# Towards a programmable Dataverse: Finding, getting and analysing the Data you need

Jan. 2022

**Lars Kaczmirek, Lisa Hirsch**

## Abstract

The presented Open Educational Resources help mitigate common problems that arise when scientific data is used for educational purposes: Often, finding, getting and analyzing data is a time-consuming task. To address this problem, we demonstrate how teachers and their students can use the methods to directly download data. Good analysis scripts begin with code that fetches the data from a repository. This contributes to making research reproducible and supports proper citation of research data. Two examples, one executable in Python, one in a local R installation, will show how to download datafiles from a Dataverse repository, and take first steps in analysing data.

# 1 Introduction

AUSSDA - The Austrian Social Science Data Archive is a rich source of data for research purposes. Some data can also be used for other purposes under the specified conditions. In these materials which AUSSDA distributes freely we show how to access the AUSSDA Dataverse via different programming tools, and download data that is open access. We also show how to dive into data analysis by introducing summary statistics and plotting data.

Imagine the following ideal situation: A teacher in a master class discusses a scientific article with students. They do so by re-running and making changes to the analysis code that was distributed with the published article. The programming code automatically fetches the data from the repository that was used by the authors of the article to archive and share the respective data. Students are enabled to reproduce the results including tables and figures. As the teacher makes suggestions to change the code students are encouraged to increase their analysis and critical thinking skills.

Compare the previous scenario to the often experienced situation in which teachers or students have to register to gain access to data and time is spent on delivering the data to the students. Often more time is lost in figuring out how downloaded data has to be prepared before the above described analyses code runs without errors. Thus the current situation often means that teachers have to spend extra hours to prepare article, analysis or replication code and the corresponding data before the work with the students can begin.

These materials help mitigate the problems that arise from the scenario above: As a teacher, you use the direct data download methods in your teaching to make sure that everyone uses the correct files. No more wasted time with distributing data files to a class and checking on files not found. As a student, starting your analysis script with code to download the data goes a long way to make your research reproducible. No more confusion about which data set or version has actually been used to run the analysis.

The materials are hosted on `https://github.com/AUSSDA/dataverse-oer-resources`. It contain the public domain dedication under which this work is published, a general readme document, this README.pdf, a Jupyter Notebook to run the example in Python via `mybinder.org` and a file that passes the requirements to `mybinder.org`, an R script to execute the example using a local R installation and a pdf that explains the R code.

Whenever you reuse data from others you have the moral obligation and in many cases also the legal obligation to cite your sources appropriately. Remember to always include the persistent identifier - for data in the AUSSDA Dataverse this is the DOI. You can also refer to `https://dataverse.org/best-practices/data-citation` for best practices.


Figure 1: Structure of the materials hosted on Github

## 2 Using the Dataverse with Python

You will learn how to navigate to the repository and download a datafile, and how to use different datafiles.

```
# This installs the packages so Jupyter Notebook can execute the
    code.
# Version info: python 3.9.7
# If you run this in a mybinder.org environment, you do not need to
     execute this block of code.
import sys
!conda install --yes --prefix {sys.prefix} requests
!conda install --yes --prefix {sys.prefix} jsonschema
!conda install --yes --prefix {sys.prefix} pandas
!conda install --yes --prefix {sys.prefix} numpy
!conda install --yes --prefix {sys.prefix} matplotlib

!{sys.executable} -m pip install pyDataverse
```

Start by installing and loading the packages you need.

```
# The package pyDataverse handles everything you need to connect
    and download files from a Dataverse installation.
# Check out the documentation:
# https://pydataverse.readthedocs.io/en/latest/user/basic-usage.
    html#download-and-save-a-dataset-to-disk

from pyDataverse.api import NativeApi, DataAccessApi
import io
import pandas as pd
```

Decide which data repository you want to use, and which dataset you want to download. This example is based on the AUSSDA Dataverse. AUSSDA - The Austrian Social Science Data Archive is a rich source for data for research purposes. Some data can also be used for other purposes under the specified licence conditions.

