Applied Economic Analysis 1 Assignment

Vector Autoregressive Model Estimation and Simulation for the Netherlands

Weiyuan Liu

Janurary 9 2023

1. Introduction

Vector autoregressive (VAR) models are a simple and powerful approach for macroeconomic simulations and predictions. They are widely used by monetary institutions such as Fed, ECB and DNB. An VAR model chosen from an paper of ECB working paper series, *How to Estimate a VAR after March 2020?*[1] (Lenza & Primiceri, 2020), is modified as presented as follows,

$$y_t = C + B_1 y_{t-1} + \epsilon_t \tag{1}$$

$$\epsilon \sim N(0, \Sigma),$$

where y_t is an $n \times 1$ vector of variables, modeled as a function of a constant term, their own lagged values, and an $n \times 1$ vector of forecast errors ϵ_t.

A VAR model is simple because I can estimate it using OLS. It is poweful in two ways. First, once I obtain estimates, I can use them for simulation and forecasting. The mechanism is also simple. In order to forecast values for one period ahead, I use the last values of my dataset, put them into the RHS of equation (1), adding the constant term and a random draw from the distribution of forecast errors. However, in the forcasting process, it is required that the forecast error vector must be multiplied by the Cholesky decomposition of covariance matrix. The result of this process is the projected values for the next period. Then I repeat the process using the values of the next period to obtain projected values for the subsequent period. In this way, I can generate a sequence of values for the time interval I am interested in. The following equation, is an illustration of how it works.

$$\underbrace{y_{t+1}}_{\substack{\text{The values of}\\ \text{next period}}} = C + B_1 \times \underbrace{y_t}_{\substack{\text{The values of}\\ \text{current period}}} + \epsilon_t \times \underbrace{Chol(\Sigma)}_{\substack{\text{Cholesky decomposition of}\\ \text{Covariance matrix}}}$$

Because each random draw could be different, repeating the whole process for numerous times allows me to record a large range of possible future values and estimate

[1] Lenza, M., & Primiceri, G. E. (2020). *How to Estimate a VAR after March 2020* (No. w27771). National Bureau of Economic Research.

other statistical properties such as confidence interval. This is one way to apply VAR for simulation.

Another powerful function of VAR models is impulse responses. It characterizes if I artificially impose a shock to a variable at time $t = 0$, how other variables would response at time $t = 1$, ..., n. The mechanism is simple, too. After I obtain estimates, I can artificially put a number in the error term of RHS of equation (1) at time $t = 0$, repeat the same process mentioned previously, and record a sequence of values. The following equations, taken from *Introductory Econometrics for Finance, 4th edition* (Brooks, page 323), are an illustration of how it works. Consider a bivariate case, suppose I impose a unit shock to y_{1t} at time $t = 0$,

$$
\begin{aligned}
y_0 &= \begin{bmatrix} y_{10} \\ y_{20} \end{bmatrix} = \begin{bmatrix} \epsilon_{10} \\ \epsilon_{20} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\
y_1 &= B_1 y_0 = \begin{bmatrix} 0.5 & 0.3 \\ 0.0 & 0.2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} \\
y_2 &= B_1 y_1 = \begin{bmatrix} 0.5 & 0.3 \\ 0.0 & 0.2 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.25 \\ 0 \end{bmatrix}
\end{aligned}
$$

After I derive all values of y_t, it would thus be possible to plot the results.

2. Motivation

The theoretical paper we study all more or less involve in dynamic optimization problems. I don't know how to use python to solve these models, especially when they are associated with eigenvalues, Jacobian thing, state variables, etc. Therefore I choose an empirical model. This also paves the way for my master thesis, because I am interested in writing one related to macroeconomic modelling. This model will require skills such as data importing and processing, estimation, simulation, and graphing, will be sufficient to show off my programming skills and satisfy the assignment requirements.

