GU4205/5205-Linear Regression Models-Lab1b

Authors: Dr. Banu Baydil, Dr. Ronald Neath, Dr. Gabriel Young

Fall 2022

Section 1: Fitting a Linear Model

In this section we will work with the ufcwc dataset (Western red cedar trees) from STAT GU4205/5205 course textbook. You can get the dataset from the course textbook website directly, or through first installing the alr4 package.

First, install alr4 package, using the command **install.packages("alr4")**. You only need to install the package once; better to do this in the command window, than within the RMarkdown file.

Next, load the alr4 package using the command **library(alr4)**. You will need to load the package every time you will use it. The system will also load other packages needed to run the alr4 package.

```
library(alr4)
```

Let's study the data first:

```
dim(ufcwc)  # tells us we have 139 records, and 5 variables
```

```
## [1] 139   5
```

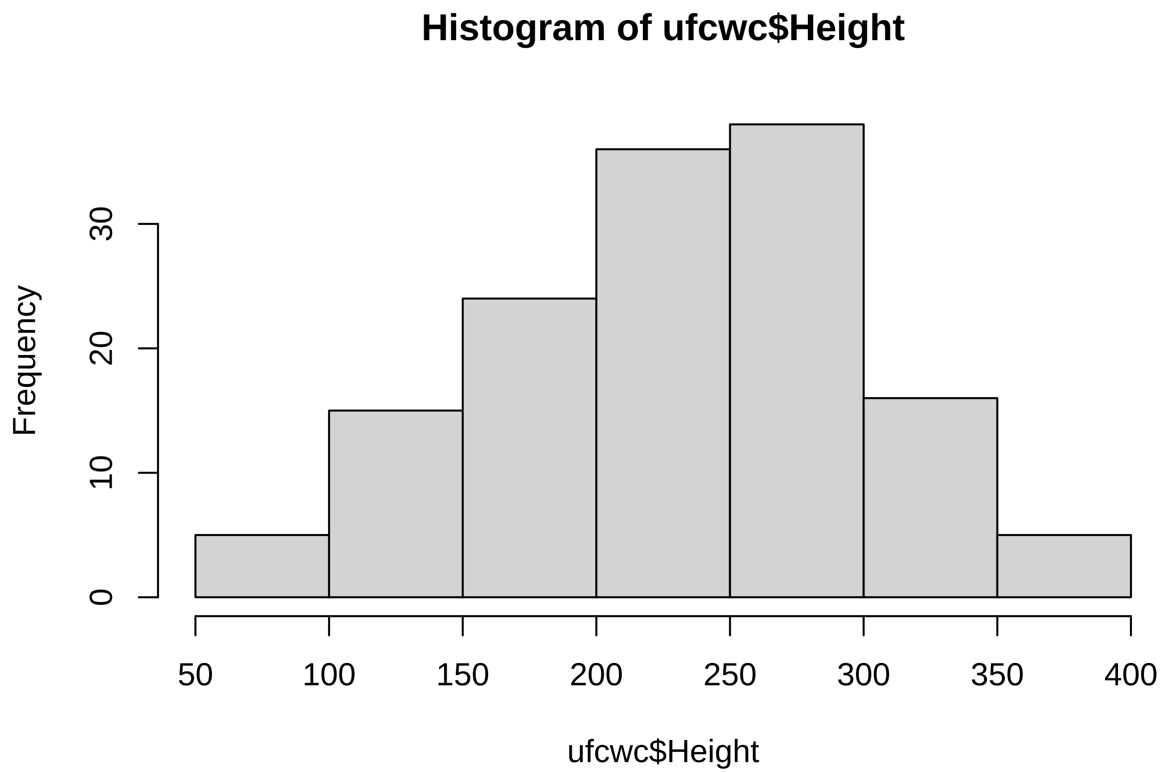
```
names(ufcwc)  # tells us what the variables are
```

```
## [1] "Plot"    "Tree"    "Species" "Dbh"     "Height"
```

