# Lab No. 3 — Factor models for SGPE

## Advanced Time Series Econometrics Labs


### Ping Wu (ping.wu@strath.ac.uk (mailto:ping.wu@strath.ac.uk))


---

## Outline of today's lecturer-led lab

- Principal component analysis (PCA)
- Factor augmented VARs

## Packages we will use in this lab

- zoo (https://cran.r-project.org/web/packages/zoo/zoo.pdf): For working with time series data. The package includes important functions such as computing rolling correlations, rolling standard deviations and aggregation (e.g., converting data from daily to monthly frequency).
- vars (https://cran.r-project.org/web/packages/vars/vars.pdf): Includes methods and tools specifically for VAR analysis.
- factoextra (https://cran.r-project.org/web/packages/factoextra/factoextra.pdf): Easy-to-use functions to extract and visualise the output of PCA
- corrplot (https://cran.r-project.org/web/packages/corrplot/corrplot.pdf): Provides a visual exploratory tool on correlation matrix.

```
# Important packages described above
library(zoo)
library(vars)
library(factoextra)
library(corrplot)

options(scipen = 9) # Avoid scientific notation
```

## Part 0: Plot time series over time

In this lab, we will use the monthly data set for the US from January 1980 to December 2019. The data set includes year-on-year (y-o-y) consumer price inflation ( `CPIAUCSL` ), the federal funds rate ( `FEDFUNDS` ), and y-o-y growth rates for seven different industrial production measures: `INDPRO` , `IPMANSICS` , `IPCONGD` , `IPMINE` , `IPDCONGD` , `IPBUSEQ` , `IPMAT` . The time series were directly transformed and downloaded as a `.csv` file from FRED (https://fred.stlouisfed.org).

```r
# Allocate the macroeconomic variables to the object "usmacro"
if("usmonthly.csv" %in% list.files()){
  usmacro <- read.table("usmonthly.csv", sep=",",header=T)
}else{ # file.choose() allows to choose the file interactively
  usmacro <- read.table(file.choose(), sep=",",header=T)
}
# Select the following macroeconomic variables
var.slct <- c("INDPRO", "IPMANSICS", "IPCONGD", "IPMINE", "IPDCONGD",  "IPBUSEQ", "
IPMAT",
              "CPIAUCSL", "FEDFUNDS")
# Specify "usmacro" as time series object
usmacro <- ts(usmacro[,var.slct], end = c(2019, 12), frequency = 12)
colnames(usmacro) <- c("IP1", "IP2", "IP3", "IP4", "IP5",  "IP6", "IP7",
                       "CPI", "FFR") # Use shorter labels for variables
```

