STAT 378: Linear Regression Analysis

Adam B Kashlak

2025-07-22

Table of contents

Preface

> I never felt such a glow of loyalty and respect towards the sovereignty and magnificent sway of mathematical analysis as when his answer reached me confirming, by purely mathematical reasoning, my various and laborious statistical conclusions.

Regression towards Mediocrity in Hereditary Stature
Sir Francis Galton, FRS (1886)

This collection of lecture notes is an updated version of my original lecture ntoes from 2017. They are now typeset in Quarto thanks to the suggestion of my former PhD student, Dr. Katie L Burak, who is currently an assistant professor of teaching at the University of British Columbia.

These revised and enhanced notes have some nice additions. Most notably, there is embedded `R` code and datasets within the text, which is (perhaps) the main point of using Quarto over my old LaTex notes. I have also taken it upon myself to add new sections and expand on those I find most interesting. This mostly occurs at the end of the notes with additional sections on advanced topics like logisitic regression, LASSO, and some Bayesian regression models.

The material in these notes is now too much for a single semester course in linear regression. Of note, I personally plan to skip the sections on influential points in regression models as the formulae are both tedious and ad-hoc. Nevertheless, I did copy these bits into the new version of my notes for completeness sake.

Adam B Kashlak
Edmonton, Canada
August 2025

The following are lecture notes originally produced for an upper level undergraduate course on linear regression at the University of Alberta in the fall of 2017. Regression is one of the main, if not the primary, workhorses of statistical inference. Hence, I do hope you will find these notes useful in learning about regression.

The goal is to begin with the standard development of ordinary least squares in the multiple regression setting, then to move onto a discussion of model assumptions and issues that can arise in practice, and finally to discuss some specific instances of generalized linear models (GLMs) without delving into GLMs in full generality. Of course, what follows is by no means a unique exposition but is mostly derived from three main sources: the text, *Linear Regression Analysis*, by Montgomery, Peck, and Vining; the course notes of Dr. Linglong Kong who lectured this same course in 2013; whatever remains inside my brain from supervising (TAing) undergraduate statistics courses at the University of Cambridge during my PhD years.

Adam B Kashlak
Edmonton, Canada
August 2017

1 Ordinary Least Squares

1.1 Introduction

Linear regression begins with the simple but profound idea that some observed output or {response} variable, $Y \in \mathbb{R}$, is a function of p input or *regressor* variables $x_1, \dots, x_p$ with the addition of some unknown noise variable ε. Namely,

$$Y = f(x_1, \dots, x_p) + \varepsilon$$

where the noise is generally assumed to have mean zero and finite variance. The function f is unknown and relates the inputs x_i to the output Y. Our goal is to ascertain what f could be.

In this setting, Y is usually considered to be a random variable while the x_i are considered fixed. Hence, the expected value of Y is in terms of the unknown function f and the regressors:

$$\mathrm{E}\left(Y | x_1, \dots, x_p\right) = f(x_1, \dots, x_p).$$

While f can be considered to be in some very general classes of functions, we begin with the standard linear setting. Let $\beta_0, \beta_1, \dots, \beta_p \in \mathbb{R}$. Then, the *multiple regression model* is

$$Y = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = \beta^{\mathrm{T}} X$$

where $\beta = \left(\beta_0, \dots, \beta_p\right)^{\mathrm{T}}$ and $X = \left(1, x_1, \dots, x_p\right)^{\mathrm{T}}$. The *simple regression model* is a submodel of the above where $p = 1$, which is

$$Y = \beta_0 + \beta_1 x_1 + \varepsilon,$$

and will be treated concurrently with multiple regression.

In the statistics setting, the parameter vector $\beta \in \mathbb{R}^p$ is unknown. The analyst observes multiple replications of regressor and response pairs, $(X_1, Y_1), \dots, (X_n, Y_n)$ where n is the *sample size*, and wishes to choose a "best' ' estimate for β based on these n observations. This setup can be concisely written in a vector-matrix form as

$$Y = X\beta + \varepsilon \tag{1.1}$$

where

$$Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}, \quad X = \begin{pmatrix} 1 & x_{1,1} & \dots & x_{1,p} \\ 1 & x_{2,1} & \dots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \dots & x_{n,p} \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$

Note that $Y, \varepsilon \in \mathbb{R}^n$, $\beta \in \mathbb{R}^{p+1}$, and $X \in \mathbb{R}^{n \times p+1}$.

As Y is a random variable, we can compute its mean vector and covariance matrix as follows:

$$\mathrm{E}Y = \mathrm{E}\left(X\beta + \varepsilon\right) = X\beta$$

and

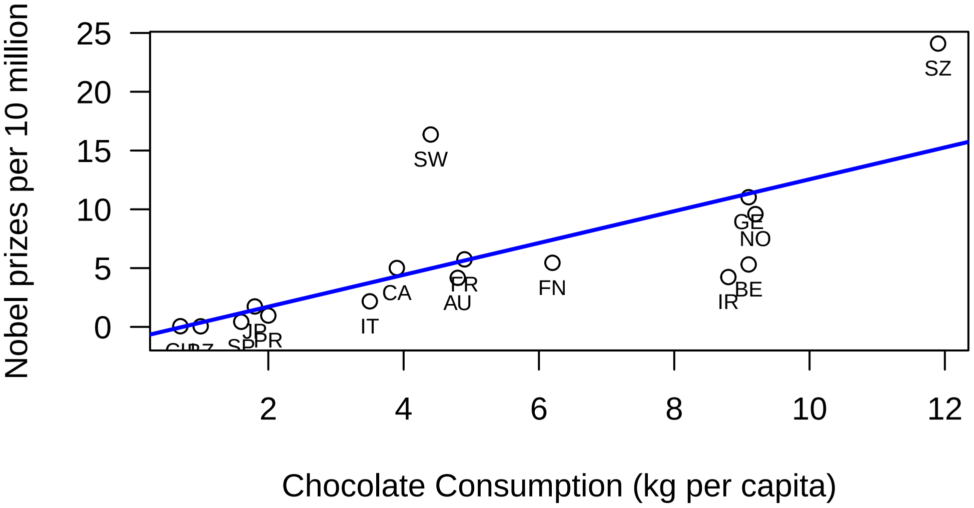
$$\mathrm{Var}\left(Y\right) = \mathrm{E}\left((Y - X\beta)(Y - X\beta)^{\mathrm{T}}\right) = \mathrm{E}\left(\varepsilon\varepsilon^{\mathrm{T}}\right) = \mathrm{Var}\left(\varepsilon\right) = \sigma^2 I_n.$$

An example of a linear regression is this following study from the New England Journal of Medicine[1] can be found in the code below. This study highlights the correlation between chocolate consumption and Nobel prizes received in 16 different countries.

```r
# Read in Table
dat = read.table("data/chocoTable.r");

# Plot data
plot(
  dat$choco, dat$nobel,
  xlab="Chocolate Consumption (kg per capita)",
  ylab="Nobel prizes per 10 million",
  las=1,#xlim=c(0,max(dat$choco)+1),
  ylim=c(-1,max(dat$nobel))
);

# Label data
text(
  x=dat$choco, y=dat$nobel,
  labels=dat$abbrev,
  pos=1,cex=0.7
)

# Construct a linear model
lmDat = lm( nobel ~ choco, data = dat );
```

[1] Messerli, F. H. (2012). Chocolate consumption, cognitive function, and Nobel laureates. New England Journal of Medicine, 367(16), 1562-1564.

```
# plot regression line
abline(lmDat,col='blue',lwd=2)
```



1.1.1 Definitions

Before continuing, we require the following collection of terminology.

The *response* Y and the *regressors* X were already introduced above. These elements comprise the *observed data* in our regression. The *noise* or *error* variable is ε. The entries in this vector are usually considered to be independent and identically distributed (iid) random variables with mean zero and finite variance $\sigma^2 < \infty$. Very often, this vector will be assumed to have a multivariate normal distribution: $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I_n\right)$ where I_n is the $n \times n$ identity matrix. The variance σ^2 is also generally considered to be unknown to the analyst.

The unknown vector β is our *parameter* vector. Eventually, we will construct an *estimator* $\hat{\beta}$ from the observed data. Given such an estimator, the *fitted values* are $\hat{Y} := X\hat{\beta}$. These values are what the model believes are the expected values at each regressor.

Given the fitted values, the *residuals* are $r = Y - \hat{Y}$ which is a vector with entries $r_i = Y_i - \hat{Y}_i$. This is the difference between the observed response and the expected response of our model. The residuals are of critical importance to testing how good our model is and will reappear in most subsequent sections.

Table 1.1: The four variables in the linear regression model of Equation 1.1 split between whether they are fixed or random variables and between whether or not the analyst knows their value.

	Known	Unknown
Fixed	X	β
Random	Y	ε

Lastly, there is the concept of sum of squares. Letting $\bar{Y} = n^{-1}\sum_{i=1}^{n} Y_i$ be the sample mean for Y, the *total sum of squares* is $SS_{\text{tot}} = \sum_{i=1}^{n}(Y_i - \bar{Y})^2$, which can be thought of as the total variation of the responses. This can be decomposed into a sum of the *explained sum of squares* and the *residual sum of squares* as follows:

$$SS_{\text{tot}} = SS_{\text{exp}} + SS_{\text{res}} = \sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2 + \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

The explained sum of squares can be thought of as the amount of variation explained by the model while the residual sum of squares can be thought of as a measure of the variation that is not yet contained in the model. The sum of squares gives us an expression for the so called *coefficient of determination*, $R^2 = SS_{\text{exp}}/SS_{\text{tot}} = 1 - SS_{\text{res}}/SS_{\text{tot}} \in [0, 1]$, which is treated as a measure of what percentage of the variation is explained by the given model.

1.2 Point Estimation

In the ordinary least squares setting, the our choice of estimator is

$$\hat{\beta} = \arg\min_{\tilde{\beta} \in \mathbb{R}^p} \sum_{i=1}^{n}(Y_i - X_{i,\cdot} \cdot \tilde{\beta})^2 \tag{1.2}$$

where $X_{i,\cdot}$ is the ith row of the matrix X. In the simple regression setting, this reduces to

$$(\hat{\beta}_0, \hat{\beta}_1) = \arg\min_{(\tilde{\beta}_0, \tilde{\beta}_1) \in \mathbb{R}^2} \sum_{i=1}^{n}(Y_i - (\tilde{\beta}_0 + \tilde{\beta}_1 x_i))^2.$$

Note that this is equivalent to choosing a $\hat{\beta}$ to minimize the sum of the squared residuals.

It is perfectly reasonable to consider other criterion beyond minimizing the sum of squared residuals. However, this approach results in an estimator with many nice properties. Most notably is the Gauss-Markov theorem:

Theorem 1.1 (Gauss-Markov Theorem)**.** *Given the regression setting from Equation 1.1 and that for the errors,* $\mathrm{E}\varepsilon_i = 0$ *for* $i = 1, \dots, n$*,* $\mathrm{Var}\left(\varepsilon_i\right) = \sigma^2$ *for* $i = 1, \dots, n$*, and* $\mathrm{cov}\left(\varepsilon_i, \varepsilon_j\right) = 0$ *for* $i \neq j$*, then the least squares estimator results in the minimal variance over all linear unbiased estimators.*
(This is sometimes referred to as the "Best Linear Unbiased Estimator' ' or BLUE)

Hence, it can be shown that the estimator is unbiased, $\mathrm{E}\hat{\beta} = \beta$. Furthermore, as long as the model contains an intercept term β_0, the constructed least squares line passes through the centre of the data in the sense that the sum of the residuals is zero, $\sum_{i=1}^{n} r_i = 0$ and that $\bar{Y} = \hat{\beta}\bar{X}$ where $\bar{Y} = n^{-1}\sum_{i=1}^{n} Y_i$ is the sample mean of the Y_i and where $\bar{X}$ is the vector of column means of the matrix X.

1.2.1 Derivation of the OLS estimator

The goal is to derive an explicit solution to Equation 1.2. First, consider the following partial derivative:

$$\begin{aligned}
\frac{\partial}{\partial \hat{\beta}_k} \sum_{i=1}^{n} (Y_i - X_{i,\cdot} \cdot \hat{\beta})^2 &= -2 \sum_{i=1}^{n} (Y_i - X_{i,\cdot} \cdot \hat{\beta}) X_{i,k} \\
&= -2 \sum_{i=1}^{n} (Y_i - \textstyle\sum_{j=1}^{p+1} X_{i,j}\hat{\beta}_j) X_{i,k} \\
&= -2 \sum_{i=1}^{n} Y_i X_{i,k} + 2 \sum_{i=1}^{n} \sum_{j=1}^{p+1} X_{i,j} X_{i,k} \hat{\beta}_j
\end{aligned}$$

The above is the kth entry in the vector $\nabla \sum_{i=1}^{n} (Y_i - X_{i,\cdot}\hat{\beta})^2$. Hence,

$$\nabla \sum_{i=1}^{n} (Y_i - X_{i,\cdot}\hat{\beta})^2 = -2X^{\mathrm{T}}Y + 2X^{\mathrm{T}}X\hat{\beta}.$$

Setting this equal to zero results in a critical point at

$$X^{\mathrm{T}}Y = X^{\mathrm{T}}X\hat{\beta}$$

or $\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$ assuming $X^{\mathrm{T}}X$ is invertible. Revisiting the terminology in the above definitions sections gives the following table:

Least Squares Estimator:	$\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$
Fitted Values:	$\hat{Y} = X\hat{\beta} = X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$
Residuals:	$r = Y - \hat{Y} = (I_n - X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}})Y$

In the case that $n > p$ and that the columns of X are linearly independent, the matrix $P_X := X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$ is a rank $p+1$ projection matrix. Similarly, $I_n - P_X$ is the complementary rank $n{-}p{-}1$ projection matrix. Intuitively, this implies that the fitted values are the projection on the observed values onto a p-dimensional subspace while the residuals arise from a projection onto the orthogonal subspace. As a result, it can be shown that $\mathrm{cov}\left(\hat{Y}, r\right) = 0$.

Now that we have an explicit expression for the least squares estimator $\hat{\beta}$, we can show that it is unbiased.

$$\mathrm{E}\hat{\beta} = \mathrm{E}\left((X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y\right) = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}\mathrm{E}Y = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X\beta = \beta.$$

Following that, we can compute its variance.

$$\begin{aligned}
\mathrm{Var}\left(\hat{\beta}\right) &= \mathrm{E}\left((\hat{\beta}-\beta)(\hat{\beta}-\beta)^{\mathrm{T}}\right)\\
&= \mathrm{E}\left(\hat{\beta}\hat{\beta}^{\mathrm{T}}\right) - \beta\beta^{\mathrm{T}}\\
&= \mathrm{E}\left((X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y((X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y)^{\mathrm{T}}\right) - \beta\beta^{\mathrm{T}}\\
&= (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}\mathrm{E}\left(YY^{\mathrm{T}}\right)X(X^{\mathrm{T}}X)^{-1} - \beta\beta^{\mathrm{T}}\\
&= (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}(\sigma^2 I_n + X\beta\beta^{\mathrm{T}}X^{\mathrm{T}})X(X^{\mathrm{T}}X)^{-1} - \beta\beta^{\mathrm{T}}\\
&= \sigma^2(X^{\mathrm{T}}X)^{-1}.
\end{aligned}$$

Thus far, we have only assumed that ε is a random vector with iid entries with mean zero and variance σ^2. If in addition, we assumed that ε has a *normal* or *Gaussian* distribution, then

$$\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I_n\right), \quad Y \sim \mathcal{N}\left(X\beta, \sigma^2 I_n\right), \text{ and } \hat{\beta} \sim \mathcal{N}\left(\beta, \sigma^2(X^{\mathrm{T}}X)^{-1}\right).$$

Furthermore, with a little work, one can show that for the fitted values and residuals also have normal distributions in this setting:

$$\hat{Y} \sim \mathcal{N}\left(X\hat{\beta}, \sigma^2 P_X\right), \quad \text{and} \quad r \sim \mathcal{N}\left(0, \sigma^2(I_n - P_X)\right).$$

Notice that the two above covariance matrices are not generally of full rank. This assumption that the errors follow a normal distribution is a very common assumption to make in practice.

1.2.2 Maximum likelihood estimate under normality

In the previous section, the OLS estimator is derived by minimizing the sum of the squared errors. Now, given the additional assumption that the errors have a normal distribution, we can compute an alternative estimator for β: the maximum likelihood estimate (MLE). We can also use this to simultaneously compute the MLE for σ^2.

From above we have that $Y \sim \mathcal{N}\left(X\beta, \sigma^2 I_n\right)$, and hence the likelihood is

$$L(\beta, \sigma^2; X, Y) = (2\pi\sigma^2)^{-n/2}\exp\left(-\frac{1}{2\sigma^2}(Y - X\beta)^{\mathrm{T}}(Y - X\beta)\right).$$

The log likelihood is then

$$\ell(\beta, \sigma^2; X, Y) = \log L(\beta, \sigma^2; X, Y) = \\ = -\frac{n}{2}\log 2\pi - \frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(Y - X\beta)^{\mathrm{T}}(Y - X\beta).$$

This implies that the MLE for β comes from solving

$$0 = \frac{\partial \ell}{\partial \beta} = \frac{\partial}{\partial \beta}(Y - X\beta)^{\mathrm{T}}(Y - X\beta),$$

which is solved by the OLS estimator from above. Hence, the MLE under normality is the least squares estimator.

For the variance term σ^2, the MLE is similarly found by solving

$$0 = \frac{\partial \ell}{\partial \sigma^2} = -\frac{n}{2}(\sigma^2)^{-1} + \frac{(\sigma^2)^{-2}}{2}(Y - X\beta)^{\mathrm{T}}(Y - X\beta).$$

This occurs for $\hat{\sigma}^2 = n^{-1}(Y - X\hat{\beta})^{\mathrm{T}}(Y - X\hat{\beta})$, which is just the average sum of squares of the residuals: $\hat{\sigma}^2 = n^{-1}\sum_{i=1}^{n} r_i^2$. However, this is a biased estimator of the variance as the residuals are not independent and have a degenerate covariance matrix of rank $n - p - 1$. Intuitively, this implies that the sum of squared residuals has $n - p - 1$ degrees of freedom resulting in

$$\frac{SS_{\text{res}}}{\sigma^2} = \frac{1}{\sigma^2}\sum_{i=1}^{n} r_i^2 \sim \chi^2\left(n - p - 1\right)$$

and the unbiased estimator of σ^2 being $SS_{\text{res}}/(n - p - 1) = \hat{\sigma}^2(n/(n - p - 1))$.
For a more precise explanation of where this comes from, see Cochran's Theorem which is beyond the scope of this course.

1.2.2.1 Chocolate-Nobel Data

Running a regression in R on the chocolate consumption vs Nobel prize data from above results in a fitted model

$$(\text{Nobel Prizes}) = -0.991 + 1.3545(\text{Chocolate})$$

This indicates that a 1 kg increase in chocolate consumption per capita corresponds to an expected increase in 1.35 Nobel prizes per 10 million people.

```r
# Read in Table
dat = read.table("data/chocoTable.r");
# Construct a linear model
lmDat = lm( nobel ~ choco, dat=dat );
# print summary of model
summary(lmDat);
```

```
Call:
lm(formula = nobel ~ choco, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max 
-6.6876 -1.6504 -0.5288  0.1484 11.3922 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)  -0.9910     2.1327  -0.465  0.64932   
choco         1.3545     0.3446   3.931  0.00151 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.666 on 14 degrees of freedom
Multiple R-squared:  0.5246,    Adjusted R-squared:  0.4907 
F-statistic: 15.45 on 1 and 14 DF,  p-value: 0.001508
```

1.2.3 Proof of the Gauss-Markov Theorem

Proof. Any linear estimator can be written as AY for some non-random matrix $A \in \mathbb{R}^{(p+1)\times n}$. We can in turn write $A = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} + D$ for some matrix $D \in \mathbb{R}^{(p+1)\times n}$. Then, as

$$\begin{aligned}
\mathrm{E}\,(AY) &= AX\beta \\
&= \left[(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} + D\right] X\beta \\
&= \beta + DX\beta,
\end{aligned}$$

the unbiased condition implies that $DX\beta = 0$ for any $\beta \in \mathbb{R}^{p+1}$ and hence that $DX = 0$.

Next, we compute the variance of the arbitrary linear unbiased estimator to get

$$\begin{aligned}
\mathrm{Var}\,(AY) &= A\mathrm{Var}\,(Y)\,A^{\mathrm{T}} \\
&= \sigma^2 \left[(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} + D\right] \left[(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} + D\right]^{\mathrm{T}} \\
&= \sigma^2 \left[(X^{\mathrm{T}}X)^{-1} + (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}D^{\mathrm{T}} + DX(X^{\mathrm{T}}X)^{-1} + DD^{\mathrm{T}}\right] \\
&= \sigma^2 \left[(X^{\mathrm{T}}X)^{-1} + DD^{\mathrm{T}}\right].
\end{aligned}$$

Hence, to minimize the variance, we must minimize DD^{T} as DD^{T} is necessarily a positive semi-definite matrix. This is achieved by setting $D = 0$ and arriving at $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$ having minimial variance. □

Remark 1.1. Note that DD^{T} is positive semi-definite for any choice of D as for any $w \in \mathbb{R}^{p+1}$, we have

$$w^{\mathrm{T}}(DD^{\mathrm{T}})w = \left(D^{\mathrm{T}}w\right)^{\mathrm{T}}(Dw) = \|Dw\|_2 \geq 0.$$

Remark 1.2. While $\hat{\beta}$ has minimal variance over all unbiased estimators, we can lessen the variance further if we allow for biased estimators. This is considered in many more advanced regression methods such as ridge regression and lasso.

