Textbook Notes

library(ggplot2)  
library(UsingR)  
library(dplyr)  
library(reshape)  
library(datasets)  
data(diamond)  
data(galton)

## Chapter 1. Introduction

### Before beginning

The recommendation for the swirl module for this course is ill-advised in my opinion. It's terrible and confusing.

### Regression models

"Regression models are the workhorse of data science. They are the most well described, practical and theoretically understood models in statistics. A data scientist well versed in regression models will be able to solve and incredible array of problems.

Perhaps the key insight for regression models is that they produce highly interpretable model fits. This is unlike machine learning algorithms, which often sacrifice interpretability for improved prediction performance or automation."

### Motivating examples

* Francis Galton's data on parent and child heights
* Simply Statistics versus Kobe Bryant

### Summary notes: questions for this book

Questions that can be answered using regression:

1. Prediction: e.g., use parents' heights to predict childrens' heights
2. Modeling: Find a parsimonious, easily-described mean relationship between parent and children heights
3. Covariation: Investigate the variation in child heights that appears unrelated to parent heights

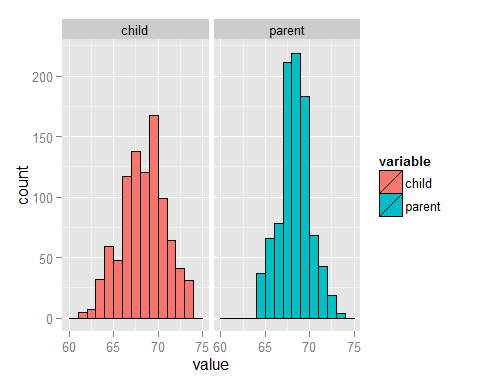
### Exploratory analysis of Galton's data

**NOTE** that the following slyly and silently throws in usage of the reshape library, without making any reference to it. It's a great library and ought to be learned, but it is not easy to grasp on first contact.

long <- melt(galton)

## Using as id variables

g <- ggplot(long, aes(x = value, fill = variable))  
g <- g + geom\_histogram(colour = "black", binwidth=1)  
g <- g + facet\_grid(. ~ variable)  
g



#### Finding the middle via least squares

Find the middle, or , that minimizes:

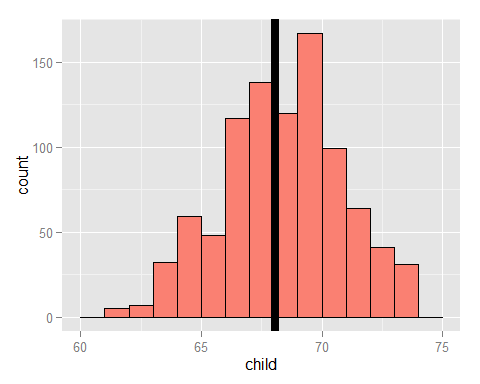
Spoiler: The answer is .

#### Experiment

library(manipulate)  
myHist <- function(mu){  
mse <- mean((galton$child - mu)^2)  
g <- ggplot(galton, aes(x = child)) + geom\_histogram(fill = "salmon", colour\  
= "black", binwidth=1)  
g <- g + geom\_vline(xintercept = mu, size = 3)  
g <- g + ggtitle(paste("mu = ", mu, ", MSE = ", round(mse, 2), sep = ""))  
g  
}  
manipulate(myHist(mu), mu = slider(62, 74, step = 0.5))

Uses the manipulate function to allow for interactive exploration to discover that the value of that minimizes squared differences is about 68, which is near .

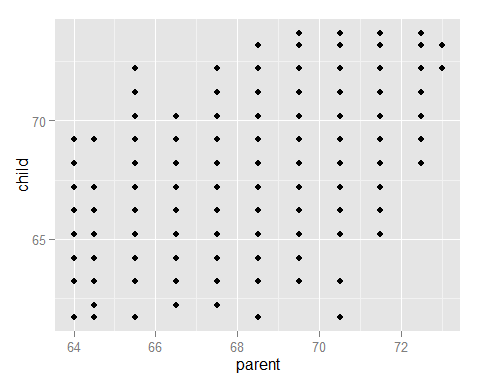
g <- ggplot(galton, aes(x = child)) + geom\_histogram(fill = "salmon", colour = "black", binwidth=1)  
g <- g + geom\_vline(xintercept = mean(galton$child), size = 3)  
g



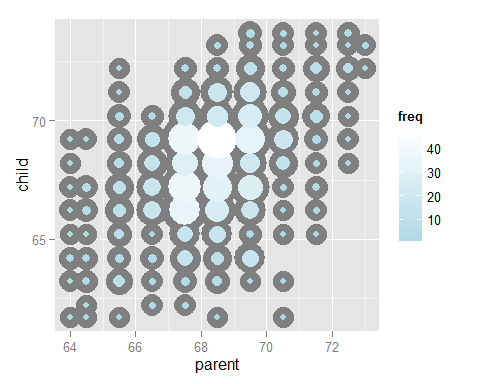
### The math (not required)

### Comparing children's heights and their parents' heights

ggplot(galton, aes(x = parent, y = child)) + geom\_point()



freqData <- as.data.frame(table(galton$child, galton$parent))  
names(freqData) <- c("child", "parent", "freq")  
freqData$child <- as.numeric(as.character(freqData$child))  
freqData$parent <- as.numeric(as.character(freqData$parent))  
g <- ggplot(filter(freqData, freq > 0), aes(x = parent, y = child))  
g <- g + scale\_size(range = c(2, 20), guide = "none" )  
g <- g + geom\_point(colour="grey50", aes(size = freq+20, show\_guide = FALSE))  
g <- g + geom\_point(aes(colour=freq, size = freq))  
g <- g + scale\_colour\_gradient(low = "lightblue", high="white")   
g



### Regression through the origin

A line requires two parameters, intercept and slope. If we want to focus only on the slope, we can subtract the mean from each data point, so that the mean becomes 0, and thus the line passes through the origin. So now we can determine the slope that minimizes:

Each is the vertical height of a line through the point . So is the vertical distance between each observed data point and the point on the line .

The following is another interactive experiment that demonstrates that the value that minimizes is about 0.646.

y <- galton$child - mean(galton$child)  
x <- galton$parent - mean(galton$parent)  
freqData <- as.data.frame(table(x, y))  
names(freqData) <- c("child", "parent", "freq")  
freqData$child <- as.numeric(as.character(freqData$child))  
freqData$parent <- as.numeric(as.character(freqData$parent))  
myPlot <- function(beta){  
g <- ggplot(filter(freqData, freq > 0), aes(x = parent, y = child))  
g <- g + scale\_size(range = c(2, 20), guide = "none" )  
g <- g + geom\_point(colour="grey50", aes(size = freq+20, show\_guide = FALSE))  
g <- g + geom\_point(aes(colour=freq, size = freq))  
g <- g + scale\_colour\_gradient(low = "lightblue", high="white")  
g <- g + geom\_abline(intercept = 0, slope = beta, size = 3)  
mse <- mean( (y - beta \* x) ^2 )  
g <- g + ggtitle(paste("beta = ", beta, "mse = ", round(mse, 3)))  
g  
}  
manipulate(myPlot(beta), beta = slider(0.6, 1.2, step = 0.02))

### The solution

And here is the official answer. Both and have been centered at 0. The -1 in the formula indicates that we do not want an intercept in the model.

lm(I(child - mean(child))~ I(parent - mean(parent)) - 1, data = galton)

##   
## Call:  
## lm(formula = I(child - mean(child)) ~ I(parent - mean(parent)) -   
## 1, data = galton)  
##   
## Coefficients:  
## I(parent - mean(parent))   
## 0.6463

**IMPORTANT NOTE** re the graph in this section: "(Note that I shifted the origin back to the means of the original data.)" The code in the previous section centers the data at the origin. But it's irrelevant in terms of the slope, which is unaffected by the intercept or centering:

lm(child ~ parent, data=galton)

