Macro_assignment_3

November 11, 2018

1 Assignment 3

By Chris Hayes, Ismael Moreno Martinez, Filip Mellgren. We present our replication in a Jupyter notebook, meaning that the code is integrated with comments and the equations we use to motivate our code. The final table is presented at the bottom.

We begin by loading some packages into Python:

```
In [1]: import pandas as pd
        import os

        os.chdir("/Volumes/GoogleDrive/Min enhet/Learning/MSc/Macro")
```

1.0.1 Data source

Apx B points to a prepared data set, which we dowloaded from: https://www.sciencedirect.com/science/article/pii/S0304393208001402?via%3Dihub (Log in to ScienceDirect, download the spreadsheet under "Extras").

Some slight manipulation were made in a google sheet to make it easier for the python console to read. These manipulations were: deleting empty rows at the top of the sheet and putting variable names at the top row, deleting the descriptor row

1.0.2 Loading the data into Python

We iterated over the sheets of the Excel file to create a data frame object for each sheet. We then added a multiindex so that we would have a meaningful unique identifier for all combinations of countries and years.

Next, we created a decade variable (since the analysis groups the data by decades).

```
In [2]: # The data was stored with each country in a separate sheet, named the following:
        country_list = ["AUS", "BEL", "AUT", "CAN", "FIN", "FRA", "GER", "ITA", "JAP",
                        "NETH", "UK", "US", "SPA", "SWE", "SWI"]

In [3]: # This loop loads the data into a dataframe and creates a multiindex based on
        # country identifier (from sheet name) and time.
        df = {}
        for country in country_list:
            df[country] = pd.read_excel("ORRdata.xlsx", sheet_name = country)
            df[country]["country"] = country
```

```
        df[country] = df[country].dropna()
        df[country]["YEAR"] = df[country]["YEAR"].astype("int32")
        df[country]["YEAR"] = pd.to_datetime(df[country]["YEAR"],
          format="%Y", errors="coerce")
        df[country] = df[country].set_index(["YEAR", "country"])

    data = pd.concat([df["AUS"], df["BEL"], df["AUT"], df["CAN"], df["FIN"],
                      df["FRA"], df["GER"], df["ITA"], df["JAP"],
                      df["NETH"], df["UK"], df["US"], df["SPA"], df["SWE"], df["SWI"]],
                     sort = True)

    data.sort_index(level=["country", "YEAR",], ascending=True, inplace=True)
```

```
In [4]: ####  Variable manipulation:
        # Turn the "L = (H*E)/P" variable into a numeric "L".
        # Non numeric values are set to missing:
        data["L"] = pd.to_numeric(data["L = (H*E)/P"], errors = "coerce")

        # Add a decade variable
        data["decade"] = (data.index.get_level_values('YEAR').year//10)*10
```

1.1 Actual changes

Having prepared the data set, we calculated actual changes to confirm that we thus far handled all the data correctly. These values can then be compared with the author's table 2.

```
In [5]: #### Actual change ####
        # Calculate change
        data["L_change"] = (data["L"] - data["L"].shift(1))/(data["L"].shift(1))

        # Calculate average change, grouping by decade and country
        table2 = data["L_change"].groupby([data.index.get_level_values('country'),
                    data["decade"]]).mean()
        table2 = (table2.unstack()*100).round(2)


        data_post_1956 = data[data.index.get_level_values("YEAR").year > 1956]
        table2["1956-2003"] = data_post_1956["L_change"].groupby(data_post_1956.index.get_level
        table2["1956-2003"] =  (table2["1956-2003"]*100).round(2)

        # Remove redundant columns
        table2 = table2[[1960, 1970, 1980, 1990, '1956-2003']]
        table2 = table2.sort_index()
```

```
In [6]: %%latex
        \newpage
```

```
In [7]: table2 # Result for actual changes
```

```
Out[7]: decade   1960  1970  1980  1990  1956-2003
        country
        AUS      0.11 -0.53  0.49 -0.01       0.01
        AUT     -1.03 -1.14 -0.22 -0.53      -0.64
        BEL     -1.03 -1.64 -0.74  0.06      -0.72
        CAN      0.18  0.34  0.70 -0.16       0.17
        FIN     -1.15 -0.45  0.32 -1.31      -1.54
        FRA     -0.63 -1.47 -1.53  0.11      -0.87
        GER     -1.09 -1.55 -1.20 -0.74      -1.08
        ITA     -1.60 -1.19 -0.43 -0.16      -0.61
        JAP     -0.39 -0.47  0.14 -0.74      -0.31
        NETH    -0.65 -1.81 -0.95  1.40      -0.44
        SPA      0.25 -1.67 -1.56  0.85      -0.28
        SWE     -1.00 -0.36  0.54 -0.71      -0.48
        SWI     -0.23 -1.26  0.31  0.27      -0.38
        UK      -0.42 -1.17 -0.22 -0.34      -0.48
        US       0.29 -0.25  0.73  0.46       0.03
```

