Linear Regression in R

In this document we describe how to perform a simple linear regression in R. We show how to get coefficients of the regression line, test for significance of the slope, find R^2 statistic, make transformations to variables and much more. We also show how to make plot of the data with regression as well as plotting the residuals.

Introduction

We start with the 'Babies Crawling' data set and we want to investigate the relationship between the temperature and the average crawling age in weeks. First we download the data set into R (since we have names for the variables in the data set we need to put 'header=TRUE' in the 'read.table' function):

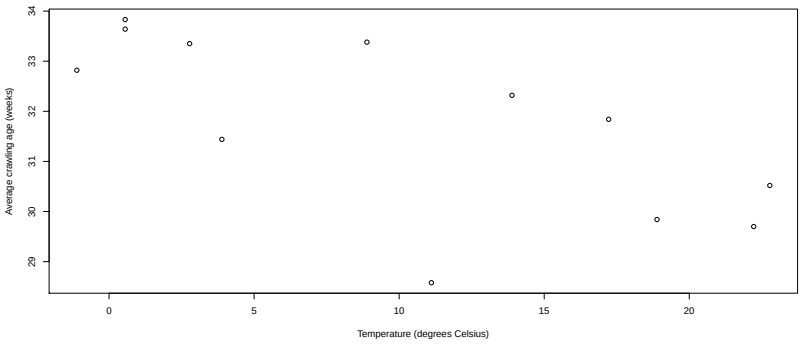
```
Crawling.data = read.table('BabiesCrawlingData.txt',header=TRUE)
attach(Crawling.data)
```

As usual we use 'attach' function in order to use the variables in the data set individually with the names correspond to the header titles. Since the 'temperature' variable here is in Fahrenheit but we want it in Celsius, we transform the variable by subtracting 32 and multiplying by $\frac{5}{9}$:

```
temperature=(temperature-32)/9*5
```

Next let's make a scatter plot of the 'Average crawling age' versus 'Temperature', note that the first variable in the 'plot' function is horizontal, second one is vertical (response), xlab and ylab specify the x-axis and y-axis labels:

```
plot(temperature,avg_crawling_age,xlab='Temperature (degrees Celsius)',
  ylab='Average crawling age (weeks)')
```

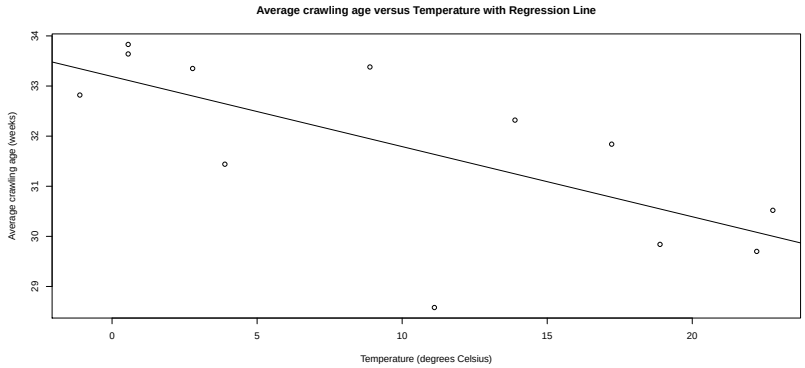


It seems that linear regression model can be appropriate in this case and we want to plot these data with the regression line. To do that we first make a new object which contains all the information about the regression line and we call it 'lin.reg.crawl':

```
lin.reg.crawl=lm(avg_crawling_age~temperature)
```

Note that the first variable in the 'lm' function is the response variable, the second one is our predictor or independent variable and in the middle is a 'tilde' symbol. Now we are ready to plot the scatterplot, then add the regression line with 'abline' function and add the title with the 'title' function:

```
plot(temperature,avg_crawling_age,xlab='Temperature (degrees Celsius)',
  ylab='Average crawling age (weeks)')
abline(lin.reg.crawl)
title('Average crawling age versus Temperature with Regression Line')
```



To find the regression coefficients we can first find mean of the temperature, mean of the crawling age, standard deviation of the temperature, standard deviation of the crawling age and correlation between temperature and crawling age:

```
mean.temp=mean(temperature)
sd.temp=sd(temperature)
mean.crawl=mean(avg_crawling_age)
sd.crawl=sd(avg_crawling_age)
cor.temp.crawl=cor(avg_crawling_age,temperature)
```

Here the 'cor' function finds the correlation between two variables. Now we find slope (b1) and intersection (b0) from these statistics:

```
b1=sd.crawl/sd.temp*cor.temp.crawl
b0=mean.crawl-b1*mean.temp
b0;b1
```

```
[1] 33.19041
[1] -0.1399305
```

Hence we see that the slope is about -0.14 while the intersection is 33.19. Note that the object 'lin.reg.crawl' which we have constructed before contains all the information about the regression line, therefore instead of finding the coefficients using different statistics as we did above, we can just type:

```
lin.reg.crawl
```

```
Call:
lm(formula = avg_crawling_age ~ temperature)

Coefficients:
(Intercept)  temperature
    33.1904      -0.1399
```

We get exactly the same coefficients. Finally if we want to predict the average crawling age when temperature is 15 degrees we can get it using a simple equation:

```
b0+b1*15
```

```
[1] 31.09145
```

Some caution

We start this section with the 'CFC' data set. First we download the data set ('head' function shows the first six observations):

```
CFC.data = read.table('CFCdata.txt',header=TRUE)
head(CFC.data)
attach(CFC.data)
```

```
  year month    time cfc11
1 1977     1 1977.00 139.9
2 1977     2 1977.08 139.5
3 1977     3 1977.17 139.0
4 1977     4 1977.25 134.1
5 1977     5 1977.33 135.0
6 1977     6 1977.42 143.4
```