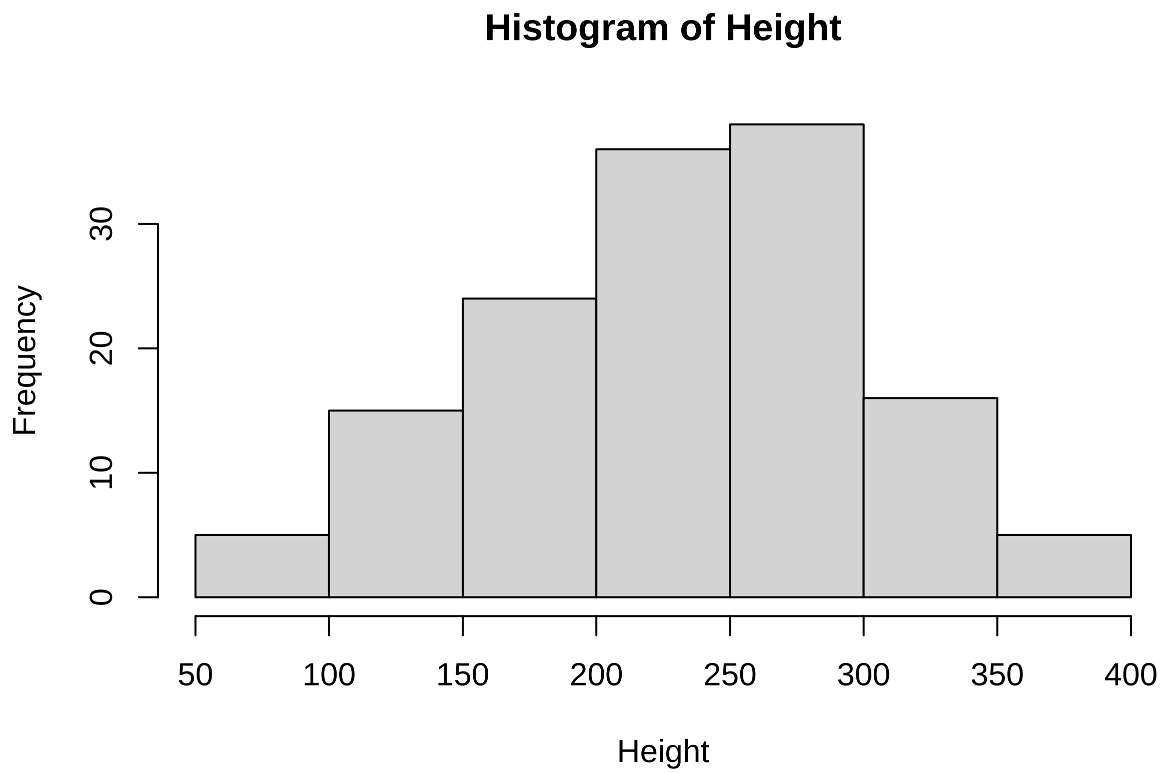
```
summary(ufcwc)  # summary statistics
```

```
##       Plot             Tree         Species        Dbh              Height
##  Min.   :  3.00   Min.   : 1.000   WC:139   Min.   : 101.0   Min.   : 90.0
##  1st Qu.: 33.50   1st Qu.: 1.000            1st Qu.: 260.0   1st Qu.:183.5
##  Median : 57.00   Median : 2.000            Median : 377.0   Median :245.0
##  Mean   : 62.75   Mean   : 2.561            Mean   : 388.4   Mean   :234.9
##  3rd Qu.: 93.50   3rd Qu.: 3.000            3rd Qu.: 492.0   3rd Qu.:285.0
##  Max.   :143.00   Max.   :10.000            Max.   :1015.0   Max.   :400.0
```

```
hist(ufcwc$Height)  # histogram of tree heights
```



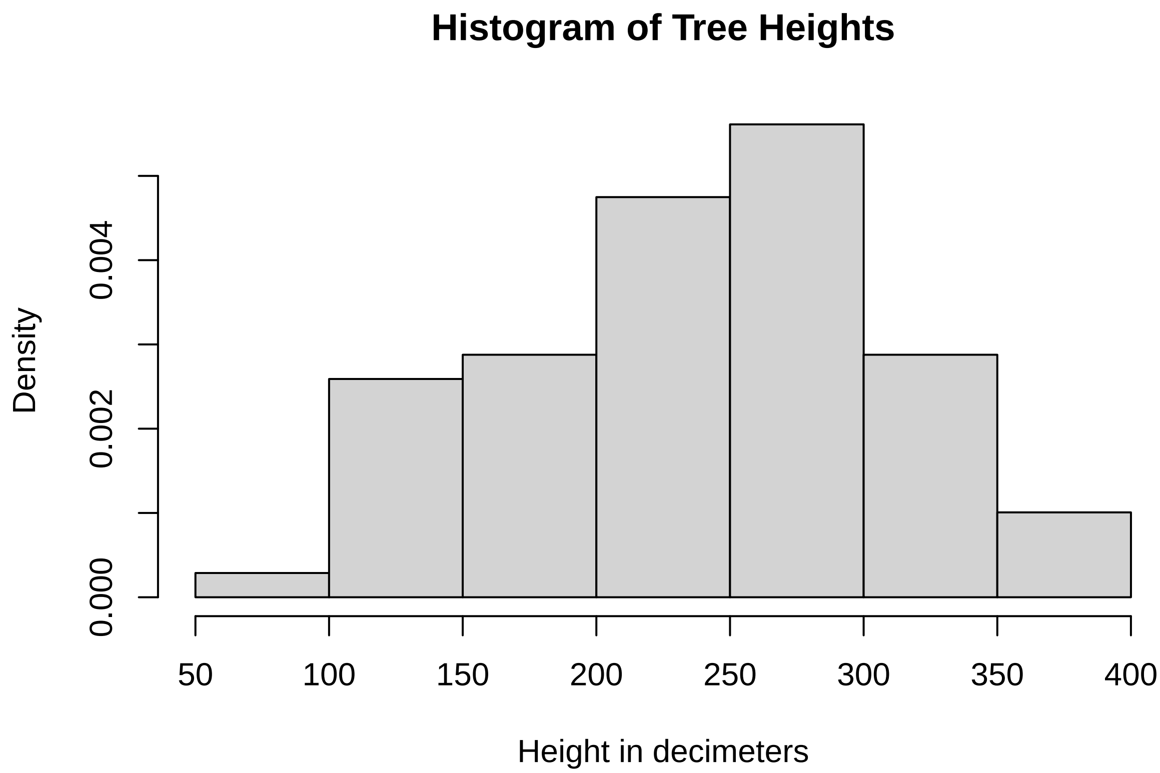

```
attach(ufcwc) #allows variables in this object to be called directly by their names
hist(Height) #so that now we can do this
```