3. Variables and Data

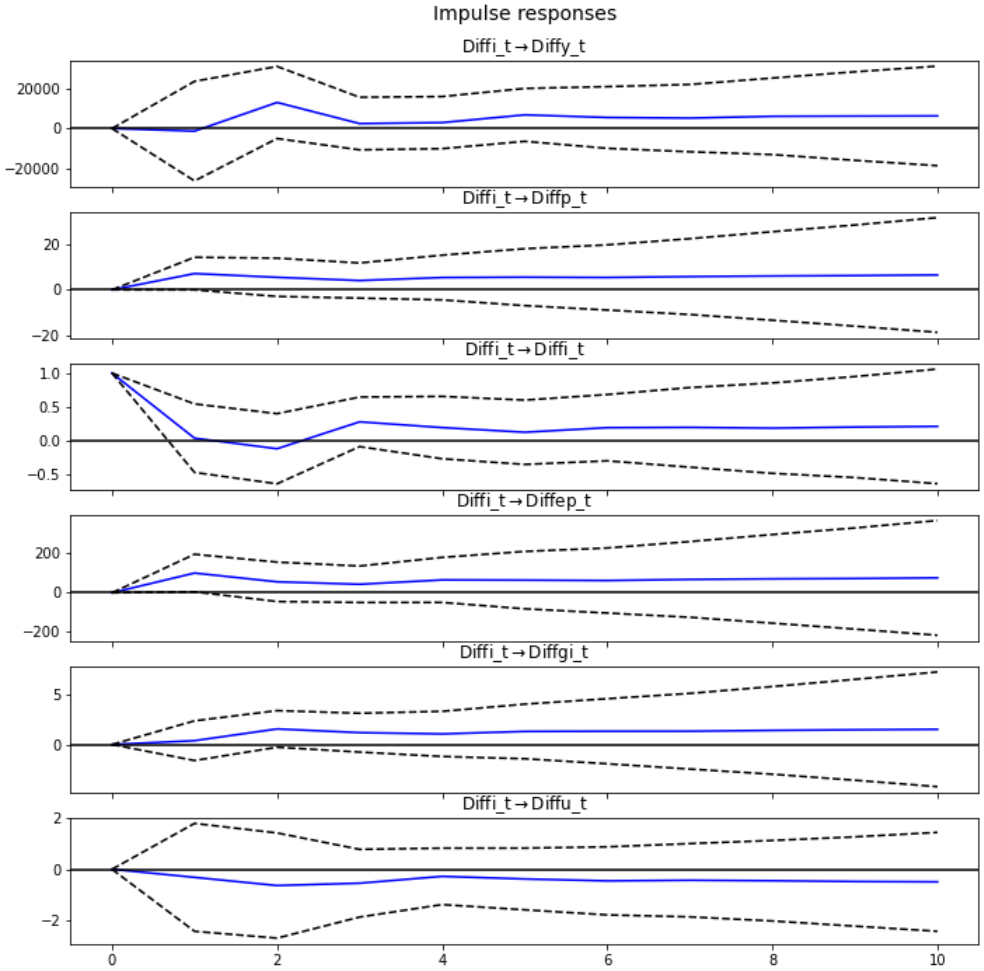
I choose quarterly data for ten years, 2012.07.01 - 2022.07.01 from FRED Economic Data, St. Louis FED[2]. The corresponding names of each variable in the database are provided. The variables are Output: y_t, Real Gross Domestic Product for Netherlands; Inflation: p_t, Consumer Price Index: Harmonized Prices: Total All Items for the Netherlands; Energy Inflation: ep_t, Harmonized Index of Consumer Prices: Electricity, Gas, Solid Fuels and Heat Energy for Netherlands; Unemployment rate: u_t, Harmonized Unemployment Rate: Total: All Persons for the Netherland; Short-term interest rate: i_t, Immediate Rates: Less than 24 Hours: Call Money/Interbank Rate for the Netherlands; Long-term government bond rate: gi_t, Long-Term Government Bond Yields: 10-year: Main (Including Benchmark) for the Netherlands. Therefore, the model I want to esti-