The goal is to investigate the relationship between 'time' and 'cfc11'. These data contain information till 2005 but we first want to analyse relationship before 1990, which are observations from 1 to 156. So we construct new variables with first 156 observations:

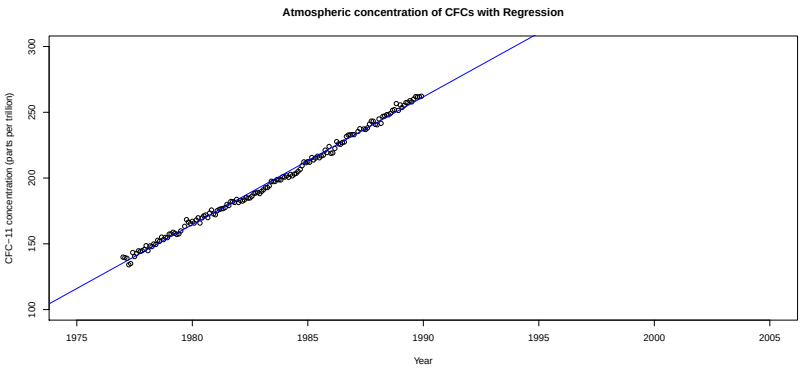
```
time.till.1990=time[1:156]
cfc11.till.1990=cfc11[1:156]
```

Now we create linear regression model for these two variables ('cfc11.till.1990' is the response while 'time.till.1990' is the predictor)

```
lin.reg.cfc=lm(cfc11.till.1990~time.till.1990)
```

Next we make a plot of the 'cfc11' versus 'time' until 1990 with regression line:

```
plot(time.till.1990,cfc11.till.1990,xlab='Year',ylim=range(100:300),
  xlim=range(1975:2005),ylab='CFC-11 concentration (parts per trillion)',
  main='Atmospheric concentration of CFCs with Regression')
abline(lin.reg.cfc,col='blue')
```



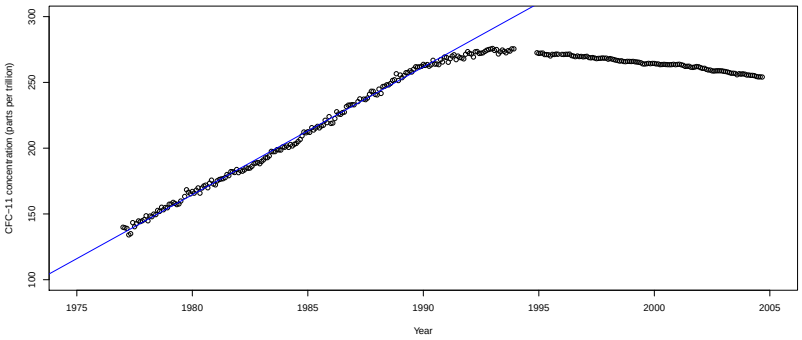
It seems that linear regression fits quite well. We also find coefficients of the regression line:

```
lin.reg.cfc
```

```
Call:
lm(formula = cfc11.till.1990 ~ time.till.1990)

Coefficients:
   (Intercept)  time.till.1990
    -19064.028           9.711
```

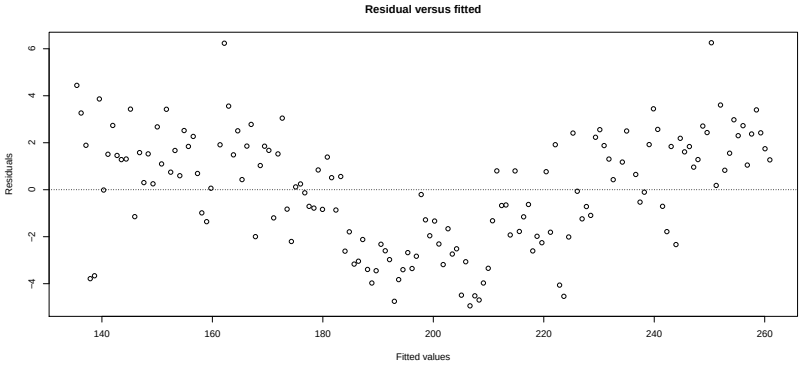
Now lets plot all the data points with the regression line:

```
plot(time,cfc11,xlab='Year',ylim=range(100:300),xlim=range(1975:2005),
  ylab='CFC-11 concentration (parts per trillion)')
abline(lin.reg.cfc,col='blue')
```



We clearly see that the linear model is not appropriate here. A very important part in checking whether a linear regression is appropriate or not is to plot residuals versus fitted values. Lets make this plot (note that 'lin.reg.cfc$fitted' and 'lin.reg.cfc$residuals' extract the fitted value and residuals form the regression object)

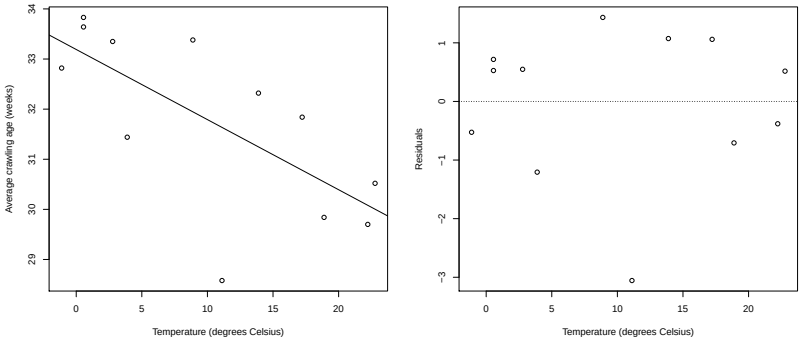
```
plot(lin.reg.cfc$fitted,lin.reg.cfc$residuals,xlab='Fitted values',
  ylab='Residuals',main='Residual versus fitted' )
abline(h=0,lty=3)
```



Here 'abline(h=0,lty=3)' produces a horizontal, dashed line (h=0 indicates a horizontal line with 0 y-axis intercept, while lty prodices dashed line). This plot also shows that there is a problem with simple regression since we observe some pattern in the residual plot.

