Assignment 1: The Causes of Economic Growth
Introduction to Applied Data Science
2022-2023

Bas Machielsen
a.h.machielsen@uu.nl

April 2023

Assignment 1: The Causes of Economic Growth

In this assignment, you will gather data from the *World Bank* website, and augment it with data from the *Clio Infra* website. Then, you will visualize these data using several graphs and tables, and test several hypotheses about the causes of economic growth. You will use this document to complete the code chunks which I have left unfinished to produce your own data analysis & visualization.

To start with, please replace my name and e-mail address with yours. Then, remove the lines:

```
output:
  pdf_document:
    includes:
      in_header: "preamble.tex"
```

from the document and replace them by:

```
output: pdf_document
```

Now, we're ready to start. For all of the code-related questions, please answer with code, and do not type (or copy) the answer from the console. Rather, let R *generate* your answer.

1. World Bank Data

The *World Bank* collects and processes large amounts of data and generates them on the basis of economic models. These data and models have gradually been made available to the public in a way that encourages reuse. In particular, the databases of the World Bank are available on https://data.worldbank.org/. It pays the effort to browse through the website, see if you can navigate your way through the website, and use the interface the World Bank provides you.

Normally, if you were looking for data from the World Bank, you would go to the website, find your dataset, download it to `.xlsx` or any other format, and then import it into an R `data.frame` using `read_xslx(.)`, or something else. But, this reliance on manual downloads of spreadsheets of the data they are interested in can quickly become overwhelming, as the work is manual, time consuming, and not easily reproducible.

Fortunately, however, there also exist an R package which allows you to browse swiftly through World Bank data, and easily download it as an R `data.frame`. You can get this package by installing:

```
library(pacman)
p_load("wbstats", "tidyverse")
```

You can navigate the database by searching for terms:

```
wbstats::wb_search("gdp per capita")
```

```
## # A tibble: 24 x 3
##    indicator_id       indicator                                          indic~1
##    <chr>              <chr>                                              <chr>
##  1 5.51.01.10.gdp     Per capita GDP growth                              GDP pe~
##  2 6.0.GDPpc_constant GDP per capita, PPP (constant 2011 international ~ GDP pe~
##  3 NV.AGR.PCAP.KD.ZG  Real agricultural GDP per capita growth rate (%)   The gr~
##  4 NY.GDP.PCAP.CD     GDP per capita (current US$)                       GDP pe~
##  5 NY.GDP.PCAP.CN     GDP per capita (current LCU)                       GDP pe~
##  6 NY.GDP.PCAP.KD     GDP per capita (constant 2010 US$)                 GDP pe~
##  7 NY.GDP.PCAP.KD.ZG  GDP per capita growth (annual %)                   Annual~
##  8 NY.GDP.PCAP.KN     GDP per capita (constant LCU)                      GDP pe~
##  9 NY.GDP.PCAP.PP.CD  GDP per capita, PPP (current international $)      This i~
## 10 NY.GDP.PCAP.PP.KD  GDP per capita, PPP (constant 2017 international ~ GDP pe~
## # ... with 14 more rows, and abbreviated variable name 1: indicator_desc
```

Afterwards, you can proceed to download data by executing `wb_data("indicator_id")`. You can then write this to a data.frame, and merge this data with other indicators to create a dataset. There exist many of these packages, and we will also use another today.

Apart from being easy to use, these packages also have another advantage: reproducibility. Collecting data by means of code allows other users to unambiguously reproduce your data collection process.

Firstly, we will look for GDP growth data.

Question x: pass a search query to `wb_search` for GDP growth data, and download the indicator for which the description matches "*GDP (current US$)*". The full description should read:

> GDP at purchaser's prices is the sum of gross value added by all resident producers in the economy plus any product taxes and minus any subsidies not included in the value of the products. It is calculated without making deductions for depreciation of fabricated assets or for depletion and degradation of natural resources. Data are in current U.S. dollars. Dollar figures for GDP are converted from domestic currencies using single year official exchange rates. For a few countries where the official exchange rate does not reflect the rate effectively applied to actual foreign exchange transactions, an alternative conversion factor is used.

```
wbstats::wb_search("GDP")
```

```
## # A tibble: 541 x 3
##    indicator_id         indicator                                        indic~1
##    <chr>                <chr>                                            <chr>
##  1 5.51.01.10.gdp       Per capita GDP growth                            GDP pe~
##  2 6.0.GDP_current      GDP (current $)                                  GDP is~
##  3 6.0.GDP_growth       GDP growth (annual %)                            Annual~
##  4 6.0.GDP_usd          GDP (constant 2005 $)                            GDP is~
##  5 6.0.GDPpc_constant   GDP per capita, PPP (constant 2011 internationa~ GDP pe~
##  6 BG.GSR.NFSV.GD.ZS    Trade in services (% of GDP)                     Trade ~
```

```
##  7 BI.WAG.TOTL.GD.ZS    Wage bill as a percentage of GDP                 <NA>
##  8 BM.KLT.DINV.WD.GD.ZS Foreign direct investment, net outflows (% of G~ Foreig~
##  9 BN.CAB.XOKA.GD.ZS    Current account balance (% of GDP)              Curren~
## 10 BN.CUR.GDPM.ZS       Current account balance excluding net official ~ Curren~
## # ... with 531 more rows, and abbreviated variable name 1: indicator_desc
```

```
gdp <- wbstats::wb_data('NY.GDP.MKTP.CD')
```

