Applied Simple Linear Regression

Devere Anthony Weaver

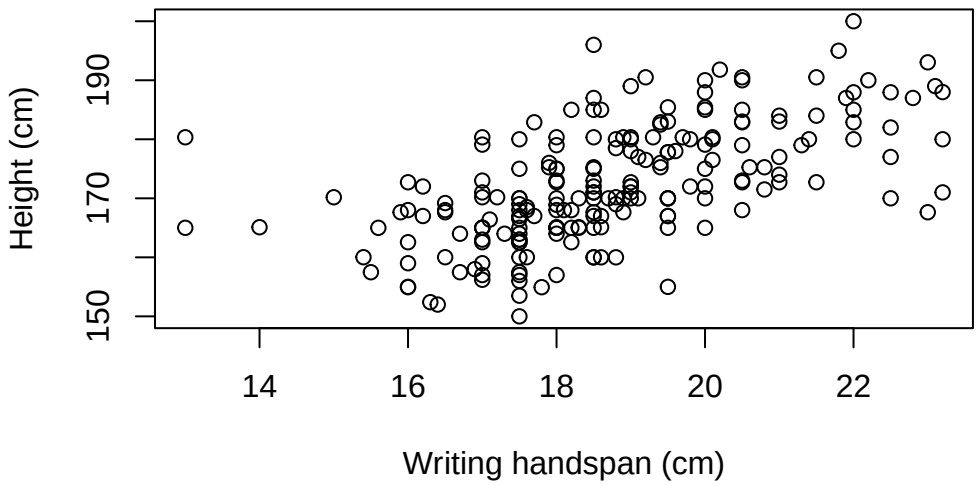
Note: This document doesn't contain much theory and instead contains mostly implementation details of linear regression modeling in R. This is deliberate since the theory is covered in-depth and much better in other resources such as "Applied Linear Statistical Models" by Kuter et al.

An Example of a Linear Relationship

```
library("MASS")
```

Using the common student survey data, we'll investigate the relationship between student heights (response) and their handspans of their writing hands (explanatory).

```
plot(survey$Height ~ survey$Wr.Hnd, xlab = "Writing handspan (cm)",
     ylab = "Height (cm)")
```



Graphically, there appears to be a positive linear relationship between the variables. We can (and probably should) also quantify this relationship numerically.

```
cor(x = survey$Wr.Hnd, y = survey$Height, use = "complete.obs")
```

```
[1] 0.6009909
```

By default, R will remove "incomplete" pairs from the plots. We should also try to figure out how many of the observations are considered incomplete.

```
incomplete.obs <- which(is.na(survey$Height) | is.na(survey$Wr.Hnd))
length(incomplete.obs)
```

```
[1] 29
```

Thus of the 237 observations, 29 of them have incomplete data. Observe the use of "complete.obs" as an argument for the correlation function. This means the function will only take into account those observations that are complete (not NA for either variable) when computing the statistic.

General Concepts

The purpose of a linear regression model is to come up with a function that estimates the mean of one variables given a particular value of another. In other words, we're looking for the mean of distribution of the response variable Y at a given level of X.

Fitting Linear models with `lm`

In R, the command `lm` performs the estimation of the parameters of a linear regression model. We'll use it to create a fitted linear object of the mean student height by handspan.

```
survey_fit <- lm(Height ~ Wr.Hnd, data = survey)
```

The most basic output of a fitted linear model class simply gives the estimators for β_0 and β_1.

```
survey_fit
```

```
Call:
lm(formula = Height ~ Wr.Hnd, data = survey)

Coefficients:
(Intercept)       Wr.Hnd
    113.954        3.117
```

