Chapter 4 & 5 - Linear Regression, Hypothesis Test, and Confidence Interval with One Regressor

Ercio Munoz

September 19, 2018

Linear Regression with one regressor

This example uses a panel data set on test performance, school characteristics, and student demographic backgrounds for California school districts, 1998-1999.

The question we have in mind is whether or not student-teacher ratio (STR) affects student test scores (testscr). We can represent this relationship using the population regression line as:

$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$

where y_i represents testscr of school i, x_i represents STR of school i and ϵ_i a random disturbance. In this case, β_1 is our parameter of interest, which represents the expected change in test score for a unit change in STR (in this case, a unit means one student more per teacher).

First, we import the data set from the web site, and given that it is formatted for Stata (.dta), we need to first install the package "foreign" (it allow us to use data formatted for another econometric software) using "install.packages()" command (from now on we will omit this and we will just call the package assuming we have installed it before):

```
install.packages("foreign")
```

Now we call the package using the command "library()" and import the data set as a data.frame object using the command "read.dta()":

```
library(foreign)
a = "http://fmwww.bc.edu/ec-p/data/stockwatson/caschool.dta"
data_set = read.dta(a)
# class() command tell us what kind of object we have
class(data_set)
```

```
## [1] "data.frame"
```

We should be able to see an object called "data_set" in the environment (upper-right side of R-studio). We can check some descriptive statistics of its content using the commands "summary()" or look at the first 6 observations of each variable using the command "head()":

```
summary(data_set)
```

```
##  observation_number    dist_cod        county             district
##  Min.   :  1.0      Min.   :61382   Length:420         Length:420
##  1st Qu.:105.8      1st Qu.:64308   Class :character   Class :character
##  Median :210.5      Median :67760   Mode  :character   Mode  :character
##  Mean   :210.5      Mean   :67473
##  3rd Qu.:315.2      3rd Qu.:70419
##  Max.   :420.0      Max.   :75440
##    gr_span             enrl_tot          teachers          calw_pct
##  Length:420         Min.   :   81.0   Min.   :   4.85   Min.   : 0.000
##  Class :character   1st Qu.:  379.0   1st Qu.:  19.66   1st Qu.: 4.395
##  Mode  :character   Median :  950.5   Median :  48.56   Median :10.520
##                     Mean   : 2628.8   Mean   : 129.07   Mean   :13.246
```

