Project Phoenix learnR Session 4 UNAM

Regression

Dr Mark Kelson - University of Exeter

13 June 2017

Figure 1:

Welcome to regression

Linear regression is a workhorse of applied statistics and is a very flexible approach that can be used in a wide variety of applications.

This is taken from here http://www.artifex.org/~meiercl/R_statistics_guide.pdf

Correlation and simple linear regression

Correlation quantifies how well two variables X and Y covary together. Correlation only makes sense when both X and Y are variables that you measure. If you control X and are measuring Y, then use linear regression. Correlation does not discriminate between X and Y, and linear regression does – only use regression if you can clearly define which variable is X (i.e. is independent) and which is Y (dependent). In addition to importing data from .csv files, it is possible to create vectors of data for use with R. To demonstrate how to carry out a Pearson's correlation analysis and a linear regression, first make vectors of elevation, precipitation, temperature, and soil pH data:

```
#Input of elevation data in meters
m <- c(1500,2020,2720,3400)
#Input of precip data (mm) at each elevation
ppt <- c(938.3,797.6,593.2,475.6)
#Input of mean temperature data (degrees celsius) at each elevation
temp <- c(17.9,14.9,10.8,7.1)
#Input of soil C:N ratio at each elevation
cn <- c(21.04,21.56,19.33,19.27)
#Input of O-horizon soil pH at each elevation
pH <- c(4.25,4.1,4.2,4.12)
```

Calculation of Pearson's correlation coefficients for selected pairs of variables

For the "cor" function, Pearson's correlation coefficient is the default method. It is possible to add a method ="spearman" or "kendall" argument if these coefficients are desired.

```
cor(m,ppt)
```

```
## [1] -0.9941432
```

```
cor(m,temp)
```

```
## [1] -0.9998673
```

```
cor(temp,ppt)
```

```
## [1] 0.995771
```

```
cor(m,cn)
```

```
## [1] -0.8520494
```

```
cor(m,pH)
```

```
## [1] -0.4933305
```

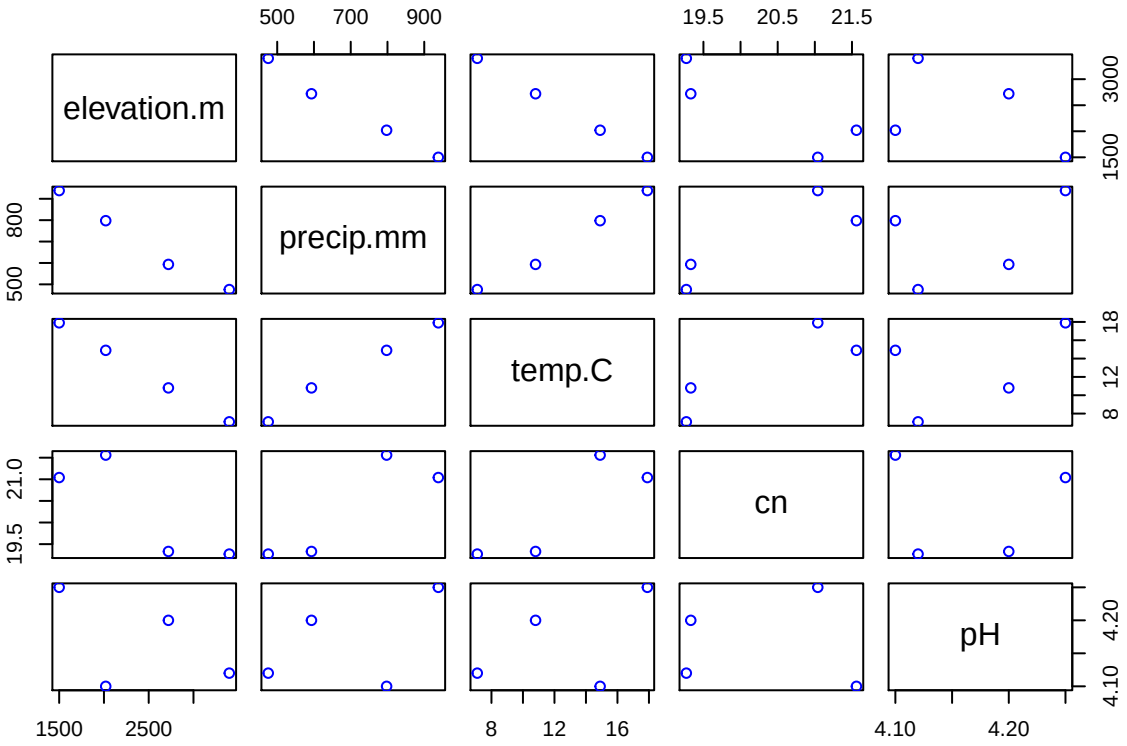
```
cor(temp,pH)
```

```
## [1] 0.489425
```