1.3 Hypothesis Testing

1.3.1 Goodness of fit

We now have a model for our data, and in some sense, this model is optimal as it minimizes the squared errors. However, even being optimal, we are still interested in knowing whether or not this is a good model for our data. This is a question of *goodness of fit*.

The first question to ask is, do any of the regressors provide information about the response in the linear model framework? This can be written mathematically as

$$H_0 : \beta_1 = ... = \beta_p = 0, \qquad H_1 : \exists i \geq 1 \; s.t. \; \beta_i \neq 0, \tag{1.3}$$

which is asking is there at least one β_i that we can claim is non-zero and hence implies that the regressor x_i has some nontrivial influence over y.

To test this hypothesis, we revisit the explained and residual sums of squares introduced in the Definitions section. Specifically, we already have that $SS_{\text{res}}/\sigma^2 \sim \chi^2\left(n-p-1\right)$ from above. Similarly, $SS_{\text{exp}}/\sigma^2 \sim \chi^2\left(p\right)$ under the null hypothesis where $\beta_1 = ... = \beta_p = 0$, and hence any variation in those terms should be pure noise. Lastly, it can be demonstrated that SS_{res} and SS_{exp} are independent random variables, which intuitively follows from the orthogonality of the fitted values and the errors. Once again, this can be made precise via Cochran's Theorem.

The usual test statistic for the hypothesis in Equation 1.3 is

$$\frac{SS_{\text{exp}}/p}{SS_{\text{res}}/(n-p-1)} \sim F\left(p, n-p-1\right),$$

which leads to an *F test*. If the test statistic is large, then the explained variation is larger than the noise resulting in a small p-value and a rejection of the null hypothesis.

1.3.1.1 F test on Chocolate-Nobel data

From the final line of the R output from the `summary()` command, we have a test statistic value of 15.45 with degrees of freedom 1 and 14. This results in a very small p-value of 0.001508.

```r
summary(lmDat)
```

```
Call:
lm(formula = nobel ~ choco, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max 
-6.6876 -1.6504 -0.5288  0.1484 11.3922 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)  -0.9910     2.1327  -0.465  0.64932   
choco         1.3545     0.3446   3.931  0.00151 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.666 on 14 degrees of freedom
Multiple R-squared:  0.5246,	Adjusted R-squared:  0.4907 
F-statistic: 15.45 on 1 and 14 DF,  p-value: 0.001508
```

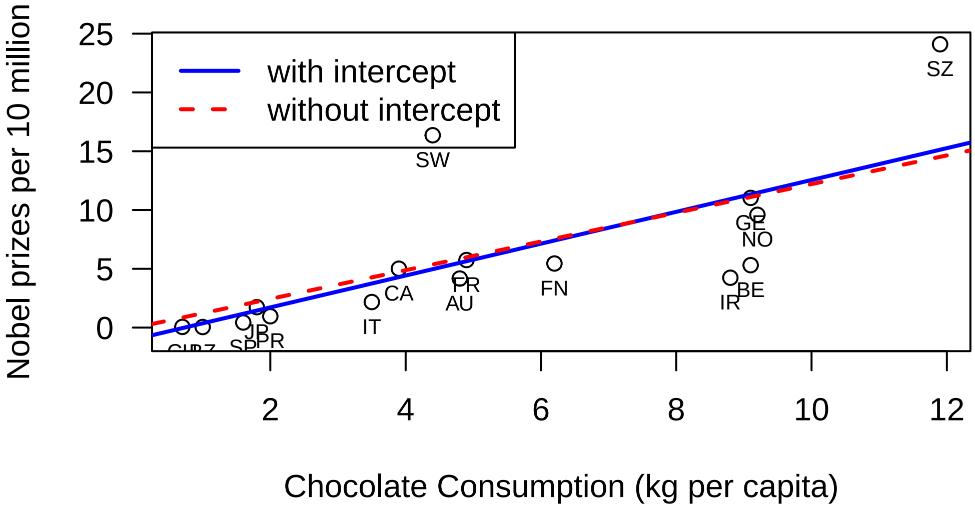
If you were to run the regression in R without the intercept term, which is fixing $\beta_0 = 0$, then the result is $\hat{\beta}_1 = 1.22$, a value for the test statistic for the F test of 44.24, now with degrees of freedom 1 and 15, and an even smaller p-value of 7.7×10^{-6}. Typically, regression models always include an intercept term. However, there are some situations where we wish the enforce that an input of zero returns an output of zero.

```r
# Construct a linear model
lmDat0 = lm( nobel ~ choco - 1, dat=dat );
# print summary of model
summary(lmDat0);
```

```
Call:
lm(formula = nobel ~ choco - 1, data = dat)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-6.4991 -1.7891 -1.3237 -0.1955 10.9910

Coefficients:
      Estimate Std. Error t value Pr(>|t|)
choco   1.2205     0.1835   6.651 7.73e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.543 on 15 degrees of freedom
Multiple R-squared:  0.7468,    Adjusted R-squared:  0.7299
F-statistic: 44.24 on 1 and 15 DF,  p-value: 7.725e-06
```

```
# Plot data
plot(
  dat$choco, dat$nobel,
  xlab="Chocolate Consumption (kg per capita)",
  ylab="Nobel prizes per 10 million",
  las=1,#xlim=c(0,max(dat$choco)+1),
  ylim=c(-1,max(dat$nobel))
);
# Label data
text(
  x=dat$choco, y=dat$nobel,
  labels=dat$abbrev,
  pos=1,cex=0.7
)

# plot regression line
abline(lmDat,col='blue',lwd=2,lty=1)
abline(lmDat0,col='red',lwd=2,lty=2)
# Add a legend
legend(
  "topleft",legend = c("with intercept","without intercept"),
  col = c("blue","red"),lwd=2,lty=1:2
)
```



1.3.2 Regression coefficients

Given that the previous F test results in a significant p-value, the subsequent question is to ask which of the p regressors are significant? Hence, we have the following hypotheses for $j = 0, 1, \ldots, p$.

$$H_{0,j} : \beta_j = 0 \qquad H_{1,j} : \beta_j \neq 0.$$

Each individual $\hat{\beta}_j \sim \mathcal{N}\left(\beta_j, \sigma^2 (X^{\mathrm{T}}X)^{-1}_{j,j}\right)$ where $(X^{\mathrm{T}}X)^{-1}_{j,j}$ is the jth entry in the diagonal of $(X^{\mathrm{T}}X)^{-1}_{j,j}$.

Remark 1.3. We will index the entries of the matrix from $0, 1, \ldots, p$ to conform with the indexing of the β's. Note that this is a $(p+1) \times (p+1)$ matrix.

Thus, under the null hypothesis that $\beta_j = 0$, we have that

$$\hat{\beta}_j / \sqrt{\sigma^2 (X^{\mathrm{T}}X)^{-1}_{j,j}} \sim \mathcal{N}(0, 1).$$

However, we cannot perform a z test as σ^2 is unknown. To rectify this, the unbiased estimator for σ^2 is used in its place resulting in

$$\frac{\hat{\beta}_j}{\sqrt{(X^{\mathrm{T}}X)^{-1}_{j,j} SS_{\text{res}}/(n-p-1)}} \sim t(n-p-1),$$

and a t test can be performed. If the value of the test statistic is large, then there may be sufficient evidence to reject the null that $\beta_j = 0$. The denominator is often referred to as the *standard error*. To simplify future formulae, this will be denoted as $\mathrm{se}(\beta_j)$.

It is worth noting that this test looks for significant influence of the jth regressor on the response given all of the other regressors. Hence, it quantifies the marginal as opposed to the absolute effect of that variable on the model. These ideas will be investigated further when discussing variable selection later in this book. However, as a quick word of caution, when p hypothesis tests are performed, the analyst needs to consider multiple testing corrections.

1.3.2.1 t test on Chocolate-Nobel data

The R commands `lm()` and `summary()` will return a table of regression coefficients, t test statistics and p-values associated with each coefficient. For the Chocolate-Nobel prize data, the table looks like

	Estimate	Std. Error	t value	Pr($> \lvert t \rvert$)
(Intercept)	-0.9910	2.1327	-0.465	0.64932
choco	1.3545	0.3446	3.931	0.00151

```r
summary(lmDat)
```

```
Call:
lm(formula = nobel ~ choco, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max 
-6.6876 -1.6504 -0.5288  0.1484 11.3922 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)  -0.9910     2.1327  -0.465  0.64932   
choco         1.3545     0.3446   3.931  0.00151 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.666 on 14 degrees of freedom
Multiple R-squared:  0.5246,	Adjusted R-squared:  0.4907 
F-statistic: 15.45 on 1 and 14 DF,  p-value: 0.001508
```

1.3.3 Partial F-test

In the previous two sections, we first tested as to whether or not there exists at least one β_j, $j = 1, \dots, p$, that is non-zero. Then, we tested whether or not a specific β_j is non-zero. The next logical question is whether or not some collection of β_j's of size strictly between 1 and p has a non-zero element. That is, for a fixed q,

$$H_0 : \beta_{p-q+1} = \dots = \beta_p = 0 \qquad H_1 : \exists i \geq p - q + 1 \; s.t. \; \beta_i \neq 0. \tag{1.4}$$

Here, we are comparing two different models, which are the partial and full models, respectively,

$$Y = \beta_{0:p-q}X + \varepsilon, \quad \text{and} \quad Y = \beta_{0:p-q}X + \beta_{p-q+1:p}X + \varepsilon,$$

and want to know whether the final q regressors add any significant explanation to our model given the other $p - q$. For the above notation,

$$\beta_{i:j} = (0, \dots, 0, \beta_i, \beta_{i+1}, \dots, \beta_j, 0, \dots, 0)^{\mathrm{T}}.$$

To run the hypothesis test in Equation 1.4, we would have to compute the least squares estimator in the partial model, $\hat{\beta}_{1:p-q}$, and the standard least squares estimator in the full model, $\hat{\beta}$. Then, we will have to compute the additional explained sum of squares gained from adding the q extra regressors to our model, which is

$$SS_{\text{exp}}(\beta_{p-q+1:p}|\beta_{1:p-q}) = SS_{\text{exp}}(\beta) - SS_{\text{exp}}(\beta_{1:p-q}),$$

the explained sum of squares from the full model minus the explained sum of squares from the partial model.

Similarly to the full F-test from above, we have under the null hypothesis that $SS_{\text{exp}}(\beta_{p-q+1:p}|\beta_{1:p-q})/\sigma^2 \sim \chi^2(q)$. Hence,

$$\frac{SS_{\text{exp}}(\beta_{p-q+1:p}|\beta_{1:p-q})/q}{SS_{\text{res}}/(n-p-1)} \sim F(q, n-p-1),$$

so if this test statistic is large, then we have evidence to suggestion that at least one of the additional q regressors adds some explanatory power to our model.

1.4 Interval Estimators

1.4.1 Confidence Intervals

Confidence intervals play a complementary role with hypothesis testing. From the development of the above test for an individual β_j, we have that

$$\frac{\hat{\beta}_j - \beta_j}{\text{se}(\beta_j)} \sim t(n-p-1),$$

Hence, a $1-\alpha$ confidence interval for the parameter β_j is

$$\hat{\beta}_j - t_{\alpha/2,n-p-1}\mathrm{se}(\beta_j) \le \beta \le \hat{\beta}_j + t_{\alpha/2,n-p-1}\mathrm{se}(\beta_j)$$

where $t_{\alpha/2,n-p-1} \in \mathbb{R}^+$ is such that $\mathrm{P}\left(T \le t_{\alpha/2,n-p-1}\right) = \alpha/2$ when $T \sim t\left(n-p-1\right)$.

While the above can be used to produce a confidence interval for each individual parameter, combining these intervals will not result in a $1-\alpha$ confidence set for the entire parameter vector. To construct such a confidence region, a little more care is required. Also, we will construct a confidence set for the entire vector $(\beta_0, \beta_1, \ldots, \beta_p)$, which results in $p+1$ degrees of freedom in what follows. As $\hat{\beta} \sim \mathcal{N}\left(\beta, \sigma^2(X^{\mathrm{T}}X)^{-1}\right)$ we have that

$$\sigma^{-2}(\hat{\beta}-\beta)^{\mathrm{T}} X^{\mathrm{T}}X(\hat{\beta}-\beta) \sim \chi^2\left(p+1\right).$$

From before, we have that $SS_{\mathrm{res}}/\sigma^2 \sim \chi^2\left(n-p-1\right)$. Hence

$$\frac{(\hat{\beta}-\beta)^{\mathrm{T}} X^{\mathrm{T}}X(\hat{\beta}-\beta)/(p+1)}{SS_{\mathrm{res}}/(n-p-1)} \sim F\left(p+1, n-p-1\right).$$

Thus, a $1-\alpha$ confidence ellipsoid can be constructed as

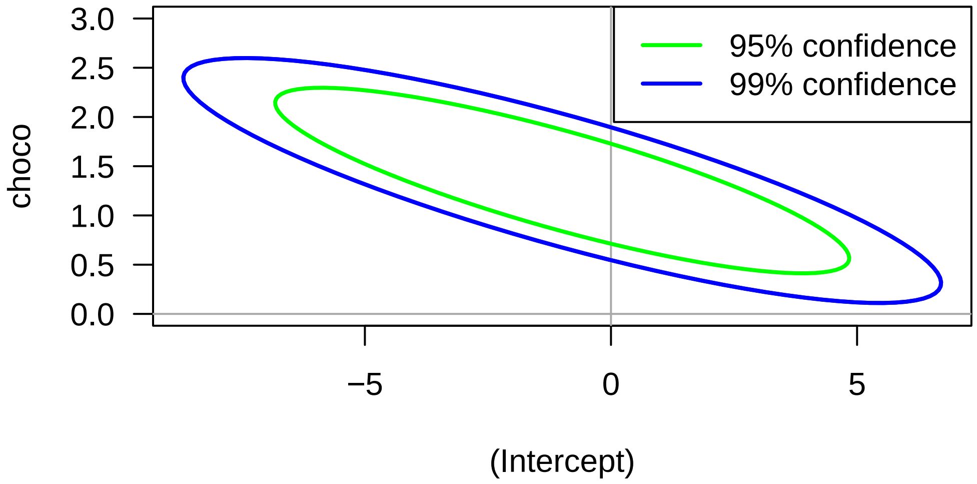
$$\frac{(\hat{\beta}-\beta)^{\mathrm{T}} X^{\mathrm{T}}X(\hat{\beta}-\beta)/(p+1)}{SS_{\mathrm{res}}/(n-p-1)} \le F_{\alpha,p+1,n-p-1}.$$

A 95% and a 99% confidence ellipsoid for the Chocolate-Nobel prize data is displayed in the code below. Notice that both ellipses contain $\hat{\beta}_0 = 0$ which had a t statistic p-value of 0.649. Meanwhile neither contain $\hat{\beta}_1 = 0$ whose p-value was the very significant 0.0015. The confidence ellipses were plotted with help from the R library `ellipse`.

```
library(ellipse)
```

```

Attaching package: 'ellipse'


The following object is masked from 'package:graphics':

    pairs
```

```
plot(
  ellipse(lmDat,level=.99),type='l',
  ylim=c(0,3),col='blue',lwd=2,las=1
);
abline(h=0,v=0,col='darkgray');
lines(ellipse(lmDat,level=.99),type='l',col='blue',lwd=2);
lines(ellipse(lmDat,level=.95),type='l',col='green',lwd=2);
legend(
  "topright",legend = c("95% confidence","99% confidence"),
  col = c("green","blue"),lwd = 2
)
```



1.4.2 Prediction Intervals for an expected observation

Given the least squares model, the analyst may be interested in estimating the expected value of Y have some specific input $x = (1, x_1, \dots, x_p)$. Our new random variable is $\hat{Y}_0 = \hat{\beta} \cdot X$ where X is fixed and $\hat{\beta}$ is random. Of course, the expected value is just

$$\mathrm{E}\left(\hat{Y}_0 \middle| X = x\right) = \mathrm{E}\hat{\beta} \cdot x = \beta_0 + \sum_{i=1}^{p} \beta_i x_i.$$

To find a $1-\alpha$ interval estimate for $\hat{Y}_0$ at $X=x$, recall once again that $\hat{\beta} \sim \mathcal{N}\left(\beta, \sigma^2(X^{\mathrm{T}}X)^{-1}\right)$. Thus,

$$\hat{Y}_0 | (X=x) \sim \mathcal{N}\left(\beta \cdot x, \sigma^2 x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x\right).$$

Hence,

$$\frac{\hat{\beta} \cdot x - \mathrm{E}\left(\hat{Y}_0 \middle| X=x\right)}{\sqrt{\sigma^2 x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x}} \sim \mathcal{N}\left(0,1\right),$$

and

$$\frac{\hat{\beta} \cdot x - \mathrm{E}\left(\hat{Y}_0 \middle| X=x\right)}{\sqrt{(SS_{\mathrm{res}}/(n-p-1))x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x}} \sim t\left(n-p-1\right),$$

which results in the following $1-\alpha$ confidence interval:

$$\begin{aligned}\hat{\beta} \cdot x - t_{\alpha/2,n-p-1}\sqrt{\frac{SS_{\mathrm{res}}}{n-p-1}x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x} \le \\ \le \mathrm{E}\left(\hat{Y}_0 \middle| X=x\right) = \beta \cdot x \le \\ \le \hat{\beta} \cdot x + t_{\alpha/2,n-p-1}\sqrt{\frac{SS_{\mathrm{res}}}{n-p-1}x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x}.\end{aligned}$$

1.4.3 Prediction Intervals for a new observation

In the previous subsection, we asked for a confidence interval for the expected value of the response given a new vector of regressors, which was a confidence interval for $\mathrm{E}\left(\hat{Y}_0 \middle| X=x\right) = \beta \cdot x$ based on $\hat{\beta} \cdot x$. Now, we want to determine a confidence interval for the future response given a vector of regressors. That is, we want an interval for $Y_0 = \beta \cdot x + \varepsilon_0 \sim \mathcal{N}\left(\beta \cdot x, \sigma^2\right)$, but, as usual, β unknown.
To circumvent this, note that

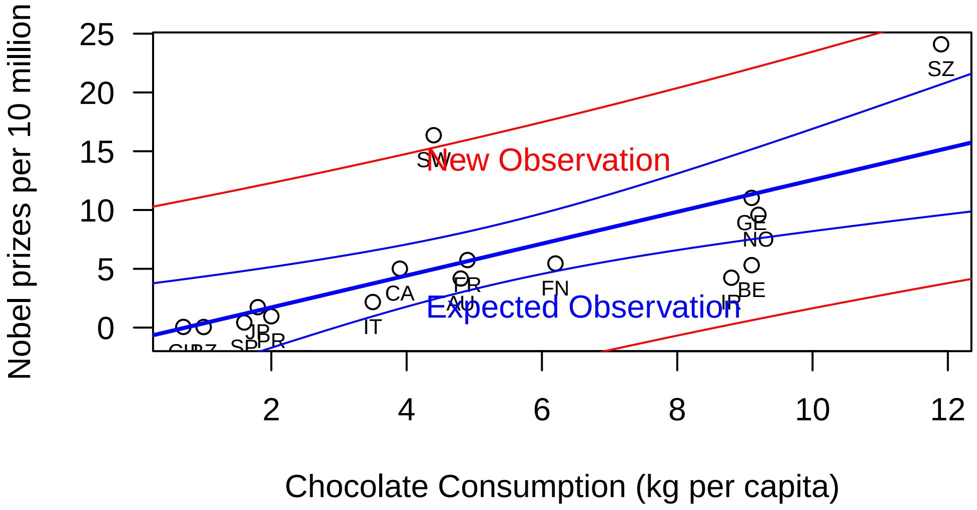
$$Y_0 - \hat{Y}_0 = (\beta \cdot x + \varepsilon_0) - \hat{\beta} \cdot x \sim \mathcal{N}\left(0, \sigma^2(1 + x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x)\right),$$

because the variances of ε_0 and $\hat{\beta} \cdot x$ sum as these are independent random variables. Hence, applying the usual rearrangement of terms and replacement of σ^2 with $SS_{\mathrm{res}}/(n-p-1)$ results in

$$\begin{aligned}\hat{\beta} \cdot x - t_{\alpha/2,n-p-1}\sqrt{\frac{(1 + x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x)SS_{\mathrm{res}}}{n-p-1}} \le Y_0 \le \\ \le \hat{\beta} \cdot x + t_{\alpha/2,n-p-1}\sqrt{\frac{(1 + x^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}x)SS_{\mathrm{res}}}{n-p-1}}.\end{aligned}$$

To demonstrate these prediction intervals, we once again consider the Chocolate-Nobel prize data for both the expected mean and for a new observation.

```r
# Plot data
plot(
  dat$choco, dat$nobel,
  xlab="Chocolate Consumption (kg per capita)",
  ylab="Nobel prizes per 10 million",
  las=1,#xlim=c(0,max(dat$choco)+1),
  ylim=c(-1,max(dat$nobel))
);
# Label data
text(
  x=dat$choco, y=dat$nobel,
  labels=dat$abbrev,
  pos=1,cex=0.7
)
# plot regression line
abline(lmDat,col='blue',lwd=2)

# plot 95% confidence and prediction intervals
tt = seq(0,13,0.1)
prDat  = predict(
  lmDat,newdata=data.frame(choco=tt),
  interval='confidence',level=0.95
)
prDat2 = predict(
  lmDat,newdata=data.frame(choco=tt),
  interval='prediction',level=0.95
)
lines(tt,prDat[,2],col='blue')
lines(tt,prDat[,3],col='blue')
lines(tt,prDat2[,2],col='red')
lines(tt,prDat2[,3],col='red')
text(
  c(4,4),c(1.5,14),
  labels=c("Expected Observation","New Observation"),
  col=c("blue","red"),pos=4
)
```



1.5 Indicator Variables and ANOVA

1.5.1 Indicator variables

Thus far, we have considered models of the form

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \varepsilon$$

where the regressors $x_1, \ldots, x_p \in \mathbb{R}$ can take on any real value. However, very often in practice, we have regressors that take on categorical values. For example, male vs female, employed vs unemployed, treatment vs placebo, Edmonton vs Calgary, etc. When there is a binary choice as in these examples, we can choose one category to correspond to zero and the other category to correspond to one.