Next we move back to the 'Average crawling age' data. In the last section we have already created the 'lin.reg.crawl' regression model. Now we make a scatter plot of the response versus predictor with the regression line and the plot of residuals versus the temperature in the same plot. To make two charts in the same plot, just type 'par(mfrow=c(1,2))' before the plots, c(1,2) means that we have two plot in one row:

```
par(mfrow=c(1,2))
plot(temperature,avg_crawling_age,xlab='Temperature (degrees Celsius)',
  ylab='Average crawling age (weeks)')
abline(lin.reg.crawl)

plot(temperature,lin.reg.crawl$residuals,xlab='Temperature (degrees Celsius)',
  ylab='Residuals')
abline(h=0,lty=3)
```



As always we can get coefficients of the regression line:

```
lin.reg.crawl
```

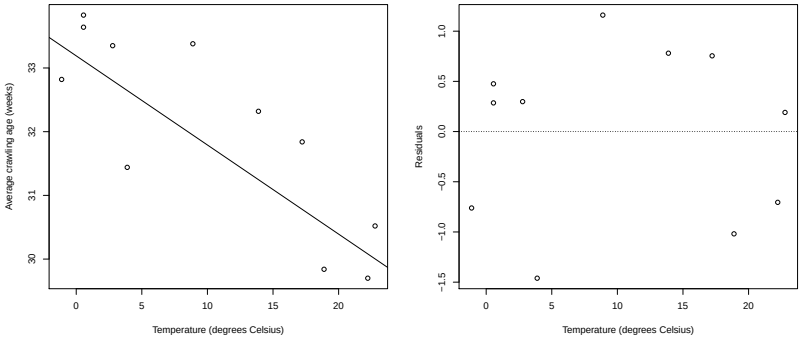
```
Call:
lm(formula = avg_crawling_age ~ temperature)

Coefficients:
(Intercept)  temperature
    33.1904      -0.1399
```

Based on the above plot we observe one observation which has lowest residual and might be an influential point. Hence we want to make the analysis again but without this observation. To do that we construct two new variables without this point (note that this unusual observation is on the fifth place in the data set)

```
avg_crawling_age.out.rem=avg_crawling_age[-5]
temperature.out.rem=temperature[-5]
```

Square bracket with minus ([-x]) just removes the xth point from the variable. Next we create a linear regression object for these variables without the potential outlier and make a scatter plot and residual plot:

```
lin.reg.crawl.out.rem=lm(avg_crawling_age.out.rem~temperature.out.rem)
par(mfrow=c(1,2))
plot(temperature.out.rem,avg_crawling_age.out.rem,xlab='Temperature
  (degrees Celsius)',ylab='Average crawling age (weeks)')
abline(lin.reg.crawl)

plot(temperature.out.rem,lin.reg.crawl.out.rem$residuals,
  xlab='Temperature (degrees Celsius)',ylab='Residuals')
abline(h=0,lty=3)
```



Then we get regression coefficients for the data without the point in a usual way:

```
lin.reg.crawl.out.rem
```

```
Call:
lm(formula = avg_crawling_age.out.rem ~ temperature.out.rem)

Coefficients:
        (Intercept)  temperature.out.rem
            33.4299              -0.1361
```

Since the coefficients have not changed by much we cannot say that the removed observation is influential.

The coefficient of determination

An important question in the regression analysis is to find how well a regression line fits the data. One measure of the fit is the coefficient of determination or R^2. Consider first the 'Average crawling age' data. To find R^2 we need to find sum of squares total, sum of squares regression or sum of squares residuals. Lets find all of them using R software. First we find mean of the response:

```
mean.crawl=mean(avg_crawling_age)
```

Now we can find total SS:

```
SStot=sum( (avg_crawling_age-mean.crawl)^2 )
SStot
```

```
[1] 34.09617
```

Note when you just type the name of the variable then R returns the value of this variable. Since we have already created the 'lin.reg.crawl' object of linear regression we can use it to find fitted values. So we easily get SS regression and SS residual:

```
SSreg=sum( (lin.reg.crawl$fitted-mean.crawl)^2 )
SSreg
```

```
[1] 16.69332
```

```
SSres=sum( (avg_crawling_age-lin.reg.crawl$fitted)^2 )
SSres
```

```
[1] 17.40285
```

Note that if we add SS regression with SS residual, we get exactly SS total. Now we are ready to find R^2 statistic:

```
R.square=SSreg/SStot
R.square
```

```
[1] 0.4895952
```

Also in a simple linear regression, R^2 should be the same as correlation squared. Lets check that it is true

```
cor(temperature,avg_crawling_age)^2
```

```
[1] 0.4895952
```

