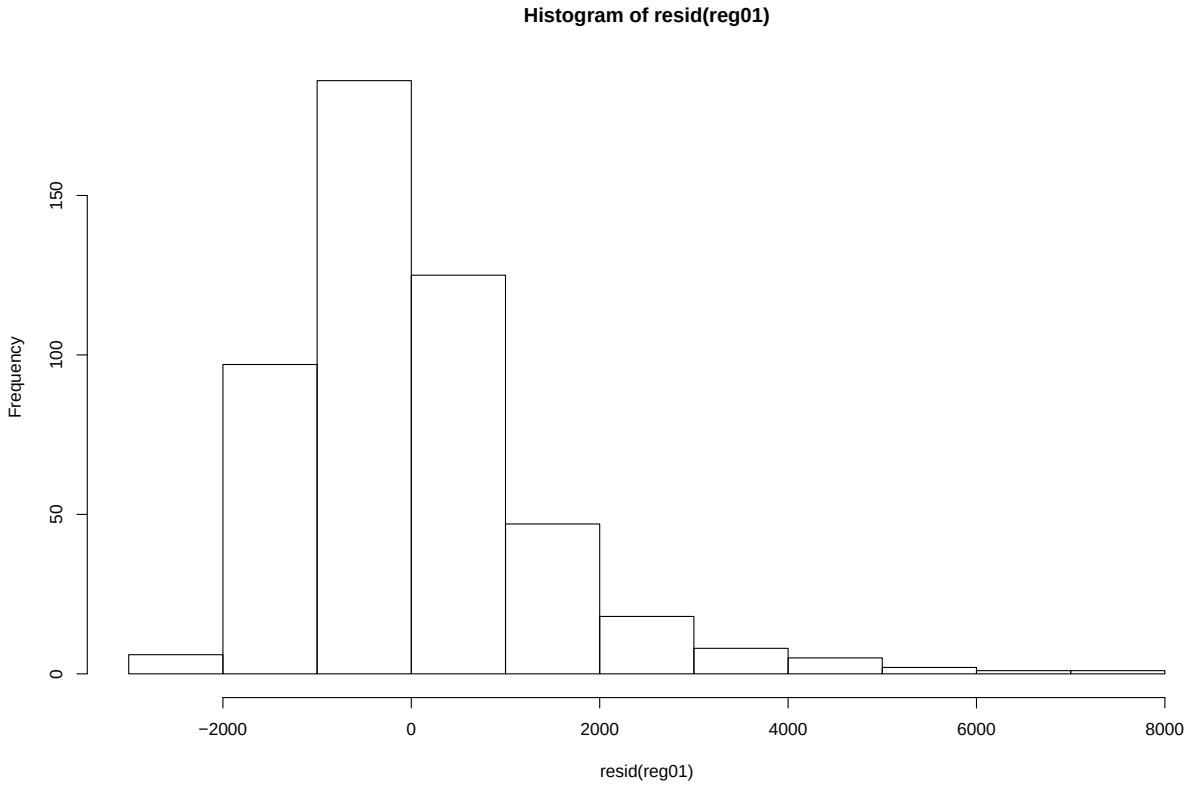
```
setwd("G:\\UTD_Classes\\2020Spring\\GISC7310_AdvancedDataAnalysis\\02Bivariate Regression Analysis")
Concord <- foreign::read.spss("Concord1.sav",to.data.frame=TRUE)
reg01 <- lm(water81~income, data=Concord)
summary(reg01)
```

```
##
## Call:
## lm(formula = water81 ~ income, data = Concord)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2765.3  -889.8  -239.8   536.8  7010.2
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1201.124    123.325    9.74   <2e-16 ***
## income        47.549      4.652   10.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1352 on 494 degrees of freedom
## Multiple R-squared:  0.1745, Adjusted R-squared:  0.1729
## F-statistic: 104.5 on 1 and 494 DF,  p-value: < 2.2e-16
```

Residual Analysis

```
hist(resid(reg01))
```




```
round(sum(resid(reg01)),14)
```

```
## [1] -2.132e-11
```

```
cbind("Coef"=coef(reg01), confint(reg01, level=0.95))
```

```
##                   Coef     2.5 %    97.5 %
## (Intercept) 1201.12436 958.81911 1443.4296
## income        47.54869  38.40798   56.6894
```

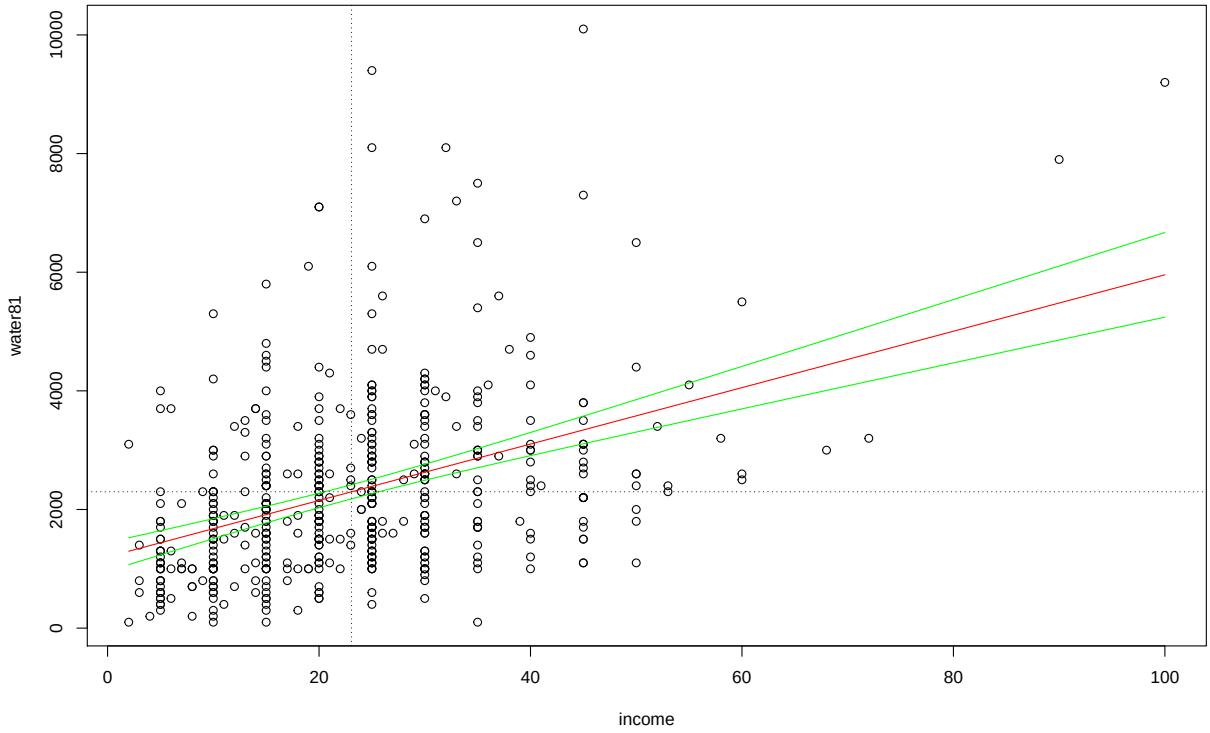
Prediction and Observation

Line confidence interval & fit

```
predDf <- data.frame(income=min(Concord$income):
          max(Concord$income))   # data-frame for independent vars
(predDf <- data.frame(predDf, predict(reg01,
      newdata=predDf, interval="confidence", level=0.95)))[1:10,] # Line confidence interval & fit
```

```
##    income      fit      lwr      upr
## 1       2 1296.222 1069.653 1522.791
## 2       3 1343.770 1124.921 1562.620
## 3       4 1391.319 1180.076 1602.563
## 4       5 1438.868 1235.104 1642.631
## 5       6 1486.417 1289.992 1682.841
## 6       7 1533.965 1344.724 1723.207
## 7       8 1581.514 1399.279 1763.748
## 8       9 1629.063 1453.638 1804.487
## 9      10 1676.611 1507.777 1845.445
## 10     11 1724.160 1561.669 1886.651
```

```
plot(water81~income,data=Concord)
lines(predDf$income,predDf$fit,col="red")  # predicted value
lines(predDf$income,predDf$lwr,col="green") # lower confidence interval limits
lines(predDf$income,predDf$upr,col="green") # upper confidence interval limits
abline(h=mean(Concord$water81),v=mean(Concord$income),lty=3)  # Regression line goes thru the means
```