The values look fine, meaning that we are on the right track. Let's move on to replicate the predicted changes.

1.2 Predicted change

From here on, we focus on deriving the model predicted values.

1.2.1 Parameters given

To begin with, we were given the following parameters:

$\bar{H} = 5110$, reflecting the potential to work 14 hours per day, 365 days per year.

$\gamma = 1$, is the elasticity between consumption and leisure. In a balanced equilibrium, it should be 1.

$\lambda = 1$ measures how households value government consumption. Having the value "1" means they value it as if it were their own consumption.

```
In [8]: # We were given the following parameters:
        H_bar = 5110
        gamma = 1
        Lambda = 1

        # Store base year variables in a separate data frame for future use:
        # Base years. 1956 for all countries but Australia, for which it is 1960
        # tax wedge defined as (1-"labour")/(1+"consumption")

        base_var = data.loc[data.index.get_level_values('YEAR').year == 1956,
                            ["taxwedge", "GDP", "CONSUMPTION ",
                             "GOVT. EXP", "L", "TOT HRS", "POPULATION"]]
        # Just for the special case, Australia:
```

```
base_var_aus = data.loc[data.index.get_level_values('YEAR').year == 1960,
                        ["taxwedge", "GDP", "CONSUMPTION ",
                         "GOVT. EXP", "L", "TOT HRS", "POPULATION"]]

base_var_aus = base_var_aus.loc[base_var_aus.index.get_level_values("country") == "AUS"
base_var = base_var.append(base_var_aus)

# Need to get rid of time as index, redundant for this data frame.
base_var = base_var.reset_index()
base_var = base_var.set_index("country")
```

1.2.2 Calculating variable values

We begin with calculating $\bar{c}$, which cccording to the authors:

"that is equal to 5 percent of total US consumption in 1956."

Unfortunately, they didn't tell us to adjust for population. We'll use the following formula for country i:

$$\bar{c}_i = 0.05(C_{US,0} + G_{US,0})\frac{N_{i,0}}{N_{US,0}}$$

```
In [9]: C_us0 = base_var.loc[base_var.index == "US", "CONSUMPTION "].iloc[0]
        G_us0 = base_var.loc[base_var.index == "US", "GOVT. EXP"].iloc[0]
        N_us0 = base_var.loc[base_var.index == "US", "POPULATION"].iloc[0]

        # Value for c_bar
        c_bar = 0.05*(C_us0 + G_us0)* base_var["POPULATION"] /N_us0
```

Next, we move on the base period output to consumption, which was defined as:

$$\frac{Y_0}{C_0 + \lambda G_0 - \bar{c}}$$

Similarly, we can calculate the ratio for each t:

$$\frac{Y_t}{C_t + \lambda G_t - \bar{c}}$$

These are both used to calculate hours worked to hours of leisure:

$$\frac{H_t^p}{(\bar{H} - H_t^p)^\gamma} = \frac{1-\tau_t}{1-\tau_0}\frac{\frac{Y_0}{C_0+\lambda G_0-\bar{c}}}{\frac{Y_t}{C_t+\lambda G_t-\bar{c}}}\frac{H_0}{(\bar{H}-H_0)^\gamma}$$

With H_t^p being model predicted hours of work. We do this for all countries, hence the implementation is a vectorised version of the above.

