Simple Linear Regression with R

Martin Schweinberger

December 28, 2017

Regression models are used to evaluate if one, or more predictors (or independent variables) correlate significantly with a dependent or outcome variable. The underlying logic is almost exactly identical to ANOVA (Analysis of Variance) designs, but in linguistics regression models have traditionally been used in analyses based on corpus data (e.g. sociolinguistic studies) while ANOVAs were the method of choice in psycholinguistics. The field-specific bias is, however, historical and cultural rather than caused by design specific advantages.

One difference between ANOVA and regression analysis is that regression analyses use a *t-statistic* rather than an *F-statistic* as ANOVAs do (but F is simply t squared ;-)). Both measure the ratio of explained to unexplained variance though.

The idea behind regression analyses can be best understood visually: imagine a line going through the data points in a coordinate system. What regression analyses do is to aim at finding that line going through the data points which has the smallest sum of *residuals*. Residuals are the difference between any observed data point and the predicted value for that data point. And the *sum of residuals* is the error in/of the regression model. The line of best fit line is then called the *regression line* (also called abline or the regression model). The slope of the line is called the *coefficient* and the point where the line crosses the y-axis is called the *intercept*.

The output in regression analyses provide a *standard error* (SE) which shows how much the coefficients (or estimates) differ across samples. A low standard error means that the coefficient will not differ much if we look at many samples while a high SE indicates that the coefficient will differ a lot between samples.

The example we are going to look at concerns the frequency of prepositions in the history of the English language. Using the *Penn Corpora of His-*

torical English (aka the Penn Parsed Corpus; see http://www.ling.upenn.edu/hist-corpora/) which consist of 605 texts written between 1125 and 1900, I extracted all words with the part-of-speech tag "/P" (preposition), calculated the per-1,000-words frequency of prepositions per text. After some data processing, we are left with a table of two columns: one with the text.id and the other with the normalized relative frequency of prepositions in the text. The question I want to investigate is whether the frequency of prepositions has increased over time.

The r code below is used to a) extract and process the data; b) visualize the data; c) perform the regression analysis, and d) test if the assumptions, that a linear regression is based on, are met.

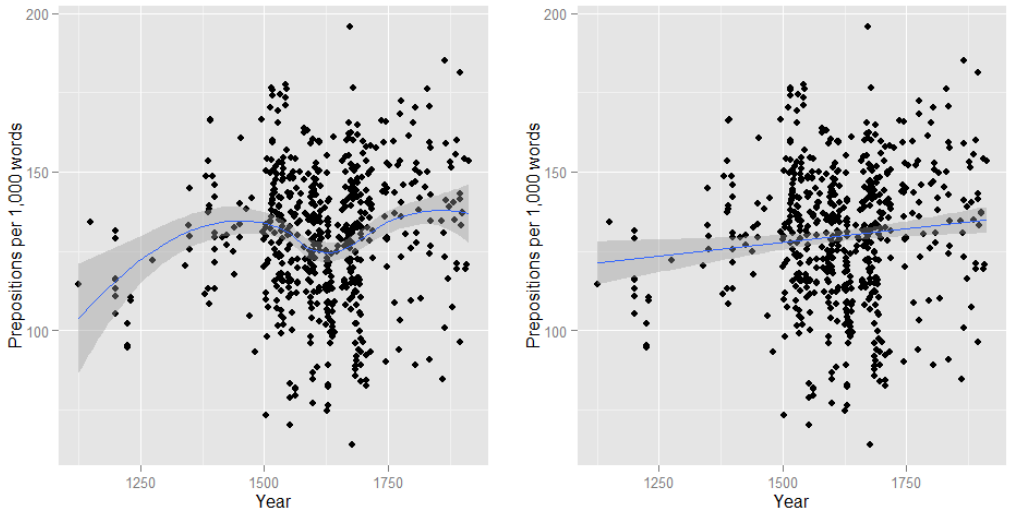
```
### --- Prepare data
# Remove all lists from the current workspace
rm(list=ls(all=T))
# Install packages we need or which may be useful
# (to activate just delete the #)
#install.packages("QuantPsyc")
#install.packages("car")
# Initiate the packages
library(QuantPsyc)
library(car)
library(ggplot2)
source("C:\\R/multiplot_ggplot2.R")
# this call will only work, if you have stored this function in
# the directory specified in the call
source("C:\\R/slr.summary.tb") # load modified summary function for slr

##################################################################
### Load and manipulate data
##################################################################
# WARNING: To use this script you need to set our own paths!
# Your path should be the path to the corpus on your own computer!
# Remember to use double backslash instead of single backslash, if
# you use Windows on your machine.
# Read in data
slr.data <- read.delim("C:\\MyProjects\\SimpleLinearRegression/slr.data.txt", header = TRUE)
attach(slr.data) # we attach the data so we don't need to
```