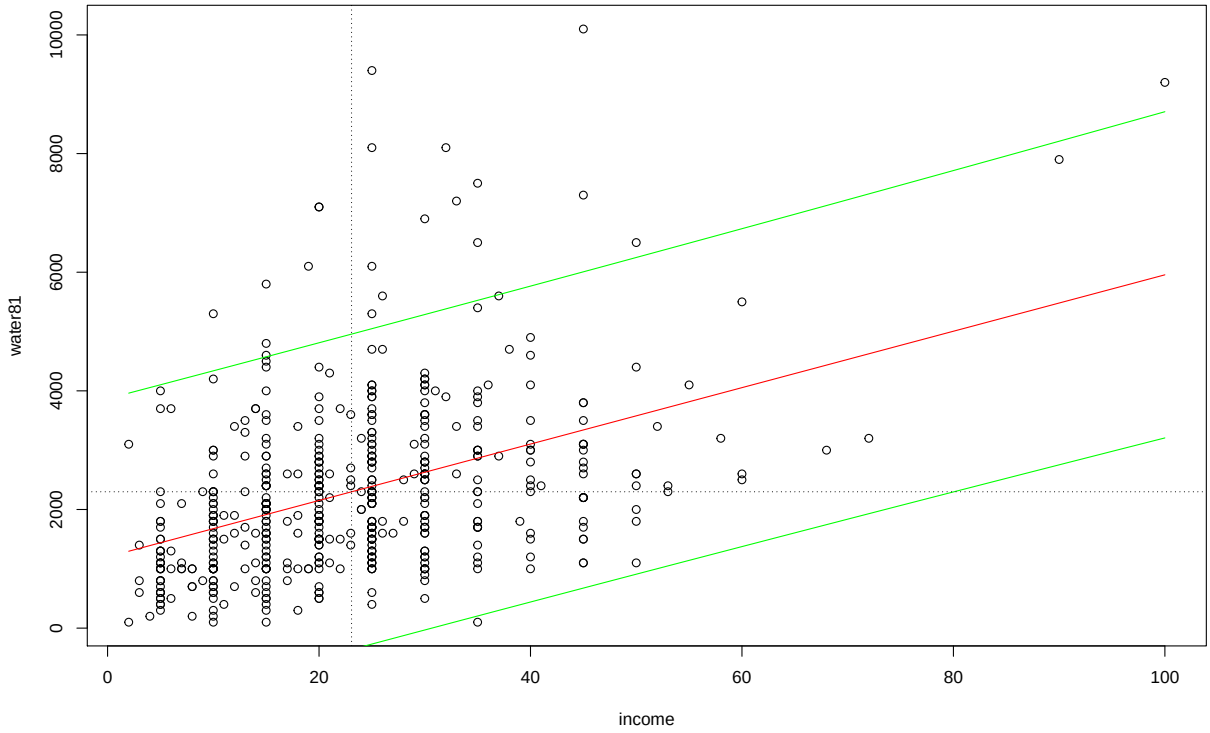


Point confidence interval & fit

```
predDf <- data.frame(income=min(Concord$income):
                       max(Concord$income))   # data-frame for independent vars
(predDf <- data.frame(predDf, predict(reg01,
       newdata=predDf, interval="prediction", level=0.95)))[1:10,]  # Point confidence interval & fit
```

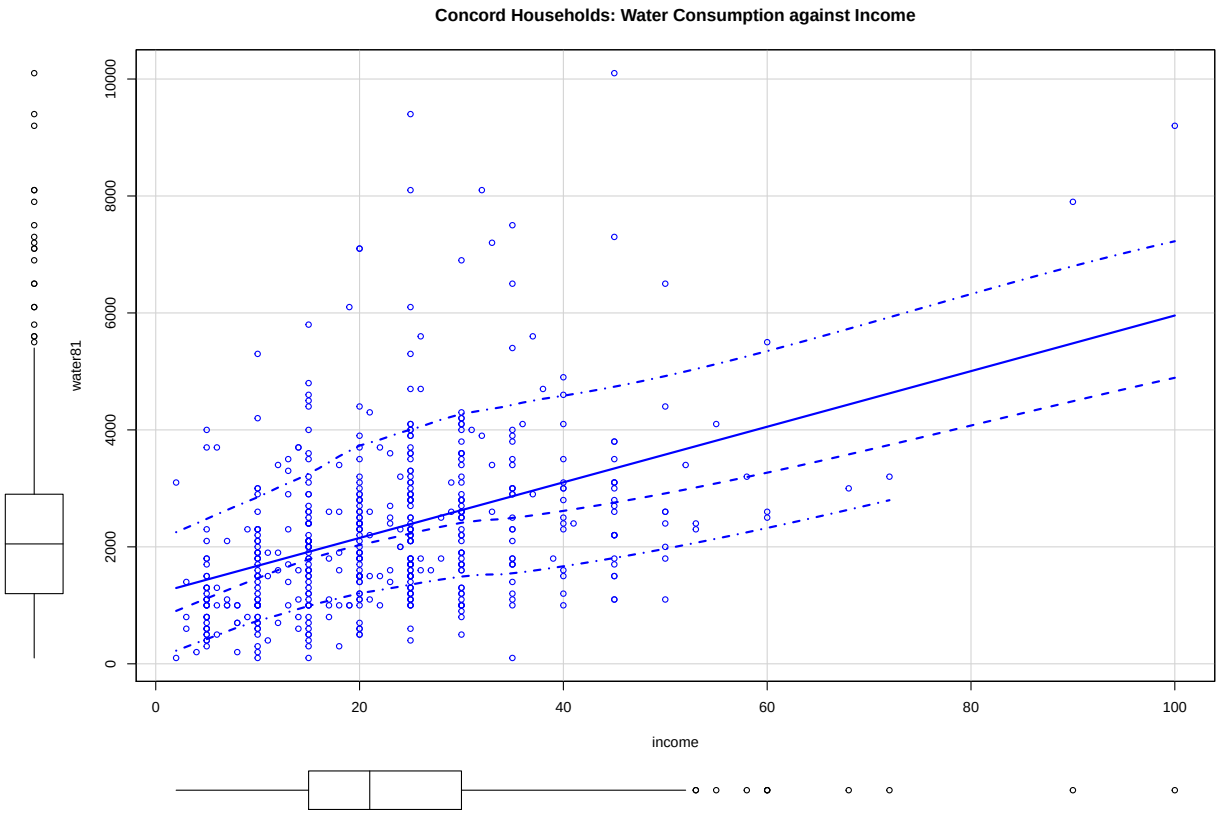
```
##    income      fit        lwr      upr
## 1       2 1296.222 -1368.9723 3961.416
## 2       3 1343.770 -1320.7785 4008.319
## 3       4 1391.319 -1272.6158 4055.254
## 4       5 1438.868 -1224.4844 4102.220
## 5       6 1486.417 -1176.3843 4149.217
## 6       7 1533.965 -1128.3154 4196.246
## 7       8 1581.514 -1080.2778 4243.306
## 8       9 1629.063 -1032.2715 4290.397
## 9      10 1676.611  -984.2966 4337.519
## 10     11 1724.160  -936.3530 4384.673
```

```
plot(water81~income,data=Concord)
lines(predDf$income,predDf$fit,col="red")    # predicted value
lines(predDf$income,predDf$lwr,col="green") # lower confidence interval limits
lines(predDf$income,predDf$upr,col="green") # upper confidence interval limits
abline(h=mean(Concord$water81),
       v=mean(Concord$income),lty=3) # Regression line goes thru the means
```



