Leibniz Institute for Financial Research SAFE
Theodor-W.-Adorno-Platz 3 I 60629 Frankfurt am Main I Germany

LEIBNIZ INSTITUTE SAFE Sustainable Architecture for Finance in Europe

Introduction to R

Ina Krapp
SAFE Research Datacenter*

Last Update: November 20, 2023

This tutorial gives an introduction into linear regression with R. For complete beginners, it is recommended to first go through the Introduction to R tutorial. This tutorial is intended to be run as a QuartoMarkdown file in RStudio, which allows you to execute the code yourself. To run it, download the QuartoMarkdown version here and open it in a recent version of RStudio.

Contents

1 Contact

If you encounter any difficulties or just want general information, do not hesitate to contact us.

SAFE Research Datacenter: datacenter@safe-frankfurt.de

More information about the SAFE Research Datacenter and further guides can be found here.

*krapp@safe-frankfurt.de

2 Prerequisite

R can be downloaded here.
We'll also use RStudio. RStudio is a user interface which makes working with R much more convenient. You can get it here.

3 Load data

If you did not already do it when working through 'Introduction to R', you'll need to download the data first.

```
library(readr)
```

We'll use the package to load the data from here: https://github.com/allisonhorst/palmerpenguins/blob/m
Download it and put it into the same folder that you have currently open in RStudio.
Read it into R:

```
penguins <- read_csv("penguins.csv")
```

4 Modifying data

Often, when you've loaded the data, you will not be immediately able to work with it. More often than not, it needs to be changed beforehand.
You can not simply write in a dataframe by clicking on it. This is because such changes would not be reproducible. You can modify a dataframe in any way you want, but in R, it has to be done using code.
This dataset contains no identifier yet. We'll add one:

```
penguins$id <- 1:nrow(penguins)
```

We can also change individual values. For example, assume that we knew the penguin in row 48 (which currently has a NA value) was female. The gender is in the seventh column. You indicate the position of a value in the dataframe by giving the row first and the column second.

```
penguins[48, 7] <- 'female'
```

If we want our dataset to only include rows in which all values are known, we remove the NA's with this command:

```
penguins <- na.omit(penguins)
```

Remember that in R, it are always entire rows which get removed. It is impossible to only remove individual values because a dataframe always needs to have one entry for each column in every row. If an individual value is not known, you set it to be NA instead.
There are also options such as replacing NA's with the mean value of their column, but these are not always scientifically sound. Which strategy for missing data is appropriate depends on the dataset and your research question.
Once we are confident the data is what it should be, we can start our regression.

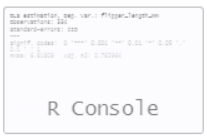
5 Running a regression

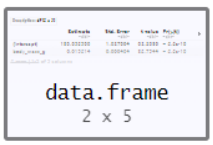
We'll use the fixest package to run regressions. Many packages in R can be used for this purpose, and base R allows to run regressions, too. But the fixest package is very fast - for large datasets, that's important.

```
library(fixest)
```

A regression in R is a function. It has a name and function arguments. Since we work with linear regressions, we use the feols command (ols are ordinary least squares).
The first regression checks if penguins who weigh more have longer flippers:

```
feols(flipper_length_mm ~ body_mass_g, data = penguins)
```





Description: df [2 × 5]

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
(Intercept)	136.952599	1.987904	68.8930	< 2.2e-16	***
body_mass_g	0.015214	0.000464	32.7544	< 2.2e-16	***

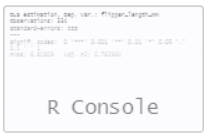
2 rows

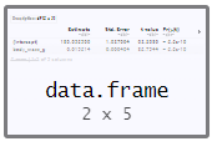
They do, the estimate is positive and significant (with a very small p-value). The effect is not very large, though. Also note that the results are themselves a data frame, with two rows and five columns.
Regression results can be saved by assigning them to a name:

```
feols(flipper_length_mm ~ body_mass_g, data = penguins)
```

This ensures that you can always access them. You can call the table of results we saw above with the summary function:

```
summary(flipper_regression)
```





Description: df [2 × 5]

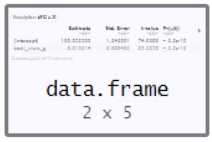
	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
(Intercept)	136.952599	1.987904	68.8930	< 2.2e-16	***
body_mass_g	0.015214	0.000464	32.7544	< 2.2e-16	***

2 rows

It is good practice to calculate heteroskedasticity-robust standard errors. This is done with the vcov-argument. You can do it in the regression function or in the summary function.

```
summary(flipper_regression, vcov = 'hetero')
```

As you can see, the estimate remains the same. But the standard error and the metrics calculated based on it (the t-values and p-values) change.
In the next step, write your own regression. Is the bill length positively related to the bill depth? Calculate heteroskedasticity-robust standard errors for this.

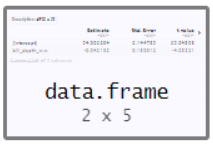


Description: df [2 × 5]

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
(Intercept)	136.952599	1.848691	74.0809	< 2.2e-16	***
body_mass_g	0.015214	0.000432	35.2356	< 2.2e-16	***

2 rows

```
1   # Specifying vcov = 'hetero' in the regression or the summary is
    possible
2   bill_regression <- feols(bill_length_mm ~ bill_depth_mm, data = penguins
    , vcov = 'hetero')
>>  3   summary(bill_regression, vcov = 'hetero')
```



Description: df [2 × 5]

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
(Intercept)	54.998804	2.144735	25.64363	< 2.2e-16	***
bill_depth_mm	-0.642132	0.130018	-4.93881	1.2477e-06	***

2 rows

Surprisingly, the relation is significantly negative. Penguins, it appears, either have a short, but thick bill or a long, but thin one.