You will learn how to download the Open Access edition of the Social Survey

Austria 2016 (Bacher et al., 2018). First you define the repository URL, then the persistent identifier (DOI).

```
# Step 1: Define and connect to repository API
repository_url = 'https://data.aussda.at/'
native_api = NativeApi(repository_url)

# Step 2: Define and download dataset
DOI = "doi:10.11587/EHJHFJ"
dataset = native_api.get_dataset(DOI)
```

To download the data file automatically, the following code proceeeds in several steps. First, you create a list of available files. Then, you download the first datafile. In Dataverse, there are several files stored under each DOI: datafiles in various formats suitable for STATA, SPSS, R, tab-delimited, and documentation as pdf-files. The code first goes through all files in the file list (i.e. all files in the dataset), and identifies all files that are tab-delimited files. These are the datafiles you want to download when using Python. Then, it picks the first of these tab-delimited files which is usually the original datafile by default. Then you download this file.

```
# Create a list of files that are available in the dataset
files_list = dataset.json()['data']['latestVersion']['files']

# Find the first tab-file
ident = []

for file in files_list:
    filename = file["dataFile"]["filename"]
    file_id = file["dataFile"]["id"]
    if filename.endswith('.tab'):
        ident.append(file_id)

# Save the ID of the first tab file that occurs in the list
datafile_id = ident[0]

# Step 3: Connect to API for data access at the repository
data_access_api = DataAccessApi(repository_url)

# Step 4: Download data file using its id
response = data_access_api.get_datafile(datafile_id)
```

Now that you have downloaded the datafile, you can work with it: It needs to be transformed into a format that can be used by the Pandas package.

```
## Working with the datafile Pt. 1

# Transform the response into a Pandas data frame
data = io.StringIO(str(response.content,'utf-8'))

# The file is a tab-delimited file in the repository, i.e. the
    seperator between columns (variables) is a tab.
# The first line (the header) consists of the variable names.
data = pd.read_csv(data, sep="\t", index_col=0)
```

In the next step you check if the download worked, and what the data looks like.

The rows are unique observations. In your case each row is an individual respondent, because you use survey data. The columns are variables, which are mostly questions in survey data. The variable names in the header show which question is stored in which column, and the values show the answer each respondent gave to each question.

```
## Working with the datafile Pt. 2

# Show sample data by displaying the first rows
data.head()
```

Now that you know how to download a dataset you may want to try a different dataset that is also published under an open access license such as CC BY that does not require a login or API key.

```
# If you want to use the survey data of the Corona Panel Project (
    Kittel et al, 2020) change the DOI in the code above to
    "10.11587/P5YJ0O" in Step 2. Look at the codebook or the
    questionnaire to determine which variable contains information
    on which question.
```

There is also a way to access a file driectly: If the automatic download described above does not leave you with the desired file, you can specify the name of the file you want to retrieve manually, and download it using the function get_dataframe_by_name. In order to find the name, check the object files_list that you created after Step 2, or look at the landing page for the Social Survey Austria in the repository at `https://doi.org/10.11587/EHJHFJ`.

```
# Manual Process: Specify the id of the file you want to download,
    and proceed with Step 1 and Step 2
# Check the id numbers for each file.
for file in files_list:
    filename = file["dataFile"]["filename"]
    file_id = file["dataFile"]["id"]
    print("File name {} has id {}".format(filename, file_id))

# Specify which file you want to retrieve.
datafile_id = 188
# Step 1: Connect to API for data access at the repository
data_access_api = DataAccessApi(repository_url)
# Step 2: Retrieve data file
response = data_access_api.get_datafile(datafile_id)
```

# 3 Using the Dataverse with R

You will learn how to navigate to the repository and download a datafile, and how to use different datafiles.

In order to execute the code, you need an installation of the software R, for example RStudio. Please refer to `https://www.rstudio.com/products/rstudio/download/` for more information on how to install RStudio. You can then copy the code below into a R Script, and execute the commands locally.

Start by installing and loading the packages you need. If you have not already installed the packages, remove the comment signifier in the first line to execute the function install.packages.

```
# install.packages("dataverse", "tibble", "ggplot2")

library(dataverse)
library(tibble)
library(ggplot2)
```

Decide which data repository you want to use, and which dataset you want to download. This example is based on the AUSSDA Dataverse.

You will learn how to download the Social Survey Austria 2016 (Bacher et al., 2018) from the Dataverse which is open access. First you define the repository URL, then the persistent identifier to the dataset (DOI).

```
# Step 1: Define and connect to repository API
repository_url <- "https://data.aussda.at/"
Sys.setenv("DATAVERSE_SERVER" = repository_url)

# Step 2: Define dataset and retrieve dataset
DOI <- "doi:10.11587/EHJHFJ"
dataset <- get_dataset(DOI)
```

To download the data file automatically, the following code proceeds in several steps: First, you create a list of available files. Then, you download the first datafile.