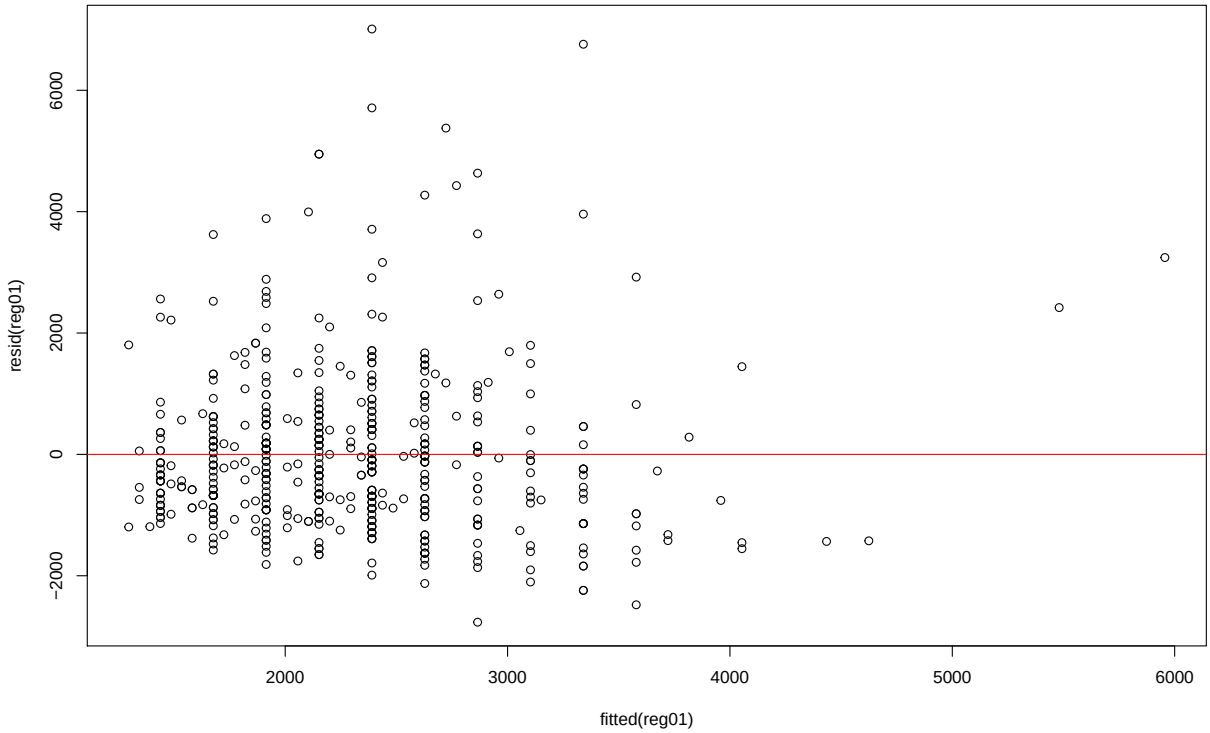
Scatter Plot

```
car::scatterplot(water81~income, data=Concord,
                 main="Concord Households: Water Consumption against Income" )
```



The residual variance is not constant and mean are not equal to 0

```
plot(resid(reg01)~fitted(reg01))
abline(h=0,col= "red")
```



Box Cox Transformation

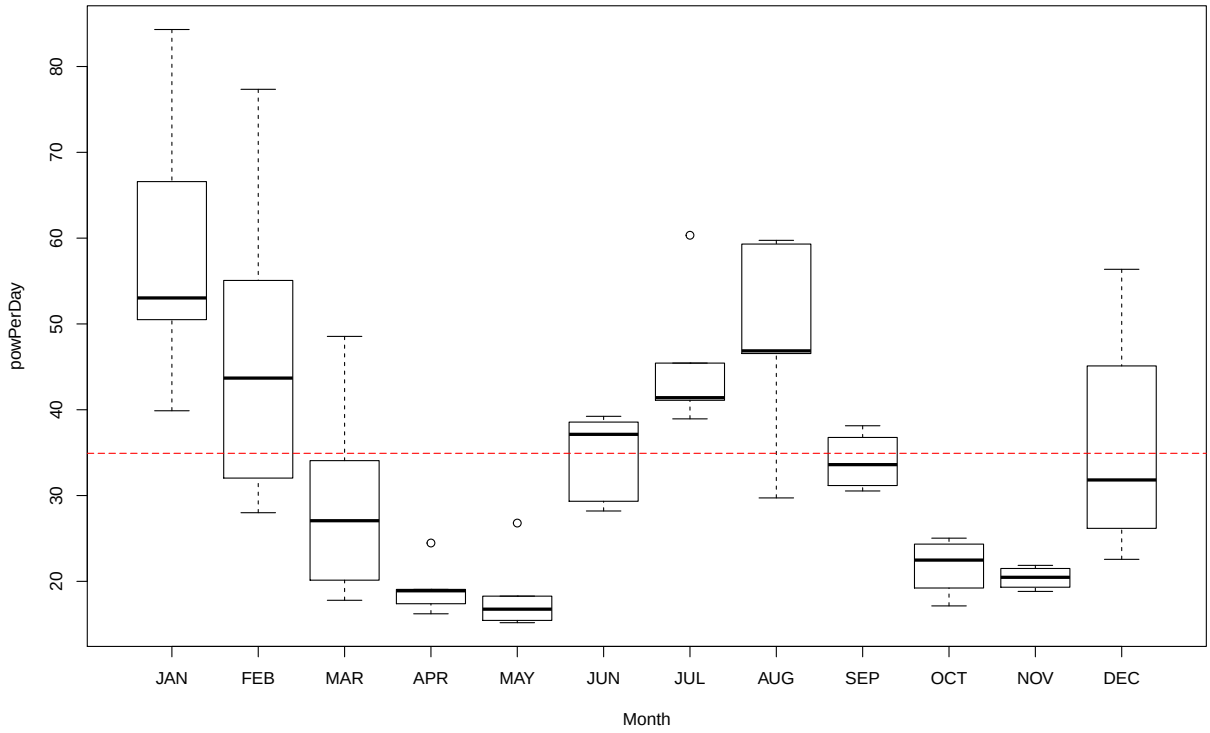
Initialize data

```
setwd("G:\\UTD_Classes\\2020Spring\\GISC7310_AdvancedDataAnalysis\\02Bivariate Regression Analysis")
library(foreign); library(car)
myPower <- read.spss("DallasTempPower.sav", to.data.frame= TRUE)

myPower$powPerDay <- myPower$kWhBill/myPower$DaysBill # calculate kWh per day

## Exploration
summary(myPower)
```

