HDS 0. Linear model

Matti Pirinen, University of Helsinki

29.10.2019

Suppose that we want to model how the value of a numerical outcome variable y could be predicted from p predictors $\boldsymbol{x} = (x_1, \ldots, x_p)$. For example, y could be the price of an apartement and x_1 could be the district within Helsinki and x_2 could be the number of bedrooms. Thus, we want to find such a function f that our prediction $\widehat{y} = f(\boldsymbol{x})$ is a good approximation to y. We call such a function f a **regression function** and we talk about "regressing y on $\boldsymbol{x}$" when we estimate the regression function. A common approach in statistical modeling is to aim for using the conditional expectation as the regression function: $f(\boldsymbol{x}) = \mathrm{E}(Y \,|\, \boldsymbol{X} = \boldsymbol{x})$. A theoretical justification is that this function has the smallest mean squared error among all possible regression functions. This means that the conditional expectation is the function f that solves the minimization problem

$$\min_{f(\boldsymbol{x})} \mathrm{E}(Y - f(\boldsymbol{x}))^2,$$

for any given $\boldsymbol{x}$, where the expectation is over random variable Y for fixed $\boldsymbol{x}$.

The simplest of regression models is the linear regression model that is our basic building block on this course.

1. Linear model gives a baseline to which compare more complex models: if your complex model does not do better than the linear model, most likely you want to use the linear model because of simplicity!
2. Linear model has very good interpretability.
3. Weighted and correlated linear regression, generalized linear models (e.g. logistic regresison) and high-dimensional regression (e.g. LASSO, ridge regression) are built on top of the linear model.
4. Linear model is a great vehicle to demonstrate statistical concepts (e.g. uncertainty, model fit, predictions, inference).

To refresh the linear model,

- read an excellent chapter "**3. Linear Regression**" from ISLR.

Below is a brief summary of linear model including key formulas and R commands.

- If you have not applied linear regression much *in practice* using **R**, go through additional notes for Practical linear regression.

Definition

Suppose outcome y and p other variables (or predictors) $\boldsymbol{x} = (x_1, \ldots, x_p)$ have been measured on n individuals: data for individual i is $(y_i, x_{i1}, \ldots, x_{ip})$. We want to study relationship between y and x's. The standard linear model assumes that there is a relationship

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i,$$

where ε_i is an **error term** that captures the difference between the outcome and the **linear predictor**: $\varepsilon_i = y_i - \left(\beta_0 + \sum_j \beta_{ij} x_{ij}\right)$. We can turn this linear model formulation into a useful statistical model by assigning some distributional assumptions on the error terms ε_i. We always assume that $\mathrm{E}(\varepsilon_i) = 0$. One typical assumption is **homoscedasticity** i.e, that the error terms have a constant variance $\mathrm{Var}(\varepsilon_i) = \sigma^2$. Another is that the error terms for observations i and h are uncorrelated with each other, $\mathrm{E}(\varepsilon_i \varepsilon_h) = 0$, and that the error terms are independent of predictors $\boldsymbol{x}$. Often the model is further restricted to the case where the error terms are assumed to follow a **Normal distribution**, also called the **Gaussian distribution**.

Note, however, that there are also widely-used linear models that have correlated errors, possibly with heteroscedasticity (varying error variance) and/or linear models where the errors are not Gaussian (i.e. are not distributed according to a Normal distribution).

With the assumption $\mathrm{E}(\varepsilon_i) = 0$, we see that the conditional expectation, which is our natural candidate for the regression function, takes the form of the linear predictor:

$$\mathrm{E}(Y \mid X = \boldsymbol{x}) = \mathrm{E}\left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i \,\middle|\, X = \boldsymbol{x} \right) = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij},$$

and hence our goal is to estimate the unknown parameters $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_p)^T$ of this function.

Terminology. It is conceptually important to distinguish the model parameters $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_p)^T$ from their estimates $\widehat{\boldsymbol{\beta}} = \left(\widehat{\beta_0}, \ldots, \widehat{\beta_p}\right)^T$ as well as to separate the error terms ε_i from their estimates that are called **residuals** $\widehat{\varepsilon_i} = y_i - \left(\widehat{\beta_0} + \sum_{j=1}^{p} \widehat{\beta_j} x_{ij}\right)$. The parameters and error terms are unknown quantities that we will not know exactly, but we can estimate them, with varying levels of precision, using parameter estimates and residuals.

When we have parameter estimates available, we can make a (linear model) **prediction** of outcome value y_u of an unobserved individual u as

$$\widehat{y_u} = \widehat{\beta_0} + \sum_{j=1}^{p} \widehat{\beta_j} x_{uj}.$$

With this notation, the residual can be written as $\widehat{\varepsilon_i} = y_i - \widehat{y_i}$.

