Econ UH 1112: Introduction to Macroeconomics
GDP report: Germany

Atith Adhikari
Spring 2026

I am studying Germany. Table 1 lists a brief summary of the country's characteristics.

Table 1: Overview of Chosen Country

Country	Currency
Germany	Euro

Problem 1

Summary

For this problem, I found data at the German Federal Statistical Office Website, accessible at `https://destatis.de`[1]. The precise labels of the downloaded series is provided in Table 2.

I have collected the data for the time period 2009-2025 in Germany's local currency, Euro.

Table 2: Description of Data Series

Description	Identifier	Units	Frequency
nominal GDP	PAGDP	billions of Euro	annual

Part 1.1-1.3

Please refer to the Appendix for the code for data analysis.

Part 1.4

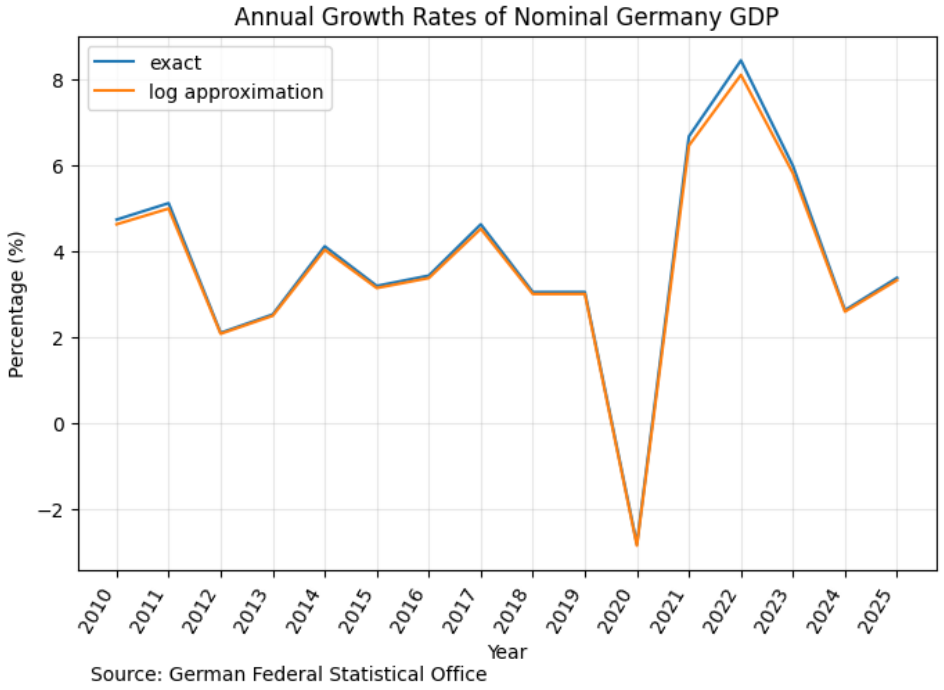
Let y_t denote the nominal GDP at time t.

The exact growth rate of nominal GDP, γ_t^e, is:

$$\gamma_t^e = \left(\frac{y_t - y_{t-1}}{y_t} \right) \cdot 100 \tag{1}$$

[1] `https://www.destatis.de/EN/Themes/Economy/Short-Term-Indicators/IMF/national-accounts-sdds-plus.html`

Figure 1: Exact and Approximate Annual Growth Rates for Germany GDP, 2010 - 2025



The log approximation growth rate of nominal GDP, γ_t^a, is:

$$\gamma_t^a = 100 \cdot (\ln y_t - \ln y_{t-1}) \tag{2}$$

Part 1.5

As tabulated in Table 3 and depicted in Figure 1, the log approximation is always less than or equal to the exact growth rates.

Table 3: Comparison of exact and approximate growth rates for Germany GDP, (2010 - 2025)

Name	average annual growth rate	minimum annual growth rate	maximum annual growth rate
nominal GDP (exact)	3.76	-2.81	8.43
nominal GDP (log)	3.66	-2.85	8.09

The reason why logarithmic approximation to computing the growth rates works:

We derive the mathematical formula for logarithmic approximation through percentage change formula.

