Eurostat database and package

{eurostat}

{eurostat} R package provides tools to access open data from Eurostat:

- data search,
- download,
- manipulation,
- visualization.

Installation

```
# Note the "eval=FALSE" argument: {r} chunk is not run.
# .. to install the package each time Markdown file is compiled (not recommended),
# .. just remove the "eval=FALSE" argument.
install.packages("eurostat")
```

Using the package

- Cheat sheet: eurostat R package
- Tutorial (vignette) for the eurostat R package
- Detailed documentation for eurostat functions

Command	Description
`get_eurostat_toc()`	Download table of contents of Eurostat datasets
`search_eurostat()`	Retrieve (grep) datasets titles from Eurostat
`get_eurostat()`	Read Eurostat data
`label_eurostat()`	Get Eurostat code descriptions
`get_eurostat_geospatial()`	Download geospatial data from GISGO

Search Eurostat for data

```
options(readr.default_locale=readr::locale(tz="Europe/Berlin"))
require(eurostat)
require(knitr)
require(tidyr)
require(dplyr)
require(ggplot2)
require(RColorBrewer)
```

```
# To actually run this {r} chunk, change the eval argument
toc <- get_eurostat_toc() # Downloads Table of Contents of Eurostat Data Sets
```

```
class(toc)
dim(toc)
str(toc,list.len = 10) # only few items listed
```

With `search_eurostat()`, you can search the table of contents for particular text (text patterns).

- *regex*: R regular expression syntax is used: see `?regex` for details.
- `.*` is particularly useful basic "tool" in text pattern search:
  - The period `.` matches any single character.
  - `*`: The preceding item (`.`) will be matched zero or more times.
- **regex is case sensitive** – see next example, where we search Eurostat for unemployment data:

```
# kable() generates tabular (formatted) output in Rmd files
kable(search_eurostat(".*unemployment.*rates.*NUTS", fixed=F))
```

title	code	type	last update of data	last ta
Dispersion of regional unemployment rates by NUTS 3 regions (%)	lfst_r_lmdur	dataset	20.03.2019	19.03.
Dispersion of regional unemployment rates by NUTS 3 regions (%)	lfst_r_lmdur	dataset	20.03.2019	19.03.

```
kable(search_eurostat(".*Unemployment.*rates.*NUTS", fixed=F))
```

title	code	type	last update of data
Unemployment rates by sex, age and NUTS 2 regions (%)	lfst_r_lfu3rt	dataset	29.04.2019
Unemployment rates by sex, age, country of birth and NUTS 2 regions	lfst_r_lfur2gac	dataset	29.04.2019
Unemployment rates by sex, age, citizenship and NUTS 2 regions	lfst_r_lfur2gan	dataset	29.04.2019
Unemployment rates by sex, age and NUTS 2 regions (%)	lfst_r_lfu3rt	dataset	29.04.2019
Unemployment rates by sex, age, country of birth and NUTS 2 regions	lfst_r_lfur2gac	dataset	29.04.2019
Unemployment rates by sex, age, citizenship and NUTS 2 regions	lfst_r_lfur2gan	dataset	29.04.2019
Unemployment rates by sex, age, country of birth and NUTS 2 regions	lfst_r_lfur2gac	dataset	29.04.2019
Unemployment rates by sex, age, citizenship and NUTS 2 regions	lfst_r_lfur2gan	dataset	29.04.2019

```
# Alternatively, you can use grep() to search a downloaded TOC
# .. this way, you can ignore the case-sensitive "issue"
toc <- get_eurostat_toc() # Downloads Table of Contents of Eurostat Data Sets
toc[grep(".*unemployment.*rates.*NUTS", toc$title, ignore.case = T),]
# ... this R code is not executed, provided for your information only
# ... you can switch the `eval` argument to produce the output table
```

Download data

As an example, let's choose the **Unemployment rates by sex, age and NUTS 2 regions (%)** dataset `lfst_r_lfu3rt`

- All datasets are available through a web browser (see the last string in the web address)
- http://appsso.eurostat.ec.europa.eu/nui/show.do?dataset=lfst_r_lfu3rt