Example 0.1

Let's generate some data from a linear model with $p = 1$. We want that our single predictor x_1 explains proportion ϕ of the total variance of outcome y (whereas the remaining proportion $1 - \phi$ is left for the error terms to account for). In data generation, we can first choose whichever variances for the predictor x_1 and error ε, and then determine β_1 based on those variances and the value of ϕ. For simplicity, let's use

$$\mathrm{Var}(x_1) = \mathrm{Var}(\varepsilon) = 1.$$

This leads to following formulas

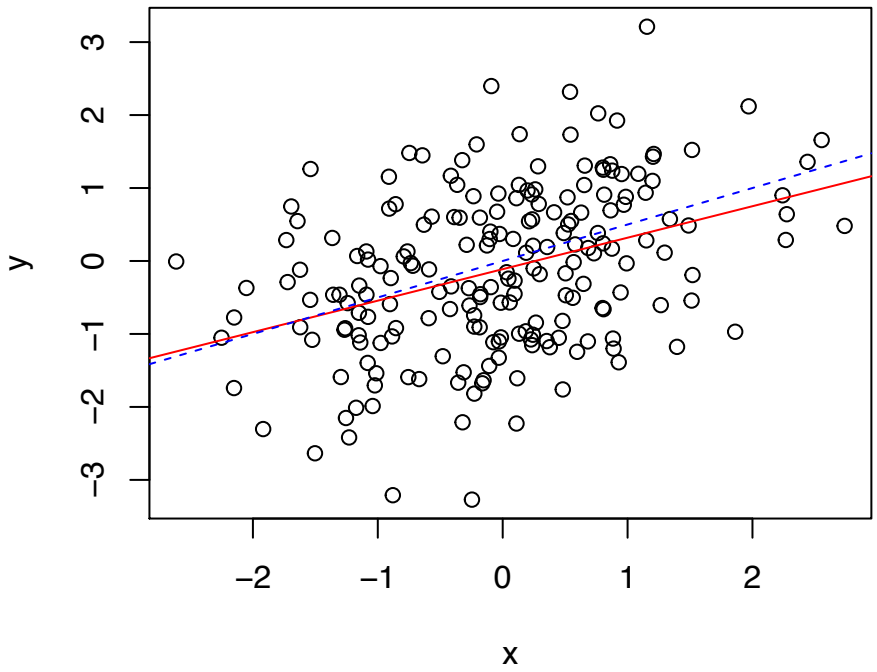
$$\begin{aligned} \mathrm{Var}(x_1\beta_1) &= \beta_1^2 \mathrm{Var}(x_1) = \beta_1^2 \\ \mathrm{Var}(y) &= \mathrm{Var}(x_1\beta_1) + \mathrm{Var}(\varepsilon) = \beta_1^2 + 1 \\ \phi &= \mathrm{Var}(x_1\beta_1)/\mathrm{Var}(y) = \beta_1^2/(\beta_1^2 + 1) \\ \beta_1 &= \pm\sqrt{\phi/(1-\phi)} \end{aligned}$$

where the first uses $\mathrm{Var}(aX) = a^2\mathrm{Var}(X)$ for a constant a and random variable X, the second uses the fact that x_1 and ε are independent and hence their covariance term is zero, and the fourth solves β_1 from the second order polynomial of formula 3.

```
set.seed(3102017)
n = 200 # individuals (or samples) to be simulated
phi = 0.2 # variance explained by x_1, should be 0 < phi < 1.
b.0 = 0 # set intercept to 0, could equally well be any other value.
b.1 = sqrt(phi/(1-phi)) # could equally well have minus sign.
x = rnorm(n, 0, 1) # use Normal distribution, could be any other as long as var=1
eps = rnorm(n, 0, 1) # should be Normal in order that the standard tests are valid (or should it? see E
y = b.0 + x*b.1 + eps #use linear model formula to generate outcome y.
# check variance explained by x:
var(x*b.1) / var(y) # should be close to phi.
```

```
## [1] 0.1983043
```

```
plot(x, y)
lm.fit = lm( y ~ x ) # Fit a linear model by least squares method.
abline(lm.fit, col = "red") # Add estimated regression line in red.
abline( a = b.0, b = b.1, col = "blue", lty = 2) # Add true population regression line in blue.
```



Where are the error terms and where are the residuals in the figure above?

Exercise. Above we derived β_1 to yield a given variance explained. Sometimes we might want to derive β_1 to yield a particular correlation between x_1 and y. Show that by choosing $\beta_1 = \text{sign}(r)\sqrt{r^2/(1-r^2)}$, we have that $\text{cor}(x_1, y) = r$ when $\text{Var}(x_1) = \text{Var}(\varepsilon) = 1$.

Let's see the basic output of the linear model in R using `summary()`.

```
summary(lm.fit) # Show output from lm().
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.05209 -0.75674  0.00068  0.72571  2.82434
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.11214    0.07310  -1.534    0.127
## x            0.43150    0.07367   5.858 1.93e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.033 on 198 degrees of freedom
## Multiple R-squared:  0.1477, Adjusted R-squared:  0.1434
## F-statistic: 34.31 on 1 and 198 DF,  p-value: 1.931e-08
```

