# MATH 60604A Statistical Modelling

## Chapter 2 Part 1: Simple Linear Regression

---

## 1) Introduction: Exploring the data

We'll consider the following example throughout chapter 2 to illustrate the use of linear regression models. The data here (although not real) are inspired by studies from the Tech3Lab.

The study: subjects navigated a website that contained, among other things, an ad for candy. During the site navigation, an *eye-tracker* mesured the location on the screen on which the subject's eyes were fixated. They were also able to measure whether the subject saw the ad and for how long it was in sight. Additionally, facial expression analysis software (FaceReader) allowed the researchers to guess the subject's emotions when the ad was in sight. At the end of the study, a questionnaire measured the subject's intention to buy this type of candy, as well as other variables (including socio-demographic variables).

The study objectives are two-fold:

1) Evaluate whether there is a link between the duration of fixation on the ad and the intention to buy the candy.
2) Evaluate whether the perceived emotion is linked to the intention to buy the candy.

Only subjects that had actually seen the ad in question are included in the analysis, giving a total sample size of $n = 120$. The data are saved in the `intention.csv` file. The data include the following variables:

- `intent`: the intention to buy measured through two questions based on a Likert scale (1=strongly disagree,...,7=strongly agree). The variable intention is the sum of the two and takes the values 2 to 14. The higher the value, the more the subject expressed interest in buying the product.
- `fix`: the total duration of fixation on the ad (in seconds).
- `emo`: a measure of emotion / reaction during fixation. It is the ratio of the probability of showing a positive emotion to the probability of showing a negative emotion.
- `sex` : the subject's sex (dichotomization considered with 0=male, 1=female).
- `age` : the subject's age (years).
- `rev` : the subject's annual income (categorized as 1=0-$20 000 ; 2=$20 000 - $60 000 ; 3=$60 000+).
- `educ` : the subject's level of education (categorized as 1=less than high school ; 2=high school ; 3=university)
- `stat` : marital status (categorized as 0=single ; 1=in a relationship).

We're ultimately interested in measuring the effect of **fixation** and **emotion** on the variable **intention**, while **adjusting for socio-demographic variables**. Here,

- **Dependent variable** ($Y$): intention
- **Explanatory variables** ($X$): fixation, emotion, sex, age, revenue, education, marital status

In the case of simple linear regression, we'll only estimate the relationship between `intent` and a single explanatory variable. In the case of multiple linear regression (the next part of this chapter), we'll analyze the relationship between `intent` and **all** explanatory variables **simultaneously**.

We'll start off by exploring the data...

```r
intention<-read.csv("Data/intention.csv")
head(intention)
```

```
##     fix   emo sex age rev educ stat intent
## 1 0.081 1.417   1  27   1    2    0     11
## 2 2.235 1.146   0  27   1    1    0     12
## 3 1.675 0.296   1  26   1    2    1      6
## 4 0.630 0.731   1  34   3    3    0      4
## 5 2.197 0.841   1  30   1    2    1     11
## 6 0.424 0.334   0  29   3    3    1      4
```

Some descriptive statistics:

```r
# using summary function:
summary(intention)
```

```
##       fix             emo              sex              age
##  Min.   :0.028   Min.   :0.0530   Min.   :0.0000   Min.   :19.00
##  1st Qu.:0.836   1st Qu.:0.7175   1st Qu.:0.0000   1st Qu.:27.00
##  Median :1.307   Median :0.9260   Median :1.0000   Median :30.00
##  Mean   :1.578   Mean   :1.0380   Mean   :0.5167   Mean   :30.06
##  3rd Qu.:2.066   3rd Qu.:1.3790   3rd Qu.:1.0000   3rd Qu.:33.25
##  Max.   :5.835   Max.   :2.7970   Max.   :1.0000   Max.   :45.00
##       rev             educ            stat             intent
##  Min.   :1.000   Min.   :1.000   Min.   :0.0000   Min.   : 2.000
##  1st Qu.:1.000   1st Qu.:1.750   1st Qu.:0.0000   1st Qu.: 6.000
##  Median :2.000   Median :2.000   Median :1.0000   Median : 8.000
##  Mean   :2.067   Mean   :2.042   Mean   :0.5417   Mean   : 8.258
##  3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:11.000
##  Max.   :3.000   Max.   :3.000   Max.   :1.0000   Max.   :14.000
```

```r
# alternatively:
summary<-sapply(intention,function(x) c(mean(x),sd(x),min(x),max(x),length(x)))
row.names(summary)<-c("mean","sd","min","max","n")
summary
```

```
##             fix        emo       sex        age       rev       educ
## mean   1.577808   1.038000  0.5166667  30.058333  2.0666667  2.0416667
## sd     1.093448   0.533982  0.5018174   5.018111  0.8068336  0.7378806
## min    0.028000   0.053000  0.0000000  19.000000  1.0000000  1.0000000
## max    5.835000   2.797000  1.0000000  45.000000  3.0000000  3.0000000
## n    120.000000 120.000000 120.0000000 120.000000 120.0000000 120.0000000
##             stat     intent
## mean   0.5416667   8.258333
## sd     0.5003500   2.934855
## min    0.0000000   2.000000
## max    1.0000000  14.000000
## n    120.0000000 120.000000
```

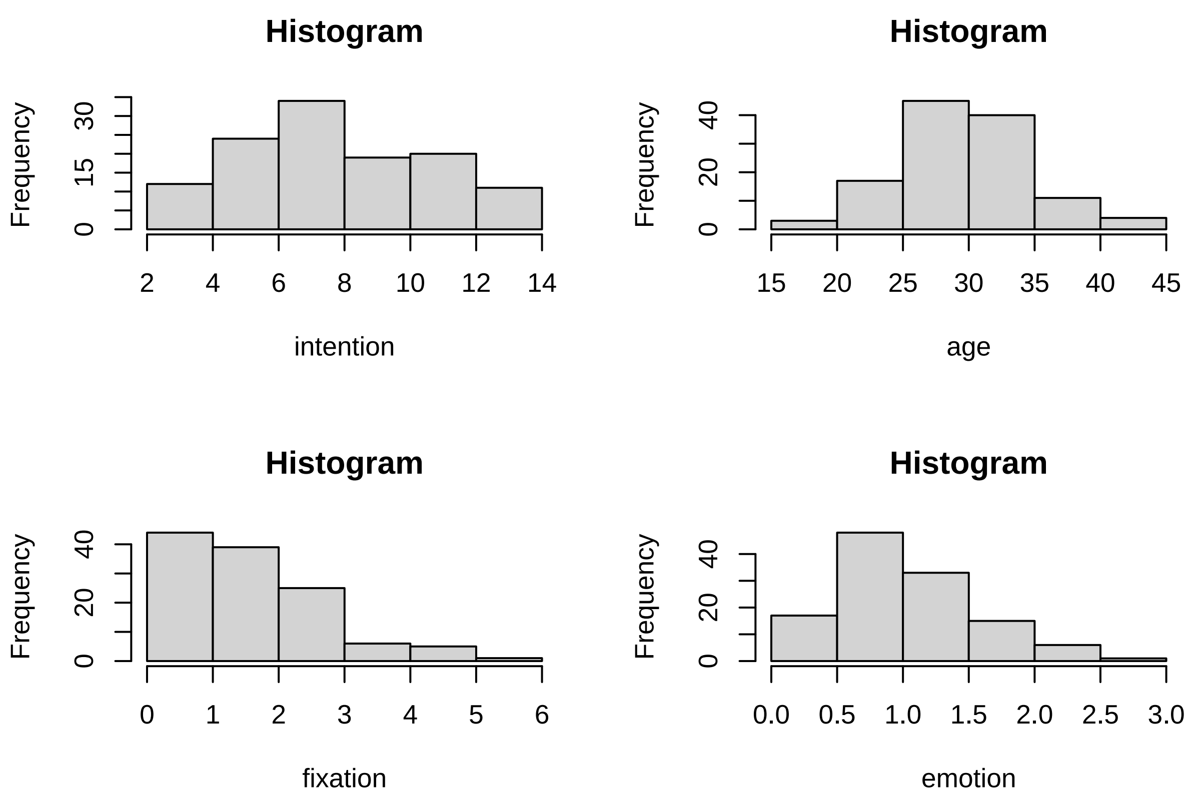
```r
apply(intention[,c(3,5:7)],2,table)
```

```
## $sex
##
##  0  1
## 58 62
##
## $rev
```

```
##
##  1  2  3
## 35 42 43
##
## $educ
##
##  1  2  3
## 30 55 35
##
## $stat
##
##  0  1
## 55 65
```