Plot all combinations of variables to examine the shapes of the relationships

First we can make a dataframe from our previously input vectors

```
abiotic <- data.frame(elevation.m= m, precip.mm=ppt,temp.C = temp, cn = cn, pH= pH)
#Plot all possible combinations of these variables with blue symbols
plot(abiotic, col= "blue")
```



Using linear regression to calculate a line of best fit between two variables.

In order to be theoretically valid, assume that there is a "dependent/indepent" type of relationship between soil pH and temperature, with temperature as the independent variable.

```
m1 <- lm(pH~temp.C, data= abiotic)
summary(m1)
```

```
##
## Call:
## lm(formula = pH ~ temp.C, data = abiotic)
##
## Residuals:
##         1         2         3         4
##  0.044610 -0.083635  0.046097 -0.007072
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.075586   0.121677  33.495  0.00089 ***
## temp.C      0.007252   0.009136   0.794  0.51057
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0747 on 2 degrees of freedom
## Multiple R-squared:  0.2395, Adjusted R-squared:  -0.1407
## F-statistic:  0.63 on 1 and 2 DF,  p-value: 0.5106
```

Note that the R-squared value calculated by the linear regression procedure gives the same "r" – by taking the square root – as does the Pearson's correlation analysis.

Checking the homogeneity of variance assumption

Plotting residuals vs. fitted values is a good way to check whether the variance in the data are constant. The vertical scatter surrounding a horizontal line running through y=0 should be random: We will use the mtcars dataset