6 A technical remark on lists

A regression is stored in R as a list. A list is a special form of datatype. You can access elements from this list, for example, the number of observations the regression was run with ("nobs"). Important to know: If you extract an element from a list with single brackets, even if it only contains a single value, it will be a smaller list. For example, if I extract the value "nobs" with single brackets, it gets extracted as a list.

```
1   bill_regression["nobs"]
2   class(bill_regression["nobs"])
```

```
> bill_regression["nobs"]
$nobs
[1] 334

> class(bill_regression["nobs"])
[1] "list"
```

To get the element itself, use double brackets:

```
    bill_regression[["nobs"]]
    class(bill_regression[["nobs"]])
```

```
> bill_regression[["nobs"]]
[1] 334
> class(bill_regression[["nobs"]])
[1] "integer"
```

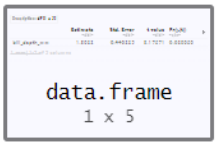
7 Fixed effects

In many datasets, such regressions as the ones above are not a good approach. That is because they are only valid when the data is a random sample of the underlying population. But what is the population we are considering here?
The penguin data contains three species, covers three islands and was collected over 3 years. The bills may not be formed the same for all three species. To test this, we use a fixed effect.

```
    species_regression <- feols(bill_length_mm ~ bill_depth_mm |  species,
   data = penguins)
```

Again, we need to think carefully about standard errors. If the data is clustered according to specific variables (here: species), the standard errors have to be clustered as well.

```
    summary(species_regression, vcov = 'cluster')
```



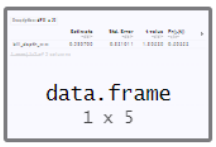
Description: df [1 × 5]

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
bill_depth_mm	1.3988	0.440885	3.17271	0.086629	.

1 row

As we can see, the estimate is no longer negative.
How does it look like when we also want to control for the sex of the penguins?

```
species_regression <- feols(bill_length_mm ~ bill_depth_mm |  species + sex,
    data = penguins )
summary(species_regression, vcov = 'cluster')
```



Description: df [1 × 5]

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
bill_depth_mm	0.530706	0.381011	1.39289	0.29828	

1 row

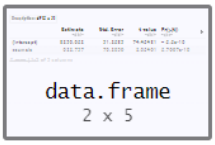
8 Running a regression with factor variables

This form of clustering is to control for a certain variable, like species. If, instead, we wanted to know the effect such a variable has, we need to run the regression with a factor variable. Transforming a variable into a factor is straightforward:

```
    penguins$sex = as.factor(penguins$sex)
```

Note that the type changed: Previously, the type of sex you saw when klicking on 'penguins' in the Environment Pane was 'chr'. Now, it is 'Factor w/ 2 levels' "female", "male".
Levels are the different values a factor variable can take.

```
weight_regression <- feols(body_mass_g ~ sex, data = penguins)
summary(weight_regression, vcov = 'hetero')
```



Description: df [2 × 5]

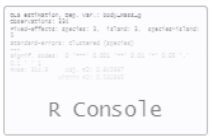
	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
(Intercept)	3856.928	51.8235	74.42431	< 2.2e-16	***
sexmale	688.757	79.8650	8.62401	2.7007e-16	***

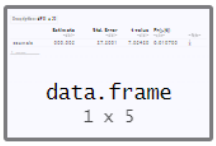
2 rows

The variable 'sexmale' is created automatically when the regression has a factor or character vector as input. The system creates a dummy, a variable 'sexmale' that can take 0 or 1, depending on if the penguin is male or not. It then estimates the effect of this dummy, compared to a baseline (here, female penguins are the baseline).
Again, we want to control for species effects:

```
    weight_regression <- feols(body_mass_g ~ sex  | species + island +
  species^island, data = penguins)
    summary(weight_regression, vcov = 'cluster')

```





Description: df [1 × 5]

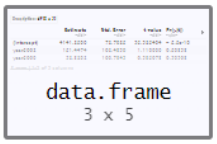
	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
sexmale	669.992	87.8691	7.62489	0.016769	*

1 row

We see that although the effect is still significant at the 5%-level, the species fixed effects have a certain influence as well.
In the next step, create your own factor variable. Answer the following question: Does the body weight of the penguins change with the years?

```
penguins$year = as.factor(penguins$year)
year_regression <- feols(body_mass_g ~ year, data = penguins)
summary(year_regression, vcov = 'hetero')

```



Description: df [3 × 5]

>>

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <chr>	<fctr>
(Intercept)	4141.8269	78.7682	52.582494	< 2.2e-16	***
year2008	121.4474	108.4356	1.119996	0.26353	
year2009	58.3868	109.7345	0.532073	0.59503	

3 rows

No. The estimates are positive, but not significant (the p-value is very large). That is also important to know: If the variables for 2008 or 2009 negatively impacted their weight, it may mean that the colonies were endangered.