The tax wedge: $1 - \tau_t = \frac{1-\tau_{ht}}{1+\tau_{ct}}$ is the ratio of what's left after the government has taxed your income to the price of a good after a consumption tax has been added. Luckily, this was already given in the data set.

```
In [10]: # Define variables we'll be working with (for clarity in formulas):
         C0 = base_var["CONSUMPTION "]
         G0 = base_var["GOVT. EXP"]
         Y0 = base_var["GDP"]
         tw0 = base_var["taxwedge"]
         H0 = base_var["L"]
         YCG0 = (Y0)/(C0 + Lambda*G0 - c_bar) # Base period output to consumption ratio

         C = data["CONSUMPTION "]
         G = data["GOVT. EXP"]
         Y = data["GDP"]
         tw = data["taxwedge"]
         YCGt = (Y)/(C + G * Lambda - c_bar) # Output to consumption ratio at t for all t
```

Calculate the initial leisure to work variable:

$$pct\_work = \frac{H_0}{(\bar{H} - H_0)^{\gamma}}$$

```
In [11]: pct_work = H0/(H_bar - H0)**gamma # fraction worked to leisure
```

We now have everything on the RHS of the following and can compute values predicted by the model:

$$\frac{H_t^p}{\bar{H} - H_t^p} = \frac{1 - \tau_t}{1 - \tau_0} \frac{\frac{Y_0}{C_0 + \lambda G_0 - \bar{c}}}{\frac{Y_t}{C_t + \lambda G_t - \bar{c}}} \frac{H_0}{(\bar{H} - H_0)^{\gamma}}$$

```
In [12]: data["work_leisure_t"] = (tw * YCGt)/(tw0 * YCG0) * pct_work
         data["work_leisure_t"] = pd.to_numeric(data["work_leisure_t"])
```

Next, we solve for model predicted hours, exploiting the fact that we now know $\frac{H_t^p}{\bar{H} - H_t^p}$ (from the relationship above) and also know $\bar{H}$:

$$H_t^p = \frac{\frac{H_t^p}{\bar{H} - H_t^p} \bar{H}}{1 + \frac{H_t^p}{\bar{H} - H_t^p}}$$

We then calculate predicted change for every time period (and country), group by country and decade and take the arithmetic average (because that's what the authors did). We then add a column for the entire period.

```
In [13]: # Solve for predicted hours:
         data["pred_hours"] = data["work_leisure_t"]*H_bar/(1 + data["work_leisure_t"])

         # Predict change in worked hours:
         data["pred_change"] = (data["pred_hours"] - data["pred_hours"].shift(1))/(data["pred_h

         table3 = data["pred_change"].groupby([data.index.get_level_values('country'), data["de
         table3 = (table3.unstack()*100).round(2)
```

```
# Need to add a column of average change between 1956 and 2003:
data_post_1956 = data[data.index.get_level_values('YEAR').year > 1956]
table3["1956-2003"] = data_post_1956["pred_change"].groupby(data_post_1956.index.get_l
table3["1956-2003"] =  (table3["1956-2003"]*100).round(2)

# Remove redundant columns:
table3 = table3[[1960, 1970, 1980, 1990, '1956-2003']]
table3 = table3.sort_index()
```

In [14]:
```
table3
```

Out[14]:

decade	1960	1970	1980	1990	1956-2003
country					
AUS	-0.16	-0.86	-0.33	0.14	-0.30
AUT	-0.70	-0.73	-0.11	-0.68	-0.49
BEL	-0.76	-1.60	-0.09	-0.25	-0.65
CAN	-0.74	0.08	-0.58	0.21	-0.24
FIN	-0.71	-0.70	-1.00	-0.51	-0.57
FRA	-0.29	-0.65	-0.62	-0.28	-0.38
GER	-0.65	-1.43	-0.13	-0.78	-0.68
ITA	-0.70	-0.70	-1.70	-1.37	-0.92
JAP	0.57	-0.64	-0.38	-0.65	-0.28
NETH	-1.30	-1.59	0.80	0.42	-0.45
SPA	-0.17	-1.00	-1.25	-0.21	-0.70
SWE	-2.08	-1.39	-1.00	0.50	-0.76
SWI	-0.19	-0.78	0.25	-0.72	-0.38
UK	-0.68	-0.07	-0.35	-0.06	-0.33
US	-0.49	0.06	-0.32	0.10	-0.20