Question: Rename the variable `NY.GDP.MKTP.CD` to `gdp`. Remove the `NA` observations from the dataset. How many observations are there in the dataset in total?

```
gdp <- gdp %>%
  rename(gdp = `NY.GDP.MKTP.CD`) %>%
  filter(!is.na(gdp))

nrow(gdp)
```

```
## [1] 10336
```

Question x: How many observations per country are there? Show the first ten observations.

```
gdp %>%
  group_by(country) %>%
  summarize(count = n()) %>%
  head(10)
```

```
## # A tibble: 10 x 2
##    country             count
##    <chr>               <int>
##  1 Afghanistan            41
##  2 Albania                38
##  3 Algeria                62
##  4 American Samoa         19
##  5 Andorra                52
##  6 Angola                 42
##  7 Antigua and Barbuda    45
##  8 Argentina              60
##  9 Armenia                32
## 10 Aruba                  35
```

Question x: How many different years are there in the dataset? Put them in increasing order.

```
gdp %>%
  select(date) %>%
  pull() %>%
  unique() %>%
  sort()
```

```
##  [1] 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974
## [16] 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989
## [31] 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004
## [46] 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019
## [61] 2020 2021
```

Question x: For each country, what is the first and last year? Again show the first ten observations.

```
gdp %>%
  group_by(country) %>%
  summarize(first_year = min(date), last_year = max(date))
```

```
## # A tibble: 214 x 3
##    country             first_year last_year
##    <chr>                    <dbl>     <dbl>
##  1 Afghanistan               1960      2020
##  2 Albania                   1984      2021
##  3 Algeria                   1960      2021
##  4 American Samoa            2002      2020
##  5 Andorra                   1970      2021
##  6 Angola                    1980      2021
##  7 Antigua and Barbuda       1977      2021
##  8 Argentina                 1962      2021
##  9 Armenia                   1990      2021
## 10 Aruba                     1986      2020
## # ... with 204 more rows
```

Question: Make a summary of the data, with the mean, median, sd, min and max values for `gdp`.

```
gdp %>%
  summarize(mean = mean(gdp),
            median = median(gdp),
            sd = sd(gdp),
            min = min(gdp),
            max = max(gdp))
```

```
## # A tibble: 1 x 5
##            mean      median            sd      min     max
##           <dbl>       <dbl>         <dbl>    <dbl>   <dbl>
## 1 184192121208. 7824737792. 991080605921. 8824448. 2.30e13
```

Question: What country, in which year, had the lowest GDP? And the highest? Hint: use a function similar to `slice` from the `dplyr` package.

```
slice_min(gdp, gdp, n = 1)
```

```
## # A tibble: 1 x 9
##   iso2c iso3c country  date      gdp unit  obs_status footnote last_updated
##   <chr> <chr> <chr>   <dbl>    <dbl> <chr> <chr>      <chr>    <date>
## 1 TV    TUV   Tuvalu   1990 8824448. <NA>  <NA>       <NA>     2022-09-16
```

```
slice_max(gdp, gdp, n = 1)
```

```
## # A tibble: 1 x 9
##   iso2c iso3c country        date     gdp unit  obs_status footnote last_updated
##   <chr> <chr> <chr>         <dbl>   <dbl> <chr> <chr>      <chr>    <date>
## 1 US    USA   United States  2021 2.30e13 <NA>  <NA>       <NA>     2022-09-16
```

Next, we'll have a look at population data, which we can also retrieve from the World Bank database.

```
population <- wb_data("SP.POP.TOTL")
```

Question: Rename the population variable to `population` and overwrite this to memory.

```
population <- population %>%
  rename(population = `SP.POP.TOTL`)
```

Finally, we'll merge `population` with `gdp` on the basis of *country* and *year*.

Question: use `left_join` to merge `gdp` (left data.frame) with `population` (right data frame). Check whether everything has gone correctly. Save this dataframe to memory as `gdp_pop`. Select only `country`, `date`, `isco3c.x`, `gdp` and `population`. Then, use `mutate()` to create a new variable, `gdp_cap = gdp / population`. Then, again apply `na.omit()`. Write this data.frame to memory to `data`.

```
data <- left_join(gdp, population,
          by = c("country" = "country",
                 "date" = "date")) %>%
  select(country, date, iso3c.x, gdp, population) %>%
  na.omit() %>%
  mutate(gdp_cap = gdp / population)
```

Now, let's collect a pre-made version of GDP per capita from the World Bank website.

```
wb_search("gdp per capita")

alt_gdp_pc <- wb_data('NY.GDP.PCAP.CD') %>%
  filter(!is.na(`NY.GDP.PCAP.CD`))
```

Question: What is the correlation between these two variables? What does that mean?

```
cor(data$gdp_cap, alt_gdp_pc$NY.GDP.PCAP.CD)
```

2. World Tables: Capital Stock

Next, we'll proceed to find some potential determinants of GDP growth. One of the classical determinants of GDP per capita growth is the level of physical capital. Many models in macroeconomics explain economic well-being on account of the amount of capital in an economy. In particular, we'll look for a few measures from the Penn World Tables. This data has to be downloaded manually from this website. You can either manually download an Excel file, in which case, make sure to put it in the right directory when reading it. Or, you can run the following chunk, which downloads the file to your working directory:

```
pacman::p_load("readxl")

wd <- getwd()
download.file("https://www.rug.nl/ggdc/docs/pwt100.xlsx", file.path(wd, "pwt100.xlsx"))

pwt <- readxl::read_excel('pwt100.xlsx', sheet = 3)
```

We are looking for the `cn` variable, which indicates *Capital stock at current PPPs (in mil. 2017US$)*.

