



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich

# **Applied Statistical Regression**

**AS 2015**

**Dr. Marcel Dettling**

Institute for Data Analysis and Process Design

Zurich University of Applied Sciences

CH-8401 Winterthur



<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	WHAT IS REGRESSION?	1
1.2	REGRESSION MATHEMATICS	2
1.3	GOALS WITH REGRESSION	3
<b>2</b>	<b>SIMPLE REGRESSION</b>	<b>5</b>
2.1	EXAMPLE: ZURICH AIRPORT DATA	5
2.2	SCATTERPLOT SMOOTHING	6
2.2.1	RUNNING MEAN ESTIMATION	6
2.2.2	GAUSSIAN KERNEL SMOOTHING	9
2.2.3	THE LOESS SMOOTHER	10
2.3	SIMPLE LINEAR REGRESSION	12
2.3.1	THE MODEL	12
2.3.2	THE LEAST SQUARES ALGORITHM	13
2.3.3	ASSUMPTIONS FOR OLS ESTIMATION	16
2.3.4	RESIDUAL PLOTS	16
2.3.5	HISTORY OF LEAST SQUARES	18
2.3.6	MATHEMATICAL OPTIMALITY OF OLS	19
2.3.7	ESTIMATING THE ERROR VARIANCE	20
2.4	INFERENCE	21
2.4.1	THE COEFFICIENT OF DETERMINATION	21
2.4.2	CONFIDENCE INTERVAL FOR THE SLOPE	22
2.4.3	TESTING THE SLOPE	23
2.4.4	TESTING THE INTERCEPT	24
2.5	PREDICTION	24
2.5.1	CONFIDENCE INTERVAL FOR THE REGRESSION LINE	25
2.5.2	PREDICTION INTERVAL FOR FUTURE DATA POINTS	25
2.5.3	VISUALIZING CONFIDENCE AND PREDICTION INTERVALS	26
2.6	MODEL EXTENSIONS	27
2.6.1	EXAMPLE: AUTOMOBILE BRAKING DISTANCE	27
2.6.2	CURVILINEAR REGRESSION	29
2.6.3	EXAMPLE: INFANT MORTALITY	30
2.6.4	THE LOG-LOG MODEL	32
2.6.5	THE LOGGED RESPONSE MODEL	35
2.6.6	WHEN AND HOW TO LOG-TRANSFORM	37
2.6.7	FINAL CONSIDERATIONS	38
<b>3</b>	<b>MULTIPLE LINEAR REGRESSION</b>	<b>41</b>
3.1	EXAMPLE: AIR POLLUTION AND MORTALITY	41
3.2	PREPARING THE DATA	42
3.2.1	MARGINAL PLOTS	42
3.2.2	VARIABLE TRANSFORMATIONS	43
3.3	MODEL AND ESTIMATION	44
3.3.1	NOTATION	46
3.3.2	OLS: METHOD & IDENTIFIABILITY	47
3.3.3	PROPERTIES OF THE ESTIMATES	50
3.4	INFERENCE	52

3.4.1	THE COEFFICIENT OF DETERMINATION	52
3.4.2	CONFIDENCE INTERVALS FOR THE COEFFICIENTS	53
3.4.3	INDIVIDUAL HYPOTHESIS TEST	54
3.4.4	COMPARING HIERARCHICAL MODELS	55
3.4.5	GLOBAL F-TEST	57
<b>3.5</b>	<b>PREDICTION</b>	<b>58</b>
<b>3.6</b>	<b>CATEGORICAL PREDICTORS</b>	<b>59</b>
3.6.1	EXAMPLE WITH 1 CATEGORICAL PREDICTOR	59
3.6.2	MIX OF CATEGORICAL AND CONTINUOUS PREDICTORS	61
3.6.3	INTERACTION TERMS	63
3.6.4	CATEGORICAL INPUT WITH MORE THAN TWO LEVELS	65
3.6.5	CATEGORIZING QUANTITATIVE PREDICTORS	68
<b>3.7</b>	<b>MODEL DIAGNOSTICS</b>	<b>69</b>
3.7.1	WHAT DO WE NEED TO CHECK FOR, AND HOW?	69
3.7.2	CHECKING ERROR ASSUMPTIONS	70
3.7.3	INFLUENCE DIAGNOSTICS	77
3.7.4	EXAMPLE: MORTALITY DATASET	81
3.7.5	FURTHER RESIDUAL PLOTS	81
3.7.6	DEALING WITH CORRELATED ERRORS	86
<b>3.8</b>	<b>MULTICOLLINEARITY</b>	<b>91</b>
3.8.1	IDENTIFYING MULTICOLLINEARITY	92
3.8.2	DEALING WITH MULTICOLLINEARITY	94
<b>3.9</b>	<b>VARIABLE SELECTION</b>	<b>96</b>
3.9.1	WHY VARIABLE SELECTION?	96
3.9.2	BACKWARD ELIMINATION WITH P-VALUES	97
3.9.3	FORWARD SELECTION WITH P-VALUES	99
3.9.4	AIC/BIC	100
3.9.5	USING R FUNCTION <code>STEP ( )</code>	101
3.9.6	ALL SUBSETS SELECTION	104
3.9.7	FINAL REMARKS ABOUT VARIABLE SELECTION	105
<b>3.10</b>	<b>CROSS VALIDATION</b>	<b>105</b>
<b>3.11</b>	<b>MODELLING STRATEGIES</b>	<b>108</b>
<b>3.12</b>	<b>SIGNIFICANCE VS. RELEVANCE</b>	<b>110</b>

# 1 Introduction


## 1.1 What is Regression?

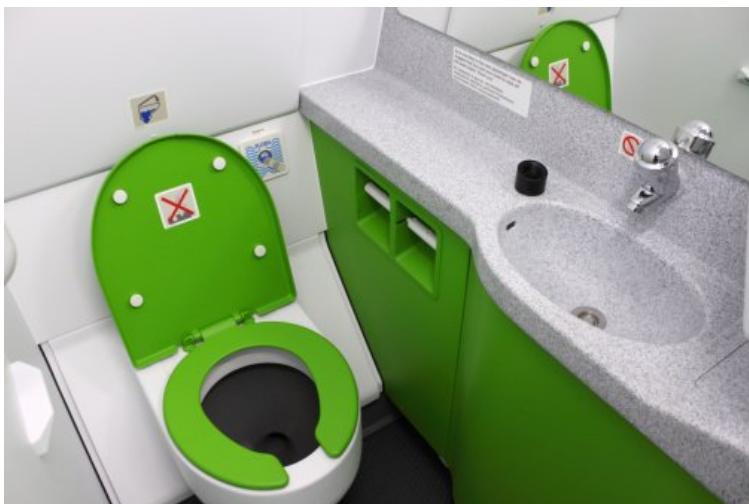
Regression analysis can be seen as the duct tape of statistics. The reason why it is widely used is because it may yield the answer to an everyday question, namely how a target value of special interest depends on several other factors or causes. Examples are numerous and include:

- how fertilizer and soil quality affects the growth of plants
- how size, location, furnishment and age affect apartment rents
- how age, sex, experience and nationality affect car insurance premiums

In all quantitative settings, regression techniques can provide an answer to these questions. They describe the relation between some *explanatory* or *predictor variables* and a variable of special interest, called the *response* or *target variable*. Regression techniques are of high practical importance, and probably the most widely used statistical methodology.

### Example

In an applied research project at ZHAW, we tried to understand and manage the fresh water consumption on board of  planes. Fresh water is mostly used in the toilet. Minimizing the carried amount was identified as important, because this reduces the weight of the airplane, and thereby fuel consumption and cost. The project goal was to relate the consumption on the *number of passengers* and *flight duration*, but also on less obvious parameters such as *daytime* and *destination*. Furthermore, it was required to quantify a well-calculated reserve, to set up a simple prediction scheme and to perform operations management on the filling of the tank.



## 1.2 Regression Mathematics

In the Edelweiss Air example, we can identify the fresh water consumption as the target value and denote it as the *response variable*  $y$ . Among the explanatory causes or *predictors* are *number of passengers*, and *flight duration*, plus a few more. These are denoted with  $x_1, x_2, \dots, x_p$ , assuming that there are  $p$  predictors. The goal is linking the target to the predictors, which could happen with this model:

$$y = f(x_1, x_2, \dots, x_p) + E$$

The target value is obtained as the sum of some function  $f(\cdot)$  applied on the predictors, plus an error term  $E$ . Why the error? In practice, it is highly unlikely that  $f(x_1, x_2, \dots, x_p)$  yields an all-case perfect explanation of the fresh water consumption. The error is there to catch the imperfection and summarizes the remaining variation in the response. It is assumed to be *random* and can neither be controlled or predicted. On the other hand,  $f(x_1, x_2, \dots, x_p)$  is called the *systematic* or *deterministic* part of the regression equation.

The task is thus to learn about the function  $f(\cdot)$ . In full generality, without any restrictions, this is a very difficult problem: function space is infinite-dimensional, thus there are just too many options such that we could come to a unique solution based on just a few dozens of observations. It has proven practical to be very restrictive with the form of functions  $f(\cdot)$  that are considered, i.e. normally, a linear model is assumed:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p + E$$

This setup is called *linear modeling*. It boils down to determine some parameters  $\beta_0, \beta_1, \beta_2, \dots, \beta_p$  from observed data points, a task we call *estimation* in statistics. Please note that this is mathematically much simpler than finding  $f(\cdot)$  without imposing any conditions.

One might of course fear that the limitation to linear modeling is too restrictive. However, practice proves this not to be the case, with the main reason being that only the parameters, but not the predictors need to enter linearly. In particular, the following structure is still a linear model:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 (x_1)^2 + \beta_3 \log(x_2) + \beta_4 x_1 x_2 + E$$

For such models, it is possible to estimate the parameters from a relatively low number of data points with the least squares algorithm that will be presented shortly. Using variable transformations as outlined above, linear modeling becomes a very rich and flexible tool. Truly non-linear models are rarely absolutely necessary in practice and most often arise from a theory about the relation between the variables rather than from necessity in an empirical investigation. Of course, the right variable transformations need to be found, but using some simple guidelines and visual displays this is a manageable task, as we will see later.

## 1.3 Goals with Regression

There are a variety of reasons to perform regression analysis. The two most prominent ones are:

*Understanding on the predictor-response relation, i.e. doing inference*

In the mortality example outlined in the chapter about multiple linear regression, one is interested in testing whether air pollution affects mortality, under control of potentially confounding factors such as weather and the socio-demographic variables. We will see that regression, i.e. linear modeling offers tools to answer whether air pollution harms in statistically significant way. Drawing conclusions on true causal relationship, however, is a somewhat different matter.

*Target value prediction as a function of new explanatory variables*

In the fresh water consumption example from above, an airplane crew or the ground staff may want to determine the amount of water that is necessary for a particular flight, given its parameters. Regression analysis, i.e. linear modeling incorporates the previous experience in that matter and yields a quantitative prediction. It also results in prediction intervals which give a hint on the uncertainty such a prediction has. In practice, the latter might be very useful for the amount of reserve water that needs to be loaded.





## 2 Simple Regression

The term simple regression means that there is a response and only one single predictor variable. This has several practical advantages: we can easily visualize the two variables and their relation in a scatterplot, and the involved mathematics is quite a bit easier. We will first address non-parametric curve fitting, also known as *smoothing*. Later, we proceed to linear modeling which in its most basic form amounts to laying a straight line into the scatterplot. But as we will see, linear modeling can also be used for fitting curves.

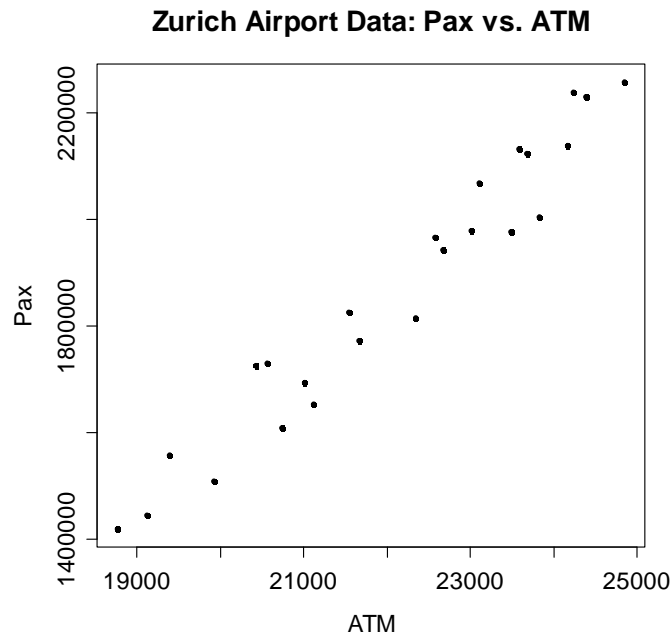
### 2.1 Example: Zurich Airport Data

The example we consider for developing the methodology is from Zurich Airport. Every month, the number of air traffic movements as well as the number of passengers is reported. The two variables are named *ATM* and *Pax*, with the former being the predictor, and the latter being the response. The goal is to predict passenger figures for future months based on the flight plan, and to quantify the uncertainty in these forecasts. The data are publicly accessible here: <http://www.flughafen-zuerich.ch/desktopdefault.aspx/tabid-612/>



We could display the figures in a table, but a much better solution is to visualize them in a scatterplot, as shown on the next page. As the first step, we need to import the data into R. Assuming that the data exist in form of an Excel spreadsheet; we recommend exporting them in a comma- or tab-separated text file. In R, we can then use the function `read.table()`, respectively one of the tailored versions like `read.csv()` (for comma separation) or `read.delim()` (for tab separation), for importing the data. This will result in a so-called *data frame*, the structure which is most suitable for performing regression analysis in R. In our example, the Zurich Airport Data are stored in a data frame named `unique2010`. For producing a scatterplot, we can employ the generic `plot()` function, where several additional arguments can be set.

```
> plot(Pax ~ ATM, data=unique2010)
> title("Zurich Airport Data: Pax vs. ATM")
```



The question is how the systematic relation between  $Pax$  and  $ATM$  can be described. We could imagine that an arbitrary, smooth function  $f(\cdot)$  that fits well to the data points, without following them too closely, is a good solution. Another viable and popular option would be to use a straight line for capturing the relation.

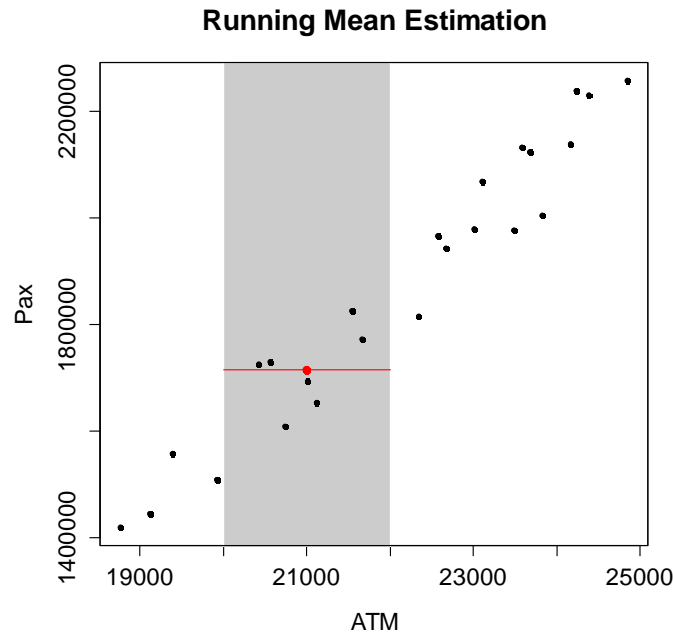
The advantages of smoothing are its flexibility and the fact that less assumptions are made on the form of the relation. This comes with the price that the functional form generally remains unknown, and that we can overfit, i.e. adapt too much to the data. With linear modeling, we have the benefit that formal inference on the relation is possible and that the efficiency is better, i.e. less data are required for a good estimate. The downside of the parametric approach is that it is only sensible if the relation is linear, and that it might falsely imply causality.

## 2.2 Scatterplot Smoothing

We start out with the smoothing approach. The goal here is to visualize the relation between  $Pax$  and  $ATM$ , but we are not after the functional form of  $f(\cdot)$ . Because there is no parametric function that describes the response-predictor relation, smoothing is also known as *non-parametric regression analysis*.

### 2.2.1 Running Mean Estimation

A simple yet intuitive smoother is the *running mean*. In colloquial language it involves taking a fixed width window on the  $x$ -axis, and compute the mean over all the within-window data point's  $y$ -values. That value then is the estimate for the function value at the window center.



The above illustration shows the running mean estimate at  $ATM = 21'000$  with window width  $\lambda = 2000$ . The  $y$ -values of all the observations which fall into the grey highlighted region are averaged, which yields a value of 1'716'195. The respective point is marked by the red symbol in the plot. The grey window is the slid over the  $x$ -axis which defines the running mean smooth. In mathematical notation, the running mean estimate for the unknown function  $f(\cdot)$  denoted as  $\hat{f}_\lambda(\cdot)$ , is defined as follows:

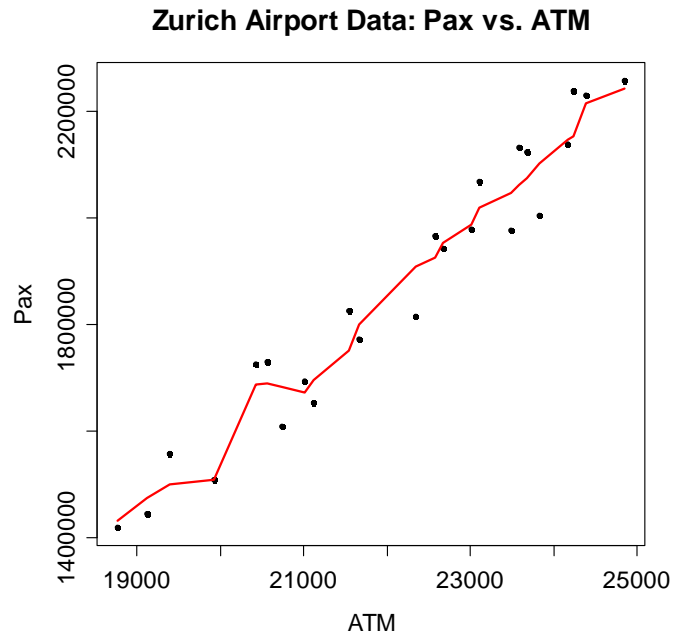
$$\hat{f}_\lambda(x) = \frac{\sum_{i=1}^n w_i y_i}{\sum_{i=1}^n w_i}, \text{ with weights } w_i = \begin{cases} 1 & \text{if } |x - x_i| \leq \lambda/2 \\ 0 & \text{else} \end{cases}.$$

The parameter  $\lambda$  is the window width and controls the amount of smoothing. Small values mean close adaptation to the data, while large values indicate averaging over more data points and thus a smoother solution. In R, running mean smoothing can be done with function `ksmooth()`:

```
> fit <- ksmooth(unique2010$ATM, unique2010$Pax, kernel="box",
  bandwidth=1000, n.points=24, x.points=
  unique2010$ATM)
```

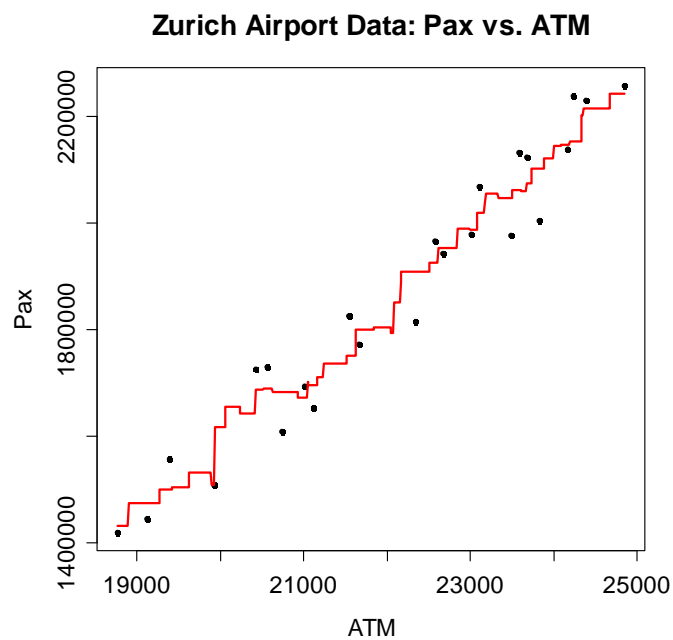
The argument `kernel="box"` tells R to use a rectangular kernel, and the `bandwidth=1000` argument steers the window width. Finally, `n.points` and `x.points` regulate at how many and which  $x$ -values the estimate is computed. We chose to do that for the observed  $ATM$  values. The solution can be plotted:

```
> plot(Pax ~ ATM, data=unique2010, main="...")
> title("Zurich Airport Data: Pax vs. ATM")
> lines(fit, col="red", lwd=2)
```



Perhaps a little more smoothing is required here, because we would hardly believe in a (systematic) relation that shows a decrease in passengers if the number of air traffic movements raises from 20'500 to 21'000. However, we leave this as an exercise to the reader. To point out an important drawback of running mean estimation, we increase the number of evaluation points to 1000 that uniformly cover the range of *ATM* and then plot the result:

```
> fit <- ksmooth(unique2010$ATM,unique2010$Pax, kernel="box",  
                bandwidth=1000, n.points=1000)
```



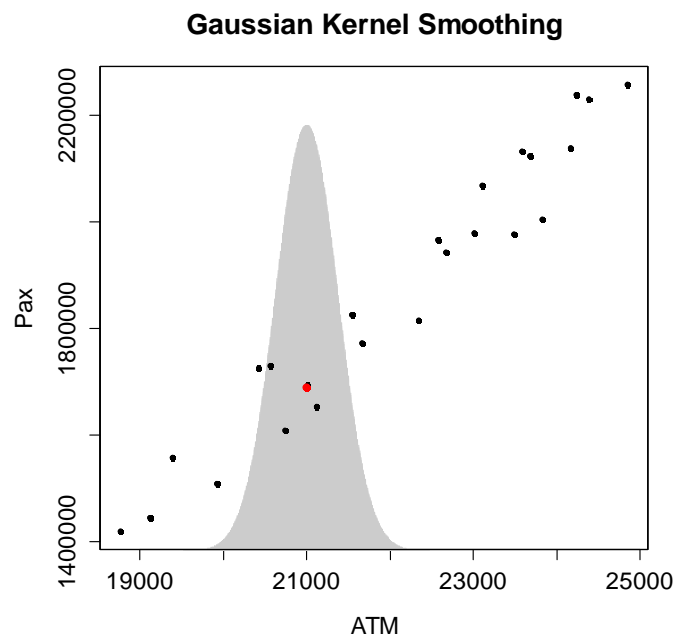
We obtain a function that is not smooth at all, but this is not a surprise. By construction, due to the rectangular kernel, data points drop out of the running mean computation abruptly, and hence we have the jumps.

## 2.2.2 Gaussian Kernel Smoothing

We can fix the above problem by using a kernel with infinite support, i.e. none of the weights should be exactly zero. An obvious choice for a weighting scheme that puts emphasis on nearby data points, down weights distant observations and is never zero is the probability density function of the Gaussian distribution. The definition is:

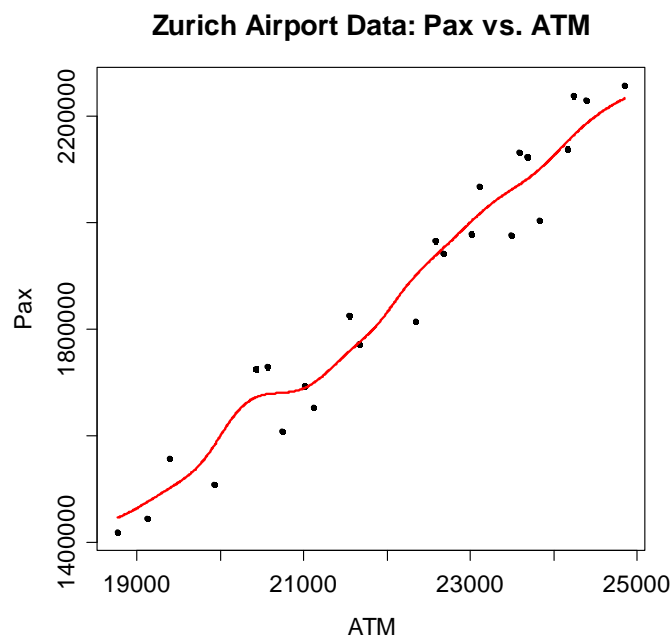
$$\hat{f}_\lambda(x) = \frac{\sum_{i=1}^n w_i y_i}{\sum_{i=1}^n w_i} \quad \text{with weights } w_i = \exp\left(-\frac{(x - x_i)^2}{\lambda}\right).$$

Thus, there is no longer a window that determines which data points take part in the running mean computation. But we use a Gaussian bell curve that determines the weights for the observations – no matter where, always all of them are used to compute the estimate. As we can easily imagine, this solves the issue with the data points that are lost abruptly, and the result is a smooth function



Above, computing a Gaussian Kernel Smoother is illustrated. At  $x = 21'000$ , a weighted average over all data points'  $y$ -values is taken. The weights come from the grey Gaussian bell curve. Its width is such that it corresponds to  $\text{bandwidth} = 1000$  in R, which means that the 25%-quantile of the distribution is at  $-0.25 \cdot 1'000 = -250$  and the 75%-quantile is at  $250$ . Fitting in R is convenient, i.e.:

```
> fit <- ksmooth(unique2010$ATM, unique2010$Pax,
                kernel="normal", n.points=1000,
                bandwidth=1000)
```



While visually, the solution may look more or less reasonable here, a closer inspection suggests that it is rather sensitive to outliers. Moreover, there is a severe boundary effect associated with both the running mean and the Gaussian kernel estimator. Because near the boundaries, we do not observe a full window, we have a bias. At the lower end of the  $x$ -range, the smoother overestimates, while at the upper end of the range, it underestimates.

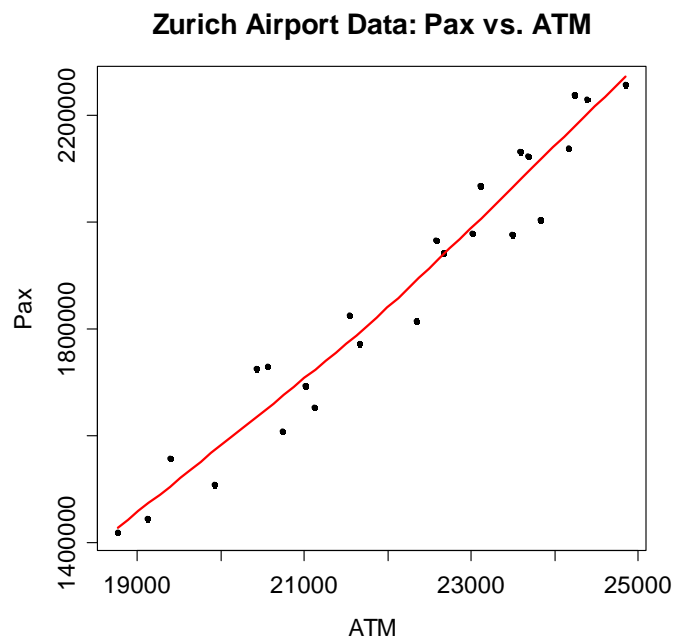
### 2.2.3 The LOESS Smoother

There is a wealth of literature that suggests improvements on kernel smoothing. However, with this scriptum, we will not further embark in that topic. But we present the *LOESS smoother*: it is a robust procedure that has nicer mathematical properties than the kernel smoothers, and that should be preferred in practice. LOESS is based on local parametric regressions: for obtaining the estimate at  $x$ , linear or polynomial models are fitted using data points in a neighborhood of  $x$ , weighted by their distance from  $x$ . The type of models used (linear or polynomial), the size of the neighborhood and also the type of fitting algorithm (least squares or robust) can be controlled in R. We do here without giving any theoretical details about the LOESS estimator. This is beyond the scope of our course, and it also requires intimate knowledge of linear modeling, which we do not yet have. However, as we will encounter LOESS smoothers throughout our studies in linear modeling, and it is a handy tool for visualizing the relation between two variables, we provide the necessary R commands:

```
> smoo <- loess.smooth(unique2010$ATM, unique2010$Pax)
```

For the `loess.smooth()` function, we need to specify the  $x$  and  $y$  variables. There are some further adjustments that can be made, but this is rarely necessary, because the default settings usually yield good results. Argument `span` controls the amount of smoothing and is set to  $2/3$ . Per default, we have `degree=1` which means local linear fitting, setting this to 2 means more flexibility through local polynomial fitting. Finally, argument `family` is set to `"symmetric"`, thus robust fitting is applied. A least squares fitting routine can be invoked by using `"gaussian"`. Lastly, we can control the number of points at which the smoother is evaluated. Mostly, the default of `evaluation=50` is fine, though it may sometimes be required to increase that number for relations with high curvature. We leave it to the reader to experiment with those settings and focus on displaying the result.

```
> plot(Pax ~ ATM, data=unique2010, main="...")
> lines(smoo, col="red", lwd=2)
```



We observe that the LOESS fit is almost, but not exactly a straight line, as it shows some progressive increase. Surely, when comparing to the Running Mean and the Gaussian Kernel Smoother, this is the most trustworthy result so far.

## 2.3 Simple Linear Regression

Instead of the non-parametric smoothing approaches, we will now turn our attention to linear modeling in the case where there is a response variable  $y$  and only one single predictor  $x$ . This problem is known as *simple linear regression*.

### 2.3.1 The Model

In our example, it seems logical that the more air traffic movements we have, the more passengers there are – at least on average. Also, it seems plausible that the systematic relation is well represented by a straight line. It is of the form:

$$Pass = \beta_0 + \beta_1 \cdot ATM, \text{ respectively } f(x) = \beta_0 + \beta_1 x$$

While this is the mathematically simplest way of describing the relation, it proves itself as very useful in many applications. And as we will see later, just some slight modifications to this concept render it to a very powerful tool when it comes to describing predictor-response relations. The two parameters  $\beta_0, \beta_1$  are called *intercept* and *slope*. The former is the expected value of  $y$  when  $x=0$ , and the latter describes the increase in  $y$  when  $x$  increases by 1 unit.

We now bring the data into play. It is obvious from the scatterplot that there is no straight line that runs through all the data points. It may describe the systematic relation well, but there is scatter around it, due to various reasons. We attribute these to randomness, and thus enhance the model equation by the error term:

$$y_i = \beta_0 + \beta_1 x_i + E_i, \text{ for all } i = 1, \dots, n.$$

The index  $i$  stands for the observations, of which there are  $n$  in total. In our example, we have  $n = 24$ . The interpretation of the above equation is as follows:

$y_i$  is the response or target variable of the  $i^{\text{th}}$  observation. In our example, this is the passenger number in the  $i^{\text{th}}$  month. Note that the response is a random variable, as it is the sum of a systematic and a random part.

$x_i$  is the explanatory or predictor variable, i.e. the number of air traffic movements in the  $i^{\text{th}}$  month. The predictor is treated as a fixed, deterministic value and has no randomness.

$\beta_0, \beta_1$  are unknown parameters, and are called *regression coefficients*. These are to be estimated by using the data points which are available.  $\beta_0$  is called *intercept*, whereas  $\beta_1$  is the *slope*. The latter indicates by how much the response changes, if the  $x$ -value is increased by 1 unit.

$E_i$  is the *error term*. It is a random variable, or more precisely, the random difference between the observed value  $y_i$  (which is seen as the realization of a random variable) and the model value fitted by the regression.

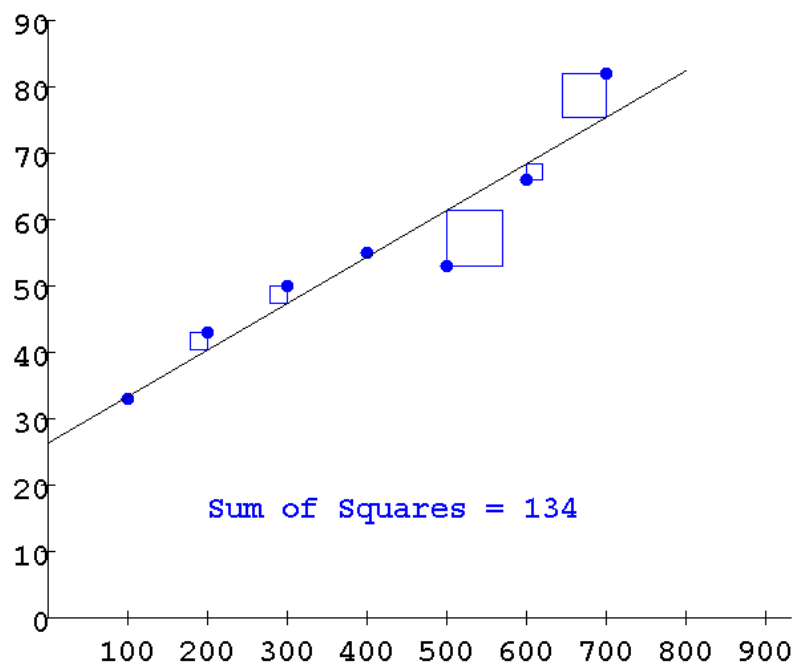


### 2.3.2 The Least Squares Algorithm

The goal in simple linear regression is to lay a straight line through the data points. If we did this by eyeballing, the solution between different persons would perhaps be similar, but not identical. It is clear that we cannot leave any arbitrariness for the regression line. Thus, we need a clear definition for the *best fitting line*, as well as an algorithm that unveils it.

*Our paradigm for linear modeling is to determine the regression line such that the sum of squared residuals is minimal!*

There are a number of reasons for this paradigm which are explained below. We illustrate the least squares idea with the help of a very nice Java applet found at <http://sambaker.com/courses/J716/demos/LeastSquares/LeastSquaresDemo.html>:



The applet allows interactive search of the solution by positioning the regression line according to the users wish. The squared residuals and their total sum can be displayed. While experimentation by hand will eventually lead to the minimum, it is cumbersome and laborious. Is there a mathematical procedure that finds the solution? The answer is yes, it is the ordinary least squares (OLS) algorithm.

Picking up the above paradigm, the goal is to fit the regression line such that the sum of squared differences  $r_i$  between the observed values  $y_i$  and the regression line is minimal, given a fixed set of data points  $(x_i, y_i)_{i=1, \dots, n}$ . We can thus define the following function that measures the quality of the fit:

$$Q(\beta_0, \beta_1) = \sum_{i=1}^n r_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2 = \min!$$

The goal is to minimize  $Q(\cdot, \cdot)$ . Since the data are fixed, this has to be done with respect to the two regression coefficients  $\beta_0, \beta_1$ . Or in other words, the parameters need to be found such that the sum of squared residuals is minimal. The idea for the solution is to set the partial derivatives to zero:

$$\frac{\partial Q}{\partial \beta_0} = 0 \quad \text{and} \quad \frac{\partial Q}{\partial \beta_1} = 0.$$

We leave the calculus as an exercise, but the result is a linear equation system with two equations and the two unknowns  $\beta_0, \beta_1$ . In linear algebra, these are known as the normal equations. Under some mild conditions (in simple linear regression this is: we have at least two data points with different values for  $x_i$ ), the solution exists, is unique and can be written explicitly:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} \quad \text{and} \quad \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}.$$

We put a hat symbol (“^”) on the optimal solutions. This is to indicate that they are estimates, i.e. determined from a data sample. Given the data pairs  $(x_i, y_i)_{i=1, \dots, n}$  they could now be computed with a pocket calculator. Or better, and more conveniently, with R:

```
> lm(Pax ~ ATM, data=unique2010)
```

Call:

```
lm(formula = Pax ~ ATM, data = unique2010)
```

Coefficients:

(Intercept)	ATM
-1197682.1	138.8

The `lm()` command (from *linear modeling*) is based on the formula interface. The relation has to be provided in the form  $y \sim x$ , and with argument `data`, it is specified in which data frame these variables can be found. The output repeats the function call and provides the estimates  $\hat{\beta}_0$  and  $\hat{\beta}_1$ .

The interpretation of this solution is straightforward: every additional air traffic movement on average provides  $\hat{\beta}_1 = 138.8$  additional passengers. And if there were no air traffic movements, we would have  $\hat{\beta}_0 = -1'197'682$  passengers. While the solution for  $\hat{\beta}_1$  is plausible, this is not the case for  $\hat{\beta}_0$ . How can this happen?

It is because the observed set of data points is very far to the right of  $x = 0$ . It tells us that the linear relation we identified does not hold for very small numbers of air traffic movements. From a practical viewpoint, this is well acceptable. If the demand was that much smaller at Zurich Airport, it would be serviced by smaller airplanes. Or in other words: the regression line (at best) holds for the data we observed, and not for hypothetical values far beyond the range of observed

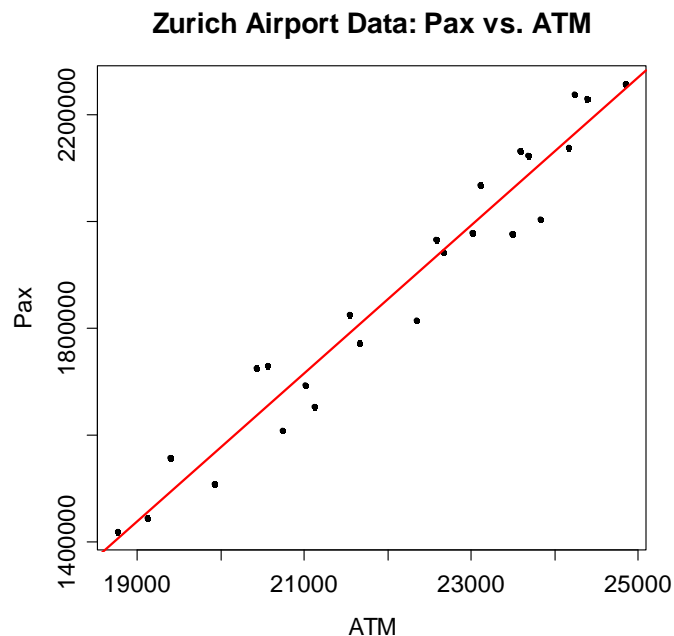
$x$ -values. Thus, we do not need to worry much about the negative value for  $\hat{\beta}_0$ . Some further explanations on this as well as a potential remedy are provided later in this script. Using the estimated parameters, we obtain the *fitted values*, defined as:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i \text{ for all } i = 1, \dots, n.$$

These can of course be interconnected by the regression line. We here address the issue how the fitted values are accessed in R, and how the regression line is visualized:

```
> fit <- lm(Pax ~ ATM, data=unique2010)
> fitted(fit)
      1      2      3      4      5      6      7
1654841 1808312 2165068 2156465 2184911 2250545 2108731
      8      9     10     11     12     13     14
2062107 1493184 1902115 1456135 1679680 1637219 1718394
     15     16     17     18     19     20     21
2008267 1994391 2088333 2074873 1947490 1935418 1791799
     22     23     24
1733381 1406597 1566867

> plot(Pax ~ ATM, data=unique2010, pch=20)
> title("Zurich Airport Data: Pax vs. ATM")
> abline(fit, col="red", lwd=2)
```



The next issue that needs to be addressed is the quality of the solution. The OLS algorithm could be applied to any set of data points, even if the relation is curved instead of linear. In that case, it would not provide a good solution. The next section digs deeper and goes beyond the obvious.

