# Relationships in R

Jan Rovny

## Relationships

One of the most interesting questions we have about phenomena out in the real world concerns their connections. Is A related to B? If one achieves higher level of education, what impact is it likely to have on her income?

## Covariance and Correlation

The simplest way of assessing relationships is by considering how different variables change (or vary) together, that is to say, how do they covary or (in a mathematically related assessment) correlate.

Let's first consider a simple example of madeup data:

```
a<-c(1,2,3,4,5,6,7,8,9,10) #vector *a* with 10 values
b<-c(10,7,5,4,6,3,4,2,3,1) #vector *b* with 10 values
```

You should be able to note that as $a$ increases, $b$ tends to decrease, though it is not 'perfect'. Can we describe the relationship between $a$ and $b$, and assess its strength?

To do so, we can turn to two useful measures: covariance and correlation. These measures address the extent to which the values of two variables 'change together', in other words, to what extent they covary or correlate.

Covariance is measured like this:

$Cov(XY) = S_{(XY)} = \frac{\sum(x_i-\bar{X})(y_i-\bar{Y})}{N-1}$

Correlation is measured like this:

$Corr(XY) = r_{(XY)} = \frac{\sum(\frac{x_i-\bar{X}}{S_X})(\frac{y_i-\bar{Y}}{S_Y})}{N-1}$

Generally speaking, correlation is a preferred measure because it 'standardizes' each variable by dividing its variance by its standard deviation. Consequently, correlation ranges from -1

to 1, where -1 means that the two variables are perfectly negatively correlated, 1 means that the variables are perfectly (positively) correlated, and 0 means that there is no association between the two variables.

In R we can assess this simply:

```
cov(a,b)
```

```
[1] -7.055556
```

```
cor(a,b)
```

```
[1] -0.8843145
```

Let us now turn to real data:

```
library(rio)
D<-import("https://jan-rovny.squarespace.com/s/France.dta")
```

Let's consider some relationships, for example one between education and income:

```
cov(D$educ,D$inc, use="complete.obs")
```

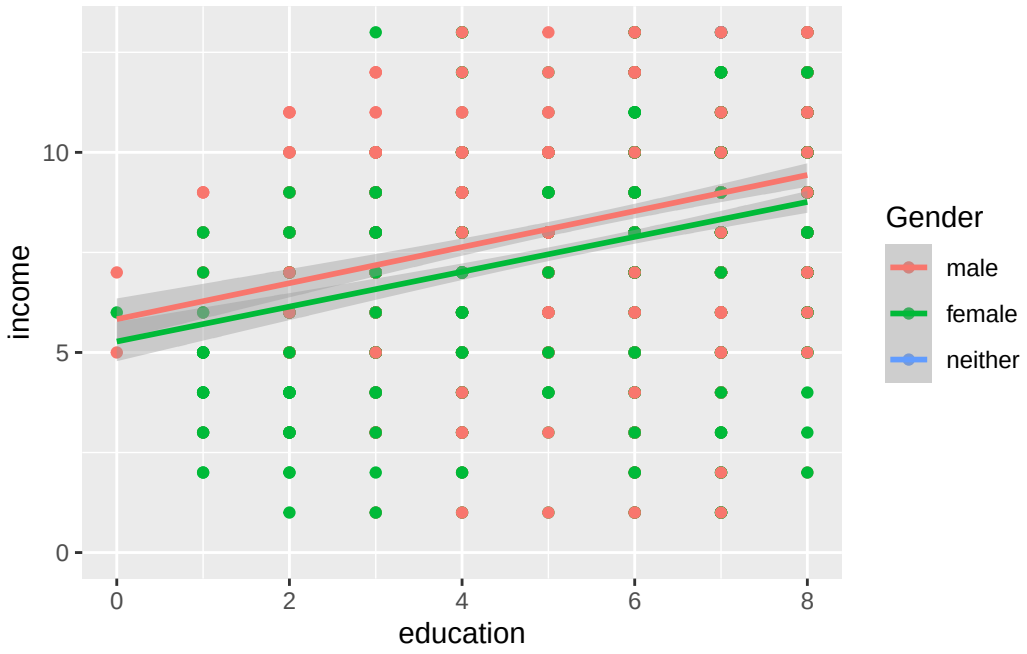
```
[1] 1.672296
```

```
cor(D$educ,D$inc, use="complete.obs")
```

```
[1] 0.3299377
```