Download the data:

```r
# http://appsso.eurostat.ec.europa.eu/nui/show.do?dataset=lfst_r_lfu3rt
Un.DF <- get_eurostat("lfst_r_lfu3rt", time_format = "num") # note the simplified time format
dim(Un.DF)
```

```
## [1] 135435      6
```

```r
# (5 age groups) * (3 gender categories) * (19 year 1999-2017) * (504 geo units) = 143.640
#
#
#
str(Un.DF)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    135435 obs. of  6 variables:
##  $ unit  : Factor w/ 1 level "PC": 1 1 1 1 1 1 1 1 1 1 ...
##  $ age   : Factor w/ 5 levels "Y15-24","Y15-74",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ sex   : Factor w/ 3 levels "F","M","T": 1 1 1 1 1 1 1 1 1 1 ...
##  $ geo   : Factor w/ 499 levels "AT","AT1","AT11",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ time  : num  2018 2018 2018 2018 2018 ...
##  $ values: num  9.4 12.4 NA 7.7 15.9 8.4 NA NA 6.7 7.8 ...
```

```r
# note the "value" variable - it contains Unemployment rates for a given "row"
#
#
#
summary(Un.DF) # note that observations are annual
```

```
##  unit           age        sex           geo              time
##  PC:135435   Y15-24:27073   F:45138   AT     :   300   Min.   :1999
##              Y15-74:27093   M:45140   AT1    :   300   1st Qu.:2004
##              Y20-64:27089   T:45157   AT11   :   300   Median :2009
##              Y_GE15:27093             AT12   :   300   Mean   :2009
##              Y_GE25:27087             AT13   :   300   3rd Qu.:2014
##                                       AT2    :   300   Max.   :2018
##                                       (Other):133635
##      values
##  Min.   : 0.60
##  1st Qu.: 5.20
##  Median : 8.00
##  Mean   :10.91
##  3rd Qu.:13.70
##  Max.   :90.70
##  NA's   :7513
```

```r
head(Un.DF,10)
```

```
## # A tibble: 10 x 6
##    unit  age    sex   geo    time values
##    <fct> <fct>  <fct> <fct> <dbl>  <dbl>
##  1 PC    Y15-24 F     AT     2018    9.4
##  2 PC    Y15-24 F     AT1    2018   12.4
##  3 PC    Y15-24 F     AT11   2018   NA
##  4 PC    Y15-24 F     AT12   2018    7.7
##  5 PC    Y15-24 F     AT13   2018   15.9
##  6 PC    Y15-24 F     AT2    2018    8.4
##  7 PC    Y15-24 F     AT21   2018   NA
##  8 PC    Y15-24 F     AT22   2018   NA
```

```
##  9 PC    Y15-24 F     AT3    2018    6.7
## 10 PC    Y15-24 F     AT31   2018    7.8
```

By default, variable identification is provided through Eurostat **codes**.

- To get human-readable labels instead, we can use `label_eurostat()` function
- Good for orientation in the dataset, NOT for `gather()` , `spread()` data handling

```
Un.DF.l <- label_eurostat(Un.DF, fix_duplicated = T)
```

```
head(Un.DF.l,6)
```

```
## # A tibble: 6 x 6
##   unit       age                 sex     geo               time values
##   <fct>      <fct>               <fct>   <fct>            <dbl>  <dbl>
## 1 Percentage From 15 to 24 years Females Austria           2018    9.4
## 2 Percentage From 15 to 24 years Females Ostösterreich     2018   12.4
## 3 Percentage From 15 to 24 years Females Burgenland (AT)   2018   NA
## 4 Percentage From 15 to 24 years Females Niederösterreich  2018    7.7
## 5 Percentage From 15 to 24 years Females Wien              2018   15.9
## 6 Percentage From 15 to 24 years Females Südösterreich     2018    8.4
```

Also, codes and their "descriptions" can be shown side-by-side:

```
head(kable(cbind(as.character(unique(Un.DF$geo)),as.character(unique(Un.DF.l$geo)))),17)
```

```
##  [1] "-----  ---------------------------------------------------------------------"
##  [2] "AT     Austria                                                              "
##  [3] "AT1    Ostösterreich                                                        "
##  [4] "AT11   Burgenland (AT)                                                      "
##  [5] "AT12   Niederösterreich                                                     "
##  [6] "AT13   Wien                                                                 "
##  [7] "AT2    Südösterreich                                                        "
##  [8] "AT21   Kärnten                                                              "
##  [9] "AT22   Steiermark                                                           "
## [10] "AT3    Westösterreich                                                       "
## [11] "AT31   Oberösterreich                                                      "
## [12] "AT32   Salzburg                                                            "
## [13] "AT33   Tirol                                                               "
## [14] "AT34   Vorarlberg                                                          "
## [15] "BE     Belgium                                                             "
## [16] "BE1    BE1 Région de Bruxelles-Capitale / Brussels Hoofdstedelijk Gewest   "
## [17] "BE10   BE10 Région de Bruxelles-Capitale / Brussels Hoofdstedelijk Gewest  "
```

```
# note the NUTS-code format:
# NUTS0 (states) have 2-digit IDs ... "AT"
# NUTS1 regions have 3-digit IDs
# NUTS2 regions have 4-digit IDS
#
head(kable(cbind(as.character(unique(Un.DF$age)),as.character(unique(Un.DF.l$age)))),5)
```

```
## [1] "-------  --------------------" "Y15-24   From 15 to 24 years "
## [3] "Y15-74   From 15 to 74 years " "Y20-64   From 20 to 64 years "
## [5] "Y_GE15   15 years or over    "
```