In Dataverse, there are several files saved under each DOI: datafiles (usually .tab files), and documentation as pdf-files. The following code saves a list of all these files.

```
# Downloading Data: list files
files_list <- dataset$files[c("filename", "contentType")]
```

The following code first goes through all files in the file list (i.e. all files in the dataset), and identifies all files that are tab-delimited files. In Dataverse, there are several files stored under each DOI: datafiles in various formats suitable for STATA, SPSS, R, tab-delimited, and documentation as pdf-files. The code first goes through all files in the file list (i.e. all files in the dataset), and identifies all files that are tab-delimited files. These are the datafiles you want to download when using R. Then you download this file.

```
# Downloading Data: identify desired file
filename <- files_list$filename
ext <- unlist(strsplit(filename, "\\."))[c(F,T)]
ident <- filename[which(ext %in% "tab")][1] # save the filename of
    the first .tab file
```

Use the function get_dataframe_by_name from the package Dataverse to download the datafile directly.

```
# Downloading Data: download data
data <- get_dataframe_by_name(
  paste(ident),
  paste(DOI),
  )
```

The function get_dataframe_by_name already reads in the data in the format that can be used by R.

The rows are unique observations. In your case each row is an individual respondent, because you use survey data. The columns are variables, which are mostly questions in survey data. The variable names in the header show which question is stored in which column, and the values show the answer each respondent gave to each question.

```
# Show sample data by displaying the first rows
head(data)
```

After having downloaded data, you can now start to analyse the data. In order to know what each variable means, and what its values mean, look at the questionnaire or into the codebook for the dataset you are retrieving. Go to `https://doi.org/10.11587/EHJHFJ` and look for a pdf file called codebook. There you will find information on each variable, for example the question that was asked, and possible answers.

The code below is an example of how to conduct some data analysis. The variable bf29_7 holds information on how much importance respondents assign to the societal importance of their job in their professional lives. (german wording: "Geben Sie auf der folgenden Liste bitte jeweils an, für wie wichtig Sie persönlich das für die berufliche Arbeit halten. Ein Beruf, der für die Gesellschaft nützlich ist."). The answers range from "1 = very important" to "5 = not important at all".

First, in order to exclude respondents who did not or could not answer the question, you set their responses to missing.

```
# Plotting Data
# recode missings and "cannot say" as missing values
# -99 = respondent has not worked before
# 8 = cannot say
data$bf29_7[data$bf29_7==-99] <- NA
data$bf29_7[data$bf29_7==8] <- NA
```

The code below shows how to produce a boxplot that shows the distribution of replies in relation to the respondents' gender.

```
# Where to go from here? Pt. 1: Plotting Data
boxplot(bf29_7~SEX_2016,data=data,
        horizontal=TRUE,
        names=c("male","female"),
        col=c("wheat","thistle"),
        xlab="1=very important, 5=not important at all
        (without cannot say)",
        ylab="Gender",
        main="Importance of societal relevance of one's job")
```

The code below shows how to produce a grouped bar plot of the replies in relation to the respondents' gender (see Figure 2).

```
# Plotting Data
# ggplot for bar plot by groups
# define facet labels
to_string <- as_labeller(c(‘1‘ = "male", ‘2‘ = "female"))

# plot
myplot <- ggplot(data=data, aes(x= bf29_7,  group=SEX_2016)) +
        geom_bar(aes(y = ..prop.., fill = factor(..x..)), stat="
    count") +
        scale_y_continuous(labels=scales::percent) +
        labs(y = "Percent",
             x="1=very important, 5=not important at all, without
    cannot say",
             title="Importance of societal relevance of one's job",
             fill="Wichtigkeit") +
        guides(fill=FALSE)+
        facet_grid(.~SEX_2016, labeller = to_string)
myplot
```

Another option is to use a different dataset that is published under an open access license (CC BY) that does not require a login or API key.

If you want to use the Corona Panel Project (Kittel et al., 2020) instead, change the DOI in the code above to "10.11587/P5YJ0O" in Step 2. Look at the codebook or the questionnaire again to determine which variable contains information on which question.

If the automatic download described above does not work, you can specify the name of the file you want to retrieve manually, and download it using the function get_dataframe_by_name. In order to find the name, check the object files_list that you created after Step 2, or look at the landing page for the Social Survey Austria in the repository at `https://doi.org/10.11587/EHJHFJ`.