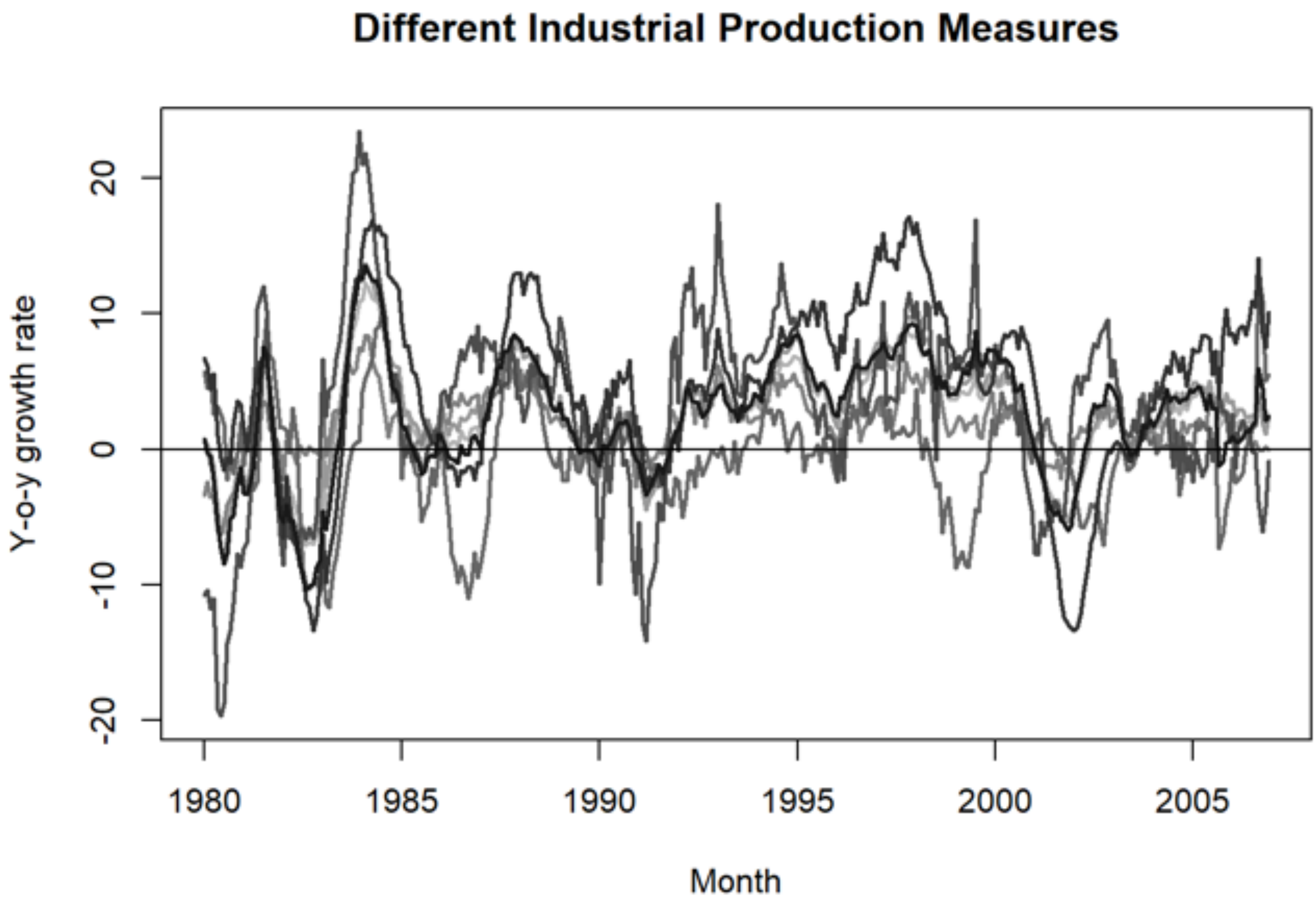
Again, we may wish to subset our time series.

```r
# Subset time series and define a new end date
usmacro <- window(usmacro,           # Select the ts object
                  end = c(2006,12)   # Define new end date of time series
)
```

# Part I: Some descriptives

In the first two parts we focus solely on the different industrial production measures. Let’s plot the seven different series with `ts.plot()`.