Estimates

Let's remind ourselves what is in this output and how it is computed. Linear model is typically fit by the least squares criterion. That is, we look for the parameter vector estimate $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \ldots, \widehat{\beta}_p)^T$ that minimizes the **residual sum of squares** (RSS):

$$RSS = (\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}})^T(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}) = \sum_i \left(y_i - \widehat{\beta}_0 - \sum_j x_{ij}\widehat{\beta}_j \right)^2$$

where $\boldsymbol{X}$ is $n \times (p+1)$ matrix whose first column contains ones to represent the constant **intercept** (or baseline) value β_0. Using rules for derivatives for matrix calculations

$$\frac{\partial RSS}{\partial \widehat{\boldsymbol{\beta}}} = -2\boldsymbol{X}^T\boldsymbol{y} - 2(\boldsymbol{X}^T\boldsymbol{X})\widehat{\boldsymbol{\beta}},$$

that has the zero at the Least Squares solution

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y},$$

assuming that $\boldsymbol{X}^T\boldsymbol{X}$ is of full rank. By treating $\boldsymbol{X}$ as a fixed matrix and assuming that the error term $\boldsymbol{\varepsilon}$ has mean 0, we have that the least squares estimator (LSE) is unbiased:

$$\mathrm{E}\left(\widehat{\boldsymbol{\beta}} \,|\, \boldsymbol{\beta}\right) = \mathrm{E}\left(\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\left(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}\right) \,|\, \boldsymbol{\beta}\right) = \boldsymbol{\beta} + \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\mathrm{E}(\boldsymbol{\varepsilon}) = \boldsymbol{\beta},$$

When we consider a linear model in which error terms are uncorrelated and have the same variance σ^2, i.e, the variance matrix of $\boldsymbol{\varepsilon}$ is $\mathrm{Var}(\boldsymbol{\varepsilon} \,|\, \sigma^2) = \sigma^2\boldsymbol{I}$, then LSE has a sampling variance

$$\begin{aligned}
\mathrm{Var}\left(\widehat{\boldsymbol{\beta}} \,|\, \boldsymbol{\beta}\right) &= \mathrm{Var}\left(\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\left(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}\right) \,|\, \boldsymbol{\beta}\right) = \mathrm{Var}\left(\boldsymbol{\beta} + \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{\varepsilon} \,|\, \boldsymbol{\beta}\right) \\
&= \mathrm{Var}\left(\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{\varepsilon}\right) = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\left(\sigma^2\boldsymbol{I}\right)\boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1} \\
&= \sigma^2\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1},
\end{aligned}$$

where the second row uses the fact that $\mathrm{Var}(\boldsymbol{A}\boldsymbol{\varepsilon}) = \boldsymbol{A}\mathrm{Var}(\boldsymbol{\varepsilon})\boldsymbol{A}^T$ for a constant matrix $\boldsymbol{A}$ and random vector $\boldsymbol{\varepsilon}$.

Often σ^2 is also estimated, using $\widehat{\sigma^2} = \mathrm{RSS}/(n-p-1)$. Residual standard error, $\sqrt{\widehat{\sigma^2}}$, describes average distance of an observation from the regression line. (It does not model exactly the expected distance, but rather the square root of the expected squared distance.)

The `summary(lm.fit)` command produced

- parameter estimates (or Coefficients),
- their standard errors (SE) (estimates for square root of the sampling variance of the parameter estimates),
- t-statistic (estimate/SE) and
- P-value under the null hypothesis that the parameter is 0 and errors are uncorrelated and have a Normal distribution $N(0, \sigma^2)$.

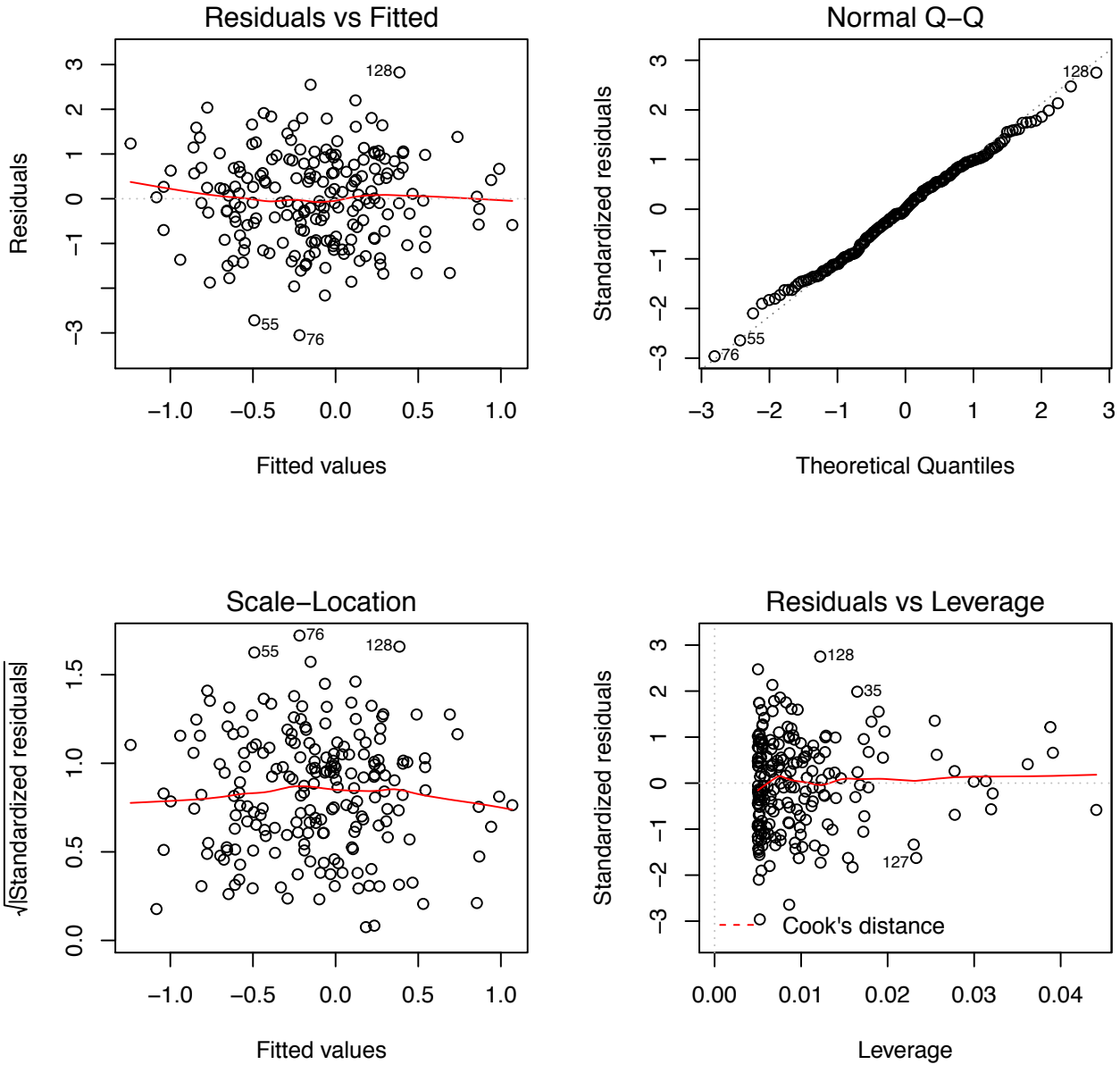
Under the above assumptions, the sampling distribution of t-statistic is t-distribution and hence q% confidence intervals are determined as $\widehat{\beta} \pm a \times \mathrm{SE}$, where a is the q/2% quantile of t-distribution with $n-p-1$ degrees of freedom. When σ^2 is known, the t-distribution is replaced by a Normal distribution, and same is approximately true when n becomes large, even if an estimate $\widehat{\sigma^2}$ is used in computing SE. In these cases, we often talk about z-statistic instead of t-statistic.