Some histograms:

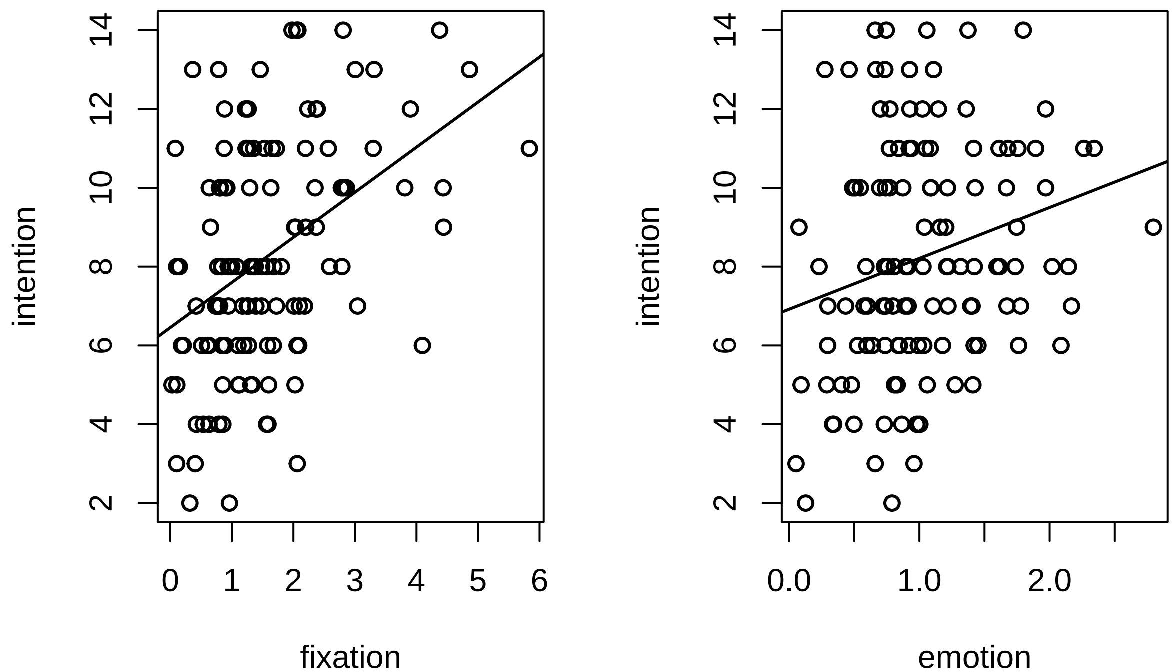
```r
# Histograms
par(mfrow=c(2,2))
hist(intention$intent,xlab="intention",main="Histogram")
hist(intention$age,xlab="age",main="Histogram")
hist(intention$fix,xlab="fixation",main="Histogram")
hist(intention$emo,xlab="emotion",main="Histogram")
```




We can also examine the relationship between fixation and intention as well as between emotion and intention using scatterplots.

```r
# Scatterplots and correlation
par(mfrow=c(1,2))
```

```r
# intention, fixation
plot(intent~fix,xlab="fixation",ylab="intention",data=intention,lwd=1.5)
# to include best linear model
abline(lm(intent~fix,data=intention),lwd=1.5)
# intention, emotion
plot(intent~emo,xlab="emotion",ylab="intention",data=intention,lwd=1.5)
abline(lm(intent~emo,data=intention),lwd=1.5)
```




There seems to be a relationship between both of these explanatory variables and the response variable intention:

- As fixation increases, intention also tends to increase. The relationship is similar for emotion.
- The relationship appears to be stronger for fixation, but it's difficult to tell just by inspection.

Note that the lines in these plots are the fitted linear regression lines, which we'll learn about in more detail soon…

## 2) Correlation

The `cor` function can be used to calculate the correlation. The sample correlation for the variables `intent`, `fix`, `emo` can be found as follows:

```r
attach(intention)
# correlation
cor(cbind(intent,fix,emo))
```

```
##           intent       fix       emo
## intent 1.0000000 0.4262547 0.2347929
```

```
## fix    0.4262547 1.0000000 0.1320953
## emo    0.2347929 0.1320953 1.0000000
```

The correlation between intention and fixation is 0.43 whereas the correlation between emotion and intention is 0.23. This quantifies the strength and the direction of the observed relationships seen in the earlier plots.

We can also carry out statistical tests to assess whether the correlation is significantly different from 0. This can be done using the `cor.test` function, of using `rcorr` (from the `Hmisc` library):

```r
# test
cor.test(intent,fix)
```

```
##
##  Pearson's product-moment correlation
##
## data:  intent and fix
## t = 5.1186, df = 118, p-value = 1.209e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2674470 0.5625183
## sample estimates:
##       cor
## 0.4262547
```

```r
cor.test(intent,emo)
```

```
##
##  Pearson's product-moment correlation
##
## data:  intent and emo
## t = 2.6239, df = 118, p-value = 0.009843
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.05799212 0.39731345
## sample estimates:
##       cor
## 0.2347929
```

```r
cor.test(fix,emo)
```

```
##
##  Pearson's product-moment correlation
##
## data:  fix and emo
## t = 1.4476, df = 118, p-value = 0.1504
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.04828934  0.30413567
## sample estimates:
##       cor
## 0.1320953
```

```r
library("Hmisc")
```

```
##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
```

```
##     format.pval, units
```

```r
# alternative
rcorr(cbind(intent,fix,emo))
```

```
##        intent  fix  emo
## intent   1.00 0.43 0.23
## fix      0.43 1.00 0.13
## emo      0.23 0.13 1.00
##
## n= 120
##
##
## P
##        intent fix    emo
## intent        0.0000 0.0098
## fix    0.0000        0.1504
## emo    0.0098 0.1504
```

We see that the p-value for testing the correlation between intention and fixation is $1.209 \times 10^{-6}$, and thus the correlation is significantly different from 0 (for any reasonable level $\alpha$). The correlation between intention and emotion is also significantly different from 0, with a p-value of 0.0098. Note that the correlation between fixation and emotion is positive ($r = 0.13$), but it is not significantly different from 0 (p-value of 0.15).