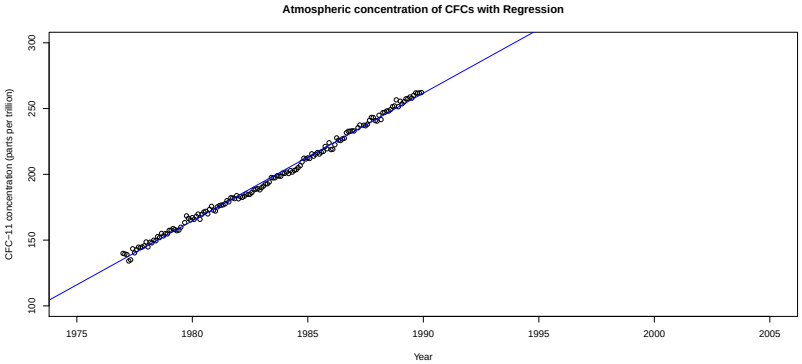
As you can see we get exactly the same answers. There is a third way (most convenient) to find R^2. As we have mentioned before 'lin.reg.crawl' object contains all the information about the regression and certainly it also contains the coefficient of determination. We can get it using the next command:

```
summary(lin.reg.crawl)$r.squared
```

```
[1] 0.4895952
```

Once again we get exactly the same answer. To finish this section let us return to the 'CFC11' data set. We focus on the data before 1990.

```
plot(time.till.1990,cfc11.till.1990,xlab='Year',ylim=range(100:300),
  xlim=range(1975:2005),ylab='CFC-11 concentration (parts per trillion)',
  main='Atmospheric concentration of CFCs with Regression')
abline(lin.reg.cfc,col='blue')
```



Even though we know that linear regression is not appropriate for these data, lets get R^2 anyway.

```
summary(lin.reg.cfc)$r.squared
```

```
[1] 0.995721
```

Hence we see that 99.6% of variation is explained by this regression line.

Inference for the slope

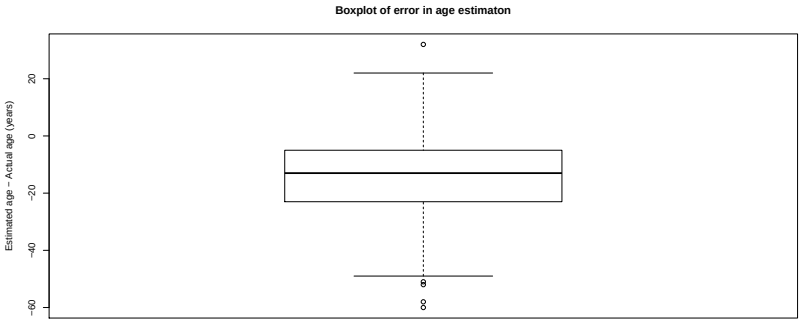
In this section we show how to test that a particular estimate (of slope or intersection) is statistically significant or not. We start with 'Skeleton' data. Doing the standard procedures we get:

```
Skeleton.data = read.table('SkeletonDataComplete.txt',header=TRUE)
head(Skeleton.data)
attach(Skeleton.data)
```

```
  Sex      BMIcat BMIquant Age DGestimate DGerror SBestimate SBerror
1   2 underweight    15.66  78         44     -34         60     -18
2   1      normal    23.03  44         32     -12         35      -9
3   1  overweight    27.92  72         32     -40         61     -11
4   1  overweight    27.83  59         44     -15         61       2
5   1      normal    21.41  60         32     -28         46     -14
6   1 underweight    13.65  34         25      -9         35       1
```

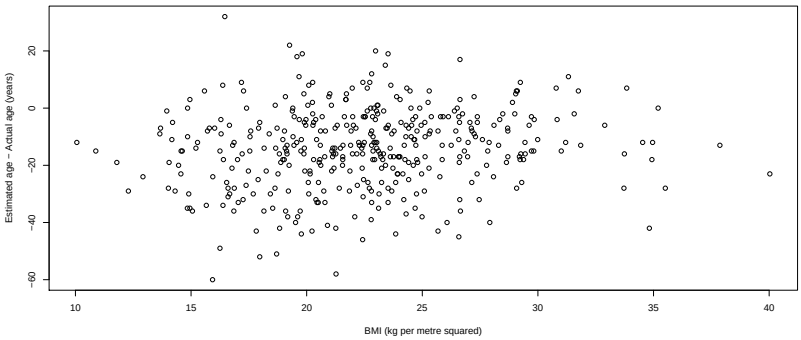
Our response in this analysis would be 'DGerror' variable, lets first make a boxplot of this variable.

```
boxplot(DGerror,main='Boxplot of error in age estimation',ylab='Estimated age
   - Actual age (years)' )
```



Since we are interested in the relationship between 'BMIquant' and 'DGerror', we make a scatter plot of these two variables:

```
plot(BMIquant,DGerror,xlab='BMI (kg per metre squared)',ylab='Estimated age
  - Actual age (years)')
```

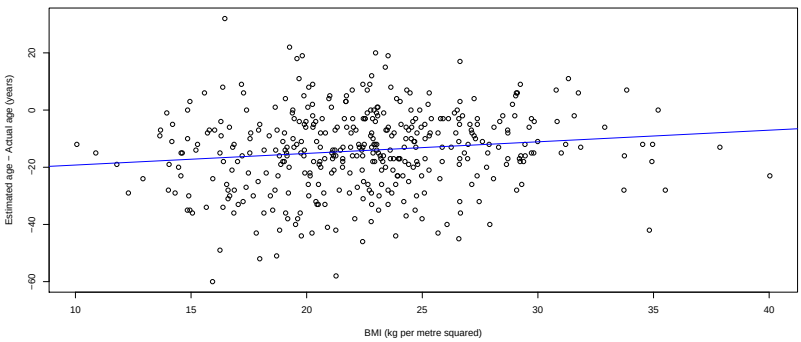


Next we want to fit a regression line to this plot. So we first construct a regression object in a usual way:

```
lin.reg.error=lm(DGerror~BMIquant)

Call:
lm(formula = DGerror ~ BMIquant)

Coefficients:
(Intercept)     BMIquant
   -23.2875       0.4061
```