##   
## Call:  
## lm(formula = child ~ parent, data = galton)  
##   
## Coefficients:  
## (Intercept) parent   
## 23.9415 0.6463

### Exercises

1. Consider the data set given by x=c(0.725, 0.429, -0.372, 0.863). What value of minimizes sum((x - mu)^2)?

x <- c(0.725, 0.429, -0.372, 0.863)  
mu <- mean(x)  
mu

## [1] 0.41125

1. Reconsider the previous question. Suppose that weights given were w=c(2, 2, 1, 1) so that we wanted to minimize sum(w \* (x - mu)^2) for mu. What value would we obtain?

w <- c(2, 2, 1, 1)  
mu <- sum(w \* x) / sum(w)  
mu

## [1] 0.4665

1. Take the Galton and obtain the regression through the origin slope estimate where the centered parental height is the outcome and the child's height is is the predictor.

yc <- galton$parent - mean(galton$parent)  
xc <- galton$child - mean(galton$child)  
lm(yc ~ xc)

##   
## Call:  
## lm(formula = yc ~ xc)  
##   
## Coefficients:  
## (Intercept) xc   
## -1.997e-15 3.256e-01

#Or  
sum(yc \* xc) / sum(xc^2)

## [1] 0.3256475

The phrasing "the regression through the origin slope estimate" is unclear. The point of the exercise was unclear. What is wanted (which the video answer makes clear) is to demonstrate that the slope is equal to:

This is because, when we use regression through the origin, the residuals sum to 0.

But this is not covered until the next chapter (and in fact, the chapter after that). This is an example of what made my first attempt to take this course conclude that it was sloppily arranged and poorly organized. Why not throw in a few questions from the last chapter? Or from the last chapter of the next course?

## Chapter 2. Notation

### Some basic definitions

### Notation for data

describes data points for a random variable (in this case, ). For example, as .

### The emprical mean

If we subtract the mean from each data point, the new data set has mean 0. So this:

has a mean of 0. This is called **centering** the variable.

### The empirical standard deviation and variance

Variance:

Standard deviation:

### Normalization

What is shown here is *standardization* using Z-scores, which is only one form of normalization. Others include min-max. It's somewhat misleading to present Z-score standardization as what appears to be **the** definition of "normalization".

The above would have mean 0 and a standard deviation of 1.

### Empirical covariance (and correlation)

### Some facts about correlation

* Correlation is exactly 1 or -1 only when all observations fall exactly on a straight line with a non-zero slope.
* measures the strength of a *linear* relationship between two variables, with a stronger relationship the closer the correlation is to 1 or -1.
* A correlation of 0 indicates no linear relationship at all between two variables.

### Exercises

1. Take the Galton dataset and find the mean, standard deviation and correlation between the parental and child heights.

parent.mean <- mean(galton$parent)  
parent.sd <- sd(galton$parent)  
child.mean <- mean(galton$child)  
child.sd <- sd(galton$child)  
galton.rho <- cor(galton$child, galton$parent)  
parent.mean

## [1] 68.30819

parent.sd

## [1] 1.787333

child.mean

## [1] 68.08847

child.sd

## [1] 2.517941

galton.rho

## [1] 0.4587624

1. Center the parent and child variables and verify that the centered variable means are 0.

parent.c <- galton$parent - mean(galton$parent)  
child.c <- galton$child - mean(galton$child)  
mean(parent.c)

## [1] 9.775954e-16

mean(child.c)

## [1] 4.817867e-16

1. Rescale the parent and child variables and verify that the scaled variable standard deviations are 1.

parent.scale <- galton$parent / sd(galton$parent)  
child.scale <- galton$child / sd(galton$child)  
sd(parent.scale)

## [1] 1

sd(child.scale)

## [1] 1

1. Normalize the parent and child heights. Verify that the

What is wanted is standardization, a particular form of normalization.

parent.c <- galton$parent - mean(galton$parent)  
child.c <- galton$child - mean(galton$child)  
parent.std <- parent.c / sd(parent.c)  
child.std <- child.c / sd(child.c)  
mean(parent.std)

## [1] 5.501733e-16

sd(parent.std)

## [1] 1

mean(child.std)

## [1] 2.183943e-16

sd(child.std)

## [1] 1

## Chapter 3. Ordinary Least Squares

### General least squares for linear equations

The goal is to minimize the sum of the squares:

The result:

If we use regression through the origin (to review, that means centering both and on their means), then:

This is simply the covariance of and . That is because, when the data for both variables has been centered, they both have standard deviations of 1, so and .

### My side note

This is always true:

For example, from the father.son dataset:

data(father.son)  
fit <- lm(sheight ~ fheight, data=father.son)  
coef(fit)[2]

## fheight   
## 0.514093

cov(father.son$sheight, father.son$fheight) / sd(father.son$fheight)^2

## [1] 0.514093

### Revisting Galton's data

Calculate the coefficients using child height as response and parent height as predictor:

y <- galton$child  
x <- galton$parent  
beta1 <- cor(y, x) \* sd(y) / sd(x)  
beta0 <- mean(y) - beta1 \* mean(x)  
rbind(c(beta0, beta1), coef(lm(y ~ x)))

## (Intercept) x  
## [1,] 23.94153 0.6462906  
## [2,] 23.94153 0.6462906

Now reverse the relationship:

beta1 <- cor(y, x) \* sd(x) / sd(y)  
beta0 <- mean(x) - beta1 \* mean(y)  
rbind(c(beta0, beta1), coef(lm(x ~ y)))

## (Intercept) y  
## [1,] 46.13535 0.3256475  
## [2,] 46.13535 0.3256475

Regression through the origin gives us the same slope; i.e., changing the intercept does not affect the slope:

yc <- y - mean(y)  
xc <- x - mean(x)  
beta1 <- sum(yc \* xc) / sum(xc ^ 2)  
c(beta1, coef(lm(y ~ x))[2])

## x   
## 0.6462906 0.6462906

But normalizing (standardizing) the data makes the slope equal to the correlation:

yn <- (y - mean(y))/sd(y)  
xn <- (x - mean(x))/sd(x)  
c(cor(y, x), cor(yn, xn), coef(lm(yn ~ xn))[2])

## xn   
## 0.4587624 0.4587624 0.4587624

### Exercises

1. Install and load the package UsingR and load the father.son dataset with data(father.son). Get the linear regression fit where the son's height is the outcome and the father's height is the predictor. Give the intercept and slope, plot the data and overly the fitted regression line.

data(father.son)  
x <- father.son$fheight  
y <- father.son$sheight  
fit <- lm(y ~ x)  
coef(fit)

## (Intercept) x   
## 33.886604 0.514093

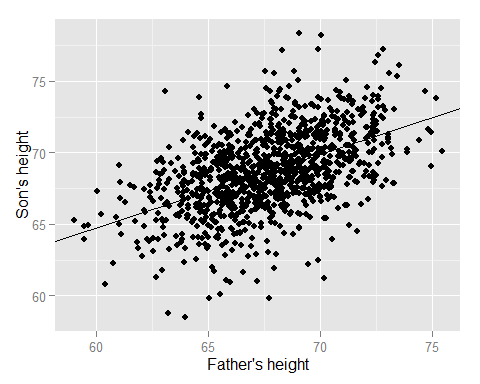
itc <- coef(fit)[1]  
slope <- coef(fit)[2]  
itc

## (Intercept)   
## 33.8866

slope

## x   
## 0.514093

ggplot(father.son, aes(x=fheight, y=sheight)) +  
 geom\_point() +  
 geom\_abline(intercept = itc, slope=slope) +  
 xlab("Father's height") +  
 ylab("Son's height")



1. Refer to problem 1. Center the father and son variables and refit the model omitting the intercept. Verify that the slope estimate is the same as the linear regression fit from prolem 1.

xc <- father.son$fheight - mean(father.son$fheight)  
yc <- father.son$sheight - mean(father.son$sheight)  
fitc <- lm(yc ~ xc)  
slopec <- coef(fitc)[2]  
slopec

## xc   
## 0.514093

round(slope, digits=6) == round(slopec, digits=6)

## x   
## TRUE

1. Refer to problem 1. Normalize the father and son data and see that the fitted slope is the correlation.

xn <- (father.son$fheight - mean(father.son$fheight)) / sd(father.son$fheight)  
yn <- (father.son$sheight - mean(father.son$sheight)) / sd(father.son$sheight)  
fitn <- lm (yn ~ xn)  
corrn <- cor(yn, xn)  
slopen <- coef(fitn)[2]  
corrn

## [1] 0.5013383

slopen

## xn   
## 0.5013383

round(corrn, digits=6) == round(slopen, digits=6)

## xn   
## TRUE

1. Go back to the linear regression line from problem 1. If a father's height was 63 inches, what would you predict the son's height to be?

predict(fit, newdata=data.frame(x=63))

## 1   
## 66.27447

1. Consider a dataset where the standard deviation of the outcome variable is double that of the predictor. Also, the correlation of the variables is 0.3. If you fit a linear regression model, what would be the estimate of the slope?