## 4) Estimation: simple linear regression

### Least squares estimates

We can fit the linear regression model using the `lm` function in R:

```r
# Simple linear regression
lmod<-lm(intent~fix)
lmod
```

```
##
## Call:
## lm(formula = intent ~ fix)
##
## Coefficients:
## (Intercept)          fix
##       6.453        1.144
```

```r
summary(lmod)
```

```
##
## Call:
## lm(formula = intent ~ fix)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.813 -1.828 -0.207  2.176  6.130
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.4532     0.4285  15.060  < 2e-16 ***
## fix           1.1441     0.2235   5.119 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 2.666 on 118 degrees of freedom
## Multiple R-squared:  0.1817, Adjusted R-squared:  0.1748
## F-statistic:  26.2 on 1 and 118 DF,  p-value: 1.209e-06
```

We obtain parameter estimates $\hat{\beta}_0 = 6.45$ and $\hat{\beta}_1 = 1.14$, with corresponding standard errors 0.4285 and 0.2235. The output also provides an estimate for $\hat{\sigma}$ as the `Residual standard error`, here 2.67.

### Interpretations

The estimated regression line is:

$$\widehat{\texttt{Intention}} = 6.45 + 1.14\,\texttt{Fixation}$$

We can interpret the regression coefficients as follows:

- **Slope**: $\hat{\beta}_1 = 1.14$ : for every 1 second increase in fixation, the intention increases by 1.14, on average. So the longer a person fixates on an ad, the greater their intention is to buy the product.
- **Intercept**: $\hat{\beta}_0 = 6.45$. This represents the mean of intention when fixation=0. In this example, we have only included people who looked at the ad. The value fixation=0 is not actually possible. Therefore, the parameter does not have a reasonable interpretation here.

### Hypothesis testing & confidence intervals

Results for the tests $H_0 : \beta_j = 0$ vs. $H_1 : \beta_j \neq 0$ are given within the model `summary`:

```r
# Simple linear regression
summary(lmod)
```

```
##
## Call:
## lm(formula = intent ~ fix)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.813 -1.828 -0.207  2.176  6.130
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.4532     0.4285  15.060  < 2e-16 ***
## fix           1.1441     0.2235   5.119 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.666 on 118 degrees of freedom
## Multiple R-squared:  0.1817, Adjusted R-squared:  0.1748
## F-statistic:  26.2 on 1 and 118 DF,  p-value: 1.209e-06
```

For example, for the test involving the slope $H_0 : \beta_1 = 0$ vs. $H_1 : \beta_1 \neq 0$,

- $t = 1.1441/0.2235 = 5.119$
- p-value $= 1.21 \times 10^{-06}$

So we can reject $H_0$ (for any reasonable $\alpha$, e.g. $\alpha = 0.05$) and conclude fixation has a significant (linear) effect on intention.

We can also obtain confidence intervals using the `confint` function:

```r
# confidence intervals
confint(lmod,level=0.95)
```

```
##                 2.5 %   97.5 %
## (Intercept) 5.6046573 7.301720
## fix         0.7014641 1.586703
```

```r
confint(lmod,level=0.99)
```

```
##                 0.5 %   99.5 %
## (Intercept) 5.3313373 7.575040
## fix         0.5588921 1.729275
```

A 95% C.I. for $\hat{\beta}_1$ is $(0.7014641, 1.586703)$. Since the C.I. does not contain 0, we can conclude that $\beta_1$ is significantly different from 0 - exactly as before, at the $\alpha = 5\%$ significance level.

## 5) Prediction

Recall that our fitted model has the form

$$\widehat{intention} = 6.45 + 1.14\,fixation$$

Suppose that we're interested in predicting intention for different values of fixation, ranging from 1 to 10 seconds. We can do this in R using `predict.lm`. To start, we'll create a new dataset containing the values of the explanatory variabe (`fix`) for the predictions we want to make:

```r
# Create new data file containing the explanatory
# variables going from 1 to 10
new<-data.frame(fix=c(1:10))
new
```

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

Note that it's important that we use the same variable name, here `fix`, to use `predict.lm`.

We can obtain the predicted values (which is the same as the estimated mean) for the values of fixation in `newdata` :

```r
predict(lmod,newdata=new)
```

```
##         1         2         3         4         5         6         7         8
##  7.597272  8.741356  9.885440 11.029523 12.173607 13.317691 14.461775 15.605858
##         9        10
## 16.749942 17.894026
```

We can also obtain confidence and prediction intervals:

```r
# estimates + confidence interval for estimated mean
predict.lm(lmod,newdata=new,interval=c("confidence"),
           level=0.95)
```

```
##          fit       lwr       upr
## 1   7.597272  7.051659  8.142885
## 2   8.741356  8.224436  9.258276
## 3   9.885440  9.092632 10.678247
## 4  11.029523  9.854064 12.204983
## 5  12.173607 10.584051 13.763164
## 6  13.317691 11.301878 15.333503
## 7  14.461775 12.013891 16.909658
## 8  15.605858 12.722702 18.489015
## 9  16.749942 13.429570 20.070315
## 10 17.894026 14.135172 21.652880
```

```r
# predicitons + prediction intervals
predict.lm(lmod,newdata=new,interval=c("prediction"),
           level=0.95)
```

```
##          fit       lwr      upr
## 1   7.597272  2.289540 12.90500
## 2   8.741356  3.436497 14.04622
## 3   9.885440  4.546632 15.22425
## 4  11.029523  5.620639 16.43841
## 5  12.173607  6.659896 17.68732
## 6  13.317691  7.666335 18.96905
## 7  14.461775  8.642285 20.28126
## 8  15.605858  9.590302 21.62141
## 9  16.749942 10.513020 22.98686
## 10 17.894026 11.413030 24.37502
```

```r
##
# displaying things together in one table
library("dplyr")
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:Hmisc':
##
##     src, summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
pred1<-data.frame(predict.lm(lmod,newdata=new,interval=c("confidence"),
                             level=0.95))
pred2<-data.frame(predict.lm(lmod,newdata=new,interval=c("prediction"),
                             level=0.95))
predictions<-left_join(pred1,pred2,by=c("fit"))
names(predictions)<-c("prediction","lwr.ci","upr.ci","lwr.pi","upr.pi")
predictions
```

```
##    prediction    lwr.ci    upr.ci    lwr.pi   upr.pi
## 1    7.597272  7.051659  8.142885  2.289540 12.90500
## 2    8.741356  8.224436  9.258276  3.436497 14.04622
```

```
## 3    9.885440  9.092632 10.678247  4.546632 15.22425
## 4   11.029523  9.854064 12.204983  5.620639 16.43841
## 5   12.173607 10.584051 13.763164  6.659896 17.68732
## 6   13.317691 11.301878 15.333503  7.666335 18.96905
## 7   14.461775 12.013891 16.909658  8.642285 20.28126
## 8   15.605858 12.722702 18.489015  9.590302 21.62141
## 9   16.749942 13.429570 20.070315 10.513020 22.98686
## 10  17.894026 14.135172 21.652880 11.413030 24.37502
```