As an example, consider

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \tag{1.5}$$

where $x_1 \in \mathbb{R}$ and $x_2 \in \{0, 1\}$. Then, we effectively have two models:

$$\begin{aligned} y &= \beta_0 + \beta_1 x_1 + 0 + \varepsilon \\ y &= \beta_0 + \beta_1 x_1 + \beta_2 + \varepsilon = (\beta_0 + \beta_2) + \beta_2 x_1 + \varepsilon. \end{aligned}$$

What we have is two models with the same slope β_1 but with two different intercepts β_0 and $\beta_0 + \beta_2$, which are two parallel lines.

Remark 1.4. A first thought is to merely split the data and train two separate models. However, we want to use the entire dataset at once specifically to estimate the common slope β_1 with as much accuracy as possible.

While the range of the regressors has changed, we will fit the least squares estimate to the model precisely as before. Now, considering the model in Equation 1.5, assume that we have m samples with $x_2 = 0$ and n samples with $x_2 = 1$. Our design matrix takes on a new form:

$$Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_m \\ Y_{m+1} \\ \vdots \\ Y_{m+n} \end{pmatrix}, \quad X = \begin{pmatrix} 1 & x_1 & 0 \\ \vdots & \vdots & \vdots \\ 1 & x_m & 0 \\ 1 & x_{m+1} & 1 \\ \vdots & \vdots & \vdots \\ 1 & x_{m+n} & 1 \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}, \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_{m+n} \end{pmatrix}.$$

However, the least squares estimate is computed as before as $\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$. Furthermore, we can perform hypothesis tests on the fitted model such as

$$H_0 : \beta_2 = 0 \qquad H_1 : \beta_2 \neq 0,$$

which is equivalently asking whether or not the regression lines have the same intercept.

Models can be expanded to include multiple indicator variables as long as the matrix $X^{\mathrm{T}}X$ is still invertible. For example, let's suppose we want to look at wages in Alberta with respect to age but partitioned for male vs female and for Edmonton vs Calgary. Then, the model would look like

$$(\text{wage}) = \beta_0 + \beta_1(\text{age}) + \beta_2(\text{Is male?}) + \beta_3(\text{Is from Edmonton?}).$$

In the silly case that our data only consisted of men from Calgary and women from Edmonton, then the final regressor is redundant and $X^{\mathrm{T}}X$ will not be invertible. While this extreme case should not occur, it is possible to have an imbalance in the categories, which we will discuss later.

1.5.2 ANOVA

ANOVA, or the Analysis of Variance, is a slightly overloaded term in statistics. We already considered ANOVA tables when comparing nested models in the hypothesis tests. However, ANOVA can also be used in the setting of the so-called One-Way Analysis of Variance. In this case, we want to compare k samples for equality of the means. For example, we take height measurements from randomly selected citizens from different countries and ask whether or not there is significant evidence to reject the claim that all nations have roughly the same height distribution.

The reason for discussing ANOVA in these notes is that it can be written in a linear regression context as follows. Imagine that we have k different groups of observations with sample sizes

n_j, $j = 1, \dots, k$ for each group. Let $y_{i,j}$ be the ith observation from the jth group where $i \in \{1, \dots, n_j\}$ and $j \in \{1, \dots, k\}$. The model is

$$y_{i,j} = \mu_j + \varepsilon_{i,j},$$

which is each observation is just some group mean, μ_j, with the addition of random noise.

From here, one can show that the fitted values as just $\hat{y}_{i,j} = n_j^{-1} \sum_{l=1}^{n_j} y_{l,j}$, which is the jth sample mean. Then, an F-test can be performed similar to what we did above with F tests to test

$$H_0 : \mu_1 = \dots = \mu_k \quad H_1 : \exists j_1 \neq j_2 \ s.t. \ \mu_{j_1} \neq \mu_{j_2}.$$

To reformulate the model to align with our F-test from before, we rewrite it as a linear regression with indicator variables for the regressors

$$y = \beta_0 + \beta_1 x_1 + \dots + \beta_{k-1} x_{k-1} + \varepsilon_{i,j}$$

with $\beta_0 = \mu_k$ and $\beta_j = \mu_j - \mu_k$ for $j = 1, \dots, k-1$. Then, we can test for whether or not there exists at least one $\beta_j \neq 0$ for $j = 1, \dots, k-1$. Here, the degrees of freedom for the explained sum of squares is $k-1$ and the degrees of freedom for the residual sum of squares is $N - (k-1) - 1 = N - k$ with $N = \sum_{j=1}^{k} n_j$.
If all of the n_j are equal, this reduces to $k(n-1)$.

In this case, the vector Y and the design matrix X will take on the form

$$Y = \begin{pmatrix} y_{1,1} \\ y_{2,1} \\ \vdots \\ y_{n_1,1} \\ y_{1,2} \\ \vdots \\ y_{n_k,k} \end{pmatrix}, \quad X = \begin{pmatrix} 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \end{pmatrix}.$$

1.6 Data Example: Exam Grades

To illustrate the topics discussed in this chapter, we will consider a dataset of course grades for a linear regression class. This dataset consists of $n = 56$ undergraduate students who are either in their 3rd year (22 students) or 4th year (33 students) of university studies. The year of study acts as a binary variable as discussed in the previous section. This dataset also has columns corresponding to the students overall mark on written homework assignments, online coding assignments, the midterm exam, and the final exam. In what follows, we will see what variables if any can be used to predict the final exam grade and, furthermore, if there is a difference between the final exam grades for the 3rd year and 4th year students.

```
# Read in Data
exams = read.csv("data/statGrades.csv")

# Tell R that "year" is a categorical variable
exams$year <- as.factor(exams$year)

# Look at the data
head(exams)
```

```
      year written    online  midterm   final
1 4th Year  80.668  83.84127 70.00000 56.4948
2 3rd Year  96.668 100.00000 50.00000 54.0000
3 4th Year  90.000  93.73737 50.00000 43.5024
4 3rd Year  98.000  98.57143 56.66667 62.9964
5 3rd Year  46.000  48.72727 50.00000 43.5024
6 4th Year  75.334  96.34921 93.33333 81.0000
```

When comparing two independent groups, we can use the classic two sample t test using `t.test()` in R. The default is to assume that the variances between the two groups are not equal, which is the so-called Welch's t-test. Otherwise, we can tell the function to treat the variances as equal by setting `var.equal=T`.

```
t.test(
  final ~ year,
  data = exams
)
```

```

    Welch Two Sample t-test

data:  final by year
t = -0.42107, df = 43.775, p-value = 0.6758
alternative hypothesis: true difference in means between group 3rd Year and group 4th Year i
95 percent confidence interval:
 -11.709616   7.662733
```

```
sample estimates:
mean in group 3rd Year mean in group 4th Year 
              65.67292               67.69636 
```

```
t.test(
  final ~ year,
  data = exams,
  var.equal=T
)
```

```

    Two Sample t-test

data:  final by year
t = -0.43015, df = 54, p-value = 0.6688
alternative hypothesis: true difference in means between group 3rd Year and group 4th Year i
95 percent confidence interval:
 -11.454430   7.407547
sample estimates:
mean in group 3rd Year mean in group 4th Year 
              65.67292               67.69636 
```

We can also view a two sample t test as a linear regression of the response variable (final exam grade) on the dummy variable (student year). The regression equation becomes

$$(\text{final exam grade}) = \beta_0 + \beta_1(\text{student year}) + \varepsilon$$

What we can see from the output below is that this model is equivalent to the two sample t test assuming equal (homogeneous) variances among the two samples. In the end, there is no noticable difference between the performance of 3rd and 4th year students.

```
# A two-sample test
md.year = lm(
  final ~ year,
  data = exams
)
summary(md.year)
```

```

Call:
lm(formula = final ~ year, data = exams)
```

```
Residuals:
    Min      1Q  Median      3Q     Max 
-31.700 -14.444   1.318  12.550  36.322 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)    65.673      3.611   18.19   <2e-16 ***
year4th Year    2.023      4.704    0.43    0.669    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 17.32 on 54 degrees of freedom
Multiple R-squared:  0.003415,	Adjusted R-squared:  -0.01504 
F-statistic: 0.185 on 1 and 54 DF,  p-value: 0.6688
```

However, the `lm()` function allows us to fit more complex models than just a two sample comparison. Here, we use the students performance on the written homework, the online coding homework, and the midterm exam as predictors of performance on the final exam. The result below is that the midterm mark is a strongly significant predictor of the final exam mark with an estimated coefficient of 0.953 and that the written assignment mark is a weak but possibly still relevant predictor of the final exam mark.

```
# Fit Linear Model
md.exams = lm(
  final ~ year+written+online+midterm,
  data = exams
)
summary(md.exams)
```

```
Call:
lm(formula = final ~ year + written + online + midterm, data = exams)

Residuals:
     Min       1Q   Median       3Q      Max 
-20.2987  -7.0874  -0.8767   6.4742  31.3874 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -14.3982    12.6807  -1.135   0.2615    
year4th Year   1.5523     3.0561   0.508   0.6137    
```

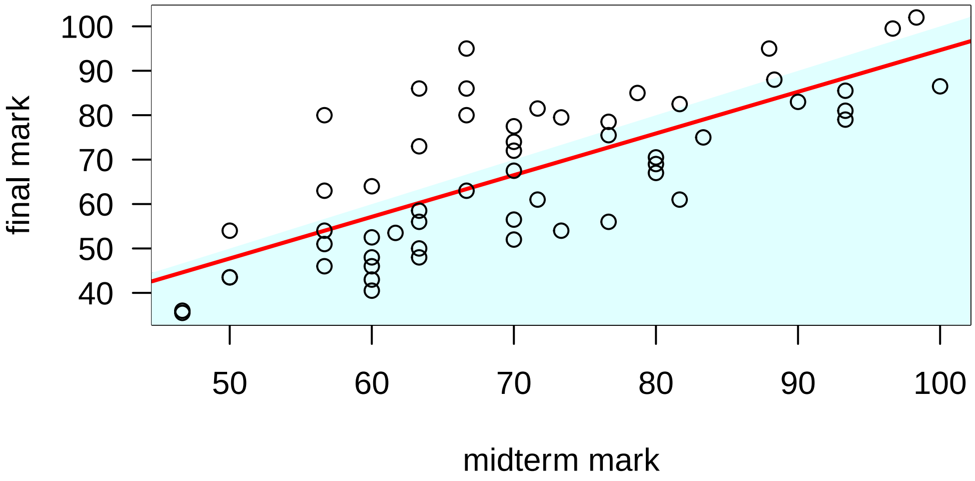
```
written        0.3707     0.1962   1.889   0.0646 .
online        -0.2024     0.2143  -0.944   0.3494
midterm        0.9527     0.1113   8.558 1.97e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 11.24 on 51 degrees of freedom
Multiple R-squared:  0.6032,    Adjusted R-squared:  0.5721
F-statistic: 19.39 on 4 and 51 DF,  p-value: 9.478e-10
```

We can also consider the simpler regression model that only takes the midterm mark into account as a predictor variable. The fitted model has a slope parameter of 0.938 and an intercept not significantly different from zero. This indicates that the student's final exam mark was on average 0.938 × (midterm mark), or simply that students had slightly lower grades on average on the final exam compared to the midterm exam. The cyan coloured region indicates those who had a higher midterm mark than final exam mark.

```
md.exams0 = lm(
  final ~ midterm,
  data = exams
)
summary(md.exams0)
```

```
Call:
lm(formula = final ~ midterm, data = exams)

Residuals:
    Min      1Q  Median      3Q     Max
-16.758  -8.683  -2.942   7.620  31.621

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.8376     8.1986   0.102    0.919
midterm       0.9381     0.1144   8.201 4.68e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 11.58 on 54 degrees of freedom
Multiple R-squared:  0.5547,    Adjusted R-squared:  0.5464
F-statistic: 67.25 on 1 and 54 DF,  p-value: 4.684e-11
```

```
plot(
  exams$midterm,exams$final,las=1,
  xlab="midterm mark",ylab="final mark"
)
polygon(
  c(0,200,200),c(0,200,0),col = 'lightcyan',
  border = NA
)
polygon(
  c(0,0,200),c(0,200,200),col = 'white',
  border = NA
)
abline(md.exams0,col='red',lwd=2)
points(exams$midterm,exams$final)
```



We can now imagine a hypothetical student in their 4th year who got a 95% on the written assignemnts, 85% on the online assignemnts, and an 83% on the midterm exam. Using the `predict()` function in R, we get an expected final exam grade of 84.24%. Going further, a 95% prediction interval for a new value gives a very wide range from 60.3% to 108.17%. This is where the model believes the final exam grade will lie for our hypothetical student.

```
predict(
  md.exams,
  newdata=data.frame(
    year="4th Year",written=95,online=85,midterm=83
  ), interval="prediction",level=0.95
)
```

```
       fit      lwr      upr
1 84.24118 60.30866 108.1737
```

We can use the `anova()` function to compare nested models, which will perform a partial F-test. In this example, we can compare the model that only includes the `year` variable to a model that contains model `year` and `midterm` to a model that contains all four predictor variables.

In the ANOVA table below, we have that comparing model 1 to model 2 results is a very significant p-value indicating that including `midterm` as a predictor results in a large decrease in the residual sum of squares (column 2 in the table below). Secondly, we compare model 2 to model 3 and get a weak but possibly significant p-value of 0.0597. This gives some weak evidence that including `written` and `online` assignment marks may further improve the fit of the model.

Note that the degrees of freedom for the partial F-tests performed can be found in columns 1 and 3 in the ANOVA table. These are the degrees of freedom for the residual sum of seqares (column 1) and the explained sum of squares (column 3).

```
md.exams1 = lm(
  final ~ year+midterm,
  data = exams
)
anova(md.year,md.exams1,md.exams)
```

```
Analysis of Variance Table

Model 1: final ~ year
Model 2: final ~ year + midterm
Model 3: final ~ year + written + online + midterm
  Res.Df     RSS Df Sum of Sq       F    Pr(>F)
1     54 16195.2
2     53  7200.8  1    8994.4 71.1454 3.058e-11 ***
3     51  6447.5  2     753.3  2.9792   0.05974 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

If we incorrectly try to compare non-nested models, we get erroneous results in our ANOVA table.

```
md.exams2 = lm(
  final ~ year+written+online,
  data = exams
)
anova(md.exams1,md.exams2)
```

```
Analysis of Variance Table

Model 1: final ~ year + midterm
Model 2: final ~ year + written + online
  Res.Df     RSS Df Sum of Sq F Pr(>F)
1     53  7200.8
2     52 15706.6  1   -8505.8
```

The general equation for a linear regression is $Y = X\beta + \varepsilon$. To recover the design matrix X, we can use the function `model.matrix()` in R.
Notice that the column corresponding to `year` is now encoded as a binary variable with a 1 indicating 4th year and a 0 indicating 3rd year.

```
model.matrix(md.exams)
```

```
   (Intercept) year4th Year   written    online   midterm
1            1            1  80.66800  83.84127  70.00000
2            1            0  96.66800 100.00000  50.00000
3            1            1  90.00000  93.73737  50.00000
4            1            0  98.00000  98.57143  56.66667
5            1            0  46.00000  48.72727  50.00000
6            1            1  75.33400  96.34921  93.33333
7            1            0  82.66600  89.77778 100.00000
8            1            0  80.33400  96.34921  60.00000
9            1            0  96.66600  96.34921  56.66667
10           1            0  82.66600  87.46609  70.00000
11           1            1  72.66667 100.00000  56.66667
12           1            0  95.33600  98.57143  63.33333
13           1            1  98.00000 100.00000  93.33333
14           1            0  92.00000 100.00000  88.33333
15           1            1  63.33200  74.66667  63.33333
16           1            1  98.00200  98.57143  76.66667
```

```
17           1            1  86.00000  85.73737  60.00000
18           1            1 100.00000  97.77778  73.33333
19           1            1  96.66600  98.57143  60.00000
20           1            1 100.00000  93.10245  81.66667
21           1            1  97.33400  98.57143  46.66667
22           1            0  98.66600 100.00000  98.33333
23           1            1  96.00000 100.00000  80.00000
24           1            1  95.33400  97.77778  60.00000
25           1            0  85.33400  94.53102  66.66667
26           1            1  77.33267  93.10245  81.66667
27           1            1  94.00000  97.14286  87.96667
28           1            1  96.00000 100.00000  70.00000
29           1            1  86.66600  93.14286  73.33333
30           1            1  82.66400  88.92064  70.00000
31           1            0  85.33400  91.55556  83.33333
32           1            0  86.66533  94.34921  96.66667
33           1            1  99.33400 100.00000  63.33333
34           1            0  91.33400  95.95960  70.00000
35           1            0  75.33400  93.73737  76.66667
36           1            0  76.66600  87.06205  80.00000
37           1            1  60.62733  92.69841  71.66667
38           1            1  85.33400  92.34921  66.66667
39           1            1  74.66800  79.79221  60.00000
40           1            1  92.00000  98.00000  78.70000
41           1            1  70.66400  88.34921  56.66667
42           1            1  93.33400 100.00000  71.66667
43           1            0  98.66800  85.44012  63.33333
44           1            1  97.33400 100.00000  56.66667
45           1            0  71.99800  93.77778  63.33333
46           1            1  96.66600  95.77778  61.66667
47           1            1  98.66600 100.00000  90.00000
48           1            1   0.00000   0.00000  93.33333
49           1            1  99.33400 100.00000  66.66667
50           1            0  97.33400  98.57143  60.00000
51           1            1  95.33400 100.00000  76.66667
52           1            0  94.66733 100.00000  70.00000
53           1            0  93.33200  98.57143  63.33333
54           1            0  87.33200  95.77778  80.00000
55           1            1  93.33400  96.00000  66.66667
56           1            0  70.00000  81.77778  46.66667
attr(,"assign")
[1] 0 1 2 3 4
attr(,"contrasts")
attr(,"contrasts")$year
[1] "contr.treatment"
```

2 Model Assumptions and Choices

2.1 Introduction

Thus far, we have considered linear regression given some key assumptions. Namely, for models of the form

$$y = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p + \varepsilon$$

we assume that the noise or errors ε have zero mean, constant finite variance, and are uncorrelated. Furthermore, in order to perform hypothesis tests–F test, t test, partial F-tests–and to construct confidence and prediction intervals as we did in the previous chapter, we further assumes that ε has a normal distribution.

In this chapter, we will consider deviations from these assumptions, which will lead to questions such as

1. What happens if the variance of ε is not constant?
2. What happens if ε has heavier tails than those of a normal distribution?
3. What effect can outliers have on our model?
4. How can we transform our data to correct for some of these deviations?
5. What happens if the true model is not linear?

2.2 Plotting Residuals

In the last chapter, we constructed the residuals as follows. Assume we have a sample of n observations (Y_i, X_i) where $Y_i \in \mathbb{R}$ and $X_i \in \mathbb{R}^p$ with p being the number of regressors and $p < n$. The model, as before, is

$$Y = X\beta + \varepsilon$$

where $Y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n\times(p+1)}$, $\beta \in \mathbb{R}^{p+1}$, and $\varepsilon \in \mathbb{R}^n$. The least squares estimator is $\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$ and the vector of residuals is thus

$$r = (I - P)Y = (I - X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}})Y.$$

We can use the residuals to look for problems in our data with respect to deviations from the assumptions. But first, they should be normalized in some way. As we know from the previous chapter, the covariance of the residuals is $(I - P)\sigma^2$ and that $SS_{\text{res}}/(n-p-1)$ is an unbiased estimator for the unknown variance σ^2. This implies that while the errors ε_i are assumed to be uncorrelated, the residuals are, in fact, correlated. Explicitly,

$$\mathrm{Var}\left(r_i\right) = (1 - P_{i,i})\sigma^2, \text{ and } \mathrm{cov}\left(r_i, r_j\right) = -P_{i,j}\sigma^2 \text{ for } i \neq j$$

where $P_{i,j}$ is the (i, j)th entry of the matrix P.
Hence, a standard normalization technique is to write

$$s_i = \frac{r_i}{\sqrt{(1 - P_{i,i})SS_{\text{res}}/(n-p-1)}},$$

which are denoted as the *studentized residuals*. For a linear model fit in `R` by the function `lm()`, we can extract the residuals with `resid()` and extract the studentized residuals with `rstudent()`.

2.2.1 Plotting Residuals

2.2.1.1 Studentized Residuals

A plot of studentized residuals from a simple linear regression is displayed below. Generally, abnormally large studentized residuals indicate that an observation may be an outlier.

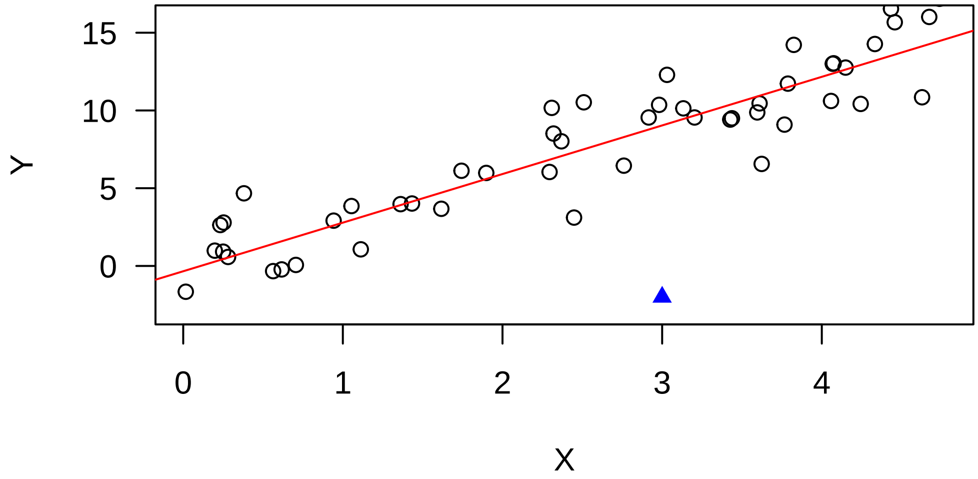
::: {#rem-residuals} There are other types of residuals that can be computed such as standardized residuals, PRESS residuals, and externally studentized residuals. These are also used to look for outliers. :::