Apparently, everything looks fine but the values for Finland which are way off (Australia is slightly off as well). Eye balling the data we had at hands revealed that the L value for Finland had accidentally been coded as the population in the data. Imputing a more reasonable value (next period's value) using:

In [15]:
```
%%capture
data.set_value(data.L.idxmax(), "L", 1473.98)
```

And then re running the whole code and recreate the table gives us something closer to what the authors found. We supress the code since it is precisely what we had above run again after we've changed one value in the data for Finland. We also add info on what group the countries are in according to the authors definition, as well as give the tables appropriate names.

In [16]:
```
%%latex
\newpage
```

In [17]:
```
#### Actual change ####
# Calculate change
data["L_change"] = (data["L"] - data["L"].shift(1))/(data["L"].shift(1))

# Calculate average change, grouping by decade and country
table2 = data["L_change"].groupby([data.index.get_level_values('country'),
            data["decade"]]).mean()
table2 = (table2.unstack()*100).round(2)


data_post_1956 = data[data.index.get_level_values("YEAR").year > 1956]
table2["1956-2003"] = data_post_1956["L_change"].groupby(data_post_1956.index.get_leve
table2["1956-2003"] =  (table2["1956-2003"]*100).round(2)

# Remove redundant columns
table2 = table2[[1960, 1970, 1980, 1990, '1956-2003']]
table2 = table2.sort_index()

H_bar = 5110
gamma = 1
Lambda = 1

# Store base year variables in a separate data frame for future use:
# Base years. 1956 for all countries but Australia, for which it is 1960
# tax wedge defined as (1-"labour")/(1+"consumption")

base_var = data.loc[data.index.get_level_values('YEAR').year == 1956,
                    ["taxwedge", "GDP", "CONSUMPTION ",
                     "GOVT. EXP", "L", "TOT HRS", "POPULATION"]]
# Just for the special case, Australia:
base_var_aus = data.loc[data.index.get_level_values('YEAR').year == 1960,
                    ["taxwedge", "GDP", "CONSUMPTION ",
                     "GOVT. EXP", "L", "TOT HRS", "POPULATION"]]

base_var_aus = base_var_aus.loc[base_var_aus.index.get_level_values("country") == "AUS
base_var = base_var.append(base_var_aus)

# Need to get rid of time as index, redundant for this data frame.
base_var = base_var.reset_index()
base_var = base_var.set_index("country")

# Calculate cbar: "that is equal to 5 percent of total US consumption in 1956."
# Adjust for population for all countries
C_us0 = base_var.loc[base_var.index == "US", "CONSUMPTION "].iloc[0]
G_us0 = base_var.loc[base_var.index == "US", "GOVT. EXP"].iloc[0]
N_us0 = base_var.loc[base_var.index == "US", "POPULATION"].iloc[0]

# Value for c_bar
```

```
c_bar = 0.05*(C_us0 + G_us0)* base_var["POPULATION"] /N_us0

# Define variables we'll be working with (for clarity in formulas):
C0 = base_var["CONSUMPTION "]
G0 = base_var["GOVT. EXP"]
Y0 = base_var["GDP"]
tw0 = base_var["taxwedge"]
H0 = base_var["L"]
YCG0 = (Y0)/(C0 + Lambda*G0 - c_bar) # Base period output to consumption ratio

C = data["CONSUMPTION "]
G = data["GOVT. EXP"]
Y = data["GDP"]
tw = data["taxwedge"]
YCGt = (Y)/(C + G * Lambda - c_bar) # Output to consumption ratio at t for all t

pct_work = H0/(H_bar - H0)**gamma # fraction worked to leisure

data["work_leisure_t"] = (tw * YCGt)/(tw0 * YCG0) * pct_work # Intuition for this?
data["work_leisure_t"] = pd.to_numeric(data["work_leisure_t"])

data["pred_hours"] = data["work_leisure_t"]*H_bar/(1 + data["work_leisure_t"]) # Intu

data["pred_change"] = (data["pred_hours"] - data["pred_hours"].shift(1))/(data["pred_l

table3 = data["pred_change"].groupby([data.index.get_level_values('country'), data["d
table3 = (table3.unstack()*100).round(2)

# Need to add a column of average change between 1956 and 2003:
data_post_1956 = data[data.index.get_level_values('YEAR').year > 1956]
table3["1956-2003"] = data_post_1956["pred_change"].groupby(data_post_1956.index.get_l
table3["1956-2003"] =  (table3["1956-2003"]*100).round(2)

# Remove redundant columns:
table3 = table3[[1960, 1970, 1980, 1990, "1956-2003"]]
table3 = table3.sort_index()
Model_hours_worked = table3
Actual_hours_worked = table2

# Add group information for sorting purposes
table2["Group"] = 1
table2["Group"] = table2["Group"] + (table2["1956-2003"] > -0.5)
table2["Group"] = table2["Group"] + (table2["1956-2003"] > 0)

Actual_hours_worked["Group"] = table2["Group"]
Model_hours_worked["Group"] = table2["Group"]
Actual_hours_worked = Actual_hours_worked.sort_values("Group")
Model_hours_worked = Model_hours_worked.sort_values("Group")
```