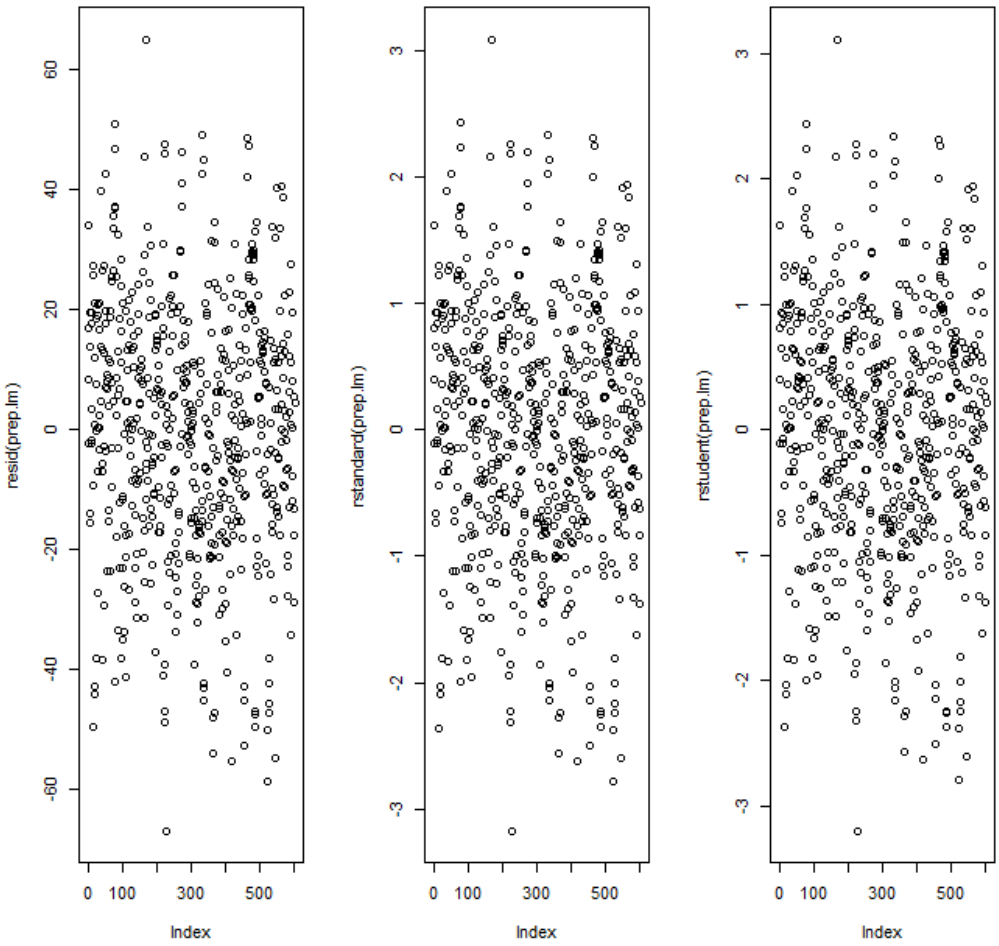
```
   specify the path all the time
# remove columns we do not need
slr.data <- as.data.frame(cbind(slr.data$datems, slr.data$P.
   ptw))
colnames(slr.data) <- c("year", "prep.ptw") # add column
   names
slr.data <- slr.data[!is.na(slr.data$year) == T, ] # delete
   incomplete cases
head(slr.data) # inspect data
# year prep.ptw
#1 1736 166.01
#2 1711 139.86
#3 1808 130.78
#4 1878 151.29
#5 1743 145.72
#6 1807 152.59
str(slr.data) # inspect data structure
#'data.frame': 603 obs. of 2 variables:
# $ year : num 1736 1711 1808 1878 1743 ...
# $ prep.ptw: num 166 140 131 151 146 ...
summary(slr.data) #inspect data values
# year prep.ptw
# Min. :1125 Min. : 63.97
# 1st Qu.:1545 1st Qu.:115.66
# Median :1615 Median :130.78
# Mean :1619 Mean :129.81
# 3rd Qu.:1687 3rd Qu.:144.08
# Max. :1913 Max. :195.86
########################################
### Visualize and eyeball data
# set up this plot from scratch
p2 <- ggplot(slr.data, aes(year, prep.ptw)) +
  geom_point() +
  labs(x = "Year") +
  labs(y = "Prepositions per 1,000 words") +
  geom_smooth()

# set up this plot from scratch
p3 <- ggplot(slr.data, aes(year, prep.ptw)) +
  geom_point() +
  labs(x = "Year") +
  labs(y = "Prepositions per 1,000 words") +
  geom_smooth(method = "lm") # with linear model smoothing!

multiplot(p2, p3, cols = 2)
```




```
# set up Simple Linear egression model and inspect properties of the model
prep.lm <- lm(prep.ptw ~ year, data = slr.data)
summary(prep.lm)
#Call:
#lm(formula = prep.ptw ~ year, data = slr.data)
#Residuals:
# Min 1Q Median 3Q Max
#-66.842 -13.523 1.183 14.086 65.117
#Coefficients:
# Estimate Std. Error t value Pr(>|t|)
#(Intercept) 1.021e+02 1.086e+01 9.397 <2e-16 ***
#year 1.713e-02 6.691e-03 2.560 0.0107 *
#---
#Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#Residual standard error: 21.11 on 601 degrees of freedom
# (2 observations deleted due to missingness)
#Multiple R-squared: 0.01079, Adjusted R-squared: 0.00914
#F-statistic: 6.553 on 1 and 601 DF, p-value: 0.01071
# use plots to check if there are problems with the model
# set graphic's parameters to display 3 plots in one row
par(mfrow = c(1, 3))
plot(resid(prep.lm))
plot(rstandard(prep.lm))
plot(rstudent(prep.lm))
par(mfrow = c(1, 1)) # restore original graphic's parameters
```