From equation 1,

$$y_t = y_{t-1} \cdot \left(1 + \frac{\gamma}{100}\right)$$

$$\left(1 + \frac{\gamma}{100}\right) = \frac{y_t}{y_{t-1}}$$

Taking the natural log on both sides of the equation, we get,

$$\ln\left(1+\frac{\gamma}{100}\right) = \ln y_t - \ln y_{t-1}$$

We use the approximation that for small values of $\frac{\gamma}{100}$, $\ln\left(1+\frac{\gamma}{100}\right) \approx \frac{\gamma}{100}$; hence, we obtain equation 2.

$$\gamma_t^a = 100 \cdot (\ln y_t - \ln y_{t-1})$$

Because the log approximation implicitly assumes that the actual growth rate is very small, the approximation yields a value closer to the exact values for smaller growth rates. We can observe in Figure 1 that for higher growth rates, the difference between the exact and approximate growth rates are noticeable than for lower growth rates.

Part 1.6

Please refer to Table 3.

Problem 2

Summary

For this problem, I found data at the World Bank Data Website, accessible at `https://data.worldbank.org`[2]. The precise labels of the downloaded series is provided in Table 4. I have collected the data for the time period 1960-2024 in Germany's local currency, Euro.

Table 4: Description of Data Series

Description	Identifier	Units	Frequency
nominal GDP	nGDP	Euro	annual
real GDP	rGDP	Euro	annual
Consumer Price Index	CPI	Euro	annual

Part 2.1

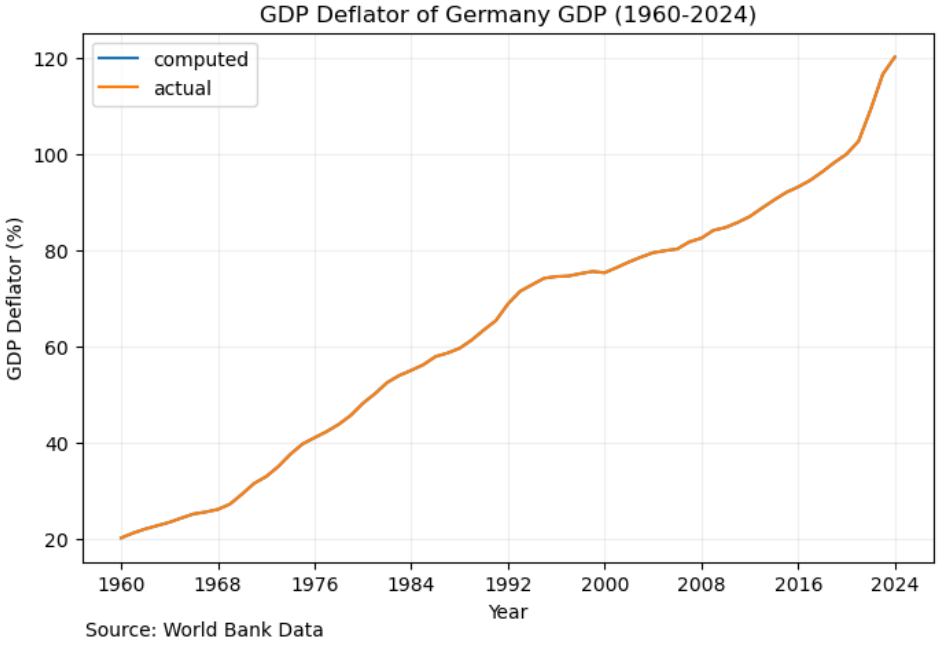
To compute the GDP deflator for a year i, we use the following formula:

$$\text{GDP Deflator} = \left(\frac{nGDP_i}{rGDP_i}\right) \cdot 100 \tag{3}$$

[2]

- Real GDP: `https://data.worldbank.org/indicator/NY.GDP.MKTP.KN?locations=DE`
- Nominal GDP: `https://data.worldbank.org/indicator/NY.GDP.MKTP.CN?locations=DE`
- Consumer Price Index (CPI): `https://data.worldbank.org/indicator/FP.CPI.TOTL?locations=DE`

Figure 2: Actual and Computed GDP Deflators for Germany GDP, 1960 - 2024



The time-series graphs (Figure 2) of the actual and computed GDP Deflators completely overlap. This suggests that the original producer of the data should have used the Equation 3 to compute it.