Question: Select the variables `countrycode`, `year`, and `cn`, and rewrite the dataframe to memory.

```
pwt <- pwt %>%
  select(countrycode, year, cn)
```

Now, we're looking to merge the two data.frames `data` and `pwt`, on the basis of common country names and years.

Question: Have a look at the two datasets below. On the basis of which two matched variables in both datasets do you have to perform the merge?

```
pwt
```

```
## # A tibble: 12,810 x 3
##    countrycode  year    cn
##    <chr>       <dbl> <dbl>
##  1 ABW          1950    NA
##  2 ABW          1951    NA
##  3 ABW          1952    NA
##  4 ABW          1953    NA
##  5 ABW          1954    NA
##  6 ABW          1955    NA
##  7 ABW          1956    NA
##  8 ABW          1957    NA
##  9 ABW          1958    NA
## 10 ABW          1959    NA
## # ... with 12,800 more rows
```

```
data
```

```
## # A tibble: 10,333 x 6
##    country  date iso3c.x         gdp population gdp_cap
##    <chr>   <dbl> <chr>         <dbl>      <dbl>   <dbl>
##  1 Aruba    1986 ABW      405586592.      62645   6474.
##  2 Aruba    1987 ABW      487709497.      61838   7887.
##  3 Aruba    1988 ABW      596648045.      61072   9770.
##  4 Aruba    1989 ABW      695530726.      61033  11396.
##  5 Aruba    1990 ABW      764804469.      62152  12305.
##  6 Aruba    1991 ABW      872067039.      64623  13495.
##  7 Aruba    1992 ABW      958659218.      68240  14048.
##  8 Aruba    1993 ABW     1083240223.      72495  14942.
##  9 Aruba    1994 ABW     1245810056.      76705  16242.
## 10 Aruba    1995 ABW     1320670391.      80324  16442.
## # ... with 10,323 more rows
```

We can do this in various ways: we can perform `left_join`, `right_join`, `inner_join`, or `outer_join`, but we can also use the `merge` function. Although their arguments differ somewhat, the results should absolutely be the same provided you specify the `by` arguments correctly. In that case, you match one particular country-year from the left data.frame to the identical particular country-year from the right data-frame, and put all variables together.

Question: Do this. Perform a merge and save the resulting data.frame to `merged_data`.

```
merged_data <- merge(data, pwt, by.x = c("iso3c.x", "date"), by.y=c("countrycode", "year"))
```

Next, we want to average GDP per capita and Capital stock for each country present in the dataset. This can be done easily using the `mutate` function from the `tidyverse` package. However, we want to investigate *current* GDP per capita, so we do not want to take too long an average.

Say we want to take an average over the years 2010-2020.

Question: Finish the following code to compute the average of GDP per capita and Capital stock. Make sure you deal with NA's explicitly. Save this again to `merged_data`

```
merged_data <- merged_data %>%
  group_by(country, iso3c.x) %>%
  filter(between(date, 2010, 2020)) %>%
  summarize(
    # fill in your answer here
    avg_gdpc = mean(gdp_cap, na.rm = TRUE),
    avg_cn = mean(cn, na.rm = TRUE)
  )
```

3. Historical Antecedents: Clio Infra Data

Another important element of capital might be not only physical capital, but human capital! Instead of using *contemporary* human capital to explain economic development in 2010-2020, we will use *historical* human capital. For this, we can again use a package, called `Clio`, which aggregates various historical datasets. You can install and load it by:

```
devtools::install_github("basm92/Clio")
library(Clio)
```

You can see what variables are available in this dataset by running:

```
Clio::clio_overview() %>% head(10)
```

```
##            variable_name from   to  obs
## 1      Cattle per Capita 1500 2010 7456
## 2    Cropland per Capita 1500 2010 6226
## 3       Goats per Capita 1500 2010 7037
## 4     Pasture per Capita 1500 2010 5963
## 5        Pigs per Capita 1500 2010 6841
## 6       Sheep per Capita 1500 2010 6835
## 7           Total Cattle 1500 2010 7457
## 8         Total Cropland 1500 2010 6191
## 9  Total Number of Goats 1500 2010 7037
## 10  Total Number of Pigs 1500 2010 6841
```

As a proxy to measure historical human capital, we'll use *Average Years of Education* in 1930. We can download this by running:

```
educ <- Clio::clio_get("Average Years of Education")
```

Question: Filter this dataset such that only observations from 1930 remain.

```
educ <- educ %>%
  filter(year == 1930)
```

Question: Merge this dataset to the `merged_data` set, so that the human capital proxy is added to the dataset.