The left plot shows the residuals of the model (the difference between the observed and the predicted value). A shortcoming of this plot is that the residuals are not standardized, i.e. we cannot compare them to residuals from another regression model. To overcome this shortcoming, we have a look at standardized residuals which are residuals divided by their standard deviation (see the plot in the middle). This way, residuals are converted into standardized residuals, but more importantly, they now represent z-scores and we can use the z-distribution to determine which points are problematic. Here are three rules of thumb for diagnosing regression models based on residuals (see Field, Miles & Field (2012:268-269):

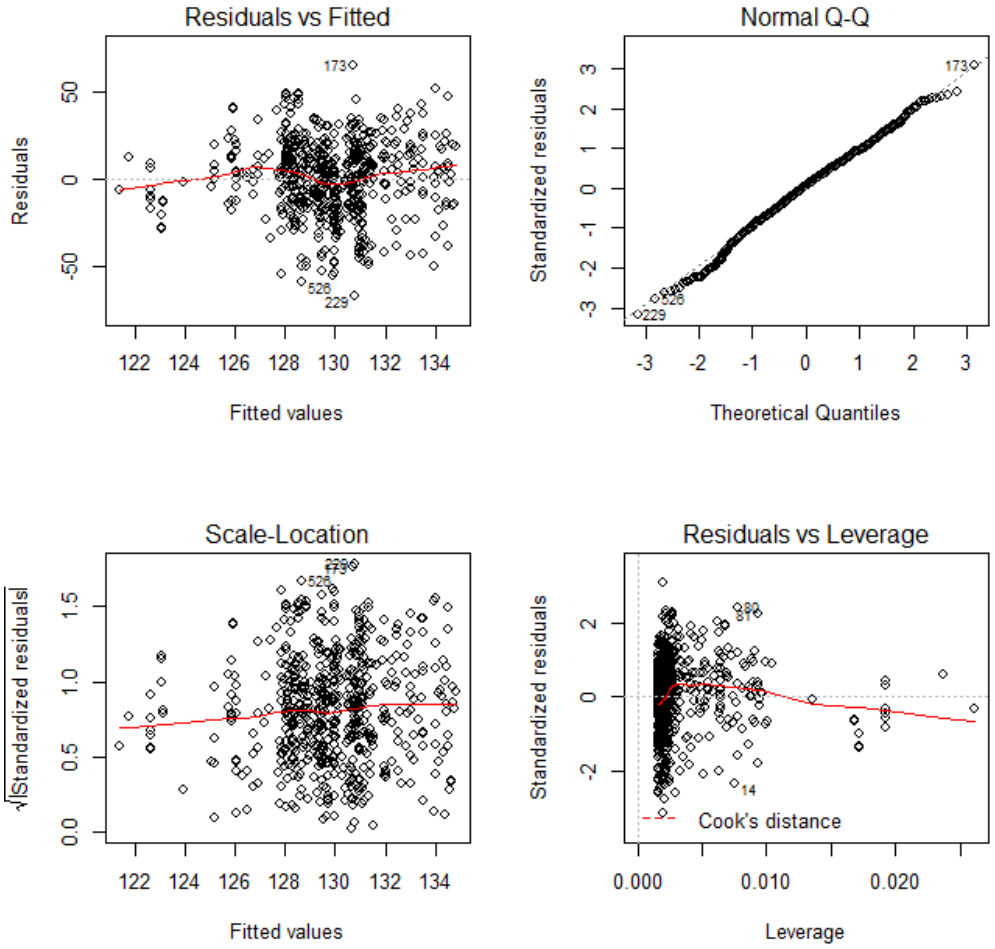
1. Points with extreme values, i.e. values above 3 (3.29 to be exact), should be excluded.