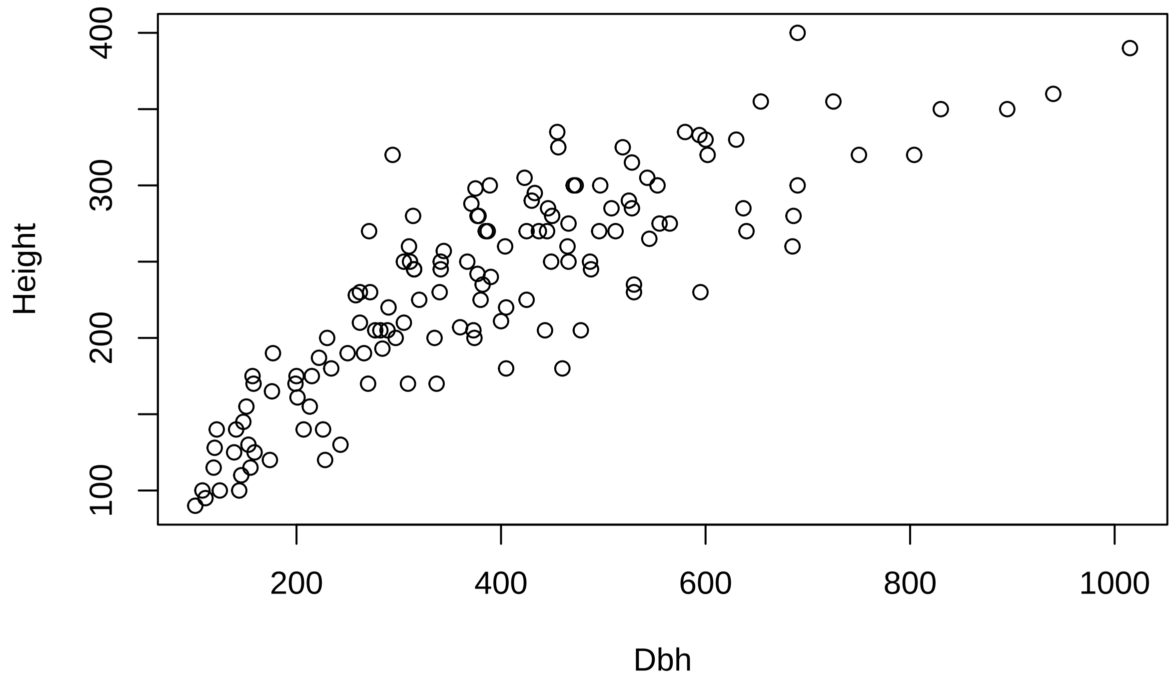

```
help(hist)  # open, in a browser window, help page for the hist() function
```

```
## starting httpd help server ... done
```

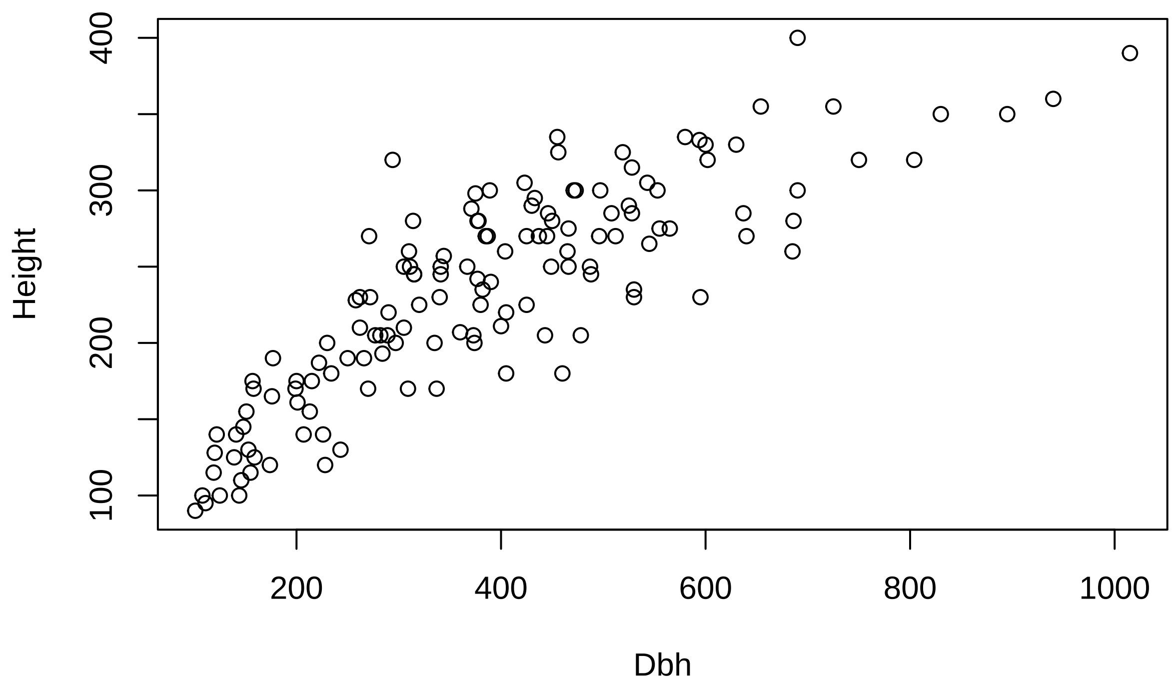
```
hist(Height, freq=F, right=F, xlab="Height in decimeters",
     main="Histogram of Tree Heights")
```




```
plot(Dbh, Height)  # one way to get scatter plot of Height versus Diameter
```




```
plot(Height ~ Dbh)  # another way to get scatter plot of Height versus Diameter
```