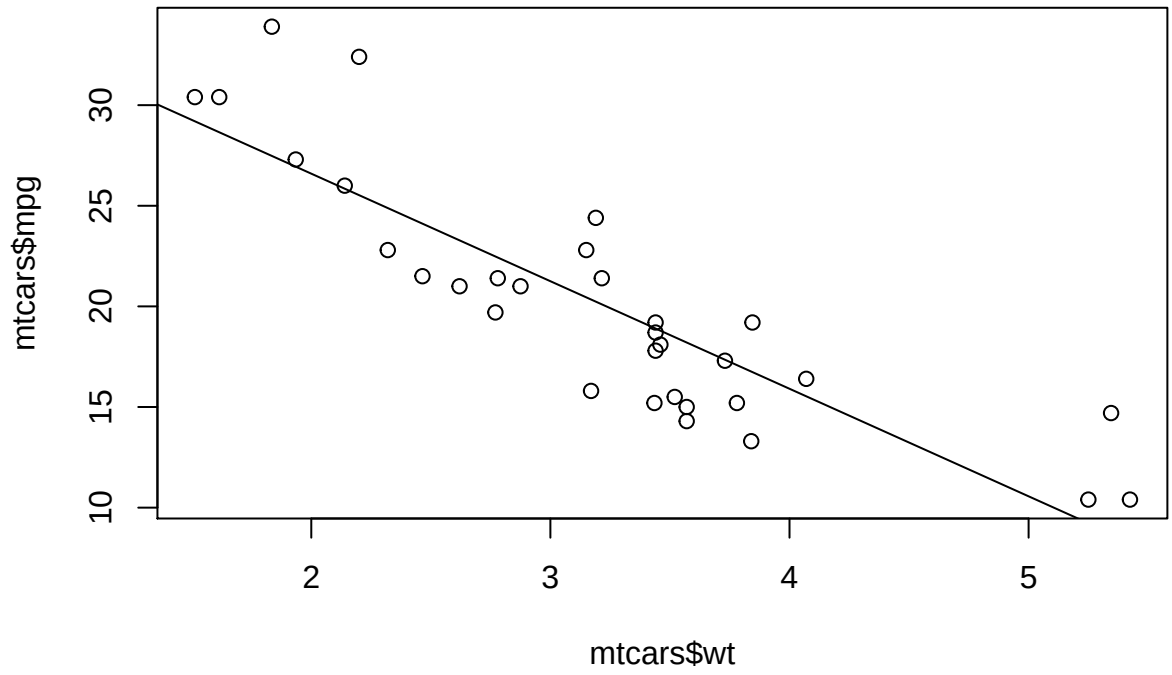
```
mtcar
```

Let's explore whether a cars weight (mtcarswt)$is associated with it's miles per gallon (mtcars$mpg)

```
model1 <- lm(mpg~wt,data=mtcars)
summary(model1)
```

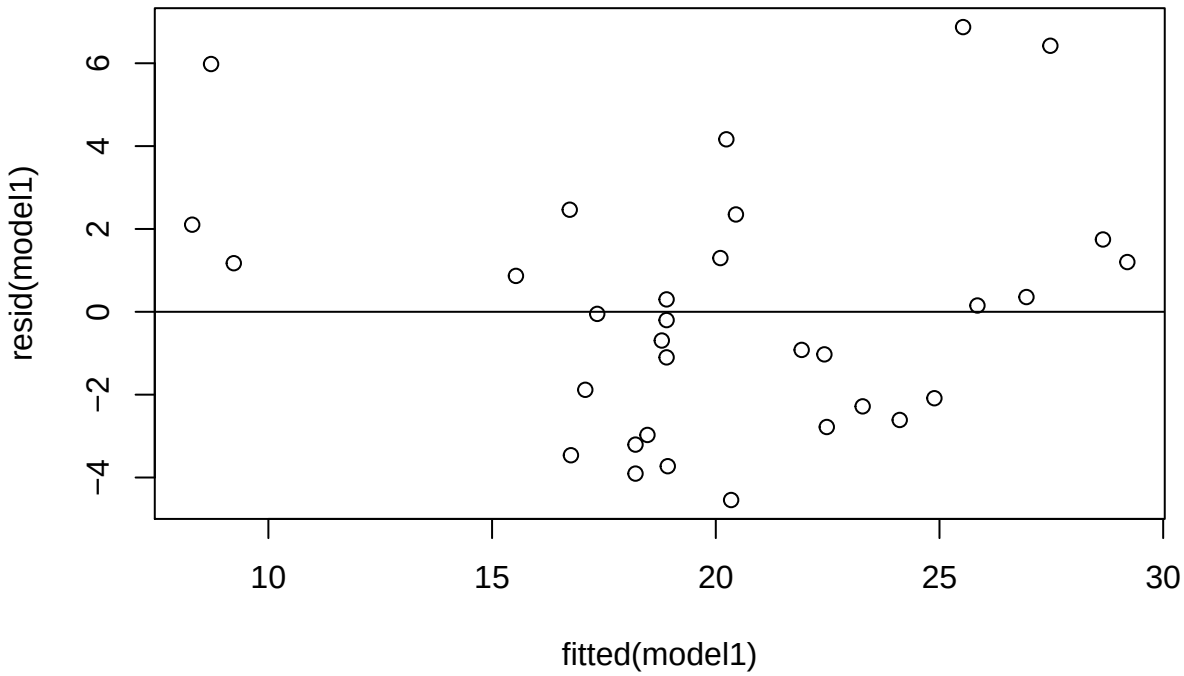
```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528, Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