### 2.3.3 Assumptions for OLS Estimation

The negative value for the estimated intercept had raised some doubts as to whether the OLS solution is trustworthy. We argued that  $x=0$  is far beyond the range of observed data, and that there is no guarantee that the regression line holds there. We can generalize this: on any dataset we perform regression, it remains (at best) unclear whether we can extrapolate the straight line, but most likely it is not the case. Within the range of observed data, we can make more statements. The OLS estimates are trustworthy, if:

$$E[E_i] = 0$$

The expectation (we could also say the best guess if we need to predict) for the errors is zero. This means that the relation between predictor and response is a linear function, or in our example: a straight line is the correct fit, there is no systematic deviation. Next, we require constant scatter for the error term, i.e.

$$\text{Var}(E_i) = \sigma_E^2.$$

Finally, there must not be any correlation among the errors for different instances, which boils down to the fact that the observations do not influence each other, and that there are no latent variables (e.g. time) that do so. In particular,

$$\text{Cov}(E_i, E_j) = 0 \text{ for all } i \neq j.$$

Last, we require that the errors are (at least approximately) normally distributed:

$$E_i \sim N(0, \sigma_E^2)$$

The OLS algorithm will not yield a good solution under the presence of severe outliers or with a skewed error distribution. Moreover, all significance tests and confidence intervals that are presented later strictly rely on the Gaussian assumption.

### 2.3.4 Residual Plots

Before the regression line is used, we need to check if the assumptions from section 2.3.3 are met. Some investigations on expectation, variance and distribution of the errors can be performed with the usual  $y$  vs.  $x$  scatterplot. However, it has proven more powerful to inspect residual plots that are directed towards identifying potential violations. As it turns out, the human eye is easily deceived when it needs to judge if some data points follow an inclined straight line. However, it is much better in detecting deviations from the horizon. This is utilized in the first residual plot, where the effect of the regression line is subtracted. This means that the *residuals are plotted against the predictor*. The visualization can be enhanced by adding a horizontal line and a scatterplot smoother (we choose a LOESS).

```

> ## Residuals vs. Predictor
> xx <- unique2010$ATM
> yy <- residuals(fit)
> plot(xx, yy, xlab="ATM", ylab="Residuals", pch=20)
> title("Residuals vs. Predictor ATM")
> lines(loess.smooth(xx,yy),col="red")
> abline(h=0, col="grey")

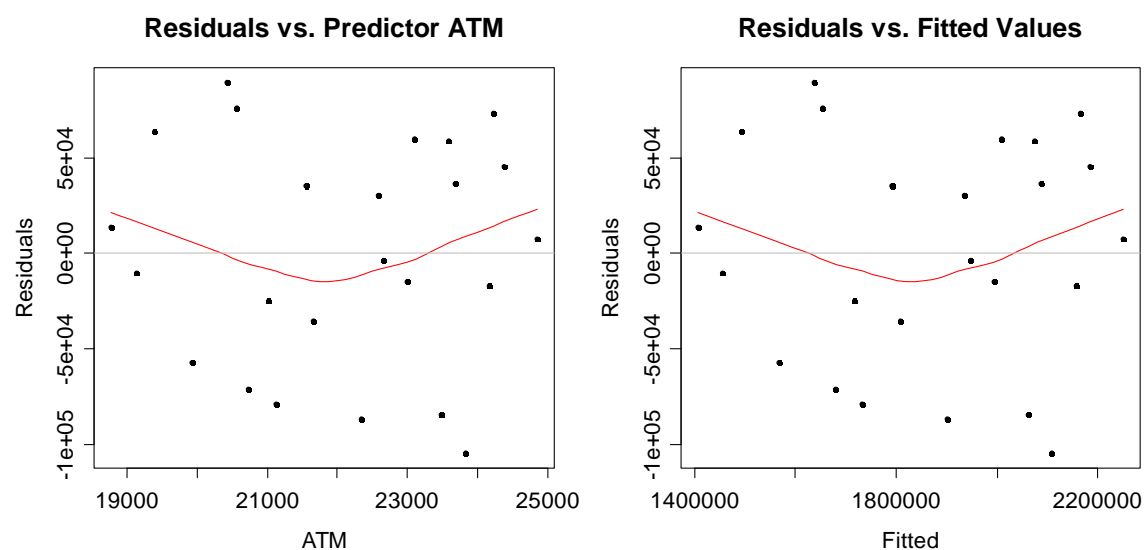
```

Another option is to plot the residuals versus the fitted values. This is known as the *Tukey-Anscombe plot*, according to the researchers who made it popular. As can be seen below, the two plots are one and the same except for the different  $x$ -axis. This is no surprise, because the fitted value  $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$  stems from a linear transformation of  $x$ . While plotting *residuals vs. predictor* is perhaps the more natural way of doing it in simple regression, the Tukey-Anscombe plot provides a simple and intuitive summary in multiple regression, where several predictors exists. Thus, it is often also applied for simple regression.

```

> ## Tukey-Anscombe Plot
> uu <- fitted(fit)
> plot(uu, yy, xlab="Fitted", ylab="Residuals", pch=20)
> title("Residuals vs. Fitted Values")
> lines(loess.smooth(uu,yy),col="red")
> abline(h=0, col="grey")

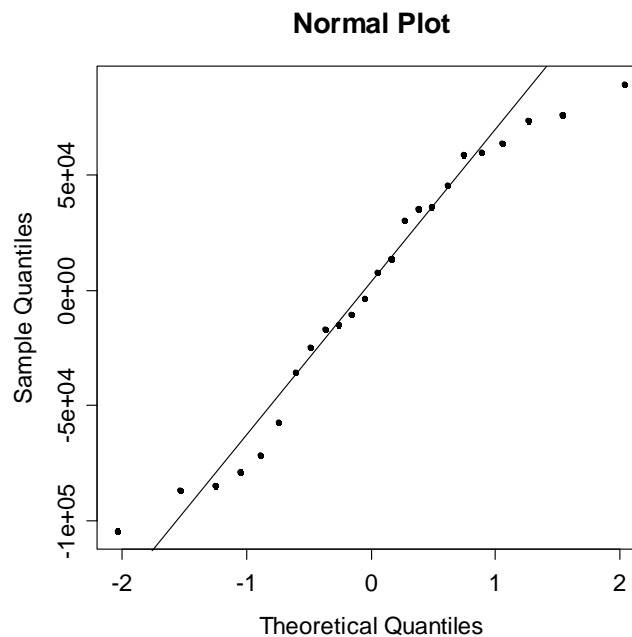
```



The smoother deviates from the horizon, and there is quite a clear kink in the relation. It seems as if the residuals for low and high *ATM* (resp. fitted) values are systematically positive, and negative for medium *ATM* values. If that is the case, it is a violation of the  $E[E_i] = 0$  assumption; a straight line is not the correct fit and improving the model is mandatory. For the moment however, we keep in mind that some doubts are raised by this residual plot, but continue with developing theory. The constant variance assumption can also be judged from the above plot. It seems as if the scatter is more or less constant for the entire range of *ATM* values. Or maybe better: there is no obvious violation.

We proceed to checking if the residuals follow a Gaussian distribution. This can be done with a so-called *Normal Plot*, sometimes also named *QQ-Plot*, where the ordered residuals are shown versus quantiles of the Gaussian distribution. The data must more or less follow a straight line. This is sufficiently met here in our case; the residuals are even slightly short-tailed with respect to the Gaussian. An in-depth discussion about what still fits within the assumption and what does not is again postponed to later.

```
> qqnorm(residuals(fit))  
> qqline(residuals(fit))
```



One last assumption has not been verified yet, namely the one whether the errors are uncorrelated. In many regression problems, this is the most difficult to verify. Also here, we could ask ourselves whether events such as the 9/11 terror attacks, or the SARS lung disease might have unduly influence. They could have led to back-to-back months with lower seat load factors, thus less passengers than expected by the air traffic movements during normal periods, and by this induce correlated errors. Because none of these events falls within our period of observation, we do not pursue the issue here. It will be addressed in detail when we talk about multiple linear regression.

### 2.3.5 History of Least Squares

You may find it somewhat arbitrary that we chose the sum of squares residuals as the criterion to minimize. We might as well optimize the absolute values' sum of the residuals, the so-called  $L_1$ -regression. There are a number of reasons to prefer the former. The first one lies in history, least squares was simply the first such algorithm that was used in practice. The English Wikipedia site on the term least squares holds the following information:

On January 1, 1801, the Italian astronomer Giuseppe Piazzi discovered the dwarf planet Ceres and was able to track its path for 40 days before it was lost in the glare of the sun. Based on these data, astronomers desired to determine the location of Ceres after it emerged from behind the sun without solving the complicated Kepler's nonlinear equations of planetary motion. The only predictions that successfully allowed relocating Ceres were those performed by the 24-year-old Carl Friedrich Gauss using the least squares algorithm.

Gauss did not publish the method until 1809, when it appeared in volume two of his work on celestial mechanics, together with a mathematical optimality result, the Gauss-Markov theorem (see below). In the meantime, the OLS algorithm was independently formulated by Adrian Marie Legendre, who was the first to publish it in 1806 as an appendix to his book on the paths of comets. Below, see a table of Piazzi's observations, and portraits of Gauss (left) and Legendre (right).

Beobachtungen des zu Palermo d. 1. Jan. 1801 von Prof. Piazzi neu entdeckten Ceres.

1801	Mittlere Sonnen-Zeit	Größe der Aufhällg. in Grad.	Gerade Aufhällg. in Grad.	Nördl. Abweich.	Größtmögl. Länge	Größtmögl. Breite	Ort. der Sonne + 20" Abstraktion	Logar. d. Distanz $\odot$ S
Jan.	1 8 43 37.3	3 27 11.25	51 47 48.8	15 27 43.5	1 23 22 58.2	5 6 42.1	9 11 1 30.5	9.9931556
	2 8 39 4.6	3 26 53.85	51 43 27.8	15 41 55.5	1 23 19 44.2	3 2 24.9	9 12 2 28.6	9.9926117
	3 8 34 53.1	3 26 38.4	51 39 36.0	15 44 21.6	1 23 16 58.6	2 53 9.9	9 13 3 26.6	9.9921214
	4 8 30 42.1	3 25 23 15.1	51 35 47.3	15 47 25.6	1 23 14 15.2	3 53 55.6	9 14 4 24.9	9.9916118
	5 8 6 15.8	3 25 32.1	51 32 14.2	16 10 35.0	1 23 7 50.2	2 39 0.6	9 20 10 17.5	9.9912544
	6 8 2 17.5	3 25 29.73	51 28 26.0	.....	.....	.....	.....	.....
	7 8 54 26.2	3 25 30.30	51 22 24.5	16 22 49.5	1 23 10 27.6	2 16 59.7	9 23 12 13.5	9.9913490
	8 8 50 39.7	3 25 31.72	51 23 55.8	16 27 5 7	1 23 12 11.2	2 12 56.7	9 24 14 13.5	9.9928509
	9 8 46 11.1	.....	.....	16 40 13.0	.....	.....	.....	.....
	10 8 35 13.3	3 25 55. 1	51 28 45.0	.....	.....	.....	.....	.....
	11 8 31 28.0	3 26 8 15	51 32 27.3	16 49 16.1	1 23 25 59.2	1 53 38.2	9 29 19 53.8	9.9930607
	12 8 24 21.5	3 26 34 37.5	51 28 24.1	16 58 35.9	1 23 34 21.1	1 45 6.0	10 1 20 48.2	9.9931434
	13 8 20 31.7	3 26 40 42.5	51 23 21.2	17 3 18.5	1 23 39 1 1	1 42 26.1	10 2 21 39.2	9.9931856
	14 8 16 45.5	3 27 6 50.1	51 46 43.5	17 8 5 5	1 23 44 15.7	1 38 52.1	10 3 32 12.7	9.9932348
	15 8 58 51.3	3 28 54 32.5	51 33 38.3	17 33 34.1	1 24 15 15.7	1 21 6 9	10 8 26 20.1	9.9935061
	16 8 51 57.9	3 29 45 14	52 27 2 1	17 43 11.0	1 24 30 9 0	11 16 0	10 10 27 46.2	9.9935332
	17 8 6 28 54.1	3 30 17 25	52 34 18.8	17 48 21.5	1 24 38 7 3	1 10 54.6	10 11 28 28.3	9.9937007
Febr.	1 6 44 59.9	3 30 47 21	52 41 48 0	17 53 36.5	1 24 45 19.2	1 7 30 9	10 12 39 9 6	9.9937703
	2 6 41 35.3	3 31 19 06	52 49 45 9	17 58 37.5	1 24 54 57 9	1 4 1 5	10 13 39 49 9	9.9938423
	3 6 38 11.8	3 32 2 29	53 15 49 5	18 12 1 0	1 25 2 45 0	54 8 9	10 16 35 45 3	9.9940751
	4 6 34 39.3	3 34 58 20	53 44 37 5	18 31 23 2	1 25 53 39 5	0 45 5 0	10 19 32 32 3	9.9943375
	5 6 31 58.2	3 37 6 24	54 16 35 1	18 47 58 8	1 26 25 40 0	0 36 2 9	10 22 25 13 4	9.9945833



Was it by coincidence that OLS was invented first? The answer is no: the quality function  $Q(\cdot, \cdot)$  is differentiable, so that a unique solution can be found and written in explicit form. This is not possible with  $L_1$ -regression, because the absolute value function is not continuously differentiable. While this problem can nowadays be circumvented with numerical methods, this was not yet feasible at the beginning of the 19<sup>th</sup> century. The reason why OLS is still popular today is because there are mathematical optimality results, and because under Gaussian errors, the exact distribution of the estimated coefficients and a number of test statistics is known.

### 2.3.6 Mathematical Optimality of OLS

The main result is the *Gauss-Markov theorem* (GMT) that dates back to 1809:

Under the model assumptions from section 2.3.3 (*zero expected value, constant variance and uncorrelatedness for the errors*), the OLS estimates  $\hat{\beta}_0, \hat{\beta}_1$  are unbiased (i.e.  $E[\hat{\beta}_0] = \beta_0$  and  $E[\hat{\beta}_1] = \beta_1$ ). Moreover, they have minimal variance among all unbiased, linear estimators, meaning that they are most precise. Please note that Gaussian errors are not required.

This theorem does not tell us to use OLS all the time, but it strongly suggests doing so if the assumptions are met. In cases where the errors are correlated or

have unequal variance, we will do better with other algorithms than OLS. Also, note that even though normality is not required for the GMT, there will be non-linear or biased estimates that do better than OLS under non-Gaussian errors.

As we have seen just before, the regression coefficients are unbiased if the assumptions from section 2.3.3 are met. It is also very instructive to study the variance of the estimates. It can be shown that:

$$\text{Var}(\hat{\beta}_0) = \sigma_E^2 \cdot \left( \frac{1}{n} + \frac{\bar{x}}{\sum_{i=1}^n (x_i - \bar{x})^2} \right), \text{ and}$$

$$\text{Var}(\hat{\beta}_1) = \frac{\sigma_E^2}{\sum_{i=1}^n (x_i - \bar{x})^2}.$$

These results also show how a good experimental design can help to improve the quality of the estimates, or in other words, how we can obtain a more precisely determined regression line. Namely:

- we can increase the number of observations  $n$ .
- we have to make sure that the predictors  $x_i$  scatter well.
- by using a suitably-chosen predictor, we can keep  $\sigma_E^2$  small.
- for  $\hat{\beta}_0$  it helps, if the average predictor value  $\bar{x}$  is close to zero.

If the errors are Gaussian, then  $\hat{\beta}_0, \hat{\beta}_1$  are normally distributed, too. With their expectation and variance specified as above, the distribution is fully known. Additionally, the OLS solution is also the maximum likelihood estimator under Gaussian errors. Some further useful properties of the OLS solution (that are independent of the error distribution) are:

- the regression line runs through the center of gravity  $(\bar{x}, \bar{y})$ .
- the sum of residuals adds up to zero:  $\sum r_i = 0$ .

The last property also implies that the mean value of the residuals is always zero.

### 2.3.7 Estimating the Error Variance

Besides the regression coefficients, we also need to estimate the error variance. It is a necessary ingredient for all tests and confidence intervals. The estimate is based on the residual sum of squares (abbreviation: RSS).

$$\hat{\sigma}_E^2 = \frac{1}{n-2} \cdot \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

In the R summary, an estimate for the error's standard deviation  $\hat{\sigma}_E$  is given as the Residual standard error.

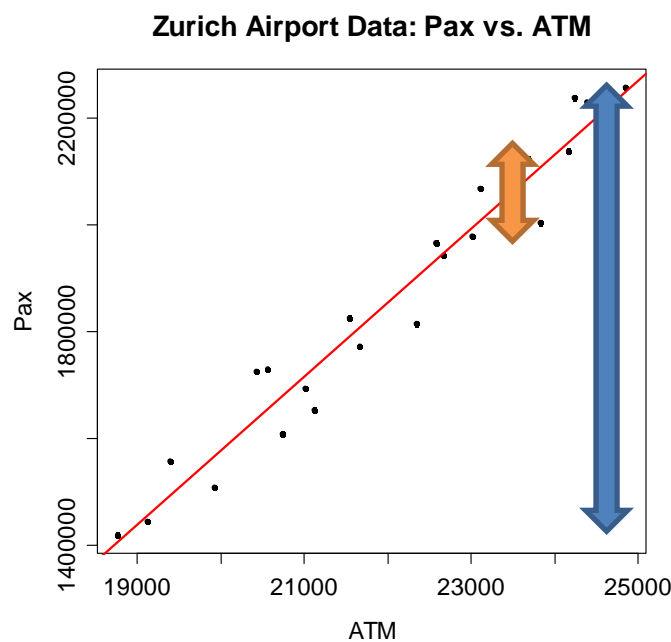


## 2.4 Inference

The goal in this section is to infer the response-predictor relation with performance indicators and statistical tests. Note that except for 2.4.1, the assumption of *independent, identically distributed Gaussian errors* is central to derive the results.

### 2.4.1 The Coefficient of Determination

An intuitive way of measuring the goodness-of-fit of a simple linear regression model is with the *coefficient of determination*  $R^2$ , also called *multiple R-squared*. It measures which portion of the total variation is accounted for by the regression.



If we needed to predict the *Pax* number without any knowledge of the *ATM* value, the best guess is the average number of passengers over the last two years. The scatter around that prediction is visualized by the blue arrow. However, since we know *ATM* and the regression line, we can come up with a more accurate forecast. The then remaining scatter is indicated by the orange arrow. It is obvious that the regression line is more useful, the smaller the orange arrow is compared to the blue. This can be measured by taking one minus the quotient of the two:

$$R^2 = 1 - \frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \in [0,1]$$

In the numerator, the orange arrow is represented by the scatter of the data points around the fitted values, i.e. the *RSS*. The denominator has the scatter of the data points around their mean. This is the *total sum of squares (TSS)*.

The maximum value is  $R^2 = 1$ . It is attained if all data points are on the regression line. The other extreme case is  $R^2 = 0$  and means that the blue and orange arrows have the same size. Then, the regression line is flat ( $\hat{\beta}_1 = 0$ ) and does not have any explanatory power. The actual value can be read from the R summary:

```
> summary(fit)
Coefficients: Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.198e+06  1.524e+05  -7.858 7.94e-08 ***
ATM          1.388e+02  6.878e+00  20.176 1.11e-15 ***
---
Residual standard error: 59700 on 22 degrees of freedom
Multiple R-squared:  0.9487, Adjusted R-squared:  0.9464
F-statistic: 407.1 on 1 and 22 DF, p-value: 1.11e-15
```

The result here is  $R^2 = 0.9487$ , thus most of the variation in the *Pax* variable is explained by *ATM*. It is important to note that for simple linear regression,  $R^2$  is equal to the *squared Pearson correlation coefficient* between predictor and response. Moreover, the summary reports the adjusted R-squared. Its value is always smaller but usually close to  $R^2$ , because:

$$aR^2 = R^2 - \frac{1 - R^2}{n - 2}.$$

An important question is now: what is a good value for  $R^2$ ? Unfortunately, it remains without an answer. There are no general guidelines as to which value needs to be met for a regression to be useful, and there are no formal tests for  $R^2$ . And please note that a high value for  $R^2$  does not automatically mean that we have a good fit that we can rely on – often, further improvement by alternative models (i.e. by using transformations) is still possible.

## 2.4.2 Confidence Interval for the Slope

The estimated slope  $\hat{\beta}_1$  is a random variable and has variability. If the assumptions for the OLS algorithm are met, we have the Gauss-Markov theorem telling us its value will be close to the truth  $\beta_1$ , but not right there. Also, the value  $\hat{\beta}_1$  was computed from a sample. Had we had a different one, or would we just omit one single data point from our current one,  $\hat{\beta}_1$  would turn out different. The goal is to reflect that uncertainty with a *95% confidence interval* (CI). The formula is:

$$\hat{\beta}_1 \pm qt_{0.975;n-2} \cdot \hat{\sigma}_{\hat{\beta}_1}, \text{ resp. } \hat{\beta}_1 \pm qt_{0.975;n-2} \cdot \sqrt{\hat{\sigma}_E^2 / \sum_{i=1}^n (x_i - \bar{x})^2},$$

where  $qt_{0.975;n-2}$  is the 97.5% quantile of Student's t-distribution with  $n - 2$  degrees of freedom. The colloquial interpretation is that the interval holds all values which, besides the point estimate  $\hat{\beta}_1$ , are plausible for  $\beta_1$ . In R, one types:

```
> confint(fit, "ATM")
      2.5 %    97.5 %
ATM 124.4983 153.025
```

We estimated the increase in passengers per additional air traffic movement as  $\hat{\beta}_1 = 138.8$ . That is the best guess given the data, but values between 124.5 and 153.0 are also plausible. This reflects the uncertainty and variability in our regression analysis. If the 95%-CI seems unacceptably wide, all we can do is trying to bring  $\hat{\sigma}_{\hat{\beta}_1}$  down, i.e. have more or better data, see section 2.3.6.

### 2.4.3 Testing the Slope

For finding out whether an arbitrary value  $b$  is plausible for the slope, we can check whether it is contained in the 95%-CI from above. Alternatively, there is a test for the *null hypothesis*  $H_0: \beta_1 = b$ . The most popular variant is  $H_0: \beta_1 = 0$ : this is asking if the slope could be zero, which would mean that the regression line runs horizontally and the predictor  $x$  has no influence on the response  $y$ . The natural goal is to reject the null for gaining evidence that the relation between  $y$  and  $x$  exists. One usually tests two-sided on the 95% level, i.e. the alternative is  $H_A: \beta_1 \neq b$ . The *test statistic* and its *distribution* are as follows:

$$T_{H_0: \beta_1 = b} = \frac{\hat{\beta}_1 - b}{\hat{\sigma}_{\hat{\beta}_1}} \sim t_{n-2}.$$

Student's t-distribution with  $n-2$  degrees of freedom can be used to determine acceptance and rejection regions, as well as the *p-value*. In fact, both the *test statistic* (t value) and the *p-value* ( $\Pr(>|t|)$ ) for  $H_0: \beta_1 = 0$  are routinely given in the R summary output:

```
> summary(fit)
```

```
Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	-1.198e+06	1.524e+05	-7.858	7.94e-08	***
ATM	1.388e+02	6.878e+00	20.176	1.11e-15	***

```
---
```

```
Residual standard error: 59700 on 22 degrees of freedom
```

```
Multiple R-squared: 0.9487, Adjusted R-squared: 0.9464
```

```
F-statistic: 407.1 on 1 and 22 DF, p-value: 1.11e-15
```

We have very strong evidence for  $\beta_1 \neq 0$  here, and thus the null hypothesis is rejected with a p-value of  $1.1 \cdot 10^{-15}$ . The fact of rejection was already clear from the 95%-CI which contains all null hypotheses that are not rejected – and zero was not therein – with a huge margin, that is, and hence the extreme p-value.

*It is very important to stress again, that all confidence intervals and test results are only to be trusted if the assumptions for OLS fitting (i.e. zero expectation, constant variance, Gaussian distribution and uncorrelatedness of the errors) are closely met. If any clear deficiencies were found from the residuals plots, it is mandatory to improve the model before these results are reassessed!*

## 2.4.4 Testing the Intercept

In many simple linear regression problems, theory dictates that we have a response of  $y=0$  whenever  $x=0$ . That is the case with the Zurich Airport Data, too. If there were no air traffic movements, we would not see any passengers. However, it is hardly ever a good idea to fit a model without an intercept term. This forces the regression line to go through the origin which is a very strong restriction, that in most cases leads to a poor fit.

Commonly, the reason for the poor fit is because the data points are far off  $x=0$ . This leads to very high leverage with respect to  $\beta_0$ , and just some slight non-linearity between response and predictor results in an intercept that is markedly different from zero. This happens in our example where  $\hat{\beta}_0 = -1'197'682$ . In analogy to sections 2.4.2 and 2.4.3, tests and confidence intervals for  $\beta_0$  exist. For the Zurich Airport data, the null hypothesis  $H_0: \beta_0 = 0$  is strongly rejected with a p-value of  $7.9 \cdot 10^{-8}$ , and the confidence interval is:

```
> confint(fit, "(Intercept)")
              2.5 %      97.5 %
(Intercept) -1513786 -881578.2
```

However, both test and confidence interval for  $\beta_0$  are of relatively low practical importance. As a *general rule*, we should *not fit regression models without an intercept term*. If the null is not rejected and thus zero is a plausible value, it is still better and safer to keep it in the model. If it turns out to be significantly different from zero, take it as evidence for either some non-linearity or calibration errors in the data. In these latter cases, the results will be clearly worse (i.e. strongly biased) without the intercept. We close here with the remark that for many regression problems which need to run through the origin, using the log-log-model displayed in section 2.6.4 is the best choice.

## 2.5 Prediction

One of the primary goals with linear regression is to generate a prediction for  $y$ , given the value of  $x$ . The result is the conditional expectation for  $y$  given  $x$ , i.e. what we expect for  $y$  if the predictor value  $x$  is known:

$$E[y|x] = \hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

For 24'000 air traffic movements, we expect  $-1'197'682 + 24'000 \cdot 138.8 = 2'133'518$  passengers. Please note that only a prediction within the range of  $x$ -values that were present for fitting is sensible. This is called *interpolation*. On the other hand, *extrapolation*, i.e. a prediction beyond the boundaries of the  $x$ -values previously observed, has to be treated with great care: there is no guarantee that the regression line holds in non-observed regions of the predictor space. Thus, we must not predict the Pax figure for ATM values such as 50'000, 5'000 or 0.

In R, we can obtain the fitted values for the training data points by just typing `predict(fit)`. If we want to use the regression line for forecasting with new  $x$ -values, they have to be provided in a data frame, where the column(s) are named equally to the predictor(s):

```
> fit <- lm(Pax ~ ATM, data=unique2010)
> dat <- data.frame(ATM=c(24000))
> predict(fit, newdata=dat)
1 2132598
```

## 2.5.1 Confidence Interval for the Regression Line

As we had seen above in section 2.4.2, the regression coefficients are random variables. Thus, also the regression line is a random variable, and might have turned out to be different with another sample (even if from the same population). Thus, it is important to understand, quantify and visualize the variability of the fitted value. This is done on the basis of a *95%-CI for the conditional expectation*. The formula is:

$$95\text{-CI for } E[y|x]: \hat{\beta}_0 + \hat{\beta}_1 x \pm qt_{0.975;n-2} \cdot \hat{\sigma}_E \cdot \sqrt{\frac{1}{n} + \frac{(x - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2}}$$

The formula itself is of relatively little importance for the practitioner, because that functionality is pre-existing in R. The syntax is:

```
> predict(fit, newdata=dat, interval="confidence")
      fit      lwr      upr
1 2132598 2095450 2169746
```

The meaning of this output is as follows: for an ATM value of 24'000, we expect 2'132'598 passengers. A 95%-CI for that conditional expectation ranges from 2'095'450 to 2'169'746.

## 2.5.2 Prediction Interval for Future Data Points

While the above 95%-CI tells characterizes the variability in the fitted value, it does not tell us where the (future)  $y$ -value will be, i.e. what number of passengers we will observe for a given ATM value. The reason is that (also within the training data), the observed  $y$ -values scatter around the regression line (i.e. their conditional expectation). Taking this into account, we can derive a *95% prediction interval (PI)* for  $y$ . The formula is:

$$95\text{-PI for } y: \hat{\beta}_0 + \hat{\beta}_1 x \pm qt_{0.975;n-2} \cdot \hat{\sigma}_E \cdot \sqrt{1 + \frac{1}{n} + \frac{(x - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2}}$$

The difference in the formula is that another unit of  $\hat{\sigma}_E$  is included to account for the scatter of the data points around the regression line. Again, the formula is implemented in R:

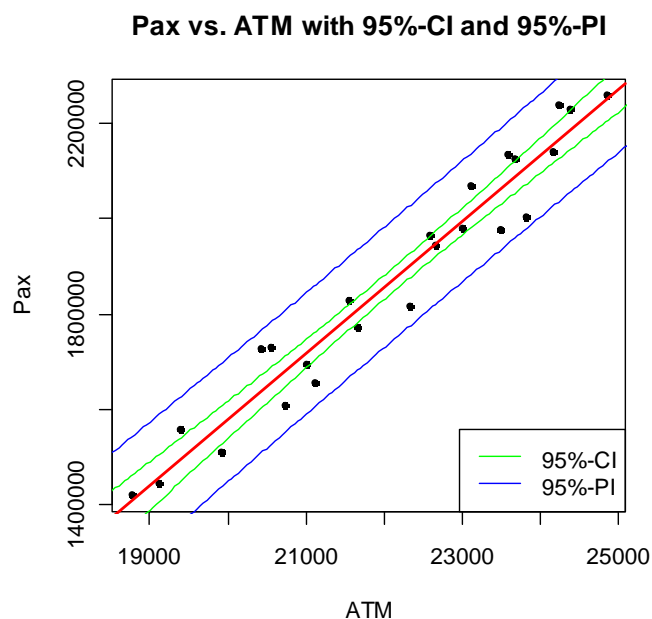
```
> predict(fit, newdata=dat, interval="prediction")
      fit      lwr      upr
1 2132598 2003343 2261853
```

Because we are still predicting for an ATM value of 24'000, we the conditional expectation remains at 2'132'598 passengers. A 95% prediction interval for a future observation when there are 24'000 air traffic movements ranges from 2'003'343 to 2'261'853.

### 2.5.3 Visualizing Confidence and Prediction Intervals

It is very instructive to compute point-wise CIs and PIs and to display them in the  $xy$ -scatterplot, along with the regression line. There is no straightforward procedure in R to do so, but some rather tedious handwork is required. A possible solution is as follows:

```
> dat <- data.frame(ATM=seq(18000, 26000, length=200))
> ci <- predict(fit, newdata=dat, interval="confidence")
> pi <- predict(fit, newdata=dat, interval="prediction")
> plot(Pax ~ ATM, data=unique2010, pch=20)
> title("Pax vs. ATM with 95%-CI and 95%-PI")
> lines(dat$ATM, ci[,2], col="green")
> lines(dat$ATM, ci[,3], col="green")
> lines(dat$ATM, pi[,2], col="blue")
> lines(dat$ATM, pi[,3], col="blue")
> abline(fit, col="red", lwd=2)
```



The result is a confidence region for the regression line, and a prediction region for future observations. The interpretation is that the former contains all plausible regression lines. The latter indicates how precisely we can forecast future observations. While the 95%-CI turns out to be rather small here, reflecting a high confidence in the estimated regression line, the 95%-PI is bigger and reflects the non-understood scatter of the observations due to reasons such as differing seat loads factors, cargo flights, et cetera.

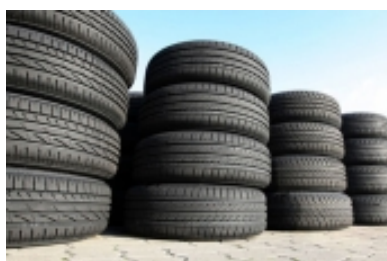
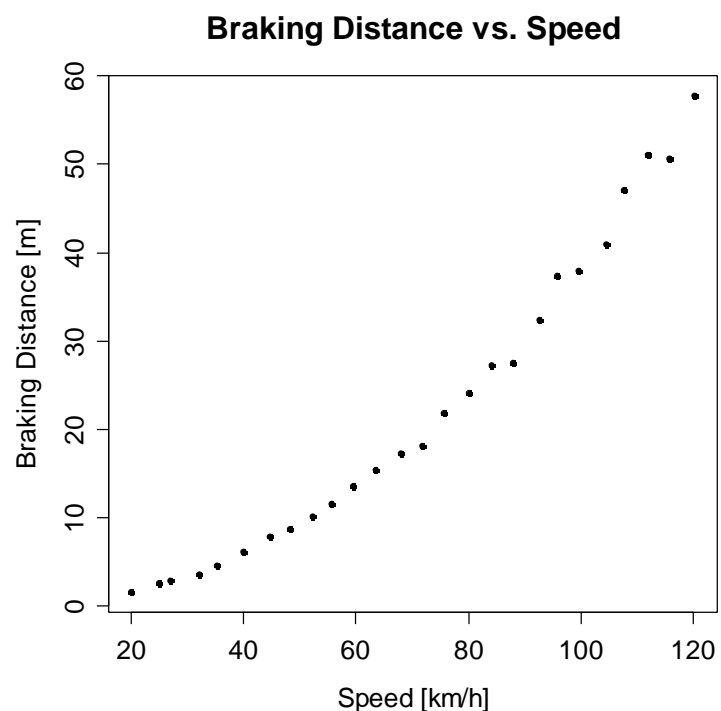
## 2.6 Model Extensions

So far, linear regression was synonym to fitting a straight line in an  $xy$ -scatterplot. However, it has to offer much more: we can also fit curves, as long as we can describe them with a relation that is linear in the regression coefficients. The following example motivates why fitting curves can be a necessity.

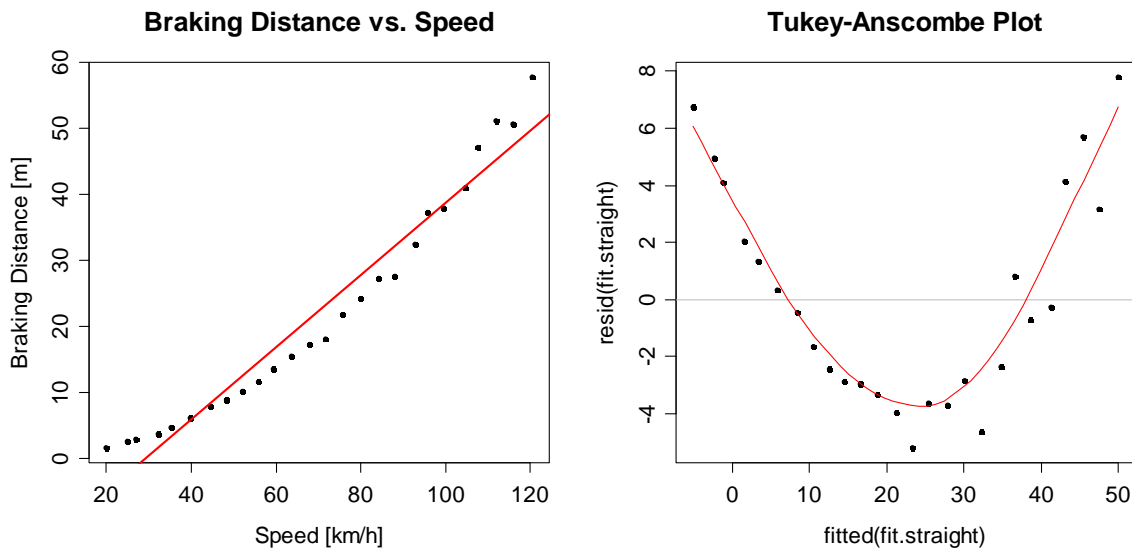
### 2.6.1 Example: Automobile Braking Distance

An automobile magazine tests summer tires with respect to the braking performance that is achieved. For acquiring data, a set of 26 test drives are made, where at various speeds the stopping distance is measured after a “pedal-to-the-metal” braking procedure. The goal is to estimate the deceleration parameter.

obs	speed	brdist
1	19.96	1.60
2	24.97	2.54
3	26.97	2.81
4	32.14	3.58
5	35.24	4.59
6	39.87	6.11
7	44.62	7.91
8	48.32	8.76
9	52.18	10.12
10	55.72	11.62
11	59.44	13.57
12	63.56	15.45
...	...	...
24	111.97	51.09
25	115.88	50.69
26	120.35	57.77



Apparently, the relation between braking distance and speed is not a straight line, but seems to have a parabolic form. This is not surprising, as it is well known from physics that the energy and thus the braking distance go with the square of the speed, i.e. at double speed it takes four times as long to standstill. Moreover, there is some variability in the data. It is due to factors that have not been taken into account, mostly the surface conditions, tire and brake temperature, head- and tailwind, etc.



Fitting a plain linear function, i.e. laying a straight line through the data points results in a poor and incorrect fit. We have a strong systematic deviation from the regression line, and the Tukey-Anscombe plot shows a strong violation of the zero error assumption. As a way out, we better fit a quadratic function:

$$BrDist_i = \beta_0 + \beta_1 \cdot Speed_i^2 + E_i, \text{ respectively}$$

$$y_i = \beta_0 + \beta_1 \cdot x'_i + E_i, \text{ where } x'_i = x_i^2 = Speed_i^2$$

The above model still is a simple linear regression problem. There is only one single predictor, the coefficients  $\beta_0, \beta_1$  enter linearly and can be estimated with the OLS algorithm. Owing to the linearity, taking partial derivatives still works as usual here, and an explicit solution for  $\hat{\beta}_0, \hat{\beta}_1$  will be found from the normal equations. In R, the syntax for fitting the quadratic function is as follows:

```
> fit.q <- lm(brdist ~ I(speed^2), data=abd)
```

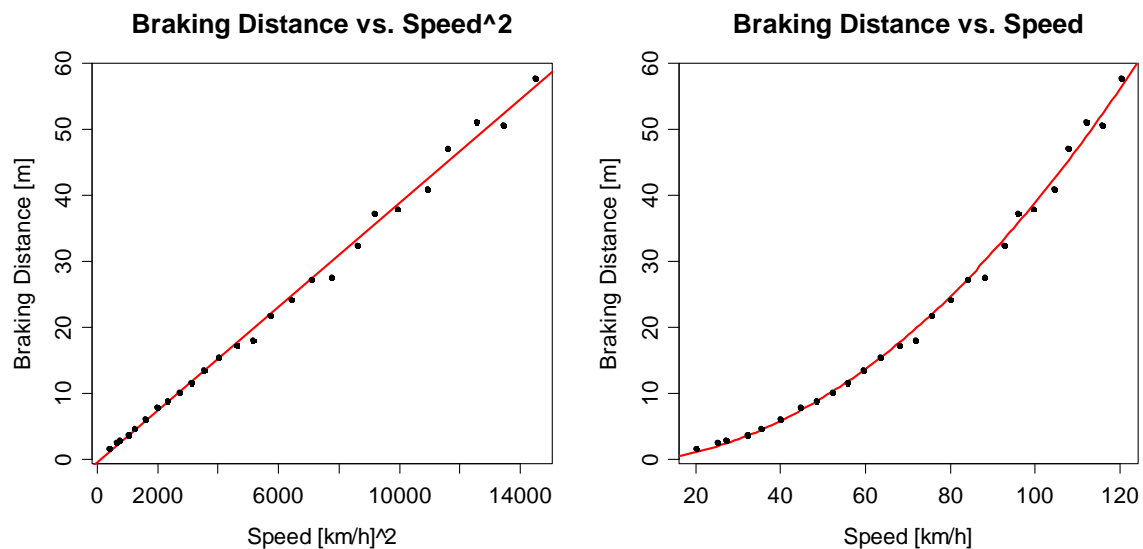
When using powers as predictors, we should always use function  $I()$ . It prevents that the power is interpreted as a formula operator, when it in fact is an arithmetic operation that needs to be performed on the predictor values. It is important to note that the quadratic relation can either be interpreted as a straight line in a  $y$  vs.  $x^2$  plot, or as a parabola in a regular  $y$  vs.  $x$  scatterplot. The following code can be used for visualizing the result:



```

> ## Braking Distance vs. Speed^2
> plot(brdist~speed^2, data=abd, main="...")
> abline(fit.q, col="red", lwd=2)
>
> ## Braking Distance vs. Speed
> yy <- predict(fit.q, newdata=data.frame(speed=10:130))
> plot(brdist ~ speed, data=abd, main="...")
> lines(10:130, yy, col="red", lwd=2)

```



As it seems at first impression, the parabola yields a good fit to the braking distance data. The regression coefficients can be used to estimate the deceleration which turns out to be roughly  $-10m/s$ . Some drawbacks of this model will be pointed out below.

## 2.6.2 Curvilinear Regression

From the automobile example, we conclude that simple linear regression is more than just fitting straight lines. In fact, any curvilinear relation can be fitted, e.g.:

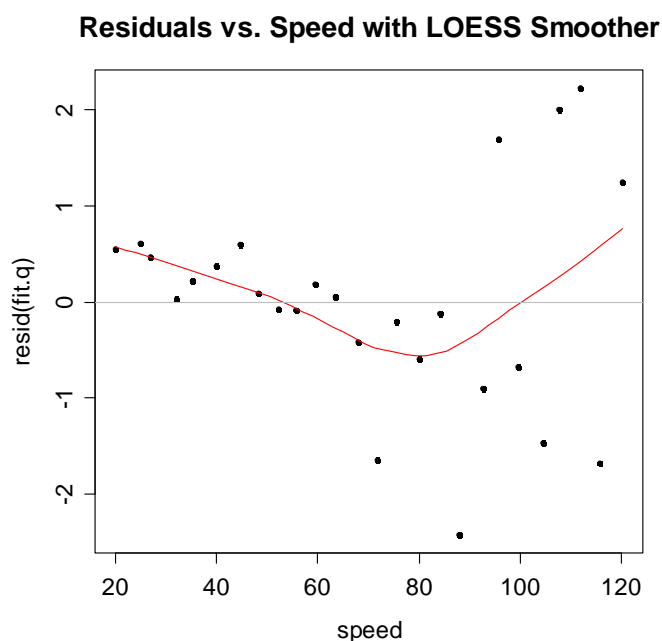
- $y = \beta_0 + \beta_1 \cdot \ln(x) + E$
- $y = \beta_0 + \beta_1 \cdot \sqrt{x} + E$
- $y = \beta_0 + \beta_1 \cdot x^{-1} + E$ ,

All these models, and many more, can be rewritten in the form  $y = \beta_0 + \beta_1 x' + E$ , where the predictor is either  $x' = \ln(x)$ ,  $x' = \sqrt{x}$  or  $x' = x^{-1}$ . Thus, estimating the parameters  $\beta_0, \beta_1$  can be reduced to the well-known simple linear regression problem, for which the OLS algorithm can be used. While this may sound like the ideal solution to many regression problems, it is not, for a number of reasons.

First, when the residuals from the quadratic model are plotted versus predictor speed, it turns out that the situation is far less than optimal. Clearly apparent is a

violation of the constant error-variance assumption. That is not so surprising, even without looking at the data; we might have expected that the scatter in braking distances becomes bigger as the speed increases. This is problematic because the high speed observations so (implicitly) obtain more weight in determining the regression coefficients. Consequently, we observe a bias for the low speed braking distances, because OLS focuses on the data points with large residuals on the right hand side, but puts less emphasis on what is going on at lower speeds.