mate is

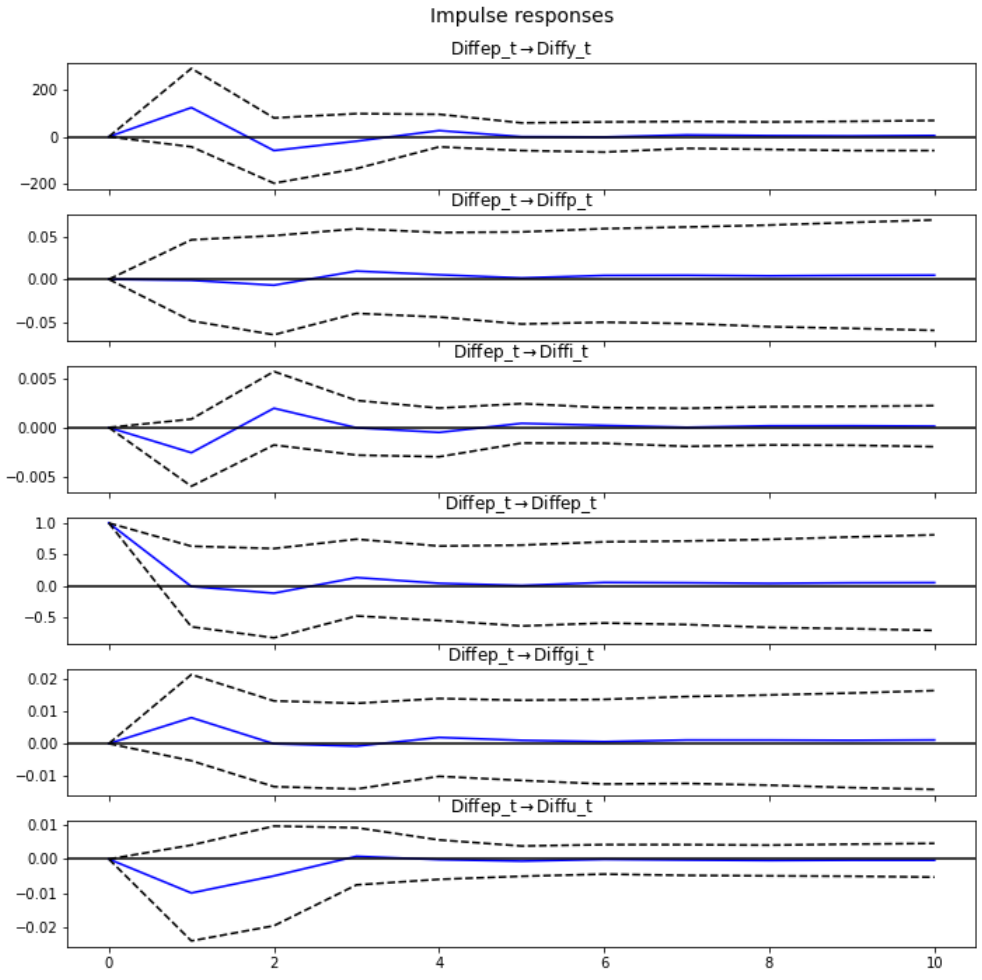
$$\begin{bmatrix} y_t \\ p_t \\ ep_t \\ u_t \\ i_t \\ gi_t \end{bmatrix} = C + B_1 \begin{bmatrix} y_{t-1} \\ p_{t-1} \\ ep_{t-1} \\ u_{t-1} \\ i_{t-1} \\ gi_{t-1} \end{bmatrix} + \epsilon_t$$

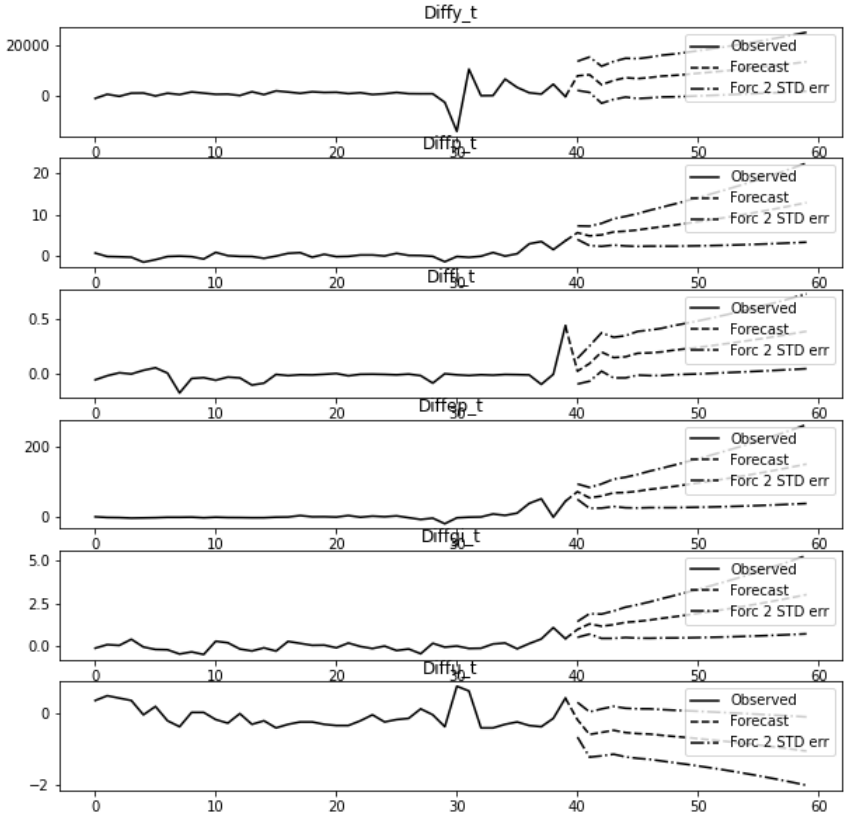
4. Results

I am interested in how macroeconomic variables response to a shock in short-term interest rate and energy inflation. Impulse response graphs show that the short term effect is strong, while as time extends, the effect would gradually disappear. Output, inflation and energy inflation might continue rise. Short-term interest rate and long-term government bond rate might rise slightly, and unemployment rate might decrease in the future.