When discussing economic growth, some people also talk about a *reversal of fortune* tendency: the countries that were relatively the most wealthy in or before the Middle Ages are among the poorest now, and vice versa.

	mean	median	min	max	N
avg_gdpc	15 798.78	6584.37	253.76	113 920.25	179
avg_cn	2 754 179.37	320 940.75	2556.14	75 221 956.00	176
av_educ	3.22	2.62	0.17	8.30	46
urban	0.09	0.08	0.00	0.32	46

We also want to investigate such an hypothesis. In order to do so, we use a proxy for wealth from 1500, the urbanization ratio. This can also be downloaded from the Clio Infra database.

Question: Now find and download Urbanization Ratio, filter the dataset such that only observations from 1500 remain, and save it to `urb`.

```
urb <- clio_get("Urbanization Ratio") %>%
  filter(year == 1500)
```

Now, let's merge `educ` and `urb` together, and then subsequently merge this to the `merged_data` data.frame.

```
educ_urb <- merge(educ, urb, by = "ccode")
```

Question: Also remove `year.x`, `year.y` and `country.name.y` from the dataset.

```
educ_urb <- educ_urb %>%
  select(-c(year.x, year.y, country.name.y))
```

Finally, we need to merge `educ_urb` to the `merged_data` data.frame.

Question: Merge these two dataframes using `left_join`, with `merged_data` being the left data.frame. Rename `Average Years of Education` and `Urbanization Ratio` to `av_educ` and `urban` respectively.

```
final <- merged_data %>%
  left_join(educ_urb,
            by = c("country" = "country.name.x")) %>%
  rename(av_educ = `Average Years of Education`, urban = `Urbanization Ratio`)
```

4. Summarizing and Analyzing the Data

Question: Create a descriptive statistics table using the variables we have obtained. In it, we want to display the mean, median, sd, min, max and number of observations. Hint: use the `modelsummary` package, and use the following syntax for the formula: `x1 + x2 + x3 + x4 ~ (mean + median + min + max)*Arguments(na.rm = TRUE) + N`.

```
library(modelsummary)

modelsummary::datasummary(data = final, avg_gdpc + avg_cn + av_educ + urban ~
                            (mean + median + min + max)*Arguments(na.rm = TRUE) + N)
```

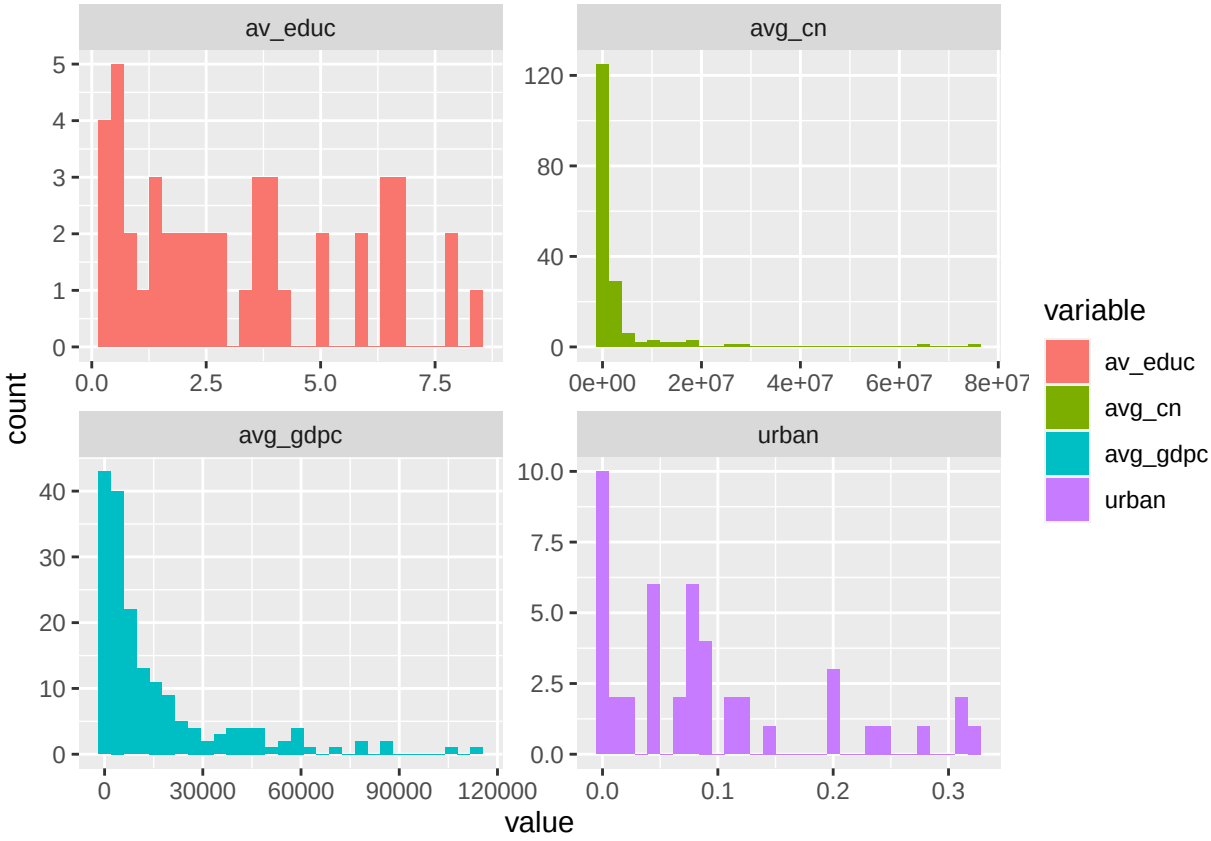
Next, we would like to make a histogram of the four aforementioned variables, to investigate their distribution. To do so, we need data in so-called long-form:

```
final_long <- final %>%
  pivot_longer(c(avg_gdpc, avg_cn, av_educ, urban), names_to = "variable", values_to = "value")
```

With this dataset, we can create a histogram with four facets:

Question x: Add the right geometry element to complete the histograms.