```
plot(mtcars$wt,mtcars$mpg)
abline(model1)
```



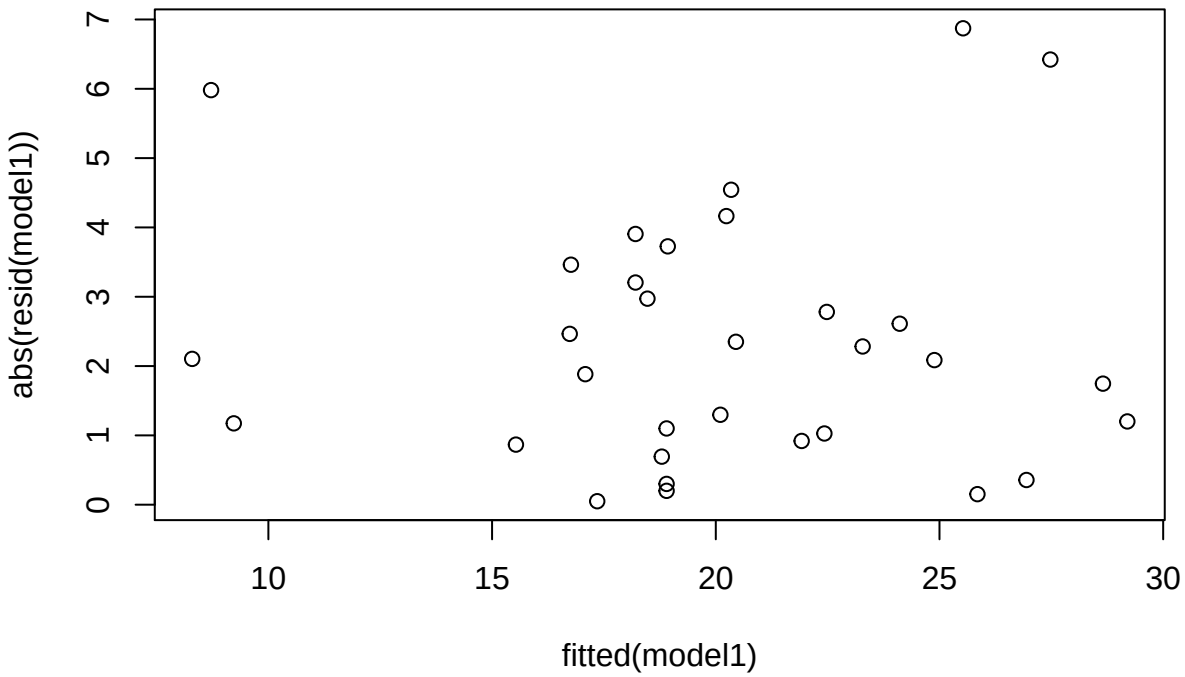
Let's explore the model fit

```
plot(fitted(model1),resid(model1))
abline(h=0)
```



The last two commands above will return a plot of the residual values vs. the fitted values for the model, with a horizontal line through y=0. We can also plot the absolute value of the residuals vs. the fitted values in order to increase the resolution of one's ability to detect whether there's a trend that would suggest non-constant variance. This is done with the following commands

```r
plot(fitted(model1),abs(resid(model1)))
```



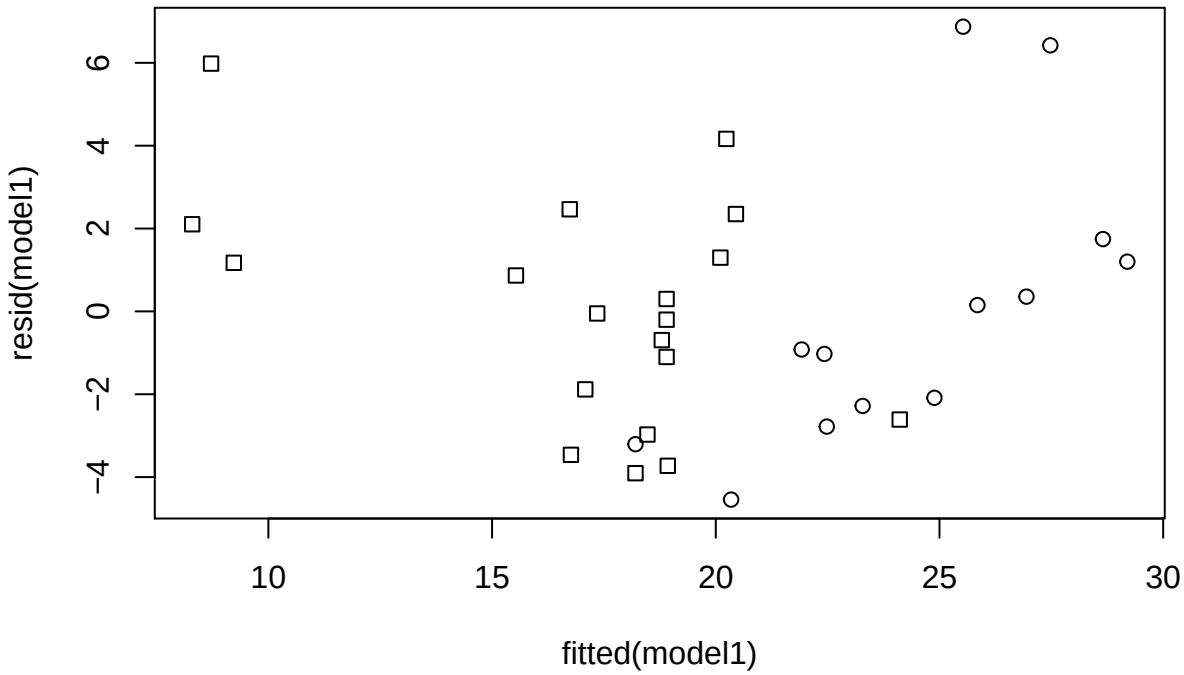
A quick way to check non-constant variance quantitatively is this simple regression