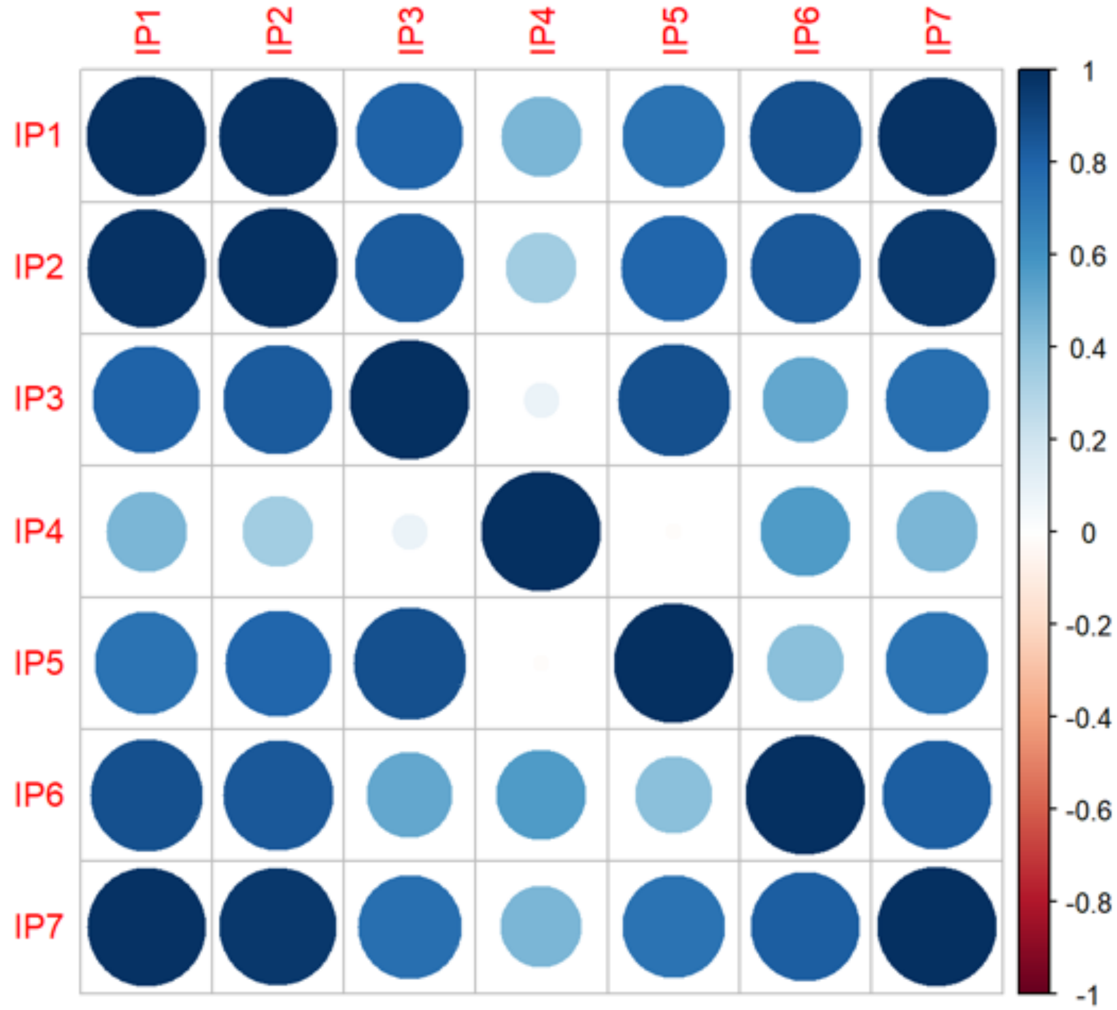
```r
# Time series plot of industrial production measures
IP.series <- usmacro[,1:7]

ts.plot(IP.series,
        main = "Different Industrial Production Measures",  # Title of plot
        ylab = "Y-o-y growth rate",                         # Label of y-axis
        xlab = "Month",                                     # Label of x-axis
        col  = c(gray(seq(0.7,0.1,length.out = 7))),        # Define colours
        lty  = "solid",                                     # Define line types
        lwd  = 2                                            # Define line width
)
abline(h = 0) # Horizontal line at zero
```




Explain comovment between series […] We see the seven measures strongly comove together […]

Let’s take a closer look a the correlations […] Nice correlation plot for the industrial production measures with `corrplot()` function…

```
# Obtain raw correlation coefficients
cor <- cor(IP.series)

# Nice correlation plot for the industrial production measures
corrplot::corrplot(cor)
```