Part 2.2

We use the percentage formula[3] to obtain the percentage growth in real and nominal GDP from 1960 to 2024. Similarly, we use the Average Annual Growth Formula (AAGR) [4] to compute the AAGR for the same period.

Table 5: Nominal and Real GDPs of Germany in Year 1960 and 2024

$nGDP_{1960}$	$nGDP_{2024}$	$rGDP_{1960}$	$rGDP_{2024}$
181.72	4328.97	898.94	3600.83

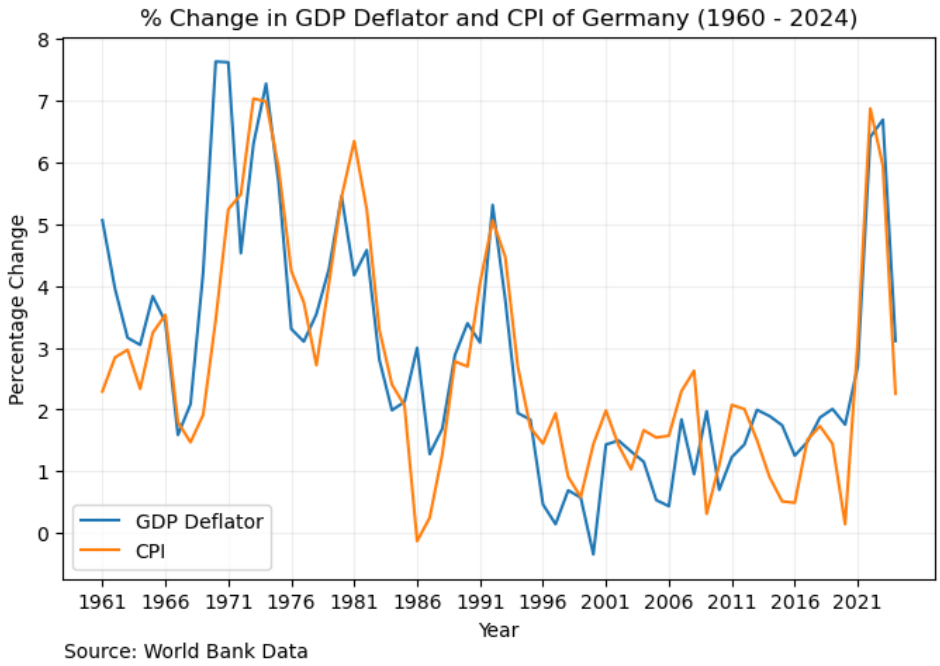
Table 6: Percentage growth and Average annual growth of Nominal and Real GDPs of Germany (1960-2024)

	rGDP	nGDP
% growth	300.56	2282.28
AAGR[5]	2.16	5.00

[3] $\% = \frac{\text{final value} - \text{initial value}}{\text{initial value}} \cdot 100$

[4] $r = \left(e^{\frac{\ln\left(\frac{y_t}{y_0}\right)}{t}} - 1 \right) \cdot 100$

Figure 3: Percentage Change in GDP Deflator and CPI of Germany, 1960 - 2024



Part 2.3

The average annual growth rate of the GDP Deflator is calculated to be: 2.78%. This value is between the average annual growth rates of real GDP (2.16%) and nominal GDP (5.00%).

$$5.00 \approx 2.16 + 2.78$$

The growth in nominal GDP of Germany during 1960 - 2024 is contributed more than half by inflation (increase in prices of products) than the increase in production itself. However, the economy has also seen a moderate growth of 2.16% over 65 years.

Part 2.4

From Figure 3, we observe that GDP Deflator and CPI generally move in the same direction. However, there are noticeable differences in the graphs, especially during 1981, 1986, and 2020. In the post-pandemic time, the prices of production and consumption in Germany rose sharply and are re-adjusting to the average level of 1-3%.