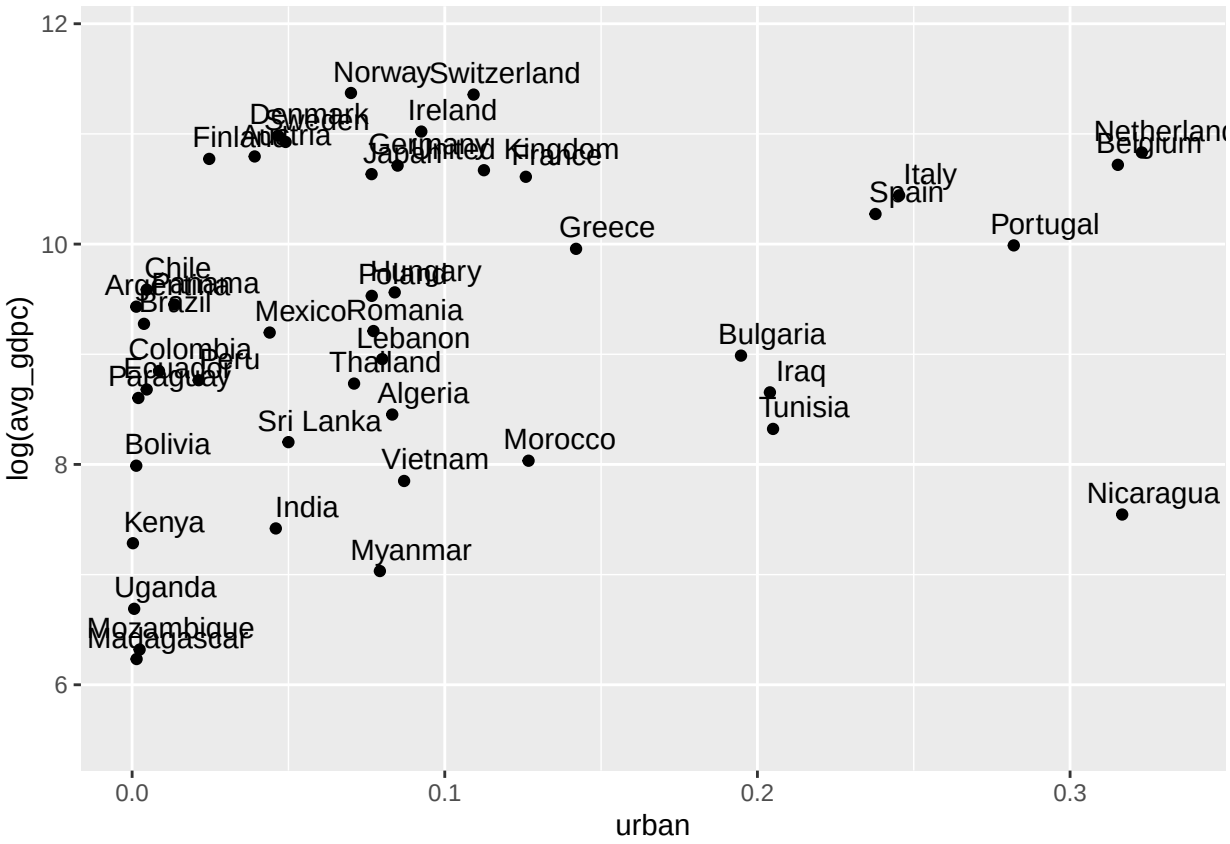
```
final_long %>%
  ggplot(aes(x = value, group = variable, fill = variable)) +
  facet_wrap(~variable, scales = "free") +
  # fill in the right answer
  geom_histogram()
```



Next, we would like to make a plot of some of the data. In particular, we can use the `ggplot` library to create a scatterplot of urbanization in 1500 on the x axis and $\log$ (GDP per capita) on the y axis. That would allow us to get insight into the *reversal of fortune* theory.