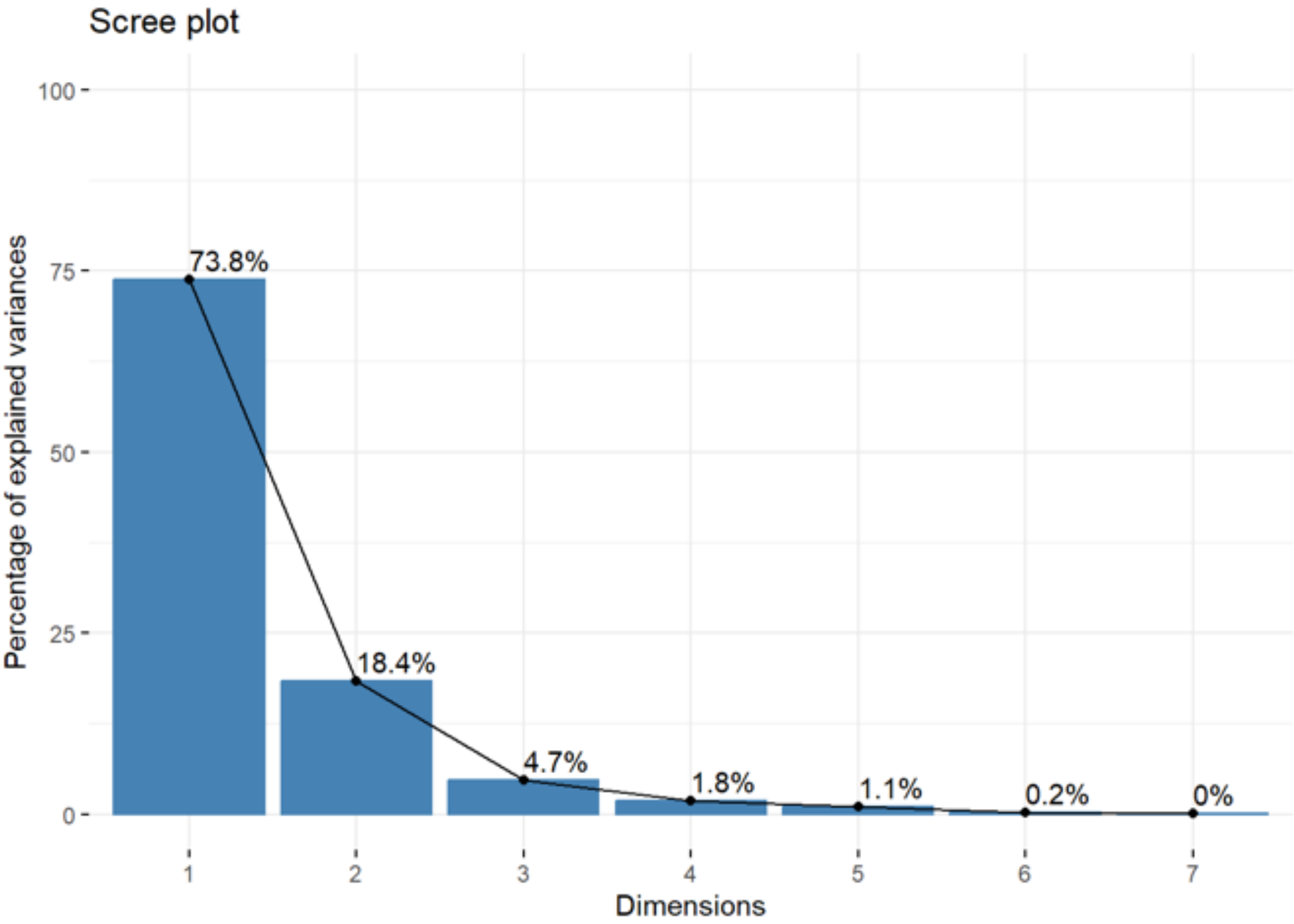

## Part II: Principal Component Analysis

Obtain principal components with … function […] and extract principal components, loadings and series-specific results …

```
# Obtain principal components
PCA <- prcomp(IP.series, scale = TRUE)

# Extract the principal component and specify them to a time series
IP.PCs <- ts(PCA$x, start = start(IP.series), frequency = 12)
# Extract the loadings
IP.load <- PCA$rotation
# Extract the results for variables
PCA.var <- get_pca_var(PCA)
```