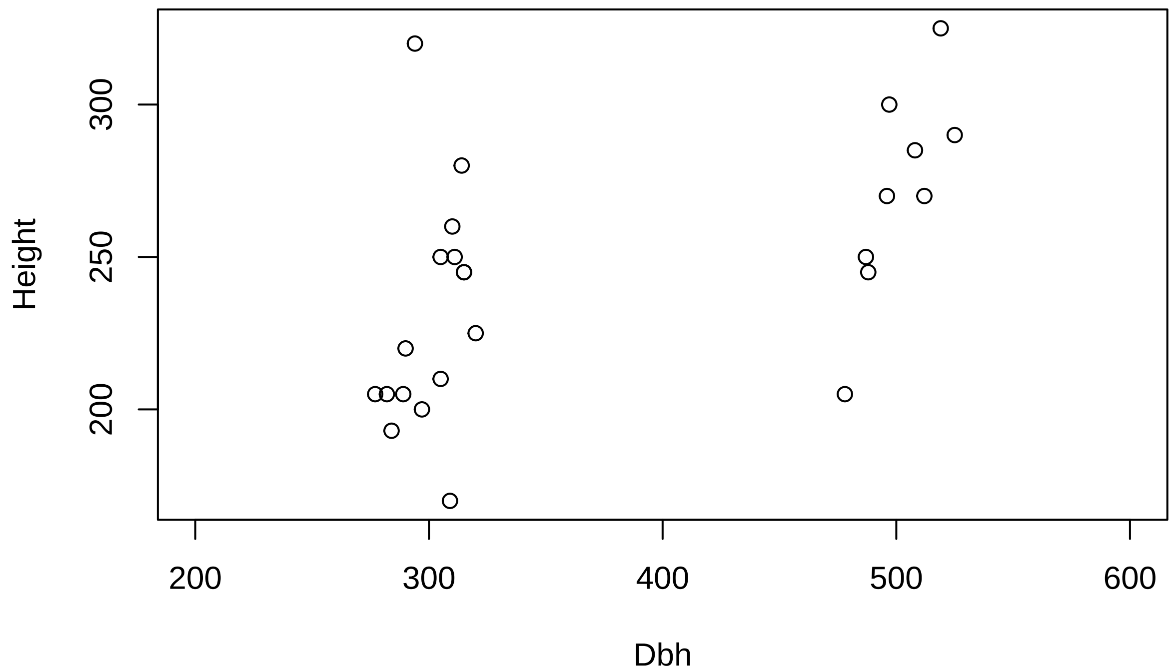

```
# Let's compare the heights of trees with 500mm diameter to those of trees with 300mm diameter.
# We first select those records corresponding to trees with diameter close to 300 (or 500).
sel.300 <- Dbh >= 275 & Dbh <= 325
sel.500 <- Dbh >= 475 & Dbh <= 525
summary(Height[sel.300])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   170.0   205.0   222.5   230.2   250.0   320.0
```

```
summary(Height[sel.500])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   205.0   250.0   270.0   271.1   290.0   325.0
```

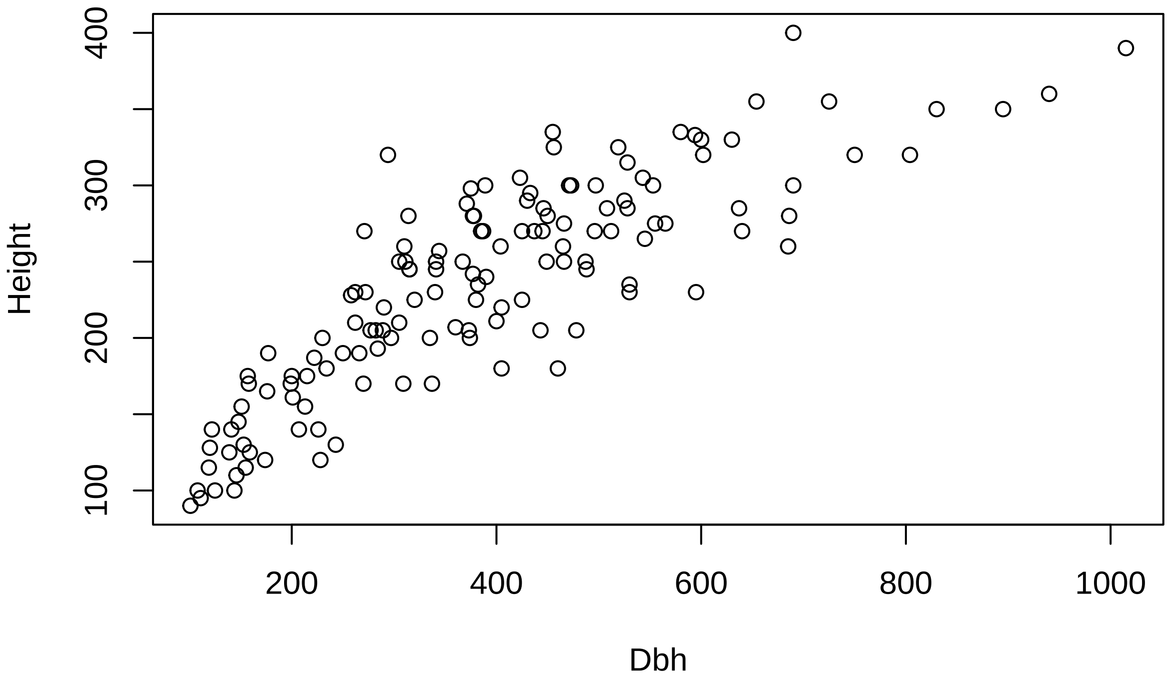
```
# Indeed, mean and median heights of the fatter trees exceed those associated with the skinnier trees.
# Scatter plot, only for selected subsets of the data:
plot(Height~Dbh,data=ufcwc[sel.300|sel.500,],xlim=c(200,600))
```



Fitting a linear model:

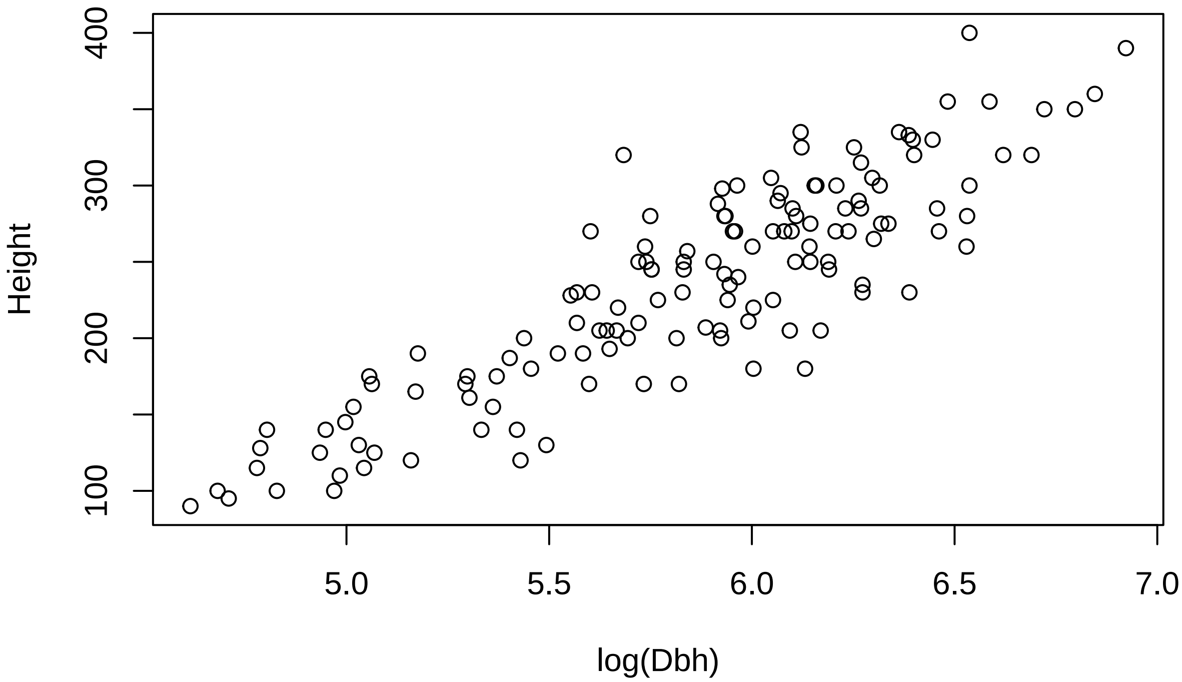
How can we estimate the mean function E(Height|Dbh=x)?

```
plot(Height ~ Dbh)
```



Clearly it is not a straight line. We can not fit a linear regression model using the original predictors in this problem as regressors. Let's try a transformation of the variables:

```
plot(Height ~ log(Dbh), data=ufcwc)
```



Since the form of the above scatter plot is linear, we can fit a linear model to the dataset with Height as the response and log(Dbh) as the explanatory (predictor) variable. We will do this by using the lm() function, and call our model model1. To get detailed information about our model we will use the summary() function. In the Estimate column the first row entry corresponds to the estimate of the intercept of the line of best fit, and the second row entry corresponds to the estimate of the slope of the line of best fit. You can alternatively get this information by using the coef() function.

```
model1=lm(Height~log(Dbh))
summary(model1)
```

```
##
## Call:
## lm(formula = Height ~ log(Dbh))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -89.485 -20.046   3.652  22.586 104.017
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -463.314     32.438  -14.28   <2e-16 ***
## log(Dbh)     119.519      5.532   21.61   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33.33 on 137 degrees of freedom
## Multiple R-squared:  0.7731, Adjusted R-squared:  0.7715
```

```
## F-statistic: 466.8 on 1 and 137 DF,  p-value: < 2.2e-16
```

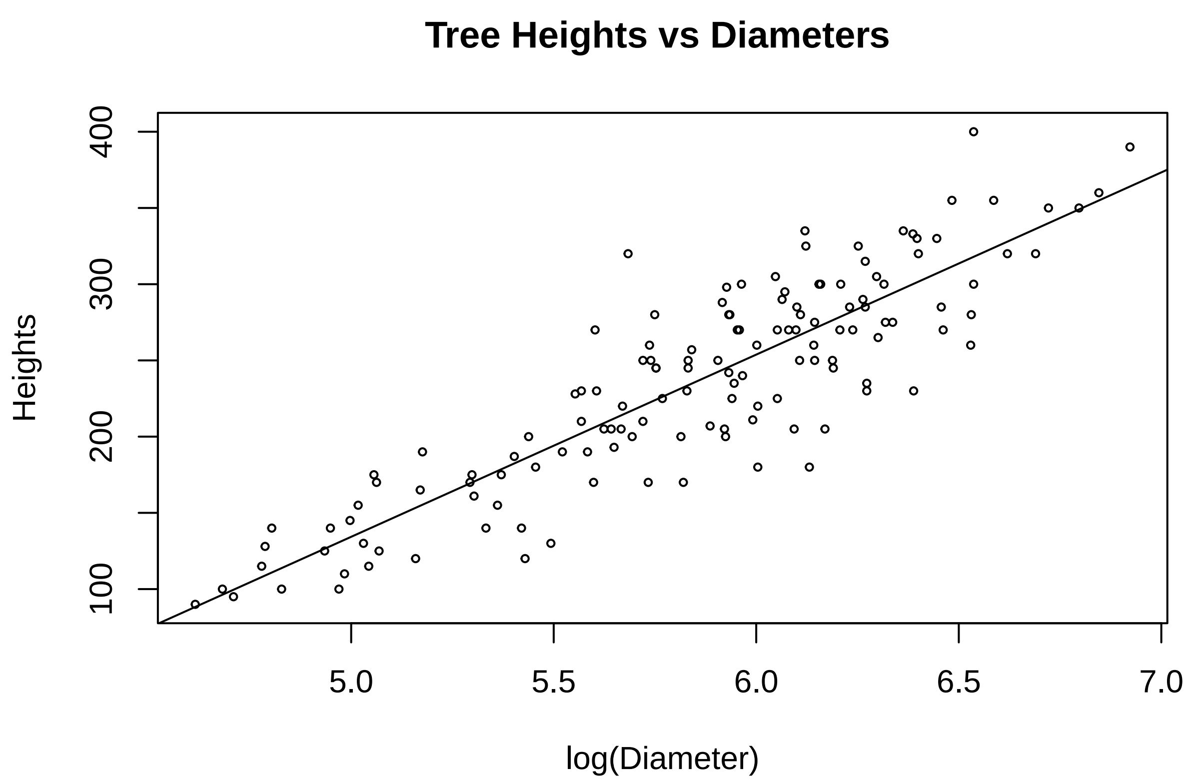
Alternate way of getting coefficient estimates:

```
coef(model1)
```

```
## (Intercept)    log(Dbh)
##   -463.3144    119.5192
```