```r
summary(lm(abs(resid(model1))~ fitted(model1)))
```

```
##
## Call:
## lm(formula = abs(resid(model1)) ~ fitted(model1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3076 -1.3262 -0.2680  0.9149  4.5662
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     2.466743   1.319687   1.869   0.0714 .
## fitted(model1) -0.006277   0.063632  -0.099   0.9221
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.853 on 30 degrees of freedom
## Multiple R-squared:  0.0003242,  Adjusted R-squared:  -0.033
## F-statistic: 0.00973 on 1 and 30 DF,  p-value: 0.9221
```

For a categorical variable, like mtcars$am (automatic or manual), we can graphically check how the levels are behaving (i.e. – is there any difference among the levels in terms of variance) like this:

```r
plot(resid(model1)~fitted(model1),pch=unclass(mtcars$am))
```



The argument "pch" specifies how the "plot characters" should be drawn. Need to look for non-constant variance across all groups, and whether there's any ordering among groups.

It is also possible to use Levene's test to assess the homogeneity of variance assumption. Levene's test is quite insensitive to non-normality. Also, because most parametric statistical tests are relatively insensitive to non-constant variance, there is no need to take action unless Levene's Test is significant at the 1% level (Faraway 2004).

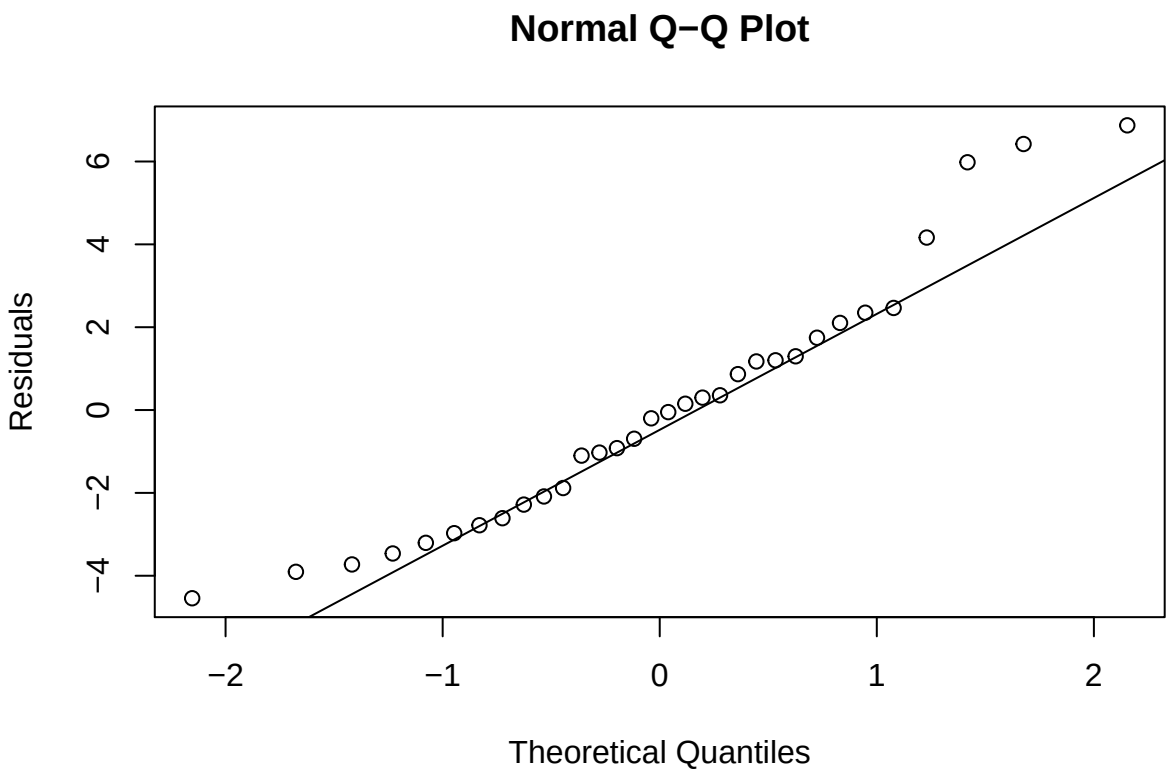
```
library(car)
leveneTest(mtcars$mpg,mtcars$am)
```