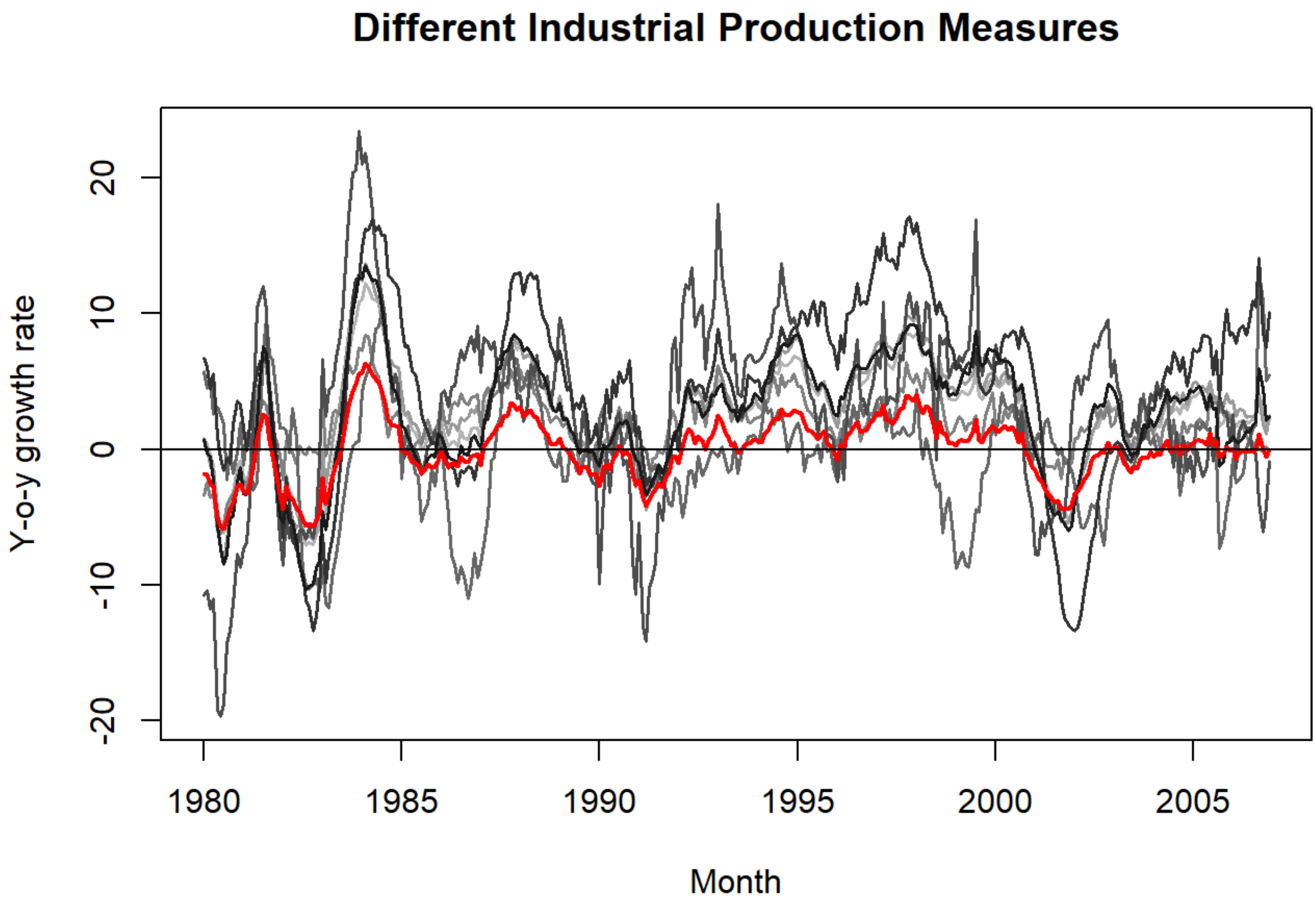
Visualise variation explained by each principal component (scree plot)

```
fviz_eig(PCA, addlabels = TRUE, ylim = c(0, 100))
```