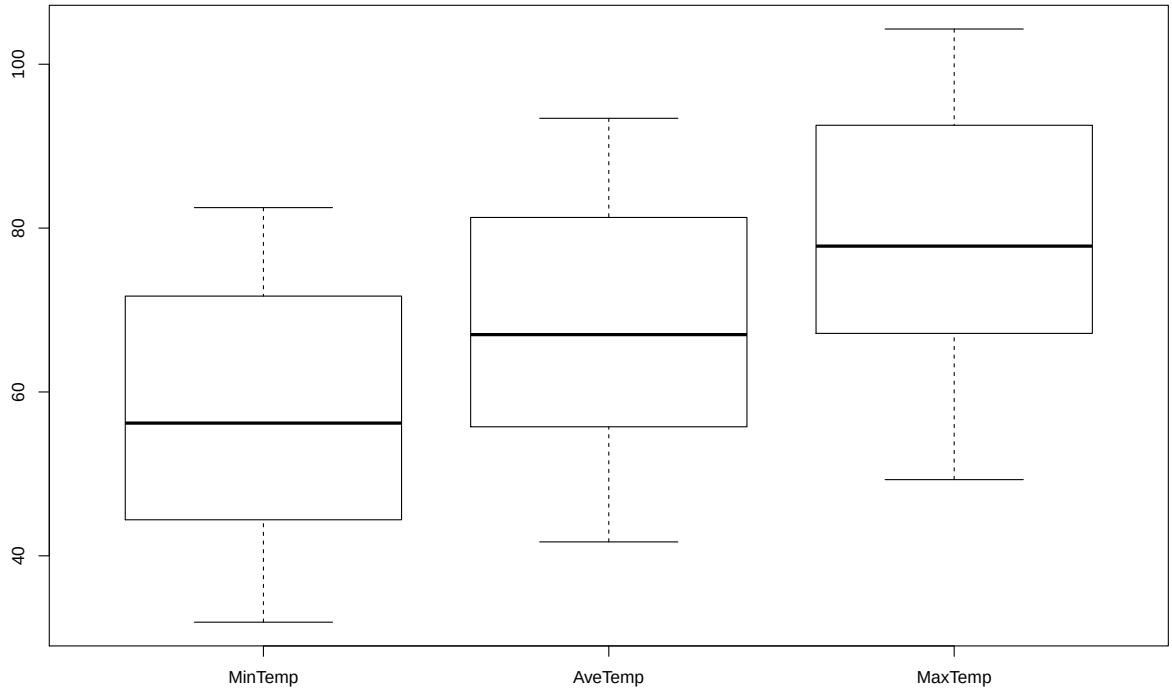
```
##      SeqID            Year          Month       MinTemp        AveTemp
##  Min.   : 1.00   Min.   :2009   JAN    : 5   Min.   :31.9   Min.   :41.70
##  1st Qu.:14.75   1st Qu.:2010   FEB    : 5   1st Qu.:44.4   1st Qu.:55.75
##  Median :28.50   Median :2011   MAR    : 5   Median :56.2   Median :67.00
##  Mean   :28.50   Mean   :2011   APR    : 5   Mean   :57.0   Mean   :67.57
##  3rd Qu.:42.25   3rd Qu.:2012   MAY    : 5   3rd Qu.:71.7   3rd Qu.:81.30
##  Max.   :56.00   Max.   :2013   JUN    : 5   Max.   :82.5   Max.   :93.40
##                                 (Other):26   NA's   :1      NA's   :1
##     MaxTemp          kWhBill          DaysBill       powPerDay
##  Min.   : 49.30   Min.   : 448.0   Min.   :28.00   Min.   :15.18
```

```
##  1st Qu.: 67.15   1st Qu.: 617.0   1st Qu.:29.00   1st Qu.:21.27
##  Median : 77.80   Median : 941.5   Median :30.00   Median :31.17
##  Mean   : 78.08   Mean   :1074.3   Mean   :30.48   Mean   :34.92
##  3rd Qu.: 92.55   3rd Qu.:1351.5   3rd Qu.:32.00   3rd Qu.:44.13
##  Max.   :104.30   Max.   :2951.0   Max.   :35.00   Max.   :84.31
##  NA's   :1
```

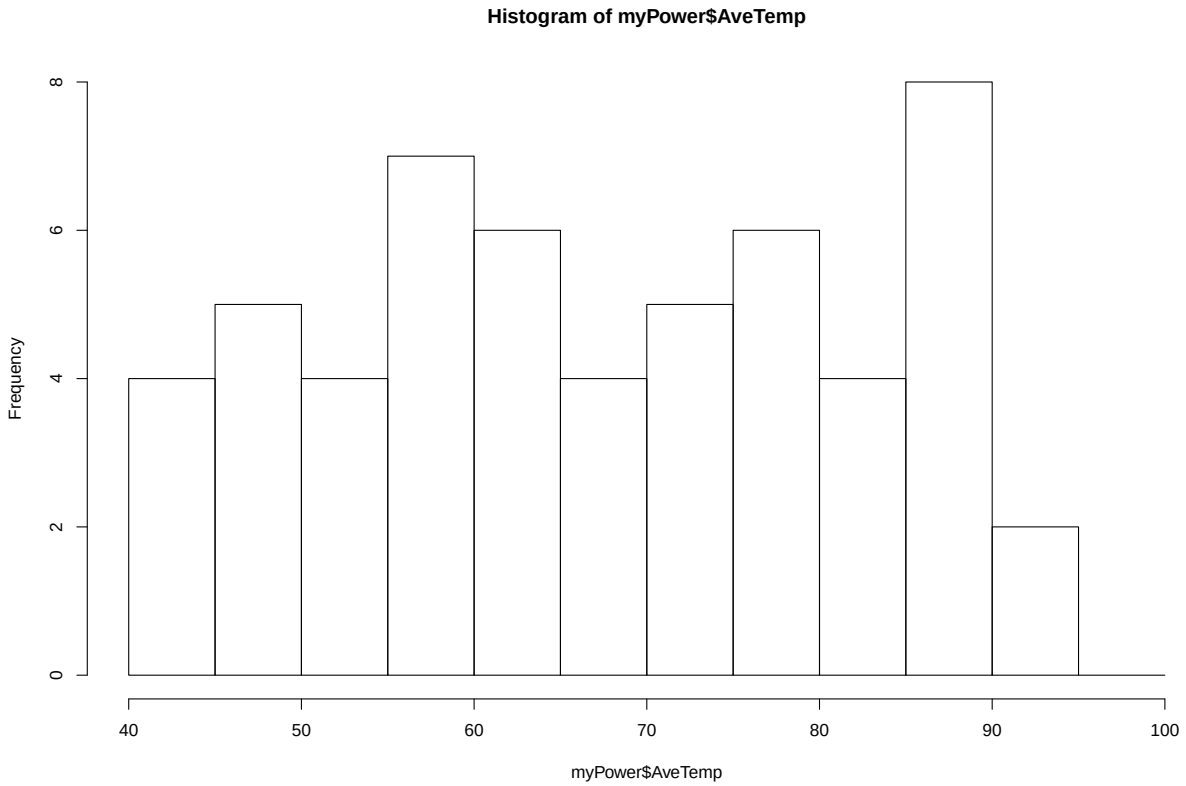
```r
boxplot(powPerDay~Month, data=myPower)
abline(h=mean(myPower$powPerDay, na.rm=TRUE), lty=5, col="red")
```




```r
boxplot(myPower[, c("MinTemp","AveTemp","MaxTemp")])
```




```
hist(myPower$AveTemp, breaks=seq(40,100, by=5))
```