Question: Use `ggplot` to create a plot as described above. Hint: use `geom_point()` as the geometric attribute. Also try to see if you can display the country name corresponding to each dot.

```
final %>%
  ggplot(aes(x = urban, y = log(avg_gdpc))) +
  geom_point() +
  geom_text(aes(label=country), nudge_x = 0.01, nudge_y = 0.2)
```

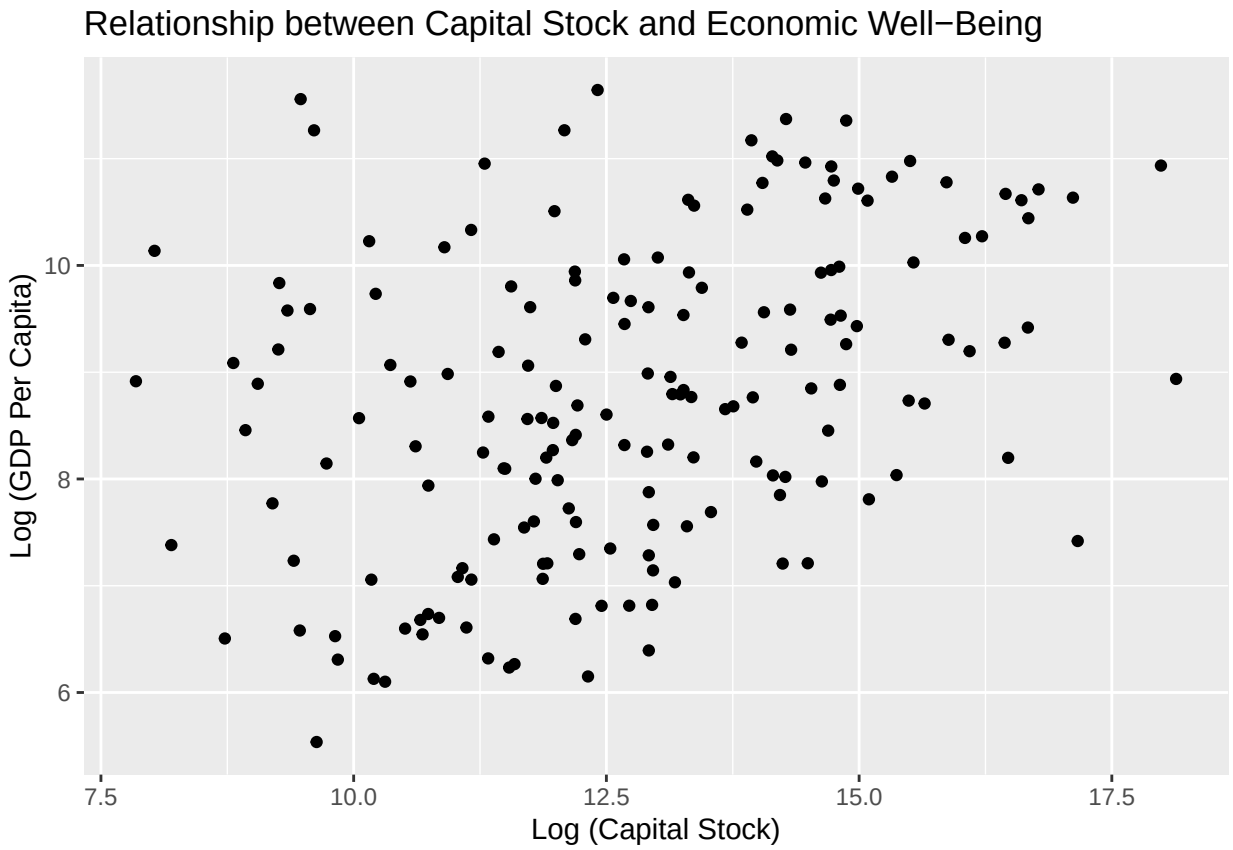


Question: Do you interpret this as evidence for the *reversal of fortune* theory? Why (not)?

Secondly, we can plot GDP per capita (y-axis) against the capital stock (x-axis). In this case, we want to log-transform both of the variables.

Question: Construct this plot. Also make sure to add a nice title and to change the axis titles appropriately.

```
final %>%
  ggplot(aes(x=log(avg_cn), y = log(avg_gdpc))) + geom_point() +
  xlab("Log (Capital Stock)") +
  ylab("Log (GDP Per Capita)") +
  ggtitle("Relationship between Capital Stock and Economic Well-Being")
```



Question: Do you interpret this as evidence for the logic of most macroeconomic models, that a higher capital stock causes a higher level of income? Why (not)?

Finally, we would also like to make a map displaying economic growth rates. In order to do so, we need the `sf` package, short for *Spatial Features*. This is an efficient format in which data used to construct maps are stored. We also need a couple of auxiliary packages:

```
library(tidyverse); library(sf)
```

Possibly, we also have to install a couple of auxiliary packages:

```
pacman::p_load("rgdal", "rgeos", "lwgeom")
```

Let us first find a map of the world:

```
library(maps)
world <- st_as_sf(map("world", plot = FALSE, fill = TRUE))
```

We converted the world map from the `maps` package to an `sf` data.frame.