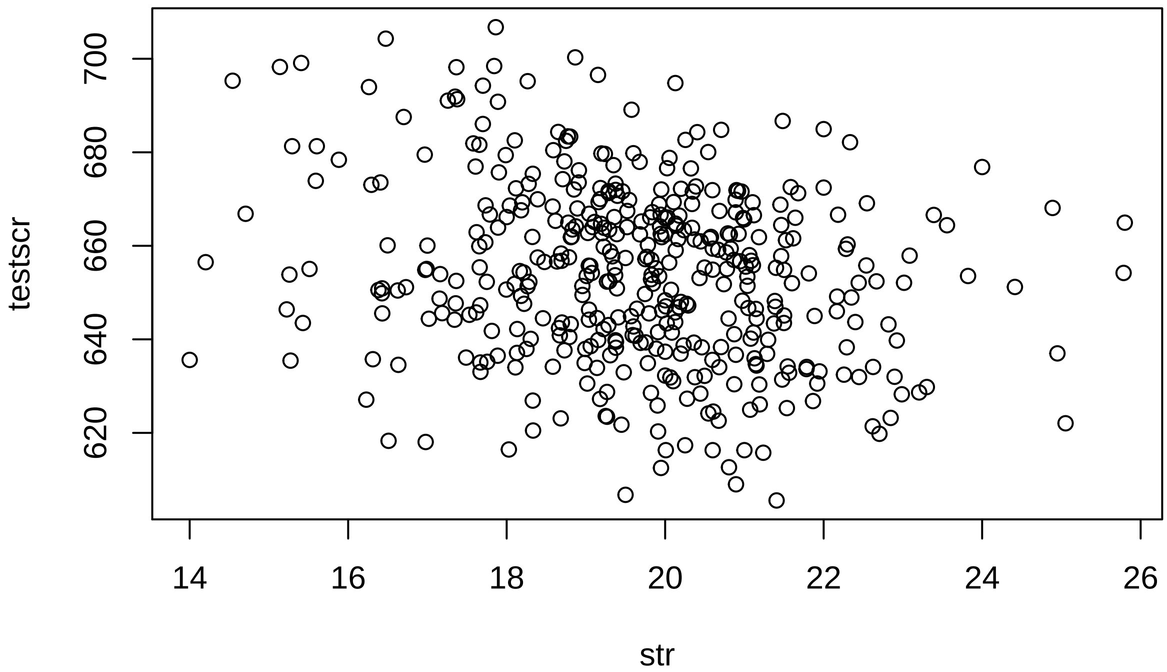
```
##                       3rd Qu.: 3008.0   3rd Qu.: 146.35   3rd Qu.:18.981
##                       Max.   :27176.0   Max.   :1429.00   Max.   :78.994
##     meal_pct         computer          testscr         comp_stu
##  Min.   :  0.00   Min.   :   0.0   Min.   :605.5   Min.   :0.00000
##  1st Qu.: 23.28   1st Qu.:  46.0   1st Qu.:640.0   1st Qu.:0.09377
##  Median : 41.75   Median : 117.5   Median :654.5   Median :0.12546
##  Mean   : 44.71   Mean   : 303.4   Mean   :654.2   Mean   :0.13593
##  3rd Qu.: 66.86   3rd Qu.: 375.2   3rd Qu.:666.7   3rd Qu.:0.16447
##  Max.   :100.00   Max.   :3324.0   Max.   :706.8   Max.   :0.42083
##     expn_stu          str            avginc           el_pct
##  Min.   :3926   Min.   :14.00   Min.   : 5.335   Min.   : 0.000
##  1st Qu.:4906   1st Qu.:18.58   1st Qu.:10.639   1st Qu.: 1.941
##  Median :5215   Median :19.72   Median :13.728   Median : 8.778
##  Mean   :5312   Mean   :19.64   Mean   :15.317   Mean   :15.768
##  3rd Qu.:5601   3rd Qu.:20.87   3rd Qu.:17.629   3rd Qu.:22.970
##  Max.   :7712   Max.   :25.80   Max.   :55.328   Max.   :85.540
##     read_scr        math_scr
##  Min.   :604.5   Min.   :605.4
##  1st Qu.:640.4   1st Qu.:639.4
##  Median :655.8   Median :652.5
##  Mean   :655.0   Mean   :653.3
##  3rd Qu.:668.7   3rd Qu.:665.9
##  Max.   :704.0   Max.   :709.5
```

```r
head(data_set)
```

```
##   observation_number dist_cod  county                        district
## 1                  1    75119 Alameda              Sunol Glen Unified
## 2                  2    61499   Butte            Manzanita Elementary
## 3                  3    61549   Butte     Thermalito Union Elementary
## 4                  4    61457   Butte Golden Feather Union Elementary
## 5                  5    61523   Butte        Palermo Union Elementary
## 6                  6    62042  Fresno         Burrel Union Elementary
##   gr_span enrl_tot teachers calw_pct meal_pct computer testscr  comp_stu
## 1   KK-08      195    10.90   0.5102   2.0408       67  690.80 0.3435898
## 2   KK-08      240    11.15  15.4167  47.9167      101  661.20 0.4208333
## 3   KK-08     1550    82.90  55.0323  76.3226      169  643.60 0.1090323
## 4   KK-08      243    14.00  36.4754  77.0492       85  647.70 0.3497942
## 5   KK-08     1335    71.50  33.1086  78.4270      171  640.85 0.1280899
## 6   KK-08      137     6.40  12.3188  86.9565       25  605.55 0.1824818
##   expn_stu      str    avginc    el_pct read_scr math_scr
## 1 6384.911 17.88991 22.690001  0.000000    691.6    690.0
## 2 5099.381 21.52466  9.824000  4.583333    660.5    661.9
## 3 5501.955 18.69723  8.978000 30.000002    636.3    650.9
## 4 7101.831 17.35714  8.978000  0.000000    651.9    643.5
## 5 5235.988 18.67133  9.080333 13.857677    641.8    639.9
## 6 5580.147 21.40625 10.415000 12.408759    605.7    605.4
```

