Seminar 4 – week 39

I. GDP across the world

The Penn World Tables provides GDP data that are comparable across time and between countries. In the comma separated file `pwt.csv` (available in Canvas), there are data on GDP per capita (the variable `rgdpna` divided by `pop`) as well as country codes and names and a year variable.

1) Download the data to your computer. Import the data into a tibble named `pwt` in R. Do this both using a script and the import function in RStudio.
2) Make a new tibble `gdp_nor` with GDP for Norway for the period 1980-2000. The tibble should only contain the variables year and GDP.
3) Use a combination of `filter`, `mutate`, and `lag` to extract annual growth rates for China for the period 1970-2000. Store the data in a new tibble `pwt.china`.

The Solow growth model tells us that poorer countries should grow faster than richer countries (ceteris paribus). To investigate this, we want to compare log GDP in 1970 with average annual growth in the period 1970-2000.

1) Construct a variable `lgdp` with the log of GDP. Keep only observations for the years 1970 and 2000, and use `pivot_wider` to obtain one country per row, hence observations for 1970 and 2000 in separate vectors. Construct a new variable named `growth` with annual economic growth in the period. Store the data in a new tibble `pwt.growth`.
2) Make a scatter plot of growth versus initial log GDP using ggplot2.
3) Create a dummy variable (0/1 variable) indicating whether GDP has increased from last year or not. Use a combination of `group_by` and `summarise` to find the fraction of years GDP has been increasing by year.
   Hint: The `na.rm` option to the `mean` is important when there are missing data
4) Compare the fraction of growth years in the period 1970-2000 to total growth in percent over the period.

III. Proportional elections

In proportional elections, i.e. elections where multiple candidates are to be elected from each district, it is necessary to have a rule to go from electoral results to the number of representatives each party gets. In this exercise, we implement some of these rules.

One common rule is the D'Hondt system. The first representative is given to the party with the largest number of votes. For the next step, this party's number of votes is divided by 2 and compared to the others. The second representative is given to the party with the largest number of votes, taking into account the shrinking of the first party. This process is repeated until all representatives are allocated. In each step, the relevant number of votes to a party is their actual number divided by one plus the number of representatives they already have.

1) Construct an algorithm that takes a vector with the number of votes to each party and produces the number of representatives allocated to each party. Implement the algorithm in R.
2) The city council of Oslo has 59 members. The number of votes in the 2023 election are given in the table on the last page. Make an R script to compute the number of representatives to each party under the D'Hondt rule (disregarding personal votes and other complicating factors).
   Hint: You need to import the table into a vector – copy/paste would work. Then the algorithm from 1) could be implemented in a for-loop.
3) The file `election.csv` (in Canvas) contains the number of votes in the regional elections in the 2019 election. This file is equivalent to the data you constructed in Exercise I, question 7). Import the data to a tibble. The variable `repr` is the number of members in each of the regional councils. Find the number of representatives for each party in each region.
   Hint: The simplest way is to loop over each region, extract the vector of the number of votes and the size of the council and reuse the code from 3). A more challenging approach is to vectorize the code.
4) (Optional) There are various rules to allocate seats besides D'Hondt's method. One common approach is the Sainte-Laguë method. Instead of dividing by (1+Number of seats), this approach implies dividing by (1+2*Number of seats) yielding divisors 1, 3, 5, etc. Repeat exercise 2 using this algorithm.
   Norway uses a modified Sainte-Laguë, where the divisors are 1.4, 3, 5, etc. Try to implement this approach as well.
   Do the different allocation mechanisms give different outcomes?

If you want to dig even further, you can also try to implement the systems with person votes, see valg.no. In empirical research, there is also approaches where we need to see how many votes we need to change before changing the electoral outcome. If this is a small number, we can say that the election is close and the outcome almost random. See Folke (2014) for a seminal contribution.

Party	Votes
Høyre	117 998
Arbeiderpartiet	66 804
Miljøpartiet De Grønne	36 975
SV - Sosialistisk Venstreparti	36 612
Venstre	32 954
Fremskrittspartiet	21 976
Rødt	20 929
Kristelig Folkeparti	6 220
Partiet Sentrum	4 461
Industri- og Næringspartiet	4 035
Senterpartiet	2 989
Pensjonistpartiet	2 054
Folkestyret-listen	1 995
Folkets Parti	1 948
Norgesdemokratene	982
Partiet Mot Bompenger	883
Konservativt	775
Liberalistene	626
Alliansen - Alternativ for Norge	468
Norges Kommunistiske Parti	394
Kystpartiet	307