DAT565/DIT407 Assignment 1

Sebastian Miles
miless@chalmers.se

Olle Lapidus
lapis@chalmers.se

2024-09-06

Problem 1

The goal of this assignment is to analyze the data downloaded from SCB [**SCB:2023**]. The data contains the number of people in Sweden by age and sex from 1860 to 2022.

Task I

The first task is to calculate the Dependency ratio of Sweden from 1860 to 2022. The dependency ratio is calculated as

$$\begin{aligned}\text{Dependency Ration } &= 100 \cdot \frac{\#\text{children} + \#\text{elderly}}{\#\text{labor force}} \\ &= 100 \cdot \frac{\#\text{people aged 0-14} + \#\text{people aged 65-}}{\#\text{people aged 15-64}}.\end{aligned}$$

The result can be viewed in Figure 1.

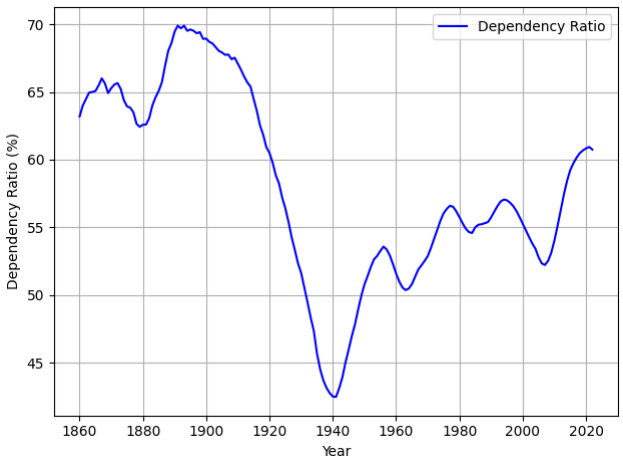

Figure 1: Dependency Ratio Over Time in Sweden (1860-2022)

Task II

In this task we calculate the ratio of children, workforce and elderly respectively and plot it from the same data used earlier. The result is show in Figure 2.

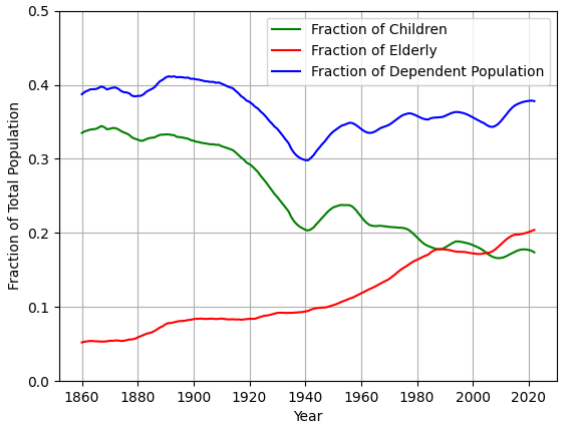

Figure 2: Fractions of Children, Elderly, and Total Dependent Population in Sweden (1860-2022).

Task III

Here we discuss the development of the Swedish population by observing Figure 1 and Figure 2. With advancements within the medical field people tend to live for longer so it is very natural for the fraction of elderly to increase. The fraction of children has decreased as well because of better access to contraceptive devices, higher costs of raising children and drifting societal values towards having smaller families. World War I and II caused both societal and economical changes which is clearly reflected in both figures above.

Problem 1 Code

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Read from file
5  df = pd.read_csv('swedish_population_by_year_and_sex_1860-2022.csv')
6
7
8  # Cast age column to integer
9  df['age'] = pd.to_numeric(df['age'], errors='coerce')
10
11 # Extract age groups
12 children = df[df['age'] <= 14]
```

```
workforce = df[(df['age'] > 14) & (df['age'] <= 64)]
elderly = df[df['age'] > 64]

# Compute the group populations
total_children  = children[children.columns[2:]].sum()
total_workforce  = workforce[workforce.columns[2:]].sum()
total_elderly  = elderly[elderly.columns[2:]].sum()

dependency_ratio = 100*(total_children+total_elderly)/(total_workforce)

plt.plot(dependency_ratio.index, dependency_ratio.values, label='Dependency
plt.xlabel('Year')
plt.ylabel('Dependency Ratio (%)')
plt.grid(True)
plt.legend()
plt.gca().xaxis.set_major_locator(plt.MaxNLocator(nbins=10))

plt.show()

# Compute the total population across all ages
total_population = total_children + total_workforce + total_elderly

# Compute the ratio of each group
ratio_children  = total_children/total_population
ratio_elderly  = total_elderly/total_population
ratio_dependent  = (total_elderly+total_children)/total_population

plt.plot(ratio_children.index, ratio_children.values, label='Fraction of Ch
plt.plot(ratio_elderly.index, ratio_elderly.values, label='Fraction of Elde
plt.plot(ratio_dependent.index, ratio_dependent.values, label='Fraction of
plt.xlabel('Year')
plt.ylabel('Fraction of Total Population')
plt.ylim(0,0.5)
plt.grid(True)
plt.legend()
plt.gca().xaxis.set_major_locator(plt.MaxNLocator(nbins=10))

plt.show()\\
```