```
set.seed(128)
# Simulate some linear regression data
xx = runif(n=50,min=0,max=5)
yy = 3*xx + rnorm(n=50,0,2)
# Add in one outlier
xx <- c(xx, 3)
yy <- c(yy,-2)
# Fit a linear regression
md = lm(yy~xx)
summary(md)
```

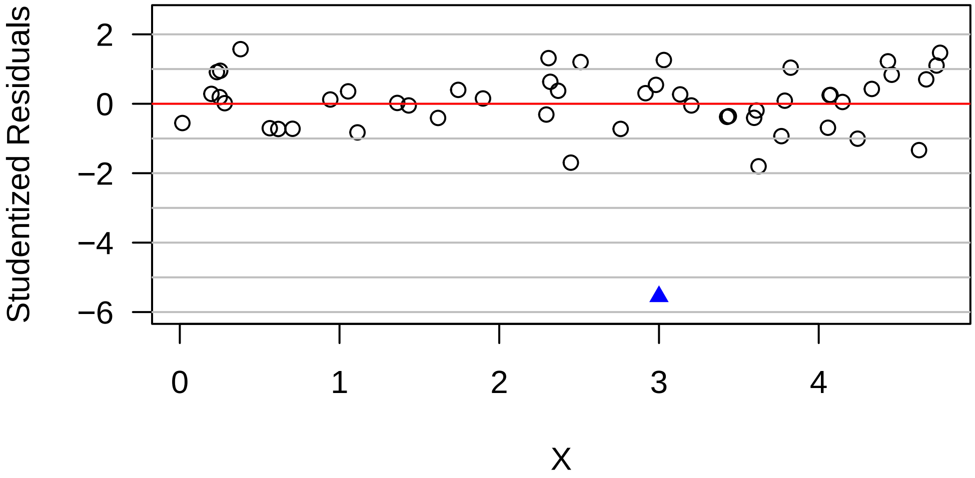
```
Call:
lm(formula = yy ~ xx)
```

```
Residuals:
     Min       1Q   Median       3Q      Max
-11.0377  -1.2028   0.3087   1.4941   3.8238

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.3412     0.7148  -0.477    0.635
xx            3.1263     0.2400  13.028   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.549 on 49 degrees of freedom
Multiple R-squared:  0.776, Adjusted R-squared:  0.7714
F-statistic: 169.7 on 1 and 49 DF,  p-value: < 2.2e-16
```

```
# Plot the data
plot(xx[1:50],yy[1:50],las=1,ylim=c(-3,16),xlab="X",ylab="Y");
points(xx[51],yy[51],pch=17,col='blue')
abline(md,col='red')
```



```
# Plot Residuals
res = rstudent(md)
plot(
  xx[1:50],res[1:50],las=1,ylim=c(-6,2.5),
  xlab="X",ylab="Studentized Residuals"
);
points(xx[51],res[51],pch=17,col='blue')
abline(h=(-10):10,col='gray')
abline(h=0,col='red')
```



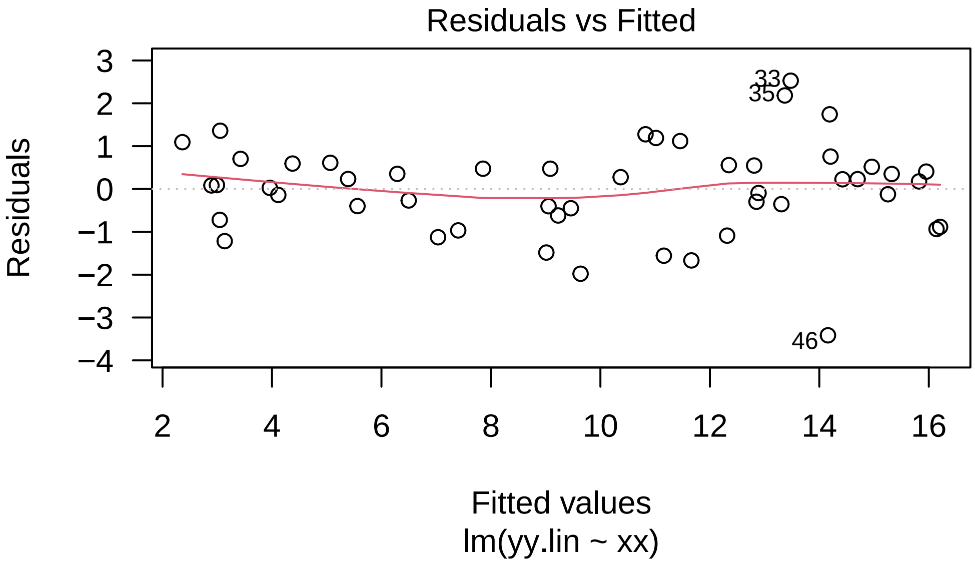
2.2.1.2 Residuals vs Fitted Values

The residuals and the fitted values as necessarily uncorrelated. That is, $\mathrm{cov}\left(r, \hat{Y}\right) = 0$. However, if ε is not normally distributed, then they may not be independent. Plotting the fitted values against the residuals can give useful diagnostic information about your data.

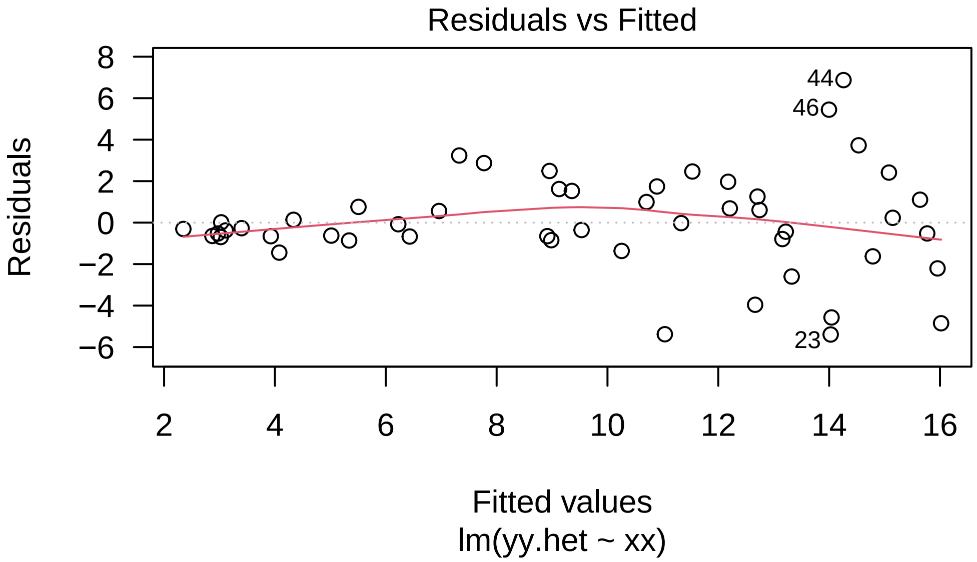
The code below gives four examples of plots of the residuals against the fitted values. The first plot in the top left came from a simple linear regression model where all of the standard assumptions are met. The second plot in the top right came from a simple linear regression but with errors $\varepsilon_i \sim \mathcal{N}\left(0, \sigma_i^2\right)$ where σ_i^2 was increasing in i. Hence, the plot has an expanding look to it. The third plot in the bottom left came from a simple linear regression with the addition of a quadratic term. Fitting a model without the quadratic term still yielded significant test statistics, but failed to account for the nonlinear interaction between x and y. The final plot in the bottom right came from a simple linear regression where the errors were correlated. Specifically, $\text{cov}\left(\varepsilon_i, \varepsilon_j\right) = \min\{x_i, x_j\}$.

Fun fact: This is actually the covariance of Brownian motion.

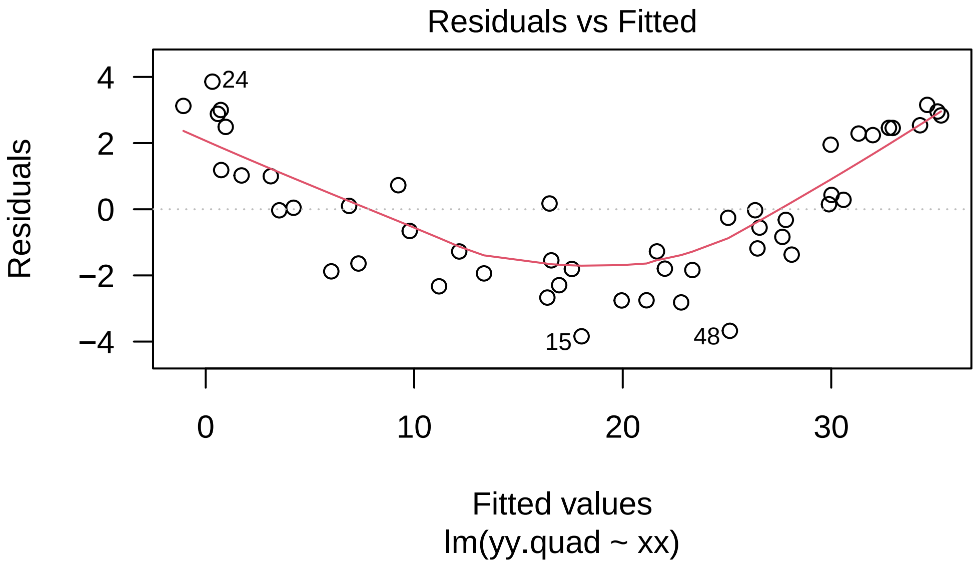
```
# remove the outlier from the last code block
xx <- xx[1:50]
set.seed(256)
# Create a "good" linear regression
yy.lin  = 3*xx + 2 + rnorm(50,0,1);
md.lin  = lm(yy.lin ~ xx)
plot(md.lin,which=1,las=1)
```



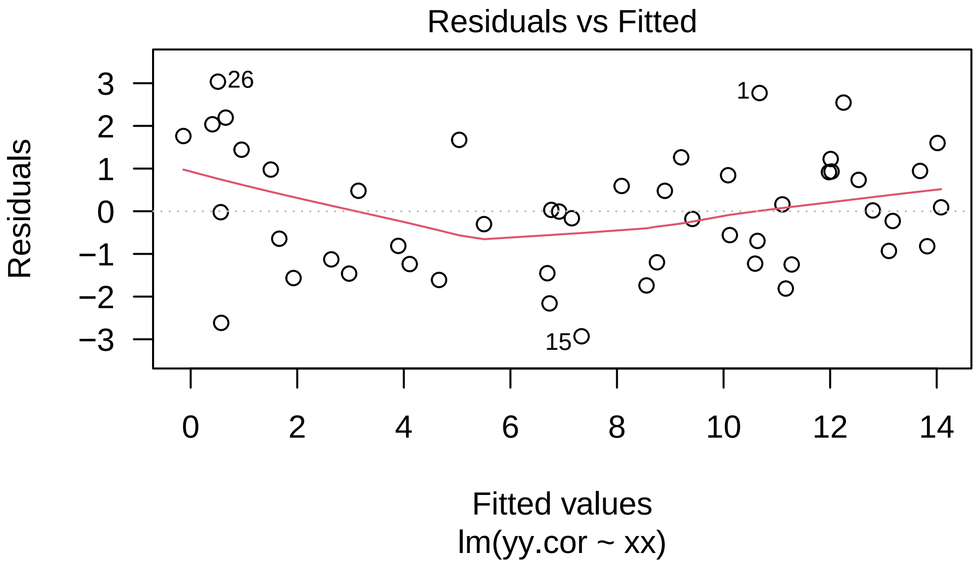
```
# Create a linear regression with increasing variance
yy.het  = 3*xx + 2 + rnorm(50,0,xx);
md.het  = lm(yy.het ~ xx)
plot(md.het,which=1,las=1)
```



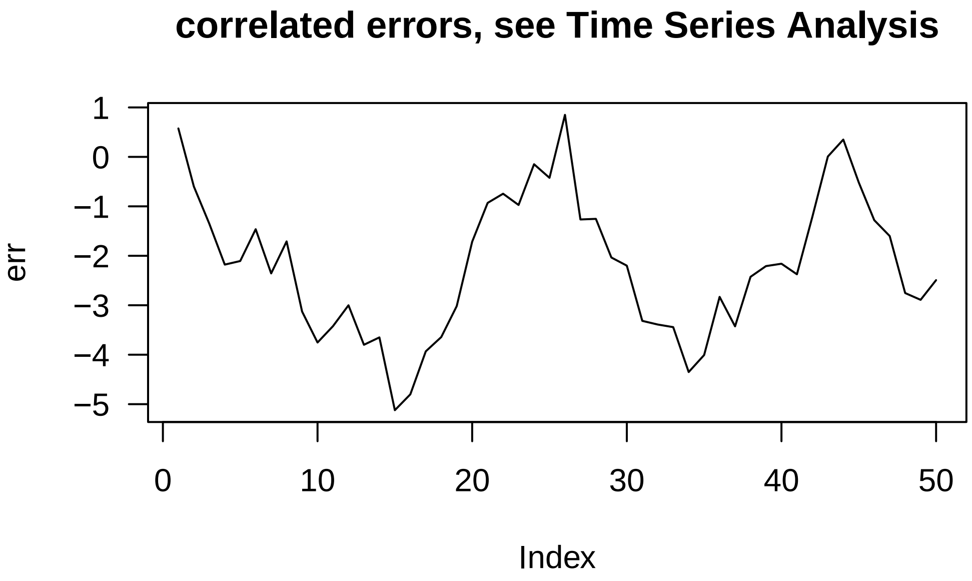
```
# Create a linear regression with quadratic trend
yy.quad = xx^2 + 3*xx + 2 + rnorm(50,0,1);
md.quad = lm(yy.quad ~ xx)
plot(md.quad,which=1,las=1)
```



```
# Create a linear regression with correlated errors
err = rnorm(50,0,1);
err <- cumsum(err);
yy.cor = 3*xx + 2 + err;
md.cor  = lm(yy.cor ~ xx)
plot(md.cor,which=1,las=1)
```



```
plot(
  err,type='l',las=1,
  main="correlated errors, see Time Series Analysis"
)
```



2.2.1.3 Normal Q-Q plots

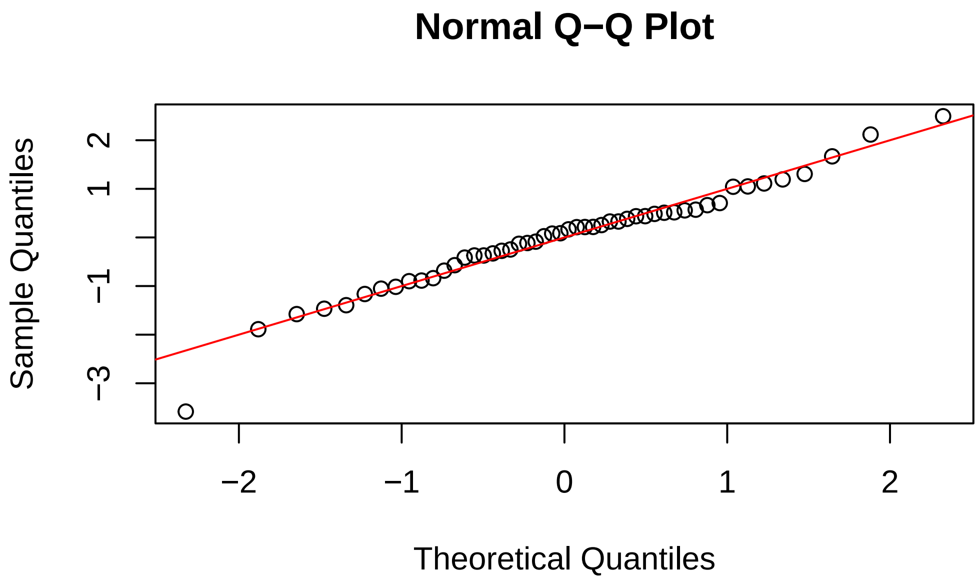
Another type of plot that can offer insight into your data is the so-called Normal Q-Q plot. This tool plots the studentized residuals against the *quantiles* of a standard normal distribution.

Definition 2.1. The quantile function is the inverse of the cumulative distribution function. That is, in the case of the normal distribution, let $Z \sim \mathcal{N}(0,1)$. Then the CDF is $\Phi(z) = \mathrm{P}(Z < z) \in (0,1)$ for $z \in \mathbb{R}$. The quantile function is $\Phi^{-1}(t) \in [-\infty, \infty]$ for $t \in [0,1]$. For more details, see QQ Plot

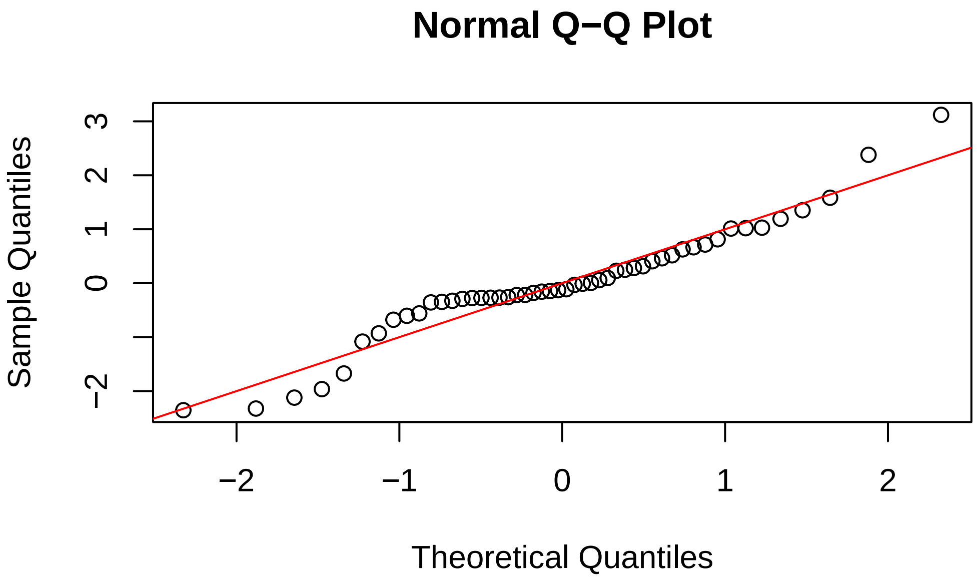
For a normal Q-Q plot, let $s_1, \ldots, s_n$ be the **ordered** studentized residuals, so that $s_1 \leq \ldots \leq s_n$. The theoretical quantiles are denoted $q_1, \ldots, q_n$ where $q_i = \Phi^{-1}(i/(n+1))$. In R, a slightly different formula is used. The figures below compare various normal Q-Q plots. In the first, the errors are normally distributed and the ordered residuals roughly follow the red line. In the second, the heteroskedastic errors cause the black points to deviate from the red line. This is also seen in the third plot, which fits a linear model to quadratic data. The fourth plot considers correlated errors and the QQ-plot has black points very close to the red line.

Remark 2.1. As noted in Montgomery, Peck, & Vining, these are not always easy to interpret and often fail to capture non-normality in the model. However, they seem to be very popular nonetheless.

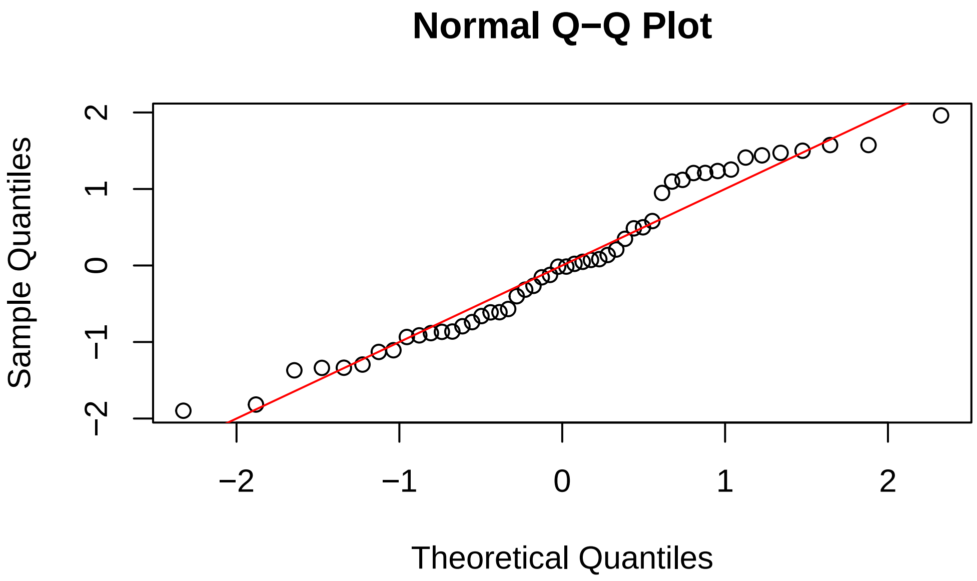
```
# plot the above residuals in QQ-plots
# against the normal distribution
qqnorm(rstudent(md.lin))
abline(0,1,col='red')
```



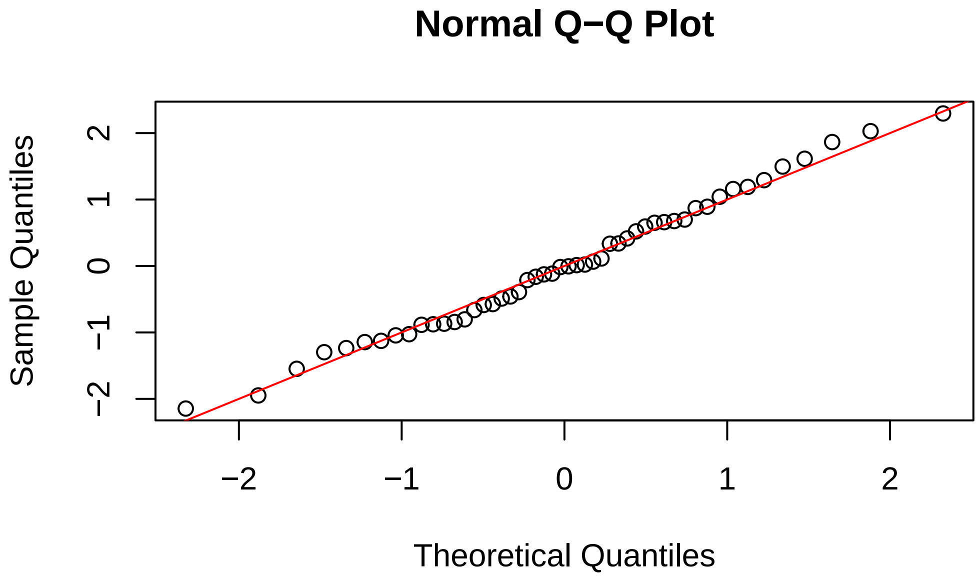
```
qqnorm(rstudent(md.het))
abline(0,1,col='red')
```



```
qqnorm(rstudent(md.quad))
abline(0,1,col='red')
```



```
qqnorm(rstudent(md.cor))
abline(0,1,col='red')
```



2.2.2 Beer Data Example

The following section considers a dataset that tracks a person's blood alcohol content after consuming some number of beers. Two covariates in this dataset are the subjects sex and weight. I got this dataset in 2018 from a statistician I met at ICOTS 2018, because, yes, statisticians trade datasets at conferences.