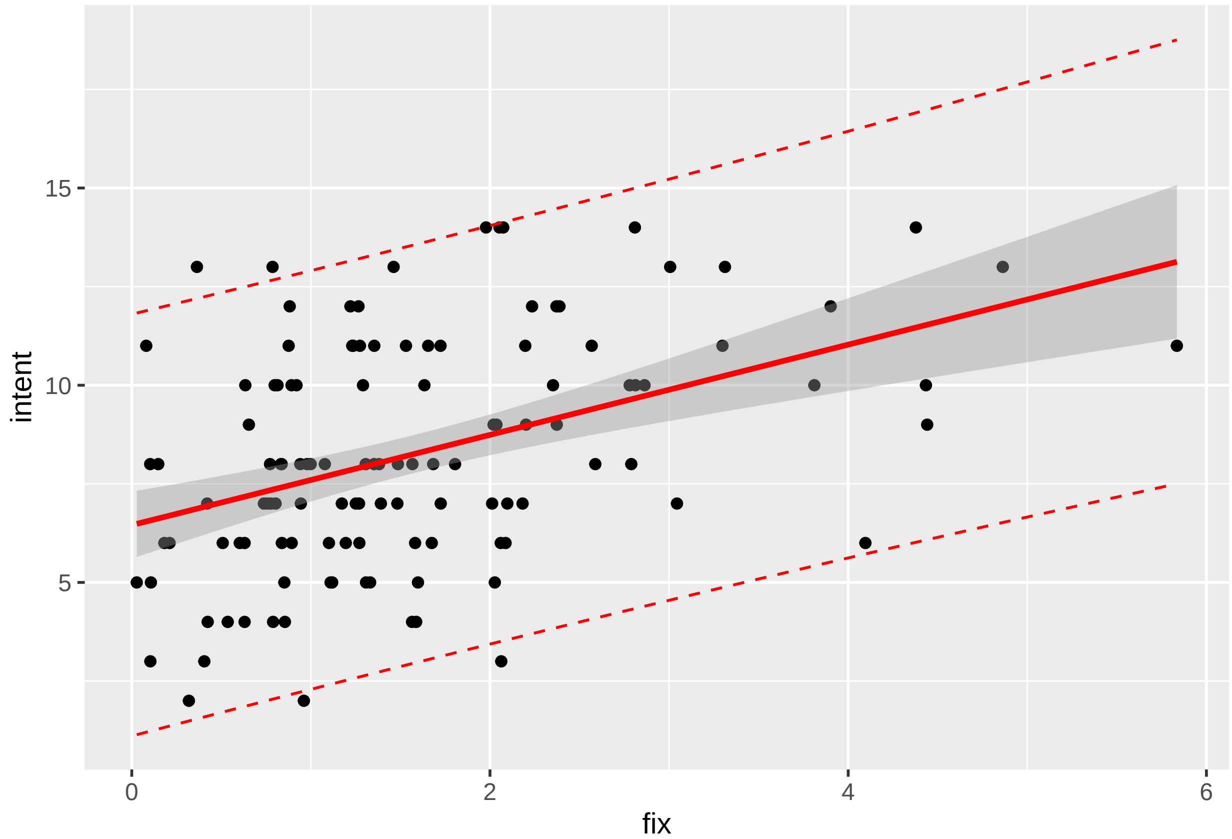
We can see that the prediction intervals are wider than the confidence intervals. We can also visualize this using `ggplot`:

```r
# graphs with ggplot
library(ggplot2)
pred.inter<- predict(lmod, interval="prediction")
```

```
## Warning in predict.lm(lmod, interval = "prediction"): predictions on current data refer to _future_ r
```

```r
intention_new<-cbind(intention,pred.inter)
# linear trend, confidence + prediction intervals
ggplot(intention_new, aes(x=fix, y=intent)) +
  geom_point() +
  geom_smooth(method=lm , color="red", se=TRUE)+
  geom_line(aes(y=lwr), color = "red", linetype = "dashed")+
  geom_line(aes(y=upr), color = "red", linetype = "dashed")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



## 6) Residuals

We'll now consider an analysis of the residuals for the simple linear regression model of intention vs. fixation. To obtain the **ordinary residuals**, we can use the `resid` function, or simply call on the `residuals` output form the fitted model.

```
# ordinary residuals
resid(lmod)
lmod$residuals
```

The **standardized residuals** can be obtained using `rstandard`:

```
# standardized
rstandard(lmod)
```

And finally, the (jackknife) **studentized residuals** can be obtained using `rstudent`:

```
rstudent(lmod)
```

While we could carry out our residual analysis using the ordinary residuals, it's generally preferable to consider the standardized or studentized versions. We'll focus on the studentized residuals here. Note that we can create our plots using simple/built-in R functions, or using `ggplot`. Both are shown in the R code, although here we'll look at the plots created using `ggplot`. First, we'll create a dataframe which includes the residuals to create the plots:

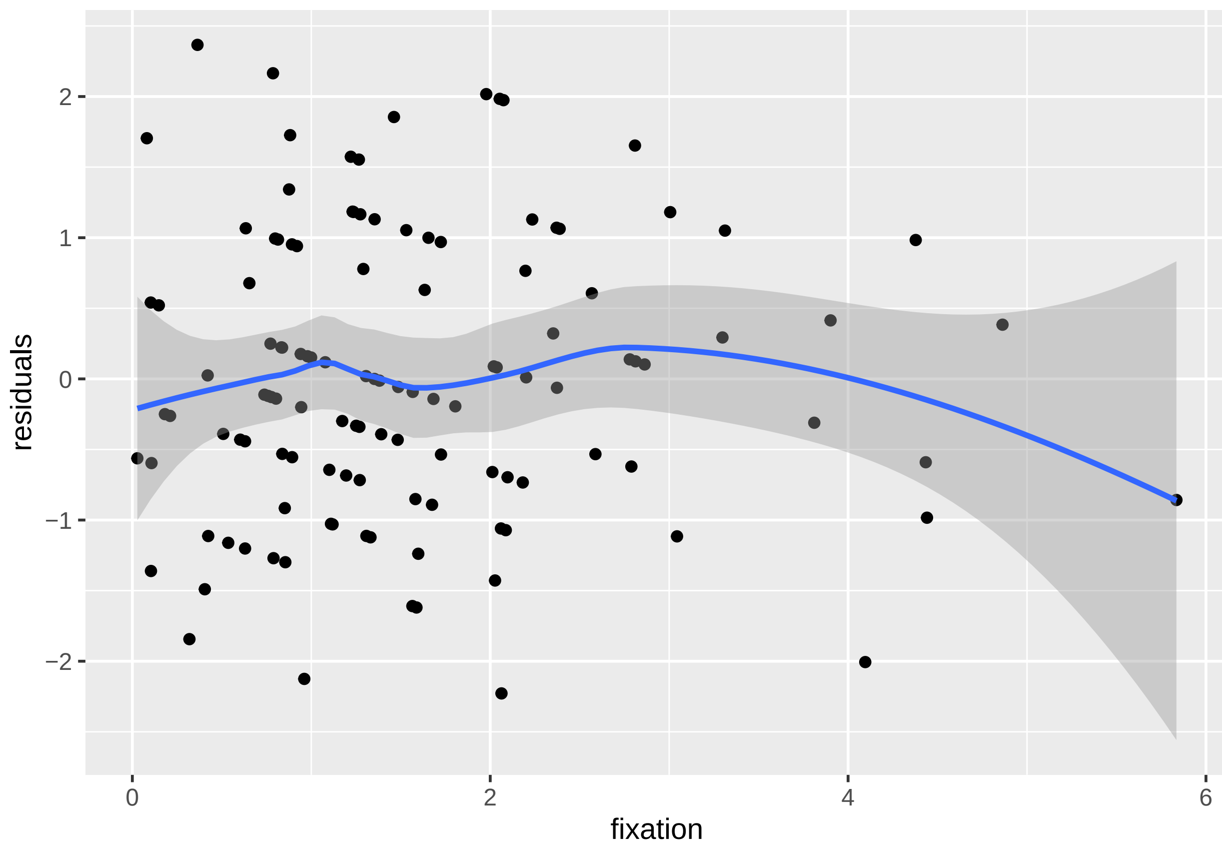
```
# plots using ggplot:
# preparing a dataframe with the residuals:
library("ggplot2")
res.dat<-data.frame(cbind(intent,fix,lmod$fitted,lmod$residuals,
                          rstandard(lmod),rstudent(lmod)))
names(res.dat)<-c("intent","fix","fitted","resid","rstand","rstud")
head(res.dat)
```