In [18]:
```
%%latex
\newpage
```

1.3 Final tables

```
In [19]: Model_hours_worked
```

Out[19]:

country	1960	1970	1980	1990	1956-2003	Group
AUT	-0.70	-0.73	-0.11	-0.68	-0.49	1
BEL	-0.76	-1.60	-0.09	-0.25	-0.65	1
FIN	-1.04	-0.98	-1.36	-0.67	-0.79	1
FRA	-0.29	-0.65	-0.62	-0.28	-0.38	1
GER	-0.65	-1.43	-0.13	-0.78	-0.68	1
ITA	-0.70	-0.70	-1.70	-1.37	-0.92	1
JAP	0.57	-0.64	-0.38	-0.65	-0.28	2
NETH	-1.30	-1.59	0.80	0.42	-0.45	2
SPA	-0.17	-1.00	-1.25	-0.21	-0.70	2
SWE	-2.08	-1.39	-1.00	0.50	-0.76	2
SWI	-0.19	-0.78	0.25	-0.72	-0.38	2
UK	-0.68	-0.07	-0.35	-0.06	-0.33	2
AUS	-0.16	-0.86	-0.33	0.14	-0.30	3
CAN	-0.74	0.08	-0.58	0.21	-0.24	3
US	-0.49	0.06	-0.32	0.10	-0.20	3

```
In [20]: Actual_hours_worked
```

Out[20]:

country	1960	1970	1980	1990	1956-2003	Group
AUT	-1.03	-1.14	-0.22	-0.53	-0.64	1
BEL	-1.03	-1.64	-0.74	0.06	-0.72	1
FIN	-1.15	-0.45	0.32	-1.31	-0.58	1
FRA	-0.63	-1.47	-1.53	0.11	-0.87	1
GER	-1.09	-1.55	-1.20	-0.74	-1.08	1
ITA	-1.60	-1.19	-0.43	-0.16	-0.61	1
JAP	-0.39	-0.47	0.14	-0.74	-0.31	2
NETH	-0.65	-1.81	-0.95	1.40	-0.44	2
SPA	0.25	-1.67	-1.56	0.85	-0.28	2
SWE	-1.00	-0.36	0.54	-0.71	-0.48	2
SWI	-0.23	-1.26	0.31	0.27	-0.38	2
UK	-0.42	-1.17	-0.22	-0.34	-0.48	2
AUS	0.11	-0.53	0.49	-0.01	0.01	3
CAN	0.18	0.34	0.70	-0.16	0.17	3
US	0.29	-0.25	0.73	0.46	0.03	3

We don't end up with perfect values for Finland and Australia – this is likely because of data issues since we end up with correct results for most countries.