```
bac = read.csv("data/BACfull.csv")
bac
```

```
    BAC weight    sex beers
1 0.100    132 female     5
2 0.030    128 female     2
3 0.190    110 female     9
4 0.120    192   male     8
5 0.040    172   male     3
6 0.095    250 female     7
7 0.070    125 female     3
```

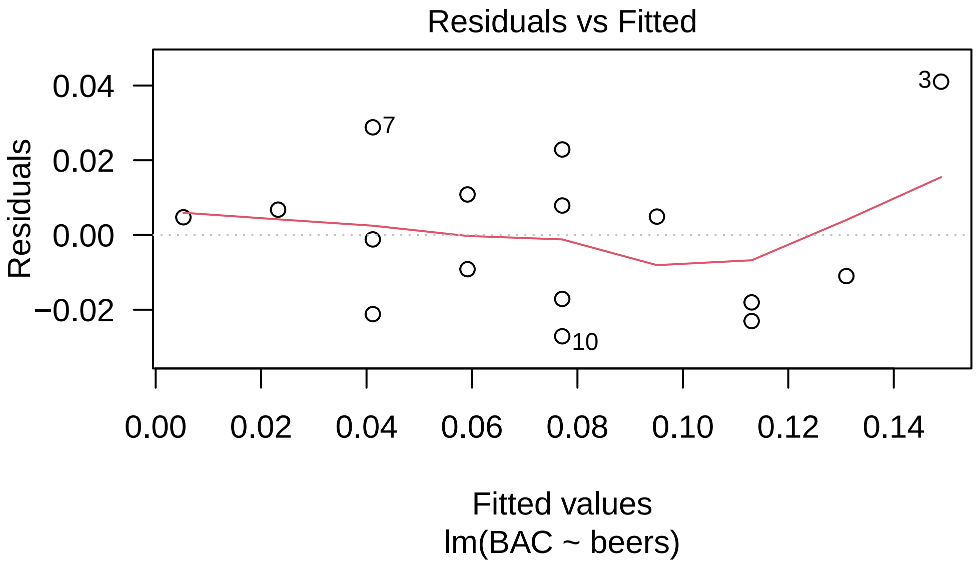
```
8  0.060    175   male     5
9  0.020    175 female     3
10 0.050    275   male     5
11 0.070    130 female     4
12 0.100    168   male     6
13 0.085    128 female     5
14 0.090    246   male     7
15 0.010    164   male     1
16 0.050    175   male     4
```

We can fit a simple regression model that only considers the number of `beers consumed` as the sole predictor for blood alcohol content. The resulting model estimates that each beer raises a subjects blood alcohol content by 0.018.
In the residual vs fitted plot, the subject with the largest residual (person 3) was also the subject with the lowest weight and most beers consumed. Of course, the following simple regression model does not consider `weight` as a predictor variable.

```r
md.bac1 <- lm( BAC~beers, data=bac  )
summary(md.bac1)
```

```
Call:
lm(formula = BAC ~ beers, data = bac)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.027118 -0.017350  0.001773  0.008623  0.041027 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.012701   0.012638  -1.005    0.332    
beers        0.017964   0.002402   7.480 2.97e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.02044 on 14 degrees of freedom
Multiple R-squared:  0.7998,    Adjusted R-squared:  0.7855 
F-statistic: 55.94 on 1 and 14 DF,  p-value: 2.969e-06
```

```r
plot(md.bac1,which=1,las=1)
```



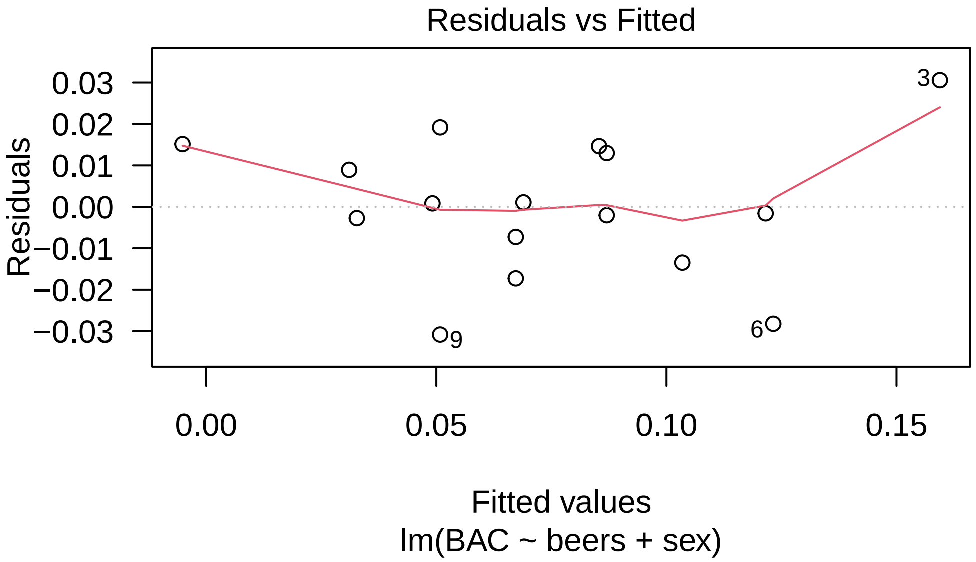
We can include **sex** as a predictor variable which results in the model below. In this case, we see a slight significant in the **sex** variable, which seems to suggest that males have on average a lower blood alcohol content than females by 0.02. The three subjects with the largest (absolute) residuals are subject 3, the female with the lowest overall weight, and subjects 6 and 9, the females with the highest overall weights. Once again, we note that without considering a subject's weight in this regression model, the largest residuals (i.e. those points that most deviate from our fitted model) come from subjects with the largest and smallest weights.

```
md.bac2 <- lm( BAC~beers+sex, data=bac  )
summary(md.bac2)
```

```

Call:
lm(formula = BAC ~ beers + sex, data = bac)

Residuals:
       Min         1Q     Median         3Q        Max 
-0.0308247 -0.0088126 -0.0003627  0.0133905  0.0305743 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.003476   0.012004  -0.290   0.7767    
beers        0.018100   0.002135   8.478 1.18e-06 ***
```

```
sexmale     -0.019763   0.009086  -2.175   0.0487 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01816 on 13 degrees of freedom
Multiple R-squared:  0.8532,    Adjusted R-squared:  0.8307
F-statistic: 37.79 on 2 and 13 DF,  p-value: 3.826e-06
```

```
plot(md.bac2,which=1,las=1)
```



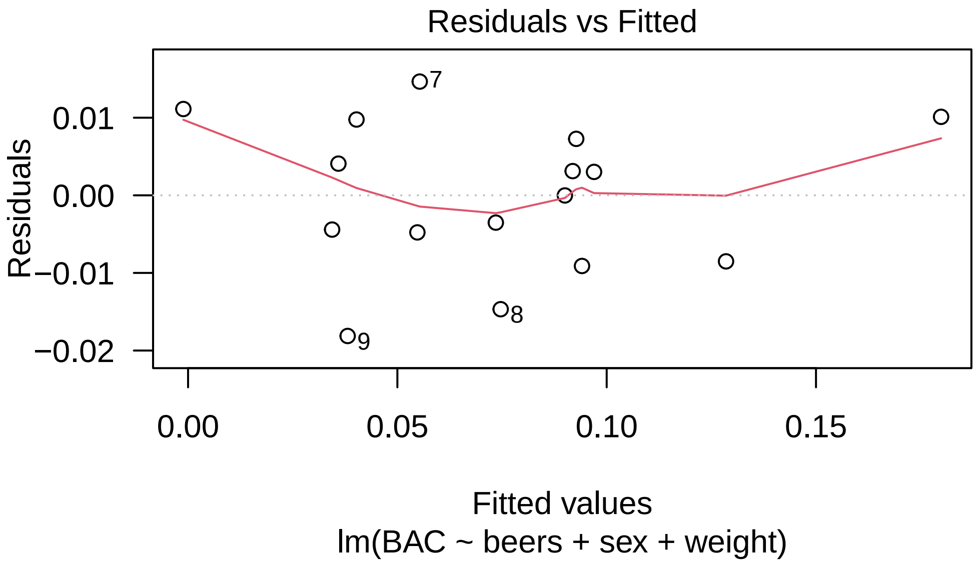
Finally, we also include weight as a predictor variable. As a result, the `sex` variable is no longer seen to have any statistical significance in determining blood alcohol content. We note that the range of the residuals has decreased from the previous models meaning that this last model provides a tighter fit to the data.

```
md.bac3 <- lm( BAC~beers+sex+weight, data=bac  )
summary(md.bac3)
```

```
Call:
lm(formula = BAC ~ beers + sex + weight, data = bac)
```

```
Residuals:
      Min        1Q    Median        3Q       Max
-0.018125 -0.005713  0.001501  0.007896  0.014655

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.871e-02  1.097e-02   3.528 0.004164 **
beers        1.990e-02  1.309e-03  15.196 3.35e-09 ***
sexmale     -3.240e-03  6.286e-03  -0.515 0.615584
weight      -3.444e-04  6.842e-05  -5.034 0.000292 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01072 on 12 degrees of freedom
Multiple R-squared:  0.9528,    Adjusted R-squared:  0.941
F-statistic: 80.81 on 3 and 12 DF,  p-value: 3.162e-08
```

```
plot(md.bac3,which=1,las=1)
```

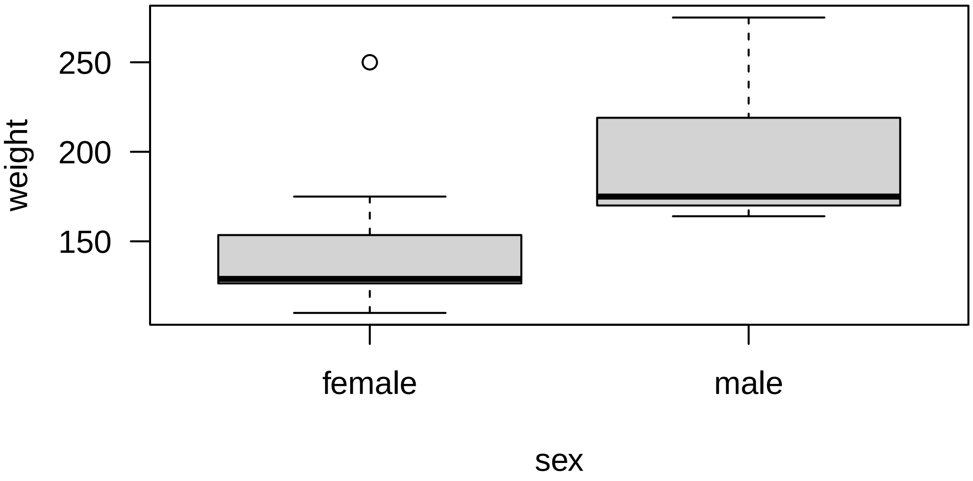


The significance of `sex` is the second model was only due to the males in this dataset having, on average, higher `weight` than the females, which can be seen in the boxplot below. Thus, these two predictor variables are confounded or correlated. This is not seen in the case of `beers consumed` vs `sex`. For our two numeric variables, `beers consumed` and `weight`, there is a slight positive correlation of about 0.25.

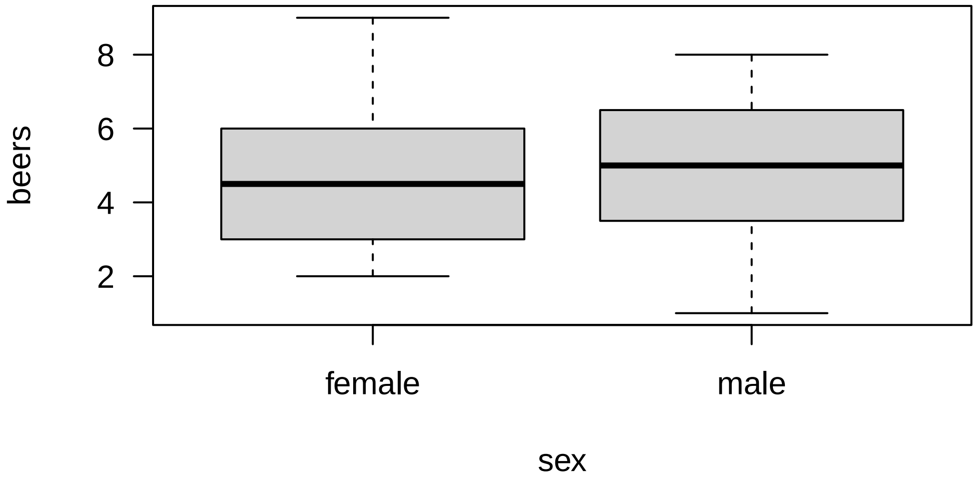
```
bac0 <- bac
bac0$sex <- as.numeric(as.factor(bac0$sex))
cor(bac0[,2:4])
```

```
          weight        sex      beers
weight 1.0000000 0.51282368 0.24887716
sex    0.5128237 1.00000000 0.02937367
beers  0.2488772 0.02937367 1.00000000
```

```
boxplot(weight~sex,data=bac,las=1)
```



```
boxplot(beers~sex,data=bac,las=1)
```



2.3 Transformation

Often, data does not follow all of the assumptions of the ordinary least squares regression model. However, it is often possible to transform data in order to correct for this deviations. We will consider such methods in the following subsections. One dissatisfying remark about such methods is that they often are applied *empirically*, which less euphemistically means in an add-hoc way. Sometimes there may be genuine information suggesting certain methods for transforming data. Other times, transforms are chosen because they seem to work.

2.3.1 Variance Stabilizing

One of the major requirements of the least squares model is that the variance of the errors is constant, which is that $\mathrm{Var}\,(y_i) = \mathrm{Var}\,(\varepsilon_i) = \sigma^2$ for $i = 1, \ldots, n$. Mainly, when σ^2 is non-constant in y, problems can occur. Our goal is thus to find some transformation $T(y)$ so that $\mathrm{Var}\,(T(y_i))$ is constant for all $i = 1, \ldots, n$.

Such a transformation $T(\cdot)$ can be determined through a tool known as the *delta method*, which is beyond the scope of these notes.
See Chapter 3, Asymptotic Statistics, A W van der Vaart or Delta Method for more. However, we will consider a simplified version for our purposes. For simplicity of notation, we write $\mathrm{E}Y = \mu$ instead of $\mathrm{E}Y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$. Furthermore, assume that T is twice

differentiable. Note that we only require the second derivative for a more pleasant exposition. Then, Taylor's theorem says that

$$T(Y) = T(\mu) + T'(\mu)(Y - \mu) + \frac{T''(\xi)}{2}(Y - \mu)^2$$

for some ξ between Y and μ. We can, with a little hand-waving, ignore the higher order remainder term and just write

$$T(Y) \approx T(\mu) + T'(\mu)(Y - \mu),$$

which implies that

$$\mathrm{E}T(Y) \approx T(\mu) \quad \text{and} \quad \mathrm{Var}\left(T(Y)\right) \approx T'(\mu)^2 \mathrm{Var}\left(Y\right).$$

We want a transformation such that $\mathrm{Var}\left(T(Y)\right) = 1$ is constant. Meanwhile, we assume that the variance of Y is a function of the mean μ, which is $\mathrm{Var}\left(Y\right) = s(\mu)^2$. Hence, we need to solve

$$1 = T'(\mu)^2 s(\mu)^2 \quad \text{or} \quad T(\mu) = \int \frac{1}{s(\mu)} d\mu.$$

Example 2.1. For a trivial example, assume that $s(\mu) = \sigma$ is already constant. Then,

$$T(\mu) = \int \frac{1}{\sigma} d\mu = \mu/\sigma.$$

Thus, T is just scaling by σ to achieve a unit variance.

Example 2.2. Now, consider the nontrivial example with $s(\mu) = \sqrt{\mu}$, which is that the variance of Y is a linear function of μ. Then,

$$T(\mu) = \int \frac{1}{\sqrt{\mu}} d\mu = 2\sqrt{\mu}.$$

This is the square root transformation, which is applied to, for example, Poisson data. The coefficient of 2 in the above derivation can be dropped as we are not concerned with the scaling.

Given that we have found a suitable transform, we can then apply it to the data $y_1, \dots, y_n$ to get a model of the form

$$T(y) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p + \varepsilon$$

where the variance of ε is constant–i.e not a function of the regressors.

2.3.2 Linearization

Another key assumption is that the relationship between the regressors and response, x and y, is linear. If there is reason to believe that

$$y = f(\beta_0 + \beta_1 x_1 + ... + \beta_p x_p + \varepsilon)$$

and that $f(\cdot)$ is invertible, then we can rewrite this model as

$$y' = f^{-1}(y) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p + \varepsilon$$

and apply our usual linear regression tools.

An example from the textbook, which is also quite common in practice, is to assume that

$$y = c\mathrm{e}^{\beta_1 x_1 + ... + \beta_p x_p + \varepsilon}$$

and apply a logarithm to transform to

$$y' = \log y = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p + \varepsilon$$

where $\beta_0 = \log c$. Furthermore, if ε has a normal distribution then e^{ε} has a log-normal distribution. This model is particularly useful when one is dealing with exponential growth in some population.

Remark 2.2. Linearization by applying a function to y looks very similar to the variance stabilizing transforms of the previous section. In fact, such transforms have an effect on both the linearity and the variance of the model and should be used with care. Often non-linear methods are preferred.

Sometimes it is beneficial to transform the regressors, x's, as well. As we are not treating them as random variables, there are less problems to consider.

$$y = \beta_0 + \beta_1 f(x) + \varepsilon \quad \Rightarrow \quad y = \beta_0 + \beta_1 x' + \varepsilon, \quad x = f^{-1}(x')$$

Examples include

$$\begin{aligned} y = \beta_0 + \beta_1 \log x + \varepsilon \quad &\Rightarrow \quad y = \beta_0 + \beta_1 x' + \varepsilon, \quad x = \mathrm{e}^{x'} \\ y = \beta_0 + \beta_1 x^2 + \varepsilon \quad &\Rightarrow \quad y = \beta_0 + \beta_1 x' + \varepsilon, \quad x = \sqrt{x'} \end{aligned}$$

This second example can be alternatively dealt with by employing polynomial regression to be discussed in a subsequent section.

2.3.3 Box-Cox and the power transform

In short, Box-Cox is a family of transforms parametrized by some $\lambda \in \mathbb{R}$, which can be optimized via maximum likelihood. Specifically, for $y_i > 0$, we aim to choose the best transform of the form

$$y_i \to y_i^{(\lambda)} = \begin{cases} \frac{y_i^\lambda - 1}{\lambda} & \lambda \neq 0 \\ \log y_i & \lambda = 0 \end{cases}$$

by maximizing the likelihood as we did in Chapter 1, but with parameters β, σ^2, and λ.

To do this, we assume that the transformed variables follow all of the usual least squares regression assumptions and hence have a joint normal distribution with

$$f(Y^{(\lambda)}) = (2\pi\sigma^2)^{-n/2} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - X_{i,\cdot}\beta)^2 \right).$$

Transforming $Y \to Y^{(\lambda)}$ is a change of variables with Jacobian

$$\prod_{i=1}^n \frac{dy_i}{dy_i}^{(\lambda)} = \prod_{i=1}^n y_i^{\lambda-1}.$$

Hence, the likelihood function in terms of X and y is

$$L(\beta, \sigma^2, \lambda | y, X) = (2\pi\sigma^2)^{-n/2} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i^\lambda - X_{i,\cdot}\beta)^2 \right) \prod_{i=1}^n y_i^{\lambda-1}$$

with log likelihood

$$\log L = -\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - X_{i,\cdot}\beta)^2 + (\lambda - 1) \sum_{i=1}^n \log y_i.$$

From here, the MLEs for β and σ^2 are solved for as before but are now in terms of the transformed $Y^{(\lambda)}$.

$$\begin{aligned} \hat{\beta} &= (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} Y^{(\lambda)} \\ \hat{\sigma}^2 &= \frac{1}{n} \sum_{i=1}^n (y_i^{(\lambda)} - X_{i,\cdot}\hat{\beta})^2 = \frac{SS_{\text{res}}^{(\lambda)}}{n}. \end{aligned}$$

Plugging these into the log likelihood gives and replacing all of the constants with some C gives

$$\begin{aligned} \log L &= -\frac{n}{2} \log(2\pi\hat{\sigma}^2) - \frac{n}{2} + (\lambda - 1) \sum_{i=1}^n \log y_i \\ &= C - \frac{n}{2} \log \hat{\sigma}^2 + \log\left((\textstyle\prod y_i)^{\lambda-1} \right). \end{aligned}$$

Defining the geometric mean of the y_i to be $\gamma = (\prod_{i=1}^{n} y_i)^{1/n}$, we have

$$\begin{aligned} \log L &= C - \frac{n}{2}\log\hat{\sigma}^2 + \frac{n}{2}\log\left(\gamma^{2(\lambda-1)}\right) \\ &= C - \frac{n}{2}\log\left(\frac{\hat{\sigma}^2}{\gamma^{2(\lambda-1)}}\right). \end{aligned}$$

Considering the term inside the log, we have that it is just the residual sum of squares from the least squares regression

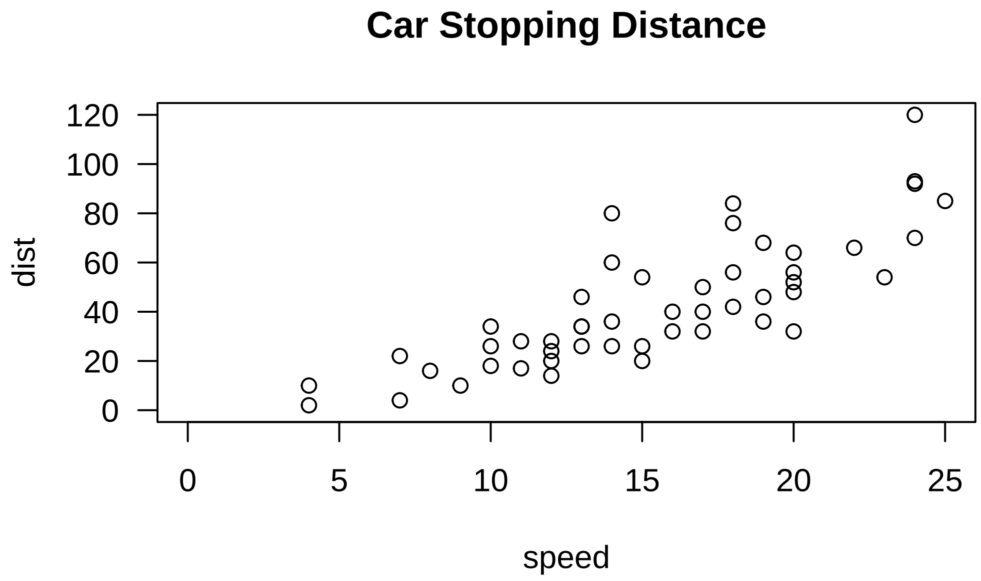
$$\frac{Y^{(\lambda)}}{\gamma^{\lambda-1}} = X\theta + \varepsilon$$

where $\theta \in \mathbb{R}^{p+1}$ is a transformed version of the original β. Hence, we can choose $\hat{\lambda}$ by maximizing the log likelihood above, which is equivalent to minimizing the residual sum of squares for this new model. This can be calculated numerically in statistical programs like R.

2.3.4 Cars Data

To test some of these linearization techniques, we consider the cars dataset, which is included in the standard distribution of `R`. It consists of 50 observations of a car's speed and a car's stopping distance. The goal is to model and predict the stopping distance given the speed of the car. Such a study could be used, for example, for influencing speed limits and other road rules for the sake of public safety. The observed speeds range from 4 to 25 mph.