```
> plot(speed, resid(fit.q))
> title("Residuals vs. Speed with LOESS Smoother")
> smoo <- loess.smooth(speed, resid(fit.q))
> lines(smoo, col="red")
> abline(h=0, col="grey")
```



Thus, while at first the parabola seemed to fit well to the data, closer inspection shows that we have not found a very good solution yet. Unfortunately, that is often the case when just single power terms are used as predictors.

### 2.6.3 Example: Infant Mortality

Our next goal is to study how infant mortality in a country depends on its wealth. We have observations from 105 countries; the data were first published in the New York Times in 1975. The infant mortality is measured as the (average) number of 1000 live born babies that do not reach the age of 5 years. The living standard is given as per-capita income in US\$. They data are accessible in R's `library(car)` as `data(Leinhardt)`. For clarity, we remove four countries with partly missing values and two outliers: Saudi Arabia and Lybia, both oil-exporting countries with an inhomogeneous population consisting of a few very rich leaders and mostly poor population. The data can be displayed in a scatterplot:

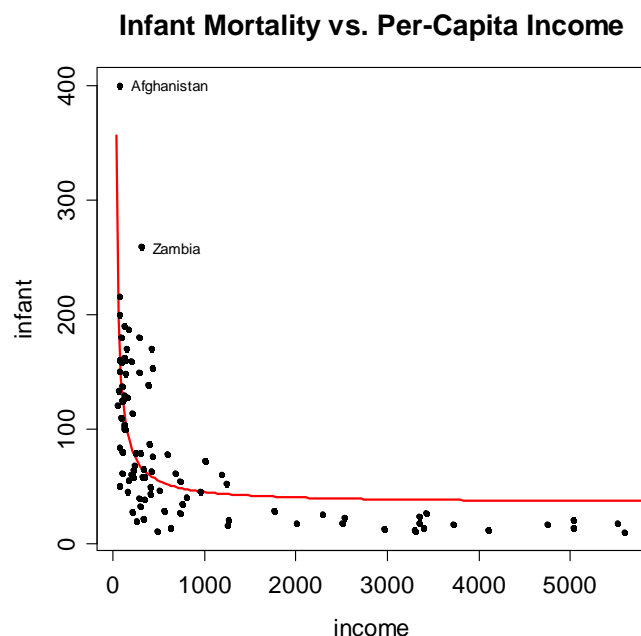
```
> plot(infant ~ income, data=im, pch=20)
> title("Infant Mortality vs. Per-Capita Income")
```

Since the relation between mortality and income seems to be inversely proportional, we might try a curvilinear regression model of the form:

$$\text{infant} \sim \beta_0 + \beta_1 \cdot (\text{income})^{-1} + E$$

As explained in 2.6.2, this is a simple linear regression problem where we can estimate the coefficients with OLS. The result is added to the scatterplot.

```
> fit <- lm(infant ~ I(income^-1), data=im)
> xx <- data.frame(income=seq(0, 6000, length=200))
> yy <- predict(fit, newdata=xx, interval="prediction")
> lines(xx$income, yy[,1], col="red", lwd=2)
> points(infant ~ income, data=im, pch=20)
```



The resulting fit is poor, as the infant mortality is strongly overestimated in all rich countries. One might conclude that this is because we failed to identify the correct exponent for the *income* variable. Rather than just trying a few different powers, we might be tempted to estimate it from data, with a model such as:

$$y = \beta_0 + \beta_1 \cdot x^{\beta_2} + E$$

That however, is no longer a relation that is linear in the parameters. Least squares fitting, i.e. taking partial derivatives in the quality function will not lead to a linear equation system, because the result is of more complicated form.

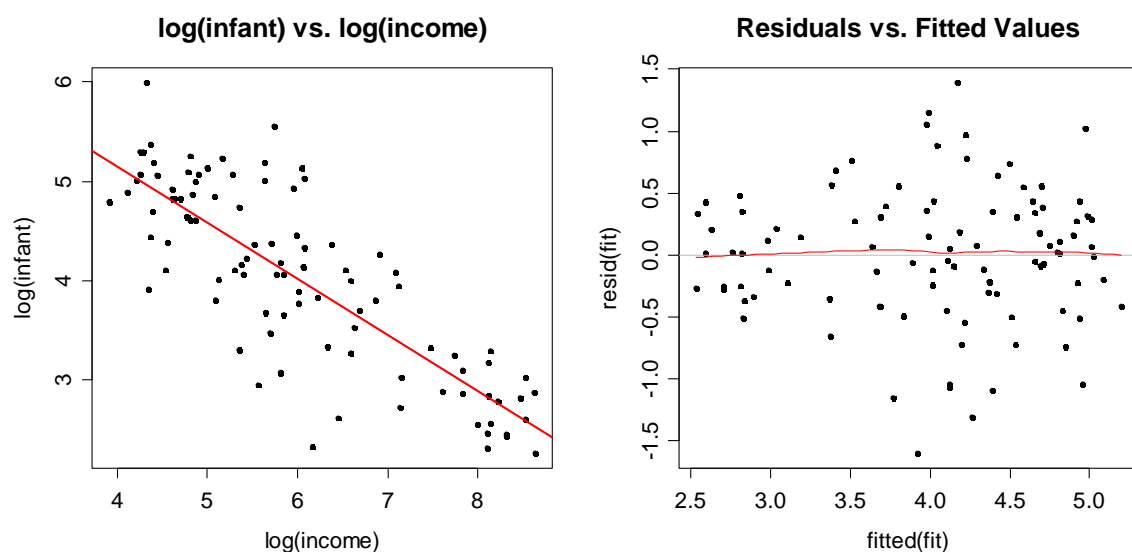
## 2.6.4 The log-log Model

In the above example, we are looking for a viable alternative to solve the regression problem. We could (and potentially would) resort to a numerical solution for minimizing the RSS, if there was not a much better analytical solution that is based on a simple, yet very powerful trick. The transformation

$$y' = \log(y), \quad x' = \log(x)$$

is of great help, as we can see with a scatterplot in the log-log scale:

```
> plot(log(infant) ~ log(income), data=im, pch=20)
> title("log(infant) vs. log(income)")
> fit <- lm(log(infant) ~ log(income), data=im)
> abline(fit, col="red", lwd=2)
> plot(fitted(fit), resid(fit), pch=20)
> abline(h=0, col="grey")
> smoo <- loess.smooth(fitted(fit), resid(fit))
> lines(smoo, col="red")
> title("Residuals vs. Fitted Values")
```



After the variable transformations, the relation seems to be a straight line. The OLS regression line fits the data well, and the Tukey-Anscombe plot does not show strongly violated assumptions, except for a maybe slightly non-constant variance (that we accept here). What has happened? If a straight line is fitted on the log-log-scale, i.e.:

$$y' = \beta'_0 + \beta'_1 \cdot x' + E' \quad \text{where } y' = \log(y), \quad x'_i = \log(x)$$

we can derive the relation on the original scale by taking the exponential function on both sides. The result is a *power law* on the original scale:

$$y = \exp(\beta'_0) \cdot x^{\beta'_1} \cdot \exp(E') = \beta_0 \cdot x^{\beta_1} \cdot E, \quad \text{with } \beta_0 = \exp(\beta'_0) \quad \text{and} \quad \beta_1 = \beta'_1.$$

The slope from the log-log-scale is the exponent to  $x$  on the original scale. Moreover, we have a multiplicative rather than an additive model, where the error term follows a log-normal distribution. Hence, the errors will scatter more the bigger  $x$  is, and are skewed towards the right, i.e. bigger values. While this model may seem arbitrary, it fits well in many cases, even more often than the canonical, transformation-free approach. The coefficients are:

```
> lm(log(infant) ~ log(income), data=im)
Coefficients:
(Intercept)  log(income)
      7.4134      -0.5661
```

The interesting part is the interpretation of the model equation. It is relative, in the following way: if  $x$ , i.e. the income increases by 1%, then  $y$ , i.e. the mortality decreases by  $\hat{\beta}_1 = 0.56\%$ . In other words,  $\beta_1$  characterizes the relative change in the response  $y$  per unit of relative change in  $x$ .

For obtaining simple predictions of the infant mortality, we can use the regression model on the transformed scale, and then just re-exponentiate to invert the log-transformation:

$$\hat{y} = \exp(\hat{y}')$$

However, some care is required: due to the skewness in the lognormal distribution, the above is an estimate for the median of the conditional distribution  $y|x$ , but not for its mean  $E[y|x]$ . Often, the difference is relatively small and neglecting it will not make much difference. However, in cases where we unbiased prediction is key, we need to apply a different procedure. There are two options for generating sound predictions. The first one is base on the theoretical correction factor that is motivated by the formula for the expected value of the lognormal distribution:

$$\hat{y} = \exp(\hat{y}' + \hat{\sigma}_E^2 / 2)$$

An alternative is to apply the smearing estimator first presented by Duan (1983). This is a more empirically motivated back-transformation with the aim to generate unbiased predictions on the original scale. The formula is as follows:

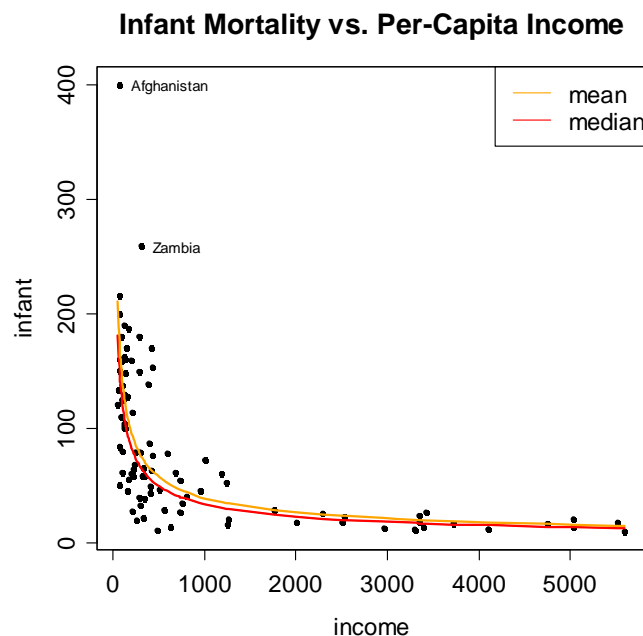
$$\hat{y} = \exp(\hat{y}') \cdot \frac{1}{n} \sum_{i=1}^n \exp(r'_i)$$

There is some scientific debate as to which of the two variants yields better results. The answer is ambiguous and depends on the true distribution of the error term. It is generally accepted that in case of violation of the Gaussianity assumption on the transformed scale, the smearing estimator will perform better. In both cases, owing to the exponential back-transformation, the fit on the original scale cannot take negative values. This is another aspect that here strongly speaks for fitting on the log-log-scale. A model that predicts negative values for infant mortality would not be plausible in practice.

For the confidence and prediction intervals, we can simply compute these as usual on the transformed scale. Simple re-exponentiating brings them back to the original scale. There is no need for a correction factor as we are dealing with quantiles of the respective distributions:

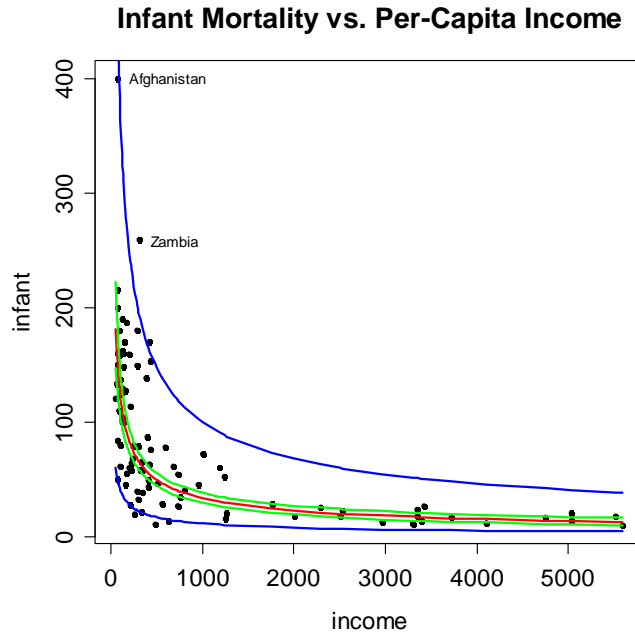
$$[lo, up] \rightarrow [\exp(lo), \exp(up)]$$

```
> ## Predictions
> po <- exp(predict(fit))
> poc <- exp(predict(fit)+(summary(fit)$sigma^2)/2)
>
> ## Scatterplot with Fitted Curves
> plot(infant ~ income, data=im, pch=20)
> lines(sort(im$income), po[order(im$income)], col="red")
> lines(sort(im$income), poc[order(im$income)], col="orange")
```



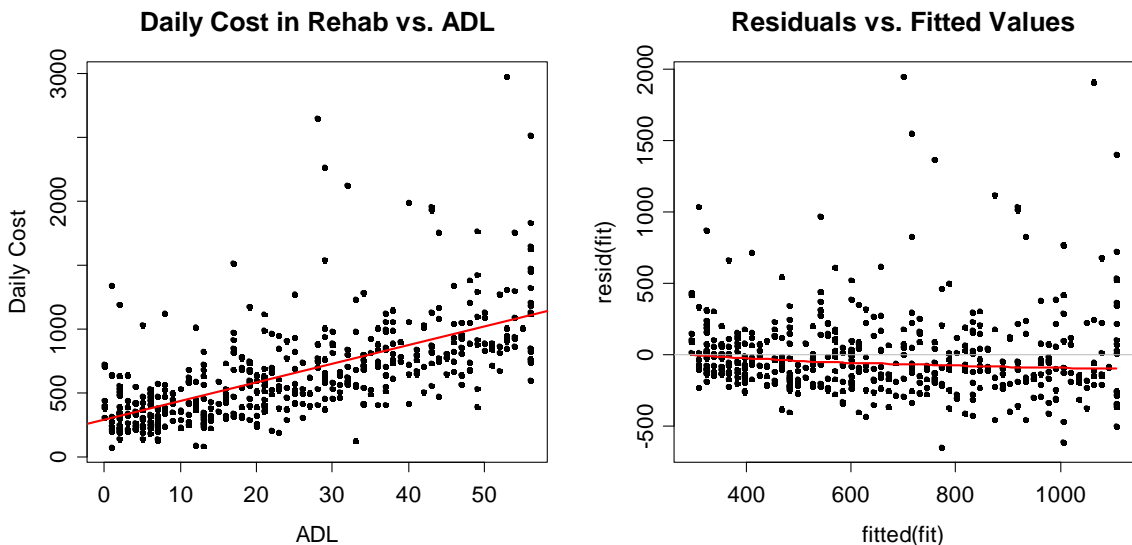
Again, an important advantage of the log-log-model is that neither of these intervals does take negative values on the original scale. Moreover, they are no longer symmetric, reflecting the fact that there is more room for error towards bigger values, and less towards smaller errors.

```
> ## Computing and Plotting the Intervals
> ci <- exp(predict(fit, interval="confidence"))
> pi <- exp(predict(fit, interval="prediction"))
>
> plot(infant ~ income, data=im, pch=20)
> lines(sort(im$income), po[order(im$income)], col="red")
> lines(sort(im$income), ci[order(im$income),2], col="green")
> lines(sort(im$income), ci[order(im$income),3], col="green")
> lines(sort(im$income), pi[order(im$income),2], col="blue")
> lines(sort(im$income), pi[order(im$income),3], col="blue")
```



### 2.6.5 The Logged Response Model

This far, we considered log-transformations for both variables, as well as for the predictor only. If one sees this as a trick, rather than having a specific model formulation in mind, we might try to work with a logged response but the original predictor. As it turns out, also this model is widely used and accepted in practice. We illustrate it with the following example:



The data originate from a research project of the author. The goal was to study the *daily cost in neurological rehabilitation*. In seven hospitals, a random sample of 473 patients was studied, most of whom were originally suffering from *craniocerebral injuries* or *apoplectic strokes*. The total (time) effort for care, therapy and medical examinations was measured, expressed as CHF/day and

serves as the response variable. Also, for each patient an ADL assessment was taken. It is based on about 20 items that quantify the autonomy of a patient in the *activities of daily life*, i.e. personal hygiene, feeding, etc..

Above, the data are visualized in a scatterplot. A simple linear regression model had been fitted, along with a Tukey-Anscombe plot for judging the quality of the fit. At first impression, the straight line does not fit too badly, but a closer inspection shows that there is a bias (i.e. *non-zero expectation for the error*), and a *right-skewed error distribution*. These are strong model violations, and thus, the simple linear model yields a poor explanation of the daily rehabilitation cost. As a way out, we suggest to log-transform the response variable, but to leave the predictor as is:

$$y' = \log(y), x' = x$$

This simple trick yields a good fit, see below. Also, we will soon outline that the log-transformation is indicated for any right-skewed variable such as cost, whereas the uniformly distributed ADL predictor does not require action. The model is:

$$y' = \log(y) = \beta'_0 + \beta'_1 x + E', \text{ respectively,}$$

if we back-transform such that the response is on the original scale:

$$y = \exp(\beta'_0) \cdot \exp(\beta'_1 x) \cdot \exp(E') = \beta_0 \cdot \beta_1^x \cdot E.$$

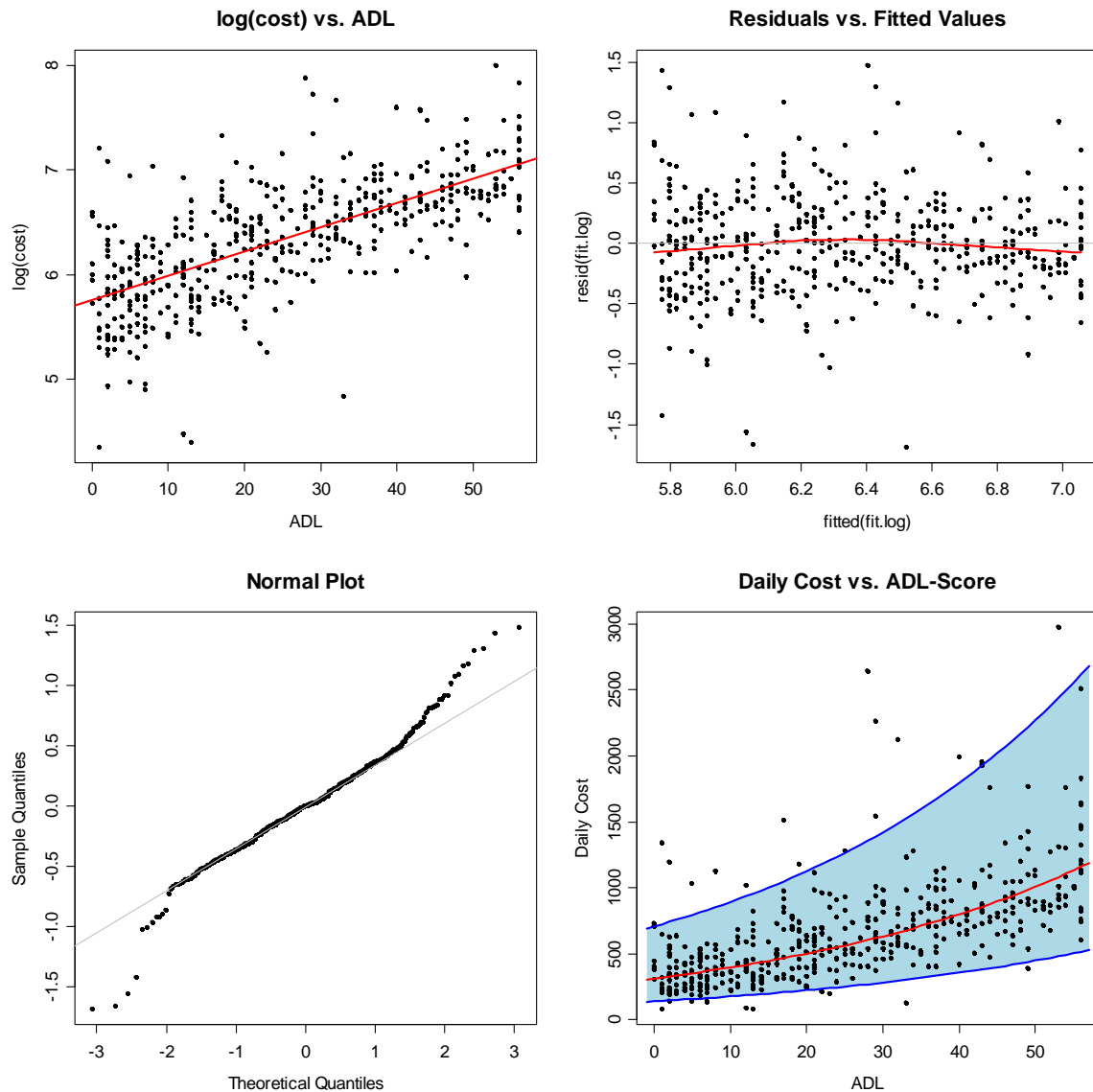
What we obtain is an exponential function, fundamentally different from the power law that results from the log-log-model. The two parameters  $\beta_0$  and  $\beta_1$  control the scale respectively the curvature. The usual assumption for the error is  $E' \sim N(0, \sigma_{E'}^2)$ , and thus, we again have a multiplicative lognormal error term on the original scale. This results in right-skewed scatter that increases with increasing daily cost, matching what we observe in the data. We obtain the fitted coefficients in R:

```
> lm(log(cost) ~ adl, data=rehabilitation)
(Intercept)          adl
    5.75106         0.02331
```

The interpretation is as follows: *an increase by one unit in the predictor  $x$  multiplies the fitted value by  $\exp(\beta'_1)$* . In our case, one additional ADL point, meaning less autonomy of the patient, increases the cost on average by a factor of  $\exp(0.02331) = 1.023584$ , i.e. 2.36%. We then display fit, diagnostics and prediction interval, see next page.

It turns out that after the transformation, a straight line provides a reasonable fit. Still, the Tukey-Anscombe plot exhibits a slight bias. The residuals follow a symmetric, but prominently long-tailed distribution. Hence, not all assumptions for OLS fitting are 100% fulfilled, but the situation is already much, much better than previously, with *daily cost vs. ADL*. Moreover, there are no more simple tricks or transformations with which we could improve the model further.





As a side note, we remark that further model improvement is possible here by using advanced methods such as Box-Cox transformations, or a generalized linear model based on the Gamma distribution. These topics are (far) beyond the scope of this introductory section on simple linear regression and thus not discussed here. It is also important to mention that while they are beneficial to the quality of the prognosis interval and the parameter tests, they do not improve the precisions of the point forecasts much.

## 2.6.6 When and How to Log-Transform

From the above examples, it is evident that variable transformations lead to novel predictor-response relations, often strongly improve the fit and are of tremendous importance to many applied regression problems. Thus, when and how to transform? Long-time practical experience has led to a few simple guidelines. A log-transformation of a variable, i.e.  $x' = \log(x)$  and/or  $y' = \log(y)$  is indicated and often very beneficial for the model fit if:

- Practice dictates that the regression line must run through  $(0/0)$ . In that case, both predictor and response require a log-transformation.
- Generally if a variable is “on a relative scale”, i.e. a change from 10 to 11 does not mean the same or have the same impact as from 100 to 101, but we rather need to care about the relative/percentage increase.
- Variables that are on a scale that is left-closed with zero as the smallest possible value, but open to the right so that it can theoretically take arbitrarily large values are often on a relative scale.
- If the marginal distribution of a variable, as we can observe it from a histogram, is clearly skewed to the right. This is often the case for the above-mentioned positive variables on a relative scale.

In summary, I dare to say that using the log-transformation is almost the norm rather than the exception when we talk about linear modeling. On the other hand, there are also variables where a transformation would be wrong, or is not possible at all. The latter concerns all variables that take negative values and even when there are zero values, we may run into problems, because the logarithm is defined for strictly positive values  $x, y > 0$  only. In summary:

- For predictor/response variables that take negative values, the log-transformation, and hence the log-log model is typically not suitable.
- If either  $y = 0$  or  $x = 0$  appears, the log-transformation is still not possible. Do not exclude these data points from the analysis, this leads to a systematic error. One can though additively shift the variable:  $x \leftarrow x + c$ .
- The usual choice for the constant is  $c = 1$ . However, this makes the regression model no longer invariant versus scale transformations. Thus, it is better (and recommended) to set  $c$  to the smallest value  $> 0$ .

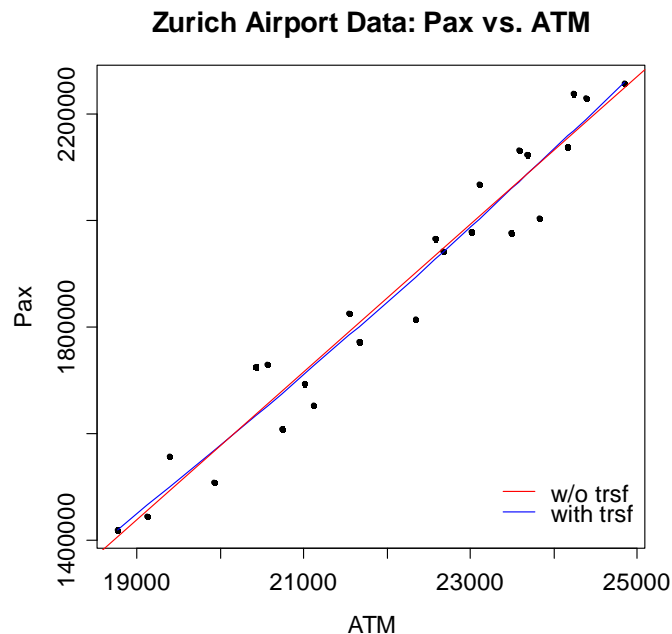
## 2.6.7 Final Considerations

By reflecting the previous examples, we notice that in the Leinhardt data both *infant mortality* and *income* are right-skewed variables which only take positive values. Thus, a log-transformation needs to be considered for both, and as the results from section 2.6.4 show, yields good results. Moreover, the *daily cost in neurological rehabilitation* is right-skewed and positive, while the predictor *ADL* is not. Hence only the response was log-transformed, again with good outcome. Finally, we turn our attention back to the Zurich Airport example. One aspect is that the residual plots in section 2.3.4 raised some doubts whether the straight line is a trustworthy result. And then, both *Pax* and *ATM* are positive variables what makes them candidates for a transformation.

$$ATM' = \log(ATM), Pax' = \log(Pax)$$

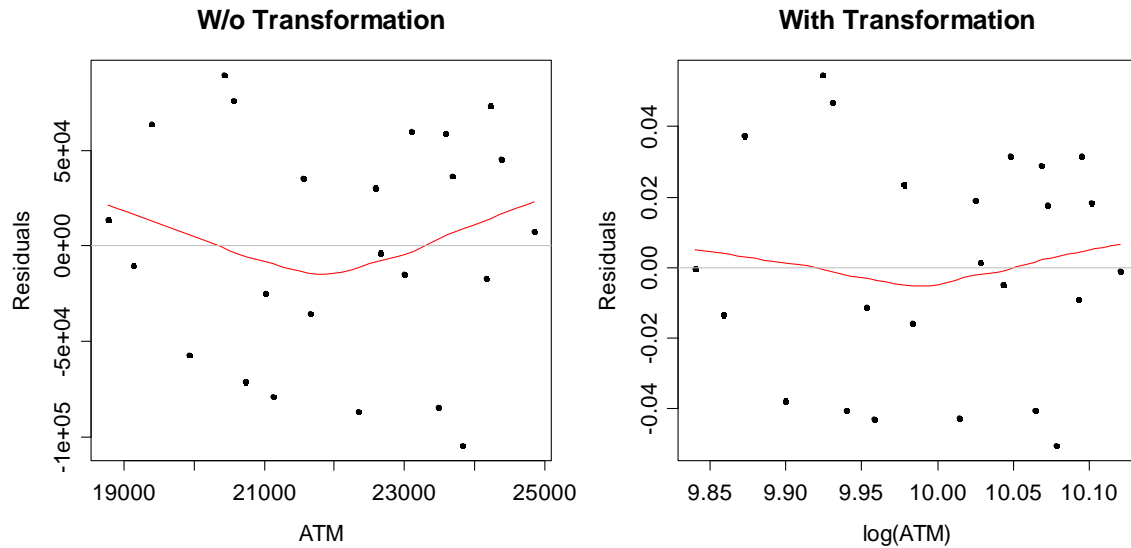
The code for fitting the model and producing a scatterplot is:

```
> fit <- lm(Pax ~ ATM, data=unique2010)
> fit.log <- lm(log(Pax) ~ log(ATM), data=unique2010)
> fit.y.orig <- exp(fitted(fit.log)[order(unique2010$ATM)])
> plot(Pax ~ ATM, data=unique2010, main="...")
> lines(sort(unique2010$ATM), fit.y.orig, col="blue")
> abline(fit, col="red")
```



The result no longer corresponds to a straight line into the scatterplot, but a curve. Additionally, the increase in  $Pax$  is no longer linear with  $ATM$ , but relative. The difference between the two solutions seems to be minimal. Still, the variable transformations improve, as we can see from the residual plots:

```
> xx <- unique2010$ATM
> yy <- residuals(fit)
>
> ## Residuals vs. Predictor w/o Transformation
> plot(xx, yy, xlab="ATM", ylab="Residuals", main="...")
> lines(loess.smooth(xx,yy),col="red")
> abline(h=0, col="grey")
>
> ## Residuals vs. Predictor w/ Transformation
> xx <- log(unique2010$ATM)
> yy <- residuals(fit.log)
> plot(xx, yy, xlab="log(ATM)", ylab="Residuals", main="...")
> lines(loess.smooth(xx,yy),col="red")
> abline(h=0, col="grey")
```



The log-log-model manages to reduce the bias of the plain linear one, although there is still some kink in the residuals. But the log-log-model has another attractive point: it does no longer predict negative *Pax* values - though that does not mean it is safe for extrapolation! The coefficients are:

```
> lm(log(Pax) ~ log(ATM), data=unique2010)
Coefficients:
(Intercept)      log(ATM)
      -2.116         1.655
```

Thus, the fitted relation corresponds to:

$$y = \exp(-2.116) \cdot x^{1.655}, \text{ resp. } Pax = 0.120 \cdot ATM^{1.655}$$

So, if *ATM* increases by 1%, then *Pax* increases by 1.655%. That is at least as plausible as an increase of 138.8 passengers per additional flight, because it is well known that the seat load factor is higher and bigger airplanes are used in busy times with more air traffic movements.

## 3 Multiple Linear Regression

It is very rare that the variation in a response variable  $y$  is due to one single predictor only. Even for the relatively trivial *Pax vs. ATM* example, the seat load factor and the amount of cargo that is handled may play an important role, too. For the other examples that were considered in section 1.1, the dependency on several input variables was clearly pointed out. We will now address the methodology for estimating multiple linear regression models where:

$$y = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p + E.$$

We will continue using OLS for estimating the coefficients  $\beta_0, \dots, \beta_p$ . However, a number of new issues arise here; the most important perhaps being the fact that visualizing the relation is no longer easily possible. Thus, understanding the input and output becomes an important and challenging task.

### 3.1 Example: Air Pollution and Mortality

Since the beginning of the environmental movement, attention has focused on the protection of human health. Soon, air pollution was identified as a major threat to well-being. Therefore, researchers at General Motors collected data on 59 US Standard Metropolitan Statistical Areas for a study whether air pollution contributes to the age-adjusted mortality of the population. The *apm* dataset includes predictors measuring demographic characteristics of the cities, variables measuring climate parameters and finally three records for the air pollution in the ambient air: concentrations of hydrocarbons ( $HC$ ), nitrous oxide ( $NO_x$ ) and sulfur dioxide ( $SO_2$ ). An excerpt of the data is as follows:

City	Mortality	JanTemp	JulyTemp	RelHum	Rain	Educ	Dens	NonWhite	WhiteCollr	Pop	House	Income	HC	NOx	SO2
Akron, OH	921.87	27	71	59	36	11.4	3243	8.8	42.6	660328	3.34	29560	21	15	59
Albany, NY	997.87	23	72	57	35	11.0	4281	3.5	50.7	835880	3.14	31458	8	10	39
Allentown, PA	962.35	29	74	54	44	9.8	4260	0.8	39.4	635481	3.21	31856	6	6	33
Atlanta, GA	982.29	45	79	56	47	11.1	3125	27.1	50.2	2138231	3.41	32452	18	8	24
Baltimore, MD	1071.29	35	77	55	43	9.6	6441	24.4	43.7	2199531	3.44	32368	43	38	206
Birmingham, AL	1030.38	45	80	54	53	10.2	3325	38.5	43.1	883946	3.45	27835	30	32	72
Boston, MA	934.70	30	74	56	43	12.1	4679	3.5	49.2	2805911	3.23	36644	21	32	62
Bridgeport, CT	899.53	30	73	56	45	10.6	2140	5.3	40.4	438557	3.29	47258	6	4	4
Buffalo, NY	1001.90	24	70	61	36	10.5	6582	8.1	42.5	1015472	3.31	31248	18	12	37
Canton, OH	912.35	27	72	59	36	10.7	4213	6.7	41.0	404421	3.36	29089	12	7	20
Chattanooga, TN	1017.61	42	79	56	52	9.6	2302	22.2	41.3	426540	3.39	25782	18	8	27
Chicago, IL	1024.89	26	76	58	33	10.9	6122	16.3	44.9	606387	3.20	36593	88	63	278
Cincinnati, OH	970.47	34	77	57	40	10.2	4101	13.0	45.7	1401491	3.21	31427	26	26	146
Cleveland, OH	985.95	28	71	60	35	11.1	3042	14.7	44.6	1898625	3.29	35720	31	21	64
Columbus, OH	958.84	31	75	58	37	11.9	4259	13.1	49.6	124833	3.26	29761	23	9	15
Dallas, TX	860.10	46	85	54	35	11.8	1441	14.8	51.2	1957378	3.22	38769	1	1	1

Most of the variables are self-explanatory: the temperatures are averages in degrees Fahrenheit, humidity is a percentage, the rainfall is given as annual sum in inches, education is the median number of years in the population, which itself is given as an absolute number, as well as a density per area and housing unit. Moreover, we have the percentages of non-white inhabitants and white collar workers, the median per-capita income and finally the concentrations of the pollutants.

The task is to study how *air pollution contributes to mortality*. Thus, the influence of the three pollution variables is of primary interest. The remaining ones can be seen as potentially confounding factors, for which we try to correct. Since we know that mortality is affected by other causes than just the pollution alone, we have to correct for the effect of these covariates. Just studying the relation between mortality and pollution would lead to flawed results. Fortunately, with multiple linear regression we can incorporate all covariates and derive sound conclusions.

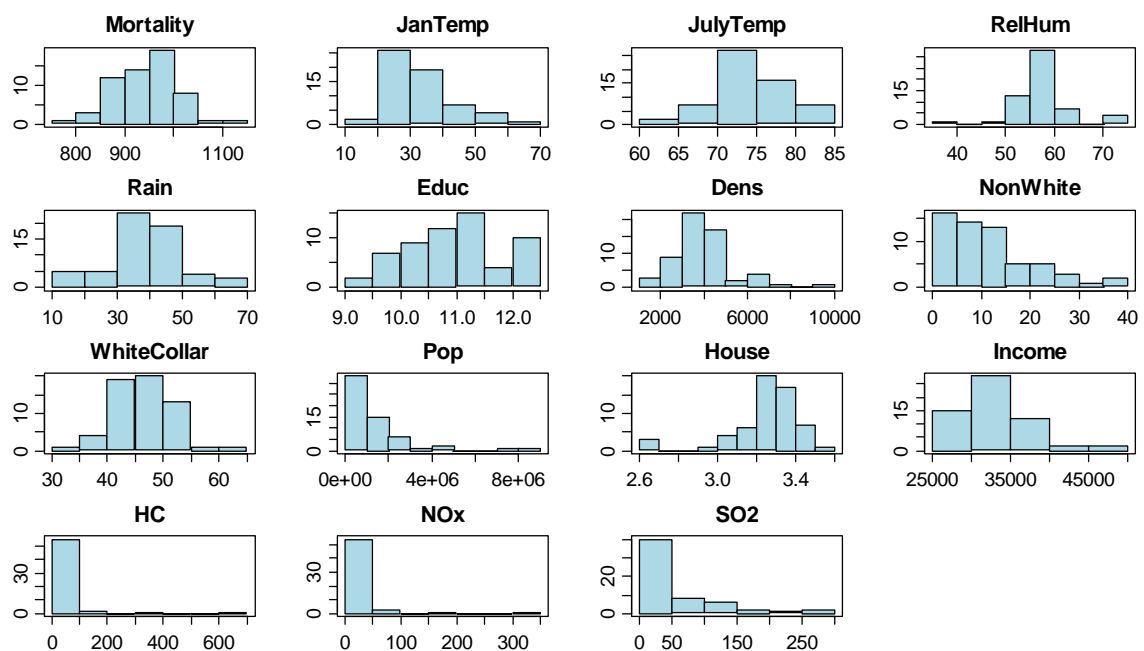
## 3.2 Preparing the Data

For simple regressions, we were able to visualize the data in an  $xy$ -scatterplot. This was beneficial for identifying the correct response-predictor relation, making variable transformations, detecting outliers and some further potential problems. In the present example, the data live in a 15-dimensional space, and there is no plot that can show them in full generality. Still, gaining an impression of the data and preparing them well for regression analysis is absolutely essential.

### 3.2.1 Marginal Plots

As a way out, we can visualize the univariate distribution of response and predictors with histograms (or barplots, should there be categorical predictors). As mentioned above, this does not give the full multivariate picture, but it still allows for detecting skewness in the variables, the presence of outliers and perhaps other important specialties such as missing values that are coded with numerical values.