```
## Warning in leveneTest.default(mtcars$mpg, mtcars$am): mtcars$am coerced to
## factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value  Pr(>F)
## group  1  4.1876 0.04957 *
##       30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

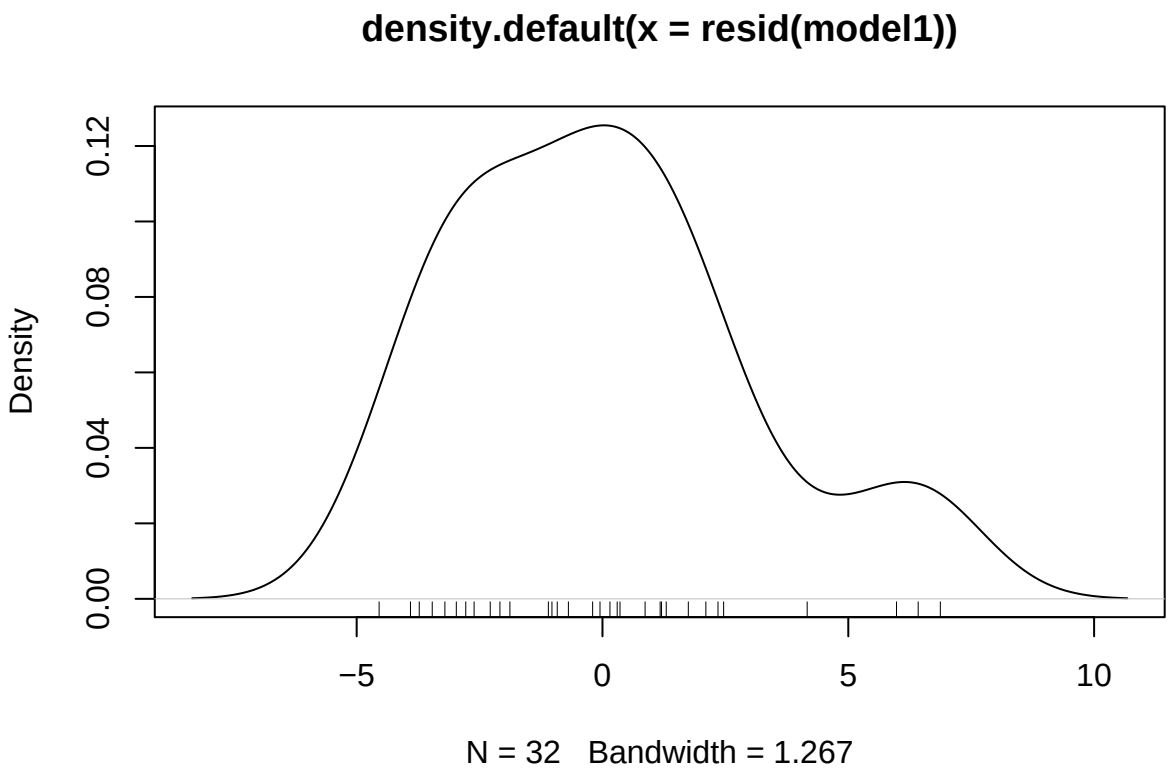
Checking the Normality assumption (residuals should be normally distributed for linear models). Note: Only long-tailed distributions cause large inaccuracies for linear models. It is possible to transform log-normal distributions, but mild non-normality can be safely ignored. Make a quantile-quantile (Q-Q) plot that compares the model's residuals to "ideal" normal observations:

```
qqnorm(resid(model1),ylab="Residuals")
qqline(resid(model1))
```