```r
plot(
  cars,las=1,main="Car Stopping Distance",
  xlim=c(0,max(cars$speed)),
  ylim=c(0,max(cars$dist))
)
```



We can first fit a simple regression to the data with `lm( dist~speed, data=cars)`. This results in a significant p-value, an $R^2 = 0.651$, and an estimated model of

$$(\text{dist}) = -17.6 + 3.9(\text{speed}) + \varepsilon.$$

If we wanted to extrapolate a bit, we can use this model to predict the stopping distance for a speed of 50 mph, which is 179 feet.

```
md.car.lin = lm( dist~speed, data=cars)
summary(md.car.lin)
```

```
Call:
lm(formula = dist ~ speed, data = cars)

Residuals:
    Min      1Q  Median      3Q     Max 
-29.069  -9.525  -2.272   9.215  43.201 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -17.5791     6.7584  -2.601   0.0123 *  
speed         3.9324     0.4155   9.464 1.49e-12 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.38 on 48 degrees of freedom
Multiple R-squared:  0.6511,    Adjusted R-squared:  0.6438 
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

```r
predict(md.car.lin,data.frame(speed=50))
```

```
       1 
179.0413 
```

We could stop here and be happy with a significant fit. However, looking at the data, there seems to be a nonlinear relationship between speed and stopping distance. Hence, we could try to fit a model with the response being the square root of the stopping distance: `lm( sqrt(dist)~speed, data=cars)`. Doing so results in

$$\sqrt{(\text{dist})} = 1.28 + 0.32(\text{speed}) + \varepsilon.$$

In this case, we similarly get a significant p-value and a slightly higher $R^2 = 0.709$. The prediction for the stopping distance for a speed of 50 mph is now the much higher 302 feet.

```r
md.car.sqrt = lm( sqrt(dist)~speed, data=cars)
summary(md.car.sqrt)
```

```

Call:
lm(formula = sqrt(dist) ~ speed, data = cars)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.0684 -0.6983 -0.1799  0.5909  3.1534 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.27705    0.48444   2.636   0.0113 *  
speed        0.32241    0.02978  10.825 1.77e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.102 on 48 degrees of freedom
Multiple R-squared:  0.7094,    Adjusted R-squared:  0.7034 
F-statistic: 117.2 on 1 and 48 DF,  p-value: 1.773e-14
```

```r
# Note that we have to use ( )^2 to get our
# prediction on the right scale
predict(md.car.sqrt,data.frame(speed=50))^2
```

```
       1 
302.6791 
```

We can further apply a log-log transform with `lm( log(dist)~log(speed), data=cars)`. This results in an even higher $R^2 = 0.733$. The fitted model is

$$\log(y) = -0.73 + 1.6 \log(x) + \varepsilon,$$

and the predicted stopping distance for a car travelling at 50 mph is 254 feet. Note that the square root transform is modelling $y \propto x^2$ whereas the log-log transform is modelling $y \propto x^{1.6}$, which is a slower rate of increase.

```r
md.car.log = lm( log(dist)~log(speed), data=cars)
summary(md.car.log)
```

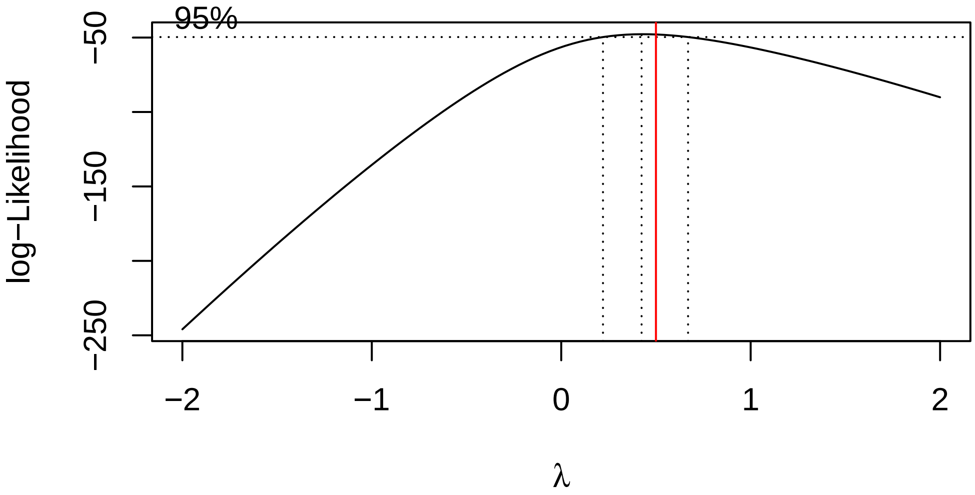
```

Call:
lm(formula = log(dist) ~ log(speed), data = cars)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.00215 -0.24578 -0.02898  0.20717  0.88289 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -0.7297     0.3758  -1.941   0.0581 .  
log(speed)    1.6024     0.1395  11.484 2.26e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4053 on 48 degrees of freedom
Multiple R-squared:  0.7331,	Adjusted R-squared:  0.7276 
F-statistic: 131.9 on 1 and 48 DF,  p-value: 2.259e-15
```

```r
# Note that we have to use exp( ) to get our
# prediction on the right scale
exp( predict(md.car.log,data.frame(speed=50)) )
```

```
       1
254.4037
```

We can also let the data decide on the best transformation by using the Box-Cox transformation. In this case, we may want to test the hypothesis that the optimal power to transform with is 1/2; i.e. the square root transformation. The Box-Cox transform choose a power of $\lambda = 0.424$. However, the value 1/2 lies within the confidence interval for λ. We may prefer to use the square root transform instead of the power 0.424, since the square root is more interpretable.

```
# Need the package MASS for boxcox
library(MASS)
# Do the Box-Cox transform
bc  = boxcox(md.car.lin);
abline(v=c(0.5),col=c('red'))
```



```
lmb = bc$x[ which.max(bc$y) ];
print(paste("Optimal lambda is ",lmb))
```

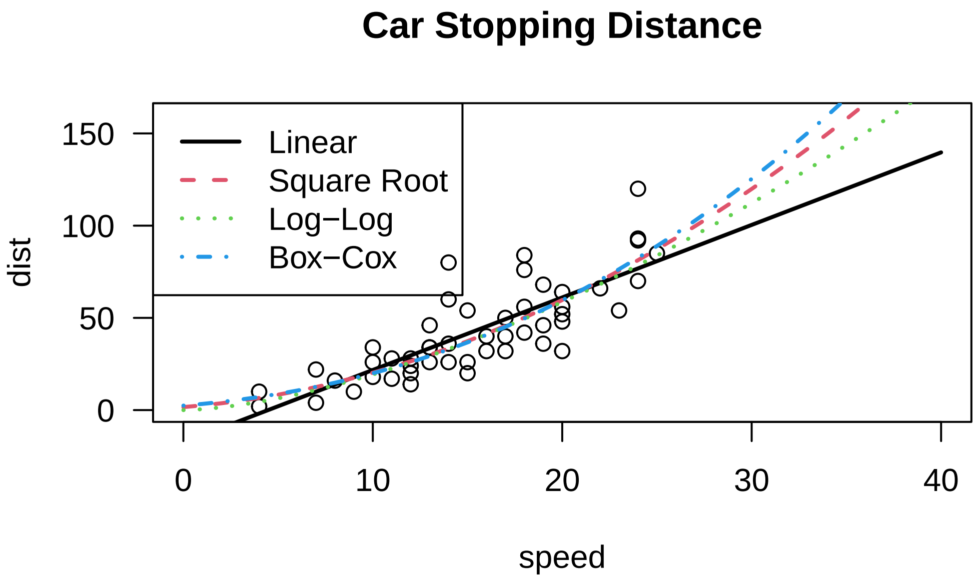
```
[1] "Optimal lambda is  0.424242424242424"
```

```
carsBC = cbind(cars, distBC = (cars$dist^lmb-1)/lmb)
md.car.bc = lm( distBC~speed, data=carsBC );
summary(md.car.bc);
```

```

Call:
lm(formula = distBC ~ speed, data = carsBC)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.0926 -1.0444 -0.3055  0.7999  4.7520 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.08227    0.73856   1.465    0.149    
speed        0.49541    0.04541  10.910 1.35e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.681 on 48 degrees of freedom
Multiple R-squared:  0.7126,	Adjusted R-squared:  0.7066 
F-statistic:   119 on 1 and 48 DF,  p-value: 1.354e-14
```

The four models considered are plotted below. As we move away from the range of the regressors–i.e 4 to 25 mph–the models begin to diverge making extrapolating a bit dangerous.

```
plot(
  cars,las=1,main="Car Stopping Distance",
  xlim=c(0,40),
  ylim=c(0,160)
)
xx = 0:40
lines(
  xx, predict(md.car.lin,data.frame(speed=xx)),
  col=1,lty=1,lwd=2
)
lines(
  xx, predict(md.car.sqrt,data.frame(speed=xx))^2,
  col=2,lty=2,lwd=2
)
lines(
  xx, exp(predict(md.car.log,data.frame(speed=xx))),
  col=3,lty=3,lwd=2
)
lines(
  xx, (lmb*predict(md.car.bc,data.frame(speed=xx))+1)^(1/lmb),
  col=4,lty=4,lwd=2
)
legend(
  "topleft",legend = c("Linear","Square Root","Log-Log","Box-Cox"),
  col=1:4,lty=1:4,lwd=2
)
```



The choice in transformation can also have a big effect on both predictions and the confidence surrounding such predictions.

```
new.val = 40;
predict(md.car.lin,data.frame(speed=new.val),interval="prediction")
```

```
       fit      lwr      upr
1 139.7173 102.3311 177.1034
```

```
predict(md.car.sqrt,data.frame(speed=new.val),interval="prediction")^2
```

```
       fit      lwr      upr
1 200.8895 132.1058 284.0361
```

```
exp(predict(md.car.log,data.frame(speed=new.val),interval="prediction"))
```

```
       fit      lwr      upr
1 177.9245 74.39848 425.5077
```

```
(lmb*predict(
  md.car.bc,data.frame(speed=new.val),interval="prediction"
)+1)^(1/lmb)
```

```
      fit      lwr      upr
1 220.465 139.8189 322.8649
```

2.4 Polynomial Regression

In the following subsections, we will only consider models with a single regressor $x \in \mathbb{R}$ until the final part below where we will consider polynomials in multiple variables $x_1, \ldots, x_p$.

One of the examples of atypical residual behavior from above occurs when there is an unaccounted for quadratic term in the model such as trying to fit a model of the form

$$y = \beta_0 + \beta_1 x_1 + \varepsilon$$

to data generated by

$$y = \beta_0 + \beta_1 x_1^2 + \varepsilon.$$

If this is suspected, we can look at the residuals and try to transform x, such as $x \to \sqrt{x}$, in order to put this back into the linear model framework. However, we can also just fit a polynomial model to our data.

Consider the setting where we observe n data pairs $(y_i, x_i) \in \mathbb{R}^2$. We can then attempt to fit a model of the form

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_p x^p + \varepsilon$$

to the observations. The $n \times p$ design matrix will look like

$$X = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^p \\ 1 & x_2 & x_2^2 & \dots & x_2^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^p \end{pmatrix}$$

and the parameters can be estimated as usual: $\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$. This matrix arises in Linear Algebra as the Vandermonde matrix.

Remark 2.3. While the columns of X are, in general, linearly independent, as p gets larger, many problems can occur. In particular, the columns become near linearly dependent resulting in instability when computing $(X^{\mathrm{T}}X)^{-1}$.

2.4.1 Model Problems

Polynomial regression is very powerful for modelling, but can also lead to very erroneous results if used incorrectly. What follows are some potential issues to take into consideration.

2.4.1.1 Overfitting

As a general rule, the degree of the polynomial model should be kept as low as possible. High order polynomials can be misleading as they can often fit the data quite well. In fact, given n data points, it is possible to fit an $n-1$ degree polynomial that passes through each data point. In this extreme case, all of the residuals would be zero, but we would never expect this to be the correct model for the data.

Problems can occur even when p is much smaller than n. As an example, two models were fit to $n = 50$ data points generated from the model

$$y = 3x^2 + \varepsilon$$

with $\varepsilon \sim \mathcal{N}(0, 1)$. The first was a cubic model. The second was a degree 20 model. The first regression resulted in three significant regressors. Note that in these two cases, orthogonal polynomials were used to maintain numerical stability.

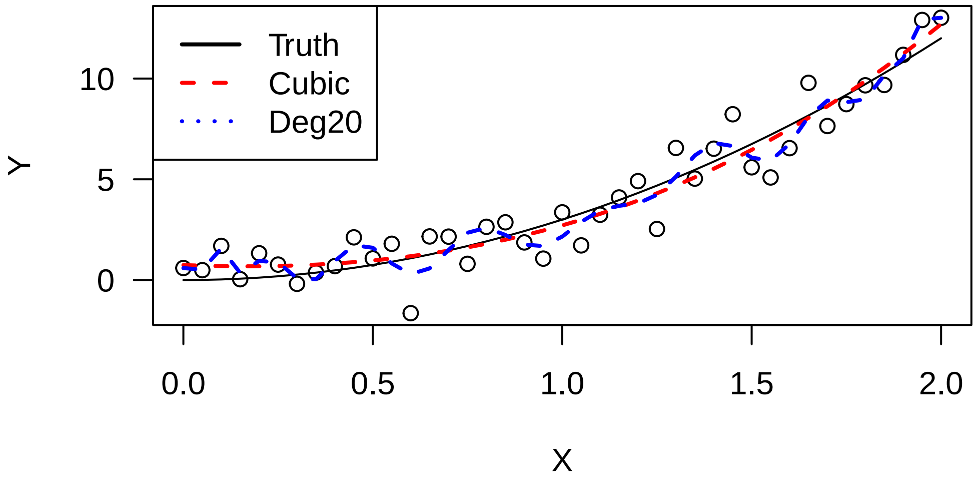
```
set.seed(128)
# Simulate some regression data with
# a quadratic trend
xx  = seq(0,2,0.05)
len = length(xx)
yy  = 3*xx^2 + rnorm(n=len,0,1)
# Fit a cubic model using orthogonal polynomials
# and extract the most significant terms
md3  <- lm( yy~poly(xx,3) )
sm3  <- summary(md3)
sm3$coefficients[which(sm3$coefficients[,4]<0.01),]
```

```
              Estimate Std. Error   t value     Pr(>|t|)
(Intercept)   4.115491  0.1730479 23.782384 5.118095e-24
poly(xx, 3)1 21.730420  1.1080471 19.611459 3.843069e-21
poly(xx, 3)2  8.025826  1.1080471  7.243217 1.347334e-08
```

```
# Fit a degree-20 model using orthogonal polynomials
# and extract the most significant terms
md20 <- lm( yy~poly(xx,20) )
sm20<- summary(md20)
sm20$coefficients[which(sm20$coefficients[,4]<0.01),]
```

```
               Estimate Std. Error   t value     Pr(>|t|)
(Intercept)    4.115491  0.1944119 21.168925 3.641662e-15
poly(xx, 20)1 21.730420  1.2448436 17.456345 1.424531e-13
poly(xx, 20)2  8.025826  1.2448436  6.447256 2.748160e-06
```

```
# Plot the data and models
plot( xx,yy, xlab='X',ylab="Y",las=1 )
lines( xx, 3*xx^2,col='black',lty=1 )
lines( xx, predict(md3),col='red',lty=2,lwd=2 )
lines( xx, predict(md20),col='blue',lty=2,lwd=2 )
legend(
  "topleft",legend = c("Truth","Cubic","Deg20"),
  col = c("black","red","blue"), lty=1:3,lwd=2
)
```



2.4.1.2 Extrapolating

The overfitting from the previous section also indicates problems that can occur when extrapolating with polynomial models. When high degrees are present, the best fit curve can change directions quickly and even make impossible or illogical predictions.