```
> par(mfrow=c(4,4))
> for (i in 1:15) hist(apm[,i], main=names(apm)[i])
```



What immediately catches the attention is the extreme skewness of the pollution variables. This needs to be addressed with variable transformations; else the results from a multiple linear regression will be poor. Furthermore, also the population is right-skewed. Apart from this, there do not seem to be too many peculiarities in the *apm* data. An analysis using the R command

```
> any(is.na(apm))
[1] FALSE
```

shows that there are no missing values coded by NA. Neither do we have any suspicions that they might be coded by some numerical value. If that was the case, we urgently need to clarify the issue, and set the respective values to NA. Besides the histograms, one could also do scatterplots of the response variable vs. each of the predictors (or boxplots, in case of categorical predictors). Again, this does not visualize the multivariate setting in full depth, and is mostly less useful than the histograms shown above.

### 3.2.2 Variable Transformations

Regression results will be much easier to understand if the data are in units that we are well familiar with. In the context of the mortality example that means converting the temperatures to degrees Celsius rather than Fahrenheit, and rainfall in *cm/year* rather than *inches/year*. We copy the original data frame, generate the new variables and drop the old ones:

```
> apm$JanTemp <- (5/9)*(apm$JanTemp-32)
> apm$JulyTemp <- (5/9)*(apm$JulyTemp-32)
> apm$Rain <- (2.54)*apm$Rain
```

All of the above are linear variable transformations of the form  $x' = ax + b$ . It is very important to notice that these do not change the regression output: all fitted values, tests and the prediction interval will remain identical. The only thing that changes is the coefficient  $\beta_j$  and its standard error, but only to account for transformation that was made.

This is clearly not the fact for non-linear transformations such as the log (or also the square root, the inverse, etc.): they ultimately change the regression relation and all results (fitted values, tests, confidence intervals, ...) will be different. The change is not necessarily for the bad, and thus we carry out the transformations that are indicated on the *apm* data. That includes taking the  $\log(\cdot)$  for the three pollution variables plus the population. Most other predictors are annual sums or averages, show sufficiently symmetrical distribution and are left alone.

Implementation-wise, we do not carry out these transformations in the data frame, but choose the convenient option of writing the  $\log(\text{Pop})$ ,  $\log(\text{HC})$ ,  $\log(\text{NOx})$  and  $\log(\text{SO}_2)$  terms directly into the model equation, see below.

### 3.3 Model and Estimation

What to do with such cases, where multiple predictor variables are available? The poor man's approach would be to do many simple linear regressions on each of the predictors separately. This has the somewhat doubtful advantage that the relation between each predictor and the response can be displayed in a two-dimensional scatterplot. However, it is very important to note that *doing many simple linear regressions is not equivalent to a multiple linear regression*. The findings, i.e. the regression coefficients and their p-values, will generally be different. The only case when they are identical is if the predictors are exactly orthogonal; and this is almost never the case with data from observational studies.

As indicated above, the appropriate tool for simultaneously including the effects of several predictors at a time is *multiple linear regression*. Geometrically speaking, one tries to fit the least squares hyperplane in the  $(p+1)$ -dimensional space ( $p$  is the number of predictors). Generally, this fit cannot be visualized if  $p > 2$ . We start our discussion with a simple example that illustrates some of the peculiarities of multiple linear regression.

#### Example

In this artificial example, there are only 2 predictors and 8 observations. Because the optimal solution is obvious, we do not need to estimate the regression coefficients but can guess them. The data are as follows:

Observation	x1	x2	yy
1	0	-1	1
2	1	0	2
3	2	1	3
4	3	2	4
5	0	1	-1
6	1	2	0
7	2	3	1
8	3	4	2

The optimal solution of the multiple regression problem for the above data is

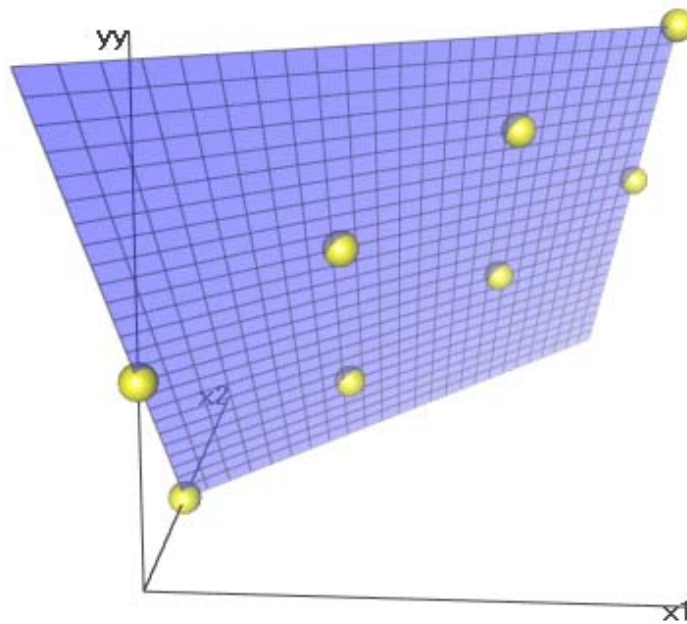
$$y_i = 2x_{i1} - x_{i2} \text{ for all } i = 1, \dots, 8$$

We are in a very special situation and have a perfect fit, thus there are no errors.

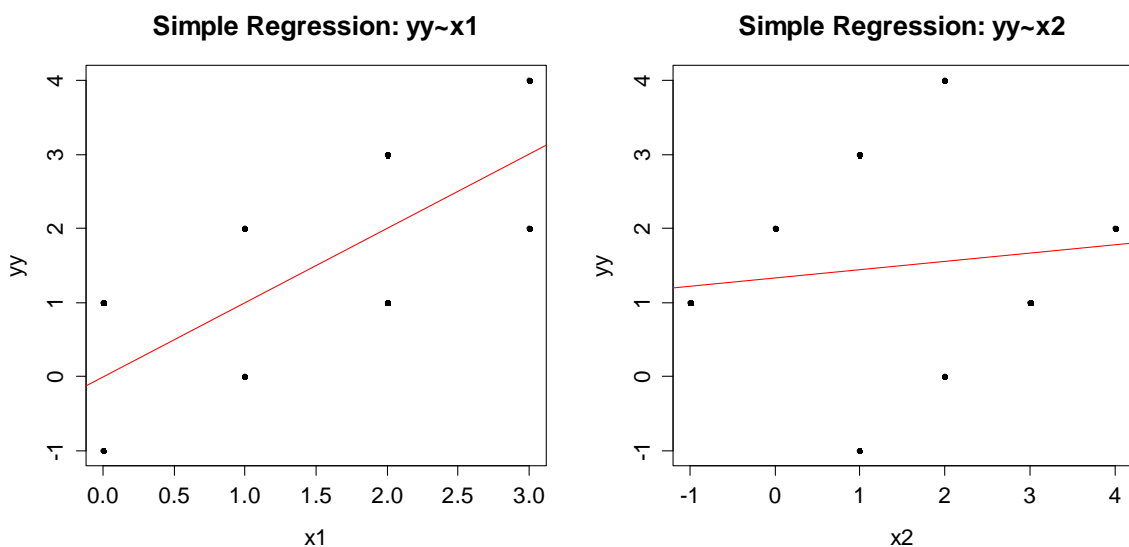


Because there are only two predictors plus the response, we can visualize the fit in a 3d-scatterplot. As we observe below, the data points lie in a plane, the regression plane.

```
> toy.ex <- data.frame(x1=c( 0,1,2,3, 0,1,2,3),
                       x2=c(-1,0,1,2, 1,2,3,4),
                       yy=c( 1,2,3,4,-1,0,1,2))
> library(Rcmdr)
> attach(toy.ex)
> scatter3d(yy ~ x1 + x2, axis.scales=FALSE)
> detach(toy.ex)
```



To convince ourselves that single and multiple linear regression is not one and the same thing, we regress  $y \sim x_1$  and  $y \sim x_2$ . We can visualize these fits in two-dimensional scatterplots.



The slope estimates from the simple regressions turn out to be 1.00 and 0.11, respectively. Hence they are both different than the coefficients for  $x_1$  and  $x_2$  in the (perfect) solution from multiple linear regression. Moreover, we do not achieve a perfect fit in neither of the two simple models. Hence, for describing the variation in  $y$ , we need to build on both variables  $x_1$  and  $x_2$  simultaneously.

### 3.3.1 Notation

We turn our attention back to the mortality example in dataset *apm*. In colloquial formulation, the multiple linear regression model is as follows:

$$\text{Mortality}_i = \beta_0 + \beta_1 \cdot \text{JanTemp}_i + \beta_2 \cdot \text{JulyTemp}_i + \dots + \beta_{14} \cdot \log(\text{SO}_2)_i + E$$

More generally and technically, the multiple linear regression model specifies the relation between response  $y_i$  and predictors  $x_{i1}, \dots, x_{ip}$  for observations  $i = 1, \dots, n$ , including a random error term  $E_i$ . The double index notation is defined as:

$$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip} + E_i, \text{ for } i = 1, \dots, n.$$

The term  $\beta_0$  is still called *intercept* and corresponds to the (theoretical) mortality value when all predictors  $x_{i1} = x_{i2} = \dots = x_{ip} = 0$ . The remaining parameters  $\beta_1, \dots, \beta_p$  are, in contrast to simple regression, no longer called slope(s), but just *regression coefficients*. The interpretation is as follows:

*The regression coefficient  $\beta_j$  is the increase in the response  $y$  when predictor  $x_j$  increases by 1 unit, but all other predictors remain unchanged.*

A more convenient way of writing down a multiple linear regression model is with the so-called matrix notation. It is simply:

$$y = X\beta + E, \text{ with } y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, X = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, E = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{pmatrix}.$$

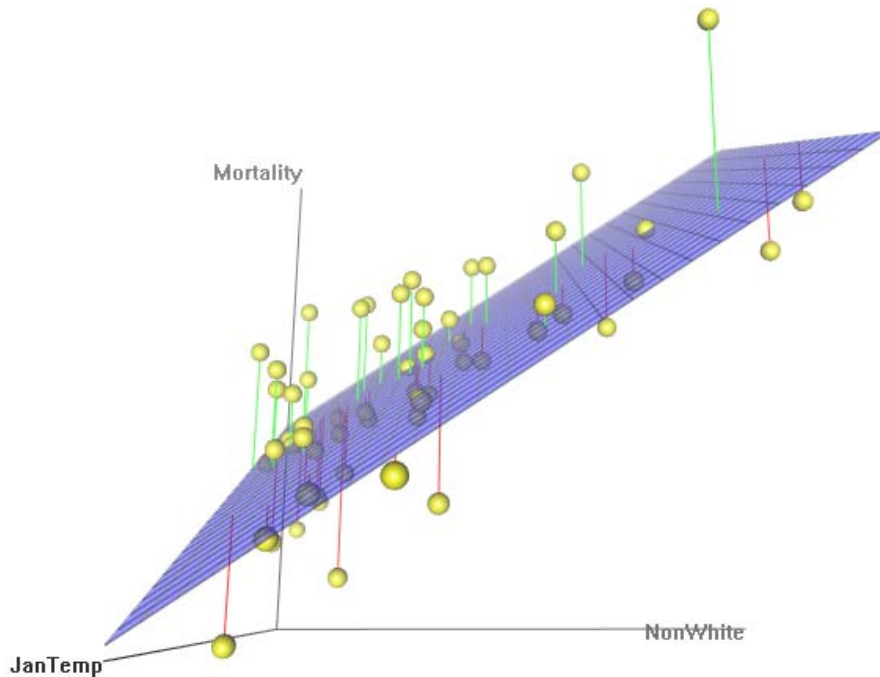
The terms in this equation are called the *response vector*, the *design matrix*, the *coefficient vector* and the *error vector*. If a matrix multiplication is carried out and the result is written down, we are back with the double index notation. This also illustrates the role of the particular first column of the design matrix: it is the intercept, which is also part of multiple linear regression.

Our next goal is to fit a multiple linear regression model. The task which needs to be done is to estimate the coefficient vector  $\beta$  from the data; in a way that the solution is optimal. The criterion is still to minimize the sum of squared residuals. The next section illustrates the concept with an example and then focuses on the solution plus some technical aspects.

### 3.3.2 OLS: Method & Identifiability

For illustrating the concept of least squares regression, we consider the mortality data with two predictors only: *NonWhite* and *JanTemp*. The regression coefficients are estimated such that the sum of squared residuals is minimal. The fitted regression plane with the residuals looks as follows:

```
> scatter3d(Mortality~NonWhite+JanTemp, axis.scale=FALSE)
```



We observe that the mortality decreases with higher winter temperatures, and increases in urban regions with more non-white population. The basis for finding this solution lies in the residuals, which are:

$$r_i = y_i - (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}).$$

Then, we choose the parameters  $\beta_0, \dots, \beta_p$  such that the sum of squared residuals is minimal. We again formulate the quality function.

$$Q(\beta_0, \beta_1, \dots, \beta_p) = \sum_{i=1}^n r_i^2 = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}))^2$$

We need to minimize this function, which can be tackled by taking partial derivatives and setting them to zero. This results in the so-called *normal equations*. We do now take full advantage of the matrix notation that was introduced above and can write the normal equations as

$$(X^T X)\beta = X^T y.$$

If  $X^T X$  is *invertible* (or *regular*), we can obtain the least squares estimates of the regression coefficients by some simple matrix calculus as  $\hat{\beta} = (X^T X)^{-1} \cdot X^T y$ .

If the regularity condition for  $X^T X$  is fulfilled, there is a unique and explicit solution for the regression coefficients  $\hat{\beta}$ , and thus no numerical optimization is needed. A side remark: in software packages, the inverse of  $X^T X$  is usually not computed for numerical reasons, but the computations will be based on a  $QR$ -decomposition or similar methods of simplifying  $X^T X$ . In R, multiple linear least squares regression is carried out with command `lm()`. The syntax is as follows:

```
fit <- lm(Mortality ~ JanTemp + JulyTemp + RelHum + Rain +
          Educ + Dens + NonWhite + WhiteCollar +
          log(Pop) + House + Income + log(HC) +
          log(NOx) + log(SO2), data=apm)
```

As in simple linear regression, we have the response variable on the left hand side. It is related to the predictors on the right hand side, which are joined by '+' signs. Note that potential log-transformations of predictors and/or response can directly be written into the formula, and that we need to specify the data frame from which the variables need to be taken.

It is worth noting that there is a simple variant of specifying regression problems with many predictors in R. The notation `lm(Mortality ~ ., data=apm)` means that mortality is explained by all the other variables that exist in data frame *apm*. However, in our example these two commands will not yield identical results, because of the log-transformations that are missing in the short notation. Once the model is fitted, we can extract the regression coefficients, here rounded to two digits, by:

```
> round(coef(fit), 2)
(Intercept)      JanTemp      JulyTemp      RelHum      Rain
    1297.38         -2.37         -1.75         0.34         1.49
      Educ         Dens      NonWhite  WhiteCollar  log(Pop)
    -10.00         0.00         5.15         -1.88         4.39
      House      Income      log(HC)      log(NOx)      log(SO2)
    -45.74         0.00        -22.04        33.97         -3.69
```

We claimed above that the normal equations have a unique solution if and only if  $X^T X$  is regular and thus invertible. This is the case if  $X$  has full rank, i.e. all columns of the design matrix, or in other words, all predictor variables are linearly independent. This is often the case in practice, and whenever the full rank condition for  $X$  is fulfilled, we are fine.

On the other hand, there will also be cases where  $X$  does not have full rank and  $X^T X$  is singular. Then, there are usually infinitely many solutions. Is this a problem? And how does it occur? The answer to the first question is "yes". When the design matrix  $X$  does not have full rank, the model is "poorly formulated", such that the regression coefficients  $\beta$  are at least partially unidentifiable. It is mandatory to improve the design, in order to obtain a unique solution, and regression coefficients with a clear meaning. Below, we list some typical mistakes that lead to a singular design.

### 1) Duplicated variables

It could be that we use a person's height both in meters and centimeters as a predictor. This information is redundant, and the two variables are linearly dependent. One thus has to remove one of the two.

### 2) Circular variables

Another example is when the number of years of pre-university education, the number of years of university education and also the total number of years of education are recorded and included in the model. These predictors will be linearly dependent, thus  $X$  does not have full rank.

### 3) More predictors than cases

Note that a necessary (but not sufficient) condition for the regularity of  $X^T X$  is  $p < n$ . Thus, we need more observations than we have predictors! This makes sense, because the regression is over-parameterized (or super-saturated) else and will not have a (unique) solution.

## What does R do in non-identifiable problems?

Generally, statistics packages handle non-identifiability differently. Some may return error messages; some may even fit models because rounding errors kill the exact linear dependence. R handles this a bit different: it recognizes unidentifiable models and fits the largest identifiable one by removing the excess predictors in reverse order of appearance in the model formula. The removed predictors will still appear in the summary, but all their values are NA, and a message also says "Coefficients: k not defined because of singularities"). While this still results in a fit, it is generally better in such cases to rethink the formulation of the regression problem, and remove the non-needed predictors manually.

## Estimation of the Error Variance

An additional quantity that is a necessary ingredient for all tests and confidence intervals needs to be estimated from the data: it is the error variance  $\sigma_E^2$ . The estimate can be obtained by standardizing the sum of squared residuals with the appropriate degrees of freedom, which is the number of observations  $n$  minus the number of estimated parameters. With  $p$  predictor variables and an intercept, this amounts to  $p + 1$ , and the error variance estimate is:

$$\hat{\sigma}_E^2 = \frac{1}{n - (p + 1)} \sum_{i=1}^n r_i^2.$$

In the next section, we will discuss if and when the OLS results are a good solution. The assumptions are identical to the ones we had in simple linear regression, as is the main result, the Gauss-Markov theorem. By assuming a Gaussian distribution for the errors, we can show even more and lay the basis for inference in multiple linear regression.

### 3.3.3 Properties of the Estimates

The use of the least squares procedure is attractive due to its simplicity and the explicit solution that can be found without any numerical optimization. Additionally, there are some mathematical optimality results that further justify its application. However, we require some conditions for being able to derive them, namely:

$$E[E_i] = 0.$$

Again this means that there is no systematic error, i.e. the true relation between predictors and response is the linear function that we imposed. Or in other words: the hyper plane is the correct fit. Additionally, we require constant variance of the error term, i.e.

$$\text{Var}(E_i) = \sigma_E^2.$$

Finally, there must not be any correlation among the errors for different instances, which boils down to the fact that the observations, respectively their errors, do not influence each other, and that there are no latent variables (e.g. time/sequence of the measurements) that do so. In particular,

$$\text{Cov}(E_i, E_j) = 0 \text{ for all } i \neq j.$$

Under these three conditions, we can derive that the coefficient estimates are unbiased and find their covariance matrix. The *Gauss-Markov theorem* states that there is *no other linear, unbiased estimator that is more efficient*.

$$E[\hat{\beta}] = \beta \text{ and } \text{Cov}(\hat{\beta}) = \sigma_E^2 \cdot (X^T X)^{-1},$$

As in simple linear regression, the precision of the regression coefficients depends on the design and the number of observations which are present. While the Gauss-Markov theorem does not require the assumption of normally distributed errors  $E_i$ , be careful in case of clearly non-Gaussian distribution. On one hand, there may be non-linear estimators that are clearly more efficient than OLS, and even more importantly, all inference results (i.e. *tests, confidence intervals, prediction interval*) to be discussed below ultimately require independent Gaussian errors. Hence it is standard to also require

$$E_i \text{ i.i.d. } \sim N(0, \sigma_E^2)$$

for OLS regression. Then, and only then, the estimators for the regression coefficients will follow an exact Gaussian distribution, as will the distribution of the fitted values. The specifications are as follows:

$$\hat{\beta} \sim N(\beta, \sigma_E^2 (X^T X)^{-1}) \text{ and } \hat{y} \sim N(X\beta, \sigma_E^2 X (X^T X)^{-1} X^T)$$

For error distributions that deviate from the Gaussian, we can rely on the central limit theorem. It tells us that asymptotically (i.e. for large samples) the normal distribution of the estimates will still hold. Thus, small deviations from Gaussian

errors may be tolerable in practice. It is generally an expert call what is alarming and what is acceptable, but the bigger the dataset and the less extreme the error distribution deviates, the more tolerable one can be. Also, deviations from normal errors are usually less worrying if the task is prediction, but more so if one is after inference with exact p-value reporting.

As mentioned above, both  $\hat{\beta}$  and  $\hat{y}$  are unbiased estimates and since their covariance matrices and distribution is known, confidence intervals and tests can be determined. Another important result from mathematical statistics is also that under Gaussian distribution, OLS is the *maximum likelihood estimator* (MLE). Hence there cannot be any other unbiased estimator that is asymptotically more efficient than OLS. Please note that this statement is stronger than the Gauss-Markov theorem, but it requires more, namely normal errors.

In summary, there are very good reasons to prefer OLS over other methods to estimate the linear regression coefficients. However, we require that the four assumptions made are at least roughly fulfilled. This needs to be verified by a number of model diagnostic plots, as shown in section 3.7 of this scriptum. In case of clear violations, one usually tries to improve the model with variable transformations, which rightly done serves to achieve better behaved errors. Alternatively, more complicated estimation procedures that require fewer assumptions can sometimes be used instead.

### Hat Matrix

For the mathematically interested, we will now take further advantage of the matrix notation and study the solution of the OLS algorithm. We can write the fitted values  $\hat{y}$  very simply as

$$\hat{y} = X \hat{\beta}.$$

We now do some further calculus and plug-in the solution for  $\hat{\beta}$  from above. We then observe that the fitted values  $\hat{y}$  are obtained by a matrix product, namely the *hat matrix*  $H$ , with the observed response values  $y$ :

$$\hat{y} = X \hat{\beta} = X (X^T X)^{-1} X^T y = H y$$

The matrix  $H$  is called hat matrix, because “it puts a hat on the  $y$ ’s”, i.e. transforms the observed values into fitted values. This clarifies that the OLS estimator is linear and opens the door to a geometrical interpretation of the procedure: the hat matrix  $H$  is the *orthogonal projection* of the response  $y$  onto the space spanned by the columns of the design matrix  $X$ . Please note that (except for some rare cases with perfect fit), we cannot linearly combine the columns of the design matrix to generate the response  $y$ . The OLS solution then is the best approximation, in the sense of an orthogonal projection.

Disclaimer: do not worry if this geometric notion of OLS regression is hard to grasp. It is a nice interpretation for those with imagination and the necessary background in linear algebra, but it is of little practical importance.

## 3.4 Inference

Here, we will discuss some methods for inferring the relation between response and predictor. While a few topics are a repetition to the inference topics in simple linear regression, quite a number of novel aspects pop up, too. Please note that except for the coefficient of determination, the assumption of *independent, identically distributed Gaussian errors* is central to derive the results.

### 3.4.1 The Coefficient of Determination

In simple linear regression, we had presented the coefficient of determination  $R^2$  as an intuitive goodness-of-fit measure that compares the scatter in  $y$ -direction with and without knowing the regression line. Though visualization is no longer possible with multiple linear regression, the idea (and formula) behind is identical:  $R^2$  expresses which portion of the total variation in the response  $y$  is accounted for by the regression hyperplane. The definition is as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \in [0,1]$$

In the numerator, we measure the scatter of the data points around the fitted values, i.e. the *RSS*. The denominator has the scatter of the data points around their mean. This is the *total sum of squares (TSS)*. Again, the maximum value is  $R^2 = 1$ . It is attained if all data points are on the regression hyperplane. The other extreme case is  $R^2 = 0$  and means that there is no explanatory power in the regression fit, and  $\hat{\beta}_1 = \hat{\beta}_2 = \dots = \hat{\beta}_p = 0$ . The actual value is provided in the R summary in the second to last row:

```
> summary(fit)
```

Call:

```
lm(formula = Mortality ~ JanTemp + JulyTemp + RelHum + Rain +
    Educ + Dens + NonWhite + WhiteCollar + log(Pop) + House +
    Income + log(HC) + log(NOx) + log(SO2), data = apm)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	1.297e+03	2.934e+02	4.422	6.32e-05	***
JanTemp	-2.368e+00	8.851e-01	-2.676	0.0104	*
JulyTemp	-1.752e+00	2.031e+00	-0.863	0.3931	
[output partly omitted...]					
log(SO2)	-3.687e+00	7.359e+00	-0.501	0.6189	
---					

```
Residual standard error: 34.48 on 44 degrees of freedom
Multiple R-squared: 0.7685, Adjusted R-squared: 0.6949
F-statistic: 10.43 on 14 and 44 DF, p-value: 8.793e-10
```



The result is  $R^2 = 0.7685$ , hence a good portion of the response variation is explained by the predictors. However, the raw  $R^2$  should be interpreted with care: the more predictors that are added to a multiple linear regression model, the smaller its residual sum of squares becomes, and the higher  $R^2$  is. This improvement may be bigger or smaller according to the predictive power of the added predictor, but the goodness-of-fit never gets worse. This makes the multiple R-squared a cumbersome tool for comparing models with different number of predictors. However, one can overcome this by using the *adjusted R-squared*. The definition is:

$$adjR^2 = 1 - \frac{n-1}{n-(p+1)} \cdot (1 - R^2) \in [0,1]$$

As we can see, there is a penalty term for more complex models, i.e. models where the number of predictors  $p$  is higher. Consequently, the *adjusted R-squared* is always smaller than the *multiple R-squared*. The difference is most pronounced when there are few observations, many predictors and a poor signal. Vice versa, it becomes almost nil if we have lots of observations, just few predictors and strong signal. Final advice in this topic: for not privileging models with excess predictors, we recommend the use of the adjusted R-squared only.

### 3.4.2 Confidence Intervals for the Coefficients

The confidence intervals for the regression coefficients  $\beta_j$ ,  $j=0, \dots, p$  provide a way of expressing the uncertainty in these estimates. They contain all the null hypotheses  $\beta_j = b$  which the corresponding individual hypothesis test fails to reject and hence all values which are plausible for  $\beta_j$ . A quick but approximate way of computing these confidence intervals is:

$$\text{Coefficient Estimate} \pm 2 \cdot \text{Standard Error}$$

The necessary information can be found in the R summary and it is valuable to know about his ad-hoc method for quickly assessing the precision of the estimated coefficients. The actual, precise formula for computing a 95% confidence interval for the regression coefficient  $\beta_j$  is:

$$\hat{\beta}_j \pm qt_{0.975; n-(p+1)} \cdot \hat{\sigma}_{\hat{\beta}_j} = \hat{\beta}_j \pm qt_{0.975; n-(p+1)} \cdot \hat{\sigma}_E \cdot \sqrt{(X^T X)^{-1}_{ii}}$$

Knowing this exact formula by heart is somewhat less important for the practitioner. However, it is important to be familiar with the command `confint()` that computes the exact confidence intervals in R:

```
> round(confint(fit), 2)
              2.5 %   97.5 %
(Intercept)  706.15 1888.61
JanTemp      -4.15  -0.58
JulyTemp     -5.84   2.34
...
```

```
[output partially omitted]
...
log(NOx)      5.26    62.68
log(SO2)     -18.52   11.14
```

As it has been mentioned above, the confidence intervals contain all values which can be seen as plausible for the regression coefficients. If in particular zero lies within the intervals, it is a plausible value, too. Hence it might be that the predictor in question does not contribute to the variation in the response and thus it is non-significant. This leads us to the individual hypothesis tests that will be discussed in the next section.

### 3.4.3 Individual Hypothesis Test

For finding out whether an arbitrary value  $b$  is plausible for the regression coefficient  $\beta_j$ , we can check whether it is contained in the 95%-CI from above. Alternatively, there is a test for the *null hypothesis*  $H_0: \beta_j = b$ . The most popular variant is  $H_0: \beta_1 = 0$ : this is asking if the coefficient could be zero, which would mean that the predictor  $x_j$  has no influence on the response  $y$ . The natural goal is to reject the null for gaining evidence that the relation between  $y$  and the predictor exists. One usually tests two-sided on the 95% level, i.e. the alternative is  $H_A: \beta_1 \neq b$ . The *test statistic* and its *distribution* are as follows:

$$T_{H_0: \beta_j = b} = \frac{\hat{\beta}_j - b}{\hat{\sigma}_{\hat{\beta}_j}} \sim t_{n-(p+1)}.$$

On this basis, it is straightforward to determine acceptance and rejection regions, as well as *p-values*. All the necessary ingredients together with the *test statistic* ( $t$  value) and the *p-value* ( $\Pr(>|t|)$ ) for  $H_0: \beta_j = 0$  are routinely given in the R summary output:

```
> summary(fit)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.297e+03  2.934e+02   4.422 6.32e-05 ***
JanTemp     -2.368e+00  8.851e-01  -2.676  0.0104 *
JulyTemp    -1.752e+00  2.031e+00  -0.863  0.3931
...
[output partially omitted]
...
log(NOx)      3.397e+01  1.425e+01   2.384  0.0215 *
log(SO2)     -3.687e+00  7.359e+00  -0.501  0.6189
```

As an additional example, we test  $\beta_1 = -5$ . The value of the test statistic is  $(-2.368 + 5)/0.8851 = 2.973675$ . The acceptance region is easily computed from R:

```
> qt(0.975, df=44)
[1] 2.015368
```

Hence, we reject the null hypothesis, if the observed value of the test statistic exceeds 2.015 in absolute value. This is the case, and hence  $H_0: \beta_1 = -5$  is rejected. The p-value with which this happens is computed by:

```
> 2*pt(-abs((-2.368+5)/0.8851),df=44)
[1] 0.004760858
```

We conclude that our null hypothesis is quite clearly rejected. While these tests are simply carried out and are useful in practice, their interpretation is a bit tricky and has a few traps that one must not fall victim to, namely:

- 1) The *multiple testing problem*: if we repeatedly do hypothesis testing on the  $\alpha=5\%$  significance level, our total type I error increases. In particular, for  $p$  hypothesis tests, it is  $1-(1-\alpha)^p$ . Note that for example with 30 predictors, the chance of making at least one false rejection in the individual hypothesis tests is already 0.785, a pretty high value!
- 2) It can happen that all individual hypothesis tests fail to reject the null hypothesis (say at the 5% significance level), although it is in fact true that some predictor variables have a known effect on the response. This does often occur due to correlation among the predictor variables, so that the predictive power is distributed and none seems too important in the presence of the others.

Another important point is the interpretation of the individual hypothesis test: it verifies the effect of predictor  $x_j$  on the response in the presence of all the other predictors. As a consequence, any change in the predictor set leads to (sometimes drastically) different test results. This is especially important because decisions about the omitting of variables are often based on the individual hypothesis tests. Due to the above, *one must not drop more than one non-significant variable at a time* – this need be done step-by-step.

### 3.4.4 Comparing Hierarchical Models

The idea behind the test presented in this section is a correct comparison of two multiple linear regression models when the smaller has more than one predictor less than the bigger. This can be useful in practice, i.e. for evaluating whether *air pollution* (which appears as 3 predictors) has an effect on *mortality*. Moreover, the test will also be required for correct handling of categorical predictors, the so-called factor variables (see in section 3.6). We assume that there are two models.

$$\text{Big model: } y = \beta_0 + \beta_1 x_1 + \dots + \beta_q x_q + \beta_{q+1} x_{q+1} + \dots + \beta_p x_p$$

$$\text{Small model: } y = \beta_0 + \beta_1 x_1 + \dots + \beta_q x_q$$

The big model must contain all the predictors that are in the small model, else the models cannot be considered as being hierarchical and the test which is presented

below does not apply. The null hypothesis is that the excess predictors in the big model do not bring any benefit, hence:

$$H_0 : \beta_{q+1} = \beta_{q+2} = \dots = \beta_p = 0$$

We test against the alternative that at least one of the excess predictors has an effect, i.e.  $\beta_j \neq 0$ ,  $j = q+1, \dots, p$ . The comparison of the two models will be based on the residual sum of squares (*RSS*). This quantity will always be smaller for the big model; the question is just by how much. If the difference is small, then one might not accept the additional variables, if it is big, then one should. The method for quantifying this is as follows:

$$F = \frac{n - (p+1)}{p - q} \cdot \frac{RSS_{Small} - RSS_{Big}}{RSS_{Big}} \sim F_{p-q, n-(p+1)}$$

Apparently, we have a relative comparison of the model adequacy, and also the number of observations, the total number of predictors and the difference in the number of predictors are taken into account. Under the null hypothesis, i.e. if the excess predictors do not contribute, the test statistic has an F-distribution with  $p - q$  and  $n - (p+1)$  degrees of freedom. Using that distribution, we can decide if the difference between the models is of significance or not. As an example, we consider the mortality data. Here, we want to test if the three predictors that are linked to air pollution can be omitted from the multiple linear regression model without any loss. We do this in R:

```
> fit.small <- update(fit, .~.-log(HC)-log(NOx)-log(SO2))
> anova(fit.small, fit)
Analysis of Variance Table
```

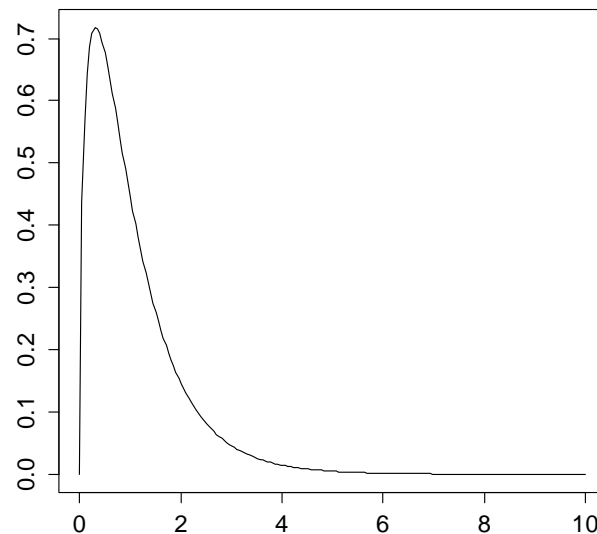
```
Model 1: Mortality ~ JanTemp + JulyTemp + RelHum + Rain +
  Educ + Dens + NonWhite + WhiteCollar +
  log(Pop) + House + Income
```

```
Model 2: Mortality ~ JanTemp + JulyTemp + RelHum + Rain +
  Educ + Dens + NonWhite + WhiteCollar +
  log(Pop) + House + Income + log(HC) +
  log(NOx) + log(SO2)
```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	47	61142				
2	44	52312	3	8829.3	2.4755	0.07388

Note that the small model was defined with an update from the big model. It is not required to do so, we could also write it explicitly using the `lm()` command. The R function for the hierarchical model comparison is `anova()`. As input, it takes the small and big models. In the output, the two model formulas are repeated, before the quantitative result is presented. We recognize the *RSS* for the two models, also the degrees of freedom and the value of the test statistic are given. This is gauged against the *F* distribution, which in this particular case looks as follows:

The F distribution with 3 and 44 df



If the excess predictors (i.e. the air pollution) do not have an effect and hence under the null hypothesis, we expect the test statistic to be smaller than:

```
> qf(0.95, 3, 44)
[1] 2.816466
```

This is the case, hence we are in the acceptance region and the null hypothesis cannot be rejected. The p-value is provided in the R output, it is 0.074. In conclusion, it might be that the air pollution, in the way it was measured here, does not affect mortality. At least we failed to reject the null that it does not have influence on the outcome with the current data and model. We finish this section by remarking that if a hierarchical model comparison is done for two models where the difference is only one single predictor, it coincides with the individual hypothesis test.

### 3.4.5 Global F-Test

The global F-test is another special case of a hierarchical model comparison, but an important one. The null hypothesis is  $H_0: \beta_1 = \beta_2 = \dots = \beta_p = 0$ , i.e. all regression coefficients are simultaneously zero. This means that none of the predictors (or also, the set of predictors as a whole) has a significant impact on the response. That, of course, would be a very bad sign in practice, and acceptance of the global null hypothesis would make a regression analysis mostly worthless. The alternative is that at least one  $\beta_j \neq 0$ , i.e. there is at least one predictor that yields a significant contribution, potentially also more than one.

Technically, the global F-test is a hierarchical model comparison. We are here comparing against the simplest conceivable model that is only made up of the intercept term  $\beta_0$ . For that model, all fitted values correspond to the average

$y$ -value. We then take the usual test statistic, but since here  $q = 0$  (i.e. there are no predictors in the small model!), it simplifies to:

$$F = \frac{n - (p + 1)}{p} \cdot \frac{RSS_{Small} - RSS_{Big}}{RSS_{Big}} \sim F_{p, n - (p + 1)}$$

Hence the value for that test statistic can be computed and the p-value can be derived. It will hardly ever be required to do so manually, since both quantities are printed in the last line of R's regression summary output. In particular:

F-statistic: 10.43 on 14 and 44 DF, p-value: 8.793e-10

The value of the test statistic lies far from the acceptance region, hence we reject the null hypothesis with a small p-value. For our example this means strong evidence that the mortality is affected by at least one of the variables which we gathered for the different metropolitan areas.