```
##   intent   fix   fitted     resid     rstand      rstud
## 1     11 0.081 6.545859  4.454141  1.6911400  1.7047445
## 2     12 2.235 9.010216  2.989784  1.1278347  1.1291480
## 3      6 1.675 8.369529 -2.369529 -0.8925168 -0.8917419
## 4      4 0.630 7.173961 -3.173961 -1.1993018 -1.2015546
## 5     11 2.197 8.966740  2.033260  0.7668732  0.7655268
## 6      4 0.424 6.938280 -2.938280 -1.1119663 -1.1130917
```

We'll start off by assessing the model specification. Recall, we can check this assumption by looking at a plot of the residuals as a function of the explanatory variable. It's helpful to also include a loess curve (i.e. a smoothed curve to the data) to better discern any patterns or trends.

```
# resid vs. fix + smooth
ggplot(data = res.dat,
       aes(x = fix, y = rstud)) +
  geom_point() +
  geom_smooth() +
  theme(legend.position = "bottom") +
  ylab("residuals") +
  xlab("fixation")
```

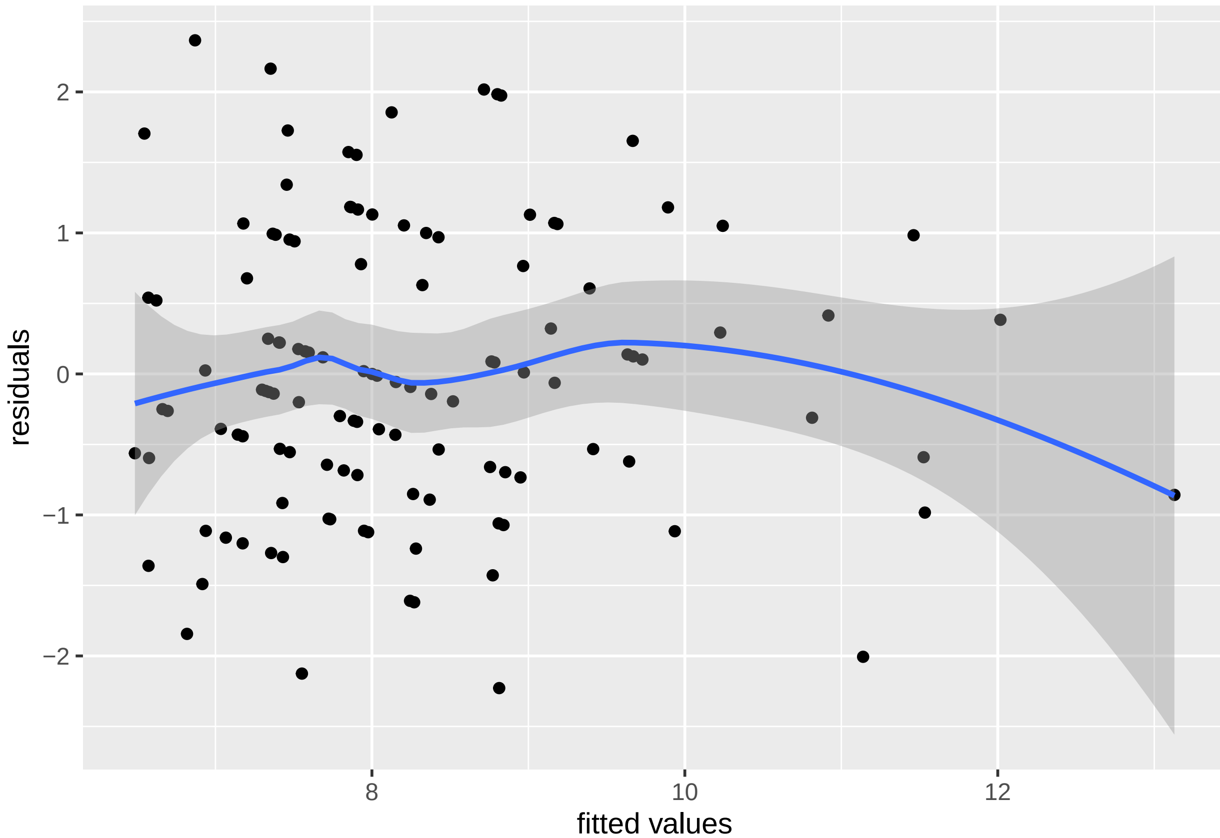
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




Note that in the case of simple linear regression, when there is a single explanatory variable in the model, the plot of residuals vs $X$ will show the same pattern as the plot of the residuals vs. the fitted values. This follows since the fitted values are simply a linear transformation of $X$, i.e. $\hat{\beta}_0 + \hat{\beta}_1 X$. (Note that this is not the case when there are several explanatory variables, i.e. in multiple linear regression).

```
# resid vs. fitted + smooth
ggplot(data = res.dat,
       aes(x = fitted, y = rstud)) +
  geom_point() +
  geom_smooth() +
  theme(legend.position = "bottom") +
  ylab("residuals") +
  xlab("fitted values")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