Data handling

We can simply save the data for subsequent use:

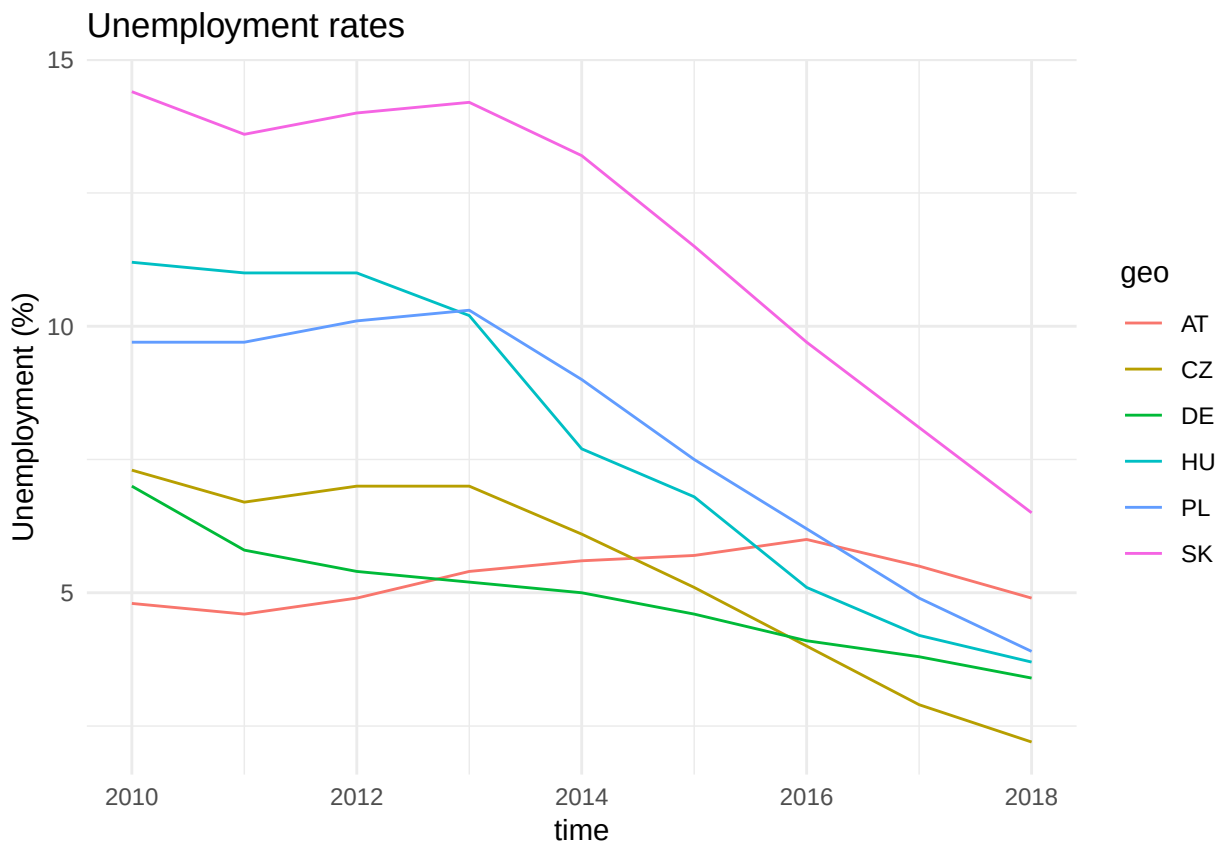
```
write.csv(Un.DF, "datasets/Unemployment.csv", row.names = F)
```

We can use `{tidyverse}` and `{ggplot2}` functions to filter and plot data.

Example 1: Unemployment plot for selected countries (time series)

- `Y15-74` i.e. age group *from 15 to 74 years*
- Select data 2010 and newer
- Total unemployment only (no M/F/T) structure
- Select only: Austria, Czech Republic, Germany, Hungary, Poland, Slovakia
- NUTS 0 (State-level)

```
Un.DF %>%
  filter(age == "Y15-74", time >= 2010, sex == "T") %>% # filter variables
  filter(geo %in% c("AT","CZ","DE","HU","PL","SK")) %>% # subset of countries
  ggplot(aes(x = time, y = values, colour = geo))+ # plot filtered data
    geom_line()+ # choose plot type
    ggtitle("Unemployment rates")+ # Define main title
    ylab("Unemployment (%)")+  # define label on the y-axis
    theme_minimal()  # choose plot "design"
```