Appendix

Python Code for Problem 2

```
# set up the environment by adding the required libraries
import requests
from bs4 import BeautifulSoup
import pandas as pd
import matplotlib.pyplot as plt

# destination URL for data extraction
url = "https://www.destatis.de/EN/Themes/Economy/Short-Term-Indicators/IMF/national-acco

# send an HTTP request to the url
response = requests.get(url)
print(response.status_code)

# parse the response data by html tags
data = BeautifulSoup(response.text, 'html.parser')
print(data)

# target the first table and its body
table = data.find('table')
tbody = table.find('tbody')

# We compute and store the quarterly GDPs for Germany in a dictionary.
gdp_dict = {}
for idx, tr in enumerate(tbody):
  year = tr.find('th').text
  is_year_end = year.isdigit()
  gdp = float(tr.find('td').text.replace(',', ''))
  if is_year_end:
    gdp_dict[year] = [ gdp ]
    year_prev = year
  else:
    gdp_dict[year_prev].append(gdp)

print(gdp_dict)

# Now, we compute the annual GDP for each year by adding the quarterly GDPs for Germany.
gdp_annual_dict = {}
for year, gdp in gdp_dict.items():
  gdp_annual_dict[year] = sum(gdp)
```

```
print(gdp_annual_dict)

# plot the time-series graph for nominal gdp
plt.plot(list(gdp_annual_dict.keys())[::-1], list(gdp_annual_dict.values())[::-1])
plt.xlabel("Year", fontsize=14)
plt.xticks(rotation=60, ha='right')
plt.title("GDP: Germany (2009-2025)")
plt.ylabel("GDP (EUR bn)", fontsize=14)
plt.show()

# take the log of annual gdp
import math
gdp_annual_log_dict = {}
for year, gdp in gdp_annual_dict.items():
  gdp_annual_log_dict[year] = math.log(gdp)
print(gdp_annual_log_dict)

# computing the percentage growth rates
gdp_growth = {}

gdp_years = list(gdp_annual_dict.keys())[::-1]
print(gdp_years)
for index, (key, value) in enumerate(list(gdp_annual_dict.items())[::-1]):
    if index != 0:
      curr_key = key
      prev_key = gdp_years[index - 1]
      diff = gdp_annual_dict[curr_key] - gdp_annual_dict[prev_key]
      pct = diff / gdp_annual_dict[prev_key] * 100
      gdp_growth[curr_key] = pct
print(gdp_growth)

# computing approximate log growth rates
gdp_growth_est = {}
gdp_years = list(gdp_annual_log_dict.keys())[::-1]
print(gdp_years)
for index, (key, value) in enumerate(list(gdp_annual_log_dict.items())[::-1]):
    if index != 0:
      curr_key = key
      prev_key = gdp_years[index - 1]
      growth = (gdp_annual_log_dict[curr_key] - gdp_annual_log_dict[prev_key]) * 100
```

```
      gdp_growth_est[curr_key] = growth
print(gdp_growth_est)

# plot the graph of actual vs estimated growth of nominal GDP
plt.figure(figsize=(8,5))
plt.plot(list(gdp_growth.keys()), list(gdp_growth.values()), label="exact")
plt.plot(list(gdp_growth_est.keys()), list(gdp_growth_est.values()), label="log approxim
plt.ylabel("Percentage (%)")
plt.xlabel("Year")
plt.xticks(rotation=60, ha="right")
plt.grid(True, alpha=0.3)
plt.legend(loc='best')
plt.title("Annual Growth Rates of Nominal Germany GDP")
plt.text(-0.5, -6, "Source: German Federal Statistical Office")

# Data Summary
import numpy as np
actual_growth = np.array(list(gdp_growth.values()))
estimated_growth = np.array(list(gdp_growth_est.values()))
print(f"The average actual GDP growth rate over the given period is {actual_growth.mean(
print()
print(f"The average estimated GDP growth rate over the given period is {estimated_growth
```