We can use a plot to check graphically whether it appears to be a relationship between the two variables of interest (Note that the command "attach()" tells R that we are going to use a particular data.frame, so we can use directly the names of the variables inside the data.frame):

```r
# Scatter plot
attach(data_set)
plot(str,testscr)
```



Now we can run our first linear regression with the command "lm()" creating an object called "reg1" containing the outcome of the regression. We can then summarize this outcome with "summary()":

```
reg1 = lm(testscr~str,data=data_set)
summary(reg1)
```

```
##
## Call:
## lm(formula = testscr ~ str, data = data_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.727 -14.251   0.483  12.822  48.540
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 698.9330     9.4675  73.825  < 2e-16 ***
## str          -2.2798     0.4798  -4.751 2.78e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.58 on 418 degrees of freedom
## Multiple R-squared:  0.05124,    Adjusted R-squared:  0.04897
## F-statistic: 22.58 on 1 and 418 DF,  p-value: 2.783e-06
```

The summary shows us the value of the estimated coefficients, standard errors, t values, p values, residual standard errors (SER), R squared, Adjusted R squared, F-statistic and p-value of this F-statistic (we will see later their meanings). Note that we have to specify the data.frame with the data for the regression.

The previous regression uses the standard OLS formula to compute the standard errors, which assumes homoskedasticity (the sequence of disturbances have the same finite variance). However, we will be using standard errors that are robust to heteroskedasticity (in other words, we are not going to be assuming homoskedasticity). To do this we call the packages "lmtest" and "sandwich", to use the commands "coeftest()" and "vcovHC()":

```r
library(lmtest)
library(sandwich)
# Now we use robust standard errors
coeftest(reg1, vcov = vcovHC(reg1, "HC1"))
```

```
##
## t test of coefficients:
##
##              Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 698.93295   10.36436 67.4362 < 2.2e-16 ***
## str          -2.27981    0.51949 -4.3886 1.447e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

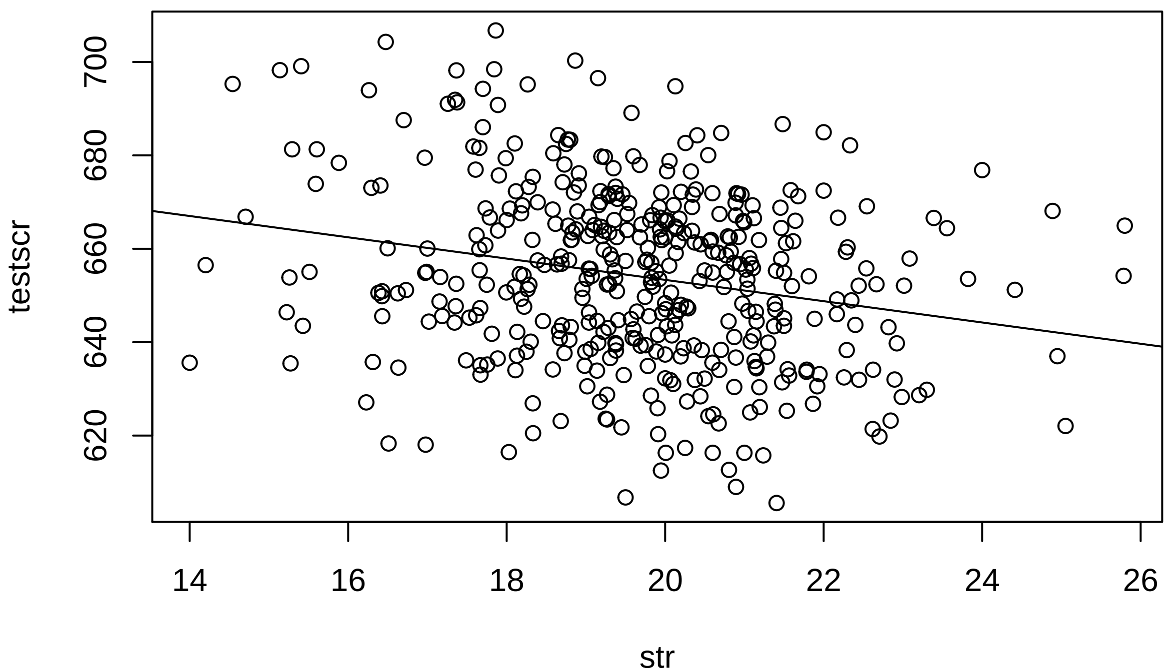
These packages also allows us to compute confidence intervals ($\beta_1 = \{\hat{\beta}_1 \pm 1.96SE(\hat{\beta}_1)\}$):