### 3.5 Prediction

Besides inference, the other very important application of the multiple regression fit is prediction. As soon as we are given the predictor values  $x_1, \dots, x_p$  for a new observation that was not part of the fitting and where the response is potentially unknown, we can provide its predicted value, i.e. the conditional expectation:

$$E[y | x_1, \dots, x_p] = \hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \dots + \hat{\beta}_p x_p$$

In simple regression, we had explained that a prediction within the range of observed  $x$ -values is safe if the regression line does not have a systematic error. The very same can be said about multiple regression, however here it is much more difficult to say what is within the range of observed  $x$ -values, and what is beyond. For understanding this, it is important to keep in mind that (usually, except for a few special cases) cannot overlook and visualize the  $p$ -dimensional predictor space. Moreover, even when all individual values in the new observations' predictors  $x_j$  lie within the values of the observed  $x_{1j}, \dots, x_{nj}$  in that predictor, it is not guaranteed that we are still not extrapolating. This phenomenon is known as the *curse of dimensionality*: the  $p$ -dimensional predictor space is huge and even when staying within the hypercube defined by the observed predictor values, the new observation can be in a location where no data were present for the fitting process. However, as long as the fit is free of a systematic error, and when the new observation is within that hypercube, the predictions are usually safe. Besides producing predictions, it is also very important to understand their precision. As in simple regression, we can provide both a 95% confidence interval for the conditional expectation  $E[y | x_1, \dots, x_p]$ , as well as the 95% prediction interval for the future observation. The meaning of these two intervals is exactly the same as in simple regression. The formulae are best written in matrix notation, in particular:

$$95\% \text{ confidence interval: } \hat{y} \pm t_{0.975; n-(p+1)} \cdot \hat{\sigma}_E \cdot \sqrt{x^T (X^T X)^{-1} x}$$

$$95\% \text{ prediction interval: } \hat{y} \pm t_{0.975; n-(p+1)} \cdot \hat{\sigma}_E \cdot \sqrt{1 + x^T (X^T X)^{-1} x}$$

In this notation,  $x^T = (1, x_1, \dots, x_p)$  is the predictor vector for the new observation, including the intercept term. The computation of these intervals is implemented in R's `predict()` function. As input for this routine, we need to provide the regression fit, the new predictor values  $x_1, \dots, x_p$  in form of a data frame with column names that are identical to the ones that were used for the fit. We illustrate with the following example, where we predict the mortality in a fictional city:

```
> new.x <- data.frame(JanTemp=32, JulyTemp=75, RelHum=55,
  Rain=51, Educ=10, Dens=3500, NonWhite=8.7,
  WhiteCollar=42, Pop=1200000, House=3,
  Income=41000, HC=22, NOx=18, SO2=38)
>
> predict(fit, newdata=new.x, interval="confidence")
      fit      lwr      upr
1 979.4028 936.9754 1021.83
>
> predict(fit, newdata=new.x, interval="prediction")
      fit      lwr      upr
1 979.4028 897.9834 1060.822
```

We observe that the predicted mortality is 979.4, with a 95% confidence interval ranging from 937.0 to 1021.8. The 95% prediction interval is (as always) wider and marks the range where we would expect a new observation. The numerical value of this interval is 898.0 to 1060.8.

## 3.6 Categorical Predictors

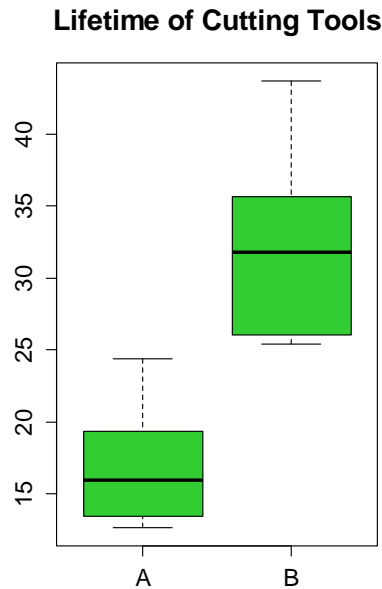
The variables we considered so far were all continuous, i.e. temperature, distance, pressure, et cetera. While the response must be continuous, it is perfectly valid to use *categorical predictors*, such as e.g. sex (male or female), *status variables* (employed or unemployed), *shifts* (day, evening, night). In general, these categorical variables have no natural scale of measurement. Thus, we must assign a set of levels to a categorical variable to account for the effect that the variable may have on the response. This is done through the use of indicator variables. In the regression context, they are better known as *dummy variables*. In the following sections, we will study the use of categorical predictors.

### 3.6.1 Example with 1 Categorical Predictor

The simplest case is a model where we have a continuous response  $y$  and one single categorical predictor  $\tilde{x}$ . The example that we consider is from a lathe (in German: "Drehbank"), where  $y$  is the lifetime of the cutting tool and the

categorical predictor  $\tilde{x}$  refers to two different tool types A and B. A typical way of displaying observed lifetimes in relation to the two tool types is with boxplots:

```
> boxplot(hours ~ tool, data=lathe)
```



We observe that the lifetimes of tools of type B are considerably higher than the ones of type A. The usual question in this setting is to estimate the expected lifetimes for the two tool types and answering the question whether the two means are identical. The boxplots let us assume that this is not the case. We can quantitatively undermine this by performing a Student's t-Test for non-paired data. The R command is as follows:

```
> t.test(hours ~ tool, var.equal=TRUE, data=lathe)
Two Sample t-test
data:  hours by tool
t = -6.435, df = 18, p-value = 4.681e-06
alternative hypothesis: true diff in means is not equal to 0
95 percent confidence interval:
-19.655814  -9.980186
sample estimates:
mean in group A mean in group B
      17.110      31.928
```

What does this have to do with regression analysis? More than you think. We can achieve the very same quantitative results by fitting a regression of  $y \sim \tilde{x}$ . Because regression is a technique for numerical variables, we need to replace the categorical predictor  $\tilde{x}$  by an indicator variable that takes values 0 and 1 to identify the tool types – this is a so-called dummy variable.



$$x = \begin{cases} 0 & \text{tool type A} \\ 1 & \text{tool type B} \end{cases}$$

The choice of 0 and 1 to identify the levels of this categorical predictor is arbitrary. In fact, any two distinct values for  $x$  would be satisfactory, although 0 and 1 are the normal choice. Then, if we consider the simple linear regression model

$$y_i = \beta_0 + \beta_1 x_i + E_i,$$

this becomes  $y_i = \beta_0 + E_i$  for observations  $i$  with tool type A and hence  $x_i = 0$ . Then, for observations  $j$  with tool type B,  $x_j = 1$  and the regression equates to  $y_j = \beta_0 + \beta_1 + E_j$ . Consequently,  $\beta_0$  is the expected lifetime for tools of type A, and  $\beta_0 + \beta_1$  the one for tools of type B. Or we can also say that  $\beta_1$  is the difference in the two lifetime expectations. With R, fitting regression models with categorical predictors is straightforward. We do not even need to take care of the generating the dummy variable, but can just provide a factor variable, i.e. `class(lathe$tool) = "factor "`. The summary output is as follows:

```
> summary(lm(hours ~ tool, data=lathe))

Call: lm(formula = hours ~ tool, data = lathe)

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   17.110      1.628   10.508 4.14e-09 ***
toolB         14.818      2.303    6.435 4.68e-06 ***
---
Residual standard error: 5.149 on 18 degrees of freedom
Multiple R-squared:  0.697, Adjusted R-squared:  0.6802
F-statistic: 41.41 on 1 and 18 DF, p-value: 4.681e-06
```

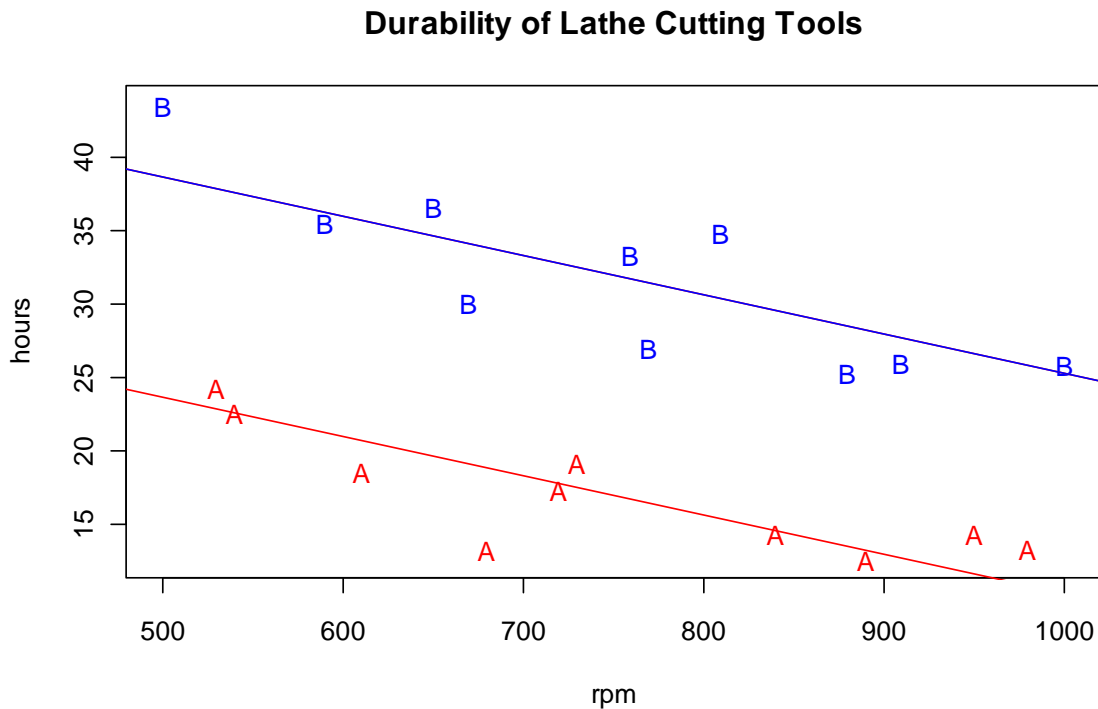
We observe that the regression coefficients are identical to the results from the testing procedure above, where arithmetic means were drawn. Furthermore, the test for the null hypothesis  $\beta_1 = 0$  addresses exactly the same question as the t-test for non-paired data does. However, not only the question is identical, but also the answer (and the methodology behind). The p-values with both approaches are one and the same. Hence, if we can do regression, we could in fact retire the non-paired t-test altogether.

### 3.6.2 Mix of Categorical and Continuous Predictors

We now enhance our previous example and want to relate the lifetime  $y$  of a cutting tool on the speed of the machine in rpm ( $x_1$ ) and the type of cutting tool used ( $\tilde{x}_2$ ). The first predictor is continuous, while the second is categorical, again with levels A and B. As before, it will be replaced it by an indicator or dummy variable that takes values 0 and 1 to identify the tool types.

$$x_2 = \begin{cases} 0 & \text{tool type A} \\ 1 & \text{tool type B} \end{cases}$$

We can display the data in a scatter plot of *hours* vs. *rpm*, and distinguish the two tool types by different plotting characters.



The plot also shows parallel regression lines for tool types A and B. We will now explain how they are found. The regression model for the situation with one continuous and one categorical predictor is as follows:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + E \text{ or in R notation } \text{hours} \sim \text{rpm} + \text{tool}.$$

The summary output for this regression model is:

```
> summary(lm(hours ~ rpm + tool, data = lathe))
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	36.98560	3.51038	10.536	7.16e-09	***
rpm	-0.02661	0.00452	-5.887	1.79e-05	***
toolB	15.00425	1.35967	11.035	3.59e-09	***

---

Residual standard error: 3.039 on 17 degrees of freedom  
 Multiple R-squared: 0.9003, Adjusted R-squared: 0.8886  
 F-statistic: 76.75 on 2 and 17 DF, p-value: 3.086e-09

We will now turn our attention to the interpretation of this regression model. We first consider an observation  $i$  where the tool is of type A. There, we have  $x_{i2} = 0$  and thus the model simplifies to:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 \cdot 0 + E_i = \beta_0 + \beta_1 x_{i1} + E_i.$$

Thus, the relation between tool life and lathe speed for tool type A is a straight line with intercept  $\beta_0 = 36.99$  and slope  $\beta_1 = -0.027$ . Important: note that the slope is generally not equal to the one we would obtain from a simple linear regression for tools of type A only! Now conversely, for any observation  $j$  with tool type B, we have  $x_{j2} = 1$ , and thus:

$$y_j = \beta_0 + \beta_1 x_{j1} + \beta_2 \cdot 1 + E_j = (\beta_0 + \beta_2) + \beta_1 x_{j1} + E_j$$

That is, for tool type B the relation between tool durability and lathe speed is also a straight line with the same slope  $\beta_1 = -0.027$ , but different intercept  $\beta_0 + \beta_2 = 51.99$ . Thus, the model estimates a common, identical slope coefficient for the two tool types. The regression coefficient  $\beta_2$  of the dummy variable  $x_2$  accounts for the additive shift in durability of tool type B vs. tool type A, i.e. measures the difference in mean tool life when changing from tool type A to tool type B. Note that the two regression lines are parallel by definition. To make the analysis complete, we would need to check the diagnostic plots. We leave this as an exercise, because there are no peculiarities for this specific example. For the diagnostic plots, it is helpful to use different plotting symbols for tool types A and B.

### 3.6.3 Interaction Terms

Above, the regression line for tools A and B had different intercept, but identical slope. In this example, the fit seemed to be pretty well even under this restriction. However, we can easily imagine a situation where two parallel regression lines are not appropriate. The question this section deals with is whether and how a model with two different regression lines can be fitted. It is possible to model this situation with a single regression equation by using indicator variables. The model is:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + E \text{ or } \text{hours} \sim \text{rpm} + \text{tool} + \text{rpm}:\text{tool}.$$

An interaction term or cross product  $x_1 x_2$  has been added to the model. To interpret the parameters in this model, we first consider an observation  $i$  with tool type A. Remember; this means that the dummy variable  $x_{i2} = 0$ .

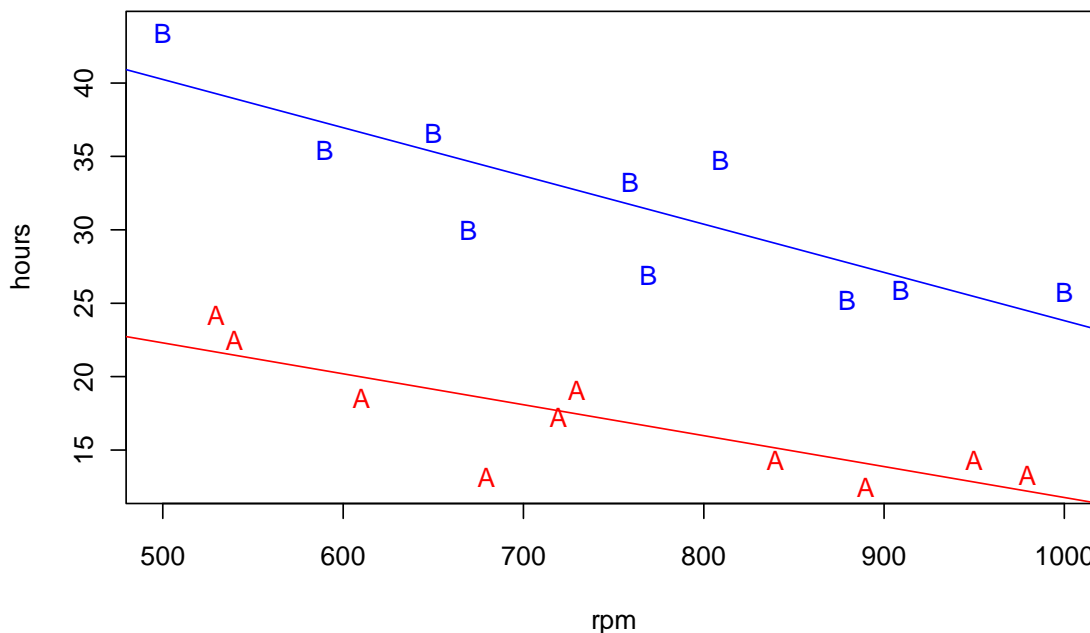
$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 \cdot 0 + \beta_3 \cdot 0 + E_i = \beta_0 + \beta_1 x_{i1} + E_i$$

Thus, this is again a regression line with intercept  $\beta_0$  and slope  $\beta_1$ . However, the slope  $\beta_1$  will generally be different to the one found with the main effect model from section 3.6.2. For an observation  $j$  with tool type B, we have  $x_{j2} = 1$  for the dummy variable. Thus, the regression model becomes:

$$y_j = \beta_0 + \beta_1 x_{j1} + \beta_2 \cdot 1 + \beta_3 x_{j1} \cdot 1 + E_j = (\beta_0 + \beta_2) + (\beta_1 + \beta_3) x_{j1} + E_j$$

This is a straight-line model with intercept  $\beta_0 + \beta_2$  and slope  $\beta_1 + \beta_3$ . Thus, the interaction model defines two regression lines with different intercepts and different slopes. Therefore the parameter  $\beta_2$  reflects the change in the intercept associated with changing from tool type A to tool type B, and  $\beta_3$  indicates the change in the slope associated with this change.

### Durability of Lathe Cutting Tools: with Interaction



The scatterplot of `hours` vs. `rpm` is shown above, together with the two regression lines that are no longer parallel. There is a large vertical shift between the two regression lines. The slope however, only differs little. An obvious question is whether fitting two regression lines with different slopes is necessary, i.e. whether the difference is statistically significant or whether we could also do with the simpler model that has two parallel lines. This amounts to testing

$$H_0: \beta_3 = 0 \text{ against } H_A: \beta_3 \neq 0.$$

This is an individual parameter test for the interaction term, and the result can be directly read from the summary output.

```
> summary(lm(hours ~ rpm + tool + rpm:tool, data = lathe))
Coefficients:      Estimate Std. Error t value Pr(>|t|)
(Intercept)    32.774760    4.633472    7.073 2.63e-06 ***
rpm             -0.020970    0.006074   -3.452 0.00328 **
toolB           23.970593    6.768973    3.541 0.00272 **
rpm:toolB      -0.011944    0.008842   -1.351 0.19553
---
Residual standard error: 2.968 on 16 degrees of freedom
Multiple R-squared:  0.9105, Adjusted R-squared:  0.8937
F-statistic: 54.25 on 3 and 16 DF,  p-value: 1.319e-08
```

The p-value for our null hypothesis is 0.196, thus the interaction term is not statistically significant. This leads to the conjecture that if there are no further (practical) reasons strongly speaking for different slopes, we would (and could) fit parallel lines. Note that the (full) interaction model always yields the same result as two separate simple linear regressions on tools of type A, and tools of type B. Does that mean we should prefer simple regressions? The answer is definitely no, because with the common model we can formally test to which extent the two tools behave identically. From the above test, we can accept the hypothesis that the  $r_{pm}$  variable has the same effect on lifetime for both tool types. Another, more general but equally interesting question is if there is any difference at all between the two tools with respect to lifetime. The associated null hypothesis is:

$$H_0: \beta_2 = \beta_3 = 0 \text{ against } H_A: \beta_2 \neq 0 \text{ and / or } \beta_3 \neq 0.$$

Under the null, both the dummy and interaction coefficient are zero, meaning that there would be a common regression line for both tools. Our graphical display shows that this is implausible, but it is helpful to formally test the claim. Because we are testing two coefficients simultaneously, we require a partial F-test. The R-code and the output are as follows:

```
> fit1 <- lm(hours ~ rpm, data=lathe)
> fit2 <- lm(hours ~ rpm + tool + rpm:tool, data=lathe)
> anova(fit1, fit2)
Model 1: hours ~ rpm
Model 2: hours ~ rpm + tool + rpm:tool
  Res.Df    RSS Df Sum of Sq    F    Pr(>F)
1     18 1282.08
2     16  140.98  2    1141.1 64.755 2.137e-08 ***
```

We observe that the p-value is very small, and the partial F-test thus highly significant. While there is no evidence for different slopes in this example, there is strong evidence of a difference (in either slope or intercept). Regarding the scatterplot, with the pronounced vertical shift between tool types A and B, this does not surprise us. Finally, we conclude this section by stating that the use of interaction models is not restricted to a combination of continuous and categorical predictors. In this case, they can be visualized most easily. However, we can have them between any types of predictors. They are appropriate whenever there is, or whenever we suspect a change in the effect of one predictor on the response, conditional on the level of another predictor.

### 3.6.4 Categorical Input with More than Two Levels

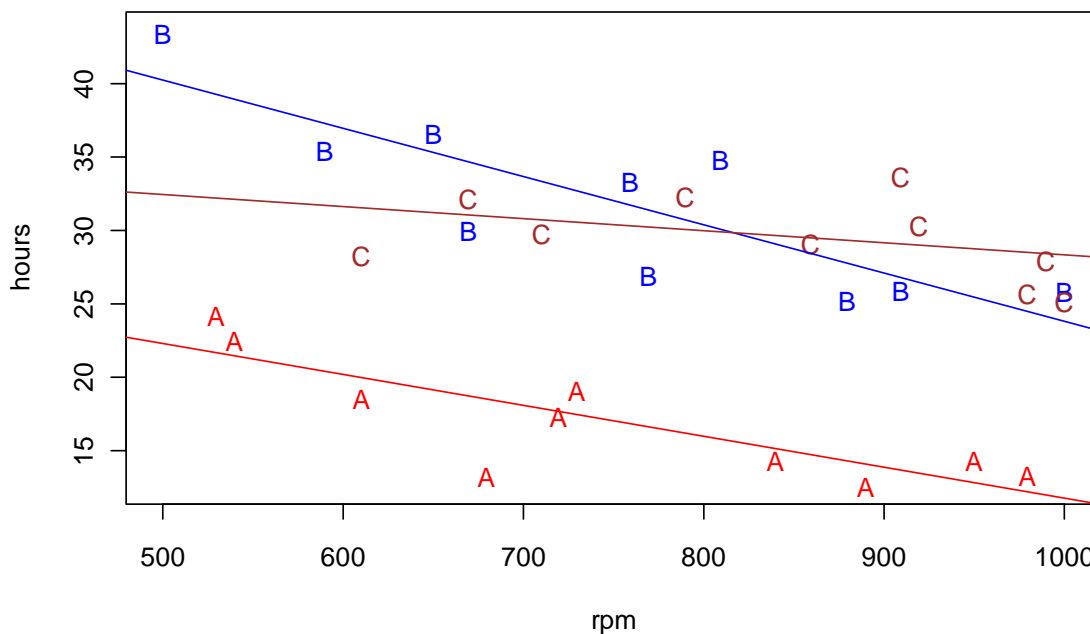
An obvious extension to the previous example with lathe cutting tools would be to consider three or more types of tools instead of only two. The tool variable then is still categorical, but no longer binary, and we need more dummy variables. For example, suppose that there are three tool types A, B and C. We then require two dummy variables to incorporate them into the model. The coding is as follows:

$x_2$	$x_3$	
0	0	for observations of type A
1	0	for observations of type B
0	1	for observations of type C

In general, a qualitative variable with  $\ell$  levels is represented by  $\ell-1$  dummy variables, each taking values 0 and 1. Be careful, categorical variables must be represented in this fashion, and generally cannot be numerically coded in one single variable with values  $0, 1, \dots, \ell-1$ . Please also note that with the here presented dummy encoding, the first level (here: tool type A) is always the reference. This is also how R codes categorical input variables by default: the first factor level is the reference. There are, however, different options for coding, called contrasts. This is more of a topic in analysis of variance, thus we do not pursue that issue here. The main effects regression model with three types of tools and their respective dummy variables is now:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + E, \text{ or in R simply } \text{hours} \sim \text{rpm} + \text{tool}.$$

### Durability of Lathe Cutting Tools: 3 Types



This will fit three parallel regression lines, where each has a different intercept. However, when we closely observe the scatter plot above, we gain the impression that the durability of tool type C seems to depend much less on  $\text{rpm}$  than the other two. While at slow speeds, its lifetime seems to be inferior to the type B tools, they seem to last longer at faster speeds. Because the main effects model cannot deal with the apparently different slopes, we fit the interaction model  $\text{hours} \sim \text{rpm} + \text{tool} + \text{rpm}:\text{tool}$ , in mathematical annotation:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_1 x_2 + \beta_5 x_1 x_3 + E$$

The interpretation of this model is as before with binary categorical input. We leave it as an exercise to write down the cases for observations  $i$ ,  $j$  and  $k$  of tool types A, B and C. The regression fit with R is again straightforward; we only need the `tool` variable to be a factor with multiple levels. R then generates the necessary encoding into dummy variables automatically in the background.

```
> summary(lm(hours ~ rpm + tool + rpm:tool, data = abc.lathe)
Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	32.774760	4.496024	7.290	1.57e-07	***
rpm	-0.020970	0.005894	-3.558	0.00160	**
toolB	23.970593	6.568177	3.650	0.00127	**
toolC	3.803941	7.334477	0.519	0.60876	
rpm:toolB	-0.011944	0.008579	-1.392	0.17664	
rpm:toolC	0.012751	0.008984	1.419	0.16869	

```
---
```

```
Residual standard error: 2.88 on 24 degrees of freedom
Multiple R-squared: 0.8906, Adjusted R-squared: 0.8678
F-statistic: 39.08 on 5 and 24 DF, p-value: 9.064e-11
```

The interpretation of this summary output now needs to be done with care. Individual parameter tests for dummy variable coefficients of categorical predictors with more than two levels are not meaningful! Thus, from the above output, we cannot conjecture that we can do without a different intercept for tool C on the basis that the test for  $H_0: \beta_3 = 0$  is not significant. Moreover, also coefficients  $\beta_4$  and  $\beta_5$  have p-values  $>0.05$ . Does that mean that we can do without the interaction? No! We can only either exclude all the interaction terms at once, or we have to keep all of them in the model. Hence, we test the hypothesis

$$H_0: \beta_4 = 0 \text{ and } \beta_5 = 0 \text{ against } H_A: \beta_4 \neq 0 \text{ and/or } \beta_5 \neq 0.$$

This is again a partial F-test. Instead of using the summary function in regression problems with factor variables that have more than two levels, or when there are interaction terms so that the terms in the model have a hierarchy, we recommend to work with the `drop1()` function:

```
> drop1(fit.abc, test="F")
Single term deletions
Model: hours ~ rpm + tool + rpm:tool
      Df Sum of Sq    RSS    AIC F value  Pr(>F)
<none>                199.10 68.779
rpm:tool  2     59.688 258.79 72.645  3.5974 0.04301 *
```

As we can see, the only term which can be excluded from the current model is the interaction term. As long as it is part of the model, the main effects which it stems from have to remain. Furthermore, as mentioned above, there is no exclusion of single factor levels resp. dummy variables possible. R function `drop1()` regards all these rules and performs the respective tests for the variables which are

candidates for exclusion from the model. In our particular case, the interaction term is significant with a p-value 0.043. Therefore, different slopes for the three tool types are to be regarded as necessary. Of course we may also ask the more general question whether there is any difference among the tool types at all. This corresponds to a broader test versus a sub-model with only  $r_{pm}$  as a predictor:

$$H_0: \beta_2 = \beta_3 = \beta_4 = \beta_5 = 0 \text{ against } H_A: \text{any of } \beta_2, \beta_3, \beta_4, \beta_5 \neq 0.$$

We need to fit both models and compare them with the `anova()` function:

```
> f.sma <- lm(hours ~ rpm, data=abc.lathe)
> f.big <- lm(hours ~ rpm + tool + rpm:tool, data=abc.lathe)
> anova(f.sma, f.big)
Analysis of Variance Table
Model 1: hours ~ rpm
Model 2: hours ~ rpm + tool + rpm:tool
  Res.Df    RSS Df Sum of Sq    F    Pr(>F)
1      28 1681.3
2      24  199.1  4    1482.2 44.665 8.811e-11 ***
```

The additional terms in the big model are highly significant. In accordance to our visual impression from the scatterplots, it is a good idea to distinguish the different tool types with respect to their life time. However, please keep in mind that many (most!) regression problems cannot be visualized so easily, such that we have to rely on test results rather than visualization for being able to draw conclusions.

### 3.6.5 Categorizing Quantitative Predictors

A sometimes useful trick is to generate a categorical predictor out of a continuous one. In the lathe example from above, we could for example categorize the continuous predictor  $r_{pm}$  into bins ranging from 400-600rpm, 600-800rpm, and 800-1000rpm. Does this make sense? At the first glance, there does not seem to be an advantage for doing so, and in this particular example, there is in fact none.

Also, the disadvantage of the categorization is that more parameters are required to represent the information of the continuous predictor. Thus, we increase the model complexity by this categorization. However, under the presence of enough data, this is sometimes desired, because it does not require the analyst to make any prior assumptions about the functional form of the relationship between the response and the predictor variable and enhances the flexibility.

Another advantage of the categorization approach is that it allows dealing with missing observations, without having to delete them. If they are numerous in a certain predictor, we could just categorize it, and assign all observations with missing information in that predictor the label “unknown”. Within the model, we would just estimate the effect of unknown status in that predictor. Such a categorization of continuous predictors is in some fields quite popular among data analysts. The approach is also known as “poor man’s GAM”.



## 3.7 Model Diagnostics

We need to check the assumptions we made for fitting a multiple linear regression model. Why? One reason is because we want to make sure that the estimates we produce and the inference we draw is valid. This seems quite technical and also somewhat fussy and boring. Still, it is absolutely essential to perform residual analysis before any findings from the summary output, confidence intervals or predictions are reported. If the model that was used is flawed, all these results might be dead wrong and presenting them unverified could bring you into a very uncomfortable situation.

However, there is a second, usually equally important reason to perform model diagnostics: any potential flaws that appear can help us to improve the model and enhance our understanding of the relation between response and predictors. In fact, we can go as far as saying “it is all in the residuals”, i.e. most of what we can learn about how to better shape a regression analysis is derived from some clever diagnostic plots. Such enhancements include response and/or predictor transformations, inclusion of further predictors or interaction terms among them, weighted regression analysis or in some situations also the use of more generally formulated or more robust models that are not based on OLS. This is explorative data analysis at its best – we fit a model, try some ideas, check the results and try to improve.

### 3.7.1 What Do We Need to Check For, and How?

We restate the assumptions we made for using the OLS procedure when fitting multiple linear regression model and drawing inference from them. One goal in model diagnostics is to detect potential deviations from them.

$$E[E_i] = 0,$$

$$\text{Var}(E_i) = \sigma_E^2,$$

$$\text{Cov}(E_i, E_j) = 0 \text{ for all } i \neq j,$$

$$E_i \sim N(0, \sigma_E^2 I), \text{ i.i.d.}$$

While the first three conditions are necessary for performing least square estimation and the validity of the fitted values, the last condition is only required for any hypothesis tests, confidence intervals and prediction intervals. Since these are very important and the OLS estimator quickly becomes inefficient under non-Gaussian distribution, the normal assumptions shall also always be verified. While zero error, constant variance and the Gaussian property are relatively easy to check and will be addressed in the next subsection, uncorrelatedness or independence are more delicate matters that are postponed to a later chapter. Please also note that most of our diagnostic techniques are visual. This requires some expertise for their interpretation, but has the benefit of a very wide scope and good power for detecting what is important to be found.

### 3.7.2 Checking Error Assumptions

In this section, we present methods for checking the zero expectation, constant variance and normality assumptions of the errors  $E_i$ . The errors  $E_i$  are random variables which tell the (potential) difference  $y_i - X\beta$  between observed and true value; they are a concept and unobservable in practice. What we can examine are the residuals. We have  $R_i = y_i - X\hat{\beta}$  which again are random variables and are the (potential) difference between observed value and the least squares and their realized values  $r_i$  on a particular dataset.

Please note that  $E_i$  and  $R_i$  are not one and the same and to a certain extent, also have different properties. Even when the usual least squares conditions on the errors are exactly met and hence  $\text{Var}(E) = \sigma_E^2 I$ , the residuals will be weakly correlated and heteroskedastic. The derivation of this result starts with writing the fitted values with the hat matrix:

$$\hat{y} = Hy = X(X^T X)^{-1} X^T y$$

The random vector of residuals can be written as  $R = y - X\hat{\beta}$  and we will now study its distribution, i.e. form, family and moments. We obtain:

$$E[R] = E[y - X\hat{\beta}] = 0, \text{ and thus } E[R_i] = 0 \text{ for all } i = 1, \dots, n$$

$$\text{Var}(R) = \text{Var}(Iy - Hy) = \dots = (I - H)\sigma_E^2, \text{ from which we derive}$$

$$\text{Var}(R_i) = (1 - H_{ii})\sigma_E^2 \text{ and } \text{Cov}(R_i, R_j) = -\sigma_E^2 \cdot H_{ij} \neq 0$$

Finally, because the residuals  $R_i = (I - H)y$  stem from a linear combination of the (under assuming  $E_i \sim N(0, \sigma_E^2)$ ) normally distributed responses, the residuals vector follows a Gaussian distribution, too:

$$R_i \sim N(0, (1 - H_{ii})\sigma_E^2)$$

We here emphasize again that due to heteroskedasticity and correlation, the residuals do not exactly match the properties that the errors are supposed to have. The “further away” from the other data points an observations lies and hence the bigger its leverage  $H_{ii}$  is, the smaller the variance of its residuals  $R_i$  will be. This raises the question whether it is sensible to perform model diagnostics and checking the assumption for the error on the basis of the residuals. Fortunately, the answer is yes. In well-posed regression problems where enough data are present, the effects of estimation-induced residual correlation and heteroskedasticity will be relatively minor and can often be neglected. Hence, even an analysis of the so called *raw residuals*  $r_i$  usually yields reasonable insight.

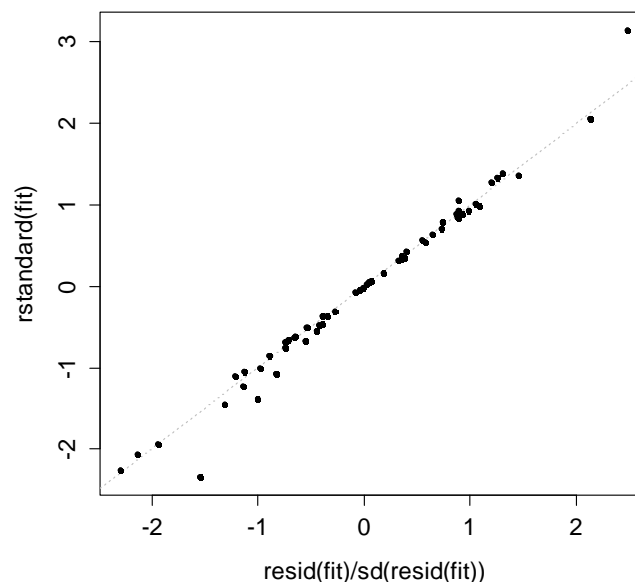
Moreover, one can try to *standardize* or *studentize* the residuals for mitigating the heteroskedasticity. The two terms refer to a division of each residual by its estimated standard deviation to bring them on a scale with unit variance:

$$\tilde{r}_i = \frac{r_i}{\hat{\sigma}_E \sqrt{1 - H_{ii}}}.$$

Here,  $H_{ii}$  is the  $i^{\text{th}}$  diagonal element of the hat matrix and  $\hat{\sigma}_E$  is an estimate of the residual standard error. Depending on whether  $\hat{\sigma}_E$  comes from the full fit or from an alternate regression without the  $i^{\text{th}}$  data point, one speaks of standardized respectively studentized residuals. The distinction between the two types is mostly for academic purpose and not too relevant for practical application. Standardized and studentized residuals can be obtained in R through functions `rstandard()` and `rstudent()`, respectively. However, the difference between either of these and the raw residuals  $r_i$  can be pronounced for data points with extreme  $x$  values. We compare the centered and rescaled raw versus the standardized residuals for the mortality data:

```
> plot(resid(fit)/sd(resid(fit)), rstandard(fit), pch=20)
```

**Comparison of Standardized vs. Raw Residuals**



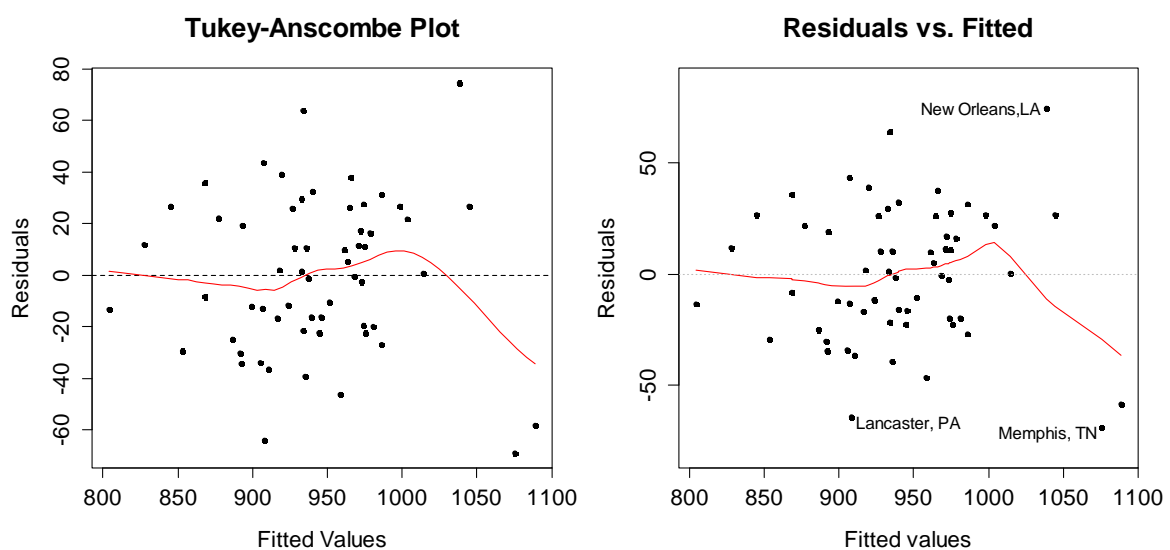
Except for the Tukey-Anscombe plot, all the forthcoming residuals plots will be based on standardized residuals. This is how things are implemented in R and unless you know much better, it is recommended to stick to this.