Last paragraph in output tells about the full model fit. R^2 is the proportion of variance explained by the linear model, i.e., $R^2 = 1 - \frac{RSS}{n-1}/\widehat{\mathrm{Var}}(y)$. The adjusted version penalizes for additional predictors and is defined as $R^2_{adj} = 1 - \frac{RSS}{n-p-1}/\widehat{\mathrm{Var}}(y)$. Note that if there is only the intercept parameter β_0 in the model

($p = 0$), then $R^2 = R^2_{adj} = 0$, and if the model explains data perfectly ($RSS = 0$), then $R^2 = R^2_{adj} = 1$. In all other cases, R^2 values are between 0 and 1 and larger values mean more variance explained by the model. (R^2_{adj} can get negative values in cases where the model explains very poorly compared to the number of parameters in the model.)

R^2 should not be the only value used to judge how suitable the model is for the data. One should always plot the data and the model fit in different ways to assess this question. For a simple linear model ($p = 1$), the above scatter plot and regression line is a good start. More generally, `plot`command applied to an `lm`-object gives the following four diagnostic plots (read http://data.library.virginia.edu/diagnostic-plots/).

```
par(mfrow=c(2,2)) #split the plot area in 2 rows and 2 columns, i.e. in 4 plots
plot(lm.fit) #make diagnostic plots
```



Validity of linear model

Let's list the assumptions behind the standard linear model and properties of its least squares estimates

(LSE).

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i$$

1. Additivity and linearity. We assume that each predictor acts **additively** (there is + between terms that involve different predictors) and that the effect of each predictor on outcome is **linear** (predictor is simply multiplied by a β coefficient). (How to extend linear model outside these assumptions?)
2. Error terms are independent of each other and of predictors. If this does not hold, then the amount of information in data does not correspond to the number of observations, and statistical inference based on theoretical distributions will be invalid. Additionally, LSE is not an optimal unbiased point estimate but a ganeralized least squares estimation, that takes into account the correlation between errors, gives more precise estimates.
3. Errors have same variance (homoscedasticity). If this does not hold, then a weighted linear regression would give more precise estimates.
4. Errors are Gaussian (i.e. have a normal distribution). Under this assumptions LSE coincides with the maximum likelihood estimate and hence has many optimality properties. However, LSE has several optimality properties even without this assumptions. For example, Gauss-Markov theorem says that LSE $\widehat{\beta}$ has the smallest sampling variance among all linear and unbiased estimators of β as long as errors are homoscedastic and uncorrelated, no matter what is their distribution. Gaussian errors is in practice the least important assumption out of the ones listed here.