```r
# Confidence interval
coefci(reg1, vcov = vcovHC(reg1, "HC1"))
```

```
##                  2.5 %     97.5 %
## (Intercept) 678.560192 719.305713
## str          -3.300945  -1.258671
```

We can see the regression line in a plot (note that we first have to write the independent variable "x" and then our independent variable "y" in the command "plot()"):

```r
reg1 = lm(testscr~str)
plot(str,testscr)
abline(reg1)
```



From the regression output we obtain coefficients, predicted values and residuals (let see the first 6 values of them):

```r
b.hat = coef(reg1)
b.hat
```

```
## (Intercept)         str
##  698.932952   -2.279808
```

```
testscr.hat = fitted(reg1)
head(testscr.hat)
```

```
##        1        2        3        4        5        6
## 658.1474 649.8608 656.3069 659.3620 656.3659 650.1308
```

```
u.hat = resid(reg1)
head(u.hat)
```

```
##         1         2         3         4         5         6
##  32.65260  11.33917 -12.70689 -11.66198 -15.51593 -44.58076
```

Let's confirm some properties of OLS:

```
# Confirm property (1) of OLS, mean of u equal zero:
mean(u.hat)
```

```
## [1] -5.764833e-16
```

```
# Confirm property (2) of OLS, residual uncorrelated to x:
cor(str, u.hat)
```

```
## [1] -5.850616e-16
```

```
# Confirm property (3) of OLS, expected value conditional on mean of x equal to mean of y:
mean(testscr)
```

```
## [1] 654.1565
```

```
b.hat[1] + b.hat[2] * mean(str)
```

```
## (Intercept)
##    654.1565
```

We can compute R^2 in three different ways:

```
var(testscr.hat) / var(testscr)
```

```
## [1] 0.0512401
```

```
1 - var(u.hat) / var(testscr)
```

```
## [1] 0.0512401
```

```
cor(testscr, testscr.hat)^2
```

```
## [1] 0.0512401
```

Finally, we can do hypothesis testing about the coefficient β_1. Let's replicate the test reported after "coeftest()" command:

```
library(lmtest)
library(sandwich)
# Store coefficients and standard errors into summary1
summary1 = coeftest(reg1, vcov = vcovHC(reg1, "HC1"))
summary1
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value  Pr(>|t|)
```

```
## (Intercept) 698.93295   10.36436 67.4362 < 2.2e-16 ***
## str          -2.27981    0.51949 -4.3886 1.447e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Let's check the t and p-value of the null beta_1=0
z = summary1[2,1]/summary1[2,2]
z
```