### Tukey-Anscombe Plot

This plot of *residuals*  $r_i$  vs. *fitted values*  $\hat{y}_i$  is named after the two researchers (who were brothers-in-law) that made it popular and is the most important diagnostic tool for any multiple linear regression fit. It is mainly aimed at verifying  $E[E_i]=0$  and so evaluates whether the model is correct and makes unbiased predictions. However, please note that it is not possible to check the assumption of zero expectation for each error individually – some residuals will be large and some will be small, but this proves nothing. What we need to check is whether the local mean of the residuals is related to some other quantity. This should not be

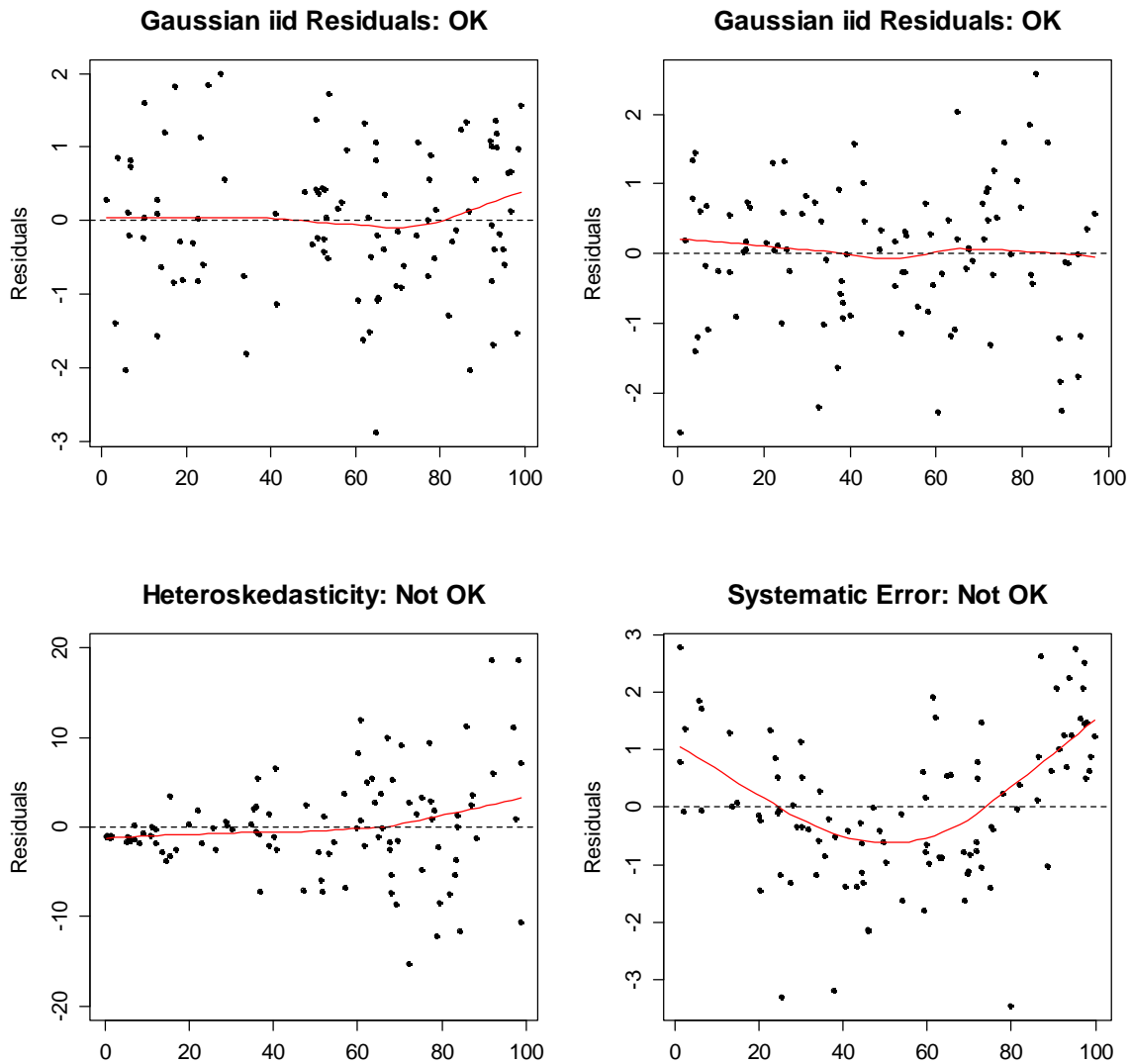
the case, no matter what that quantity is. The easiest way to generate a Tukey-Anscombe plot in R is with the `plot.lm()` function, i.e. the command is simply `plot(fit, which=1)`. Because it is more instructive, we here show the code to self-generate the plot and display it along R's default version. As you can see, the two plots are nearly identical, except for some minor differences in the smoother which is due to the different algorithms that were used. Also, `plot.lm()` always annotates the name of the three data points with the biggest residuals which is very useful in practice.

```
> plot(fitted(fit), resid(fit), main="Tukey-Anscombe")
> lines(loess.smooth(fitted(fit), resid(fit)), col="red")
> abline(h=0, lty=2)
```



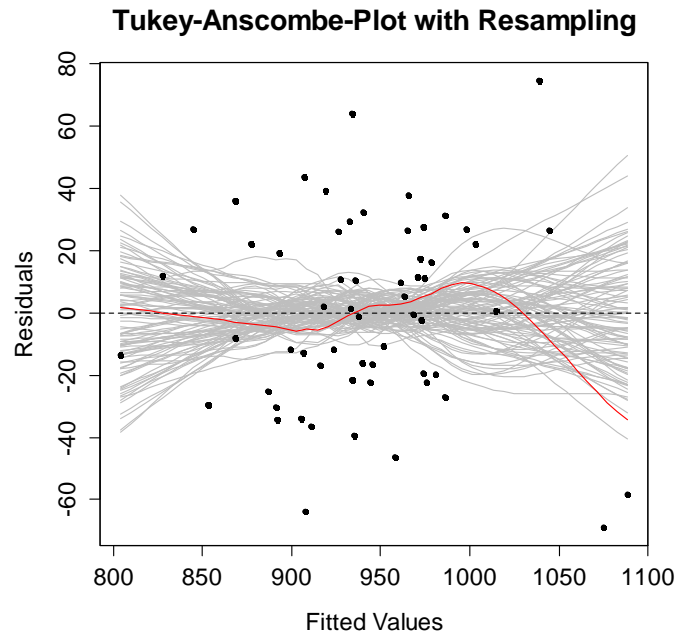
Producing a Tukey-Anscombe plot is one thing, drawing the correct conclusions from it the other. Some artificial examples which are displayed on the next page illustrate the concept. For the zero error expectation  $E[E_i] = 0$  assumption to hold, we require that the smoother does not systematically deviate from the  $x$ -axis. This is the case in the top two panels, because the “residuals” there originate from an iid Gaussian distribution. Hence, any deviation can be attributed to randomness alone. The situation is different in the bottom right panel: here, there is a systematic deviation of the smoother from the  $x$ -axis, and under no circumstances we could tolerate such a faulty model. The bottom left panel shows a situation where  $E[E_i]$  does not systematically deviate from zero, but  $Var(E_i)$  massively increases for large fitted values. Also this is a violation of the assumptions for OLS regression, although a less severe one.

So when does a smoother systematically deviate from the  $x$ -axis, and when is this just due to random variation? Generally, this is an expert call based on the magnitude of the deviation and the number of data points which are involved. An elegant way out of these (sometimes difficult) considerations is given by a resampling approach.



It is based on keeping the fitted values  $\hat{y}_i$  as they are, but for each data point a “new” residual  $r_i^*$  is assigned, obtained from sampling with replacement among the  $r_i$ . Then, with the new data pairs  $(\hat{y}_i, r_i^*)_{i=1, \dots, n}$ , a smoother is fitted and it is added to the Tukey-Anscombe plot as a grey line, the resampled data points are not shown. The entire process is repeated for a number of times, e.g. 100x. Clearly, because the residuals were randomly assigned to the fitted values, there cannot be a systematic (but just a random) deviation of the smoother. Hence, these resampled smoothers illustrate the magnitude which a random deviation from the  $x$ -axis can take and help us to assess the smoother on the original residuals.

```
> plot(fitted(fit), resid(fit), pch=20, main="...")
> for (i in 1:100) {
> +   sresid <- sample(resid(fit), replace=TRUE))
> +   lines(loess.smooth(fitted(fit), sresid), col="grey")}
> lines(loess.smooth(fitted(fit), resid(fit)), col="red")
```



You can obtain the R function `resplot()` from the lecturer; it implements the resampling idea. Please note that there is a random element in producing these plots. Hence without setting the random number generator using `set.seed()`, the grey lines will look different in each repetition. The result above tells us that up to fitted values which are  $\leq 1050$ , the original red smoother lies well within the grey scatter and thus just randomly deviates from the  $x$ -axis. At the right hand boundary, the situation remains fuzzy. The original smooth is at the edge of what can be generated by random sampling, but not clearly beyond. Also because it is only due two data points with strongly negative residuals, the author feels that we cannot reject the hypothesis  $E[E_i]=0$ . Hence, we here attribute the deviation of the smoother to randomness.

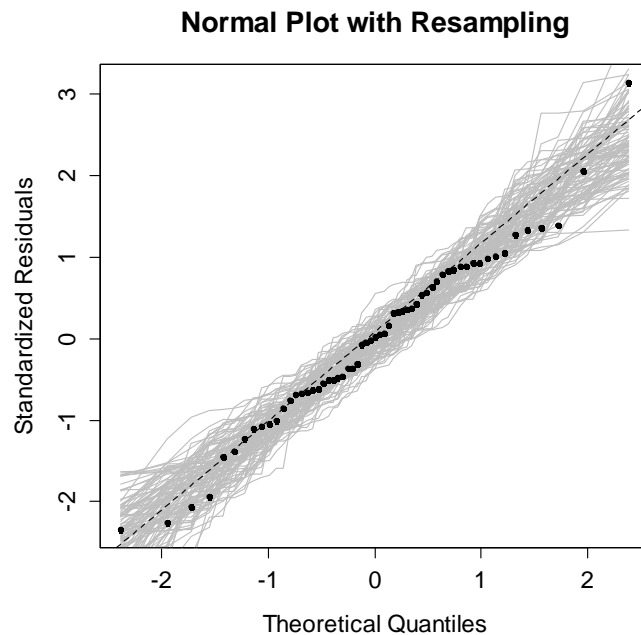
But what could we do in case of a systematic error? The answer is simple; we need to fit a better model. The difficult part is to find the correct way to improve the model. Often, (log-)transformations of the response and/or predictor variables help. In other cases, a systematic error may be cured by adding further predictors, higher order terms or interaction terms between some of the predictors. In some (pretty rare) cases, we may also need to conclude that multiple linear regression modelling is not the correct way to approach the problem at hand, so that more sophisticated procedures are required.

We conclude this section with a brief summary: if the smoother in the Tukey-Anscombe plot systematically deviates from the  $x$ -axis, the regression model has a systematic error. In this case, we should not generate predictions or report findings from the summary output, but need to improve the model. The most generic trick that helps in many situations is to consider log-transformations for the variables where this is sensible.

## Normal Plot

The assumption of *iid* normally distributed errors can be checked with the *normal plot*, i.e. for  $i=1,\dots,n$  we plot the ordered standardized residuals against the  $i/(n+1)$  quantiles of the standard normal distribution. As we have seen above, the normal distribution of the errors  $E_i$  also transfers to the (standardized) residuals. Thus, the data points in the normal plot should align, i.e. not show a systematic deviation of the line that is fitted through the 1<sup>st</sup> and 3<sup>rd</sup> quartiles of the two distributions. This may again raise some discussion as when a deviation is random or systematic. We can again support this decision by resampling, i.e. drawing 100 random samples of length  $n$  from a Gaussian distribution that shares mean and standard deviation with the residuals.

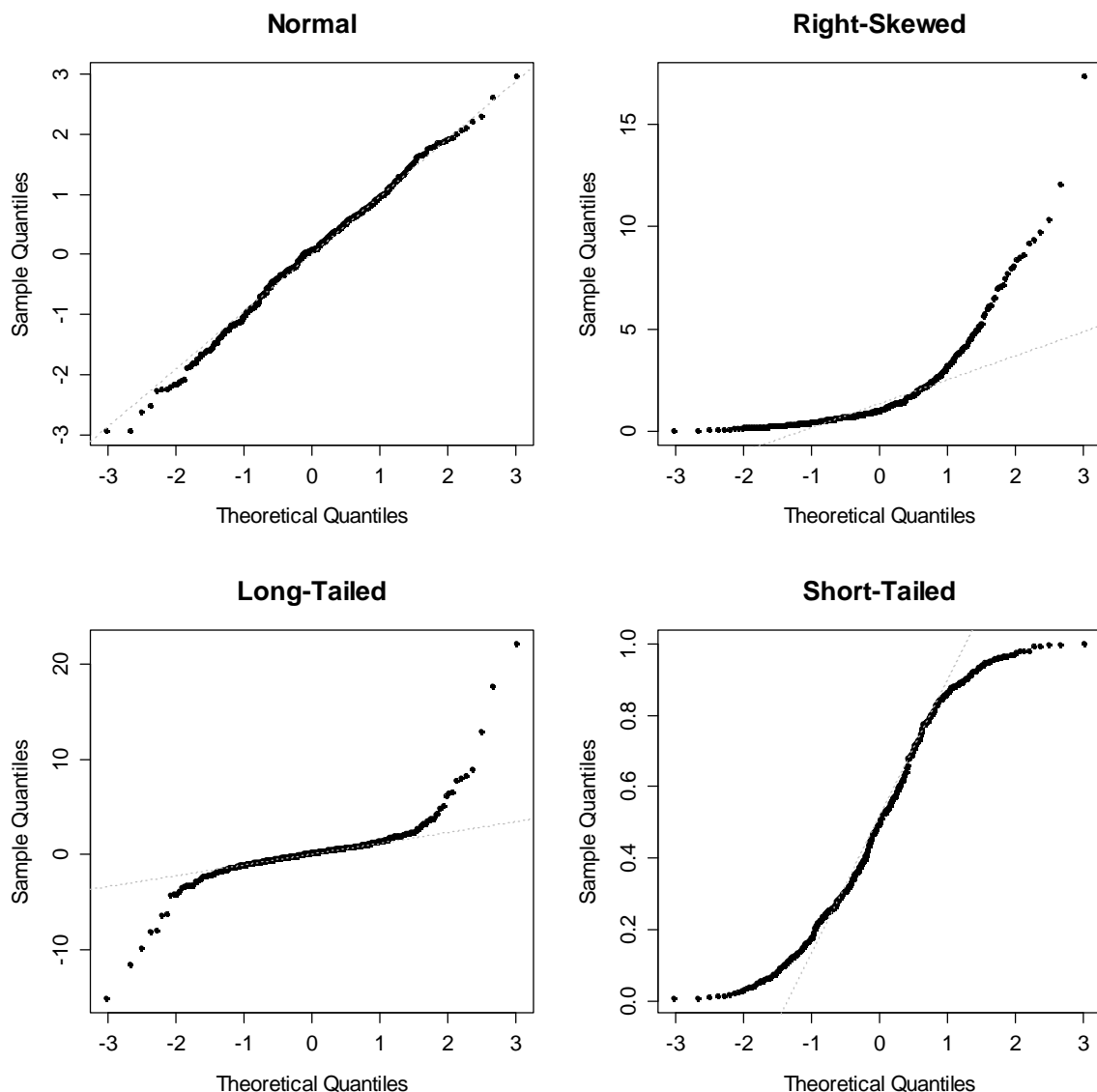
```
> qq <- qqnorm(rstandard(fit), pch=20, main="...")
> for (i in 1:100) {
> +   sresid <- rnorm(length(qq$y), mean(qq$y), sd(qq$y))
> +   lines(sort(qq$x), sort(sresid), col="grey")}
> points(qq$x, qq$y, pch=20); box()
> qqline(rstandard(fit), lty=2)
```



We observe that all residuals from the mortality dataset fall within the resampling based confidence region, thus there is no systematic deviation from the normal distribution or the *iid* assumption – please keep in mind that the above plot may show a deviation also in cases where the data are Gaussian, but (strongly) heteroskedastic! The question which remains is what to do if there is a systematic deviation. This depends on the type of non-Gaussian residuals that are observed. The OLS estimator is not tolerant to skewed residuals, especially because they mostly coincide with a systematic error, i.e. a violation in the Tukey-Anscombe plot. Heavy- and especially short-tailed residual distributions are less worrying, as long as they are symmetrically distributed. In that case, they hardly have an

adverse effect on the fitted values, which are still unbiased and trustworthy. In contrast, the accuracy in the levels of the confidence intervals that are computed and the precision of the tests suffers – an issue, which definitely has to be kept in mind. What also needs to be mentioned is that the efficiency of the OLS estimator degrades quickly for heavy-tailed distributions, meaning that there are other (i.e. robust) estimators that estimate the regression coefficients with higher precision.

Some prototypical normal plots are shown below. In clockwise order, starting from the top left, they show residuals from a normal distribution, then from the right-skewed lognormal distribution, the long-tailed Student's  $t_2$ -distribution and finally a short-tailed Uniform distribution.

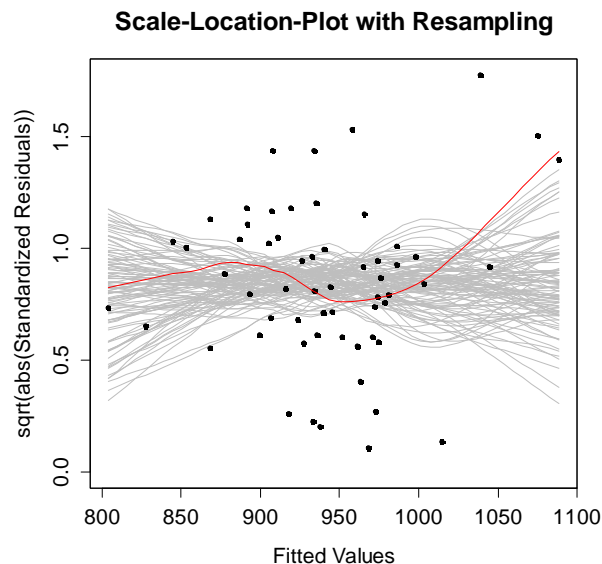


As has been mentioned above, the situation in the top right panel is the most worrying. It is very unlikely that a regression fit producing such a normal plot is trustworthy. The two plots in the bottom panels are less than ideal, too. Here at least, due to the symmetrical distribution of the residuals, the fitted values are likely to be unbiased (though not efficiently estimated).



## Scale-Location Plot

This plot facilitates detecting non-constant variance, i.e. heteroskedasticity. We had argued above that one can also detect this by looking sharply at the Tukey-Anscombe plot, but the Scale-Location plot is more specific. It displays the square root of the absolute value of the standardized residuals  $\sqrt{|\tilde{r}_i|}$  versus the fitted values. The crucial operation is the absolute value. It means that the bottom half of the Tukey-Anscombe plot is folded over, hence we can better detect a potential relation of the residuals' magnitude with the fitted value. Again, a smoother is added and if there is no heteroskedasticity, it will run horizontally. Also here, the task is to identify systematic deviations. We can again use the resampling idea of randomly drawing new data pairs  $(\hat{y}_i, r_i^*)$  to produce a confidence region in the Scale-Location plot, see below. It seems as if the variance in the residuals grows with increasing fitted value. A popular cure for this is to apply a log-transformation to the response variable. If one tries (not displayed in this scriptum), there is a small benefit.

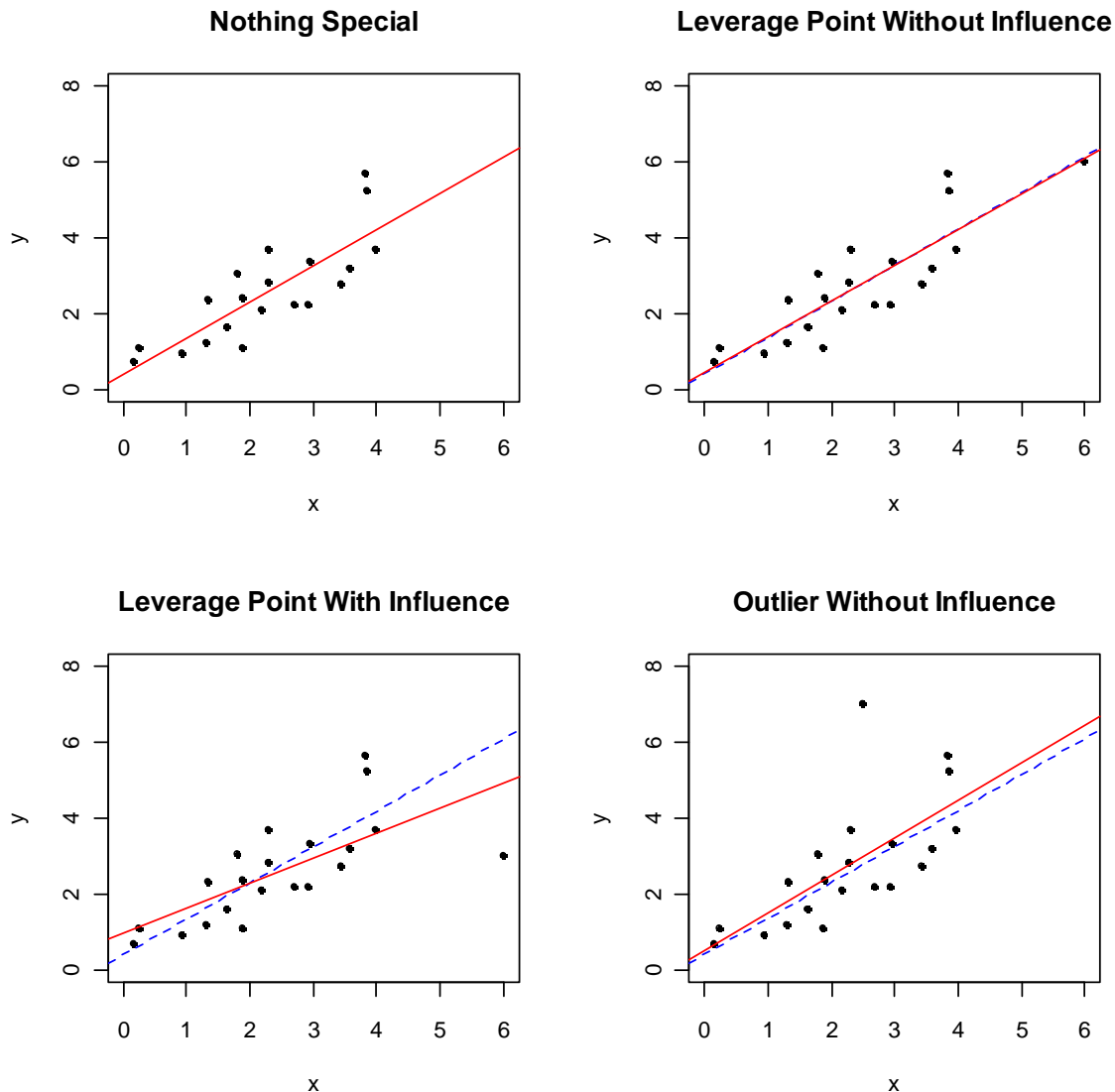


### 3.7.3 Influence Diagnostics

There are situations where the regression coefficient estimates are strongly influenced by one single, or just a few data points. This is suboptimal; it is important to recognize such situations and to identify these data points. However, the previously discussed residual plots are not always very useful for this task.

We will present the issue and the main definitions on the basis of a few artificial simple regression examples below. A *leverage point* is one with extreme  $x$ -value(s), i.e. lies “far” from the bulk of data. It is not necessarily an *influential data point*, but has a high potential to be so. The plots below illustrate this: the top left shows a “normal” situation without any leverage or influential points. Top right, a leverage point was added, but it is not influential, as it does not alter the regression line at all.

This is different at the bottom left: the leverage point now is an influential data point, i.e. the red regression line differs markedly from the blue one, which was computed by omitting the leverage point. Finally, the bottom right panel shows an outlier, which has relatively little influence on the regression line. This is because it has an  $x$ -value which is close to  $\bar{x}$ .



In the above examples, we determined the properties *leverage point* and *influential point* by visual inspection, and by omitting data points from the computation of the regression line. This works in simple situations, but is relatively cumbersome to generalize. If the influence of any data point in a sample shall be determined, we require running  $(n+1)$  regressions, i.e. one with all the data points, and one each with omitting one data point at a time. This is quite laborious, and additionally it requires some numerical criteria with which one quantifies the change in the regression line if a particular data point was left out. In the following, we will present the concepts of *Leverage* and *Cook's Distance*. They allow quantifying the potential for change, resp. the change that is induced by each data point, and this even directly without running  $(n+1)$  regressions.

### Leverage

The leverage of a data point is relatively easy to determine. It simply corresponds to  $H_{ii}$ , the  $i^{th}$  diagonal element of the hat matrix  $H$ . This makes sense, as if the response  $y_i$  changes by  $\Delta y_i$ , then  $H_{ii}\Delta y_i$  is the change in the fitted value  $\hat{y}_i$ . Thus, a high leverage for the  $i^{th}$  data point means that it has a strong potential to alter the regression line and force it to fit well to it. We have:

$$0 \leq H_{ii} \leq 1 \text{ for all } i, \text{ and } \sum H_{ii} = p+1.$$

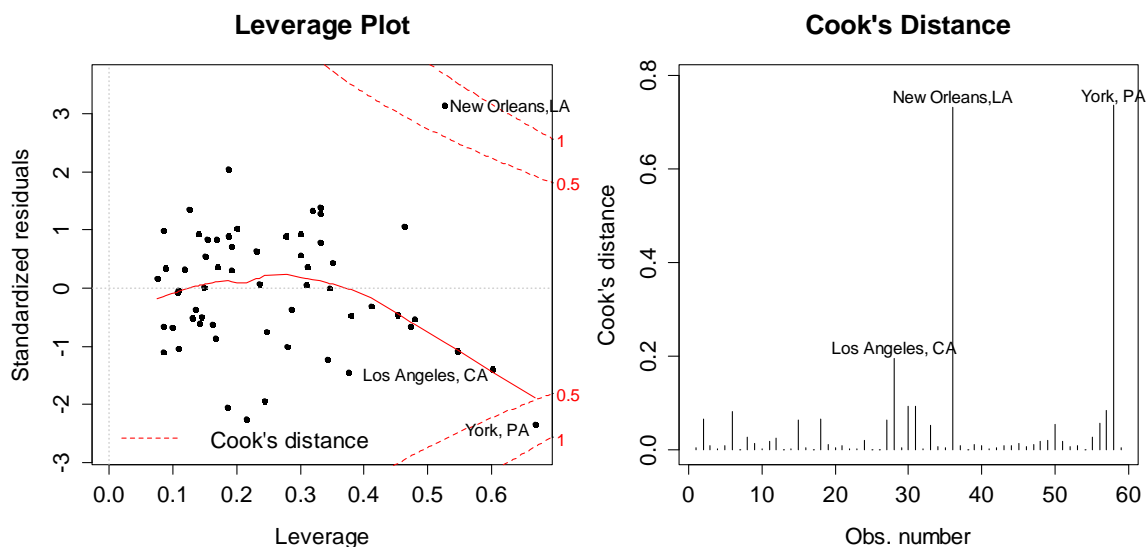
Hence, the average leverage is  $(p+1)/n$ , and all data points exceeding twice that value, i.e. have  $H_{ii} > 2(p+1)/n$ , are regarded as *leverage points*. As we have seen above, observations that have high leverage and at the same time a large residual are influential. We need to identify these!

### Cook's Distance

In brief summary, a leverage point tells us how strongly a data point may force the regression line to run through it. Whether it does so or not largely depends on the size of its residual. A direct measure for the change in the regression fit by a certain data point could be obtained by omitting the  $i^{th}$  data point and re-computing the fit without it. This is the basis for defining Cook's Distance:

$$D_i = \frac{\sum (\hat{y}_k^{[-i]} - \hat{y}_k)^2}{(p+1)\sigma_E^2} = \frac{H_{ii}}{1-H_{ii}} \cdot \frac{\tilde{r}_i^2}{(p+1)}$$

In the above equation,  $\hat{y}_k^{[-i]}$  is the fitted value for the  $k^{th}$  data point in a regression, where the  $i^{th}$  data point has been omitted. The sum in the above formula goes over all data points except the  $i^{th}$ , i.e.  $k = 1, \dots, i-1, i+1, \dots, n$ . As the right hand side shows, there is a direct way of obtaining *Cook's Distance* which does not require running multiple regressions. It suffices to know the hat matrix and the standardized residuals.



The default residual analysis in R shows the *Leverage Plot* as in the left panel above. It shows the standardized residuals vs. the leverage; but also Cook's distance is featured as contours for values of 0.5 and 1. Data points exceeding these are influential, resp. potentially damaging to the analysis. If there are no Cook's Distance contours in a Leverage Plot, this is because they do not fall within the plotting frame, hence you don't need to worry about influential data points. Obtaining Cook's Distance for all data is possible with `plot(fit, which=4)`. This yields the right panel in the plot above. We observe that except for *New Orleans (LA)* and *York (PA)*, Cook's Distance is uncritical. Treatment and explanation for these two data points are discussed below.

### Dealing with Influential Data Points and Outliers

We have seen above that the “most dangerous” data points are the ones that are leverage points and outliers at the same time. Also, we explained that Cook's Distance is a well suited measure to identify such points. However, here are some more things to consider about the presence of influential data points:

- 1) An influential data point in one model may disappear in another where variables have been changed or transformed. One needs to reinvestigate the question of influential data points when the model is changed.
- 2) The error distribution may not be Gaussian and thus, larger residuals may need to be expected. For example, day-to-day relative changes in stock indices seem Gaussian over large periods of times, but large changes also happen once in a while.
- 3) A single or very few outliers are usually much less of a problem in larger datasets. A single point will mostly not have the leverage to affect the fit very much. It is still worth identifying outliers if these types of observations are worth knowing about in the particular application.

Suppose that you detected one or several influential data points or outliers in your data. What to do with them? The following can serve as a practical guide:

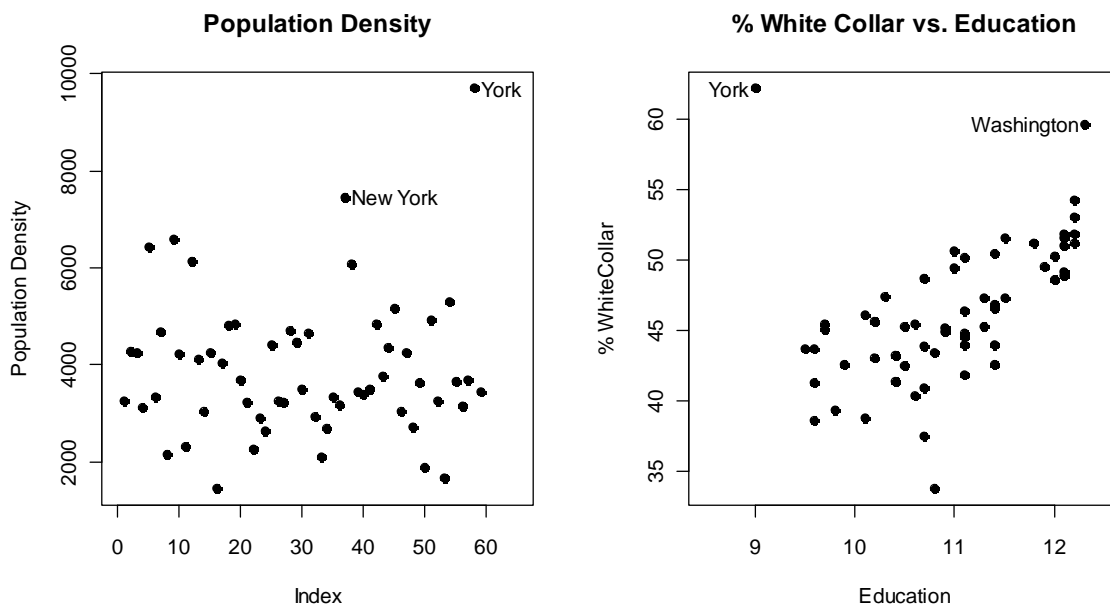
- a) Check for typos first, if the original source of the data is available.
- b) Examine the physical context – why did it happen? Sometimes, influential data points may be of little interest. On the other hand, it was often the case that scientific discoveries arose from noticing unexpected aberrations.
- c) Exclude the influential data points from the analysis, and re-fit the model. The differences can be substantial and make the difference between getting a statistically significant result, or having some “garbage” that cannot be published. To avoid any suggestion of dishonesty always report the existence of data points that were removed from the final model.
- d) Suppose there are outliers that cannot be reasonably identified as mistakes or aberrations, but are viewed as naturally occurring, e.g. due to long-tailed error distribution. Rather than excluding these instances and the using least squares, it is more efficient and reliable to use robust regression.

### 3.7.4 Example: Mortality Dataset

From the model diagnostics, we conjecture that York and New Orleans are the most influential data points. To be on the safe side, it is reasonable to re-run the regression analysis without these two data points. The most important observations from this analysis are that the residual standard error is now smaller, and the coefficient of determination increased. Thus, the fit is better now.

We now turn our attention to the interesting question why the cities of York and New Orleans were influential data points. Plotting some of the predictors, maybe even against other predictors and identifying outlying data points may help. In the plots below, we observe that the city of York has a considerably higher population density than all the other towns. It turned out that the definition of districts with which the population density was defined was somewhat suboptimal.

Moreover, it is also striking that the average years of education in York are much lower than elsewhere, but the percentage of white collar workers is higher. This anomaly is explained by the predominance of Amish people in that region. It is thus, an inhomogeneity of the sample.



### 3.7.5 Further Residual Plots

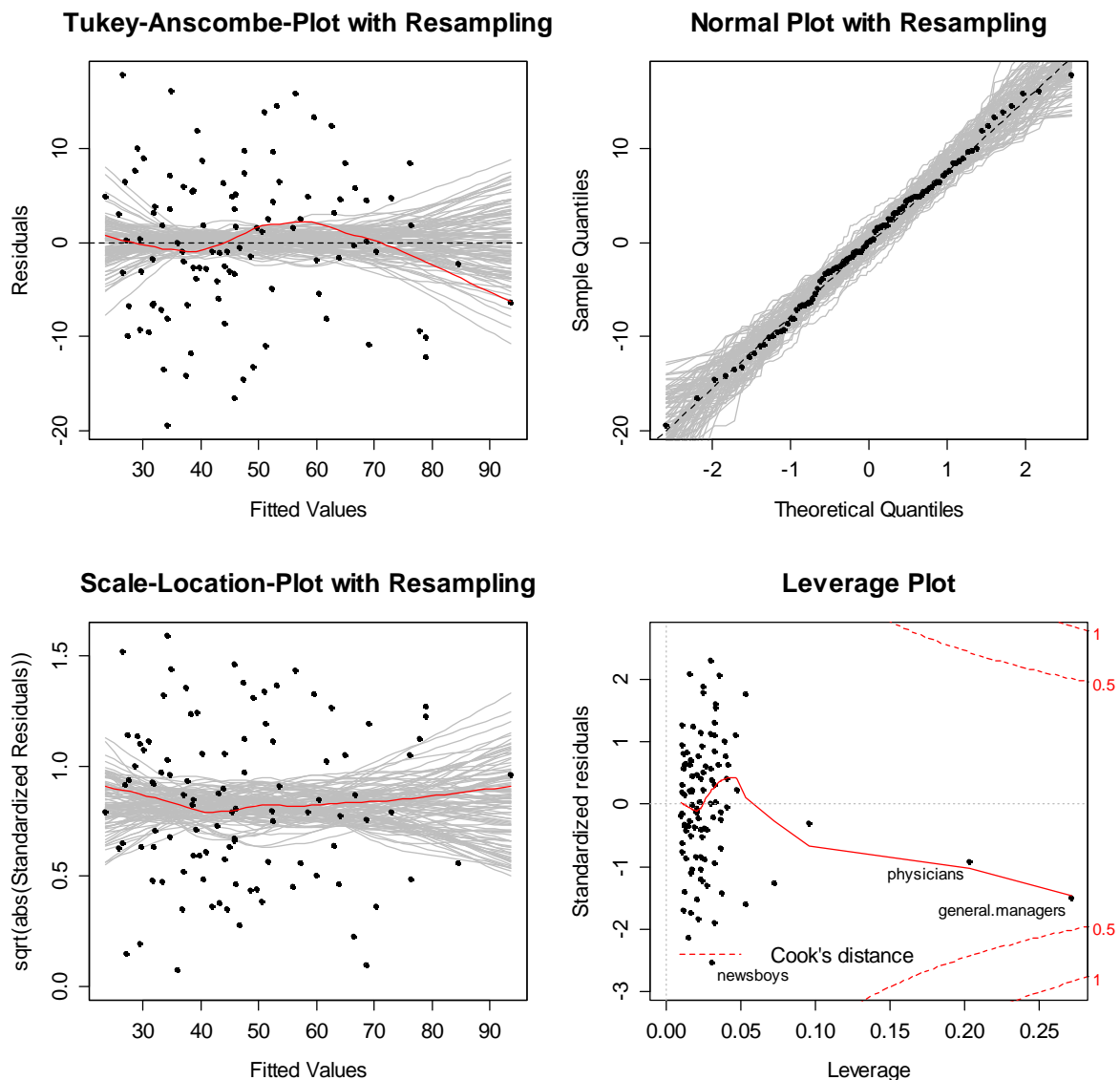
It is perfectly valid to plot the residuals from a regression model against any variable we like, be it a predictor, a not yet used potential predictor or other variables such as the time or sequence of the observations. There is one important rule for all these plots: *if any non-random structure is evident, the model has deficiencies and can be improved.* We will illustrate this using a dataset where the prestige of 102 occupations was measured with a survey. The prestige score is the response variable and there are 5 potential predictors, namely the average number of years of education that people in that profession have, the average

income, the percentage of women, the census (which is an occupational code) and the type, a categorical variable with the 3 levels professional/managerial, blue collar and white collar. For the purpose of exposition, we first fit a very simple model, i.e.

```
prestige ~ income + education
```

The further three potential predictors are omitted from the model, such that we can study the deficiencies that appear. We first fit the model and study the 4 standard residual plots in R, enhanced with the resampling based confidence regions. The author has implemented that functionality in a procedure called `resplot()`:

```
> fit <- lm(prestige ~ income + education, data=Prestige)
> resplot(fit)
```



From the summary, we gather that the global F-test is highly significant and that the R-squared reaches a value close to 0.8. Also when inspecting the residual plots, the fit seems reasonable. There may be a slight systematic error visible in

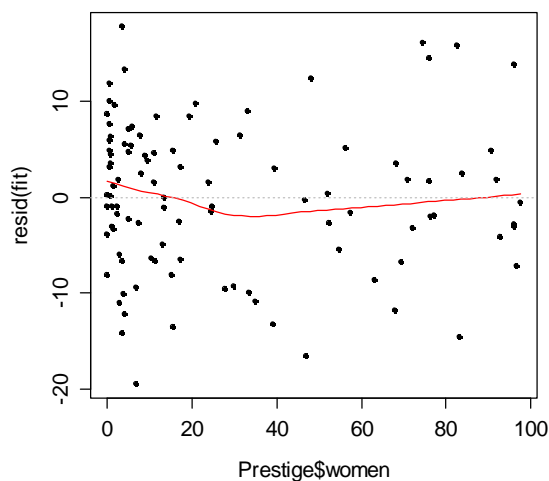
the Tukey-Anscombe plot and we observe two leverage points, namely physicians and general managers. These are the two professions with the highest average income. These problems are not severe and overall, the model is certainly of a reasonable quality. However, as we will see below, it is pretty easy to come up with a better model for `prestige`.