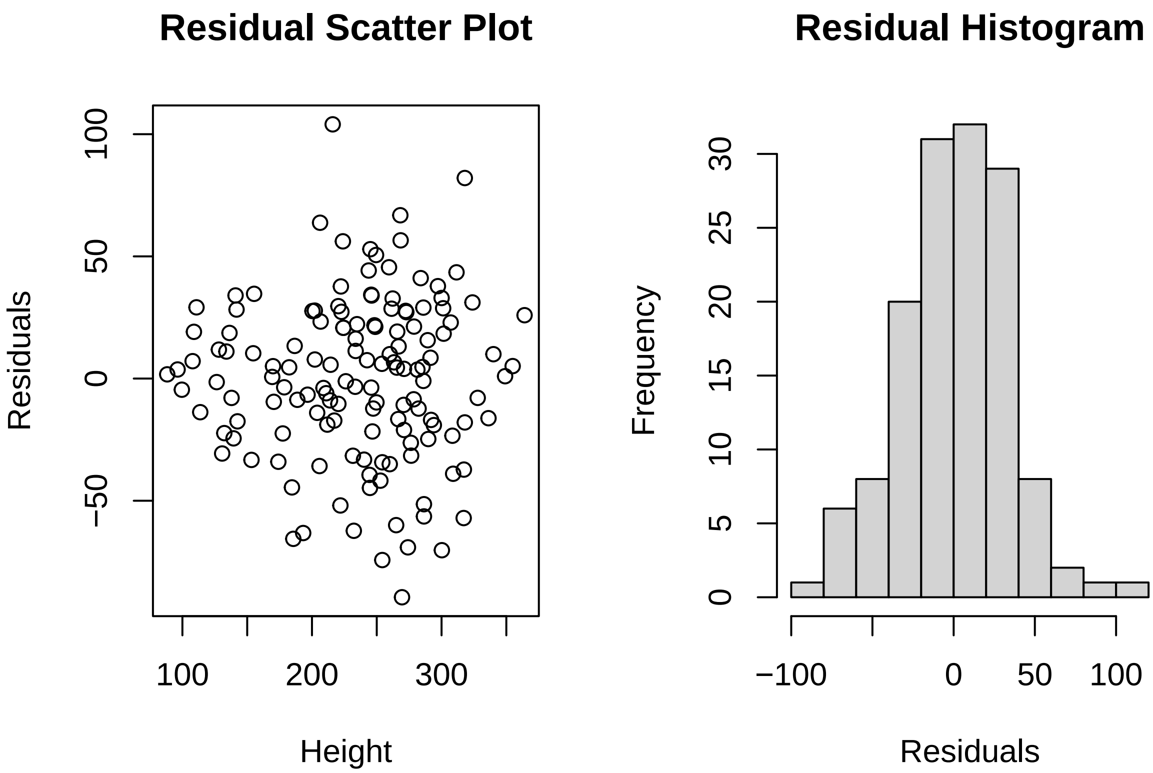
We can now plot the scatter plot of Height vs log(Dbh) together with the line of best fit.

```
plot(log(Dbh), Height,cex=.5,main="Tree Heights vs Diameters ",xlab="log(Diameter)",ylab="Heights")
abline(model1)
```