```
## [1] -4.388557
```

```
# The p value according to the normal distribution is:
2*pnorm(-abs(z))
```

```
## [1] 1.141051e-05
```

```
# The p value according to the Student t distribution s:
2*pt(-abs(z),df=reg1$df.residual)
```

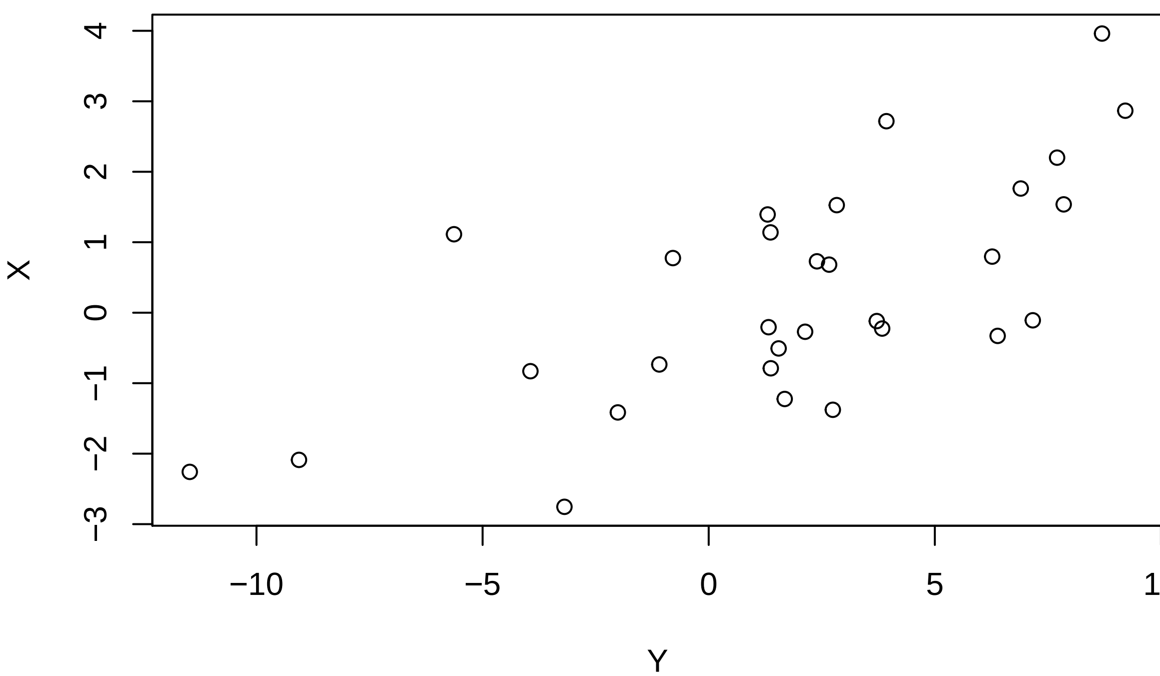
```
## [1] 1.446737e-05
```

We get the same when using the Student t distribution (which assumes disturbances are normally distributed).

Regression with a simulate a data set

We generate a very small sample of 30 observations of a random variable X and a disturbance Z that are iid distributed normal, and create an outcome Y using the equation $Y_i = 1 + 2X_i + Z_i$ (think about this equation as the population linear regression from which the sample comes):

```
# We fix a value for the seed in order to replicate each time the same random numbers
set.seed(1)
Z = rnorm(30,mean=0,sd=4)
X = rnorm(30,mean=0,sd=2)
Y = 1 + 2*X + Z
plot(Y,X)
```



Now suppose we only observe Y and X, and we would like to estimate the slope of $Y_i = \beta_0 + \beta_1 X_i + Z_i$:

```
reg2 = lm(Y~X)
summary(reg2)
```

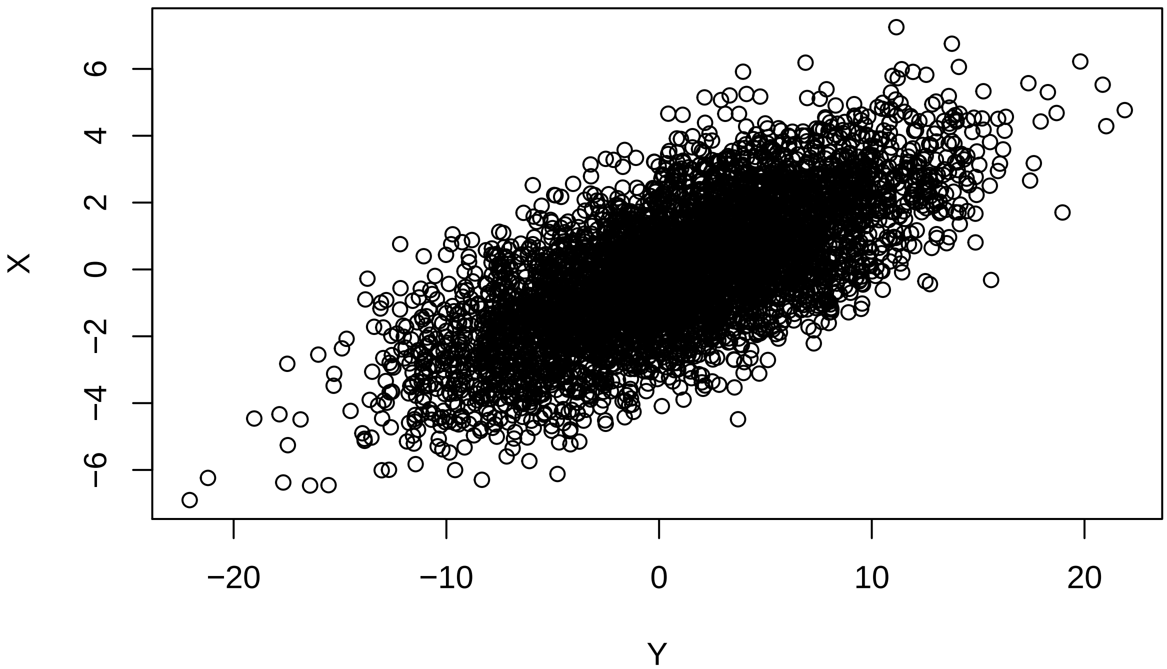
```
##
## Call:
## lm(formula = Y ~ X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.2845 -2.1795  0.8875  2.4693  6.0935
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.2998     0.6958   1.868   0.0723 .
## X             2.1131     0.4387   4.817 4.57e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.757 on 28 degrees of freedom
## Multiple R-squared:  0.4532, Adjusted R-squared:  0.4337
## F-statistic: 23.21 on 1 and 28 DF,  p-value: 4.571e-05
```

```
library(lmtest)
library(sandwich)
coeftest(reg2, vcov = vcovHC(reg2, "HC1"))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  1.29980    0.73010  1.7803   0.08588 .
## X            2.11310    0.43516  4.8560 4.112e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We got a beta close to 2. Let see what we get with a bigger sample:

```
set.seed(1)
Z = rnorm(5000,mean=0,sd=4)
X = rnorm(5000,mean=0,sd=2)
Y = 1 + 2*X + Z
plot(Y,X)
```




```r
reg3 = lm(Y~X)
summary(reg3)
```

```
##
## Call:
## lm(formula = Y ~ X)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.6698  -2.6683  -0.0478   2.8051  15.2529
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.98718    0.05809    17.0   <2e-16 ***
## X            1.99665    0.02911    68.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.107 on 4998 degrees of freedom
## Multiple R-squared:  0.4849, Adjusted R-squared:  0.4848
## F-statistic:  4706 on 1 and 4998 DF,  p-value: < 2.2e-16
```

```r
library(lmtest)
library(sandwich)
coeftest(reg3, vcov = vcovHC(reg3, "HC1"))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 0.987180   0.058096  16.992 < 2.2e-16 ***
## X           1.996649   0.029078  68.666 < 2.2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With a sample of 5000 instead of 30 we get something much closer. Note also that we are not able to reject the null of the intercept equal to 0 in the small sample (we know that the true value is 1).

Let's run a t test with the null hypothesis $H_0 : \beta_1 = 2$ that we know it is true:

```
# First we store the coefficients and standard errors into summary3
summary3 = coeftest(reg3, vcov = vcovHC(reg3, "HC1"))
# Create the t-statistic
z3 = (summary3[2,1]-2)/summary3[2,2]
z3
```

```
## [1] -0.1152521
```

```
# The p value according to the normal distribution is:
2*pnorm(-abs(z3))
```

```
## [1] 0.9082453
```

We fail to reject the null hypothesis.

Finally, notice what happen to the plot if we decrease the variance of the disturbance keeping the variance of X as before:

```
set.seed(1)
Z = rnorm(5000,mean=0,sd=.1)
X = rnorm(5000,mean=0,sd=2)
Y = 1 + 2*X + Z
plot(Y,X)
```