We know that:

If the correlation is 0.3, and the outcome variable's standard deviation is twice that of the predictor's, then the slope must be .

1. Consider the previous problem. The outcome variable has a mean of 1 and the predictor has a mean of 0.5. What would be the intercept?

We know that:

So if and , then .

1. True or false, if the predictor variable has mean 0, the estimated intercept from linear regression will be the mean of the outcome.

True. The intercept is:

So if , the entire right term is zero, and we are left only with .

1. Consider problem 5 again. What would be the estimated slope if the predictor and outcome were reversed?

This time we multiply the correlation, 0.3, by instead of , so the result is 0.15.

## Chapter 4. Regression to the mean

### A historically famous idea, regression to the mean

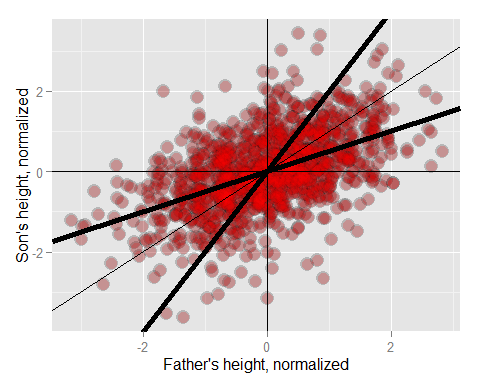
Galton discovered that children of tall parents tended to be tall, but not quite as tall, and children of short parents tended to be short, but not quite as short. They "regressed to the mean."

### Regression to the mean

Standardize both predictor and response so that we have regression through the mean, and then plot regression lines in both directions, i.e., child height is response, and then parent height is response. It shows that the slope of one is the multiplicative inverse of the other.

(Without standardizing, the slopes will not necessarily be multiplicative inverses of each other.)

data(father.son)  
y <- (father.son$sheight - mean(father.son$sheight)) / sd(father.son$sheight)  
x <- (father.son$fheight - mean(father.son$fheight)) / sd(father.son$fheight)  
rho <- cor(x, y)  
g = ggplot(data.frame(x, y), aes(x = x, y = y))  
g = g + geom\_point(size = 5, alpha = .2, colour = "black")  
g = g + geom\_point(size = 4, alpha = .2, colour = "red")  
g = g + geom\_vline(xintercept = 0)  
g = g + geom\_hline(yintercept = 0)  
g = g + geom\_abline(position = "identity")  
g = g + geom\_abline(intercept = 0, slope = rho, size = 2)  
g = g + geom\_abline(intercept = 0, slope = 1 / rho, size = 2)  
g = g + xlab("Father's height, normalized")  
g = g + ylab("Son's height, normalized")  
g



If you had to predict a son's normalized (standardized) height, it would be , where is the standardized father's height. If you had to predict a father's normalized height, it would be .

"Multiplication by this correlation shrinks toward 0 (regression toward the mean)".

### Exercises

1. You have two noisy scales and a bunch of people that you'd like to weigh. You weigh each person on both scales. The correlation was 0.75. If you normalized each set of weights, what would you have to multiply the weight on one scale to get a good estimate of the weight on the other scale?

If both variables have been normalized, then both have a standard deviation of 1. Now the slope of the regression line is therefore the correlation, or 0.75. Or 1/0.75 = 1.33.

library(UsingR)  
data(father.son)  
y <- (father.son$sheight - mean(father.son$sheight)) / sd(father.son$sheight)  
x <- (father.son$fheight - mean(father.son$fheight)) / sd(father.son$fheight)  
rho <- cor(x, y)  
rho

## [1] 0.5013383

cor(father.son$fheight, father.son$sheight)

## [1] 0.5013383

cor(father.son$sheight, father.son$fheight)

## [1] 0.5013383

1. Consider the previous problem. Someone's weight was 2 standard deviations above the mean of the group on the first scale. How many standard deviations above the mean would you estimate them to be on the second?

2 \* 0.75 = 1.5

1. You ask a collection of husbands and wives to guess how many jellybeans are in a jar. The correlation is 0.2. The standard deviation for the husbands is 10 beans while the standard deviation for wives is 8 beans. Assume that the data were centered so that 0 is the mean for each. The centered guess for a husband was 30 beans (above the mean). What would be your best estimate of the wife's guess?

The slope is . We are using regression through the origin so intercept is zero. Therefore the best estimate is the husband's guess times the slope, .

## Chapter 5. Statistical linear regression models

### Basic regression with additive Gaussian errors

The model:

The above is the "true" and unknown model; we cannot know the true values of , or . Instead, we estimate them.

Under this model:

Some notes on the above:

* is a strange notation that just means, "assuming that our X value is some specified value"
* does not mean the empirical mean; rather, it means merely an expected value
* The lack of hats on the betas again indicate that they are the true population parameters, not statistical estimates

Also:

All of the above could be written "more compactly":

But the above does not show "additive Guassian errors" and so is not as convenient.

NOTE: The above the standard deviation of the residuals, which is covered later. This can be confusing.

Remember the least squares estimate of the parameters:

And

### Interpreting regression coefficients, the intercept

When the predictor is zero, our estimate of the response is:

This is the intercept. But the intercept isn't always of interest; the height of a child for a parent with zero height makes no sense.

It is possible to "make your intercept more interpretable": by shifting all of the predictor values. We've seen an example where we have regression through the mean. This does not affect the slope, but now the intercept is significant.

The above is an abstract and head-scratching way of saying, "You can shift the predictor by any value, not just the mean" but in our example, would be .

### Interpreting regression coefficients, the slope

The above is a *very* confusing way of saying, "If our slope is , then for every unit we move to the right on the x-axis, our predicted value for goes up by the slope, ."

#### Changing units

The above is a way of saying that you multiply by the inverse of the change in units. For example, if your predictor, , is in units of kilograms, and you want to change to units of grams, you multiply the predictor values by .

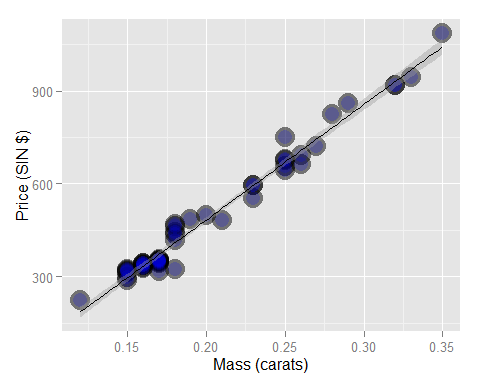
### Using regression for prediction

"If we would like to guess the outcome of a particular value of the predictor, say , the regression model guesses:""

(The use of the word "guess" above is extremely ill-advised.)