The latter command puts a straight line onto the graph that joins the 1st and 3rd quartiles (and is therefore not influenced by outliers). To graphically display how the residuals are distributed, one can plot the density distribution of the residuals:

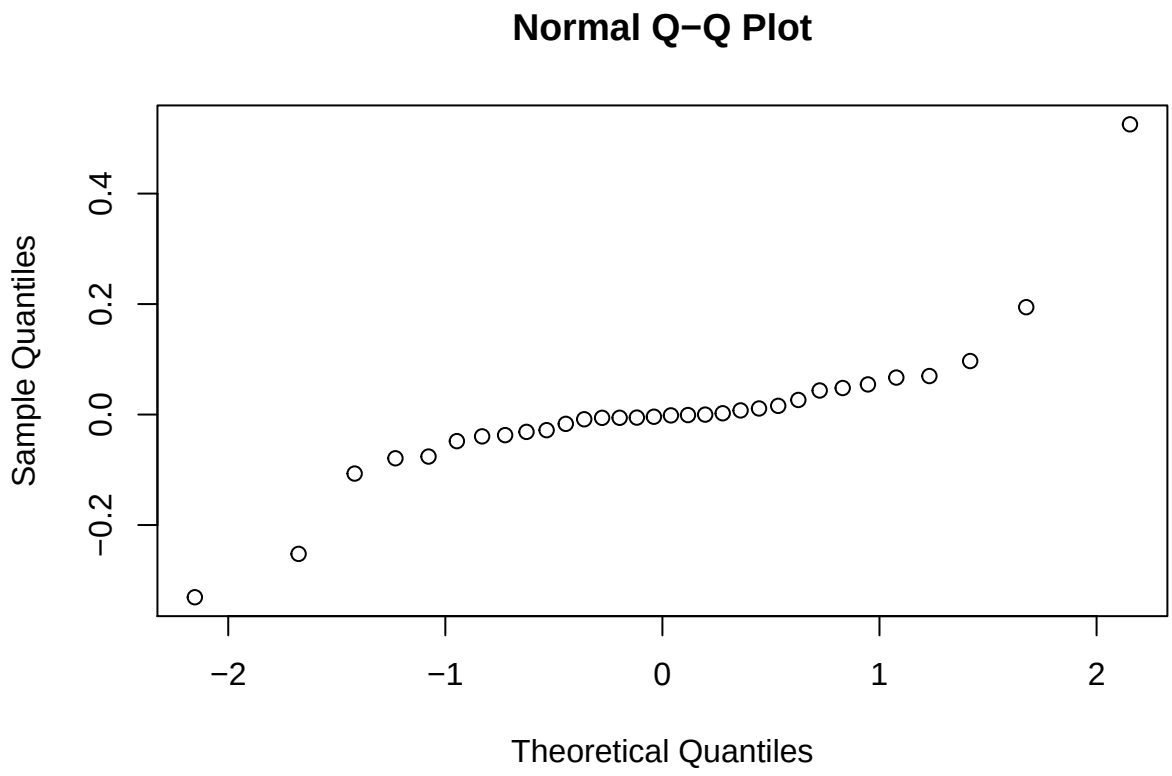
```
plot(density(resid(model1)))
rug(resid(model1))
```



The "rug" command puts the individual points as a stripchart beneath the density plot.

Checking for outliers and points with undue leverage in a linear model Use of leverage plot:

```
gi <- influence(model1)
qqnorm(gi$coef[,2])
```



This produces a leverage plot for the 2nd term in the model, since inclusion of the intercept makes the weight variable the 2nd term in the model.