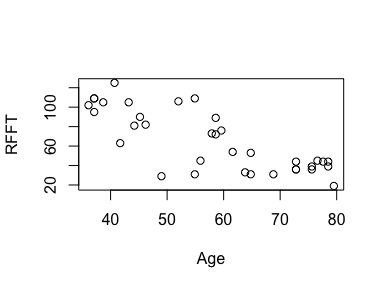
Lab 15: Simple linear regression

STAT218

## Hand fitting a regression model

### Warm-up

The plot below shows RFFT score (a cognitive assessment) against age for 35 respondents from the Prevention of REnal and Vascular END-stage Disease (PREVEND) study.



We will be using this dataset as an example to learn how to implement simple linear regression (SLR). An SLR model specifies that the mean response is linear in one explanatory variable, in this case:

This is the equation of a line in slope-intercpet form. So, as a warm up, we’ll explore “hand fitting” a line to the PREVEND data.

|  |
| --- |
| Your turn |
| Use the hand.fit() function to draw a line through the RFFT-Age scatterplot by specifying an intercept b0 and a slope b1. Adjust until you obtain a line that you think best fits the data. Write down the values that give your ‘best’ fit.  hand.fit(b0 = 80, b1 = -0.2) |

### Hand fitting a model

No model is perfect; there will always be some error. The model residuals are the differences between observed values and the values indicated by the model:

The residuals represent the model errors. The least squares estimates are the values of the slope and intercept parameters that minimize the sum of squared residuals , which is a measure of total error. Before we learn how to compute these estimates directly, it is instructive to try and find them by hand.

|  |
| --- |
| Your turn |
| Adding the argument .resid = T will show the magnitude of the residuals on the plot as red vertical lines indicating the distance from each observation to the fitted line. It will also add a title giving the sum of squared residuals (SSR):  Input your ‘best’ values from before and look at the SSR value. Adjust until you make the SSR as low as you can.  hand.fit(b0 = 80, b1 = -0.2, .resid = T) |

## Least squares estimation

We will continue to use the PREVEND data to illustrate estimation, residual diagnostics, and inference for SLR models. We’ll fit the model

Remember that the error model has an additional parameter: the standard deviation .

### The lm() function

Models are fit using the lm() function (short for linear model), which uses a formula for the model specification. Most of the time, data are supplied as a data frame.

# fit the model  
fit <- lm(RFFT ~ Age, data = prevend)  
  
# inspect output  
fit

Call:  
lm(formula = RFFT ~ Age, data = prevend)  
  
Coefficients:  
(Intercept) Age   
 162.973 -1.684

The print behavior of a stored lm object shows only the call (command executed) and the least squares coefficient estimates.

### Parameter estimates

The least squares estimates can be retrieved either directly from the lm object, or using the coef() function. The estimate of the error standard deviation can be retrieved with the sigma() function. These are shown below.

# retrieve coefficients  
coef(fit)

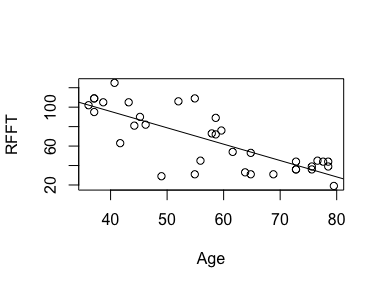
(Intercept) Age   
 162.973062 -1.684404

# retrieve estimate of error SD  
sigma(fit)

[1] 19.40402

To visualize the model, make a scatterplot and draw a line using the coefficient estimates:

# scatterplot  
plot(prevend)  
  
# add line  
abline(a = coef(fit)[1], b = coef(fit)[2])



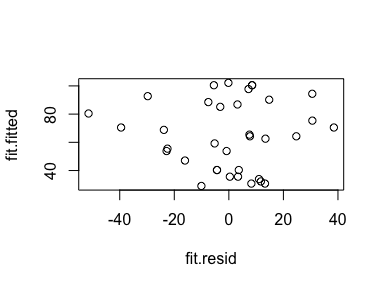
### Residual diagnostics

The fitted values and residuals can be retrieved from the output of lm() and used to generate the three common diagnostic plots:

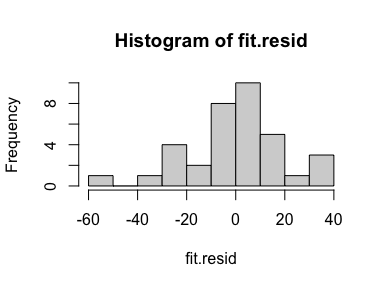
* residual-fit plot
* residual histogram
* quantile-quantile plot

The residual-fit plot is used to check for model misspecification (*e.g.*, a nonlinear relationship or nonconstant residual variability). The residual histogram and QQ plot are used to check for non-normality.

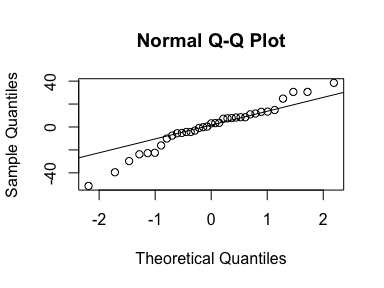
# fitted values  
fit.fitted <- fit$fitted.values  
  
# residuals  
fit.resid <- fit$residuals  
  
# residual vs fit  
plot(fit.resid, fit.fitted)  
abline(h = 0)



# residual histogram  
hist(fit.resid)



# quantile-quantile plot  
qqnorm(fit.resid)  
qqline(fit.resid)



In this case, each of the diagnostic plots looks acceptable:

* no patterns in residual-fit plot
* residual histogram is close enough to a bell curve shape
* residual quantile-quantile plot places most points on or near the line

### Inference

While it is not really a separate calculation to obtain significance tests for the coefficients, it is good practice to check the diagnostic plots first.

To view the significance tests, simply print the model summary. To obtain interval estimates, use confint().

# model summary  
summary(fit)

Call:  
lm(formula = RFFT ~ Age, data = prevend)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-51.437 -6.502 3.162 9.785 38.501   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 162.9731 13.7266 11.87 1.86e-13 \*\*\*  
Age -1.6844 0.2295 -7.34 1.99e-08 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 19.4 on 33 degrees of freedom  
Multiple R-squared: 0.6201, Adjusted R-squared: 0.6086   
F-statistic: 53.87 on 1 and 33 DF, p-value: 1.995e-08

# confidence intervals  
confint(fit, level = 0.95)

2.5 % 97.5 %  
(Intercept) 135.046173 190.899951  
Age -2.151311 -1.217498

## Age of the universe

The hubble data comprise relative velocities and distances for each of 24 galaxies. We’ll use this to first estimate the inverse of the hubble constant and subsequently obtain a confidence interval for the age of the universe.

Steps:

1. Fit the model . In this model, is the inverse of the Hubble constant.
2. Check residual diagnostics
3. Obtain a confidence interval for .
4. Multiply by a conversion factor to obtain a confidence interval for the age of the universe.

You will want to fit the model in step (1) without an intercept. This can be done using a formula of the type y ~ x - 1.

# load data  
load('data/hubble.RData')  
  
# make a plot  
  
# fit model  
  
# residual diagnostics  
  
# CI for inverse of hubble constant  
  
# conversion  
c <- 978440076094