Python Code for Problem 2

```
# import the required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import math

# load the data sets into pandas dataframe data structure
df_cpi = pd.read_csv("cpi.csv")
df_rgdp = pd.read_csv("rgdp.csv")
df_ngdp = pd.read_csv("ngdp.csv")
df_defl = pd.read_csv("gdp_deflator.csv")

# exatract data specific to germany
cpi_germany = df_cpi[df_cpi['Country Name'] == 'Germany']
ngdp_germany = df_ngdp[df_ngdp['Country Name'] == 'Germany']
rgdp_germany = df_rgdp[df_rgdp['Country Name'] == 'Germany']
defl_germany = df_defl[df_defl['Country Name'] == 'Germany']

# populate a list with years in consideration
years = [ str(year) for year in range(1960, 2025) ]

# calculate the gdp deflation for each year using the formula: defl = ngdp/rgdp * 100
defl_computed = {}
for year in years:
    defl_computed[year] = (ngdp_germany[year] / rgdp_germany[year]) * 100

# extract the values for each year and store as a list
ngdp_values = [ ngdp_germany[year] for year in years ]
rgdp_values = [ rgdp_germany[year] for year in years ]
cpi_values = [ cpi_germany[year] for year in years ]
deflator_values = [ defl_germany[year] for year in years ]

# plot the grpah of the computed and actual gdp deflator for germany
defl_germany_values = [ defl_germany[year] for year in years ]
plt.figure(figsize=(8,5))
plt.plot(years, list(defl_computed.values()), label="computed")
plt.plot(years, defl_germany_values, label="actual")
plt.grid(alpha=0.2)
plt.xlabel(xlabel="Year")
```

```
plt.ylabel(ylabel="GDP Deflator (%)")
plt.title("GDP Deflator of Germany GDP (1960-2024)")
plt.xticks(ticks=years[::8], rotation=0, ha='center')
plt.text(-3, 0, s="Source: World Bank Data")
plt.legend()
plt.show()

# first and latest values of rgdp, ngdp
print(f"The value of rGDP in 1960 was {rgdp_values[0]} and the value of rGDP in 2024 was
print(f"The value of nGDP in 1960 was {ngdp_values[0]} and the value of nGDP in 2024 was

# compute the aagr of gdp deflator
aagr_gdp_defl = (math.e**(math.log(deflator_values[-1]/deflator_values[0]) / len(years))
aagr_gdp_defl

# compute the percentage change in gdp deflator
deflator_change = []
for idx, year in enumerate(years):
    if idx != 0:
        deflator_change.append( (deflator_values[idx] / deflator_values[idx - 1] - 1) *

# compute the percentage change in cpi
cpi_change = []
for idx, year in enumerate(years):
    if idx != 0:
        cpi_change.append( (cpi_values[idx] / cpi_values[idx - 1] - 1) * 100 )

# plot the gdp deflator vs cpi percentage change graph
plt.figure(figsize=(8,5))
plt.plot(years[1:], deflator_change, label="GDP Deflator")
plt.plot(years[1:], cpi_change, label="CPI")
plt.xticks(ticks=years[1:-1:5])
plt.xlabel("Year")
plt.ylabel("Percentage Change")
plt.grid(alpha=0.2)
plt.title("% Change in GDP Deflator and CPI of Germany (1960 - 2024)")
plt.text(x=-3, y=-2, s="Source: World Bank Data")
plt.legend()

# convert ngdp and rgdp values into floating points for data computation
```

```
ngdp_values = [ float(value) for value in ngdp_values ]
rgdp_values = [ float(value) for value in rgdp_values ]

pct_rgdp_growth = (rgdp_values[-1]/rgdp_values[0] - 1)*100
pct_ngdp_growth = (ngdp_values[-1]/ngdp_values[0] - 1)*100
aagr_ngdp_growth = ((ngdp_values[-1]/ngdp_values[0])**(1/len(ngdp_values)) - 1)*100
aagr_rgdp_growth = ((rgdp_values[-1]/rgdp_values[0])**(1/len(rgdp_values)) - 1)*100

print(f"Percentage growth:\trGDP: {pct_rgdp_growth}\tnGDP: {pct_ngdp_growth}")
print(f"AAGR growth:\trGDP: {aagr_rgdp_growth}\tnGDP: {aagr_ngdp_growth}")
```