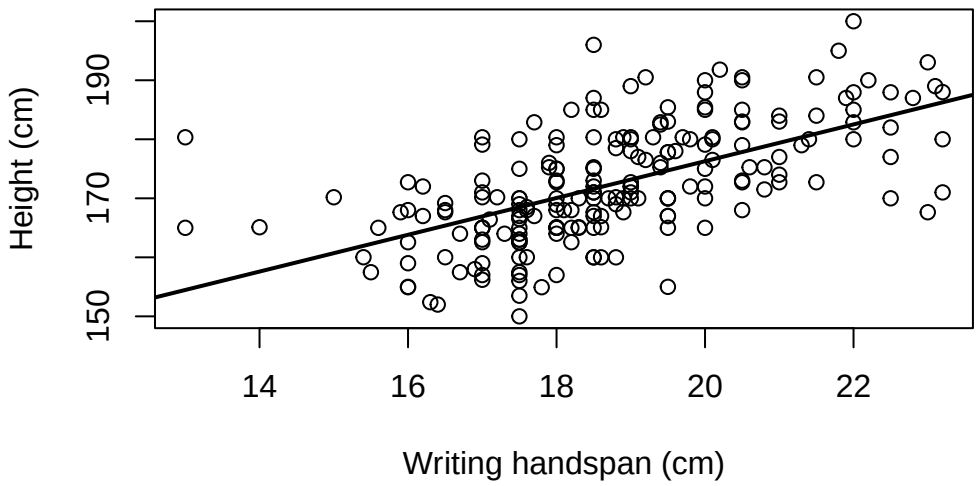
Here, our linear model is estimated as

$$\hat{y} = 113.954 + 3.117x$$

We can create a very simple graphic containing our regression line with the observations.

```
plot(survey$Height ~ survey$Wr.Hnd, xlab = "Writing handspan (cm)",
     ylab = "Height (cm)")

# add the regression line by simply passing the lm object
abline(survey_fit, lwd = 2)
```



Illustrating Residuals

We can extract the residuals from the fitted regression line. We'll demonstrate this by extracting the residuals for two specific observations.

```
# get 197th observation, arbitrary, I know
obsA <- c(survey$Wr.Hnd[197], survey$Height[197])
obsA
```

```
[1]  15.00 170.18
```

```
# now get the 154th guy
obsB <- c(survey$Wr.Hnd[154], survey$Height[154])
obsB
```

```
[1]  21.50 172.72
```

Ok, now let's inspect all of the attributes of the `lm` object to see what we can work with.

```r
names(survey_fit)
```

```
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "na.action"     "xlevels"       "call"          "terms"
[13] "model"
```

Let's extract the coefficients using a direct-access function (although $ also works).

```r
coefs <- coef(survey_fit)
coefs
```

```
(Intercept)      Wr.Hnd
 113.953623    3.116617
```

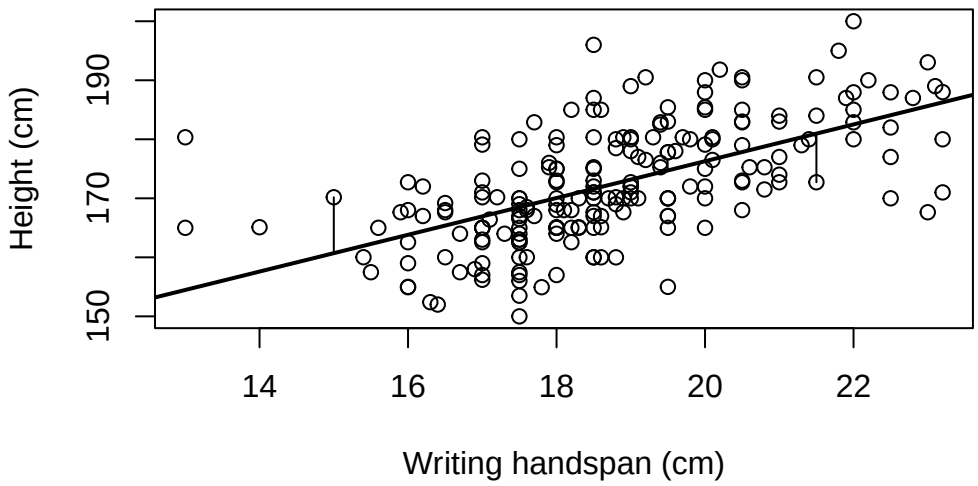
```r
beta0.hat <- coefs[1]
beta1.hat <- coefs[2]
```

Now, we can use all these values to plot some residuals for these two observations.

```r
plot(survey$Height ~ survey$Wr.Hnd, xlab = "Writing handspan (cm)",
     ylab = "Height (cm)")

# add the regression line by simply passing the lm object
abline(survey_fit, lwd = 2)

# give the coordinate pairs of where to start the segment and where to end it
segments(x0 = c(obsA[1], obsB[1]),   # starting xs
         y0 = beta0.hat + beta1.hat*c(obsA[1], obsB[1]), # starting ys (on regression line)
         x1 = c(obsA[1], obsB[1]),   # ending xs
         y1 = c(obsA[2], obsB[2])
         )
```



This graph shows the residuals for the two observations (i.e. the distances of the observed values from their expectation for a given level of X).