Hence the slope is 0.4061 and intercept is -23.2875. We make a scatterplot with the regression line:

```
plot(BMIquant,DGerror,xlab='BMI (kg per metre squared)',ylab='Estimated age
  - Actual age (years)')
abline(lin.reg.error,col='blue')
```



As explained in the last section R^2 statistic is an important measure of the fit in linear regression; we can easily get:

```
summary(lin.reg.error)$r.squared
```

```
[1] 0.01863018
```

Hence less than 2% of variation of the response is explained by this regression. But we have also a very important question: is the slope statistically significant? Because if it is not, then 'BMIquant' is not important for the prediction of 'DGerror'. We can easily answer this question (and not only for the slope but also for the intercept) using the 'summary' function:

```
summary(lin.reg.error)
```

```
Call:
lm(formula = DGerror ~ BMIquant)

Residuals:
    Min      1Q  Median      3Q     Max
-43.351  -7.918   0.545   9.171  48.602

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -23.2875     3.3973  -6.855 2.73e-11 ***
BMIquant      0.4061     0.1478   2.749  0.00625 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 14.01 on 398 degrees of freedom
Multiple R-squared:  0.01863,   Adjusted R-squared:  0.01616
F-statistic: 7.556 on 1 and 398 DF,  p-value: 0.006255
```

This output has much information but the p-values are under $Pr(> |t|)$. So we see that the p-value for the slope is 0.00625 which is quite small and therefore we conclude that 'BMIquant' variable is important for prediction and we should not ignore it. Also note that in the above output we have 'Multiple R-squared' which is the coefficient of determination.

Now we return to the 'Crawling' data set. We want to check whether temperature really effects the average crawling age or not.

```
summary(lin.reg.crawl)
```

```
Call:
lm(formula = avg_crawling_age ~ temperature)

Residuals:
    Min      1Q  Median      3Q     Max
-3.0556 -0.5712  0.5221  0.8029  1.4334

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 33.19041    0.59570  55.716 8.41e-14 ***
temperature -0.13993    0.04518  -3.097   0.0113 *
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 1.319 on 10 degrees of freedom
Multiple R-squared:  0.4896,    Adjusted R-squared:  0.4386
F-statistic: 9.592 on 1 and 10 DF,  p-value: 0.01131
```

Based on the output we conclude that the p-value for the slope is 0.0113 which can be considered as small and therefore temperature is statistically significant.

Checking for conditions

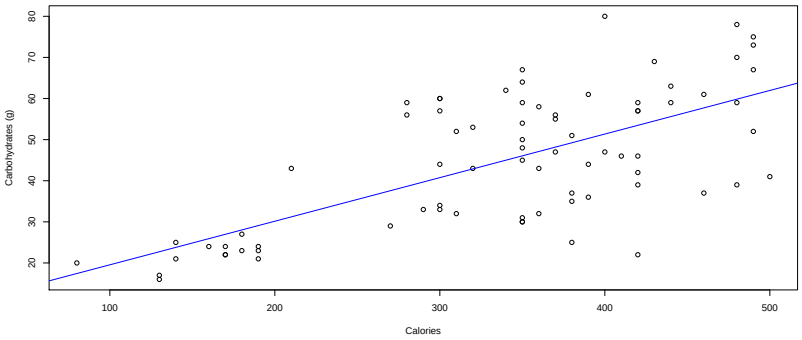
In this section we focus on the new data set 'Coffee Shop'. As usual we input the data, and observe first several observations:

```
Coffee.data = read.table('Coffee_Shop.txt',header=TRUE)
head(Coffee.data)
attach(Coffee.data)
```

```
  calories carb   type
1      350   67 bakery
2      350   64 bakery
3      420   59 bakery
4      490   75 bakery
5      130   17 bakery
6      370   47 bakery
```

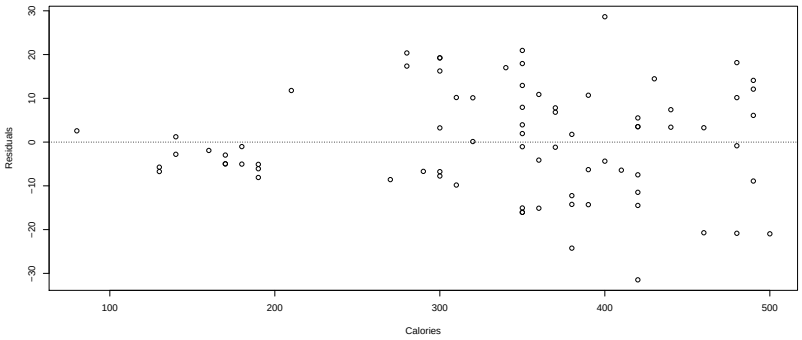
In this example we want to fit a linear regression with 'carb'(carbohydrates) as the response and 'calories' variable as the predictor.

```
lin.reg.coffee=lm(carb~calories)
plot(calories,carb,xlab='Calories',ylab='Carbohydrates (g)')
abline(lin.reg.coffee,col='blue')
```