### Example

data(diamond)  
g = ggplot(diamond, aes(x = carat, y = price))  
g = g + xlab("Mass (carats)")  
g = g + ylab("Price (SIN $)")  
g = g + geom\_point(size = 7, colour = "black", alpha=0.5)  
g = g + geom\_point(size = 5, colour = "blue", alpha=0.2)  
g = g + geom\_smooth(method = "lm", colour = "black")  
g



Now fit the model:

fit <- lm(price ~ carat, data=diamond)  
coef(fit)

## (Intercept) carat   
## -259.6259 3721.0249

Re-fit, after centering the data:

fit2 <- lm(price ~ I(carat - mean(carat)), data=diamond)  
coef(fit2)

## (Intercept) I(carat - mean(carat))   
## 500.0833 3721.0249

The new intercept shows us the predicted cost of a diamond whose weight is the mean of all observed weights.

Now re-fit again, this time changing the scale of the predictor from carats to tenth of carats:

fit3 <- lm(price ~ I(carat \* 10), data=diamond)  
coef(fit3)

## (Intercept) I(carat \* 10)   
## -259.6259 372.1025

Now the increase of price per unit (tenth of carat) is one-tenth of what it was when unit was carat.

Now predict the prices of some diamonds based on weight.

newx <- c(0.16, 0.27, 0.34)  
coef(fit)[1] + coef(fit)[2] \* newx

## [1] 335.7381 745.0508 1005.5225

We could also use R's predict function.

predict(fit, newdata=data.frame(carat=newx))

## 1 2 3   
## 335.7381 745.0508 1005.5225

### Exercises

1. Fit a linear model to the father.son dataset with the father as the predictor and the son as the outcome. Give a p-value for the slope coefficient and perform the relevant hypothesis test.

fit <- lm(sheight ~ fheight, data=father.son)  
summary(fit)$coefficients[2,4]

## [1] 1.121268e-69

The p-value is only a tiny fraction of 0.05, so using a 95% confidence level, we do reject the null hypothesis that , and conclude that the slope is non-zero, and there is some correlation between father's height and son's height.

1. Refer to question 1. Interpret both parameters. Recenter the intercept if necessary.

The intercept is 33.88, which indicates that a father with height 0 has a son with height 33.88. This clearly is invalid and is the result of interpolation.

The slope of 0.514 indicates that, for every additional 1" of height in the father, we predict that the son will have an additional 0.514" of height.

1. Refer to question 1. Predict the son's height if the father's height is 80 inches. Would you recommend this prediction? Why or why not?

predict(fit, newdata=data.frame(fheight=80))

## 1   
## 75.01405

I would not, because the maximum observed father's height is about 75, and so we are interpolating.

1. Load the mtcars dataset. Fit a linear regression with miles per gallon as the outcome and horsepower as the predictor. Interpret your coefficients, recenter for the intercept if necessary.

data(mtcars)  
fit <- lm(mpg ~ hp, data=mtcars)  
coef(fit)

## (Intercept) hp   
## 30.09886054 -0.06822828

The intercept indicates that a vehicle with 0 horsepower gets 30 mpg, which is clearly nonsense. The slope indicates that for every decrease in horsepower by -0.068, mileage increases by one.

Recenter the data:

fit <- lm(I(mpg - mean(mpg)) ~ I(hp - mean(hp)), data=mtcars)  
coef(fit)

## (Intercept) I(hp - mean(hp))   
## 9.440147e-16 -6.822828e-02

The slope is unchanged, but now the intercept is zero.

The video answer indicates that only the predictor should be re-centered.

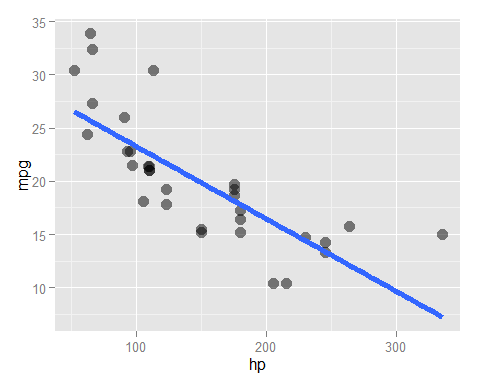
fit <- lm(mpg ~ I(hp - mean(hp)), data=mtcars)  
summary(fit)$coef

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 20.09062500 0.6828817 29.420360 1.101810e-23  
## I(hp - mean(hp)) -0.06822828 0.0101193 -6.742389 1.787835e-07

This shows that the car with mean hp gets about 20 mpg. We also see that the slope is highly statistically significant.

1. Refer to question 4. Overlay the fit onto a scatterplot.

fit <- lm(mpg ~ hp, data=mtcars)  
itc <- fit$coefficients[1]  
slope <- fit$coefficients[2]  
ggplot(mtcars, aes(x=hp, y=mpg)) +  
 geom\_point(cex=4, alpha=0.5) +  
 geom\_smooth(method=lm, se=FALSE, lwd=2)



1. Refer to question 4. Test the hypothesis of no linear relationship between horespower and miles per gallon.

summary(fit)$coef[2,4]

## [1] 1.787835e-07

The p-value for the slope is very low, indicating that there is reason to reject the null hypothesis and conclude that there is a relationship between horsepower and miles per gallon.

1. Refer to question 4. Predict the miles per gallon for a horsepower of 111.

min(mtcars$hp)

## [1] 52

max(mtcars$hp)

## [1] 335

predict(fit, newdata=data.frame(hp=111))

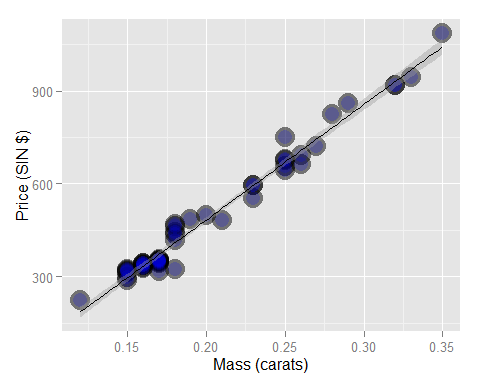
## 1   
## 22.52552

## Chapter 6. Residuals

### Residual variation

Residuals represent variation unexplained by the model. Here again is the diamond data, with price as response and carat as predictor/regressor.

g = ggplot(diamond, aes(x = carat, y = price))  
g = g + xlab("Mass (carats)")  
g = g + ylab("Price (SIN $)")  
g = g + geom\_point(size = 7, colour = "black", alpha=0.5)  
g = g + geom\_point(size = 5, colour = "blue", alpha=0.2)  
g = g + geom\_smooth(method = "lm", colour = "black")  
g

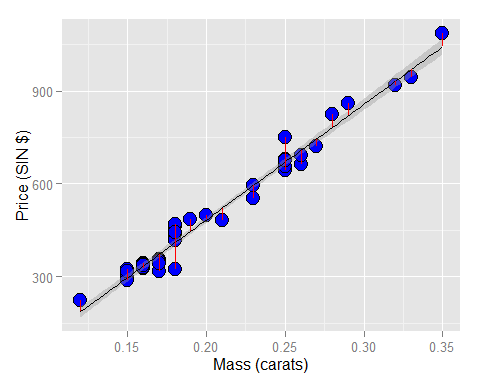


Recall the linear model:

where we assume that . (The here represents residual variance.) The observed outcome corresponds to a predictor . We can label our predicted outcome as . This will fall directly on our model's regression line:

The residual is the difference between the observed value and the predicted value:

diamond.residuals <- residuals(lm(price ~ carat, data=diamond))  
g = ggplot(diamond, aes(x = carat, y = price))  
g = g + xlab("Mass (carats)")  
g = g + ylab("Price (SIN $)")  
g = g + geom\_point(size = 5, fill="blue", color="black", shape=21)  
g = g + geom\_smooth(method = "lm", colour = "black")  
for (x in 1:length(diamond$carat)) {  
 g <- g + geom\_line(data=data.frame(carat=rep(diamond$carat[x], 2), price=c(diamond$price[x], diamond$price[x] - diamond.residuals[x])), color="red")  
}  
g



#g = g + geom\_line(data=data.frame(carat=rep(diamond$carat, each=2), price=as.vector(rbind(diamond$price, diamond$price + diamond.residuals))))

Least squares minimizes the sum of the squared residuals, . Note that the residuals, , are observable, while the errors, , are not. The residuals can be thought of as estimates of the errors.