Example 2: Chorpleth (infomap) of unemployment

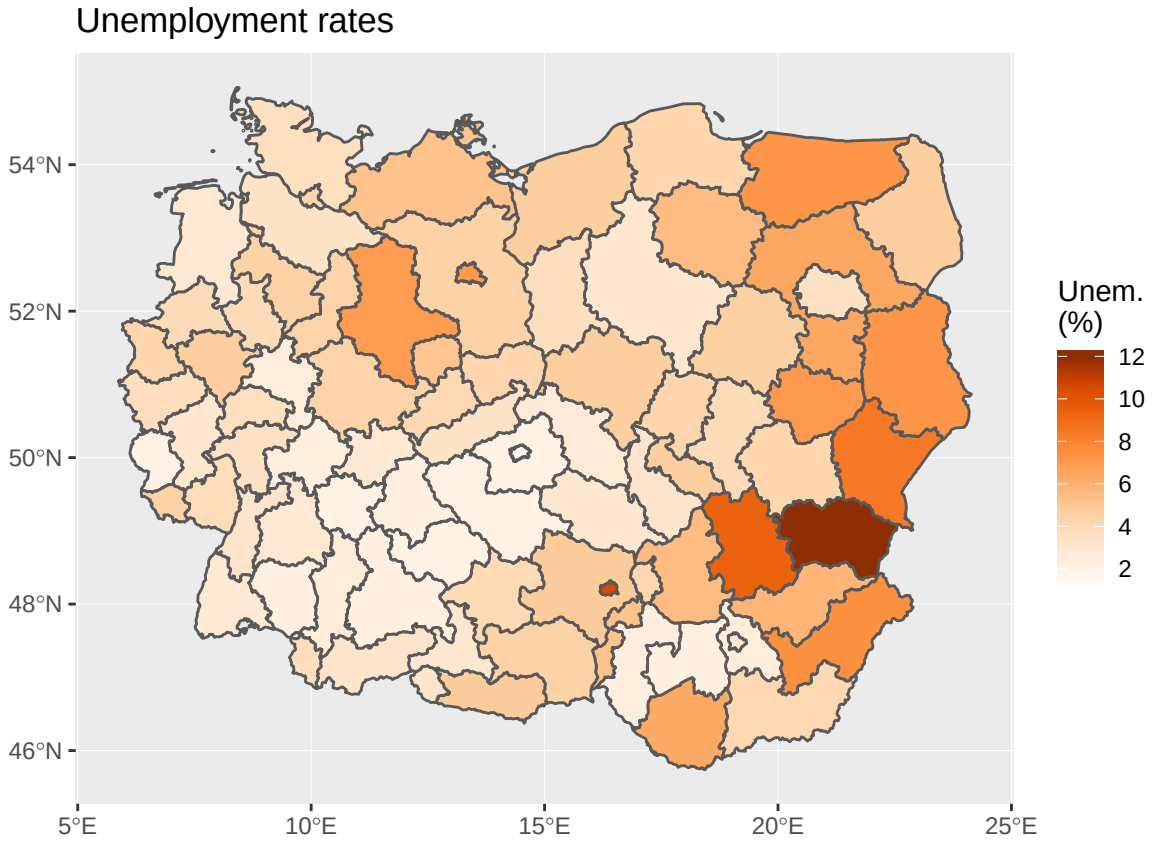
- `Y15-74` i.e. age group *from 15 to 74 years*
- Select data for 2017
- Total unemployment only (no M/F/T)
- NUTS 2
- Austria, Czech Republic, Germany, Hungary, Poland, Slovakia
- Draw choropleth (cartogram, infomap)

```
# Download geospatial data from GISCO # NUTS revisions: e.g.: 2010, 2013, 2016.
geodata <- get_eurostat_geospatial(resolution = "60", nuts_level = "2", year = 2016)
# Filter Unemployment dataset
Un.2016 <- Un.DF %>%
  filter(age == "Y15-74", time == 2017, sex == "T") %>% # filter variables, year, sex
  filter(nchar(as.character(geo)) == 4) %>% # NUTS2 regions have a 4-digit id
  mutate(NUTS0 = substr(as.character(geo), start=1, stop=2)) %>% # retrieve NUTS0 id from NUTS2
  filter(NUTS0 %in% c("AT","CZ","DE","HU","PL","SK"))
# Join Unemployment data with "map data"
map_data <- inner_join(geodata, Un.2016)
```

```
## Warning: Column `geo` joining character vector and factor, coercing into
```

```
## character vector
```

```
# plot the data
ggplot()+
  geom_sf(data = map_data, aes(fill = values))+
  # note that "values" is name of column that stores unemployment data...
  scale_fill_gradientn('Unem. \n(%) ', colours=brewer.pal(8, "Oranges"))+
  ggtitle("Unemployment rates")
```



Quick assignment: Add Netherlands (`NL`) to both plots (Examples 1 & 2).