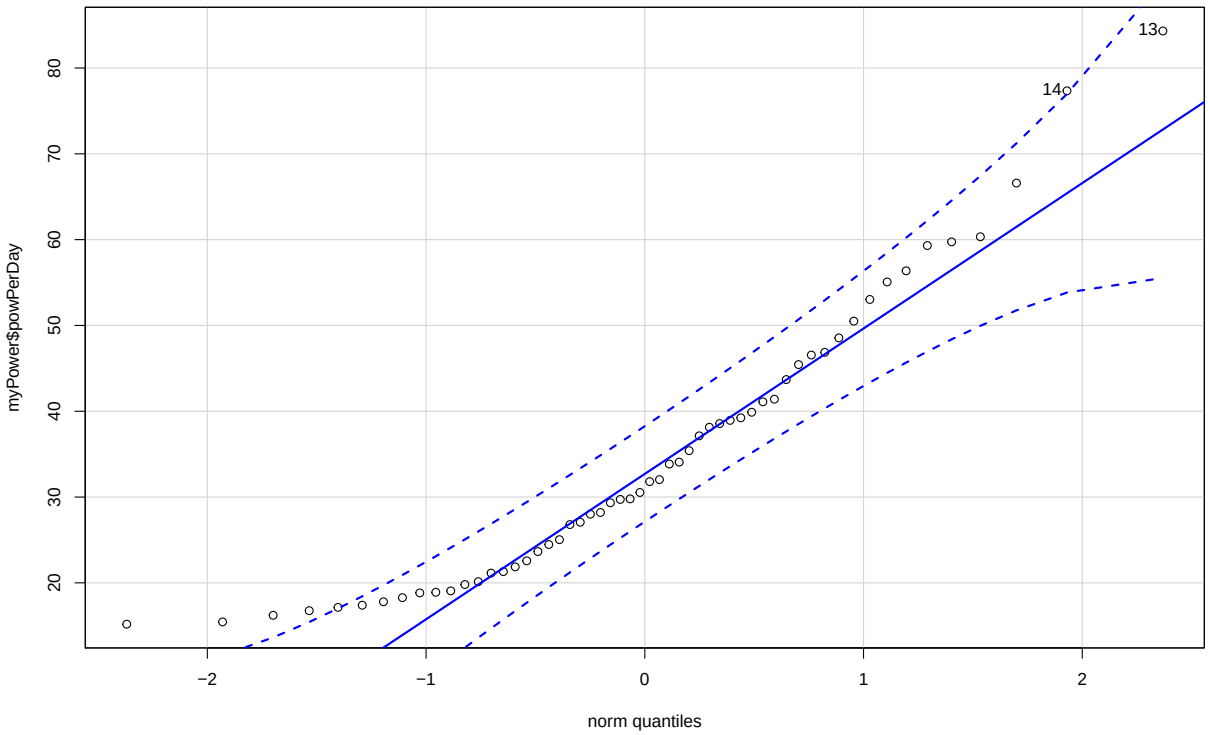
Box Cox Transformation

Check normality

```
e1071::skewness(myPower$powPerDay, na.rm=TRUE)
```

```
## [1] 0.9458044
```

```
car::qqPlot(myPower$powPerDay)
```




```
## [1] 13 14
```

Two methods for testing

```
shapiro.test(myPower$powPerDay)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  myPower$powPerDay
## W = 0.91478, p-value = 0.0007546
```

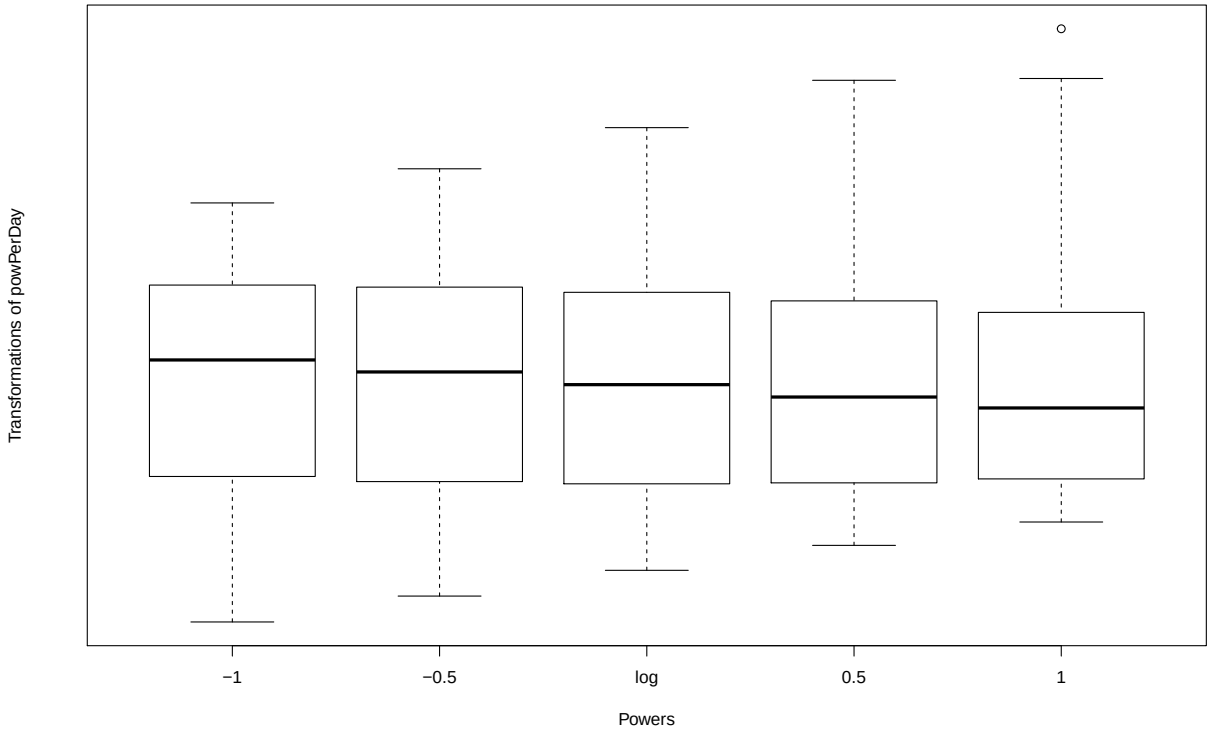
```
ks.test(myPower$powPerDay, pnorm,                    # the ks test has not as much power
        mean=mean(myPower$powPerDay), sd=sd(myPower$powPerDay))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  myPower$powPerDay
## D = 0.11274, p-value = 0.4427
## alternative hypothesis: two-sided
```