### Properties of the residuals

#### Expected value

The expected value of a residual is 0:

Confusing statement from the text:

"If an intercept is included, "

Translation: If *only* an intercept is included, the sum of the residuals is zero. This means that the regression line is horizontal. If that is the case, then we can simply add up the distance (positive or negative) of each residual from the horizontal intercept line, and the sum will be zero.

If both an intercept and a slope is given, then:

Now, in other words, we must multiply each residual value (positive or negative) with the corresponding X value, and when we sum them all, the result is zero.

Note that there are degrees of freedom, where is the number of coefficients (including the intercept) in the model. So for simple linear regression it is .

#### Example

data(diamond)  
y <- diamond$price; x <- diamond$carat; n <- length(y)  
fit <- lm(y ~ x)  
## The easiest way to get the residuals  
e <- resid(fit)  
## Obtain the residuals manually, get the predicted Ys first  
yhat <- predict(fit)  
## The residuals are y - yhat. Let's check by comparing this  
## with R's build in resid function  
max(abs(e -(y - yhat)))

## [1] 9.485746e-13

## Let's do it again hard coding the calculation of Yhat  
max(abs(e - (y - coef(fit)[1] - coef(fit)[2] \* x)))

## [1] 9.485746e-13

### Estimating residual variation

The variance of the residuals (for simple linear regression):

#### Diamond example

y <- diamond$price; x <- diamond$carat; n <- length(y)  
fit <- lm(y ~ x)  
## the estimate from lm  
summary(fit)$sigma

## [1] 31.84052

## directly calculating from the residuals  
sqrt(sum(residuals(fit)^2) / (n - 2))

## [1] 31.84052

### Summarizing variation

"The total variation in the response is the variability around an intercept. This is also the variance estimate from a model with only an intercept."

The above is classic Caffo. I cannot fathom what point he's trying to make. I'm sure it's fairly simple, and that this is another classic example of how he can obscure a simple concept behind smoke and mirrors. Anyway:

The above says that total variability is based on the squared differences between the observed values and the mean of all observed values. If the model has only an intercept and no slope, then there would be no decomposition of the variability (I think this is the point, anyway).

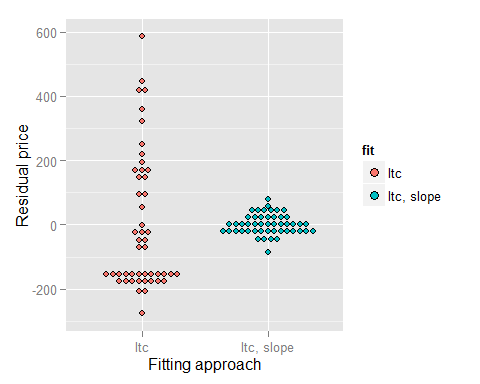
Next:

Total variability:

The above takes a little pondering, at last the rightmost term. The middle term is the regression variability; the sum of the squared differences between what we observed, and what we predicted. The last term is the unexplained variability; it's any variation between our predictions, and the mean of the observed values. Need to internalize this so that it's second nature.

The following shows variability around the intercept only (left) and both the intercept and the slope (right). Variation around the intercept only is total variation. Variation around both the intercept and the slope is variation that is not explained by the model. So the difference in the two is the amount of variation explained by the model.

e = c(resid(lm(price ~ 1, data = diamond)),  
resid(lm(price ~ carat, data = diamond)))  
fit = factor(c(rep("Itc", nrow(diamond)),  
rep("Itc, slope", nrow(diamond))))  
g = ggplot(data.frame(e = e, fit = fit), aes(y = e, x = fit, fill = fit))  
g = g + geom\_dotplot(binaxis = "y", size = 2, stackdir = "center", binwidth = 20)  
g = g + xlab("Fitting approach")  
g = g + ylab("Residual price")  
g



### R squared

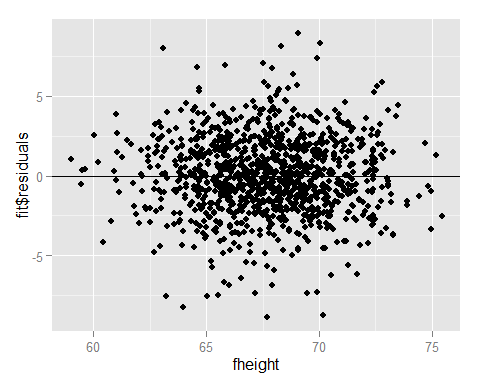
**R squared** is the percentage of the total variability that is explained by the model (by the linear relationship of the response variable with the predictor).

* is the percentage of the variation explained by the regression model
* is the sample correlation squared
* can be a misleading indicator of model fit
* Deleting data can inflate it
* (For later.) Adding terms to regression model always increases .

### Exercises

1. Fit a linear regression model to the father.son dataset with the father as the predictor and the son as the outcome. Plot the son's height (horizontal axis) versus the residuals (vertial axis).

fit <- lm(sheight ~ fheight, data=father.son)  
ggplot(father.son, aes(x=fheight, y=fit$residuals)) +  
 geom\_point() +  
 geom\_hline(yintercept=0)



Note that the text says to plot the *son's* height on the x-axis, but the video shows plotting the *father's* height on the x-axis.

1. Refer to question 1. Directly estimate the residual variance and compare the estimate to the output of lm.

predictions <- predict(fit)  
estvar <- (1/(nrow(father.son) - 2)) \* sum((father.son$sheight - predictions)^2)  
estvar

## [1] 5.936804

summary(fit)$sigma^2

## [1] 5.936804

#Note: the following won't work. Reason is that it expects degrees of freedom to be n-1  
#var(fit$residuals)

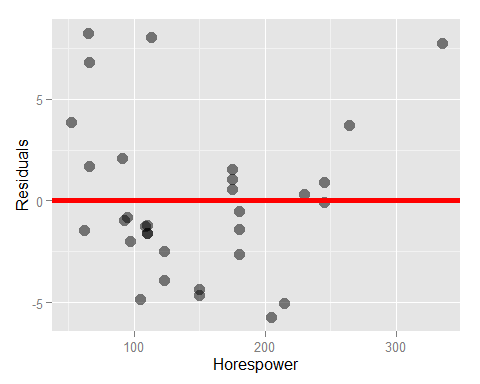
1. Refer to question 1. Give the R squared for this model.

summary(fit)$r.squared

## [1] 0.2513401

1. Load the mtcars dataset. Fit a linear regression with miles per gallon as the outcome and horespower as the predictor. Plot horsepower versus the residuals.

fit <- lm(mpg ~ hp, data=mtcars)  
ggplot(mtcars, aes(x=hp, y=fit$residuals)) +  
 geom\_point(cex=4, alpha=0.5) +  
 geom\_hline(yintercept=0, lwd=2, color="red") +  
 xlab("Horespower") +  
 ylab("Residuals")



1. Refer to question 4. Directly estimate the residual variance and compare this estimate to the output of lm.

sum(fit$residuals^2) / (nrow(mtcars) - 2)

## [1] 14.92248

summary(fit)$sigma^2

## [1] 14.92248

1. Refer to question 4. Give the R squared for this model.

summary(fit)$r.squared

## [1] 0.6024373

## Chapter 7. Regression Inference

### Reminder of the model

The regression model so far:

where .

(Note that some notation for distributions shows variance as the second term, as here; others show standard deviation, as in the DASI textboook.)

The estimates for the model parameters are:

### Review

Consider statistics like the following:

where is an estimate of interest, is an estimand (the thing whose true value we cannnot know, but are estimating), and is the standard error of .