Example 0.2 (`Auto`)

Let's have a look at the `Auto` dataset that is included in `ISLR` package.

```
#install.packages("ISLR") #do this when you run first time
library("ISLR") #This loads several datasets from the book ISLR
#?Auto #To check help
str(Auto)
```

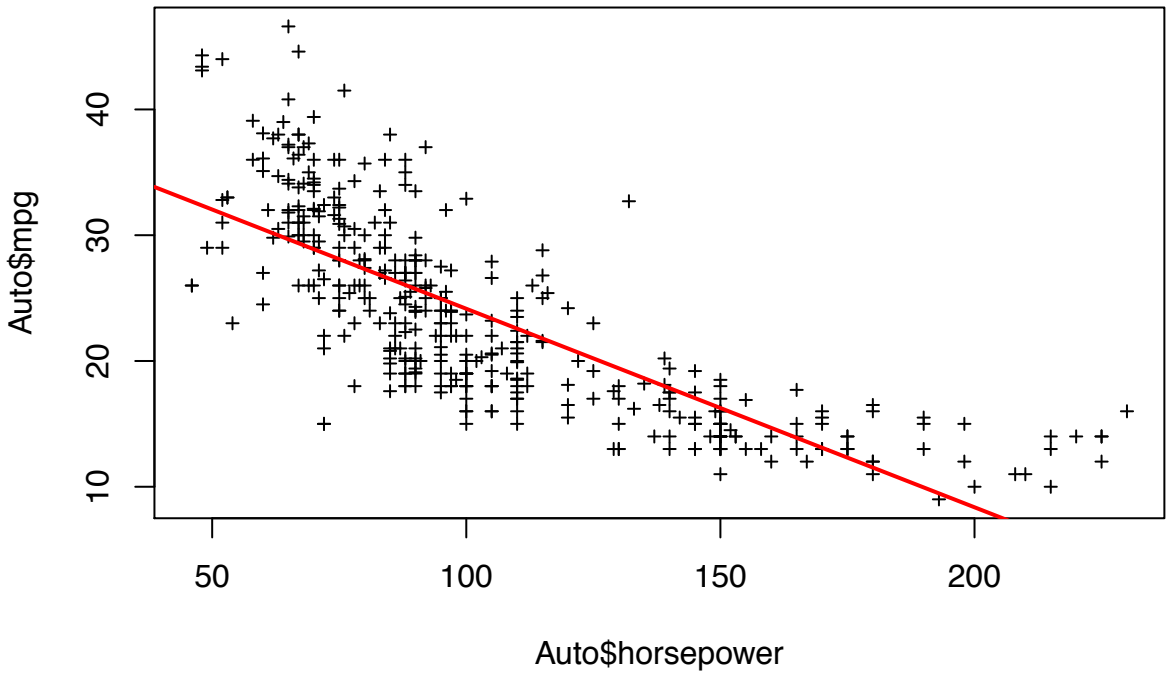
```
## 'data.frame':    392 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : num  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : num  3504 3693 3436 3433 3449 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ name        : Factor w/ 304 levels "amc ambassador brougham",..: 49 36 231 14 161 141 54 223 241 1
```

```
#plot pairs except "name"
#pairs(subset(Auto, select=-name), cex=0.5, pch="+")
#We want to explain miles per gallon, mpg
#Let's start with horsepower
lm.fit = lm(mpg ~ horsepower, data = Auto)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

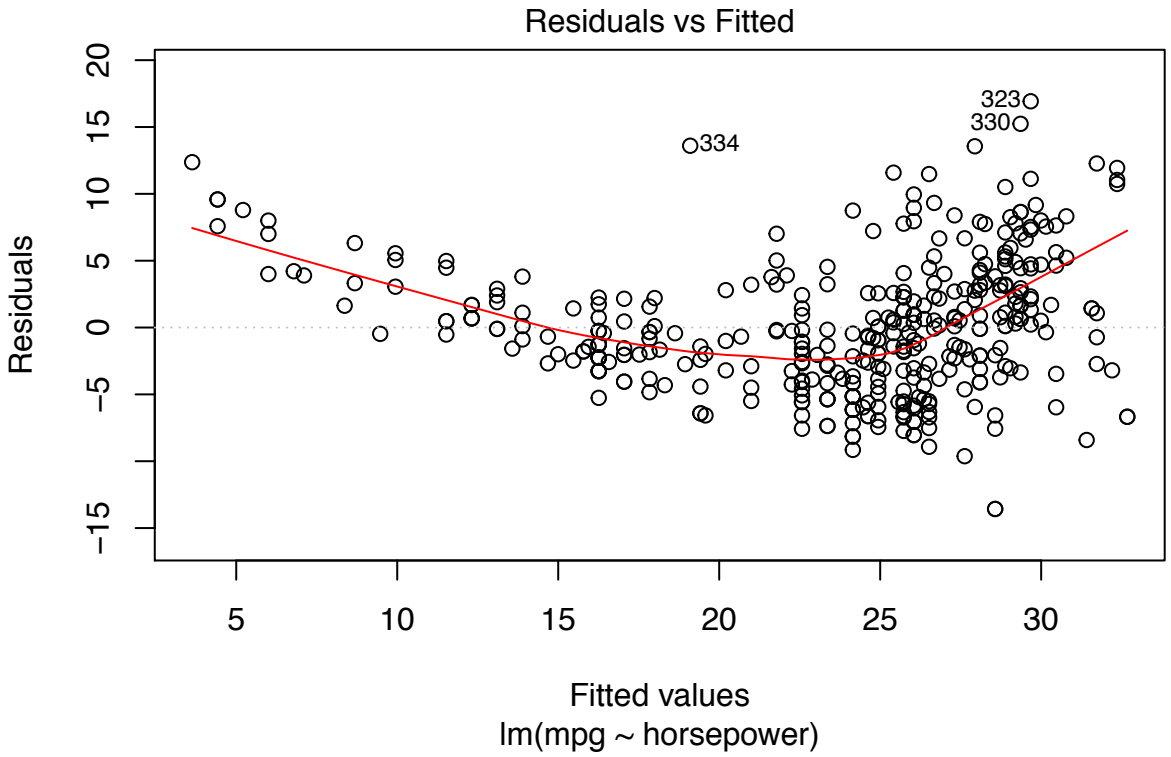
```
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.906 on 390 degrees of freedom
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
plot(Auto$horsepower, Auto$mpg, cex = 0.6, pch = 3)
abline(lm.fit, col = "red", lwd = 2)
```



This model explains 60.5% of all variation in the data and is very clearly statistically significant (P-values for horsepower and whole model are very low). But does it fit well the data, that is, does the linear model adequately describe the relationship between the two variables? Let's draw a residual plot by showing the 1st out of the standard diagnostic plots.