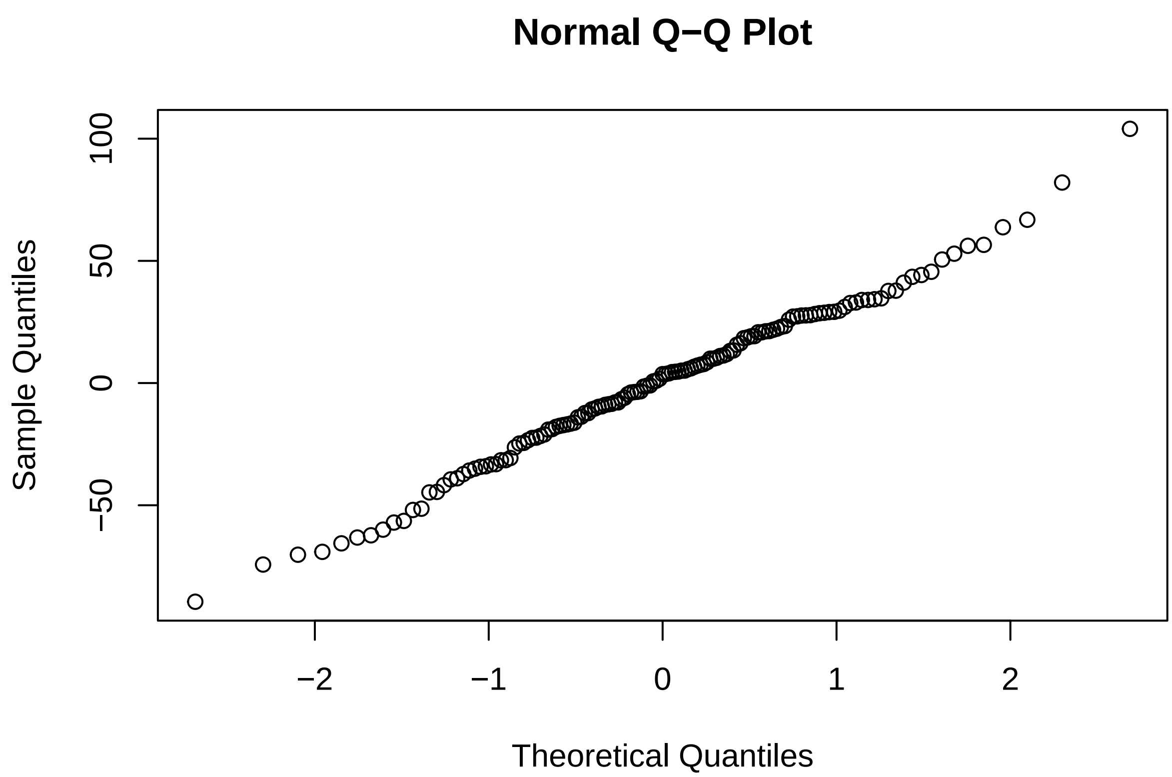
For model diagnostics, we will compute the residuals using the residuals() function, call them residualsmodel1, and plot both the scatterplot of residuals and the histogram of the residuals. We will plot the two plots together using the par() function. The function par(mfrow(1,2)) divides the plotting region into 1x2 grid of panels.

```
residualsmodel1<-residuals(model1)
par(mfrow=c(1,2))
plot(predict(model1), residualsmodel1, main="Residual Scatter Plot", xlab="Height", ylab="Residuals")
hist(residualsmodel1, main="Residual Histogram", xlab="Residuals")
```



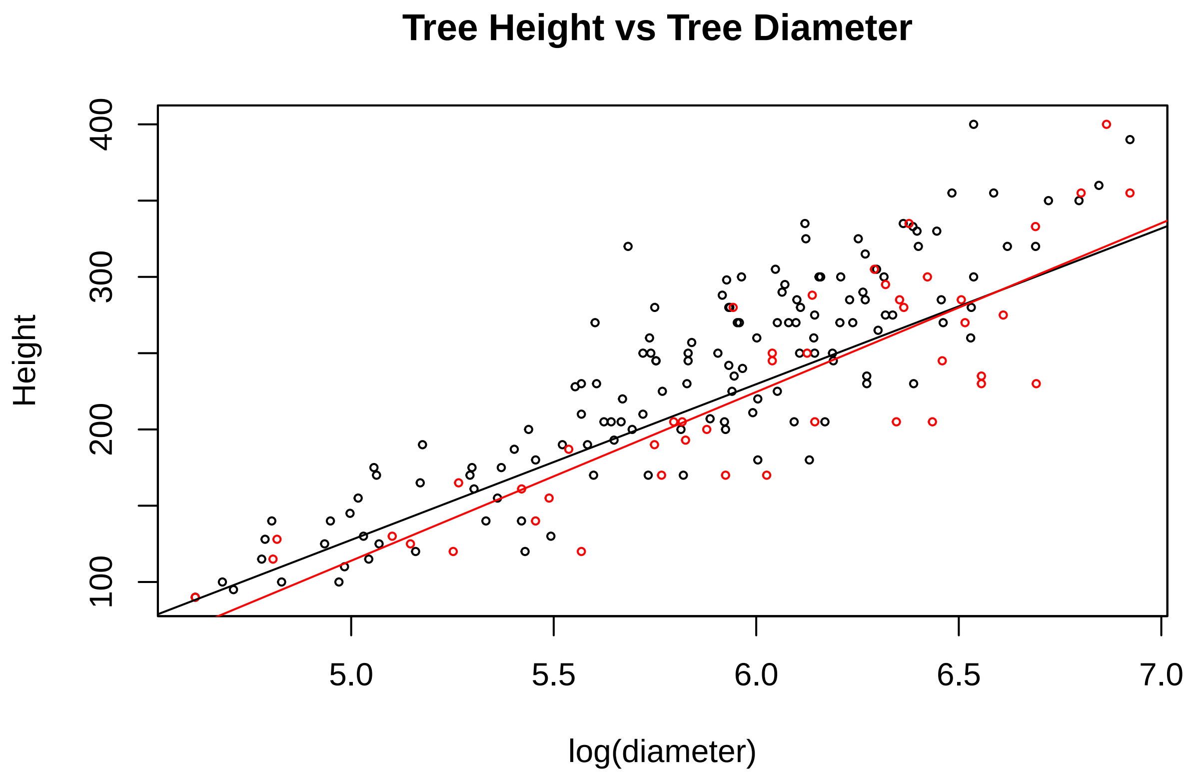
Note the very high and very low residuals, and a possible fan out(megaphone) shape in the above residual plot. You can also check if the residuals are normally distributed by plotting a normal probability plot of the residuals.

```
qqnorm(residualsmodel1)
```