This is reminiscent of determining the Z-score for a hypothesis test:

Such a statistic has these properties:

1. They are normally distributed and have a finite sample Student's T distribution under normality assumptions.
2. They can be used to test versus .
3. They can be used to create a confidence interval for via

Point 1 is somewhat confusing, but what I believe it means is that the sampling distribution is to be expected to be distributed under the Student's distribution. So if you have a collection of samples, and you calculate the above statistic for each, they follow the Student's distribution for the appropriate degrees of freedom, based on sample size.

Point 2 uses as the point estimate and as the null value.

Point 3 is straightforward for me. is the significance level, which is 1 minus the confidence level. It is divided in half because we're assuming that a confidence interval is always two-sided, so we want half of the significance level of each side of the distribution.

### Results for the regression parameters

**NOTE**: The book begins by saying "First, we need the standard errors for our regression parameters" and then goes on to define the *variance* for and . These are *not* the standard errors. This is an egregious, unpardonable error; another example of why I was so irritated by this course in the first attempt.

~~Standard errors~~ variances for regression parameters:

And

Where is the variance of the residuals:

(Note that this is coefficients(fit)$sigma where fit is the output of lm.)

### Example diamond data set

Calculate by hand:

y <- diamond$price; x <- diamond$carat; n <- length(y)  
beta1 <- cor(y, x) \* sd(y) / sd(x)  
beta0 <- mean(y) - beta1 \* mean(x)  
e <- y - beta0 - beta1 \* x  
sigma <- sqrt(sum(e^2) / (n-2))  
ssx <- sum((x - mean(x))^2)

Now calculate standard errors:

seBeta0 <- (1 / n + mean(x) ^ 2 / ssx) ^ .5 \* sigma  
seBeta1 <- sigma / sqrt(ssx)  
tBeta0 <- beta0 / seBeta0  
tBeta1 <- beta1 / seBeta1

tBeta0 and tBeta1 above correspond to "t-scores" (using a Student's distribution) for the two parameters.

Now obtain the p-values:

pBeta0 <- 2 \* pt(abs(tBeta0), df = n - 2, lower.tail = FALSE)  
pBeta1 <- 2 \* pt(abs(tBeta1), df = n - 2, lower.tail = FALSE)  
coefTable <- rbind(c(beta0, seBeta0, tBeta0, pBeta0), c(beta1, seBeta1, tBeta1, pBeta1))  
colnames(coefTable) <- c("Estimate", "Std. Error", "t value", "P(>|t|)")  
rownames(coefTable) <- c("(Intercept)", "x")  
coefTable

## Estimate Std. Error t value P(>|t|)  
## (Intercept) -259.6259 17.31886 -14.99094 2.523271e-19  
## x 3721.0249 81.78588 45.49715 6.751260e-40

Now do all of this the fast, simple way:

fit <- lm(y ~ x)  
summary(fit)$coefficients

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -259.6259 17.31886 -14.99094 2.523271e-19  
## x 3721.0249 81.78588 45.49715 6.751260e-40

### Getting a confidence interval

For the intercept:

sumCoef <- summary(fit)$coefficients  
sumCoef[1,1] + c(-1, 1) \* qt(.975, df = fit$df) \* sumCoef[1, 2]

## [1] -294.4870 -224.7649

And for the slope:

(sumCoef[2,1] + c(-1, 1) \* qt(.975, df = fit$df) \* sumCoef[2, 2]) / 10

## [1] 355.6398 388.5651

### Prediction of outcomes

We've already covered that the estimate for prediction at point is:

We need a standard error to create a prediction interval. **There is an important but subtle distinction between intervals for the regression line at point and the prediction of what would be at point .** What differes is the standard error.

For the line at the standard error is:

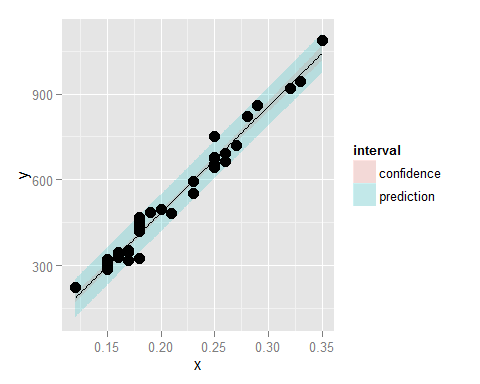
For the prediction interval at the standard error is:

"Notice that the prediction interval standard error is a little large than error for a line. Think of it this way. If we want to predict a Y value at a particular X value, and we knew the actual true slope and intercept, there would still be error. However, if we only wanted to predict the value at the line at that X value, there would be no variance, since we already know the line.

Thus, the variation for the line only considers how hard it is to estimate the regression line at that X value. The prediction interval includes that variation, as well as the extra variation unexplained by the relationship between Y and X. So, it has to be a little wider."

In R, using ggplot:

newx = data.frame(x = seq(min(x), max(x), length = 100))  
p1 = data.frame(predict(fit, newdata= newx,interval = ("confidence")))  
p2 = data.frame(predict(fit, newdata = newx,interval = ("prediction")))  
p1$interval = "confidence"  
p2$interval = "prediction"  
p1$x = newx$x  
p2$x = newx$x  
dat = rbind(p1, p2)  
names(dat)[1] = "y"  
g = ggplot(dat, aes(x = x, y = y))  
g = g + geom\_ribbon(aes(ymin = lwr, ymax = upr, fill = interval), alpha = 0.2)  
g = g + geom\_line()  
g = g + geom\_point(data = data.frame(x = x, y=y), aes(x = x, y = y), size = 4)  
g



### Exercises

1. Test whether the slope coefficient for the father.son dataset is different from zero (father as predictor, son as outcome).

fit <- lm(sheight ~ fheight, data=father.son)  
summary(fit)$coefficients[2,4]

## [1] 1.121268e-69

Value is very small, so we conclude that we reject the null hypothesis that the slope is zero.

1. Refer to question 1. Form a confidence interval for the slope coefficient.

n <- nrow(father.son)  
z.star <- pt(0.975, lower.tail=FALSE, df=n-2)  
se <- summary(fit)$coefficients[2,2]  
point.est <- summary(fit)$coefficients[2,1]  
ci <- point.est + c(-1,1) \* point.est \* se  
ci

## [1] 0.5001875 0.5279986

1. Refer to question 1. Form a confidence interval for the intercept (center the fathers' heights first to get an intercept that is easier to interpret).

n <- nrow(father.son)  
x <- father.son$fheight - mean(father.son$fheight)  
y <- father.son$sheight  
fit <- lm(y ~ x)  
z.star <- pt(0.975, lower.tail=FALSE, df=n-2)  
point.est <- summary(fit)$coefficients[1,1]  
se <- summary(fit)$coefficients[1,2]  
ci <- point.est + c(-1, 1) \* point.est \* se  
ci

## [1] 63.58697 73.78117

1. Refer to question 1. Form a mean value interval for the expected son's height at the average father's height.

n <- nrow(father.son)  
fit <- lm(sheight ~ fheight, data=father.son)  
z.star <- pt(0.975, lower.tail=FALSE, df=n-2)  
predict(fit, newdata=data.frame(fheight=mean(father.son$fheight)), interval="confidence")

## fit lwr upr  
## 1 68.68407 68.53846 68.82968

1. Refer to question 1. Form a prediction interval for the expected son's height at the average father's height.

n <- nrow(father.son)  
fit <- lm(sheight ~ fheight, data=father.son)  
z.star <- pt(0.975, lower.tail=FALSE, df=n-2)  
predict(fit, newdata=data.frame(fheight=mean(father.son$fheight)), interval="predict")

## fit lwr upr  
## 1 68.68407 63.90091 73.46723

1. Load the mtcars dataset. Fit a linear regression with miles per gallon as the outcome and horespower as the predictor. Test whether or not the horsepower coefficient is statistically different from zero. Interpret your test.

data(mtcars)  
fit <- lm(mpg ~ hp, data=mtcars)  
summary(fit)$coefficients[2,4]

## [1] 1.787835e-07

The p-value is very small so we conclude that there is a relationship between horsepower and miles per gallon.