### Residuals vs. Potential Predictors

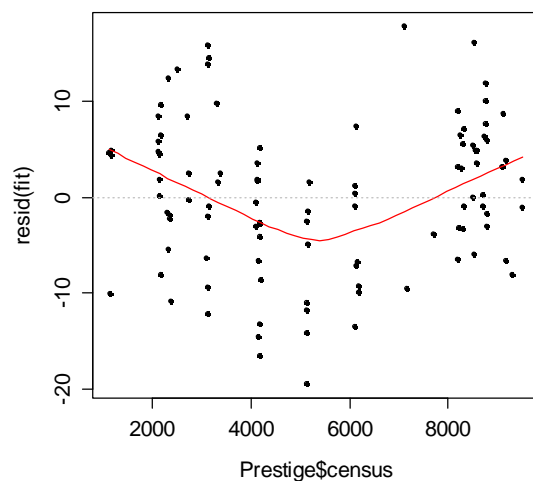
As mentioned in the introduction, we may plot the residuals against any variable we like. Here, we first study what we observe if the residuals are plotted versus the potential further predictors.

```
> plot(resid(fit) ~ Prestige$women, pch=20)
> lines(loess.smooth(Prestige$women, resid(fit)), col="red")
> abline(h=0, col="grey", lty=3)
> plot(resid(fit) ~ Prestige$census, pch=20)
> lines(loess.smooth(Prestige$census, resid(fit)), col="red")
> abline(h=0, col="grey", lty=3)
> plot(resid(fit) ~ Prestige$type, col="limegreen")
```

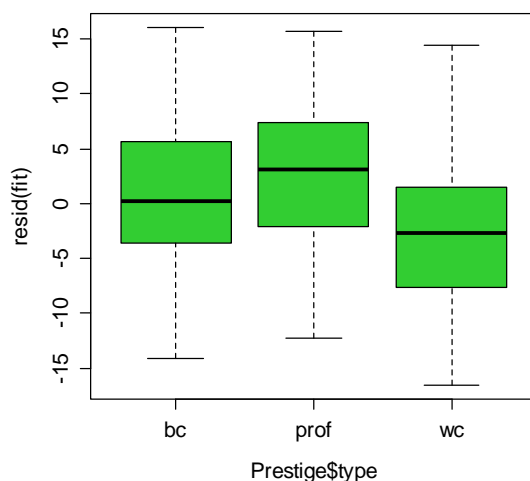
**Residuals vs. Women**



**Residuals vs. Census**



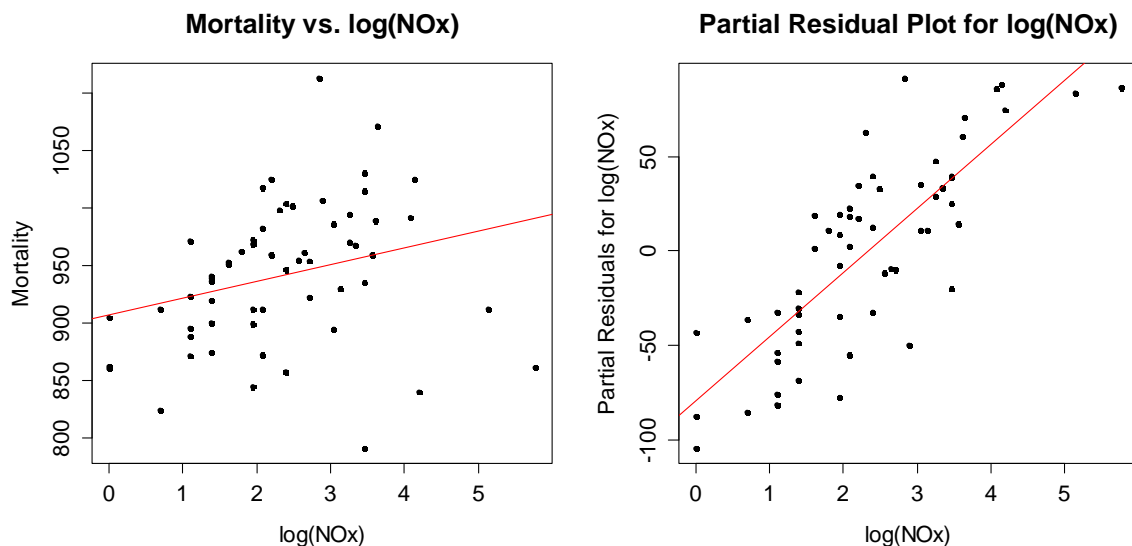
**Residuals vs. Type**



There does not seem to be much of a relation between the percentage of women in a profession and the residuals, but the other two variables `census` and `type` certainly have a non-random relation with the residuals. Especially for `census`, we quite strongly underestimate the prestige of jobs with values in the 4000-6000 range. A similar observation is made for the `type`: the prestige of managerial jobs is underestimated by our current model, while the one of white collar professions is overestimated. We can and should cure these problems by integrating the predictors into the model.

### Partial Residual Plots

In many applied problems, it is very interesting to understand and visualize the relation between the response and some arbitrary predictor  $x_k$ . However, a plot of  $y$  vs.  $x_k$  can be deceiving, because in a multiple regression setting, all other predictors  $x_1, \dots, x_{k-1}, x_{k+1}, \dots, x_p$  will simultaneously have an effect on the response. Hence, what we should aim for is displaying the relation of  $y$  vs.  $x_k$  under the presence of the other predictors. That is what the partial residual plot does. We try to illustrate the idea with an excerpt from the mortality dataset.



The left panel shows the plain relation between the response and the logged nitrous oxide. The problem with this plot is that mortality in the different cities is affected by other factors than  $\log(NO_x)$ , too. We try to improve upon this with the plot on the right. It shows the partial residual plot for  $\log(NO_x)$ . The basic idea is to generate an updated  $y$  variable, where the effect of all other predictors is removed from the response. This is the verbal definition of a partial residual. Mathematically, the partial residuals for predictor  $x_k$  are:

$$y - \sum_{j \neq k} \hat{\beta}_j x_j = \hat{y} + r - \sum_{j \neq k} \hat{\beta}_j x_j = \hat{\beta}_k x_k + r$$

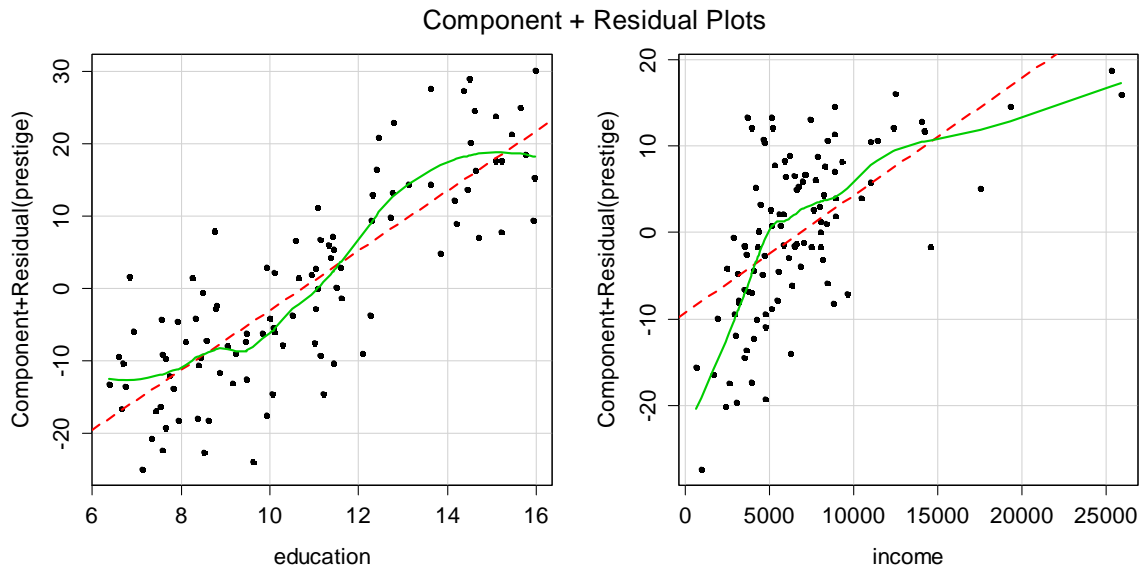
As we can see from the formula, the initial idea is to adjust the response by the estimated effect of the other predictors. We can reformulate this as adjusting the



residuals by the estimated effect of predictor  $x_k$ . While the latter is a bit more difficult to comprehend, it is much more convenient for computation. The partial residuals can easily be accessed in R by typing:

```
> residuals(fit, type="partial")
```

This yields a  $n \times p$  matrix that has the partial residuals for all the predictors. Even more convenient functionality can be found in the function `crPlots()` in `library(car)`. We display the result for the prestige example:



The partial residual plots are enhanced with the red dashed line that illustrates the actual fit according to the multiple linear regression model. The green solid line is a smoother that was added for visualizing the true relation between partial residual and predictor. This allows for gaining a lot of insight into the model. As we can see, the `income` variable has a non-linear (but approximately logarithmic) relation to its partial residuals. Hence, it would be wise to use `log(income)` rather than the untransformed variable. This should not come as a surprise, as its properties with only positive values on a right-open, relative scale and a right-skewed marginal distribution speak for that, too. Even more interesting is the observation in the left hand panel. There seems to be a pronounced difference in the prestige of jobs that require  $>12$  years of education, i.e. a university degree. Hence, adding a factor variable that codes for jobs that require a degree might improve the fit strongly. As some further experimentation shows, this is indeed the case. A similar effect is achieved if variable `type` is added to the model, as the `prof/managerial` jobs are the ones requiring  $>12$  years of education.

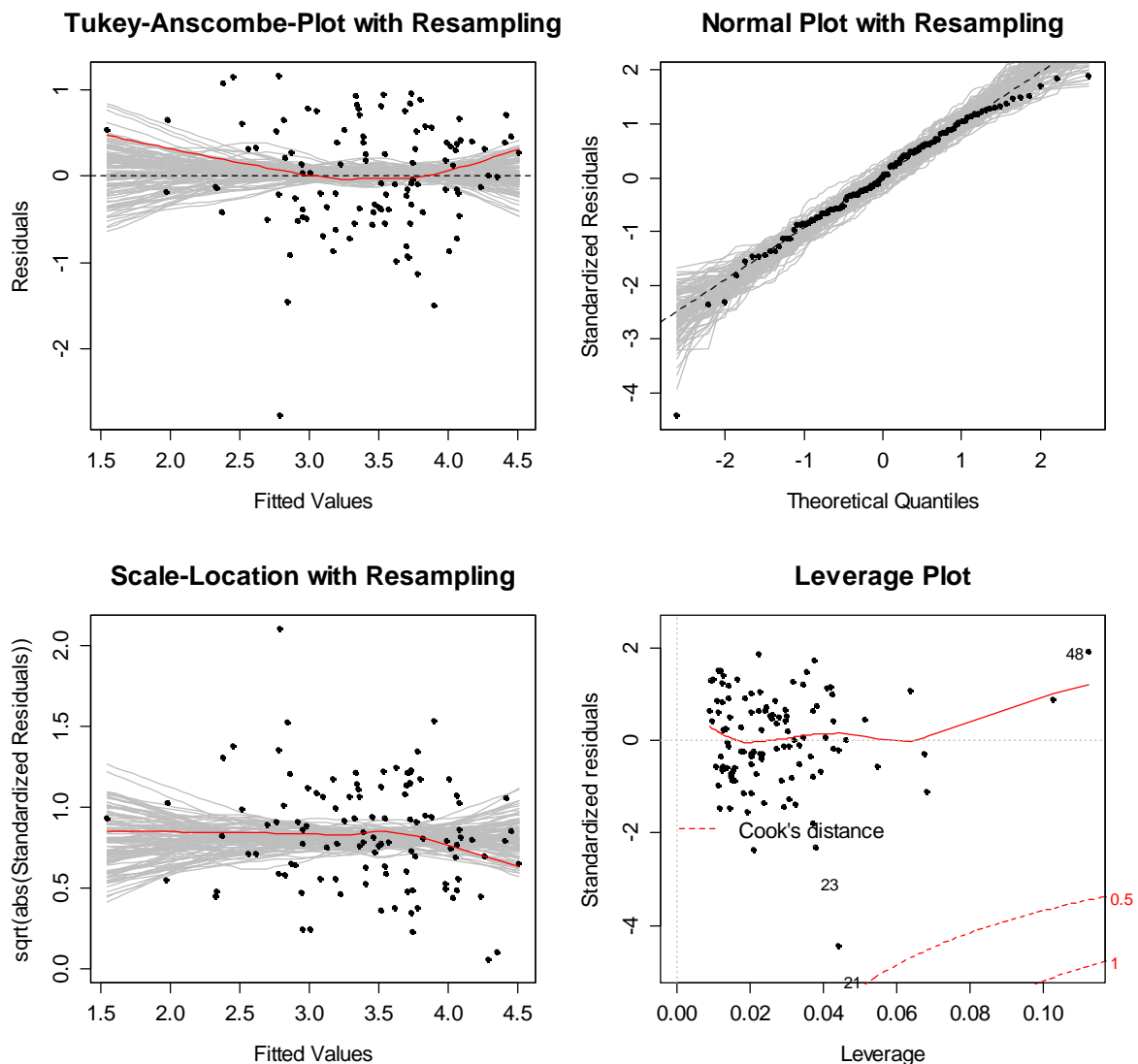
We finish this section by summarizing that the partial residual plots allow for perceiving how the predictors act in a multiple linear regression model. If there appears a significant difference between their actual, linear fit and the true relation indicated by the smoother, one should improve the model. Sometimes, we can transform predictors to achieve this; at other times adding additional predictors and/or interaction terms may help.

### 3.7.6 Dealing with Correlated Errors

For the efficiency of the OLS algorithm and the validity of inference results, we require uncorrelated resp. stochastically independent error terms. With the standard residuals plots discussed above, this condition will not be verified. As it turns out, data with temporal or spatial structure often happen to show dependent errors and require a second thought. We will illustrate the issue with an example, where ozone levels in New York are predicted from solar radiation and wind. The data can be found in `library(faraway)` by loading `data(airquality)`. The model is as follows:

$$\log(\text{Ozone}) \sim \text{Solar.R} + \text{Wind}$$

The measurements were made on 153 consecutive days, but as some data are missing, only 111 observations remain for fitting the multiple linear regression model. We use the OLS algorithm, but as the data clearly have a temporal structure, a second thought about potential error/residual correlation is necessary. The four standard residual plots for the above model are as follows:

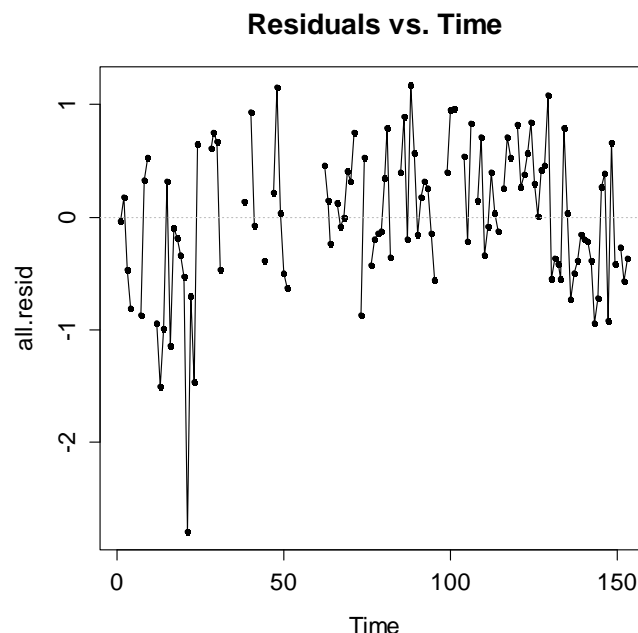


The Tukey-Anscombe Plot looks somewhat borderline, while the remaining plots do not seem worrisome. As there are no simple means resp. transformations with which the model can be improved further, we might be tempted to carry on and take the inference results at face value:

```
> summary(fit)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.9519449  0.2337241  16.909 < 2e-16 ***
Solar.R      0.0037215  0.0006773   5.494 2.63e-07 ***
Wind        -0.1231183  0.0173535  -7.095 1.42e-10 ***
---
Residual standard error: 0.6423 on 108 degrees of freedom
(42 observations deleted due to missingness)
Multiple R-squared:  0.4598, Adjusted R-squared:  0.4498
F-statistic: 45.96 on 2 and 108 DF,  p-value: 3.612e-15
```

However, this bears some danger as the reported standard errors and p-values may be wrong if the errors/residuals are (sequentially) correlated. We can verify this by generating a time series plots of the residuals:

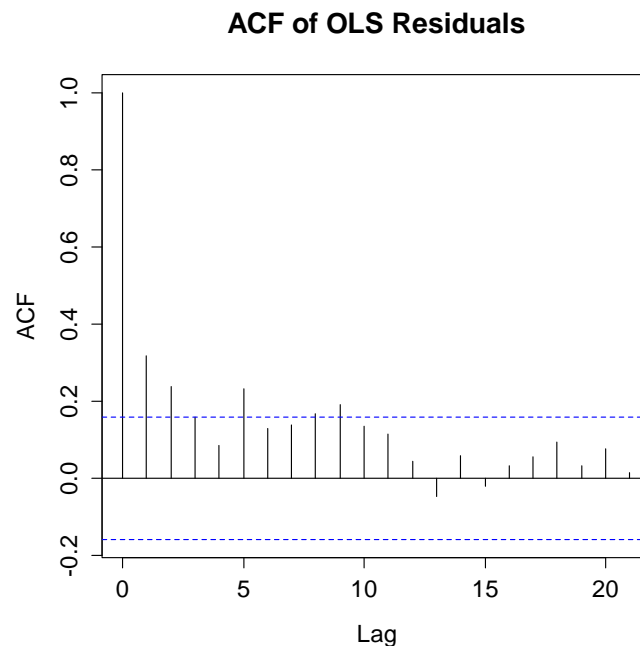
```
> all.resid <- rep(NA, 153)
> all.resid[as.numeric(names(resid(fit)))] <- resid(fit)
> ts.plot(all.resid, main="Residuals vs. Time")
> points(all.resid, pch=20)
> abline(h=0, col="grey", lty=3)
```



Coming to the right conclusion may require some experience in reading time series plots, but obviously there is some non-random structure. The residuals in the middle of the observation period are systematically positive, while those at the start and end are negative. This means that our current model underestimates the

ozone levels in the middle, but overestimates them at both ends. This is because our observation period starts in winter, ranges over summer and ends in winter again; and ozone levels are generally higher in summer. Hence, we have a temporal effect and correlation of residuals here. For those readers who are familiar with time series analysis, it may be obvious that generating a correlogram of the residuals may help in deciding whether there is correlation or not.

```
> acf(all.resid, na.action=na.pass, main="ACF ...")
```



As we can see, there is some positive sequential correlation among the residuals. This violates the OLS assumptions and should not be lightly ignored. Before addressing the consequences and remedies, we first present the Durbin-Watson-Test which provides another alternative to check for correlation among the errors/residuals. The null hypothesis is “no autocorrelation”, which is tested on the basis of the following test statistic:

$$DW = \frac{\sum_{i=2}^n (r_i - r_{i-1})^2}{\sum_{i=1}^n r_i^2}$$

If consecutive errors are uncorrelated,  $(r_i - r_{i-1})^2$  will have the same expectation but bigger scatter than  $r_i^2$  and  $DW \approx 2$ . If there is positive autocorrelation among consecutive errors, then  $(r_i - r_{i-1})^2$  will be systematically smaller than  $r_i^2$ , and vice versa if there is negative autocorrelation. Acceptance and rejection regions can be determined for an approximation to the Chi-Square distribution. We omit the details here and focus on the R implementation:

```
> library(lmtest)
> dwtest(log(Ozone) ~ Solar.R + Wind, data=airquality)
```

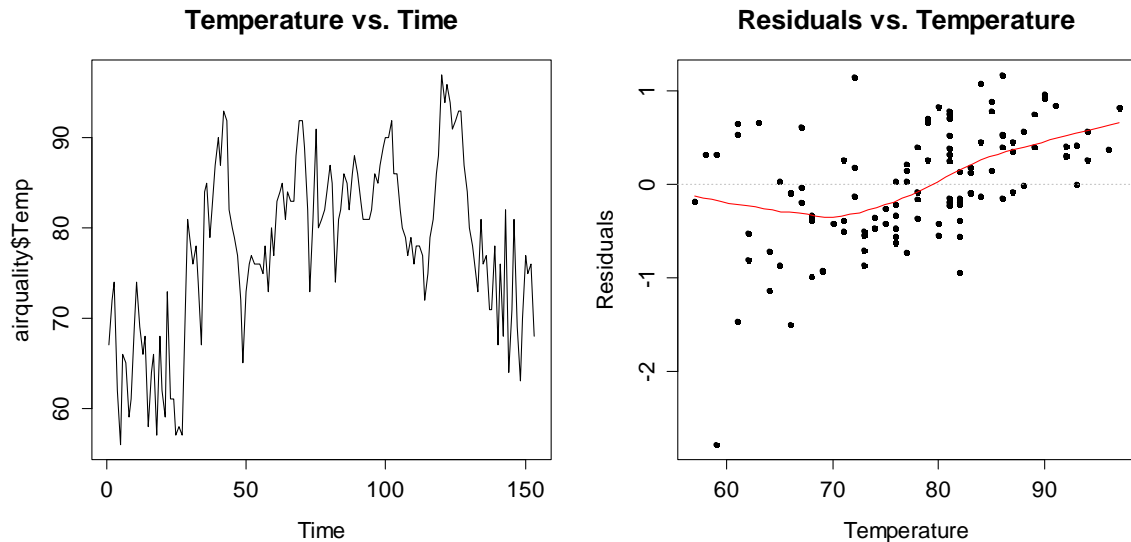
```
Durbin-Watson test
data:  log(Ozone) ~ Solar.R + Wind
DW = 1.4551, p-value = 0.001734
alternative hypothesis: true autocorrelation greater than 0
```

It turns out that for our particular example, the p-value is 0.001734 and hence the null hypothesis of no autocorrelation is rejected in favor of the alternative. Hence, as we had detected visually before, also the test diagnoses correlated residuals. The Durbin-Watson test requires some caution though: while rejection of the null hypothesis (up to the 5% of type I errors) indicates a problem with correlated residuals, the opposite is not true. If the null is accepted, we cannot necessarily conjecture that everything is fine. The Durbin-Watson-Test will reveal situations where consecutive residuals are positively or negatively correlated. But as soon as the correlation structure of the errors is more complex (even when it is still a sequential correlation) it may well fail due to the simple nature of the test statistic that is used. In such cases, only visual inspection or the use of more sophisticated tests such as Box-Ljung may reveal the issue. As intimate knowledge of time series analysis is required for these tools, we do not further pursue the issue here.

So what is the actual problem if there are correlated errors? As long as the model does not feature a systematic error, the regression coefficients and the fitted values are still unbiased. Thus, if the principal goal with a regression model is prediction, one may even leave the error correlation unaccounted for. However, theory tells us that the OLS algorithm is no longer efficient in this case, i.e. there may be alternative regression estimators that yield more precise estimates of both regression coefficients and fitted values. However, the biggest issue is with the standard errors which are biased in case of correlated errors. This will inevitably lead to flawed inference results (i.e. tests and confidence intervals) and is dangerous even for the practitioner! Please note that the standard errors, depending on the nature of the correlation, can be either too small (majority of cases, spurious significance of predictors is the consequence) or also too big.

Once correlated residuals/errors have been diagnosed, the question is how to address the issue. Ignoring it is a poor solution, but in case of weak correlation and a primary focus on prediction rather than precise inference results sometimes a viable strategy in practice. Another (much better) option is to use the Generalized Least Square (GLS) approach. This is a least square based estimation procedure which does not assume a diagonal covariance matrix of the error vector and hence can account for the correlation. However, it requires specifying a time series model for the dependency in the error term. This is beyond the scope of this course and will not be pursued here. Details about the GLS procedure, its application and evaluation are taught in the Applied Time Series Analysis course which is held by the author of this script every spring semester at ETH Zürich, or can also be found in the respective script. Finally, the best strategy for dealing with correlated errors is to identify the hidden variable which is responsible for it and add it to the model.

Here in particular, the data frame with the airquality data also contains information about the temperature on the observation days. We can well suspect that there is a temperature influence on the ozone levels. To learn more, we generate a time series plot of temperature and plot the residuals from the model versus temperature.



We observe that there is a clear temporal signal in the temperature (i.e. temperatures in summer which corresponds to times 40-120 are higher than in spring and fall) as well as non-random structure in the residual plot. As we can see, the higher the temperature, the more positive the residuals are. Or in other words, when temperatures are high, our model underestimates the ozone levels. As a remedy, we add the temperature variable to the model.

```
> fit02 <- lm(log(Ozone) ~ Solar.R + Wind + Temp, data=airq..)
> summary(fit02)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	-0.2621323	0.5535669	-0.474	0.636798
Solar.R	0.0025152	0.0005567	4.518	1.62e-05 ***
Wind	-0.0615625	0.0157130	-3.918	0.000158 ***
Temp	0.0491711	0.0060875	8.077	1.07e-12 ***

---

Residual standard error: 0.5086 on 107 degrees of freedom  
(42 observations deleted due to missingness)

Multiple R-squared: 0.6644, Adjusted R-squared: 0.655

F-statistic: 70.62 on 3 and 107 DF, p-value: < 2.2e-16

Temperature is highly significant and cures many of the problems that existed with the previous model. There may be a slight systematic error with low temperatures that raises some questions (data not shown). This is due to inversions that often happen in cold weather conditions; hence further model enhancements are necessary. However, the issue with the correlated residuals has already been solved at this state:

```

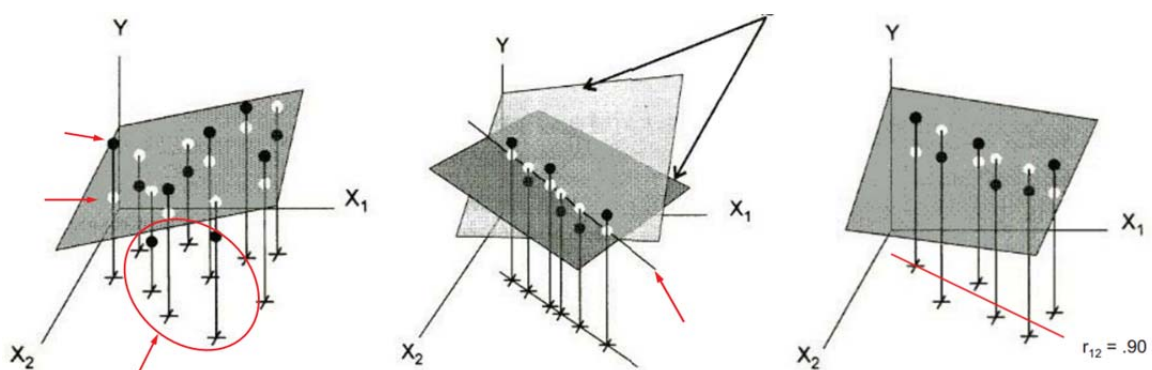
> dwtest(log(Ozone) ~ Solar.R + Wind + Temp, data=airquality)
Durbin-Watson test
data:  log(Ozone) ~ Solar.R + Wind + Temp
DW = 1.8068, p-value = 0.1334
alternative hypothesis: true autocorrelation greater than 0

```

Of course it would have been sensible to add temperature into this model right from the start and the author would advise to use all given variables that are relevant to a regression problem. The procedure in this section was tailored for exploratory purposes, i.e. for showing that sequential correlation of residuals may exist and that it is often most efficiently cured by adding some forgotten predictors to the model, rather than using more sophisticated methods as a remedy.

### 3.8 Multicollinearity

Multicollinearity is a potential problem with a regression model that does not arise from the errors/residuals, but with the predictors. As we already know, a multiple linear OLS regression will not have a unique solution if it has singular design. As explained before, this happens when exactly collinear predictors (i.e. constant variables, duplicates of variables, same variables in different units, circular variables) are present. A singular design is relatively easy to identify in practice, as R will omit some redundant terms from the model and report their coefficients as being NA. Now multicollinearity is the case where there is near, but not perfect linear dependence among the predictors in the design matrix  $X$ . This case is more delicate than perfect dependence in practical regression analysis, as there will be a (technically) unique solution, but it is often highly variable and of poor quality. In particular, the estimated coefficients will feature large or even extreme standard errors, i.e. they are imprecisely estimated with huge confidence intervals. It is also typical that the global F-Test shows a significant result, but none of the individual predictors has a p-value below 0.05. Furthermore, also numerical problems in determining the regression coefficients may arise, and extrapolation may yield extremely poor results. With the following illustration, we try to build some understanding what multicollinearity means. In all examples, a two-predictor multiple linear regression model  $\hat{y} \sim x_1 + x_2$  is fitted. We can see the regression hyperplanes, the original observations (black dots), the fitted values (white dots) and the projection of the data points (crosses) onto the  $x_1/x_2$ -plane.



The left panel shows a situation where the two predictors do not show multicollinearity. The regression hyperplane has a solid fundament and hence, the estimated coefficients will be precise with small standard errors. In the middle panel, we have exact collinearity. This is the case where there is no unique solution to the regression problem. As shown in the picture, there are several regression hyperplanes that provide an equally acceptable solution, i.e. all of them fulfill the least square condition. In this example, the  $y \sim x_1 + x_2$  relation can be reduced to either  $y \sim x_1$  or  $y \sim x_2$  without any loss of information and as mentioned above, R would automatically omit one of the predictors (more precisely, it would drop the collinear predictor that appears last in the formula specifying the model). The third panel shows a situation where the predictors feature multicollinearity. The Pearson correlation coefficient of the projected data points is 0.90. In consequence, the regression hyperplane has a poor fundament. It is wiggly in the sense that even a slight change in the data might lead to a big change in the solution, i.e. the hyperplane. This is also expressed by the large standard errors of the regression coefficients. In summary, we should try our best for avoiding such multicollinearity. But first of all, we present a real-world example and address the issue how multicollinearity can be identified.

### 3.8.1 Identifying Multicollinearity

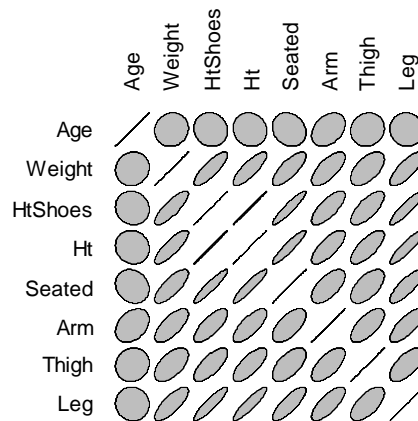
Understanding how car drivers adjust their seat would greatly help engineers to design better cars. For this reason, researchers at the HuMoSim laboratory at the University of Michigan collected data on 38 drivers. These can be found in R with `data(seatpos)` in `library(faraway)`. The response variable is `hipcenter`, which is the horizontal distance between the hips and the steering wheel. There are a number of predictors, namely `Age` (in years), `Weight` (in pounds), `HtShoes`, `Ht`, `Seated` (height with and without shoes, and seated height) and `Arm`, `Thigh`, `Leg` (length of these extremities). Variable transformations are not imminent here and we start with a model using all these (obviously strongly correlated) predictors.

```
> fit <- lm(hipcenter ~ ., data=seatpos)
> summary(fit)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  436.43213   166.57162    2.620   0.0138 *
Age           0.77572     0.57033    1.360   0.1843
Weight       0.02631     0.33097    0.080   0.9372
HtShoes     -2.69241     9.75304   -0.276   0.7845
Ht           0.60134    10.12987    0.059   0.9531
Seated       0.53375     3.76189    0.142   0.8882
Arm         -1.32807     3.90020   -0.341   0.7359
Thigh       -1.14312     2.66002   -0.430   0.6706
Leg         -6.43905     4.71386   -1.366   0.1824
---
Residual standard error: 37.72 on 29 degrees of freedom
Multiple R-squared:  0.6866, Adjusted R-squared:  0.6001
F-statistic: 7.94 on 8 and 29 DF, p-value: 1.306e-05
```



We are in the typical situation where the global F-Test is significant, but none of the individual predictors is. Together with the a-priori-knowledge about the predictors, this is a clear indication of multicollinearity. However, there are some tools with which we can gain further insight. A relatively simple option is to determine or visualize all the pairwise Pearson correlation coefficients among the predictor variables:

```
> library(ellipse)
> plotcorr(cor(seatpos[, -9]))
```



The ellipses show both sign and magnitude of the pairwise correlation. Naturally, the diagonal entries are equal to +1 and not important to our analysis. However, some additional strong pairwise correlations exist as well, e.g. between Ht and HtShoes, Ht and Seated, Ht and Leg, et cetera. That does not really come as a surprise, though. Please note that such kind of pairwise analysis will not reveal all situations where multicollinearity is present as it does not need to be a pairwise phenomenon. It is easy to construct an example with circular variables where none of the pairwise relations exhibits strong correlation, but still near or even perfect collinearity exists. For gaining deeper insight, a more sophisticated approach is required. This is presented with the Variance Inflation Factor (VIF). It is based on the notion that the variance of an estimated regression coefficient can be rewritten in the form:

$$\text{Var}(\hat{\beta}_k) = \sigma_E^2 \cdot \frac{1}{1 - R_k^2} \cdot \frac{1}{\sum_{i=1}^n (x_{ik} - \bar{x}_k)^2}$$

The first term is the error variance, the last is a design component and the term in the middle  $VIF_k = 1 / (1 - R_k^2)$ , is the variance inflation factor for the  $k^{\text{th}}$  predictor. It is obtained by determining the coefficient of determination  $R_k^2$  in a regression where predictor  $x_k$  is the response variable, and all other predictors maintain their role. Obviously, the higher the collinearity between  $x_k$  and the other predictors, the

higher are  $R_k^2$  and hence also  $VIF_k$ . Rather than with tedious handwork and running  $p$  regressions, the VIFs can easily be obtained in R:

```
> vif(fit)
      Age      Weight    HtShoes      Ht
1.997931  3.647030 307.429378 333.137832
  Seated      Arm      Thigh      Leg
8.951054  4.496368  2.762886  6.694291
```

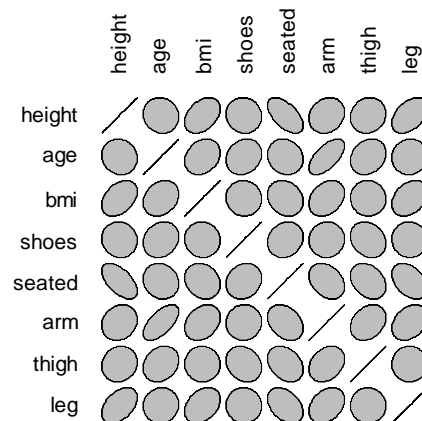
Hence predictor `Age` has the least inflated variance and `Ht` the most. For interpreting these values it is important to know that a  $VIF = 5$  corresponds to a  $R^2 = 0.8$ ; and  $VIF = 10$  to a  $R^2 = 0.9$ . As a rule of the thumb,  $VIF \geq 5$  indicates a multicollinearity problem, and  $VIF \geq 10$  dangerous multicollinearity which needs to be addressed. As we can see, in our example the factors are even much higher. As we can see, the variance of `Ht` is inflated by more than 300x, hence the standard deviation (i.e. standard error in R) is still by around 18x bigger than if no multicollinearity was present with the accordant consequences on the confidence interval for that predictor.

### 3.8.2 Dealing with Multicollinearity

Except for some simple cases, multicollinearity needs well thought-out action. A very basic approach is called *amputation*. It means that among all collinear predictors, all but one is discarded. In our example, amputation would reduce the set of predictors to `Age`, `Weight` and `Ht`. This is not really satisfactory as we would be discarding valuable information. One can well imagine that the length of the limbs relative to a person's height or the shoes may play their role in the driver's seat adjustment. We just have to make sure that the information is provided in a form in which it can be exploited by a multiple linear regression model. An often pursued strategy is the one of *creating new variables* out of the existing ones, such that the collinearities are broken. In our example, we take `Ht` as the key predictor which will be left alone. Most of the other predictors will be adjusted, i.e. taken relative to the body size. In particular:

```
age    <- Age
bmi    <- (Weight*0.454)/(Ht/100)^2
shoes  <- HtShoes-Ht
seated <- Seated/Ht
arm    <- Arm/Ht
thigh  <- Thigh/Ht
leg    <- Leg/Hat
```

Instead of weight, we are now using the BMI. The shoe height replaces the height with shoes, hence the collinearity to that variable is gone. The length variables for the limbs are expressed as fractions of the body height. This addresses potential different morphology of the test subjects. Is this relatively simple approach really successful in breaking all collinearities? We verify by plotting the pairwise correlations and then determine the VIFs from the updated model.



As we can see, the problem with pairwise correlations has been mitigated. However, this does not necessarily mean that there is no multicollinearity.

```
> my.fit <- lm(hipcenter ~ ., data=my.sp)
> summary(my.fit)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	632.0063	490.0451	1.290	0.207
height	-3.6521	0.7785	-4.691	5.98e-05 ***
age	0.7402	0.5697	1.299	0.204
bmi	0.4234	2.2622	0.187	0.853
shoes	-2.6964	9.8030	-0.275	0.785
seated	171.9495	631.3719	0.272	0.787
arm	-180.7123	655.9536	-0.275	0.785
thigh	-141.2007	443.8337	-0.318	0.753
leg	-1090.0111	806.1577	-1.352	0.187

---

Residual standard error: 37.71 on 29 degrees of freedom

Multiple R-squared: 0.6867, Adjusted R-squared: 0.6002

F-statistic: 7.944 on 8 and 29 DF, p-value: 1.3e-05

This fit still has nearly identical  $R^2$  and global F-Test result as the initial one with collinear variables. However, meaningful inference is now possible. From the summary with all variables, only height is significant, though.

```
> vif(my.fit)
  height      age      bmi      shoes
1.968447 1.994473 1.408055 1.155285
  seated      arm      thigh      leg
1.851884 2.044727 1.284893 1.480397
```

The VIFs from the updated fit are uncritical now. Creating new variables has been successful, i.e. the information was preserved, but multicollinearity was broken.

## 3.9 Variable Selection

In real-world regression problems, there is often a wealth of predictors and potential predictors available. Here, we show how we can select the “best” (or at least a good) subset of predictors. We first motivate why this is useful, then turn our attention to some strategies for finding the subset, and also discuss the meaning of the word “best” in terms of regression modeling.

### 3.9.1 Why Variable Selection?

Only in some rare special cases, we do already know the functional form with which a few specified predictors  $x_1, \dots, x_p$  explain the response  $y$ . In these cases, we would still be interested in learning about the regression coefficients, do some hypothesis tests, and potentially give some prediction and confidence intervals.

Much more frequently is the case where *regression* is used in an *explorative fashion*. This is when we do not know exactly how the relation between response  $y$  and the (potential) predictors  $x_j$  is, usually we do not even know which predictors to keep/use and which ones to skip. Our goal with regression analysis will then be to learn not only about the form of the relation between response and predictors, but also about required variable transformations, and probably most importantly, about the *predictors* that have a *relevant impact* on the outcome.