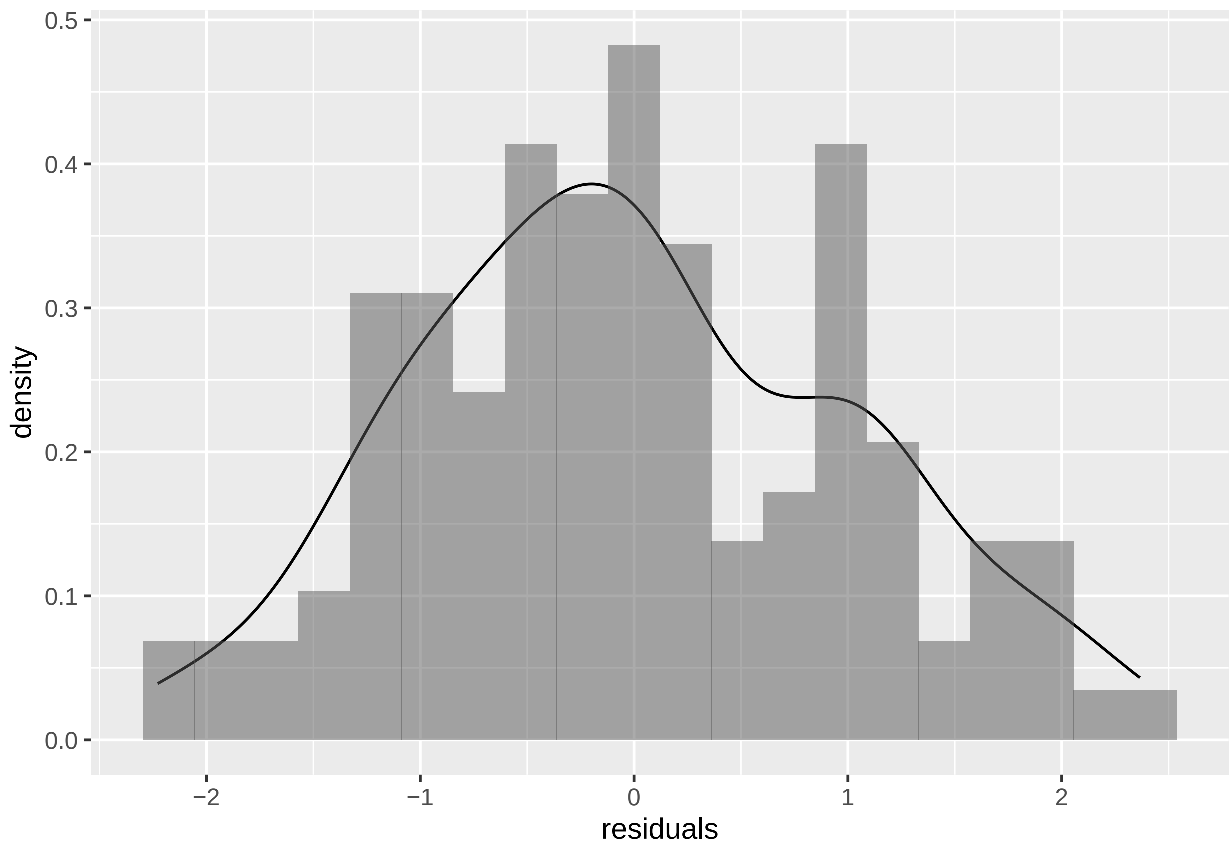



Both plots do not show any clear pattern that would indicate that the model is poorly specified, nor that there is heteroscedasticity (non constant variance).

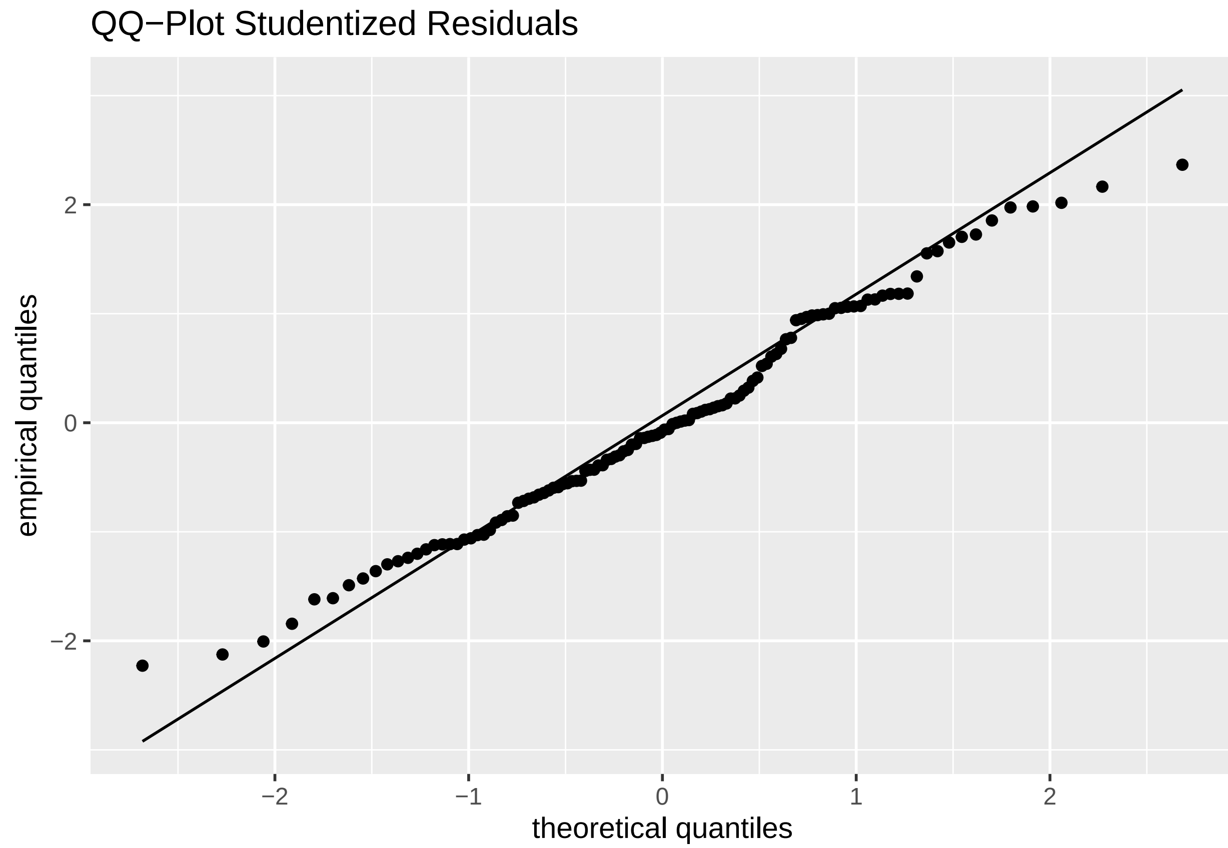
Next we can assess the normality assumption. To do this, we can consider a histogram and QQ-plot of the (studentized) residuals:

```
# histogram rstud
ggplot(data = res.dat, mapping = aes(x = rstud)) +
  geom_density() +
  geom_histogram(aes(y = ..density..), bins = 20, alpha = 0.5) +
  xlab("residuals")
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```r
# qqplot rstud
ggplot(data = res.dat, mapping = aes(sample = rstud)) +
  stat_qq(distribution = qt, dparams = lmod$df.residual) +
  stat_qq_line(distribution = qt, dparams = lmod$df.residual) +
  labs(x = "theoretical quantiles",
       y = "empirical quantiles") +
  ggtitle("QQ-Plot Studentized Residuals")
```