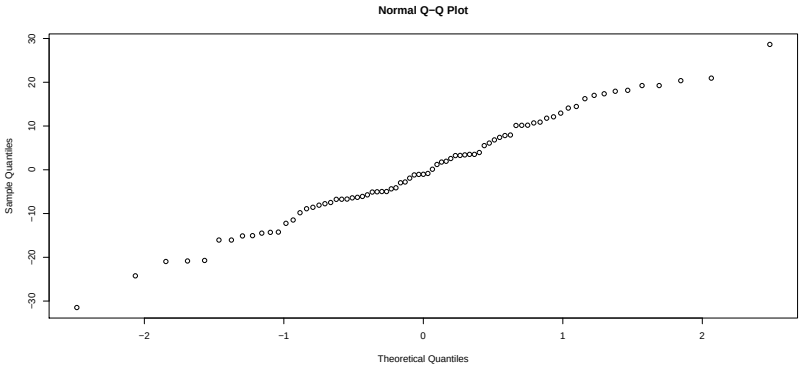
Regression analysis is not complete without the residual plot, so we make residuals versus 'calories' scatter-plot with horizontal dashed line:

```
plot(calories,lin.reg.coffee$residuals,xlab='Calories',ylab='Residuals')
abline(h=0,lty=3)
```



We immediately see a problem. The variance is not constant and increases as 'calories' increase. Hence it is not appropriate to carry out inference on the slope of the regression line in this case. Lets also make a quantile-quantile plot of the residuals:

```
qqnorm(lin.reg.coffee$residuals)
```



The points on this plot should lie on a straight line (and that would indicate that residuals have normal distribution). In this example the plot is generally straight with some small departure from linearly in the right tail.

Transformations

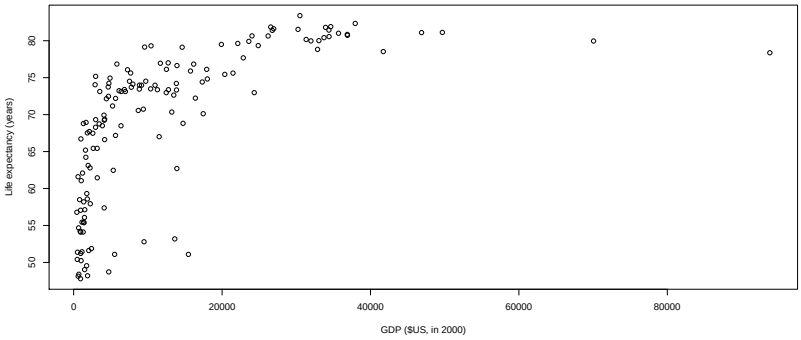
As shown in the last section, linear regression is not appropriate in some cases. In this section we show how to transform predictor and/or response to make linear regression valid. Consider the 'Life Expectancy' data set.

```
LifeExp.data = read.table('LifeExpComplete.txt',header=TRUE)
head(LifeExp.data)
attach(LifeExp.data)
```

```
      Country Region LifeExp       GDP HIV
1 Afghanistan    SAs  48.673        NA  NA
2     Albania   EuCA  76.918        NA  NA
3     Algeria   MENA  73.131  6406.817 0.1
4      Angola    SSA  51.093  5519.183 2.0
5   Argentina   Amer  75.901 15741.046 0.5
6     Armenia   EuCA  74.241  4748.929 0.1
```

Next we plot 'LifeExp' versus 'GDP':

```
plot(GDP,LifeExp,xlab='GDP ($US, in 2000)',ylab='Life expectancy (years)')
```

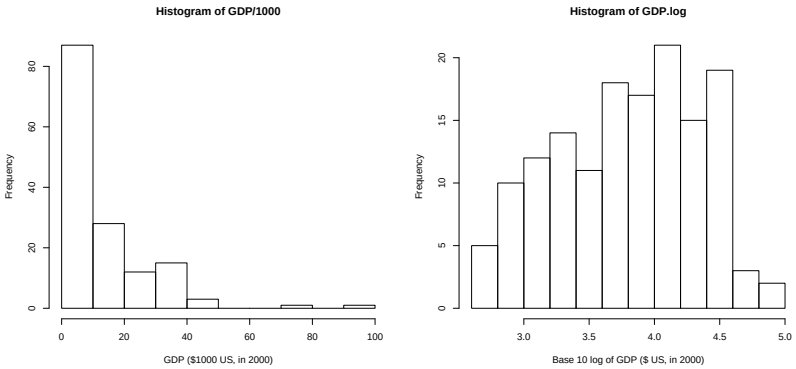


Clearly the relationship is not linear. However we see that 'GDP' variable has many small values and several observations are very large. Hence base 10 log transformation may help in this situation. First we construct a new variable 'GDP.log' that will store base 10 logs of the original 'GDP' variable

```
GDP.log=log10(GDP)
```

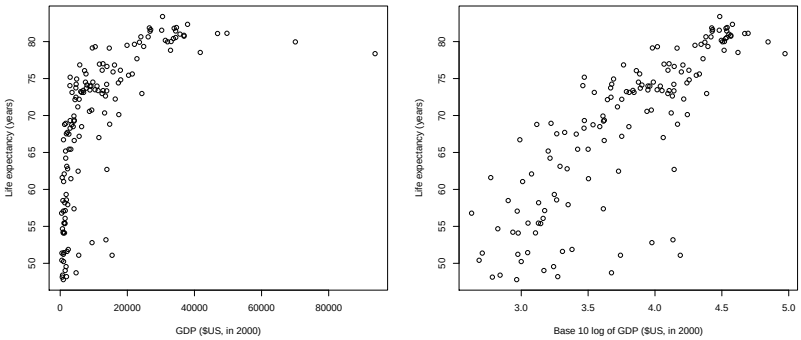
Now we make two histograms in one plot. The first one is histogram of the original variable, second one of the 'GDP.log':

```
par(mfrow=c(1,2))
> hist(GDP/1000,xlab='GDP ($1000 US, in 2000)')
> hist(GDP.log,xlab='Base 10 log of GDP ($ US, in 2000)')
```