1. Refer to question 6. Form a confidence interval for the slope coefficient.

fit <- lm(mpg ~ hp, data=mtcars)  
z.star <- pt(0.975, lower.tail=FALSE, df=nrow(mtcars) - 2)  
point.est <- summary(fit)$coefficients[2,1]  
se <- summary(fit)$coefficients[2,2]  
ci <- point.est + c(-1,1) \* z.star \* se  
ci

## [1] -0.06993519 -0.06652137

1. Refer to question 6. Form a confidence interval for the intercept (center the hp variable first).

y <- mtcars$mpg  
x <- mtcars$hp - mean(mtcars$hp)  
fit <- lm(y ~ x)  
point.est <- summary(fit)$coefficients[1,1]  
se <- summary(fit)$coefficients[1,2]  
ci <- point.est + c(-1,1) \* z.star \* se  
ci

## [1] 19.97544 20.20581

For the car with average horsepower, we are 95% confident that mpg is between 19.98 and 20.21.

1. Refer to question 6. Form a mean value interval for the expected MPG for the average HP.

fit <- lm(mpg ~ hp, data=mtcars)  
predict(fit, newdata=data.frame(hp=mean(mtcars$hp)), interval="confidence")

## fit lwr upr  
## 1 20.09062 18.69599 21.48526

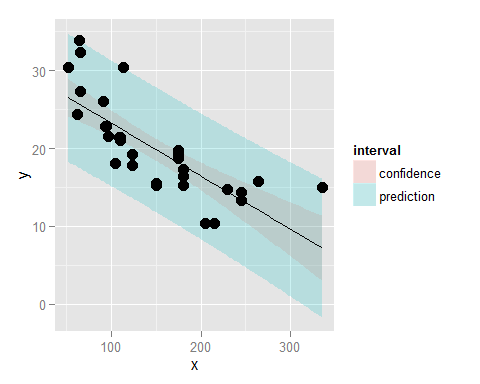
1. Refer to question 6. Form a prediction interval for the expected MPG for the average HP.

fit <- lm(mpg ~ hp, data=mtcars)  
predict(fit, newdata=data.frame(hp=mean(mtcars$hp)), interval="predict")

## fit lwr upr  
## 1 20.09062 12.07908 28.10217

1. Refer to question 6. Create a plot that has the fitted regression line plus the curves at the expected value and prediction intervals.

x <- mtcars$hp  
y <- mtcars$mpg  
fit <- lm(y ~ x)  
newx = data.frame(x = seq(min(x), max(x), length = 100))  
p1 = data.frame(predict(fit, newdata= newx,interval = ("confidence")))  
p2 = data.frame(predict(fit, newdata = newx,interval = ("prediction")))  
p1$interval = "confidence"  
p2$interval = "prediction"  
p1$x = newx$x  
p2$x = newx$x  
dat = rbind(p1, p2)  
names(dat)[1] = "y"  
g = ggplot(dat, aes(x = x, y = y))  
g = g + geom\_ribbon(aes(ymin = lwr, ymax = upr, fill = interval), alpha = 0.2)  
g = g + geom\_line()  
g = g + geom\_point(data = data.frame(x = x, y=y), aes(x = x, y = y), size = 4)  
g



## Chapter 8. Multivariable Regression Analysis

#### Multivariable regression analyses: adjustment

A multivariable model may contain invalid predictors. Adjustment is the process of removing these invalid predictors.

#### Multivariable regression analyses: prediction

The goal is to generalize simple linear regression to work with multiple predictors. We must be careful to include all predictors that are significant, and exclude all that are not.

### The linear model

(**NOTE**: Textbook shows in the rightmost term, which can't be right. Careless error.)

Here, is typically 1 (note: when would it not be?!?), so that an intercept is included. Least squares minimizes:

(Again, the textbook shows . The accompanying video shows it but says nothing specific about it.)

Important note: the linearity in linear regression is in the fact that the coefficients are constant for all observations. The observations themselves may be raised to exponents, or log transformmed, etc., but it is still linear regression.

### Estimation

Recall the least squares estimate for regression through the origin, , is . Consider two regressors, = \_i$. Least squares tries to minimize:

Estimate for slope:

The above gave me great puzzlement but in effect, what it says is that that the slope for the first regressor is the variability in with , once we've regressed out , divided by the squared variability of , once we've regressed out .

**NOTE** the worst error in the textbook so far:

"where is the residual having fit on and is the residual having fit ~~on Y~~ **on X1**". I think this textbook was largely thrown together in a weekend and not even the most cursory attempt was made to proofread it, much less to get a professional editorial eye on it. **Terrible**. I didn't pay for it, and I don't intend too.

### Example with two variables, simple linear regression

Linear regression model:

where is an intercept term (so here is really comparable to what we have previously shown as , with no value).

The fitted coefficient of therefore is . (This is another way of saying that the intercept, or the old , is the mean of .) The residuals are .

The estimate for therefore is:

Which agrees with our earlier finding.

### The general case

In the general case with regressors (**NOTE**: this includes the intercept!), least squares have to minimize:

### Simulation

n <- 100; x <- rnorm(n); x2 <- rnorm(n); x3 <- rnorm(n)  
# Generate the data  
y <- 1 + x + x2 + x3 + rnorm(n, sd=0.1)  
# Get the residuals having removed X2 and X3 from X1 and Y  
ey <- resid(lm(y ~ x2 + x3))  
ex <- resid(lm(x ~ x2 + x3))  
# Fit the regression through the origin with the residuals  
sum(ey \* ex) / sum(ex^2)

## [1] 0.9998097

# Double check with lm  
coef(lm(ey ~ ex - 1))

## ex   
## 0.9998097

# Fit the full linear model to show it agrees  
coef(lm(y ~ x + x2 + x3))

## (Intercept) x x2 x3   
## 1.0138697 0.9998097 1.0023384 1.0010667

### Interpretation of the coefficients

Predicted mean for a given set of values of the regressors:

Now consider incrementing and only by 1.

Now subtract the two equations:

**NOTE**: The textbook shows the two sums being **added** rather than **subtracted**. And the video takes no notice of this.

"Thus, the interpretation of a multivariate regression coefficient is the expected change in the response per unit change in the regressor, holding all of the other regressors fixed."

### Fitted values, residuals and residual variation

All SLR quantities can be extended to (multiple) linear models. Our statistical model:

where $$\epislon \tilde{} N(0, \sigma^2)$. Our fitted responses are:

Residuals are defined exactly as in SLR:

Variance estimate:

**NOTE** that this is the residual variance.

To get predicted responses at at new values, , simply plug them into the linear model .

Coefficients have standard errors, , and:

follows a t distribution with degrees of freedom. Predicted responses have standard errors and we can calculate predicted and expected response intervals.

(I think "predicted" above means interval="predict" in the R predict function and "expected" means interval="confidence".)

### Summary notes on linear models

* Linear models are by far the single most important applied statistical and ML learning technique
* Some amazing things you can accomplish with linear models
* Decompose a signal into its harmonics
* Flexibly fit complicated functions
* Fit factor variables as vectors
* Uncover complex multivariate relationships with the response
* Build accurate prediction models

### Exercises

1. Load the dataset Seatbelts as part of the datasets package via data(Seatbelts). Use as.data.frame to convert the object to a dataframe. Fit a linear model of driver deaths with kms and PetrolPrice as predictors. Interpret your results.

data(Seatbelts)  
sbelts <- as.data.frame(Seatbelts)  
fit <- lm(DriversKilled ~ kms + PetrolPrice, data=Seatbelts)  
summary(fit)

##   
## Call:  
## lm(formula = DriversKilled ~ kms + PetrolPrice, data = Seatbelts)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -51.06 -17.77 -4.15 15.67 59.33   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.157e+02 1.467e+01 14.711 < 2e-16 \*\*\*  
## kms -1.749e-03 6.145e-04 -2.847 0.0049 \*\*   
## PetrolPrice -6.438e+02 1.483e+02 -4.341 2.3e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23.04 on 189 degrees of freedom  
## Multiple R-squared: 0.1844, Adjusted R-squared: 0.1758   
## F-statistic: 21.37 on 2 and 189 DF, p-value: 4.292e-09

The results show that the number of drivers killed has significant negative linear relationships with both kms and PetroPrice. Neither seem to make sense. The more distance driven, the more we would expect drivers killed to increase. We would not expect petrol price to have any relationship with drivers killed. The help for the data points out that the point of the study was to measure deaths and injuries before and after a law was enacted requiring the use of seat belts. Also, kms may be correlated to other things, for example, a low number might indicate many shorter trips as opposed to fewer long ones.