```
plot(lm.fit, 1) #plot only the 1st diagnostic plot that is the residuals
```



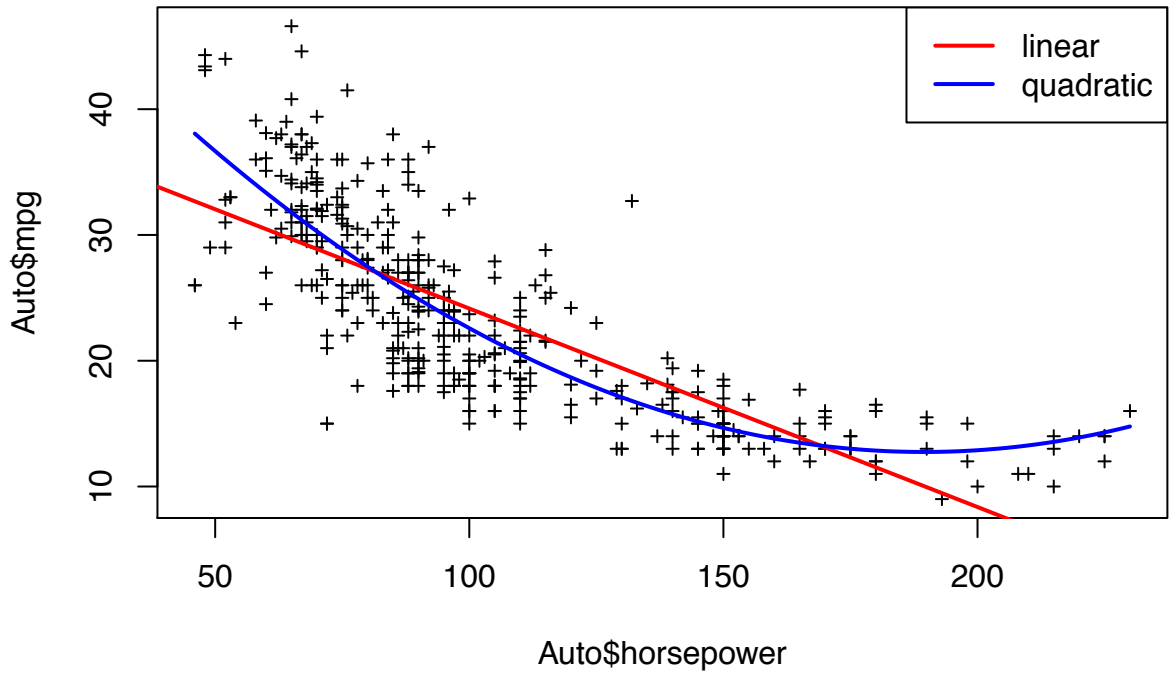
It seems that there is some pattern whereby the residuals do not agree with the assumption of errors being independent of the linear predictor. In such cases adding some polynomial terms may help. Let's add a square term and plot the model fit and the residual plot again.