[2]https://fred.stlouisfed.org/


Impulse responses
Diffep_t → Diffy_t
Diffep_t → Diffp_t
Diffep_t → Diffi_t
Diffep_t → Diffep_t
Diffep_t → Diffgi_t
Diffep_t → Diffu_t


Diffy_t
Diffp_t
Diffi_t
Diffep_t
Diffgi_t
Diffu_t
Observed
Forecast
Forc 2 STD err

5. Python Codes

In [ ]:
```
# Importing packages, I also use glob and os (learned from some websites) for sake of data processing

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.tsa.api as smt
import statsmodels.api as sm
import glob
import os
```

In [ ]:
```
# Data directly downloaded from the database. I put them in one fold. These several lines helps me to
# read all data, extract columns I need, and merge all datasets. They look simple and compact, but
# I put quite  some time to learn this from some websites and get this work.

path = 'data'
all_files = glob.glob(os.path.join(path , "*.csv"))
df_from_each_file = (pd.read_csv(f, usecols = [1]) for f in all_files)
df   = pd.concat(df_from_each_file, axis=1)
df.columns = ['y_t', 'p_t', 'i_t', 'ep_t', 'gi_t', 'u_t']
print(df)
```

In [ ]:
```
df.plot(y = 'y_t')
df.plot(y = 'p_t')
df.plot(y = 'i_t')
df.plot(y = 'ep_t')
df.plot(y = 'gi_t')
df.plot(y = 'u_t')
plt.show()
```

In [ ]:
```
# First-differencing time series data as required in time series analysis. I create a function to do so
# (also learned from a website)

def FirstDiff(x):
    x_diff = x - x.shift(1)
    x_diff = x_diff.dropna()
    return x_diff

df = pd.DataFrame({'Diffy_t':FirstDiff(df.y_t),
                   'Diffp_t':FirstDiff(df.p_t),
                   'Diffi_t':FirstDiff(df.i_t),
                   'Diffep_t':FirstDiff(df.ep_t),
                   'Diffgi_t':FirstDiff(df.gi_t),
                   'Diffu_t':FirstDiff(df.u_t)})

print(df)
```

In [ ]:
```
# Use a command from statsmodels.tsa.api, designed for VAR estimation, to estimate a VAR model

model = smt.VAR(df)
res = model.fit(maxlags=1)
print(res.summary())

res.plot();
```

In [ ]:
```
# Use functions from the same package to estimate impulse responses, and plot. I shock the short-term
# interest rate, and see how other variables response

irf = res.irf(10)
irf.plot(impulse = 'Diffi_t', orth=False);
```

In [ ]:
```
# I can also shock the energy inflation, see how other variables response

irf = res.irf(10)
irf.plot(impulse = 'Diffep_t',  orth=False);
```

In [ ]:

```
# However, there is a problem with VAR forecasting by the package because as I first-difference the data
# the forecast values should be cumulated, but I couldn't find a function in the package to do so.
# Therefore, I try to obtain the estimates and covariance matrix by hand

# Create X and Y to calculate estimates by hand
dflag_1= df.shift(1)
dflag_1 = dflag_1.dropna()
df1 = df.iloc[1:]

dflag_1.insert(0, "Intercept", 1)
X = dflag_1.to_numpy()
Y = df1.to_numpy()

# This function will obtain OLS estimates
def OLSoperation(X, y):

    XtX = np.matmul(X.T, X)
    XtY = np.matmul(X.T, y)
    XtX_Inv = np.linalg.inv(XtX)

    b = np.matmul(XtX_Inv, XtY)

    return b

B = OLSoperation(X, Y)
np.set_printoptions(suppress=True,precision=5)
print(B)

# Predicted values
y_estimates = []

for i in df.index:
    entry = df.loc[i].to_numpy()

    y_hat = np.dot(B, entry)
    y_estimates.append(y_hat)

# Forecast values

# However, I can't continue because I don't know how to calculate covariance matrix by hand, so
# I can't obtain cholesky matrix

Steps = 20
forecast_values = []
data = df.iloc[-1].to_list()

# for i in range(Steps):

# PredictY = np.dot(data, B)
#    data.insert(1, PredictY)
#    forecast_values.append(PredictY);
```