Further exploration of the data is needed.

1. Predict the number of driver deaths at the average kms and petrol levels.

predict(fit, newdata=data.frame(kms=mean(sbelts$kms), PetrolPrice=mean(sbelts$PetrolPrice)))

## 1   
## 122.8021

1. Take the residuals for DriversKilled having regressed out kms and an intercept, and the residual for PetrolPrice having regressed out kms and an intercept. Fit a regression through the origin of the two residuals and show that it is the same as your coefficient obtained in question 1.

fitFull <- lm(DriversKilled ~ kms + PetrolPrice, data=Seatbelts)  
fit1 <- lm(DriversKilled ~ kms, data=Seatbelts)  
fit2 <- lm(PetrolPrice ~ kms, data=Seatbelts)  
resid1 <- fit1$residuals - mean(fit1$residuals)  
resid2 <- fit2$residuals - mean(fit2$residuals)  
fit3 <- lm(resid1 ~ resid2 -1)  
fitFull$coef

## (Intercept) kms PetrolPrice   
## 2.157461e+02 -1.749546e-03 -6.437895e+02

fit3$coef

## resid2   
## -643.7895

**NOTE** I found this confusing. The terminology indicated that, when I fit a model for DriversKilled with kms regressed out, I should not have an intercept **at that point**, and same for fitting PetrolPrice with kms regressed out. I'm not sure now what it means by "and an intercept." The "-1" in the model for fit3 was simply to remove the intercept; it had no effect on the slope coefficient. So why did it not work if I put "-1" in fit1 and fit2? I don't know yet.

1. Take the residual for DriversKilled having regressed out PetrolPrice and an intercept. Take the residual for kms having regressed out PetrolPrice and an intercept. Fit a regression through the origin of the two residuals and show that it is the same as your coefficient obtained in question 1.

fitFull <- lm(DriversKilled ~ kms + PetrolPrice, data=Seatbelts)  
fit1 <- lm(DriversKilled ~ PetrolPrice, data=Seatbelts)  
fit2 <- lm(kms ~ PetrolPrice, data=Seatbelts)  
resid1 <- fit1$residuals - mean(fit1$residuals)  
resid2 <- fit2$residuals - mean(fit2$residuals)  
fit3 <- lm(resid1 ~ resid2 -1)  
fitFull$coef

## (Intercept) kms PetrolPrice   
## 2.157461e+02 -1.749546e-03 -6.437895e+02

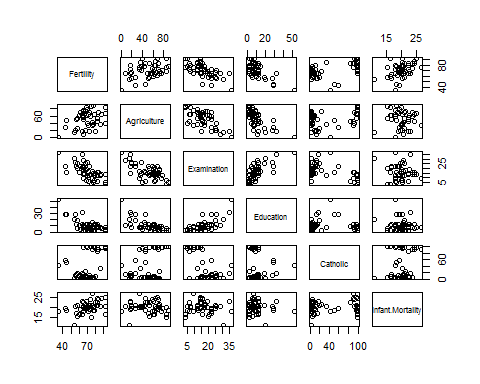
fit3$coef

## resid2   
## -0.001749546

## Chapter 9. Multivariable examples and tricks

### Data set for discussion

require(datasets)  
data(swiss)  
plot(swiss)



summary(lm(Fertility ~ ., data=swiss))

##   
## Call:  
## lm(formula = Fertility ~ ., data = swiss)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.2743 -5.2617 0.5032 4.1198 15.3213   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 66.91518 10.70604 6.250 1.91e-07 \*\*\*  
## Agriculture -0.17211 0.07030 -2.448 0.01873 \*   
## Examination -0.25801 0.25388 -1.016 0.31546   
## Education -0.87094 0.18303 -4.758 2.43e-05 \*\*\*  
## Catholic 0.10412 0.03526 2.953 0.00519 \*\*   
## Infant.Mortality 1.07705 0.38172 2.822 0.00734 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 7.165 on 41 degrees of freedom  
## Multiple R-squared: 0.7067, Adjusted R-squared: 0.671   
## F-statistic: 19.76 on 5 and 41 DF, p-value: 5.594e-10

summary(lm(Fertility ~ Agriculture, data=swiss))

##   
## Call:  
## lm(formula = Fertility ~ Agriculture, data = swiss)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -25.5374 -7.8685 -0.6362 9.0464 24.4858   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 60.30438 4.25126 14.185 <2e-16 \*\*\*  
## Agriculture 0.19420 0.07671 2.532 0.0149 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 11.82 on 45 degrees of freedom  
## Multiple R-squared: 0.1247, Adjusted R-squared: 0.1052   
## F-statistic: 6.409 on 1 and 45 DF, p-value: 0.01492

Key observation so far: The sign of the slope coefficient for agriculture was reversed from the full model to the model examining agriculture alone. This is known as "Simpson's Paradox." It points out that unadjusted and adjusted effects can have opposite signs.

### Simulation study

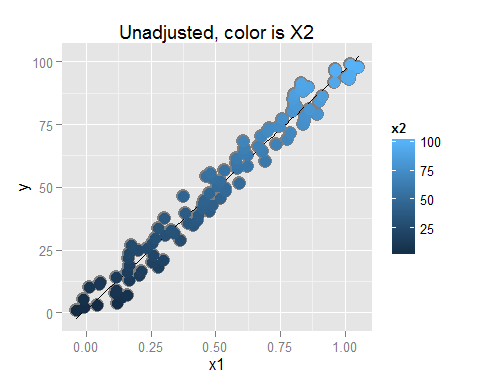
set.seed(1)  
n <- 100  
x2 <- 1:n  
x1 <- 0.01 \* x2 + runif(n=n, min=-0.1, max=0.1)  
y <- -x1 + x2 + rnorm(n, sd=0.1)  
summary(lm(y~x1))$coefficients

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 1.309868 1.057401 1.238761 2.183927e-01  
## x1 95.719096 1.801105 53.144650 4.588238e-74

summary(lm(y ~ x1 + x2))$coefficients

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 0.03326882 0.018613826 1.787317 7.700944e-02  
## x1 -0.84550277 0.173355326 -4.877282 4.209188e-06  
## x2 0.99775051 0.001761422 566.445902 1.596588e-172

dat = data.frame(y = y, x1 = x1, x2 = x2, ey = resid(lm(y ~ x2)), ex1 = resid(lm(x1 ~ x2)))  
library(ggplot2)  
par(mfrow=c(1,2))  
g = ggplot(dat, aes(y = y, x = x1, colour = x2))  
g = g + geom\_point(colour="grey50", size = 5) + geom\_smooth(method = lm, se = FALSE, colour = "black")   
g = g + geom\_point(size = 4)   
g = g + ggtitle("Unadjusted, color is X2")  
g = g + xlab("x1")  
g = g + ylab("y")  
g



g2 = ggplot(dat, aes(y = ey, x = ex1, colour = x2))   
g2 = g2 + geom\_point(colour="grey50", size = 5) + geom\_smooth(method = lm, se = FALSE, colour = "black") + geom\_point(size = 4)   
g2 = g2 + ggtitle("Adjusted, color is X2")  
g2 = g2 + xlab("resid(lm(x1 ~ x2))")  
g2 = g2 + ylab("resid(lm(y ~ x2))")  
g2