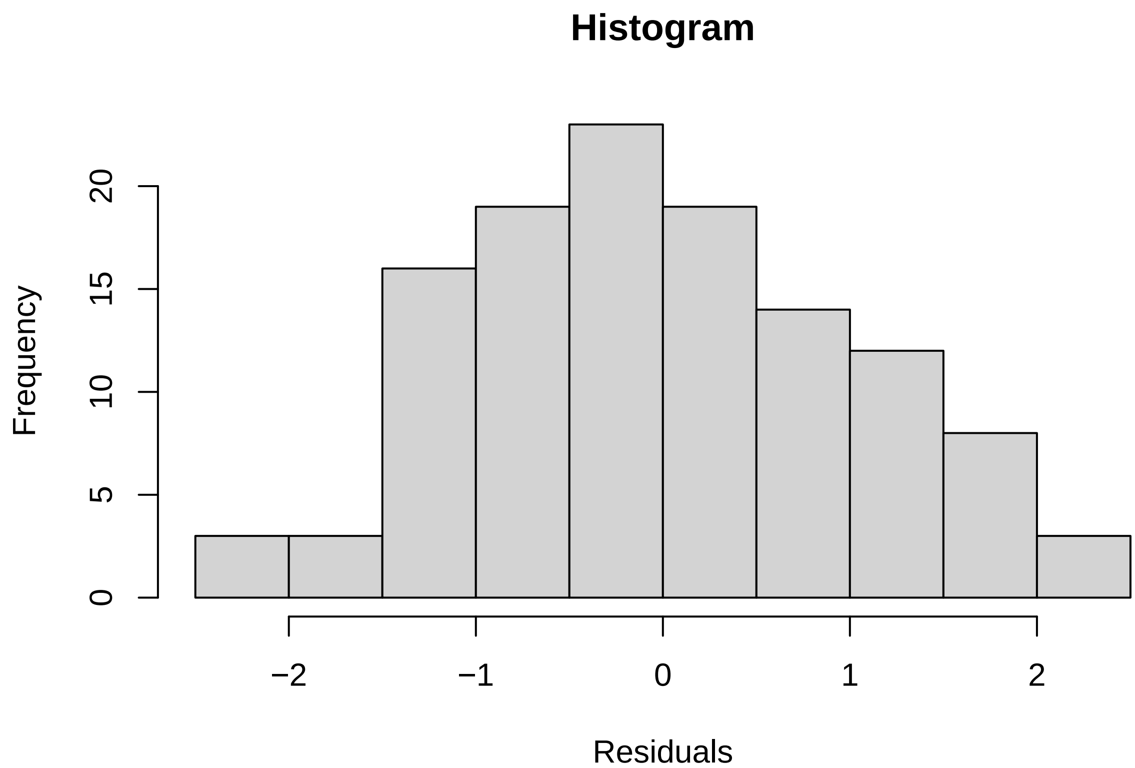

These plots (histogram + qq-plot) suggest that the assumption of normality is reasonably met here.

*Small technical detail: if the random error terms are indeed* $\epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$*, it can be shown that the jackknife studentized residuals actually follow a student-t distribution, with* $n-2$ *degrees of freedom in the case of simple linear regression. This is used in creating the qq-plot above.*

Overall, in this example, the residual analysis has demonstrated that there is no reason to doubt the underlying assumptions of the model, and thus the model seems valid.

The following chunks of code provide similar plots using more basic functions in R.

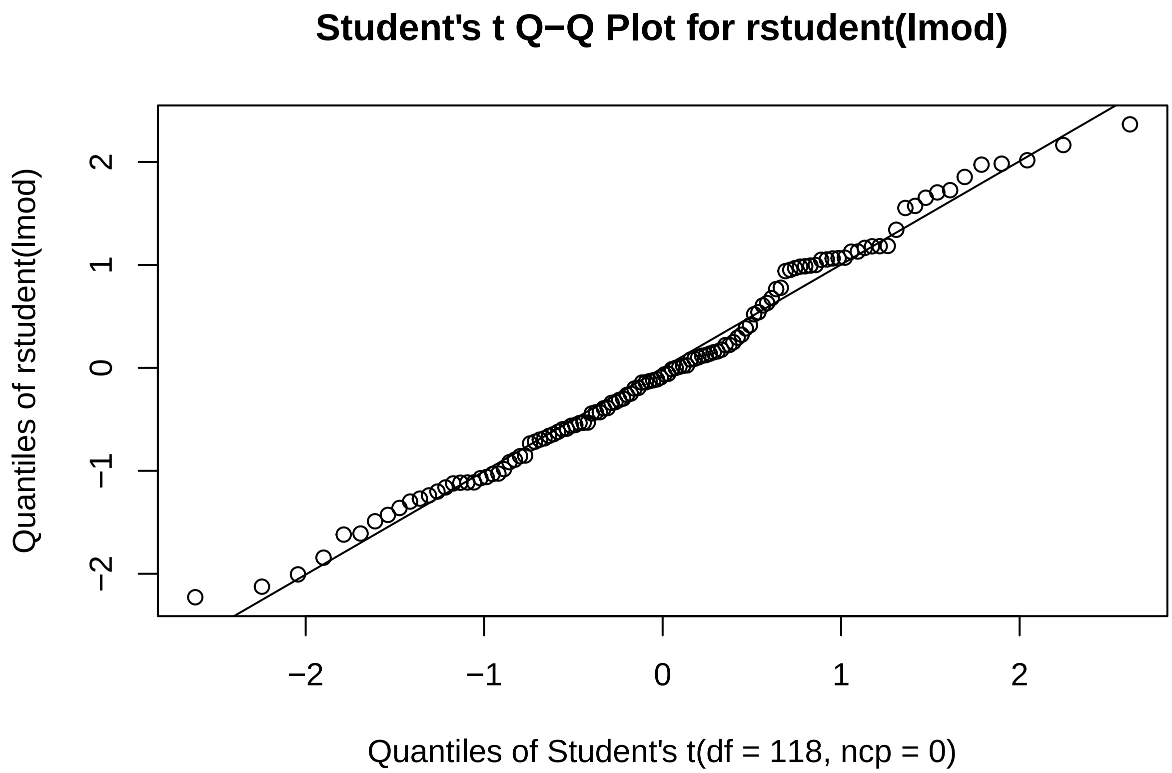
```
# plots using basic R functions:
hist(rstudent(lmod),xlab="Residuals",main="Histogram")
```



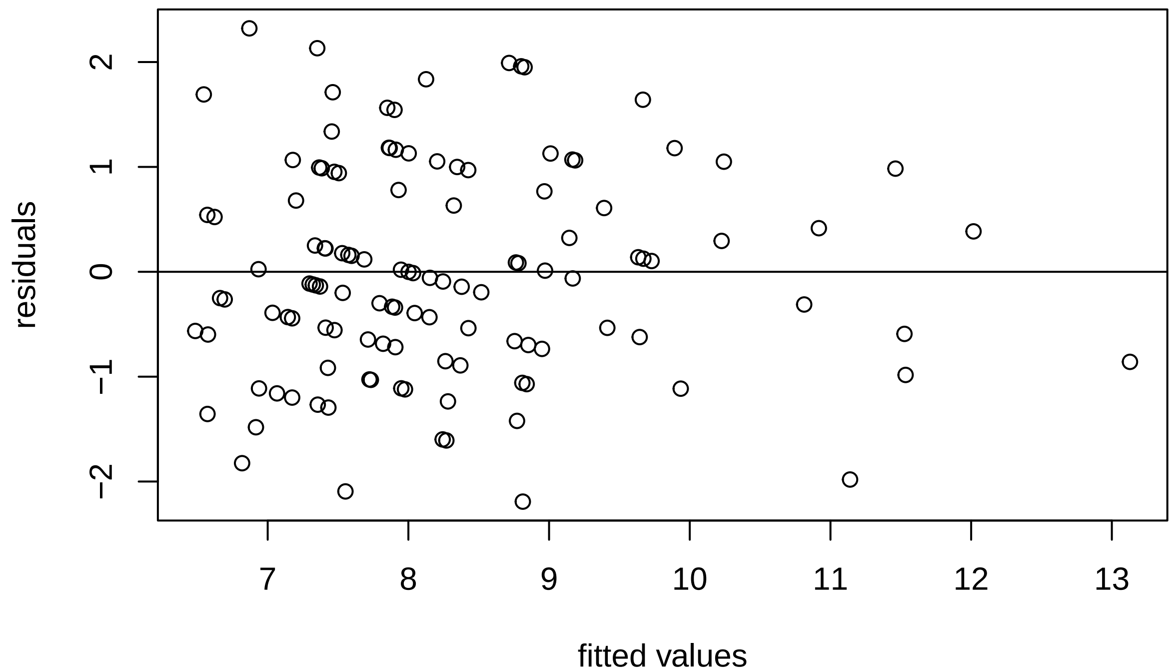

```r
# qqplot: alternative
library("EnvStats")
```

```
##
## Attaching package: 'EnvStats'

## The following object is masked from 'package:Hmisc':
##
##     stripChart

## The following objects are masked from 'package:stats':
##
##     predict, predict.lm

## The following object is masked from 'package:base':
##
##     print.default
```