2. If more than 1% of data points in our sample has values above 2.5 (2.58 to be exact), then the error in our model is too high.

3. If more than 5% of data points in our sample has values above 2 (1.96 to be exact), then the error in our model is too high.

The plot on then right shows the *studentized residuals*, i.e. the *adjusted predicted value* of each point is divided by the standard error of the residuals. This way, we can use the Student's t-distribution to diagnose our model. Adjusted predicted values are also residuals, but a specific kind of residual: a model is calculated without one data point, then, this model is used to predict the data point. The difference between the observed data point and the point predicted by the model without this point is called the adjusted predicted value. In summary, studentized residuals are very helpful to detect influential data points. The plots show that there are two potentially problematic cases – the dots which are at the very top and at the very bottom. These two data points stand apart from the other points and are thus probably outliers. We will check later if they need to be removed.

```
# Create a 2x2 matrix of diagnostic plots
par(mfrow = c(2, 2))
plot(prep.lm)
par(mfrow = c(1, 1))
```



The diagnostic plots look very good - but what does this mean? (I will go more into detail in the post on Multiple Linear Regression)

Well, the upper left plot is useful for a) finding outliers, or b) finding correlations between residuals and fitted values: if there were a noticeable trend visible in the line or the data points (e.g. an upward, downward, or zig-zag trend in the line), we would have a problem and would have to delete data points.

The upper right plot is useful for checking if the residuals are normally distributed (a necessary condition for linear regressions). If the dots lie on the line than the residuals are indeed distributed following a normal distribution. If the dots diverge from the line at the top and bottom, then this indicates

poor model fit.

Let's consider the lower left plot: regression models are also based on the assumption of homoscedasticity, i.e. that the variance of the residuals do not change with x or - to put it differently - that the variance of the residuals do not correlate with the predictor or predictors in multiple regression). The assumption of homoscedasticity is met, then we should see a flat line without upward or downward trend. If there is a trend, you have a problem called heteroscedasticity.

The lower right plot is useful for detecting leverage, i.e. points that over-proportionally influence the regression model (this should not be the case as in simple linear regression all data points have the same weight). If there are points that have a high leverage, you should try a) to use a robust linear regression (which assigns different weights to the data points or b) delete the data points as outliers. This plot also shows Cook's distance which is a measure that captures how much the regression would change if the point I question was deleted. So Cook's distance assesses the influence of individual points on the model as a whole. Points which have a value for Cook's distance that is greater than 1 are problematic (Field, Miles & Field 2012:269). Leverage is also a measure that provides an estimate of how much a data point influences the accuracy of the model. Leverage values lie between 0 (no influence) to 1 (major influence: no good!). To check if a given point is still acceptable, you need to calculate a cut-off point (either *3(k + 1)/n)* or *2(k + 1)/n)*).

