1 Structural VAR Example

We will now turn to our example on the structural VAR. We will be using a roughly similar methodology to the VAR but adding some restrictions on the values and some more conditions. Fortunately, there is a library of commands in R for this very purpose. For our example, we will be dealing with three variables, namely, the output gap y_t, the inflation rate π_t, and interest rate (RRP) which is r_t. If you recall, we used this example previously in the section on SVAR but never really ran it there. We will impose the same restrictions as the conditions which we have outlined there.

1.1 Preliminaries

As always, we start by loading and installing the packages, transforming the variables to time series objects, seeing these variables graphically, and checking some general conditions.

1.1.1 Installing and Loading the Required Packages

We start by loading and installing the required packages. The packages we need to use are roughly similar to those in VAR, except we will be installing one specialized packaged called "svars". Like always, we will load the packages using the library() command and install using the install.packages() command.

```
install.packages("svars")
library(urca)
library(vars)
library(mFilter)
library(tseries)
library(TSstudio)
library(forecast)
library(tidyverse)
library(svars)
```

1.1.2 Time Series Loading

We load our dataset which is the SVAR_Philippines.csv file. This contains the data points in our study which run from Q1 2000 to Q1 2020. The frequency of this data is quarterly. All datapoints are available at the BSP website except for the Output Gap which was estimated by the author using a *Kalman Filtering* technique.

```
macro < − read_csv(file.choose())
head(macro)
```

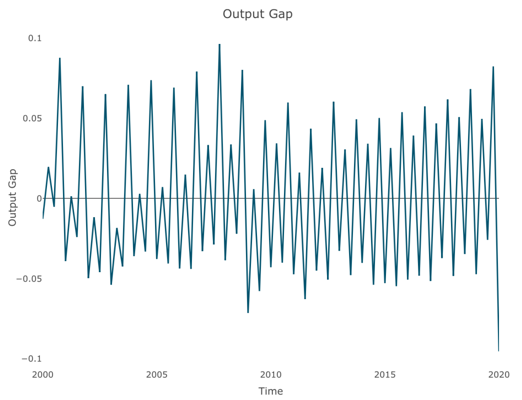
After loading the dataset, we need to turn our variables into time series variables useable for our SVAR. Like always, we use thee ts() command to do this.

```
y < − ts(macro$‘Output Gap‘, start = c(2000,1,1), frequency = 4)
pi < − ts(macro$CPI, start = c(2000,1,1), frequency = 4)
r < − ts(macro$RRP, start = c(2000,1,1), frequency = 4)
```

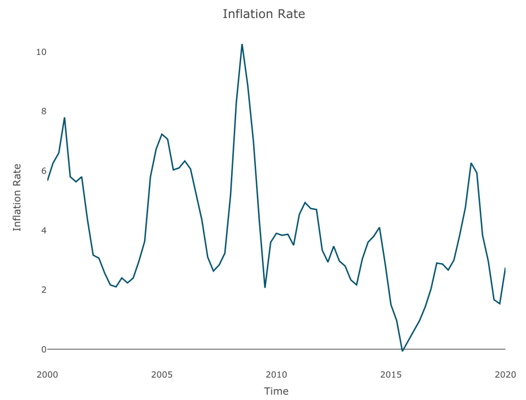
1.1.3 Plotting the Series

As always, we can plot each time series using the ts_plot() command. It is important to visualize the series to verify the loading of the series and to get a feel for certain characteristics. You may opt to use the ts_decompose command to see a better decomposition.

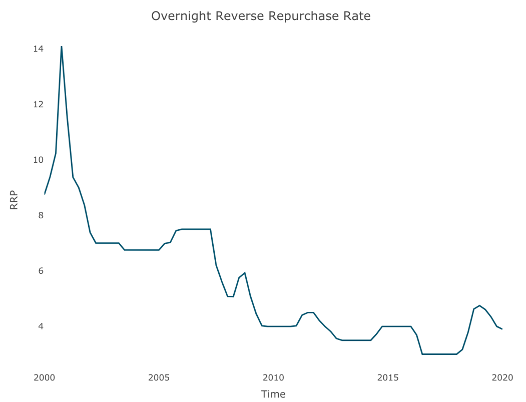
```
ts_plot(y, title = "Output Gap", Xtitle = "Time", Ytitle = "Output Gap")
ts_plot(pi, title = "Inflation Rate", Xtitle = "Time", Ytitle = "Inflation Rate")
ts_plot(r, title = "Overnight Reverse Repurchase Rate", Xtitle = "Time", Ytitle = "RRP")
```



(a) Output Gap



(b) Inflation Rate



(c) Overnight Reverse Repurchase Rate

Figure 1: Time Series Plots in SVAR

1.2 Setting Restrictions and Building the SVAR

We will now differentiate from the VAR by adding restrictions. We will need to build the matrix $\mathcal{A}$ and subsequently estimate the structural coefficients by using the recursive ordering method discussed in the last section.