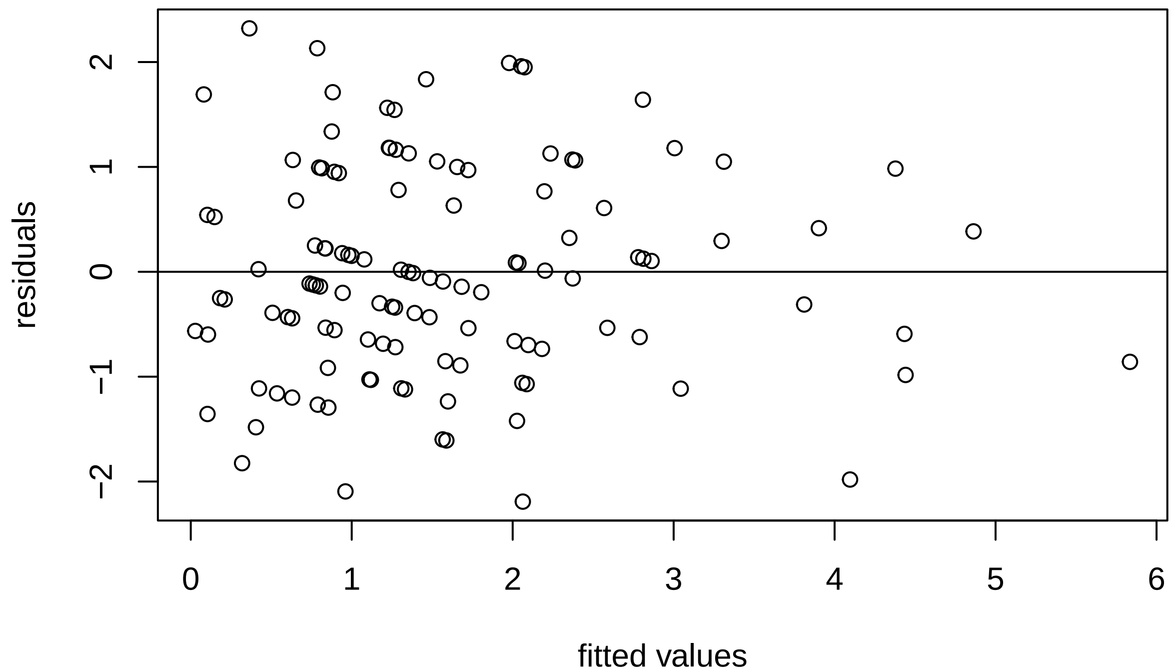
```r
qqPlot(rstudent(lmod),
       distribution = "t", param.list=list(df = lmod$df.residual),
       add.line=TRUE)
```




```r
# scatterplots
plot(rstandard(lmod)~lmod$fitted.values,
     xlab="fitted values",ylab="residuals")
abline(h=0)
```




```r
plot(rstandard(lmod)~fix,
     xlab="fitted values",ylab="residuals")
abline(h=0)
```




### 7) $R^2$

The coefficient of determination is given in the summary of the model:

```r
summary(lmod)
```

```
##
## Call:
## lm(formula = intent ~ fix)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.813 -1.828 -0.207  2.176  6.130
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.4532     0.4285  15.060  < 2e-16 ***
## fix           1.1441     0.2235   5.119 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.666 on 118 degrees of freedom
## Multiple R-squared:  0.1817, Adjusted R-squared:  0.1748
## F-statistic:  26.2 on 1 and 118 DF,  p-value: 1.209e-06
```

Here, we obtain $R^2 = 0.182$, indicating that `fix` explains 18.4% of the variability in `intent`.

## 8) Binary predictor

Incorporating a binary predictor in a linear regression model can be done in different ways. If the variable is coded as 0/1, we can simply include it as is:

```
lmod1<-lm(intent~sex)
summary(lmod1)
```

```
##
## Call:
## lm(formula = intent ~ sex)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9194 -2.5517  0.0806  2.1726  5.4483
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.5517     0.3763  20.070   <2e-16 ***
## sex           1.3676     0.5235   2.613   0.0102 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.866 on 118 degrees of freedom
## Multiple R-squared:  0.05468,    Adjusted R-squared:  0.04667
## F-statistic: 6.826 on 1 and 118 DF,  p-value: 0.01015
```

Here we obtain $\hat{\beta}_1 = 1.37$, so we can say that the mean intention to buy score is 1.37 units higher for women than for men. In other words, on average, women are more interested in buying the product than men. Moreover, this difference is significant (p-value 0.01).

We can also treat the `sex` variable as categorical using the `as.factor` function. We obtain identical results:

```
lmod2<-lm(intent~as.factor(sex))
summary(lmod2)
```

```
##
## Call:
## lm(formula = intent ~ as.factor(sex))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9194 -2.5517  0.0806  2.1726  5.4483
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       7.5517     0.3763  20.070   <2e-16 ***
## as.factor(sex)1   1.3676     0.5235   2.613   0.0102 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.866 on 118 degrees of freedom
## Multiple R-squared:  0.05468,    Adjusted R-squared:  0.04667
## F-statistic: 6.826 on 1 and 118 DF,  p-value: 0.01015
```

Notice that the way in which the results are displayed is slightly different now, since R treats `sex` as categorical rather than numerical.

When handling categorical variables in R, we can adjust which level is the *reference* level (i.e., which level is absorbed into the intercept). Previously, the `sex=0` level was the reference. We can change this as follows:

```r
levels(as.factor(sex))
```

```
## [1] "0" "1"
```

```r
sex<-relevel(as.factor(sex),2)
lmod3<-lm(intent~sex)
summary(lmod3)
```

```
##
## Call:
## lm(formula = intent ~ sex)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9194 -2.5517  0.0806  2.1726  5.4483
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.9194     0.3639  24.509   <2e-16 ***
## sex0         -1.3676     0.5235  -2.613   0.0102 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.866 on 118 degrees of freedom
## Multiple R-squared:  0.05468,    Adjusted R-squared:  0.04667
## F-statistic: 6.826 on 1 and 118 DF,  p-value: 0.01015
```

In this case, the parameter estimates are different, but the model itself is indeed still equivalent. Both models yield:

$$\widehat{E}(intention|sex = 0) = 7.55$$
$$\widehat{E}(intention|sex = 1) = 8.92$$

With this second model, $\hat{\beta}_1$ has the same magnitude but different sign as now it represents the difference in the mean intention to by for males vs. females.