We can also fit a model to a subset of the dataset, say to trees in Plots numbered larger than 80. Using the subset() function as below creates a new dataframe selecting the variables in these Plots. We fit a linear model to this subset of trees with Height as the response and log(diameter) as the explanatory (predictor) variable, call this model model2 and plot the line of best fit (in red) of model2 together with the original scatter plot of Height vs log(diameter) for the full dataset and the line of best fit (in black) of the model1 for the full dataset. We will also plot the two scatter plots in the same plot using the par() function.

```
selplot<-subset(ufcwc,Plot>80)
selplot.Height<-selplot$Height
selplot.Dbh<-selplot$Dbh
model2=lm(selplot.Height~log(selplot.Dbh))
plot(log(Dbh), Height,cex=.5,main="Tree Height vs Tree Diameter ", xlab="log(diameter)",ylab="Height")
par(new=TRUE)
plot(selplot.Height~log(selplot.Dbh), cex=.5, col="red", ylab="",yaxt="n", xlab="", xaxt="n")
abline(model1, col="black")
abline(model2, col="red")
```



Section 2: Confidence and Prediction Intervals

The R function **predict()** can compute CIs and PIs both.

95% Confidence Interval for mean height of trees with diameter=500.

```
predict(model1, data.frame(Dbh=500), interval="confidence", level=.95)
```

```
##        fit      lwr      upr
## 1 279.4506 272.5296 286.3716
```

95% Prediction Interval for the height a tree with diameter=500.

```
predict(model1, data.frame(Dbh=500), interval="prediction", level=.95)
```

```
##        fit      lwr      upr
## 1 279.4506 213.1717 345.7295
```

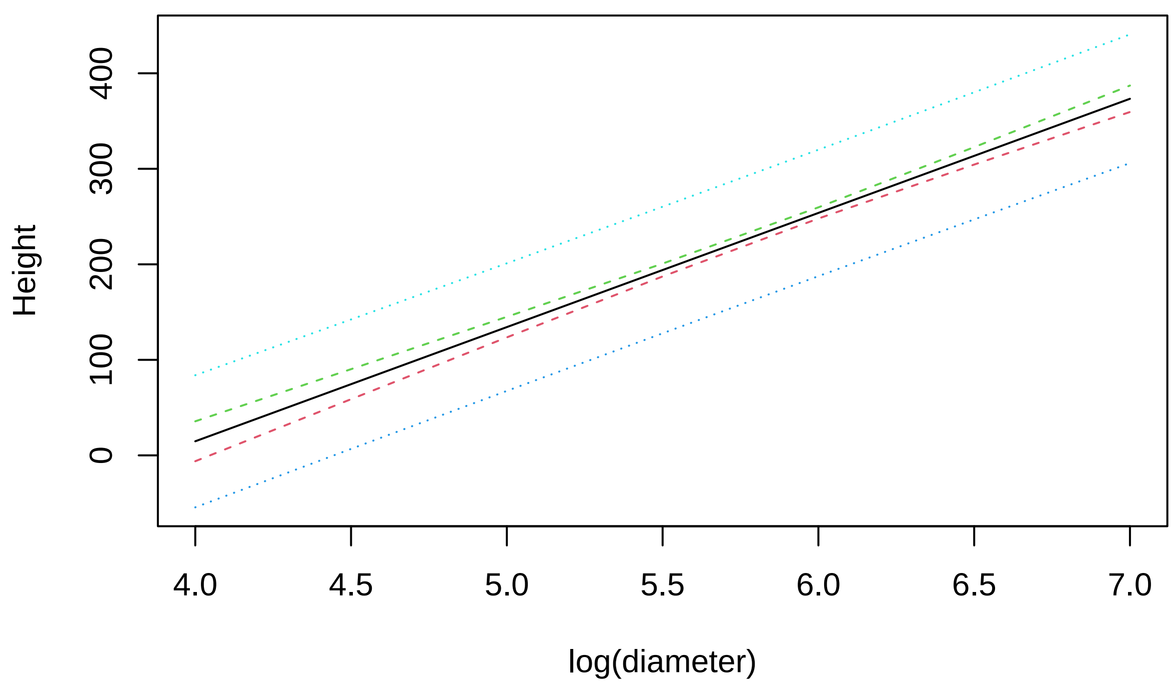
Next, let us plot 95% CI and PI bands:

```
x=log(Dbh)
new <- data.frame(x = seq(4, 7, 0.5))
predict(lm(Height~x), new, se.fit = TRUE)
```

```
## $fit
##         1         2         3         4         5         6         7
##  14.76240  74.52201 134.28161 194.04122 253.80082 313.56043 373.32003
##
## $se.fit
```

```
##         1         2         3         4         5         6         7
## 10.572661  7.942062  5.447147  3.400935  2.959931  4.610412  7.003606
##
## $df
## [1] 137
##
## $residual.scale
## [1] 33.33444
```

```
pred.w.plim <- predict(lm(Height~x), new, interval = "prediction", level=.95)
pred.w.clim <- predict(lm(Height~x), new, interval = "confidence", level=.95)
library(graphics)
matplot(new$x, cbind(pred.w.clim, pred.w.plim[,-1]),lty = c(1,2,2,3,3), type = "l", ylab = "Height",
        xlab = "log(diameter)")
```



Alternatively, you can use:

```
plot(Height ~ log(Dbh), type="n", ylim=range(pred.w.plim))

lines(new$x, pred.w.clim[,1], lty=3)

matlines(new$x, pred.w.clim[,2:3], lty=2, col="red")

matlines(new$x, pred.w.plim[,2:3], lty=1, col="blue")

legend("topleft", inset=.05, lty=c(3,2,1),
       col=c("black","red","blue"),
```

```
legend=c("OLS line", "95% CIs", "95% PIs") )
```