We will now check, if we need to exclude data points and also calculate the standardized β.

```
# Extract standardized betas
lm.beta(prep.lm)
# year
#0.1038566
#####################################################
### Write a function for a neat output table
lm.summary <- function(x) {
p.nice <- function(z) {
as.vector(unlist(sapply(z, function(w) {
ifelse(w < .001, return("p < .001***"),
ifelse(w < .01, return("p < .01**"),
ifelse(w < .05, return("p < .05*"), return(w)))) } ))) }
intercept <- c(
round(summary(x)[[4]][1], 2),
"",
round(summary(x)[[4]][3], 2),
round(summary(x)[[4]][5], 2),
round(summary(x)[[4]][7], 4),
```

```
p.nice(summary(x)[[4]][7]))
predictor <- c(
round(summary(x)[[4]][2], 2),
round(lm.beta(x)[[1]], 4),
round(summary(x)[[4]][4], 2),
round(summary(x)[[4]][6], 2),
round(summary(x)[[4]][8], 4),
p.nice(summary(x)[[4]][8]))
mdl.statz <- c("", "", "", "", "", "Value")
nbcases <- c("", "", "", "", "", length(summary(x)[[3]]))
rse <- c("", "", "", "", "",
round(summary(x)[[6]], 2))
multR2 <- c("", "", "", "", "", round(summary(x)[[8]], 4))
adjR2 <- c("", "", "", "", "", round(summary(x)[[9]], 4))
F <- c("", "", "", "", "",
round(summary(x)[[10]][1], 2))
p <- c("", "", "", "", "", round(summary(x)[[4]][8], 4))
slrm.tb <- rbind(intercept, predictor, mdl.statz, nbcases,
   rse, multR2, adjR2, F, p )
colnames(slrm.tb) <- c(colnames(summary(x)[[4]])[1],
"Std. Beta",
colnames(summary(x)[[4]])[c(2:4)],
"P-value sig.")
rownames(slrm.tb) <- c(
rownames(summary(x)[[4]])[1],
rownames(summary(x)[[4]])[2],
"Model statistics", "Number of cases in model",
paste("Residual standard error", paste("on", summary(x)
   [[7]][2],"DF")),
"Multiple R-squared", "Adjusted R-squared",
paste("F-statistic",
paste("(", round(summary(x)[[10]][2], 0), ", ",
round(summary(x)[[10]][3], 0), ")", sep = "", collapse = ""))
   ,
"Model p-value")
slrm.tb <- as.data.frame(slrm.tb)
return(slrm.tb)
}
lm.summary(prep.lm) # inspect the results
# Estimate Std. Beta Std. Error t value Pr(>|t|) P-value sig.
#(Intercept) 102.09 10.86 9.4 0 p < .001***
#year 0.02 0.1039 0.01 2.56 0.0107 p < .05*
#Model statistics Value
#Number of cases in model 603
#Residual standard error on 601 DF 21.11
#Multiple R-squared 0.0108
#Adjusted R-squared 0.0091
#F-statistic (1, 601) 6.55
#Model p-value 0.0107
```

Typically, the results of regression models are reported using a table like the one we just created (see again below).

	Estimate	Std. Beta	Std. Error	t-value	Pr(>|t|)	P-value sig.
(Intercept)	102.09		10.86	9.4	0	p < .001***
year	0.02	0.1039	0.01	2.56	0.0107	p < .05*

Model statistics	Value
Number of cases in model	603
Residual standard error on 601 DF	21.11
Multiple R-squared	0.0108
Adjusted R-squared	0.0091
F-statistic (1, 601)	6.55
Model p-value	0.0107

However, you can also summarize the results of regression analyses in prose form which would be something like this:

A simple linear regression model was fitted to the data. Visual inspection of diagnostic plots and data driven methods did not suggest problems concerning outliers and model fit. The final linear regression model was based on 603 data points and correlated significantly with the data (R¡sup¿2¡/sup¿: 0.0108, F-statistic (1, 601): 6.553, p-value: 0.0107*) and confirmed a significant positive correlation between the year the text was written in and the relative frequency of prepositions in the text (coefficient: .02, std. β: 0.1039, SE: 6.691e-03, t-value: 2.560, p-value: .0107 *).

References

Field, Andy, Jeremy Miles & Zoe Field. 2012[2]. *Discovering statistics using R*. London, Thousand oaks, CA, New Delhi, Singapore: SAGE.