Thus, there is motivation for variable selection arising purely from applied aspects. However, there is some more technical reasoning for keeping a model small:

- 1) We generally want to explain the data in the simplest way, and thus remove redundant predictors. This follows the idea that if there are several plausible explanations (i.e. models) for a phenomenon, then the simplest is the best.
- 2) Unnecessary predictors in a regression model will add noise to the estimation of the coefficients for the other predictors. Or in other words: we need more observations to have the same estimation accuracy.
- 3) What is stated in 2) above becomes even more pronounced if there is collinearity among the predictors, i.e. if there are too many variables trying to do the same job. Removing excess predictors facilitates interpretation.
- 4) If the model is to be used for prediction, we will be able to save effort, time and/or money if we do not have to collect data for redundant predictors.

Please note that variable selection is not a method. It is a process that cannot even be separated from the rest of the analysis. For example, outliers and influential data points will not only change a particular model – they can even have an impact on the model we select. Also variable transformations will have an impact on the model that is selected. Some iteration and experimentation is often necessary for variable selection, i.e. to find smaller, but better models.

### 3.9.2 Backward Elimination with p-Values

Perhaps the easiest idea to deal with the variable selection problem would be to omit all predictors which turn out to be non-significant. This is an established procedure, but requires some caution. In section 3.4 we had argued that the p-values from the individual hypothesis tests are valid only if all other predictors are kept in the model. So do not fall into the trap of omitting all non-significant predictors from a regression summary at once!

```
> summary(f.full)
```

Call:

```
lm(formula = Mortality ~ JanTemp + JulyTemp + RelHum + Rain +
    Educ + Dens + NonWhite + WhiteCollar + log(Pop) + House +
    Income + log(HC) + log(NOx) + log(SO2), data = apm)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	1.166e+03	2.470e+02	4.718	2.43e-05	***
JanTemp	-4.263e+00	1.593e+00	-2.676	0.0104	*
JulyTemp	-3.153e+00	3.656e+00	-0.863	0.3931	
RelHum	3.420e-01	1.059e+00	0.323	0.7482	
Rain	5.879e-01	2.322e-01	2.532	0.0150	*
Educ	-1.000e+01	9.087e+00	-1.101	0.2771	
Dens	4.525e-03	4.223e-03	1.072	0.2897	
NonWhite	5.152e+00	1.002e+00	5.143	6.01e-06	***
WhiteCollar	-1.883e+00	1.198e+00	-1.572	0.1232	
log(Pop)	4.391e+00	7.714e+00	0.569	0.5721	
House	-4.574e+01	3.939e+01	-1.161	0.2518	
Income	-6.892e-04	1.334e-03	-0.516	0.6081	
log(HC)	-2.204e+01	1.523e+01	-1.447	0.1550	
log(NOx)	3.397e+01	1.425e+01	2.384	0.0215	*
log(SO2)	-3.687e+00	7.359e+00	-0.501	0.6189	

---

Residual standard error: 34.48 on 44 degrees of freedom

Multiple R-squared: 0.7685, Adjusted R-squared: 0.6949

F-statistic: 10.43 on 14 and 44 DF, p-value: 8.793e-10

We emphasize again that there is no basis for dropping all the non-significant terms, i.e. JulyTemp, RelHum, Educ, Dens, et cetera from the model. However, a theoretically valid option consists of performing *stepwise backward elimination* from the full model. This is a simple and still popular variable selection procedure that does not require more than the summary output. But please be aware of the fact that it is generally better to use function `drop1()` instead of `summary()`. For the mortality dataset with exclusively numerical predictors, there will be no difference between the two, but if factor variables or interaction terms are present, only `drop1()` lets you take the correct decisions. The idea is now that the term with the highest p-value is kicked out of the model – at least as long as its p-value is bigger than  $\alpha_{crit}$ , which is mostly chosen as 0.05. In our example, this is predictor RelHum, and it is most convenient to use the `update()` function.

```

> fit01 <- update(f.full, .~-RelHum)
> drop1(fit01, test="F")
<none>                52436 428.60
JanTemp      1      8625 61062 435.58  7.4020 0.009229 **
JulyTemp     1      1576 54012 428.35  1.3524 0.250987
Rain         1      8134 60571 435.11  6.9806 0.011294 *
Educ         1      1474 53911 428.24  1.2651 0.266659
Dens         1      1326 53762 428.07  1.1379 0.291788
NonWhite     1     33533 85969 455.77 28.7771 2.71e-06 ***
WhiteCollar  1      2934 55370 429.81  2.5178 0.119571
log(Pop)     1        405 52841 427.05  0.3472 0.558636
House        1      1688 54124 428.47  1.4486 0.235039
Income       1        337 52774 426.98  0.2895 0.593217
log(HC)      1      2468 54905 429.31  2.1182 0.152497
log(NOx)     1      6910 59346 433.90  5.9298 0.018914 *
log(SO2)     1        454 52891 427.11  0.3897 0.535590

```

The `~-RelHum` notation in `update()` means that both the response and the predictors are kept as they are, except for `RelHum` which is excluded. The updated p-values (note that they all changed with respect to the previous version!) are then analyzed and the next candidate for exclusion is identified. It turns out that `Income` is the one to drop next. The procedure then carries on; due to space constraints we do here without giving all the details and outputs. The sequence of excluded variables continues with `log(Pop)`, `log(SO2)`, `JulyTemp`, `Dens`, `log(HC)`, `House` and `WhiteCollar` and then stops, because after that all terms are significant.

```

> drop1(fit09, test="F")
<none>                61337 421.85
JanTemp      1      20169 81506 436.62 17.4277 0.0001116 ***
Rain         1     11576 72913 430.05 10.0021 0.0025875 **
Educ         1      7192 68529 426.39  6.2142 0.0158323 *
NonWhite     1     51349 112687 455.73 44.3698 1.583e-08 ***
log(NOx)     1     18848 80185 435.66 16.2860 0.0001760 ***

```

When comparing this output with the full model, we observe that the remaining predictors have lower p-values than initially. This is typical, the reason is that some of their predictive power was initially taken by the (at least partially collinear) predictors that were later dropped from the model. However, do not overestimate the importance of the remaining predictors. The final result is by no means the “correct” model in the sense of a causal relationship between response and predictors and often, if the data are just slightly different, the model we end with may be different. Furthermore, the removed variables may very well be related with the response. If simple linear regressions of the response on each of the excluded predictors were run, they might even show significant coefficients. In the multiple linear model however, there seem to be better, more informative predictors available. Another remark is that setting  $\alpha_{crit} = 0.05$  is totally arbitrary. If the main purpose of a model is prediction, often a 0.15 or 0.20 cutoff yields better results.

### 3.9.3 Forward Selection with p-Values

Rather than starting variable selection from the full model with all predictors, it is also possible to start with the null model that only contains the intercept. This is a viable and often-chosen alternative when a regression problem has too many predictors for the number of cases that are available – remember the rule of the thumb, that per estimated parameter a minimum of 5 observations should be present. The selection can be based on the p-value again. However, note that with forward selection, the process of adding variables is more laborious. When we start from the null model, we will fit  $p$  simple regressions, i.e. response vs. each of the predictors and finally add the one predictor that shows the lowest p-value in the individual hypothesis test for the slope. In R, we can simplify this process by employing the `add1()` function. As input, it takes with `f.null` the model we want to expand, as well as with `f.full` the maximum model we may want to use. Moreover, argument `test="F"` ensures that p-values for partial F-Tests are given.

```
> f.full <- lm(Mortality ~ JanTemp + JulyTemp + RelHum +
              Rain + Educ + Dens + NonWhite + WhiteCollar +
              log(Pop) + House + Income + log(HC) +
              log(NOx) + log(SO2), data=apm)
> add1(f.null, scope=f.full, test="F")
Single term additions
Model: Mortality ~ 1
```

	Df	Sum of Sq	RSS	AIC	F value	Pr(>F)	
<none>			225993	488.79			
JanTemp	1	58	225935	490.78	0.0145	0.9045520	
JulyTemp	1	23407	202586	484.34	6.5858	0.0129321	*
RelHum	1	2309	223684	490.19	0.5883	0.4462331	
Rain	1	42393	183599	478.54	13.1614	0.0006117	***
Educ	1	58340	167652	473.17	19.8352	3.991e-05	***
Dens	1	14365	211627	486.92	3.8691	0.0540559	.
NonWhite	1	94473	131520	458.85	40.9440	3.172e-08	***
WhiteCollar	1	18920	207072	485.63	5.2081	0.0262337	*
log(Pop)	1	1646	224347	490.36	0.4182	0.5204471	
House	1	30608	195385	482.21	8.9292	0.0041348	**
Income	1	18138	207855	485.86	4.9739	0.0296867	*
log(HC)	1	3553	222440	489.86	0.9103	0.3440525	
log(NOx)	1	17696	208296	485.98	4.8425	0.0318330	*
log(SO2)	1	34675	191318	480.97	10.3308	0.0021550	**

As it turns out, `NonWhite` is the predictor which has the lowest p-value in a simple regression when `Mortality` is the response variable. Hence it shall be added to the model. It is again convenient to use the `update()` function:

```
> fit01 <- update(f.null, .~.+NonWhite)
> add1(fit01, scope=f.full, test="F")
```

Next we try to enhance the model by adding a second predictor. All two-predictor-models with `NonWhite` and each of the remaining variables are tried and the

p-values from the partial F-Test (resp. individual hypothesis tests where appropriate) are recorded. This process can again be facilitated by function `add1()`. The next variable to add (data not shown) turns out to be `Educ`. Some further variables are added to the model, in sequence these are `log(SO2)`, `JanTemp`, `Rain` and `log(NOx)`. At this stage, there are no further predictors that would be significant if added to the model, hence the process stops – at least if we use  $\alpha_{crit} = 0.05$ . If the main purpose is prediction, using a 0.15 or 0.20 threshold may also be sensible here. If we compare to the result from the backward elimination, we notice that the two models are different. This is the usual case, i.e. the two solutions from these different search heuristics do not coincide. If we keep an eye on the summary of the final model, we notice:

```
> summary(fit06)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 1029.4608    87.8902  11.713 3.35e-16 ***
NonWhite      4.1458     0.6285   6.597 2.18e-08 ***
Educ        -15.5812     6.4970  -2.398 0.02010 *
log(SO2)     -0.1522     6.1914  -0.025 0.98048
JanTemp     -2.1484     0.6601  -3.255 0.00200 **
Rain         1.6554     0.5395   3.068 0.00341 **
log(NOx)     18.2513     7.7826   2.345 0.02287 *
---
Residual standard error: 34.34 on 52 degrees of freedom
Multiple R-squared: 0.7286, Adjusted R-squared: 0.6973
F-statistic: 23.27 on 6 and 52 DF, p-value: 3.879e-13
```

Thus, not all predictors in the final model are significant. The last included variable `log(NOx)` is, but not `log(SO2)` which was added earlier in the process. It might seem tempting to complement the forward search with a backward elimination that is run subsequently. A similar strategy incorporating this will be presented below when we introduce `step()`, the most used variable selection procedure in R.

### 3.9.4 AIC/BIC

So far, our variable selection approaches were based on evaluating p-values of individual hypothesis tests or partial F-Tests. This is intuitive to the practitioner, but from a theoretical, mathematical perspective suffers from some drawbacks. Hence using other criteria that are based in information theory has nowadays become the established standard for model selection. The most popular variant is the Akaike Information Criterion (AIC, 1974), which gauges goodness-of-fit to the data with the complexity of the model and hence pursues a similar idea as the adjusted  $R^2$ .

$$\begin{aligned} AIC &= -2\log(L) + 2q \\ &= c + n\log(RSS/n) + 2q \end{aligned}$$

In the above formula,  $L$  is the value of the likelihood function for a particular model and  $q$  is the number of parameters that were estimated in it. When assuming



Gaussian errors and using the OLS estimator, the Likelihood function is driven by  $RSS$ , the residual sum of squares. Hence, the AIC criterion compares the magnitude of the residuals with the complexity of the model that was used and so prevents overfitting. While a larger models has the advantage of achieving a lower  $RSS$ , its penalization will be harder. As long as the data and the response variable are identical, any two models can be compared by AIC. In contrast to the testing based approaches, AIC does not require them to be hierarchical. Obviously, the smaller AIC is, the better the model. Please note that it is a relative measure, i.e. useful for comparing models on the same data, but the AIC value does not tell about the quality of the model in an absolute sense. An alternative to the AIC consists of the very similar Bayesian Information Criterion (BIC) that was developed by Schwarz (1978) who gave a Bayesian argument for it. The basic idea behind is absolutely identical, the only difference is in the penalty term:

$$\begin{aligned} BIC &= -2\log(L) + 2\log n \\ &= c + n\log(RSS/n) + 2\log n \end{aligned}$$

For any reasonably sized dataset with more than  $n=7$  observations, we have  $\log n > 2$  and hence there is a stronger penalization for model size in BIC, meaning that the models will generally be smaller with this criterion. There has been some debate and theoretically motivated comparisons among AIC and BIC and it seems as if AIC was asymptotically optimal in selecting the model with least mean squared error at an optimal convergence rate, while BIC is lacking these properties. For the practitioner, these considerations are of relatively little value. If the main purpose of a regression model is prediction, one usually profits from using bigger models, hence it is attractive to use AIC for variable selection. In contrast, the principal aim is inference or model interpretation, one often uses the BIC criterion since the typically smaller models are handier for this purpose. Now the question is how AIC/BIC variable selection can be operationalized. As we can see from the outputs of previous chapters, the R functions `drop1()` and `add1()` both report AIC values. In fact, we could use backward elimination and forward selection very much in the same way as above, by just replacing the p-value as the selection criterion with AIC/BIC. The next section will further comment on this and at the same time suggest an improvement to the search heuristics.

### 3.9.5 Using R Function `step()`

As mentioned above, we can perform both backward elimination or forward selection based on AIC/BIC by using the R functions `drop1()` and `add1()`. Using the `update()` function, we could remove resp. add the variable that leads to the most improved AIC/BIC and would stop if the criterion could not be further improved anymore. However, there is no need for this relatively tedious handwork since there is R function `step()` which can do the entire selection automatically. The syntax for starting with the full model and doing an AIC based backward elimination is very simple. Please note that for using the BIC criterion, we need to alter argument `k` to `k=log(59)` here, or `k=log(nrow(data))` more generally.

```
> f.back <- step(f.full, direction="backward", k=2)
> summary(f.back)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 1031.9491    80.2930  12.852 < 2e-16 ***
JanTemp      -2.0235     0.5145  -3.933 0.00025 ***
Rain         1.8117     0.5305   3.415 0.00125 **
Educ        -10.7463     7.0797  -1.518 0.13510
NonWhite     4.0401     0.6216   6.500 3.1e-08 ***
WhiteCollar -1.4514     1.0451  -1.389 0.17082
log(NOx)     19.2481     4.5220   4.257 8.7e-05 ***
---
Residual standard error: 33.72 on 52 degrees of freedom
Multiple R-squared: 0.7383, Adjusted R-squared: 0.7081
F-statistic: 24.45 on 6 and 52 DF, p-value: 1.543e-13
```

The arguments to `step()` are the model to start with, then the direction of the search and finally the penalty parameter for the information criterion. Here, we set `k=2`, which means that AIC is used. Please note that the (long!) output detailing the selection process was omitted here due to space constraints. Since AIC is used, the predictors in the final model do not need to be significant, which turns out to be the case for `Educ` and `WhiteCollar`. If we prefer a forward selection, we need to start with the null model and define the scope for the search:

```
> sc      <- list(lower=f.null, upper=f.full)
> f.forw <- step(f.null, scope=sc, direction="forward", k=2)
> summary(f.forw)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 1258.3217    190.7542   6.597 2.57e-08 ***
NonWhite     4.6907     0.7892   5.943 2.68e-07 ***
Educ        -13.9775     7.7488  -1.804 0.07728 .
log(SO2)     -2.6904     6.5506  -0.411 0.68304
JanTemp     -2.6780     0.8341  -3.211 0.00232 **
Rain         1.7329     0.5438   3.187 0.00248 **
log(NOx)     20.4077     7.7626   2.629 0.01135 *
WhiteCollar -1.5733     1.0534  -1.494 0.14159
House       -49.8414    37.0169  -1.346 0.18423
---
Residual standard error: 33.78 on 50 degrees of freedom
Multiple R-squared: 0.7475, Adjusted R-squared: 0.7071
F-statistic: 18.5 on 8 and 50 DF, p-value: 1.623e-12
```

The `scope` argument in `step()` requires a list where both the minimal and maximal model are specified. We observe that the result from the forward selection is different to the one from backward elimination. Also, there is a difference in contrast to the forward selection model that was obtained with the p-value as a criterion. Please note that `step()` does not offer the option to perform variable selection using p-values, but only has the information criteria implemented. This is the better option in practice – so use these!

In the above forward selection model (as well as in the one with the p-value as a criterion), we have the issue that  $\log(\text{SO}_2)$  was added at an early stage. Later on, also  $\log(\text{NO}_x)$  became part of the model, so that  $\log(\text{SO}_2)$  became non-significant and most likely also obsolete. However, in this pure one-directional search, there is no means for correcting earlier decisions, i.e. re-excluding variables that were added at a previous stage of the search. This is not mandatory; during the selection process we can well consider both adding new and removing existing variables at the same time. This makes the process more flexible, i.e. more models will be evaluated and better final solutions result. All decision will be on the basis of an information criterion. The setting for the search direction in `step()` is `direction="both"`. This is also the default for that argument, hence if nothing is explicitly specified, that will be it. The variable selection process can start from the null model, from the full model or actually also from any arbitrary model in between. With the `scope` argument, the maximal and minimal models in the search process can and need to be defined. We illustrate with the following example:

```
> sc      <- list(lower=f.null, upper=f.full)
> f.both <- step(f.null, scope=sc, direction="both", k=2)
> summary(f.both)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  944.274572   47.170880  20.018 < 2e-16 ***
NonWhite      4.194072    0.620038   6.764 1.18e-08 ***
JanTemp     -1.941998    0.516119  -3.763 0.000428 ***
Rain         1.924283    0.484198   3.974 0.000219 ***
log(NOx)     17.000178    4.879795   3.484 0.001012 **
WhiteCollar  -2.723255    0.947170  -2.875 0.005839 **
Dens         0.006437    0.003608   1.784 0.080282 .
---
Residual standard error: 33.46 on 52 degrees of freedom
Multiple R-squared:  0.7425, Adjusted R-squared:  0.7127
F-statistic: 24.98 on 6 and 52 DF, p-value: 1.028e-13
```

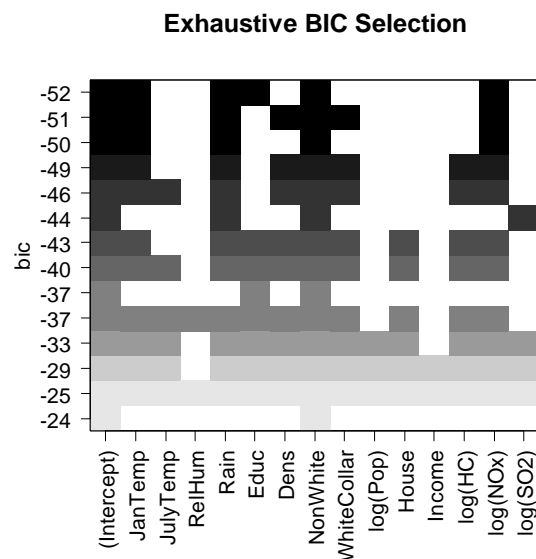
When starting from the empty model, this improved strategy with searches in both directions first adds the six predictors `NonWhite`, `Educ`, `log(SO2)`, `JanTemp`, `Rain` and `log(NOx)`. Then, after `log(NOx)` has been added, `log(SO2)` indeed becomes obsolete and is dropped from the model again. Next, `WhiteCollar` and `Dens` are added, before finally `Educ` is dropped. At this stage, no further AIC improvement is possible by either adding or removing terms from the model, hence the search has stopped. The output and final summary can be seen above.

We conclude this section with the remark that the most established standard in variable selection consists of fitting the full model with all predictors (if that is possible) and then running an AIC-based search with `direction="both"`. Due to the respective default settings in R, it suffices to simply type `step(f.full)` to run mentioned standard procedure – very quick and easy thus. However, depending on the problem, alternative settings and procedures may very well be valuable.

### 3.9.6 All Subsets Selection

If a multiple regression model has  $p$  predictors, there are actually  $2^p$  models which can be fitted. For each of the variables there are two options, namely including it or not. This shows that even with a mid-sized dataset that features 50 predictors, very many models do exist. If  $p > 10$ , the stepwise approaches from above only consider few of all existing models and it is very well conceivable that the variable selection process stops at a point where the AIC/BIC criteria reach a local, but not the global optimum. For ensuring that the one model with globally minimal AIC/BIC is found, we need to run an exhaustive search over all possible models. This is only feasible with small datasets with up to about  $p \approx 15-20$  predictors; with very fast computers perhaps also a bit more, but the complexity of the problem increases so quickly that there is no hope for big datasets. In R, there is `library(leaps)` which has function `regsubsets()` that does such an exhaustive search. Unfortunately, it cannot correctly handle factor variables or interaction terms, hence its use is limited to datasets that consist of numerical predictors only. This is the case for the mortality datasets, hence we compute:

```
> library(leaps)
> out <- regsubsets(formula(f.full), nbest=1, dat=apm, nvmax=14)
> plot(out)
```



The arguments in the `regsubsets()` function are set such that for each size, only the best model is returned, as `nbest=1`. The maximal model size is `nvmax=14`, equal to the number of predictors we have, as this is computationally feasible here. Note that with bigger datasets, you may need to limit yourself here. The function then computes the BIC value for every model and yields a graphical output. The minimal BIC value is attained with the 5 predictors `JanTemp`, `Rain`, `Educ`, `NonWhite` and `log(NOx)`. The closest competitor is the model with 6 predictors, where `Educ` is replaced by `Dens` and `WhiteCollar`. In contrast, models with very few or very many predictors seem to perform poorer.

### 3.9.7 Final Remarks about Variable Selection

In variable selection, it is important to treat models with factor variables, polynomial or interaction terms correctly. In particular, the rules are:

- If the coefficient of a dummy variable that belongs to a factor is non-significant, that dummy cannot simply be removed from the model. Either we keep the entire factor variable, or we keep it with all its dummies. If p-values are used as a selection criterion, partial F-Tests have to be run.
- If there are interaction terms between two variables that stay within the model, then the main effects of the respective variables cannot be removed.
- The same holds for the case where significant higher-order polynomial terms of a variable are present. One cannot remove the lower-order terms.

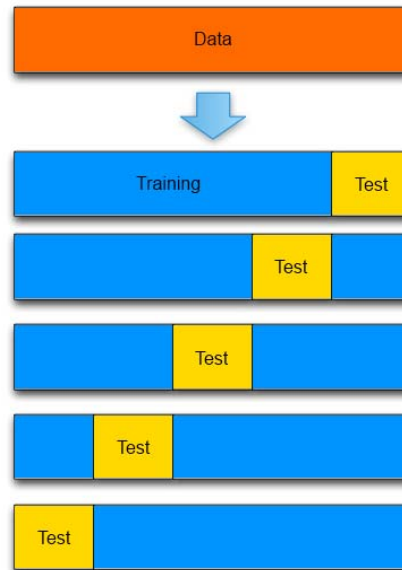
So please be careful if variable selection is performed manually. In contrast, the R functions `step()`, `drop1()` and `add1()` obey to these rules and will only consider exclusion of terms that can actually be dropped from the model. As already mentioned, `regsubsets()` in contrast does not have that feature and may invariably come to false decisions if not only plain numerical predictors are present in a dataset. This strongly limits the practical utility of that function.

A final remark on variable selection: every procedure may yield a different “best” model. Often, even the results with a fixed procedure are somewhat instable, i.e. if we received another sample from the same population and repeated the variable selection procedure, we might identify another “best” model. Thus, there is an element of chance in this declaration. How can we mitigate this in practice? It is usually advisable to not only consider the “best” model according to a particular procedure, but to check a few more models that did nearly as good, if they exist.

## 3.10 Cross Validation

Cross validation is a model evaluation technique that tells how well the results will generalize to an independent dataset from the same population. It is mainly used if the primary goal in a regression analysis is prediction, but can also be useful when other aims are pursued. By construction, it does not artificially advantage bigger models with more predictors and hence goes in line with approaches such as the adjusted  $R^2$  and AIC/BIC. On the other hand, it stands in sharp contrast to criteria like  $RSS$ , multiple  $R^2$  or the variance of the error term which would all leave to overfitting when used for comparison of model of different size.

The basic idea of cross validation is relatively simple. It consists of splitting the data into a learning set (e.g. 80% of the observations) which is used for fitting the model, and into a test set (e.g. 20% of the observations) on which the quality of the predictions is evaluated. This process is repeated a number of (e.g. 5) times, until every observation in the dataset has been predicted exactly once.



In the schematic example above, so-called 5-fold cross validation is illustrated. Each of the data points that are available will be used four times for training and one time for prediction. This will result in a  $\hat{y}_i^{Test}$  for all data points  $i = 1, \dots, n$ . Since the observed value  $y_i$  is known, we can compute the squared prediction error  $(y_i - \hat{y}_i^{Test})^2$  for all observations and finally determine the mean squared prediction error:

$$MSPE_{CV} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i^{Test})^2$$

This is used as a quality criterion for the model. As we are using out-of-sample predictions here, there is no need to penalize bigger models. If there are too many variables used in a regression and overfitting happens, spurious random phenomena will be explained which will degrade the quality of the out-of-sample results.

Cross validation is a bit more laborious and also computationally more intensive than AIC-based variable selection. Its advantages are that it is a very flexible procedure that has little restrictions. So for example, it can also be applied in the following settings, when an AIC-based comparison is not feasible:

- If the effect of a response variable transformation needs to be evaluated. The AIC values from models with and without transformation cannot be compared. When using cross validation, we can always predict on the original scale and obtain a sound comparison of results.
- If the sample is not identical. In some cases, one may wish to check whether excluding data points (i.e. outliers, leverage points, influential data points) from the fitting process yields better predictions for the entire sample or not. This is possible with cross validation.

- If the performance of alternative methods such as Lasso, Ridge Regression, Generalized Additive Models or Robust Regression shall be evaluated. In these cases, testing based model comparison does not work and AIC approaches are at least difficult (if not impossible).

So there may be good reasons to choose cross validation. Additionally, when prediction is the main aim of the analysis, the mean squared prediction error is the most relevant quality criterion for a model, plus it allows for obtaining an idea of the absolute quality of the results. While R has the `CVlm()` function in `library(DAAG)`, this is a poor implementation and the author strongly recommends to self-program a cross validation loop. This requires some basic knowledge about programming. See here for some generic code for a 5-fold cross validation.

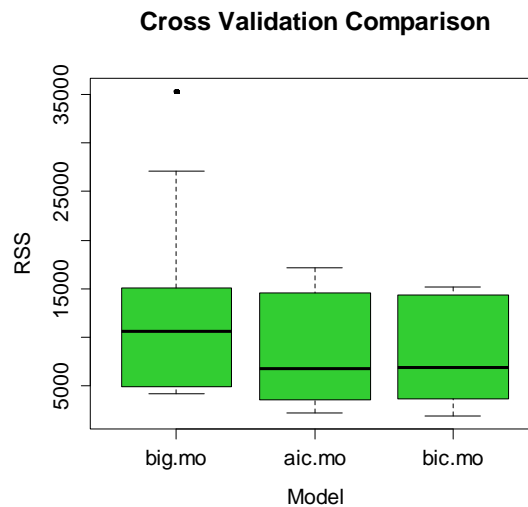
```
> rss <- c()
> folds <- 5
> sb <- round(seq(0,nrow(dat),length=(fo+1)))
> for (i in 1:folds)
> {
>   test <- (sb[((folds+1)-i)]+1):(sb[((folds+2)-i)])
>   train <- (1:nrow(dat))[-test]
>   fit <- lm(res ~ p1+..., data=dat[train,])
>   pred <- predict(fit, newdata=dat[test,])
>   rss[i] <- sum((dat$response[test] - pred)^2)
> }
```

As an example, we apply cross validation for evaluating the out-of-sample performance in mortality prediction. We will use the full model with 14 predictors, as well as the two models obtained from `step()` when starting from the full model with AIC and BIC, which have 6 resp. 5 predictors.

```
> rss <- data.frame(rss1, rss2, rss3)
> apply(rss,2,mean)
   big.mo   aic.mo   bic.mo
13213.182  8774.873  8329.638
```

The numerical results indicate that the BIC model performs best, with the AIC model close behind. The full model with all predictors is clearly worse, indicating that it suffers from overfitting. Generating boxplots of the squared prediction errors can further enhance the comprehension of the results. The results are displayed on the next page and show again, that the individual predictions with the full model are generally worse and also have bigger scatter than the ones from the models where variable selection had been performed. Please note that the distribution of the out-of-sample errors can often yield additional insight into the model and its deficiencies. For example, if there are only a few data points with poor out-of-sample predictions that are not standing out in the insample residual analysis, this well needs to ring the alarm bells. It usually indicates an overfitting phenomenon, or more deliberately said a phenomenon that does not generalize to an independent dataset.

```
> boxplot(rss)
```



### 3.11 Modelling Strategies

This is the concluding section about multiple linear regression modelling. We have learnt a number of techniques for dealing with the data. The often asked question by students is in which order the tools need to be applied in practice. Please note that the sequence with which they were presented in the script is not necessarily the correct order, as the presentation here is also motivated by didactic reasons. A good generic solution, but not the ultimate, always-optimal strategy is:

Data Preparation → Variable Transformations → Estimation of Coefficients  
 → Model Diagnostics → Variable Refinement and Selection → Evaluation  
 → Inference → Reporting

If some flaws to the analysis are noticed in any of the above steps, this may well send you back to the start again. On a more general note, professional regression analysis can be seen as the search for structure in the data. This requires technical skill, flexibility and intuition. One must be alert to the obvious as well as to the non-obvious, and needs the flair for the unexpected. Trying to follow some standard recipes usually does not work. The tries for defining automatized regression procedures where one can just hit the bottom are old, but did (and will) not make the breakthrough. You can do better! As a guideline, we here try to give some hints what to think about in the single analysis steps. Please note that these hints are by no means complete!

#### Data Screening and Processing

Learn the meaning of all variables in your dataset and give them short and informative names. Check all variables for missing values, errors and impossible values. Especially dangerous are missing values that are coded numerically. If in



doubt, be deliberate with setting these to `NA`, as it is generally better to have missing rather than wrong data. If missing values are present, ask yourself whether these are random or systematic errors. In the latter case, this may very well limit the meaning of your results, whereas random missings are usually not a bigger problem.

### Variable Transformations

First of all, bring all variables to a suitable scale that you are well familiar with. It makes the results a lot easier to interpret. Please note that using linear transformation will not change the regression analysis. Furthermore, use statistical and specific knowledge for identifying variables that require log-transformations. Anything that is clearly on a relative scale should be transformed at this point. Finally, breaking very obvious collinearities can already happen at this point.

### Fitting a First Model

We usually start by fitting a big model with potentially too many predictors. If the number of data points allows, use all predictors for the first model. The rule of the thumb is that one should roughly have five times as many observations as the number of coefficients that are estimated. If that is clearly violated, one can potentially sort out some predictors manually by previous knowledge. Or alternatively, perform variable selection from the null model with either AIC or a p-value of 0.2.

### Model Diagnostics

Always inspect the 4 standard residuals plots in R. A systematic error in the Tukey-Anscombe plot indicates that the model will generate false predictions and is never tolerable. Improve the model by using transformations, adding interaction terms, creating/obtaining new variables or applying more sophisticated methods such as Generalized Additive Models. Be aware that there may be influential data points. If they exist, try to understand them. Take care with non-constant variance and long-tailed errors. They often do not have catastrophic influence, but compromise the quality of inference results and the levels of the confidence intervals. Also think about potential correlation among the residuals, especially if the data have spatial or temporal structure. If it exists, this will degrade the quality of the inference results, too.

### Variable Selection

Try to reduce the model to the predictors that are utterly required and drop the rest. The standard procedure is to use the `step()` function with search direction “both”, starting from the full model and either the AIC or BIC criterion. If it is computationally feasible, an all-subset-search with AIC/BIC is even better. While doing variable selection, keep the quality of the models in mind. The residual plots must not degrade (substantially) when variables are excluded. If they do, rather keep a few more predictors than AIC/BIC suggest!

## Refining the Model

For understanding the role of each predictor, partial residual plots may help. Inspect for potential non-linearities and if they exist, either convert them to factor variables or use a more flexible tool such as Generalized Additive Models. Sometimes, adding interaction terms may improve the fit drastically; partial residual plots often allow gaining hints where they are required. Use the respective tools to find out whether there is multicollinearity that disturbs and if yes, take the respective actions.

## Plausibility

Now you are at a point where a technically valid model was found. If you have the respective knowledge of the application field, check the regression summary for implausible predictors, wrong signs or generally things that contradict established theory and try to find out how/why it appeared. Sometimes, it may also be justified to remove such terms from the model if there is no drastic change to the outcome, even if AIC/BIC suggest otherwise.

## Evaluation

By using cross validation, one can get another view on the performance of one or several competing models and gain an idea on the precision of out-of-sample predictions. Finally, if the decision for one particular model has been made and all the necessary assumptions are met, one can derive test results, confidence and prediction intervals.

## Reporting

Whenever results from a statistical analysis are reported, it is key to be honest and openly report all data manipulations and decisions that were made. A regression analysis is always an interpretation of the raw data material that was available. Finally, keep in mind that regression models are always descriptive only, but not causal! And do not confuse significance of terms with relevance. The next section gives some further details about this.

## 3.12 Significance vs. Relevance

It is important to keep in mind that p-values arise from a combination of *predictor effect*, *scatter* and *sample size*. When the first two remain constant, smaller and smaller p-values will be achieved by increasing the sample size. In fact, if a predictor in a regression model does not have zero influence (i.e.  $\beta_j = 0$ ) which is almost never the case in practice, we just need enough data to be able to reject the aforementioned null hypothesis. Hence with large datasets, we may very well see statistically significant predictors which are largely irrelevant from a practical viewpoint. Or in other words: do not confuse small p-values with a relevant or important predictor effect.

On the other hand, a  $\beta_j = 0$  null hypothesis that is accepted is even less meaningful. Very generally, absence of evidence is not the same as evidence of absence. Hence, failing to reject a null hypothesis of  $\beta_j = 0$  is not a proof that a predictor does not influence the response. It may very well be the case that there is just too much scatter for the present sample size and a larger dataset would yield a different answer. Or it can also happen that there is another (collinear) predictor in the multiple regression model that takes load and significance off said variable. A slightly different but related idea is to measure the relevance of the predictors in a regression model. This complements the impression that is given by their significance. A very simple statistic to do so is to compute the maximum effect that a predictor variable has on the response on the given dataset.

$$\text{Maxrel}_j = \hat{\beta}_j (\max_i x_{ij} - \min_i x_{ij})$$

So we take the span between the maximal and minimal value for the respective predictor variable and multiply it with the estimated coefficient. We illustrate with the computation of that measure in the mortality model obtained from the AIC backward elimination. Note that the code is somewhat laborious since we have to accommodate for the log-transformed variables.

```
> mort      <- apm[, -c(10,13,14,15)]
> mort$log.Pop <- log(apm$Pop)
> mort$log.HC  <- log(apm$HC)
> mort$log.NOx <- log(apm$NOx)
> mort$log.SO2 <- log(apm$SO2)
> f.full      <- lm(Mortality ~ ., data=mort)
> f.aic       <- stepAIC(f.full)
> summary(f.aic)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 1031.9491    80.2930  12.852 < 2e-16 ***
JanTemp     -2.0235     0.5145  -3.933 0.00025 ***
Rain         1.8117     0.5305   3.415 0.00125 **
Educ        -10.7463    7.0797  -1.518 0.13510
NonWhite     4.0401     0.6216   6.500 3.1e-08 ***
WhiteCollar -1.4514     1.0451  -1.389 0.17082
log.NOx      19.2481    4.5220   4.257 8.7e-05 ***
> mami <- function(col) max(col)-min(col)
> ranges <- apply(mort, 2, mami)[c(2,5,6,8,9,14)]
> ranges
      JanTemp      Rain      Educ      NonWhite
55.000000  55.000000  3.300000  37.700000
WhiteCollar  log.NOx
28.400000   5.765191
> rele <- abs(ranges*coef(f.aic)[-1])
> rele
      JanTemp      Rain      Educ      NonWhite
111.29085   99.64437  35.46263  152.31188
WhiteCollar  log.NOx
41.21900   110.96873
```

So for the January temperature, the difference between the observation with the smallest and largest values is 55 units. The regression coefficient is estimated as -2.0235, so the maximum effect this variable has on mortality is 111.29085 units. With the above code, these respective quantities are computed for all predictors that are in the final model. As we can see, `NonWhite` has the biggest impact on the response according to this statistic, whereas `Educ` is comparably less relevant. Please note that these relevance values are not strongly correlated with the p-values. It can very well happen that a very homogeneously appearing effect is small and thus has a low p-value, but little influence/relevance to the response. Finally, we can also compare the above results to the span between the maximum and minimum values for the response which is 322.43.

The advantage of using the maximum effect as a relevance measure is its simplicity and intuitivity. However, it is an extreme value statistic which bears its dangers. The computed relevance basically depends on two observations only which makes it not very robust to outliers. Hence there have been numerous tries to define alternative relevance measures that have nicer theoretical properties. We will introduce two of them here. They, as well as several more, are implemented in R package `library(relaimpo)`. We first focus on the use of standardized coefficients. This is an approach where all predictor variables are centered and rescaled so that they have mean zero and unit variance. This makes their coefficients  $\hat{\beta}_j$  directly comparable, i.e. the magnitude of the coefficient then tells us about the effect size and thus relevance.

```
> library(relaimpo)
> calc.relimp(fit.or, type="betasq", rela=TRUE)
Total response variance: 3896.423
Proportion of variance explained by model: 73.83%
Metrics are normalized to sum to 100% (rela=TRUE).
      betasq
JanTemp    0.14838879   NonWhite    0.46467477
Rain       0.15460185   WhiteCollar 0.01903859
Educ       0.02938728   logNOx     0.18390873
```

The results are somewhat similar to the ones from before, i.e. it is again `NonWhite` which is the most relevant predictor and `Educ` is the least relevant. However, also this approach with standardized coefficients has some undesired properties. A more theoretically sound variant is to use the LMG criterion. It is pretty complicated, so we do here without giving the details of its computation:

```
> library(relaimpo)
> calc.relimp(fit.or, type="lmg", rela=TRUE)
Total response variance: 3896.423
Proportion of variance explained by model: 73.83%
Metrics are normalized to sum to 100% (rela=TRUE).
      betasq
JanTemp    0.06027426   NonWhite    0.42530033
Rain       0.16412106   WhiteCollar 0.05357159
Educ       0.15815923   logNOx     0.13857353
```

We conclude this chapter on multiple linear regression with a remark about what a good model is. The *true model* is a concept that exists in theory and simulation, but whether it does in practice remains often unclear. Anyway, it is not realistic to identify the true model in observational studies with a limited amount of data. A *good model* is one that is useful for the task at hand, correctly describes the data without any systematical errors, has good predictive power and is practical and applicable for future use. Finally, we emphasize again that regression models that are found in observational studies are always only descriptive but never causal. A good model may yield an accurate idea which of the observed variables drive the variation in the response, but not necessarily reveal the true mechanism that is behind.