Statistical Inference

Summarizing the Fitted Model

```
summary(survey_fit)
```

```
Call:
lm(formula = Height ~ Wr.Hnd, data = survey)

Residuals:
     Min       1Q   Median       3Q      Max
-19.7276  -5.0706  -0.8269   4.9473  25.8704

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 113.9536     5.4416   20.94   <2e-16 ***
Wr.Hnd        3.1166     0.2888   10.79   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.909 on 206 degrees of freedom
  (29 observations deleted due to missingness)
Multiple R-squared:  0.3612,    Adjusted R-squared:  0.3581
F-statistic: 116.5 on 1 and 206 DF,  p-value: < 2.2e-16
```

The summary of the fitted linear regression object contains tests for significance. The standardized t value and p-value are reported for each parameter and these are the results of a two-tailed hypothesis test formally defined as

$$H_0 : \beta_0 = 0$$

$$H_A : \beta_0 \neq 0$$

for the intercept and

$$H_0 : \beta_1 = 0$$

$$H_A : \beta_1 \neq 0$$

for the slope.

The interpretation is the the null implies the predictor has no effect on the response and the alternative is that there is any effect of the covariate. The interpretation for the intercept case may not necessarily be meaningful depending on the scope of the model, but it can still be tested.

R also conveniently provides a function for the class of `lm` to compute confidence intervals for our parameter estimates.

```r
confint(survey_fit, level = 0.95)
```

```
                2.5 %     97.5 %
(Intercept) 103.225178 124.682069
Wr.Hnd        2.547273   3.685961
```

Coefficient of Determination

The coefficient of determination is displayed in the summary output along with the adjusted R^2.

In this example our R^2 value can be interpreted as about 36.1% of the variation in the student heights can be attributed to handspan.

The adjusted R^2 value is a measure that takes into account the number of parameters that require estimation. This is generally important only if you're using the coefficient of determination to assess the overall "quality" of the fitted model in terms of a balance between goodness of fit and complexity.

Prediction

Recall, a prediction interval for an observed response is used provide the possible range of values that an individual realization of the response variable might take, given a level of X.

Confidence Intervals for Mean Heights

We can use the `predict` command to compute these. To demonstrate, we'll compute prediction intervals of interest for the explanatory variables.

```
xvals <- data.frame(Wr.Hnd=c(14.5, 24))
xvals
```

```
  Wr.Hnd
1   14.5
2   24.0
```

Now, we can use these values to compute a confidence interval for the distribution for Y at both of these levels of X.

```
# confidence intervals to describe the variability of the mean response
pred.ci <- predict(survey_fit, newdata = xvals, interval = "confidence")
pred.ci
```

```
       fit      lwr      upr
1 159.1446 156.4956 161.7936
2 188.7524 185.5726 191.9323
```

Prediction Intervals for Individual Observations

We can also use the same function to create prediction intervals for individual observations. Again, these prediction intervals provide a possible range of values that an individual observation of the response variable might take at a given level of X.

```
pred.pi <- predict(survey_fit, newdata = xvals, interval = "prediction")
pred.pi
```

```
       fit      lwr      upr
1 159.1446 143.3286 174.9605
2 188.7524 172.8390 204.6659
```

Prediction intervals are larger than confidence intervals since raw observations at a specific X value will be more variable than their mean value at a specific X.

Plotting Intervals

It is often helpful to visualize the CI and PI for a given regression.

```
plot(survey$Height ~ survey$Wr.Hnd, xlim = c(13,24), ylim = c(140,205),
     xlab = "Writing handspan(cm)",
     ylab = "Height (cm)")

abline(survey_fit, lwd = 2)

points(xvals[,1], pred.ci[,1], pch=8)  # change the point style

# add segments for the prediction interval
segments(x0 = c(14.5, 24), y0 = c(pred.pi[1,2], pred.pi[2,2]),
         x1 = c(14.5, 24), y1 = c(pred.pi[1,3], pred.pi[2,3]),
         col = "gray", lwd = 3
         )

# add segments for the confidence interval
segments(x0 = c(14.5, 24), y0 = c(pred.ci[1,2], pred.ci[2,2]),
         x1 = c(14.5, 24), y1 = c(pred.ci[1,3], pred.ci[2,3]),
         lwd = 2
         )

# produce bands around the regression line for all values of the predictor
```

```r
xseq <- data.frame(Wr.Hnd=seq(12, 25, length=100))
ci.band <- predict(survey_fit, newdata = xseq, interval = "confidence")
pi.band <- predict(survey_fit, newdata = xseq, interval = "predict")

# plot the bands
lines(xseq[,1], ci.band[,2], lty = 2, col = "blue")
lines(xseq[,1], ci.band[,3], lty = 2, col = "blue")
lines(xseq[,1], pi.band[,2], lty = 2, col = "red")
lines(xseq[,1], pi.band[,3], lty = 2, col = "red")
```