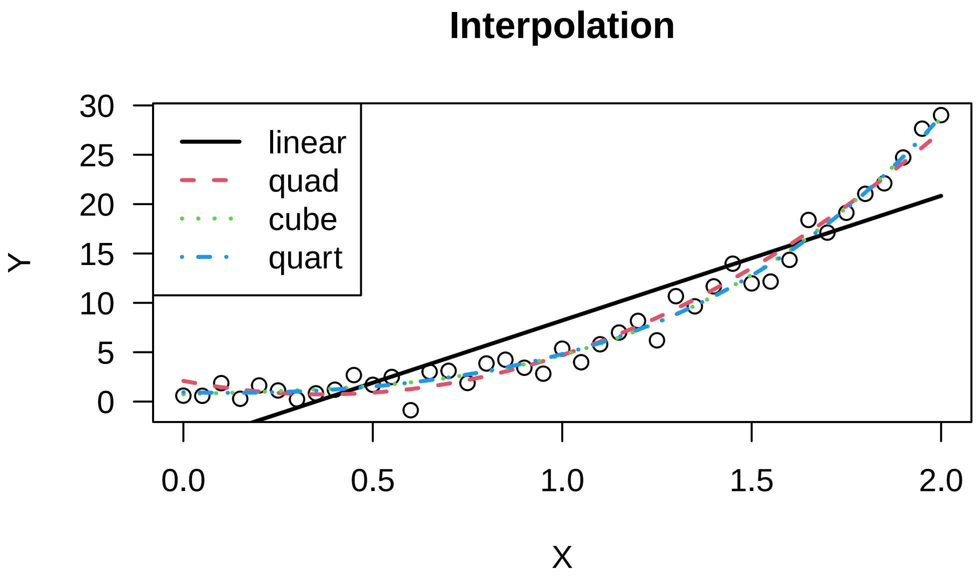
Beyond that, even when the fitted polynomial models all trend in the same general direction, polynomials of different degree will diverge quickly from one another. Consider a model where the data are generated from

$$y = 2x + 3x^3 + \varepsilon.$$

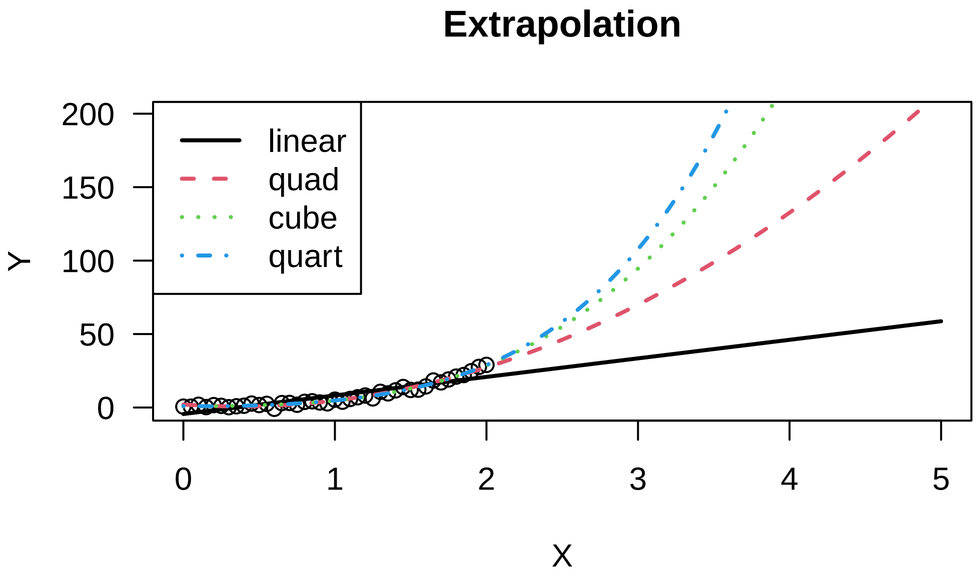
All four models displayed below generated very significant F tests. However, each one will give very different answers to predicting the value of y when $x = 5$.

```r
set.seed(128)
# Simulate some regression data with
# a quadratic trend
xx  = seq(0,2,0.05)
len = length(xx)
yy  = 2*xx + 3*xx^3 + rnorm(n=len,0,1)
# Fit some low degree polynomial models
md1 = lm( yy~poly(xx,1) )
md2 = lm( yy~poly(xx,2) )
```

```r
md3 = lm( yy~poly(xx,3) )
md4 = lm( yy~poly(xx,4) )
# plot the models (interpolate)
plot(xx,yy,las=1,xlab="X",ylab="Y",main="Interpolation")
lines(xx,predict(md1),lty=1,col=1,lwd=2)
lines(xx,predict(md2),lty=2,col=2,lwd=2)
lines(xx,predict(md3),lty=3,col=3,lwd=2)
lines(xx,predict(md4),lty=4,col=4,lwd=2)
legend(
  "topleft",legend = c("linear","quad","cube","quart"),
  lty=1:4,col=1:4,lwd=2
)
```



```r
# plot the models (extrapolate)
xx.ext = seq(0,5,0.05)
plot(
  xx,yy,las=1,xlab="X",ylab="Y",main="Extrapolation",
  xlim=c(0,5),ylim=c(min(yy),200)
)
lines(xx.ext,predict(md1,data.frame(xx=xx.ext)),lty=1,col=1,lwd=2)
lines(xx.ext,predict(md2,data.frame(xx=xx.ext)),lty=2,col=2,lwd=2)
lines(xx.ext,predict(md3,data.frame(xx=xx.ext)),lty=3,col=3,lwd=2)
lines(xx.ext,predict(md4,data.frame(xx=xx.ext)),lty=4,col=4,lwd=2)
legend(
  "topleft",legend = c("linear","quad","cube","quart"),
  lty=1:4,col=1:4,lwd=2
)
```



2.4.1.3 Hierarchy

An hierarchical polynomial model is one such that if it contains a term of degree k, then it will contain all terms of order $i = 0, 1 \ldots, k-1$ as well. In practice, it is not strictly necessary to do this. However, doing so will maintain invariance to linear shifts in the data.

Consider the simple linear regression model $y = \beta_0 + \beta_1 x + \varepsilon$. If we were to shift the values of x by some constant a, then

$$\begin{aligned} y &= \beta_0 + \beta_1(x+a) + \varepsilon \\ &= (\beta_0 + a\beta_1) + \beta_1 x + \varepsilon \\ &= \beta_0' + \beta_1 x + \varepsilon \end{aligned}$$

and we still have the same model but with a modified intercept term.

Now consider the polynomial regression model $y = \beta_0 + \beta_2 x^2 + \varepsilon$. If we were to similarly shift the values of x by some constant a, then

$$\begin{aligned} y &= \beta_0 + \beta_2 (x+a)^2 + \varepsilon \\ &= (\beta_0 + a^2\beta_2) + 2\beta_2 a x + \beta_2 x^2 + \varepsilon \\ &= \beta_0' + \beta_1' x + \beta_2 x^2 + \varepsilon. \end{aligned}$$

Now, our model has a linear term, that is $\beta_1' x$, in it, which was not there before.

In general, for a degree p model, if x is shifted by some constant a, then

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i x^i \quad \Rightarrow \quad y = \beta_0' + \sum_{i=1}^{p} \beta_i' x^i.$$

Thus, the model is invariant under linear translation.

2.4.2 Piecewise Polynomials

While a single high degree polynomial can be fit very closely to the observed data, there exist the already discussed problems of overfitting and subsequent extrapolation. Hence, an alternative to capture the behaviour of highly nonlinear data is to apply a piecewise polynomial model, which is often referred to as a spline model.

To begin, assume that the observed regressors take values in the interval $[a, b]$. Then, we partition the interval with $k+1$ *knots* by $a = t_1 < t_2 < \ldots < t_{k+1} = b$. The general spline model of order p takes on the form

$$y = \sum_{j=1}^{k} \beta_{0,j} 1\big[x \geq t_j\big] + \sum_{i=1}^{p} \sum_{j=1}^{k} \beta_{i,j} (x - t_j)_+^i \tag{2.1}$$

where $1\big[x > t_j\big]$ is the indicator function that takes on a value of 0 when $x < t_j$ and a value of 1 otherwise, and where $(x - t_j)_+^i = (x - t_j)^i 1\big[x > t_j\big]$.

Equation 2.1 is equivalent to fitting a separate pth order polynomial to the data in each interval $[t_j, t_{j+1}]$. While doing so does result in many parameters to estimate, it will allow us to perform hypothesis tests for continuity and differentiability of the data, which we will discuss in the next section.

Submodels of Equation 2.1 have a collection of interesting properties. If the coefficients $\beta_{0,j}$ are set to 0, then we will fit a model that ensures continuity at the knots. For example, the model

$$y = \beta_{0,1} + \sum_{j=1}^{k} \beta_{1,j} (x - t_j)_+$$

is a piecewise linear model with continuity at the knots. Conversely, a model of the form

$$y = \sum_{j=1}^{k} \beta_{0,j} 1\left[x \geq t_j\right]$$

fits a piecewise constant model and can be used to look for change points in an otherwise constant process.

In practice, spline models with only cubic terms are generally preferred as they have a high enough order to ensure a good amount of smoothness–i.e. the curves are twice continuously differentiable–but generally do not lead to overfitting of the data.

2.4.2.1 A Spline Example

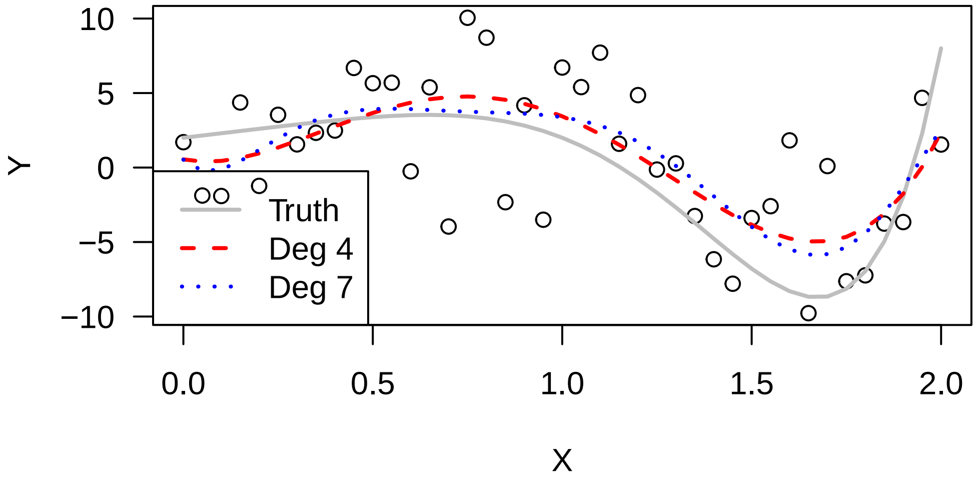
Note that there are R packages to fit spline models to data. However, in this example we will do it manually for pedagogical purposes.

Consider a model of the form

$$y = 2 + 3x - 4x^5 + x^7 + \varepsilon$$

with $\varepsilon \sim \mathcal{N}(0, 4)$ with $n = 41$ observations with regressor $x \in [0, 2]$. A degree 4 and degree 7 polynomial regression were fit to the data, and the results are displayed below. These deviate quickly from the true curve outside of the range of the data.

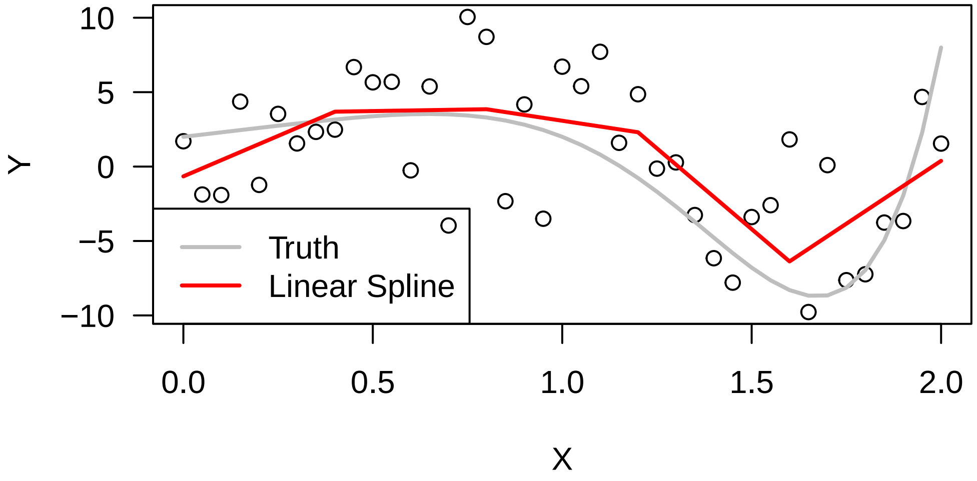
```r
set.seed(256)
# Generate Data from a degree-7 polynomial
xx  = seq(0,2,0.05)
len = length(xx)
yy  = 2 + 3*xx - 4*xx^5 + xx^7 + rnorm(len,0,4)
# Fit degree 4 and degree 7 models
md4 = lm( yy~poly(xx,4) )
md7 = lm( yy~poly(xx,7) )
# Plot polynomial models
plot( xx,yy,las=1,xlab="X",ylab="Y" )
lines( xx, 2 + 3*xx - 4*xx^5 + xx^7, col='gray',lwd=2,lty=1 )
lines( xx, predict(md4), col='red',lwd=2,lty=2 )
lines( xx, predict(md7), col='blue',lwd=2,lty=3 )
legend(
  "bottomleft",legend = c("Truth","Deg 4","Deg 7"),
  col=c("gray","red","blue"),lty=1:3,lwd=2
)
```



The next plot fits a piecewise linear spline with 6 knots–i.e. $k = 5$. The model is of the form

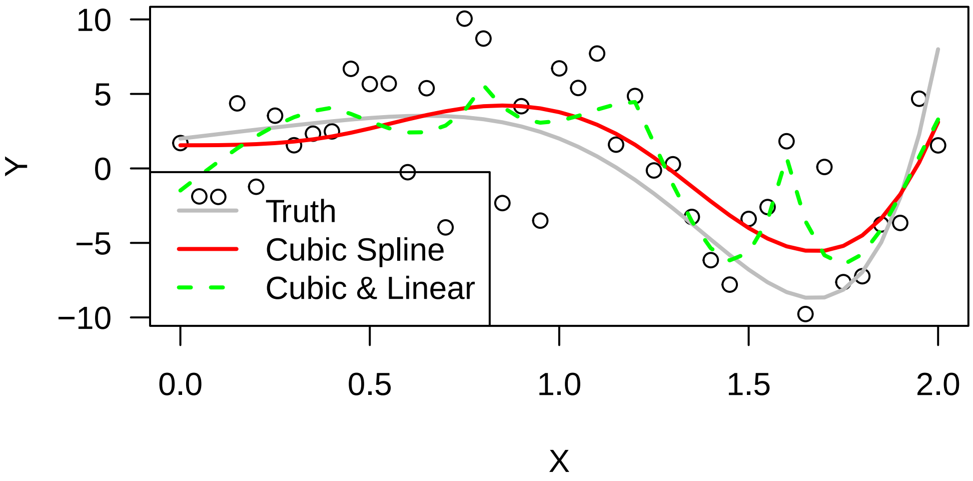
$$y = \beta_{0,1} + \beta_{1,1}(x - 0.0)_+ + \beta_{1,2}(x - 0.4)_+ + ... + \beta_{1,5}(x - 1.6)_+.$$

```
# Manually creating the inputs to the regression model
knt  = c(1,9,17,25,33)
reg1 = c(rep(0,knt[1]-1),xx[knt[1]:len] - xx[knt[1]]);
reg2 = c(rep(0,knt[2]-1),xx[knt[2]:len] - xx[knt[2]]);
reg3 = c(rep(0,knt[3]-1),xx[knt[3]:len] - xx[knt[3]]);
reg4 = c(rep(0,knt[4]-1),xx[knt[4]:len] - xx[knt[4]]);
reg5 = c(rep(0,knt[5]-1),xx[knt[5]:len] - xx[knt[5]]);
# Fit linear spline model
md.1spline <- lm( yy~reg1+reg2+reg3+reg4+reg5 )
# Plot the linear spline model
plot(
  xx,yy,las=1,xlab="X",ylab="Y"
)
lines( xx, 2 + 3*xx - 4*xx^5 + xx^7, col='gray',lwd=2,lty=1 )
lines(
  xx,predict(md.1spline),lwd=2,col='red'
)
legend(
  "bottomleft",legend = c("Truth","Linear Spline"),
  col=c("gray","red"),lty=1,lwd=2
)
```



The final plot fits a spline with only piecewise cubic terms and a spline with cubic and linear terms. The only cubic model provides a very reasonable approximation to the data. The cubic-linear spline model becomes a bit crazy.

```
# Manually creating the inputs to the regression model
knt  = c(1,9,17,25,33)
cub1 = c(rep(0,knt[1]-1),xx[knt[1]:len] - xx[knt[1]])^3;
cub2 = c(rep(0,knt[2]-1),xx[knt[2]:len] - xx[knt[2]])^3;
cub3 = c(rep(0,knt[3]-1),xx[knt[3]:len] - xx[knt[3]])^3;
cub4 = c(rep(0,knt[4]-1),xx[knt[4]:len] - xx[knt[4]])^3;
cub5 = c(rep(0,knt[5]-1),xx[knt[5]:len] - xx[knt[5]])^3;
# Fit cubic spline model
md.3spline <- lm( yy~cub1+cub2+cub3+cub4+cub5 )
# Fit linear and cubic spline model
md.13spline <- lm(
  yy~reg1+reg2+reg3+reg4+reg5+cub1+cub2+cub3+cub4+cub5
)
# Plot the linear spline model
plot(
  xx,yy,las=1,xlab="X",ylab="Y"
)
lines( xx, 2 + 3*xx - 4*xx^5 + xx^7, col='gray',lwd=2,lty=1 )
lines(
  xx,predict(md.3spline),lwd=2,col='red'
)
lines(
  xx,predict(md.13spline),lwd=2,col='green',lty=2
)
legend(
  "bottomleft",legend = c("Truth","Cubic Spline","Cubic & Linear"),
  col=c("gray","red","green"),lty=c(1,1,2),lwd=2
)
```



2.4.2.2 Hypothesis testing for spline models

It is useful to consider what the hypothesis tests mean in the context of spline models. For example, consider fitting the piecewise constant model to some data:

$$y = \sum_{j=1}^{k} \beta_{0,j} 1 \left[x \geq t_j \right] .$$

The usual F-test will consider the hypotheses

$$H_0 : \beta_{0,2} = \ldots = \beta_{0,k} = 0 \qquad H_1 : \exists i \in \{2, 3, \ldots, k\} \text{ s.t. } \beta_{0,i} \neq 0.$$

This hypothesis is asking whether or not we believe the mean of the observations changes as x increases. *Note that $\beta_{0,1}$ is the overall intercept term in this model.*

We can also compare two different spline models with a partial F-test. For example,

$$\begin{array}{ll} \text{Model 1:} & y = \beta_{0,1} + \sum_{j=1}^{k} \beta_{3,j}(x - t_j)_+^3 \\ \text{Model 2:} & y = \beta_{0,1} + \sum_{j=2}^{k} \beta_{0,j} 1\left[x \geq t_j\right] + \sum_{j=1}^{k} \beta_{3,j}(x - t_j)_+^3 \end{array}$$

The partial F-test between models 1 and 2 asks whether or not the addition of the piecewise constant terms adds any explanatory power to our model. Equivalently, it is testing for whether or not the model has any discontinuities in it. Similarly, first order terms can be used to test for differentiability.

Instead of constructing a bigger model by adding polynomial terms of different orders, it is also possible to increase the number of knots in the model. Similar to all of the past examples, we will require that the new set of knots contains the old set–that is, $\{t_1, \ldots, t_k\} \subset \{s_1, \ldots, s_m\}$. This requirement ensures that our models are nested and thus can be compared in an ANOVA table.

2.4.2.3 B-Splines

In what we considered above, the spline models were comprised of polynomials with supports of the form $[t_j, \infty)$ for knots $t_1 < \ldots < t_k$. However, there are many different families of splines with different desirable properties. Some such families are the B-splines, Non-uniform rational B-splines (NURBS), box splines, Bézier splines, and many others. Here we will briefly consider the family of B-splines due to their simplicity and popularity. Note that in the spline literature, sometimes the knots are referred to as control points.

The ultimate goal of the of the B-splines is to construct a polynomial basis where the constituent polynomials have finite support. Specifically, for an interval $[a, b]$ and knots $a = t_1 < t_2 < \ldots < t_k < t_{k+1} = b$, the constant polynomials will have support on two knots such as $[t_1, t_2]$, the linear terms on three knots such as $[t_1, t_3]$, and so on up to degree p terms, which require $p + 2$ knots.