1.2.1 Setting Restrictions

We now come to a crucial part of our analysis in SVAR which is setting the value of matrix $\mathcal{A}$. As we have said, this matrix is the matrix of contemporaneous shocks affecting the variables in the systems. In identifying the coefficients, we need to set restrictions and these restrictions are defined by an economic principle. In the matrix we will build, we will impose the behavior that if r is the policy rate, we could say that a reaction or a movement in r would be attributable to shocks in π and y in the same period. We will structure the matrix such that the policy rate r only affects pi and y with a lag. Hence, we formulate the matrix as the form below.

$$\mathcal{A} = \begin{pmatrix} 1 & 0 & 0 \\ a_{21} & 1 & 0 \\ a_{31} & a_{32} & 1 \end{pmatrix}$$

This restriction suggests that a contemporaneous shock in y affects both π and r in the same period and the lagged values in the system. Further, π shocks only affect r contemporaneously but not y. Likewise, it also affects the lagged values in the system. Lastly r doesn't affect y and π contemporaneously but affects the lagged values in the system. We first define a matrix as an object "amat" and set the conditions in the code.

```
amat < − diag(3)
amat[2,1] < − NA
amat[3,1] < − NA
amat[3,2] < − NA
amat
```

The diag() command creates a 3×3 identity matrix. The next three commands modify the lower triangular to have values of NA. Essentially, we are identifying the parameters which can freely take any value. Hence, after we estimate, the NA will get filled up. Meanwhile, we retain the upper triangular as zero which represents the restrictions we imposed akin to the economic intuition we pre-specified in the matrix.

1.2.2 Binding Variables and Selecting the Choleskey Ordering

After this, we will now specify the ordering we want. To reflect the economic intuition of our restriction, we have to follow the order of output gap first, followed by inflation, and lastly by the policy rate. Again, think of this Choleskey ordering as a timeline of events or a sequencing of events.

```
sv <- cbind(y, pi, r)
colnames(sv) <- cbind("OutputGap", "Inflation", "RRP")
```

1.2.3 Lag Order Selection

We now move on to determining the lag ordering using the VARselect() command. Since this is quarterly data, we expect a lag order of around four to six.

```
lagselect <- VARselect(sv, lag.max = 8, type = "both")
lagselect$selection
lagselect$criteria
```

Based on the AIC, SBIC, HQIC, and FPE, it appears that the oprimal lag order is five lags. Hence, let us build our model to have 5 lags.

1.2.4 Building the SVAR

To estimate the SVAR, we first need to estimate a reduced form VAR which is why we have a VAR() estimation here. Again, we set the number of lags to 5. Afterwhich, we use the SVAR() command and set the Amat option to our amat object. Doing this should yield us our estimated SVAR $\mathcal{A}$ matrix.

```
Model1 <- VAR(sv, p = 5, season = NULL, exog = NULL, type = "const")
SVARMod1 <- SVAR(Model1, Amat = amat, Bmat = NULL, hessian = TRUE, estmethod =
c("scoring", "direct"))
SVARMod1
```

If you did this correctly, you should see that we estimated our structural parameters to be 0.21, 0.09, and -0.11 which can be found as the estimated $\mathcal{A}$ matrix.

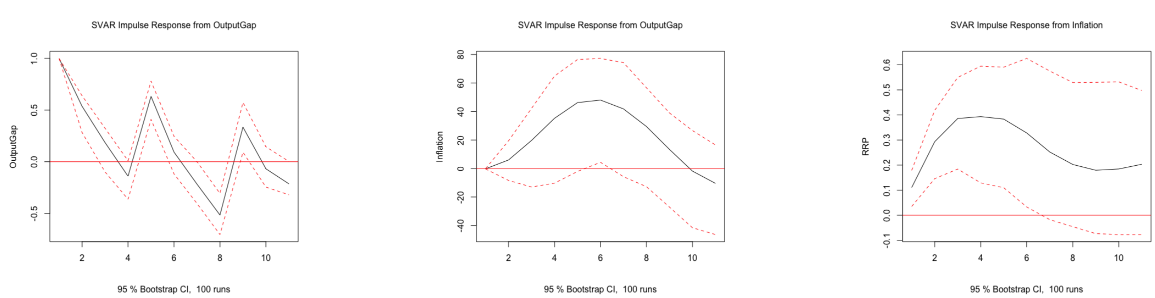
1.3 SVAR Applications

We will now move on to the applications of SVAR. It is expected that before this, you first diagnose the VAR akin to the steps we used in the last section. In this part, we will focus mainly on two main applications, which are the Impulse Response Functions and the Forecast Error Variance Decompositions.