Find Box-Cox lambda

Explore different lambda parameters

```
symbox(~powPerDay, data=myPower)
```



Test indicates log-transformation sufficient

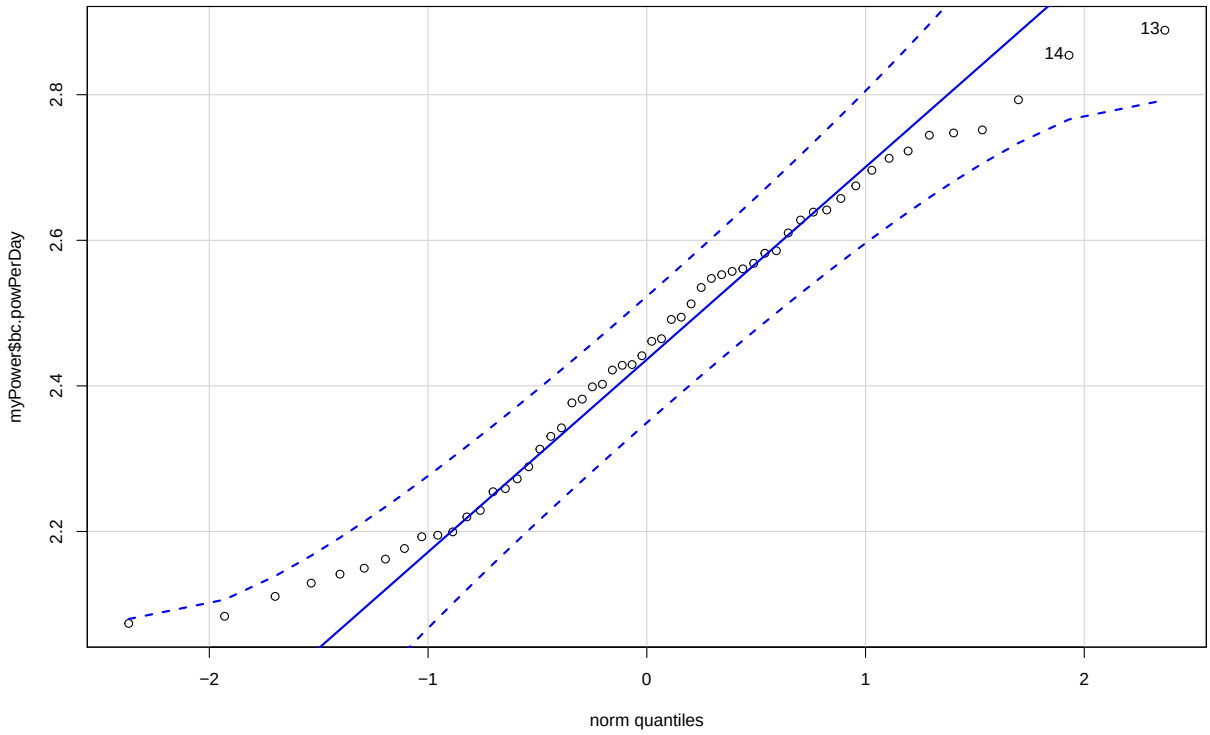
```
summary(powerTransform(lm(powPerDay~1, data=myPower)))
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts argument
## ignored

## bcPower Transformation to Normality
##    Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
## Y1   -0.2094           0      -0.8233       0.4045
##
## Likelihood ratio test that transformation parameter is equal to 0
##  (log transformation)
##                             LRT df    pval
## LR test, lambda = (0) 0.4524691  1 0.50116
##
## Likelihood ratio test that no transformation is needed
##                            LRT df       pval
## LR test, lambda = (1) 15.70194  1 7.4148e-05
```

```
lambda <- powerTransform(lm(powPerDay~1, data=myPower))$lambda
myPower$bc.powPerDay <- car::bcPower(myPower$powPerDay, lambda=lambda)
e1071::skewness(myPower$bc.powPerDay)
```

```
## [1] 0.03050209
```

```
car::qqPlot(myPower$bc.powPerDay)
```



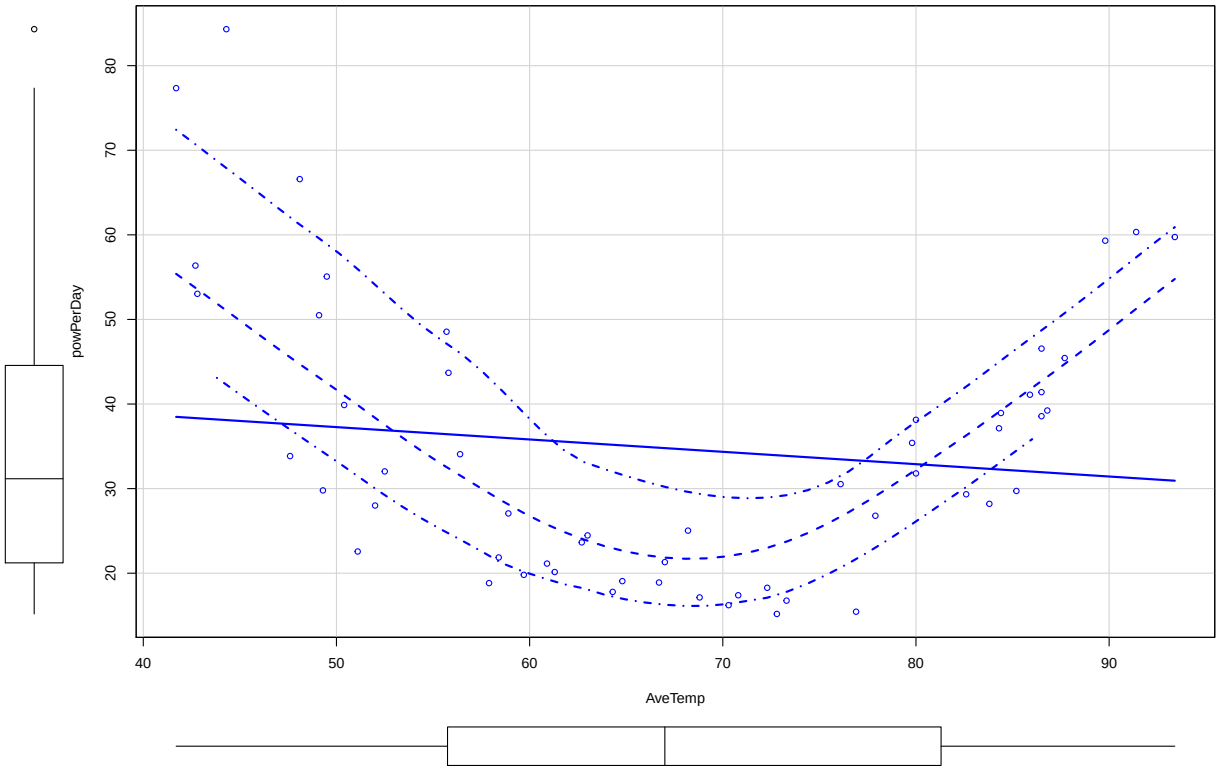

```
## [1] 13 14
```

```
shapiro.test(myPower$bc.powPerDay)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  myPower$bc.powPerDay
## W = 0.96945, p-value = 0.1659
```

Scatterplot with loess smoother

```
scatterplot(powPerDay~AveTemp, data=myPower)
```



Simultaneously transform a set of variables

```
summary(lambda <- powerTransform(lm(cbind(powPerDay,AveTemp)~1, data=myPower)))
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts argument
## ignored
```