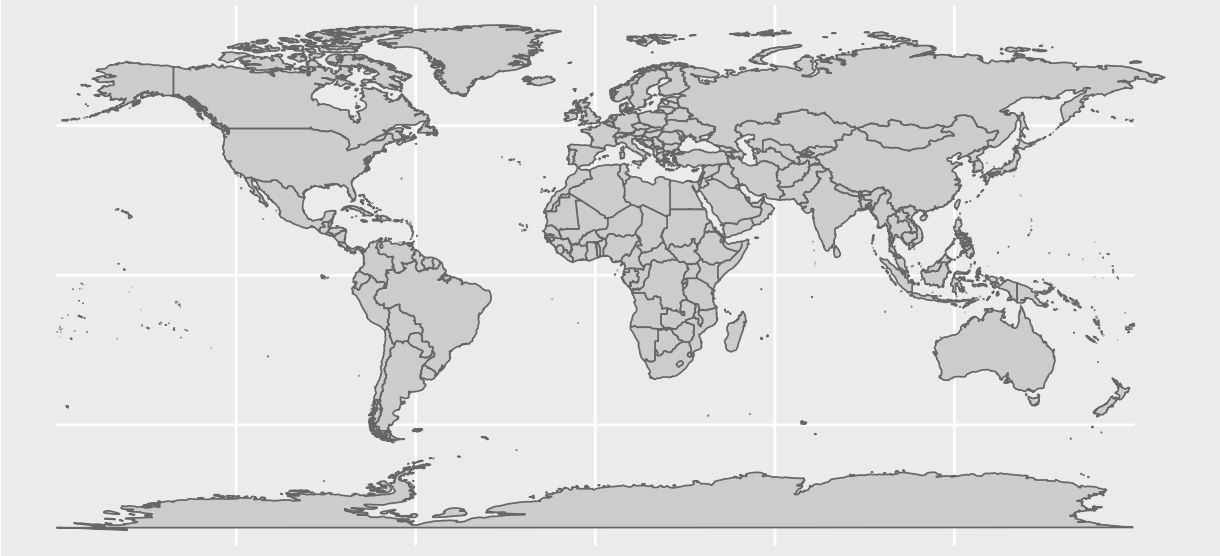
```
world
```

```
## Simple feature collection with 253 features and 1 field
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -180 ymin: -85.19218 xmax: 190.2708 ymax: 83.59961
## Geodetic CRS:  WGS 84
## First 10 features:
##                      ID                           geom
```

```
## 1                 Aruba MULTIPOLYGON (((-69.89912 1...
## 2           Afghanistan MULTIPOLYGON (((74.89131 37...
## 3                Angola MULTIPOLYGON (((23.9665 -10...
## 4              Anguilla MULTIPOLYGON (((-63.00122 1...
## 5               Albania MULTIPOLYGON (((20.06396 42...
## 6               Finland MULTIPOLYGON (((20.61133 60...
## 7               Andorra MULTIPOLYGON (((1.706055 42...
## 8  United Arab Emirates MULTIPOLYGON (((53.92783 24...
## 9             Argentina MULTIPOLYGON (((-64.54916 -...
## 10              Armenia MULTIPOLYGON (((45.55235 40...
```

`world` is now a data.frame, countaining the names of countries and associated polygons. We can use this object to create a simple map:

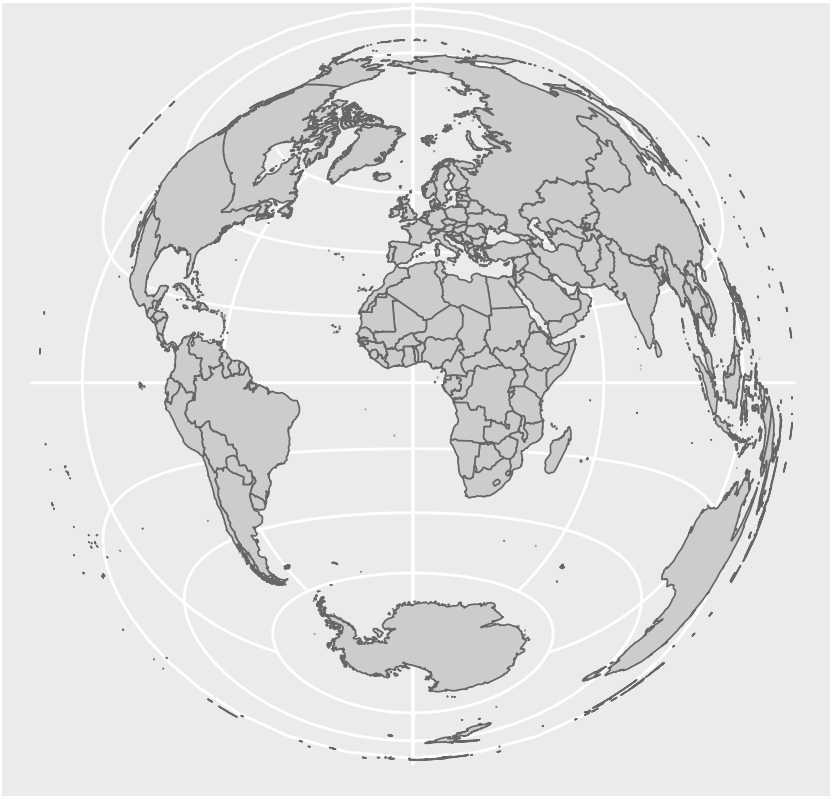
```
world_map <- ggplot(world) +
  geom_sf(fill = "grey80",
          col = "grey40",
          lwd = 0.3)

world_map
```



It is also possible to change projections. Here is a short primer on different projections. For you, this is not particularly relevant, but it allows you to pick a projection which you like. Here's an example:

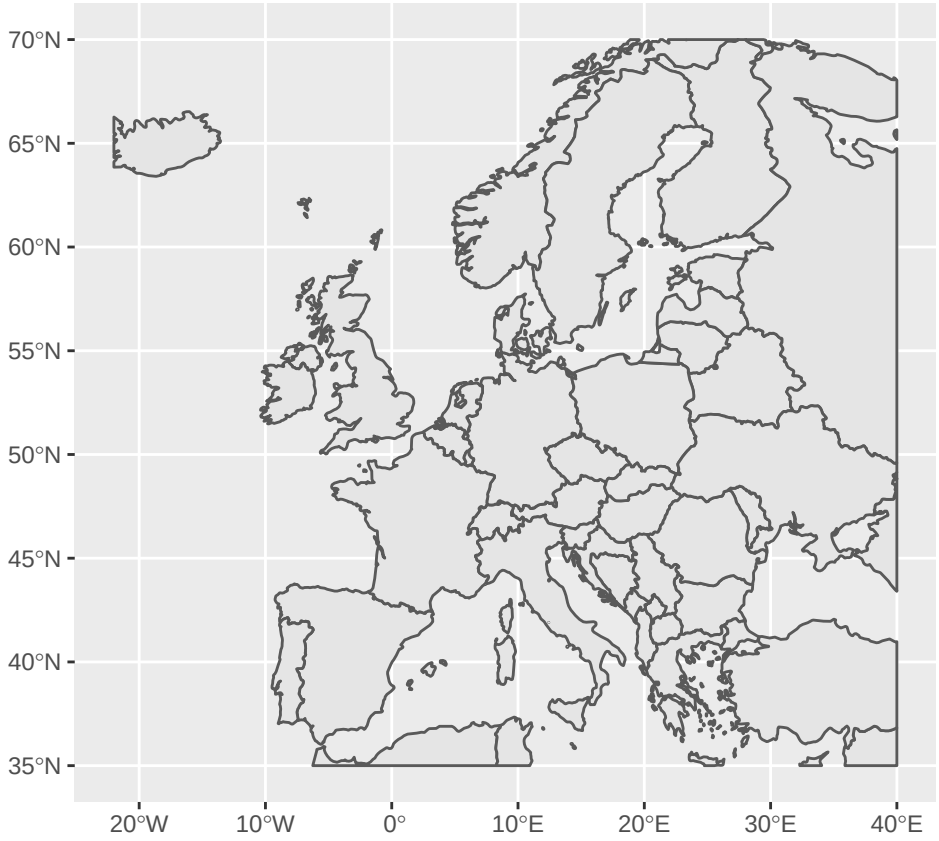
```
world_map +
  coord_sf(crs = "+proj=laea +y_0=0 +lon_0=0 +lat_0=0") +
  labs(subtitle = "Lambert Azimuthal Equal Area projection")
```

Lambert Azimuthal Equal Area projection



Should we want to zoom in on a particular part of the world, that is also possible. To do that, we can filter the dataframe based on many features, for example, on coordinates:

```
sf_use_s2(FALSE)

world %>%
  st_crop(xmin = -22,
          ymin = 35,
          xmax = 40,
          ymax = 70) %>%
  ggplot() + geom_sf()
```

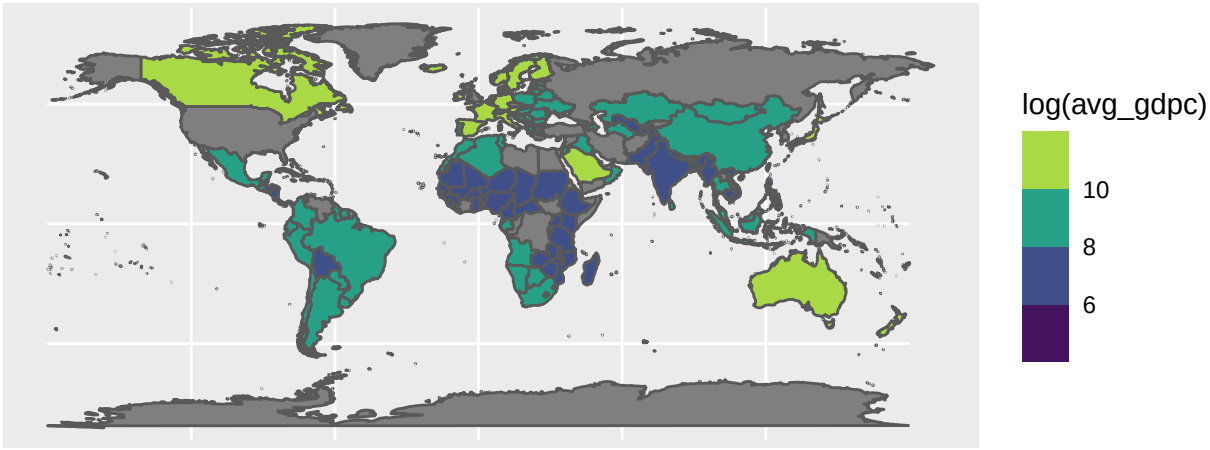


Question x: Take your `data.frame` world, and merge it with the `final` data.frame containing the GDP growth rates. Save this to a data.frame called `world_data`.

```
world_data <- world %>%
  left_join(final, by = c("ID" = "country"))
```

Now, plot a map displaying `avg_gdpc` in different countries. Hint: use the `geom_sf(fill = avg_gdpc)` as geometric attribute.

```
world_data %>%
  ggplot() + geom_sf(aes(fill = log(avg_gdpc))) + scale_fill_viridis_b()
```



The End