You can also download the files manually when using the Corona Panel Project in case the automatic download does not work.

```
# Downloading data manually by accessing a file directly
data <- get_dataframe_by_name(
  "10007_da_de_v1_2-1.tab",
  paste(DOI))
```

# 4 Using the Dataverse with SPSS and STATA

You can also download files from the Dataverse using other statistical software packages like SPSS and STATA.

# References

Bacher, J., Beham-Rabanser, M., Grausgruber, A., Haller, M., Höllinger, F., Muckenhuber, J., ... Verwiebe, R. (2018). *Social Survey Austria 2016.*

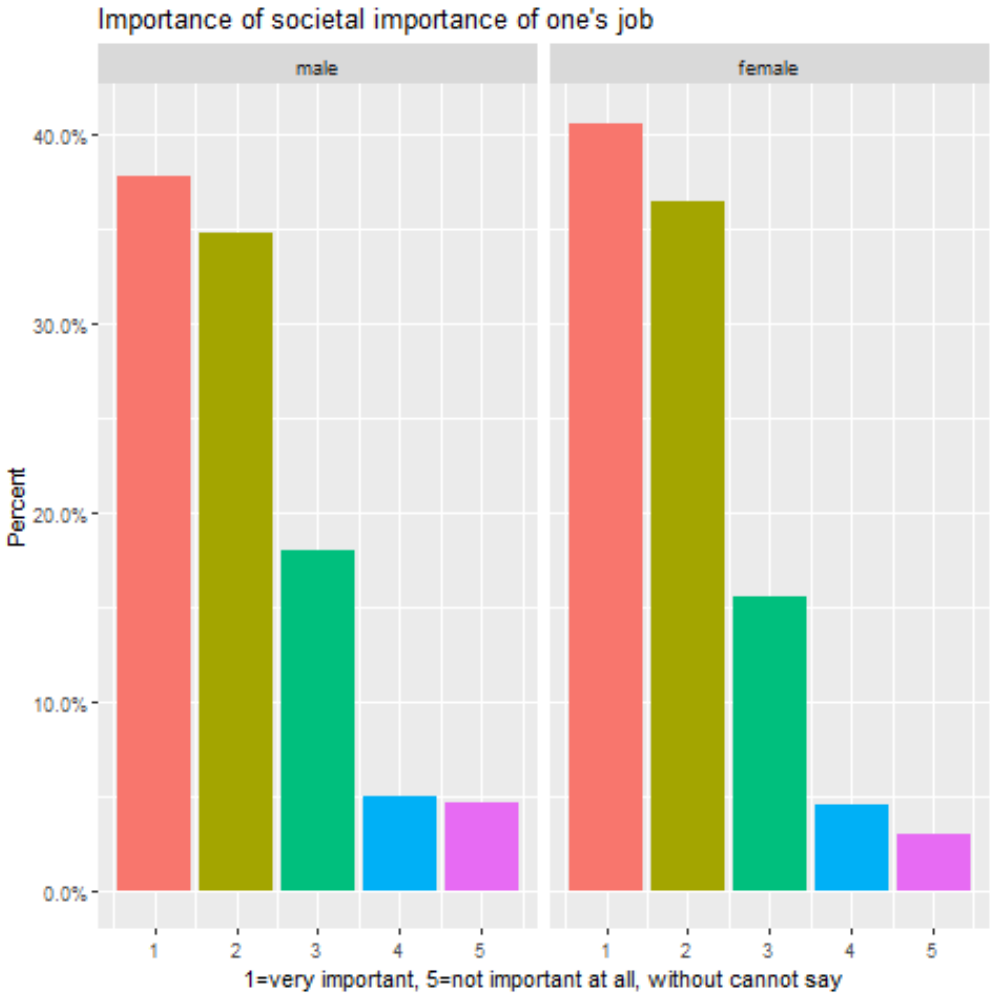


Figure 2: Example: Distribution of societal relevance of one's job

http://dx.doi.org/10.11587/EHJHFJ. (V3, AUSSDA) doi: 10.11587/EHJHFJ

Kittel, B., Kritzinger, S., Boomgaarden, H., Prainsack, B., Eberl, J.-M., Kalleitner, F., ... Schlogl, L. (2020). *Austrian Corona Panel Project (OA edition)*. http://dx.doi.org/10.11587/P5YJ0O. (V2, AUSSDA) doi: 10.11587/P5YJ0O