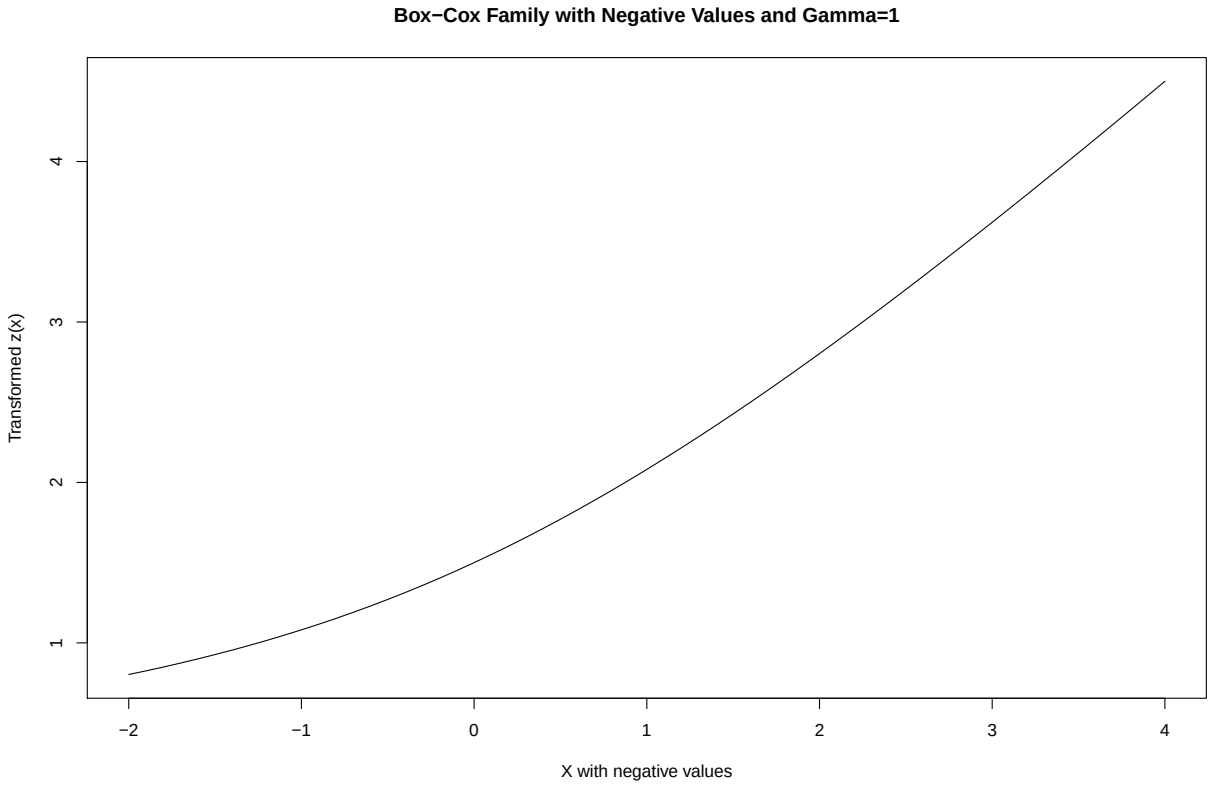
```
## bcPower Transformations to Multinormality
##            Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
## powPerDay    -0.2284           0      -0.8547       0.3979
## AveTemp       0.6093           1      -0.8203       2.0389
##
## Likelihood ratio test that transformation parameters are equal to 0
##  (all log transformations)
##                                LRT df    pval
## LR test, lambda = (0 0) 1.154093  2 0.56155
##
## Likelihood ratio test that no transformations are needed
##                                LRT df       pval
## LR test, lambda = (1 1) 16.14737  2 0.00031163
```

```
myPower <- data.frame(myPower,bcPower(cbind(myPower$powPerDay,myPower$AveTemp),
                                      coef(lambda, round=T))) # add transformed variables to myPower
```

Box Cox Transformation [Negative Values]

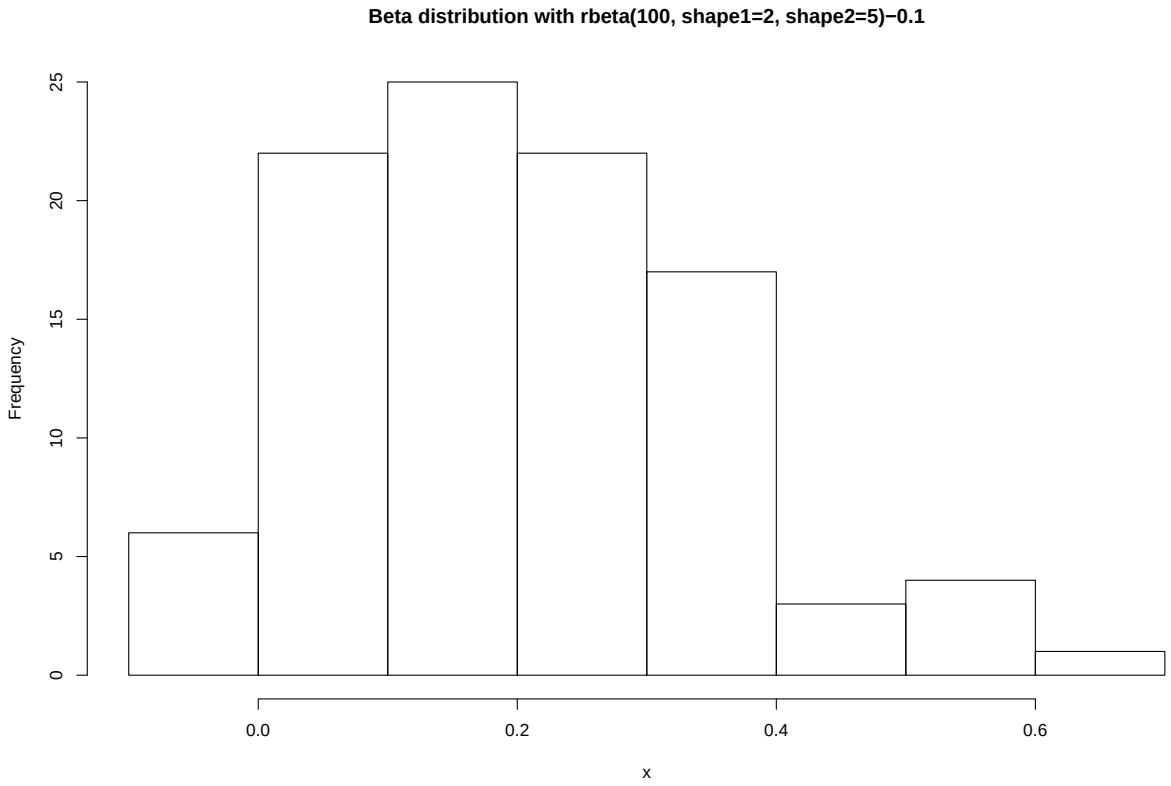
Z-Gamma Transformation

```
## Example of the z-Gamma transformation
zGamma <- function(x, gamma){(x+sqrt(x^2+gamma^2))/2}
x <- seq(-2,4, by=0.1)
gamma <- 3
zx <- zGamma(x, gamma)
plot(x,zx, type="l", xlab="X with negative values", ylab="Transformed z(x)",
     main="Box-Cox Family with Negative Values and Gamma=1")
```



An positvely skewed distribution with small negative value

```
x <- rbeta(100, shape1=2, shape2=5)-0.1
hist(x, main="Beta distribution with rbeta(100, shape1=2, shape2=5)-0.1")
```