Of course, a picture is worth a thousand words, so let's graph it!

```
library(ggplot2)
ggplot(D, aes(x=educ, y=inc,  color=as.factor(female)))+ #define the data, the space, and
  geom_point()+ #add the points for each observation
  geom_smooth(method=lm)+ #add a line (using linear model method)
  xlab("education")+ylab("income")+ #label axes
  ylim(0,13)+ #define the y-axis range
  scale_color_discrete(name="Gender", labels=c("male","female", "neither")) #name and labe
```




Here we can see the relationship between education and income by gender. Interestingly, men and women have the same relationship between eductation and gender, but men are generally better off.

## Regression

Let us now consider views on European integration as a function of one's education. We will turn to model views on the EU, which we will call $y$ as a function of education, which we will call $x$.

We will describe the relationship as $y_i = \alpha + \beta x_i + \epsilon$

Here $\alpha$ and $\beta$ are regression coefficients, describing the relationship between $x$ and $y$, and $\epsilon$ is a so-called *error term*, which captures all the deviations (or variance) that our model fails to explain.

R allows us to evaluate the values of $\alpha$ and $\beta$:

```
model<-lm(sov_eu~educ, data=D)
summary(model)
```

```
Call:
```

```
lm(formula = sov_eu ~ educ, data = D)

Residuals:
    Min      1Q  Median      3Q     Max
-1.7573 -0.6708  0.2427  0.6750  1.9344

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.06565    0.07664  26.952  < 2e-16 ***
educ         0.08645    0.01330   6.502 1.06e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.019 on 1575 degrees of freedom
  (145 observations deleted due to missingness)
Multiple R-squared:  0.02614,   Adjusted R-squared:  0.02552
F-statistic: 42.28 on 1 and 1575 DF,  p-value: 1.061e-10
```

The helpful aspect of R's object-based language is that everything we do becomes an object. The above regression model has just become another object that we can call up and work with. For example:

```
model$coefficients # vector containing all the coefficients of model
model$coefficients[1] #calls up the first item (intercept) of the coefficient vector
model$coefficients[2] #calls up the second item (the coefficient on educ) of the coefficie
predict(model) #produces a vector of fitted values (yhat) for each observation
model$residuals # is a vector containing all of the residuals of the model
summary(model)$r.squared  # produces a scalar with the R^2 of the model
vcov(model) #produces the variance-covariance matrix of the estimator
```

In a similar way, we can construct a matrix containing the key information from our model. We can then use the individual items in the matrix:

```
stat.coef  <- summary(model)$coefficients #construct a matrix of model results
stat.coef[,1]    # 1st column: coefficients
```

```
(Intercept)        educ
 2.06564743  0.08645347
```

```
stat.coef[,2]    # 2nd column: se for each coef
```

```
(Intercept)        educ
 0.07664079  0.01329630
```

```
stat.coef[,3]    # 3rd column: t-value for each coef
```

```
(Intercept)        educ
  26.952324    6.502069
```

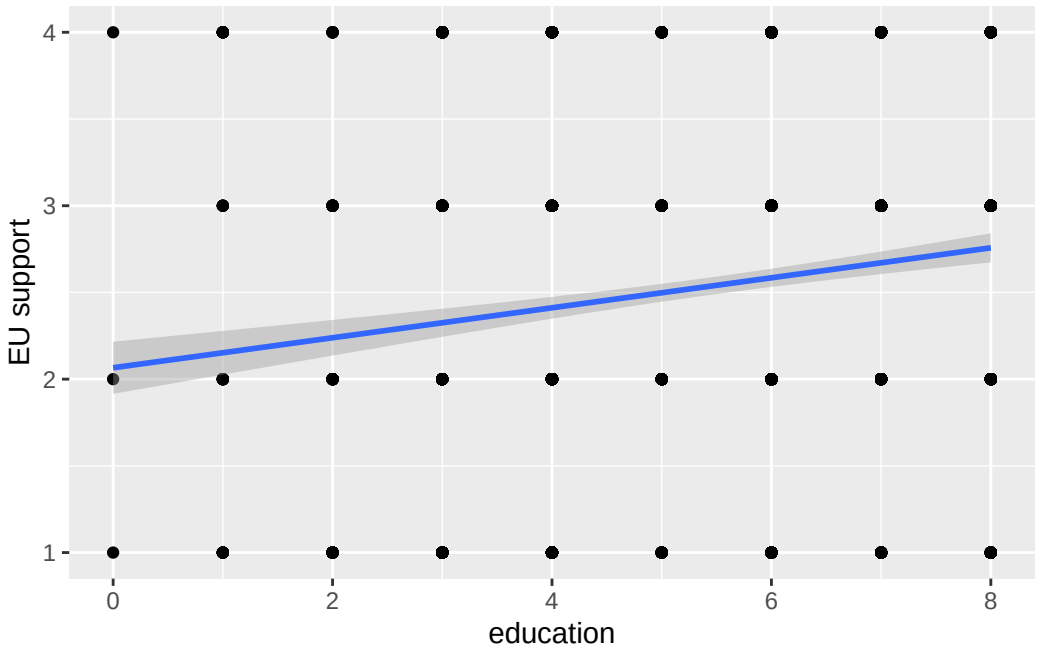
```
stat.coef[,4]    # 4th column: p-value for each coef
```

```
  (Intercept)          educ
6.895496e-132  1.060705e-10
```