```
lm.fit.2 = lm(mpg ~ horsepower + I(horsepower^2), data = Auto) #Note I() notation
summary(lm.fit.2)
```

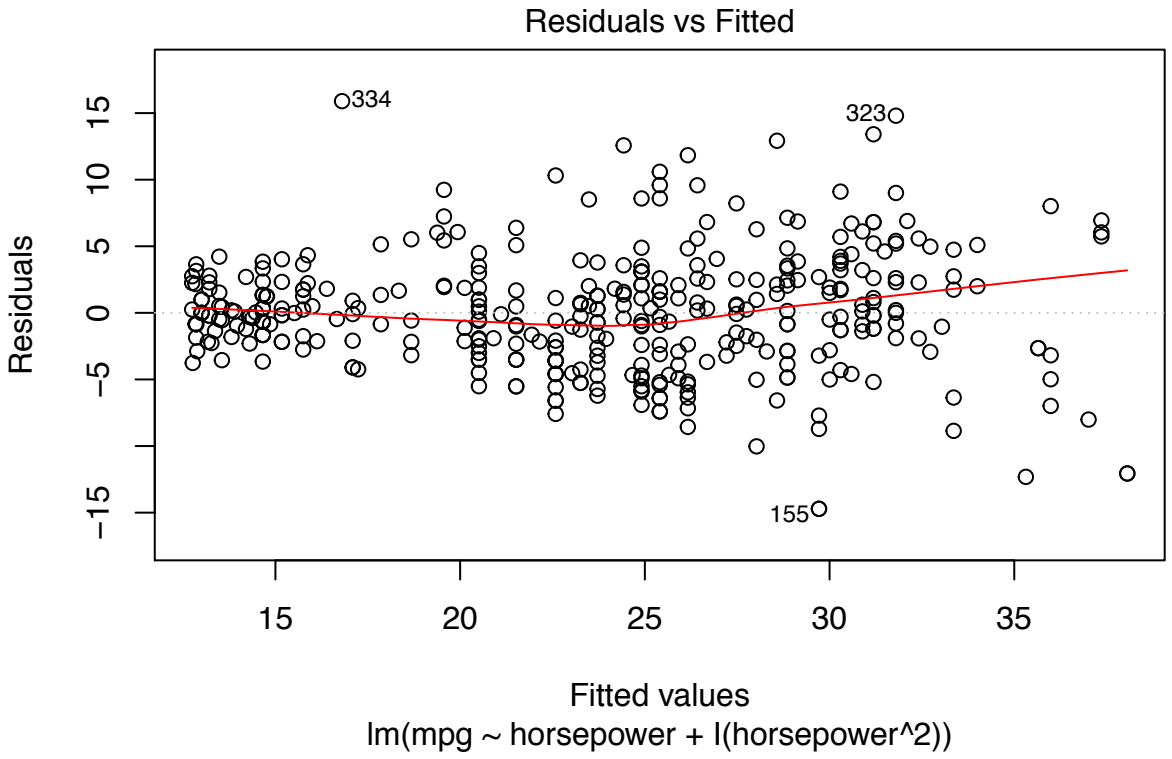
```
##
## Call:
## lm(formula = mpg ~ horsepower + I(horsepower^2), data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.7135  -2.5943  -0.0859   2.2868  15.8961
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     56.9000997  1.8004268   31.60   <2e-16 ***
## horsepower      -0.4661896  0.0311246  -14.98   <2e-16 ***
## I(horsepower^2)  0.0012305  0.0001221   10.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.374 on 389 degrees of freedom
## Multiple R-squared:  0.6876, Adjusted R-squared:  0.686
## F-statistic:   428 on 2 and 389 DF,  p-value: < 2.2e-16
```

```
plot(Auto$horsepower, Auto$mpg, cex = 0.6, pch = 3)
abline(lm.fit, col = "red", lwd = 2) #first order model in red

x.val = seq(min(Auto$horsepower), max(Auto$horsepower), length = 100) #grid of 100 points from x-axis
```

```r
#y values from the lm.fit.2 model for grid values in x.val:
y.val = predict.lm(lm.fit.2, newdata = data.frame(horsepower = x.val))

lines(x.val, y.val, col = "blue", lwd = 2) #2nd order model in blue
legend("topright", legend = c("linear","quadratic"), col = c("red","blue"), lwd = 2)
```




```r
plot(lm.fit.2, 1)
```



Now we explain 69% of `mpg` and residual plot is much more equal across fitted values. There is maybe some increased variance for larger fitted values. The points named in residual plot are possible outliers.

```
pick = c("334","155") #note quotes since these are row names here, not indexes
cbind(Auto[pick,], lm.fit.2$fitted.values[pick], lm.fit.2$residuals[pick])
```

```
##      mpg cylinders displacement horsepower weight acceleration year origin
## 334 32.7         6          168        132   2910         11.4   80      3
## 155 15.0         6          250         72   3432         21.0   75      1
##                name lm.fit.2$fitted.values[pick] lm.fit.2$residuals[pick]
## 334   datsun 280-zx                    16.80393                 15.89607
## 155 mercury monarch                    29.71355                -14.71355
```

Would cubic polynomial make even better fit?

```
lm.fit.3 = lm(mpg ~ poly(horsepower,3), data = Auto)
summary(lm.fit.3)
```

```
##
## Call:
## lm(formula = mpg ~ poly(horsepower, 3), data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.7039  -2.4491  -0.1519   2.2035  15.8159
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             23.446      0.221 106.105   <2e-16 ***
## poly(horsepower, 3)1 -120.138      4.375 -27.460   <2e-16 ***
## poly(horsepower, 3)2   44.090      4.375  10.078   <2e-16 ***
## poly(horsepower, 3)3   -3.949      4.375  -0.903    0.367
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.375 on 388 degrees of freedom
## Multiple R-squared:  0.6882, Adjusted R-squared:  0.6858
## F-statistic: 285.5 on 3 and 388 DF,  p-value: < 2.2e-16
```

Answer is no. (See uninteresting P-value and similar R2 compared to quadratic fit.)

Let's think about interpretation of parameters. In linear fit without the quadratic term each increase of 1 `horsepower` changes `mpg` by the same amount:

```
signif(lm.fit$coeff[2],2) #for linear model this is Delta mpg for 1 unit hpwr
```

```
## horsepower
##      -0.16
```

whereas for quadratic fit it varies with `horsepower` (here given near 50 and 150):