Use powerTransform

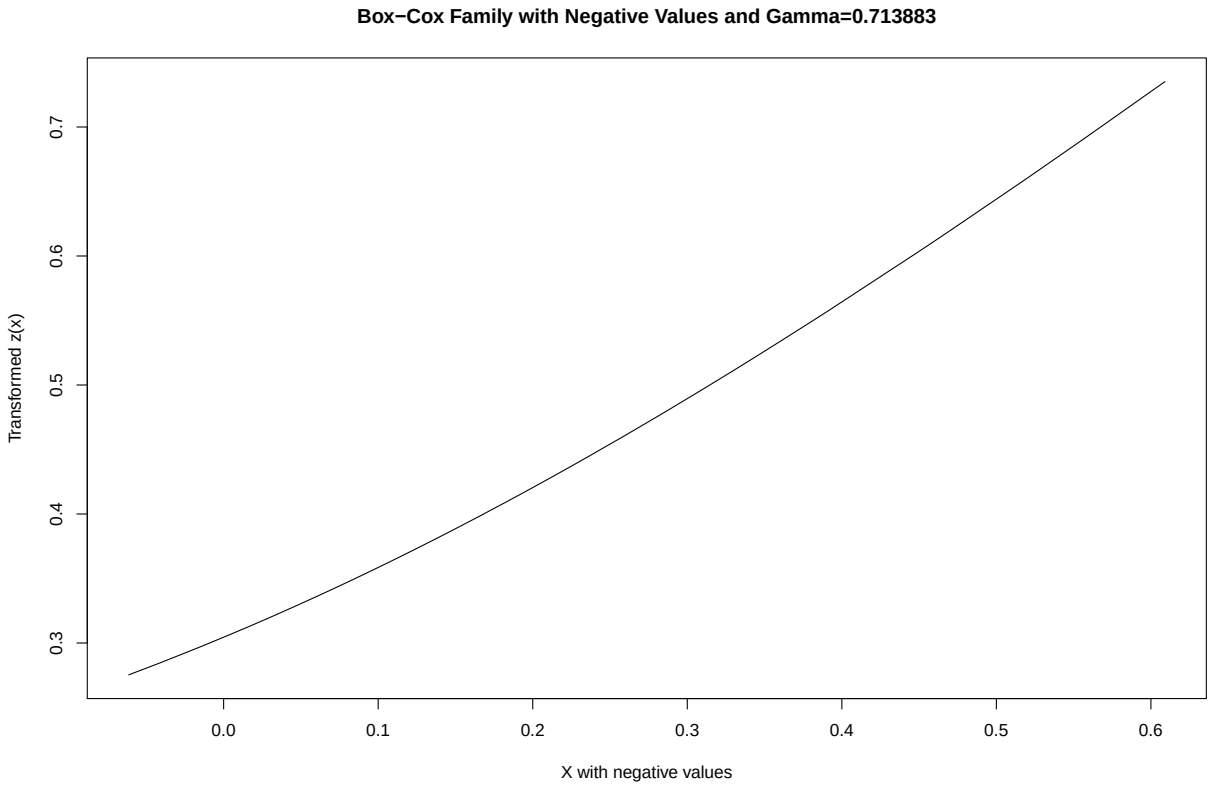
```
summary(lambda <- powerTransform(x~1, family="bcnPower"))
```

```
## bcnPower transformation to Normality
##
## Estimated power, lambda
##    Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
## Y1   -0.4885           1      -2.2802       1.3032
##
## Estimated location, gamma
##    Est gamma Std Err. Wald Lower Bound Wald Upper Bound
## Y1     0.609   0.5405                0           1.6683
##
## Likelihood ratio tests about transformation parameters
##                              LRT df         pval
## LR test, lambda = (0)  1.321648  1 0.2502964122
## LR test, lambda = (1) 12.656209  1 0.0003743205
```

Z-Gamma Transformation

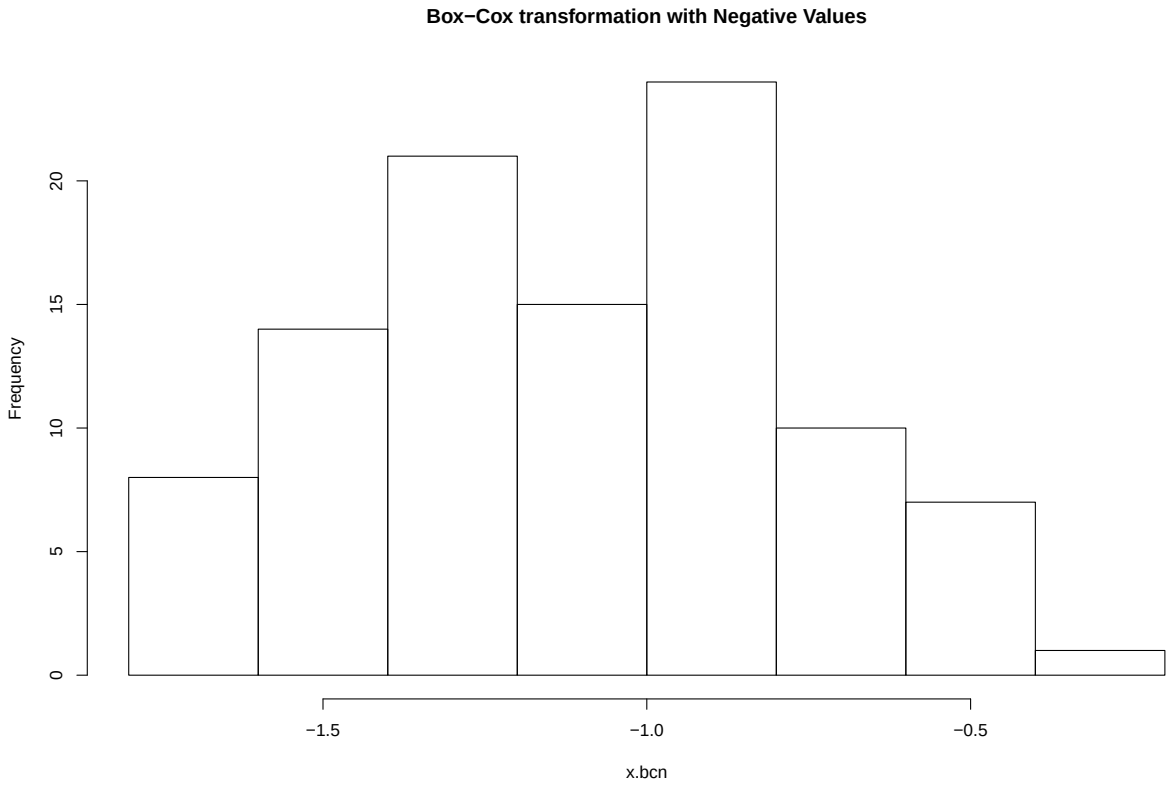
```
x <- sort(x)
zx <- zGamma(x,coef(lambda)[2])
```

```
plot(x,zx, type="l", xlab="X with negative values", ylab="Transformed z(x)",
     main="Box-Cox Family with Negative Values and Gamma=0.713883")
```



Box Cox Transformation

```
x.bcn <- bcnPower(x, lambda=coef(lambda)[1], gamma=coef(lambda)[2])
hist(x.bcn, main="Box-Cox transformation with Negative Values")
```


Box−Cox transformation with Negative Values
Frequency
0
5
10
15
20
−1.5
−1.0
−0.5
x.bcn