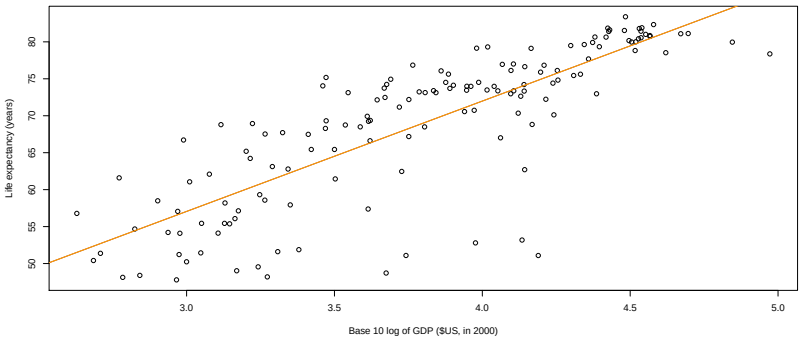
See how the distribution changed from right skewed to almost symmetric. Also lets compare scatterplot of 'LifeExp' versus original 'GDP' and versus transformed variable:

```
par(mfrow=c(1,2))
plot(GDP,LifeExp,xlab='GDP ($US, in 2000)',ylab='Life expectancy (years)')
plot(GDP.log,LifeExp,xlab='Base 10 log of GDP ($US, in 2000)',
  ylab='Life expectancy (years)')
```



Now the second plot seems much more linear than the original one. Hence we can make a linear regression of the 'Life expectancy' versus the transformed GDP:

```
lin.reg.lifeexp.log=lm(LifeExp~GDP.log)
par(mfrow=c(1,1))
plot(GDP.log,LifeExp,xlab='Base 10 log of GDP ($US, in 2000)',
  ylab='Life expectancy (years)')
abline(lin.reg.lifeexp.log,col='orange')
```

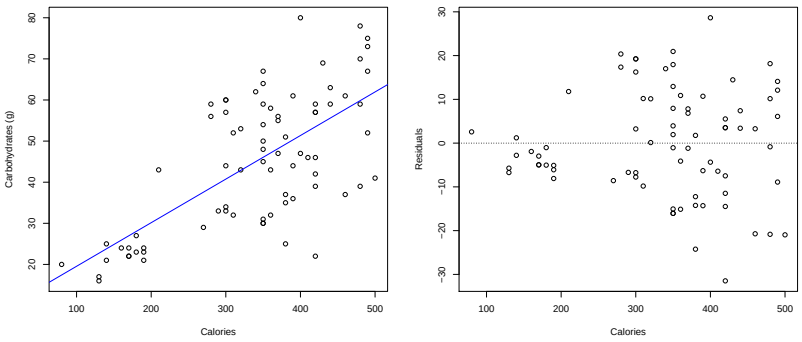


To get regression line coefficients and to test their significance, we use 'summary' function as before:

```
summary(lin.reg.lifeexp.log)


Call:
lm(formula = LifeExp ~ GDP.log)

Residuals:
    Min      1Q  Median      3Q     Max
-23.723  -2.395   0.779   3.641  11.085

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  12.2618     3.4306   3.574 0.000477 ***
GDP.log      14.9310     0.8948  16.687  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 5.989 on 145 degrees of freedom
  (50 observations deleted due to missingness)
Multiple R-squared:  0.6576,    Adjusted R-squared:  0.6552
F-statistic: 278.5 on 1 and 145 DF,  p-value: < 2.2e-16
```

To finish this section we return to the 'Coffee Shop' data set. We have already created 'lin.reg.coffee' object, which is regression of 'carbohydrates' on 'calories'. So lets plot again the scatterplot with the regression line and residuals versus predictor:

```
par(mfrow=c(1,2))
plot(calories,carb,xlab='Calories',ylab='Carbohydrates (g)')
abline(lin.reg.coffee,col='blue')
plot(calories,lin.reg.coffee$residuals,xlab='Calories',ylab='Residuals')
abline(h=0,lty=3)
```

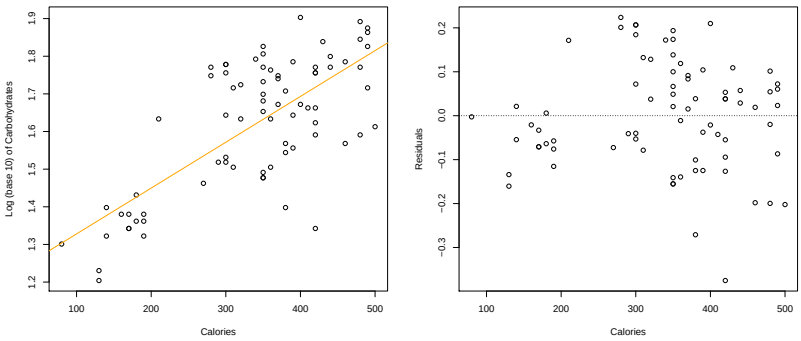


Since the variance is not constant we try to make base 10 logarithm transformation of the response. We construct a new transformed variable and make regression of the modified variable versus 'calories':

```
carb.log=log10(carb)
lin.reg.coffee.log=lm(carb.log~calories)
```