## Visualization

It is nice to visualize what we are doing. We can plot the values of $x$ and $y$ and draw a line that is mathematically expressed in the equation: $y_i = \alpha + \beta x_i$. We call this the regression line, or line of best fit.

```
ggplot(D, aes(x=educ, y=sov_eu))+ #define the data, the space, and split color by gender
  geom_point()+ #add the points for each observation
  geom_smooth(method=lm)+ #add a line (using linear model method)
  xlab("education")+ylab("EU support")+ #label axes
  ylim(1,4) #define the y-axis range
```



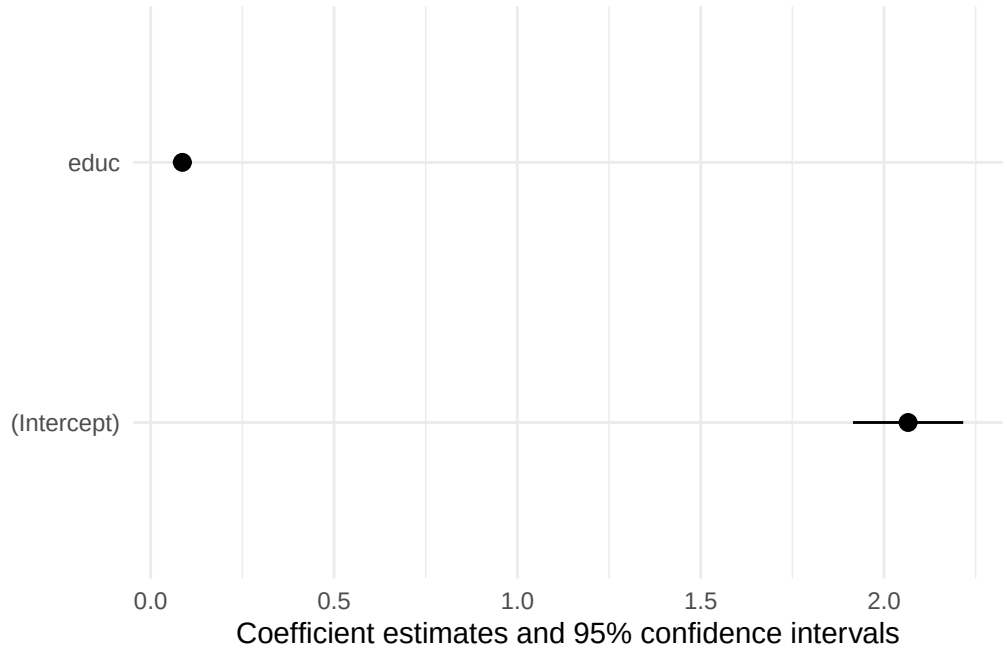

In the next step, we may be interested in seeing the confidence intervals of our results:

```
confint(model)
```

```
                2.5 %    97.5 %
(Intercept) 1.91531871 2.2159761
educ        0.06037316 0.1125338
```

Finally, we may wish to create a so-called *coefficient plot* which shows the magnitude of the coefficients:

```
library(modelsummary)
modelplot(model)
```


educ
(Intercept)
0.0
0.5
1.0
1.5
2.0
Coefficient estimates and 95% confidence intervals