B-splines can be defined recursively starting with the constant, or degree 0, terms:

$$B_{j,0}(x) = 1_{x \in [t_j, t_{j+1}]},$$

which takes a value of 1 on the interval $[t_j, t_{j+1}]$ and is 0 elsewhere. From here, the higher order terms can be written as

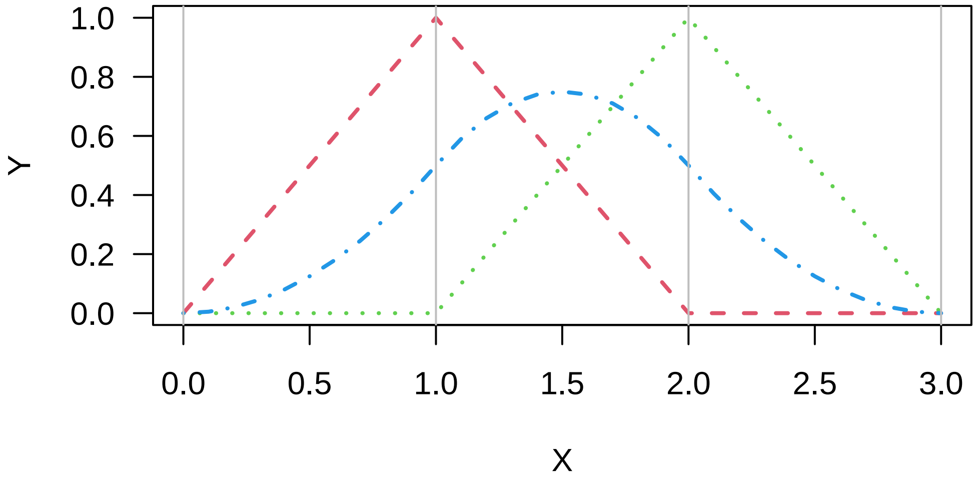
$$B_{j,i}(x) = \left(\frac{x - t_j}{t_{j+i} - t_j} \right) B_{j,i-1}(x) + \left(\frac{t_{j+i+1} - x}{t_{j+i+1} - t_{j+1}} \right) B_{j+1,i-1}(x).$$

For example, with knots $\{0, 1, 2, 3\}$, we have

$$\begin{array}{lll}
\text{Constant:} & B_{j,0} = 1_{x\in[j,j+1]} & j = 0,1,2 \\
\text{Linear:} & B_{j,1} = \begin{cases} (x-j), & x \in [j, j+1] \\ (j+2-x), & x \in [j+1, j+2] \end{cases} & j = 0,1 \\
\text{Quadratic:} & B_{j,2} = \begin{cases} x^2/2, & x \in [0,1] \\ (-2x^2+6x-3)/2, & x \in [1,2] \\ (3-x)^2/2, & x \in [2,3] \end{cases} & j = 0.
\end{array}$$

The linear and quadratic splines are displayed below.

```r
xx   = seq(0,3,0.1)
lin1 = c(xx[1:11],2-xx[12:21],rep(0,10))
lin2 = rev(lin1)
quad = c(
  xx[1:11]^2/2,(-2*xx[12:21]^2+6*xx[12:21]-3)/2,
  (3-xx[22:31])^2/2
)
plot(
  0,0,type='n',las=1,xlim=c(0,3),ylim=c(0,1),
  xlab="X",ylab="Y"
)
lines(xx,lin1,lty=2,col=2,lwd=2)
lines(xx,lin2,lty=3,col=3,lwd=2)
lines(xx,quad,lty=4,col=4,lwd=2)
abline(v=0:3,col='gray')
```



Just as we fit a spline model above, we could use the linear regression tools to fit a B-spline model of the form

$$y = \sum_{i=0}^{p} \sum_{j=1}^{k-i} \beta_{i,j} B_{j,i}(x).$$

Here, we require that $k > p$ as we will at least need knots $t_1, \dots, t_{p+1}$ for a pth degree polynomial. The total number of terms in the regression, which is number of parameters $\beta_{i,j}$ to estimate, is

$$k + (k-1) + \dots + (k-p).$$

2.4.3 Interacting Regressors

Thus far in this section, we have considered only polynomial models with a single regressor. However, it is certainly possible to fit a polynomial model for more than one regressor such as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{12} x_1 x_2 + \beta_{22} x_2^2 + \varepsilon.$$

Here we have linear and quadratic terms for x_1 and x_2 as well as an interaction term $\beta_{12} x_1 x_2$.

Fitting such models to the data follows from what we did for single variable polynomials. In this case, the number of interaction terms can grow quite large in practice. With k regressors, $x_1, \dots, x_k$, there will be $\binom{k}{p}$ interaction terms of degree p assuming $p < k$. This leads to the topic of Response Surface Methodology, which is a subtopic of the field of experimental design.

We will not consider this topic further in these notes. However, it is worth noting that in R, it is possible to fit a linear regression with interaction terms such as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_{12} x_1 x_2 + \beta_{13} x_1 x_3 + \beta_{23} x_2 x_3 + \beta_{123} x_1 x_2 x_3 + \varepsilon$$

with the simple syntax `lm( y~x1*x2*x3 )` where the symbol `*` replaces the usual `+` from before.

2.5 Influence and Leverage

The overall intuition for this section is that each observation does not have an equal influence on the estimation of $\hat{\beta}$. If a given observed regressor x lies far from the other observed values, it can have a strong effect on the least squares regression line. The goal of this section is to identify such points or subset of points that have a large influence on the regression.

If we use the R command `lm()` to fit a linear model to some data, then we can use the command `influence.measures()` to compute an array of diagnostic metrics for each observation to test its influence on the regression. The function `influence.measures()` computes DFBETAS, DFFITS, covariance ratios, Cook's distances and the diagonal elements of the so-called hat matrix. We will look at each of these in the following subsections.

Warning: You may notice that the word *heuristic* appears often in the following subsections when it comes to identifying observations with significant influence. Ultimately, these are rough guidelines based on the intuition of past statisticians and should not be taken as strict rules.

2.5.1 The Hat Matrix

The projection or "hat'' matrix, $P = X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$, directly measures the influence of one point on another.
This is because under the usual model assumptions, the fitted values have a covariance matrix $\mathrm{var}(\hat{Y}) = \sigma^2 P$. Hence, the i, jth entry in P is a measure of the covariance between the fitted values $\hat{Y}_i$ and $\hat{Y}_j$.

The function `influence.measures()` reports the diagonal entries of the matrix P. Heuristically, any entries that are much larger than the rest will have a strong influence on the regression. More precisely, we know that for a linear model

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p,$$

we have that $\mathrm{rank}(P) = p + 1$.
Hence, $\mathrm{trace}(P) = p + 1$ where the trace of a matrix is the sum of the diagonal entries. Thus, as P is an $n \times n$ matrix, we roughly expect the diagonal entries to be approximately $(p + 1)/n$.

Large deviations from this value should be investigated. For example, Montgomery, Peck, & Vining recommend looking at observations with $P_{i,i} > 2(p+1)/n$.

The ith diagonal entry $P_{i,i}$ is referred to as the leverage of the ith observation in Montgomery, Peck, & Vining. However, in R, leverage is $P_{i,i}/(1-P_{i,i})$. This sometimes referred to as the leverage factor.

2.5.2 Cook's D

Cook's D or distance computes the distance between the vector of estimated parameters on all n data points, $\hat{\beta}$, and the vector of estimated parameters on $n-1$ data points, $\hat{\beta}_{(i)}$ where the ith observation has been removed. Intuitively, if the ith observation has a lot of influence on the estimation of β, then the distance between $\hat{\beta}$ and $\hat{\beta}_{(i)}$ should be large.

For a linear model with $p+1$ parameters and a sample size of n observations, the usual form for Cook's D is

$$D_i = \frac{(\hat{\beta}_{(i)} - \hat{\beta})^{\mathrm{T}} X^{\mathrm{T}} X (\hat{\beta}_{(i)} - \hat{\beta})/(p+1)}{SS_{\text{res}}/(n-p-1)}.$$

This is very similar to the confidence ellipsoid from the last chapter. However, this is not an usual F statistic, so we do not compute a p-value as we have done before. Instead, some heuristics are used to determine what a large value is. Some authors suggest looking for D_i greater than 1 or greater than $4/n$. See Cook's Distance.

Cook's D can be written in different forms. One, in terms of the diagonal entries of P, is

$$D_i = \frac{s_i^2}{p+1}\left(\frac{P_{i,i}}{1-P_{i,i}}\right)$$

where s_i is the ith studentized residual. Another form of this measure compares the distance between the usual fitted values on all of the data $\hat{Y} = X\hat{\beta}$ and the fitted values based on all but the ith observation, $\hat{Y}_{(i)} = X\hat{\beta}_{(i)}$.
That is,

$$D_i = \frac{(\hat{Y}_{(i)} - \hat{Y})^{\mathrm{T}} (\hat{Y}_{(i)} - \hat{Y})/(p+1)}{SS_{\text{res}}/(n-p-1)}$$

Note that like $\hat{Y}$, the vector $\hat{Y}_{(i)} \in \mathbb{R}^n$.
The ith entry in the vector $\hat{Y}_{(i)}$ is the predicted value of y given x_i.

2.5.3 DFBETAS

The intuition behind DFBETAS is similar to that for Cook's D. In this case, we consider the normalized difference between $\hat{\beta}$ and $\hat{\beta}_{(i)}$. What results is an $n \times (p+1)$ matrix whose ith row is

$$\text{DFBETAS}_i = \frac{\hat{\beta} - \hat{\beta}_{(i)}}{\sqrt{(X^{\mathrm{T}}X)^{-1}_{i,i} SS_{\text{res}(i)}/(n-p-2)}} \in \mathbb{R}^{p+1}$$

where $SS_{\text{res}(i)}$ is the sum of the squared residuals for the model fit after removing the ith data point and where $(X^{\mathrm{T}}X)^{-1}_{i,i}$ is the ith diagonal entry of the matrix $(X^{\mathrm{T}}X)^{-1}$. The recommended heuristic is to consider the ith observation as an influential point if the i,jth entry of DFBETAS has a magnitude greater than $2/\sqrt{n}$.

2.5.4 DFFITS

The DFFITS value is very similar to the previously discussed DFBETAS. In this case, we are concerned with by how much the fitted values change when the ith observation is removed. Explicitly,

$$\text{DFFIT} = \frac{\hat{Y} - \hat{Y}_{(i)}}{\sqrt{(X^{\mathrm{T}}X)^{-1}_{i,i} SS_{\text{res}(i)}/(n-p-2)}} \in \mathbb{R}^{n}.$$

The claim is that DFFIT is effected by both leverage and prediction error. The heuristic is to investigate any observation with DFFIT greater in magnitude than $2\sqrt{(p+1)/n}$.

2.5.5 Covariance Ratios

The covariance ratio whether the precision of the model increases or decreases when the ith observation is included. This measure is based on the idea that a small value for $\det\left[(X^{\mathrm{T}}X)^{-1}SS_{\text{res}}\right]$ indicates high precision in our model. Hence, the covariance ratio considers a ratio of determinants

$$\text{covratio}_i = \frac{\det\left[(X^{\mathrm{T}}_{(i)}X_{(i)})^{-1}SS_{\text{res}(i)}/(n-p-2)\right]}{\det\left[(X^{\mathrm{T}}X)^{-1}SS_{\text{res}}/(n-p-1)\right]} = \\ = \left(\frac{SS_{\text{res}(i)}/(n-p-2)}{SS_{\text{res}}/(n-p-1)}\right)^{p+1}\left(\frac{1}{1-P_{i,i}}\right).$$

If the value is greater than 1, then inclusion of the ith point has increased the model's precision. If it is less than 1, then the precision has decreased. The suggested heuristic threshold for this measure of influence is when the value is greater than $1+3(p+1)/n$ or less than $1-3(p+1)/n$. Though, it is noted that this only is valid for large enough sample sizes.

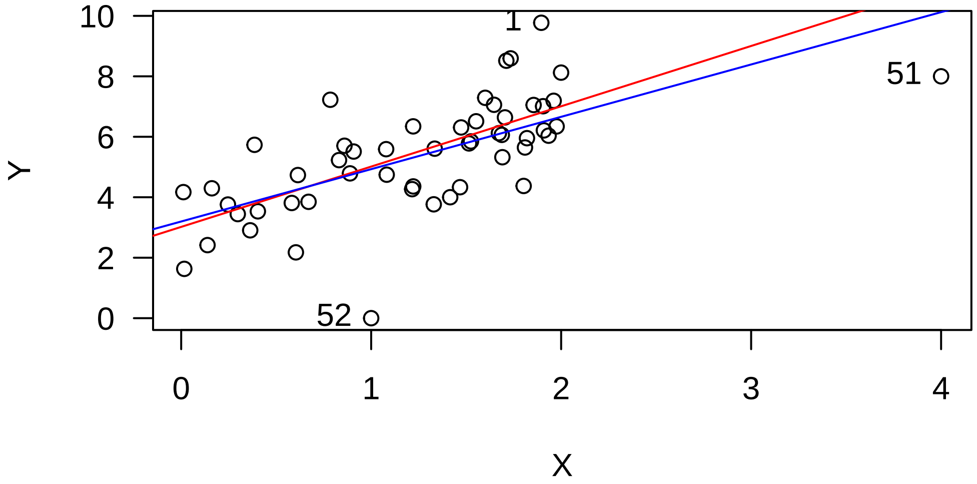
2.5.6 Influence Measures: An Example

To test these different measures, we create a dataset of $n = 50$ observations from the model

$$y = 3 + 2x + \varepsilon$$

where $\varepsilon \sim \mathcal{N}(0, 1)$ and $x \in [0, 2]$. To this dataset, we add 2 anomalous points at $(4, 8)$ and at $(1, 0)$. Thus, we fit a simple linear regression to the original 50 data points and also to the new set of 52 data points resulting in the red and blue lines, respectively.

```
set.seed(216)
# Generate some data and fit a linear regression
xx  <- runif(50,0,2);
yy  <- 3 + 2*xx + rnorm(50,0,1);
md0 <- lm( yy~xx )
# Add two influential points and fit a new regression
xx  <- c(xx,4,1);
yy  <- c(yy,8,0);
md1 <- lm( yy~xx )
# Plot the data and two linear models
plot(xx,yy,las=1,xlab="X",ylab="Y")
abline(md0,col='red')
abline(md1,col='blue')
text(
  x=c(4,1,xx[1]),y=c(8,0,yy[1]),labels = c("51","52","1"),pos = 2
)
```



We can use the R function `influence.measures()` to compute a matrix containing the DFBETAS, DFFITS, covariance ratios, Cook's D, and leverage for each data point. Applying the recommended thresholds in the previous sections results in the following extreme points, which are labelled in the above figure:

```
influence.measures(md1)
```

```
Influence measures of
	 lm(formula = yy ~ xx) :

     dfb.1_    dfb.xx     dffit cov.r   cook.d    hat inf
1  -1.02e-01  0.317757  0.476552 0.846 1.03e-01 0.0346   *
2   1.58e-01 -0.133249  0.158687 1.098 1.28e-02 0.0652
3  -1.53e-02  0.155431  0.289707 0.949 4.03e-02 0.0270
4   1.27e-02 -0.053722 -0.087782 1.064 3.91e-03 0.0307
5   6.08e-02 -0.036088  0.069801 1.061 2.48e-03 0.0262
6   6.90e-02 -0.037061  0.083446 1.055 3.53e-03 0.0240
7  -5.31e-02  0.038346 -0.055547 1.077 1.57e-03 0.0367
8  -5.16e-02  0.005189 -0.095960 1.042 4.65e-03 0.0193
9   3.75e-02 -0.013404  0.054260 1.057 1.50e-03 0.0205
10  3.50e-02 -0.153273 -0.252471 0.990 3.12e-02 0.0305
11 -1.52e-01  0.120320 -0.153509 1.076 1.19e-02 0.0499
12  8.34e-03  0.025927  0.068098 1.056 2.36e-03 0.0225
```

```
13 -1.56e-02  0.054631  0.085064 1.067 3.68e-03 0.0327
14  9.02e-03 -0.027047 -0.040094 1.077 8.19e-04 0.0353
15 -6.29e-02  0.157343  0.218130 1.036 2.37e-02 0.0401
16  2.06e-01 -0.178913  0.205853 1.108 2.14e-02 0.0786
17  3.07e-01 -0.241294  0.310959 1.014 4.75e-02 0.0483
18  7.82e-03 -0.031485 -0.050843 1.071 1.32e-03 0.0312
19  6.17e-02 -0.043728  0.065015 1.074 2.15e-03 0.0351
20 -1.57e-02  0.047578  0.070775 1.073 2.55e-03 0.0351
21 -3.96e-05  0.001572  0.003130 1.069 5.00e-06 0.0257
22 -3.36e-02 -0.041662 -0.148244 1.020 1.10e-02 0.0209
23 -2.25e-02  0.166146  0.299265 0.946 4.29e-02 0.0278
24  9.87e-03  0.064128  0.148393 1.028 1.10e-02 0.0236
25 -2.78e-01  0.198487 -0.292542 0.984 4.17e-02 0.0356
26 -4.68e-02  0.037911 -0.047125 1.100 1.13e-03 0.0545
27  1.29e-02  0.016904  0.058694 1.057 1.75e-03 0.0210
28  1.73e-03  0.055587  0.116249 1.045 6.82e-03 0.0249
29  5.59e-02 -0.005676  0.103967 1.038 5.45e-03 0.0193
30  3.01e-03 -0.049070 -0.094617 1.055 4.54e-03 0.0263
31  1.09e-02 -0.028400 -0.039997 1.081 8.16e-04 0.0388
32  3.00e-01 -0.186937  0.336339 0.917 5.34e-02 0.0278
33 -5.67e-02  0.006466 -0.104152 1.038 5.47e-03 0.0193
34 -7.21e-04 -0.001449 -0.004288 1.064 9.38e-06 0.0217
35  1.74e-02 -0.049266 -0.071575 1.075 2.61e-03 0.0365
36  3.98e-03  0.001045  0.010009 1.062 5.11e-05 0.0194
37 -2.31e-02  0.008147 -0.033508 1.061 5.72e-04 0.0204
38 -6.27e-02  0.042836 -0.067087 1.070 2.29e-03 0.0325
39 -2.67e-03  0.031091  0.058627 1.064 1.75e-03 0.0268
40 -3.47e-01  0.301114 -0.346781 1.067 5.96e-02 0.0782
41 -2.01e-01  0.170219 -0.201503 1.091 2.05e-02 0.0671
42  5.05e-03 -0.002790  0.006023 1.067 1.85e-05 0.0245
43 -7.14e-02 -0.017213 -0.176762 0.997 1.54e-02 0.0194
44 -5.01e-02 -0.036746 -0.171232 1.003 1.45e-02 0.0202
45  1.50e-04  0.000339  0.000969 1.065 4.79e-07 0.0219
46 -2.25e-02  0.060242  0.085784 1.074 3.74e-03 0.0379
47  1.88e-04 -0.003403 -0.006597 1.069 2.22e-05 0.0262
48  1.01e-01 -0.058014  0.118145 1.045 7.04e-03 0.0253
49  2.45e-02 -0.020183  0.024645 1.105 3.10e-04 0.0584
50 -5.75e-02  0.044956 -0.058414 1.090 1.74e-03 0.0472
51  8.89e-01 -1.195032 -1.234540 1.307 7.26e-01 0.3053   *
52 -4.63e-01  0.209038 -0.608400 0.594 1.41e-01 0.0218   *
```

Note that point 1 is just part of the randomly generated data while points 51 and 52 were purposefully added to be anomalous. Hence, just because an observation is beyond one of these thresholds does not necessarily imply that it lies outside of the model.

2.6 Weighted Least Squares

A key assumption of the Gauss-Markov theorem is that $\varepsilon_i = \sigma^2$ for all $i = 1, \dots, n$. What happens when $\varepsilon_i = \sigma_i^2$–i.e. when the variance can differ for each observation? Normalizing the errors ε_i can be done by

$$\frac{\varepsilon_i}{\sigma_i} = \frac{1}{\sigma_{i,i}}(Y_i - X_{i,\cdot}\beta) \sim \mathcal{N}\left(0, 1\right).$$

In Chapter 1, we computed the least squares estimator as the vector $\hat{\beta}$ such that

$$\hat{\beta} = \underset{\tilde{\beta} \in \mathbb{R}^{p+1}}{\arg\min} \sum_{i=1}^{n} (Y_i - X_{i,\cdot}\tilde{\beta})^2$$

Now, we will solve the slighly modified equation

$$\hat{\beta} = \underset{\tilde{\beta} \in \mathbb{R}^{p+1}}{\arg\min} \sum_{i=1}^{n} \frac{(Y_i - X_{i,\cdot}\tilde{\beta})^2}{\sigma_i^2}.$$

In this setting, dividing by σ_i^2 is the “weight’ ’ that gives this method the name weighted least squares.

Proceeding as in chapter 1, we take a derivative with respect to the jth $\tilde{\beta}_j$ to get

$$\begin{aligned}
\frac{\partial}{\partial \tilde{\beta}_j} \sum_{i=1}^{n} \frac{(Y_i - X_{i,\cdot}\tilde{\beta})^2}{\sigma_i^2} &= 2 \sum_{i=1}^{n} \frac{(Y_i - X_{i,\cdot}\tilde{\beta})}{\sigma_i^2} X_{i,j} \\
&= 2 \sum_{i=1}^{n} \frac{Y_i X_{i,j}}{\sigma_i^2} - 2 \sum_{i=1}^{n} \frac{X_{i,\cdot}\tilde{\beta} X_{i,j}}{\sigma_i^2} \\
&= 2 \sum_{i=1}^{n} Y_i' X_{i,j}' - 2 \sum_{i=1}^{n} X_{i,\cdot}' \tilde{\beta} X_{i,j}'
\end{aligned}$$

where $Y_i' = Y_i / \sigma_i$ and $X_{i,j}' = X_{i,j} / \sigma_i$. Hence, the least squares estimator is as before

$$\begin{aligned}
\hat{\beta} &= (X'^{\mathrm{T}} X')^{-1} X'^{\mathrm{T}} Y' \\
&= (X^{\mathrm{T}} W X)^{-1} X^{\mathrm{T}} W Y
\end{aligned}$$

where $W \in \mathbb{R}^{n \times n}$ is the diagonal matrix with entries $W_{i,i} = \sigma_i^2$.

In practise, we do not know the values for σ_i^2. Methods to find a good matrix of weights W exist such as Iteratively Reweighted Least Squares which is equivalent to finding the estimator

$$\hat{\beta} = \underset{\tilde{\beta} \in \mathbb{R}^{p+1}}{\arg\min} \sum_{i=1}^{n} |Y_i - X_{i,\cdot}\tilde{\beta}|^q$$

for some $q \in [1, \infty)$.

References