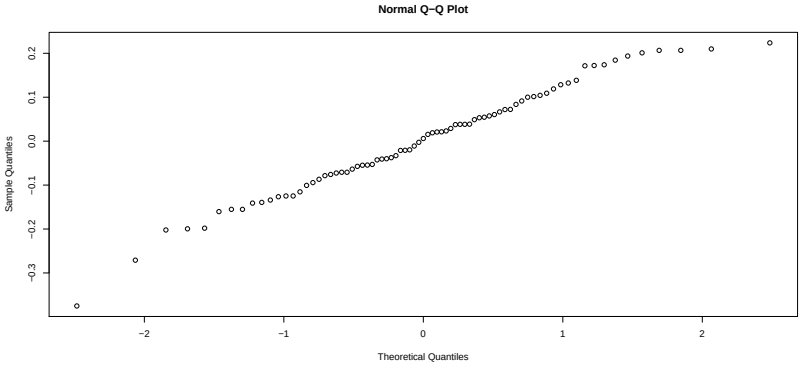
Then we plot the regression line and residuals:

```
par(mfrow=c(1,2))
plot(calories,carb.log,xlab='Calories',ylab=
  'Log (base 10) of Carbohydrates')
abline(lin.reg.coffee.log,col='orange')
plot(calories,lin.reg.coffee.log$residuals,xlab=
  'Calories',ylab='Residuals')
abline(h=0,lty=3)
```



Now the variance is constant and using linear regression is appropriate (there are two large negative residuals which we will discuss later). Lets check the condition that residuals follow a normal distribution using quantile-quantile plot

```
qqnorm(lin.reg.coffee.log$residuals)
```



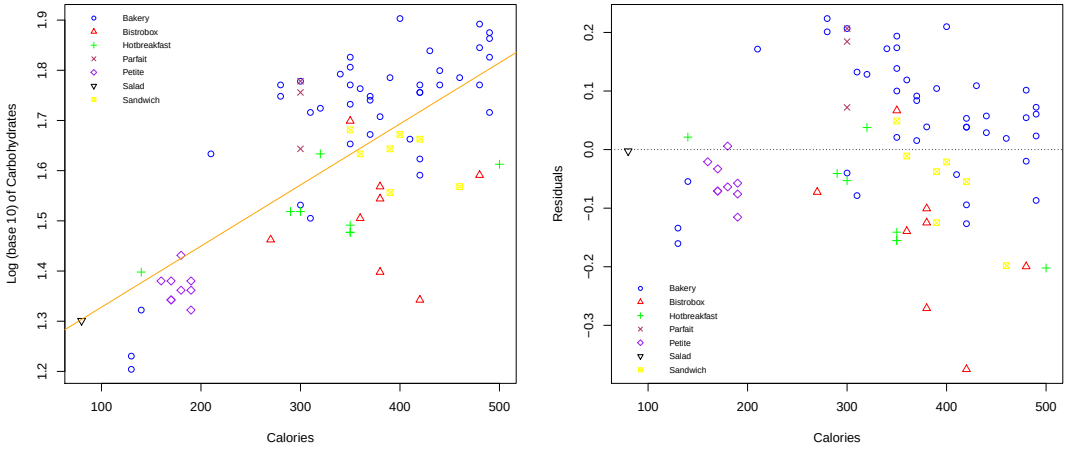
The plot looks straight and therefore we can conclude that normal assumption is satisfied.

To explain two unusual observations from the residual plot, let's make a scatterplot of 'carb.log' versus 'calories' but with different symbols and colors corresponding to the 'type' of the food. Since 'type' variable contains names of different foods, it is convenient to make them as factors:

```
type=as.factor(type)
```

Next we make a scatterplot of 'carb.log' versus 'calories' and residual plot

```
par(mfrow=c(1,2))
plot(calories,carb.log,xlab='Calories',ylab='Log (base 10) of Carbohydrates',
  pch=c(1,2,3,4,5,6,7)[as.numeric(type)], col=c("blue","red","green",
  "maroon","purple","black","yellow")[as.numeric(type)])
abline(lin.reg.coffee.log,col='orange')
legend(x="topleft",c("Bakery", "Bistrobox", "Hotbreakfast",
  "Parfait", "Petite", "Salad", "Sandwich"),pch=c(1,2,3,4,5,6,7),
  col=c("blue","red","green","maroon","purple","black","yellow"),
  cex=0.7,bty="n")

plot(calories,lin.reg.coffee.log$residuals,xlab='Calories',ylab='Residuals',
  pch=c(1,2,3,4,5,6,7)[as.numeric(type)],col=c("blue","red","green",
  "maroon","purple","black","yellow")[as.numeric(type)])
abline(h=0,lty=3)
legend(x="bottomleft",c("Bakery", "Bistrobox", "Hotbreakfast",
  "Parfait", "Petite", "Salad", "Sandwich"),pch=c(1,2,3,4,5,6,7),
  col=c("blue","red","green","maroon","purple","black","yellow"),
  cex=0.7,bty="n")
```



Here 'pch' and 'col' specify symbols and colors respectively. Since we have 7 different types, 'as.numeric(type)' can be $1, 2, 3, 4, 5, 6, 7$ with 1 corresponding to "Bakery", 2 to "Bistrobox" and so on by alphabetical order. So for example if an observation has a "Sandwich" type which corresponds to 7 then this point would have 'pch=7' symbol with 'yellow' color. In the 'legend' function, 'x' and 'cex' specify position and size of a legend respectively, 'bty="n"' means that we do not need border for it.
We see that these two unusual observations correspond to 'bistrobox' items. Actually almost all the 'bistrobox' food is below the fitted line, therefore it is important to use 'type' variable in the analysis to explain relationship between 'carbohydrates' and 'calories'.

Summary of R Functions

We give a short summary of all new R functions [and arguments] that we have learned in this module:

Linear Regression

```
lm(Response~Predictor)
summary()
summary(Object)$fitted
summary(Object)$r.squared
summary(Object)$residuals
```

Plots

```
abline() [h,v,lty,col]
legend() [x,legend,pch,col,cex,bty]
par() [mfrow]
qqnorm()
title()
```

Miscellaneous

```
as.factor()
as.numeric()
log10()
```