```
diff(predict(lm.fit.2, newdata = data.frame(horsepower = c(49.5,50.5))))
```

```
##         2
## -0.343136
```

```
diff(predict(lm.fit.2, newdata = data.frame(horsepower = c(149.5,150.5))))
```

```
##          2
## -0.0970288
```

This ends our short summary of the linear regression model with **R**. If you have not applied linear regression much in practice using **R**, go through additional lecture notes about Practical linear regression.

Centering the variables

In practice, we often mean-center the variables to simplify the linear model. Let's write down the motivation for that.

The least squares solution was derived by minimizing residual sum of squares (RSS). The solution $\widehat{\boldsymbol{\beta}}$ derived by setting the first derivative to 0 satisfies the *normal equations*

$$(\boldsymbol{X}^T\boldsymbol{X})\widehat{\boldsymbol{\beta}} = \boldsymbol{X}^T\boldsymbol{y} \quad \Leftrightarrow \quad \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}) = 0 \quad \Leftrightarrow \quad \boldsymbol{X}^T\widehat{\boldsymbol{\varepsilon}} = 0.$$

When the model includes the intercept term, then the first column of $\boldsymbol{X}$ is full of 1s, and the first element of the above vector $\boldsymbol{X}^T\widehat{\boldsymbol{\varepsilon}} = 0$ is
$\mathbf{1}^T\widehat{\boldsymbol{\varepsilon}} = \sum_i \widehat{\varepsilon}_i = 0$. This shows that **the sum of residuals of the least squares solution is 0** (when the model includes the intercept term).

By summing each term in the residual over individuals we thus get 0, hence

$$\sum_{i=1}^{n}(y_i - \widehat{\beta}_0 - x_{i1}\widehat{\beta}_1 - \ldots - x_{ip}\widehat{\beta}_p) = n(\overline{y} - \widehat{\beta}_0 - \overline{x_1}\widehat{\beta}_1 - \ldots - \overline{x_p}\widehat{\beta}_p) = 0,$$

where bar means a mean value over individuals. Hence we see that with the least squares solution the linear model equation is satisfied exactly for an "average" individual whose values of outcome y and each predictor x_i are the corresponding mean values over the samples:

$$\overline{y} = \widehat{\beta}_0 + \overline{x_1}\widehat{\beta}_1 + \ldots + \overline{x_p}\widehat{\beta}_p.$$

Suppose that we have mean-centered each predictor before fitting the model. Then $\overline{x_j} = 0$ for all j and $\widehat{\beta}_0 = \overline{y}$. If we have also mean-centered outcome y, then $\widehat{\beta}_0 = 0$. Thus, by mean-centering the predictors and the outcome, we can simplify the computation by dropping the intercept term from the model since it equals to zero in the least squares solution.

Consider such a mean-centered model and use α_j as coefficients to distinguish from β_j that correspond to the original model (without mean-centered variables). We know that $\alpha_0 = 0$ and

$$\begin{aligned} y_i - \overline{y} &= (x_{i1} - \overline{x}_1)\alpha_1 + \ldots + (x_{ip} - \overline{x}_p)\alpha_p + \varepsilon_i \\ y_i &= (\overline{y} - \overline{x}_1\alpha_1 - \ldots - \overline{x}_p\alpha_p) + x_{i1}\alpha_1 + \ldots + x_{ip}\alpha_p + \varepsilon_i \end{aligned}$$

We see that the latter form equals to the non-centered model

$$y_i = \beta_0 + x_{i1}\beta_1 + \ldots + x_{ip}\beta_p + \varepsilon_i,$$

if we set $(\overline{y} - \overline{x}_1\alpha_1 - \ldots - \overline{x}_p\alpha_p) = \beta_0$ and $\alpha_j = \beta_j$ for $j = 1, \ldots, p$. Since the least squares solution is unique (when model is of full rank), it follows that $\widehat{\alpha}_j = \widehat{\beta}_j$ for $j = 1, \ldots, p$ and hence **mean-centering does not change the values of other coefficients than the intercept**.

We conclude that by mean-centering both the outcome and all predictors in a linear regression model, we can drop the intercept term from the model (since its value will be 0) while the other coefficients have the same values as without mean-centering. In R we can fit a linear model without intercept term by using `0` or `-1` in the call: `lm(y ~ 0 + x)` or `lm(y ~ -1 + x)`.

Geometric interpretation in n-dimensional space

We can also interpret the least squares method as projecting the outcome vector $\boldsymbol{y}$ into the subspace defined by the intercept $\boldsymbol{x}_0$ and p predictor vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p$. Let's define matrix $\boldsymbol{H} = \boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T$ with which we can represent the fitted values as

$$\widehat{\boldsymbol{y}} = \boldsymbol{X}\widehat{\boldsymbol{\beta}} = \boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{H}\boldsymbol{y}.$$

$\boldsymbol{H}$ is also called "hat-matrix" that puts a hat on $\boldsymbol{y}$. In particular, $\boldsymbol{H}$ is a matrix of an orthogonal projection because $\boldsymbol{H} = \boldsymbol{H}^2$ and $\boldsymbol{H} = \boldsymbol{H}^T$. Thus we conclude that the least squares method finds the fitted values by projecting the observed values into the subspace defined by predictors.