1.3.1 Impulse Response Functions in SVAR

We will zero in on how the intuition played a role in building the IRFs and seeing how the policy rate r would respond to shocks in the system. To run the IRF, we simply use the irf() command and set the impulse and responses as we wish. As there are three variables in the system, there are a total of nine (3^2) possible IRFs to come by. We will zero in on three and I hope you'll understand the reaction clearly based on the restrictions we imposed.

```
SVARog <- irf(SVARMod1, impulse = "OutputGap", response = "OutputGap")
plot(SVARog)
SVARinf <- irf(SVARMod1, impulse = "OutputGap", response = "Inflation")
plot(SVARinf)
SVARrrp <- irf(SVARMod1, impulse = "Inflation", response = "RRP")
plot(SVARrrp)
```



(a) y_t to y_t Shock (b) y_t to π_t Shock (c) π_t to r_t Shock

Figure 2: Impulse Responses in SVAR

The impluse responses are very intuitive. If a shock in the output gap starts the sequence, it is expected to increase the output gap. This increase in the output gap increases the inflationary gap which means that the economy is producing more than it was potentially expected to which may cause the economy to overheat. This higher than expected output gap will push inflation upward as productivity continues to increase. Since the central bank sees that output gap increases and that inflation increases, it can accommodate these by increasing the RRP which is exactly what is seen in window (c). This is typically the monetary response of a central bank and was validated by our SVAR.

1.3.2 Forecast Error Variance Decomposition in SVAR

Lastly for this module on VAR and SVAR, we will turn to the FEVD decomposition in the SVAR. To do this, we use the fevd() command and set a horizon ahead. In this case, we set it to ten periods ahead. We then use the plot() command to plot the values which are seen in figure to follow.

```
SVARfevd <- fevd(SVARMod1, n.ahead = 10)
SVARfevd
plot(SVARfevd)
```

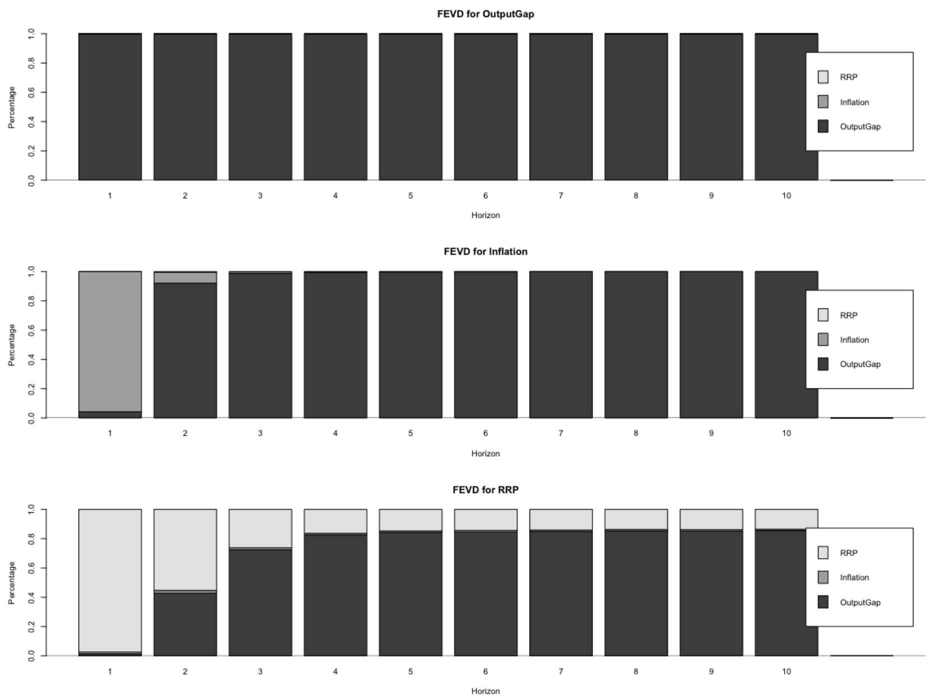


Figure 3: Forecast Error Variance Decomposition in VAR

If you will notice, the FEVD of Output Gap is explained purely by Output Gap. This is just because we ordered this one first. If you will see the shocks in the other variables, they reflect that majority of the forecast error variance decomposition was because we pushed that variable as the first in the order.

Overall, I hope you can see how adding a structure to the VAR which are the different restrictions we imposed was able to yield a key policy result. Therefore, I hope you are able to realize that SVARs are incredibly useful policy tools for government in conducting policy and controlling for unobservables.

1.4 Notes

- This section is taken from my lecture notes in time series econometrics. For a better introduction to time series, consider reading Hamilton or Brooks.