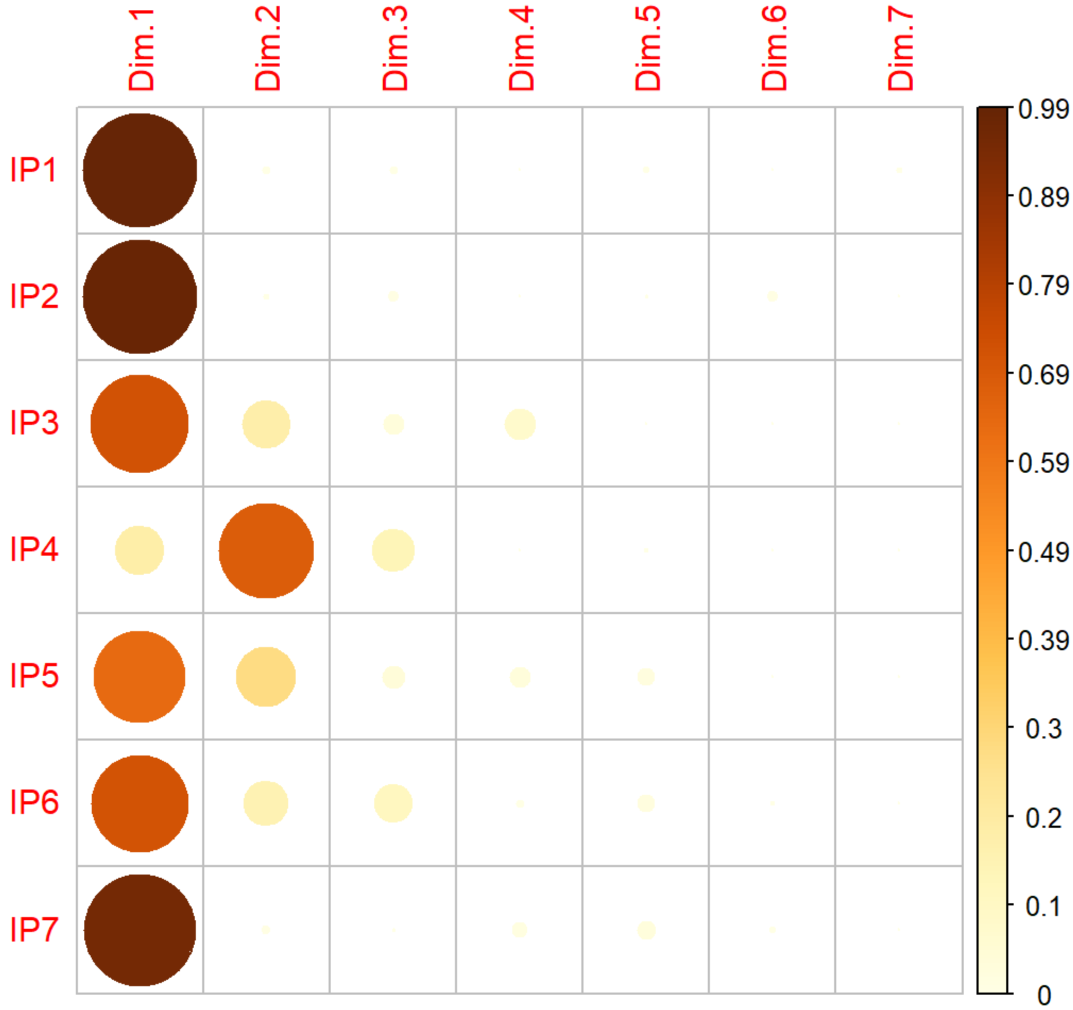

Plot the first principal component together with all the series

```
# Time series plot together with first principal component
ts.plot(ts.intersect(IP.series, IP.PCs[,1]),
        main = "Different Industrial Production Measures",    # Title of plot
        ylab = "Y-o-y growth rate",                            # Label of y-axis
        xlab = "Month",                                        # Label of x-axis
        col  =  c(gray(seq(0.7, 0.1,length.out = 7)), "red"), # Define colours
        lty  = "solid",                                        # Define line types
        lwd  = c(rep(1.5, 7), 2)                               # Define line width
)
abline(h = 0) # Horizontal line at zero
```




Series-specific variation explained by each principal component

```
corrplot::corrplot(PCA.var$cos2, is.corr=FALSE)
```




## Part III: A Factor-augmented VAR

Extract the first principal component and augment other macro variables with this first principal component

```
# Extract the first principal component
IP.PC1  <- IP.PCs[,1]

# Augment other macro variables with this first principal component
favar.data <- ts.intersect(IP.PC1, usmacro[,c("CPI", "FFR")])
colnames(favar.data) <- c("IP.PC1", "CPI", "FFR")
```

Ordering of variables matters for Cholesky [...] We will use the following ordering for our FA-VAR

```
var.order <- c("IP.PC1",   # real activity component: slow-moving (1st variable)
               "CPI",      # price index: slow-moving             (2nd variable)
               "FFR"       # policy rate: fast-moving             (3rd variable)
          )

favar.data   <- favar.data[,var.order]  # Reorder columns in dataset
```

We can use to the command `VAR()` to estimate a VAR model. Estimate a FA-VAR model, with twelve lags and include an intercept [...]

```
# Estimate a FA-VAR(12) model, including an intercept
fa.var <- vars::VAR(favar.data,     # Available data (3 endogenous variables)
                    p = 12,         # Consider 12 lags (monthly data)
                    type = "const") # Include an intercept
```

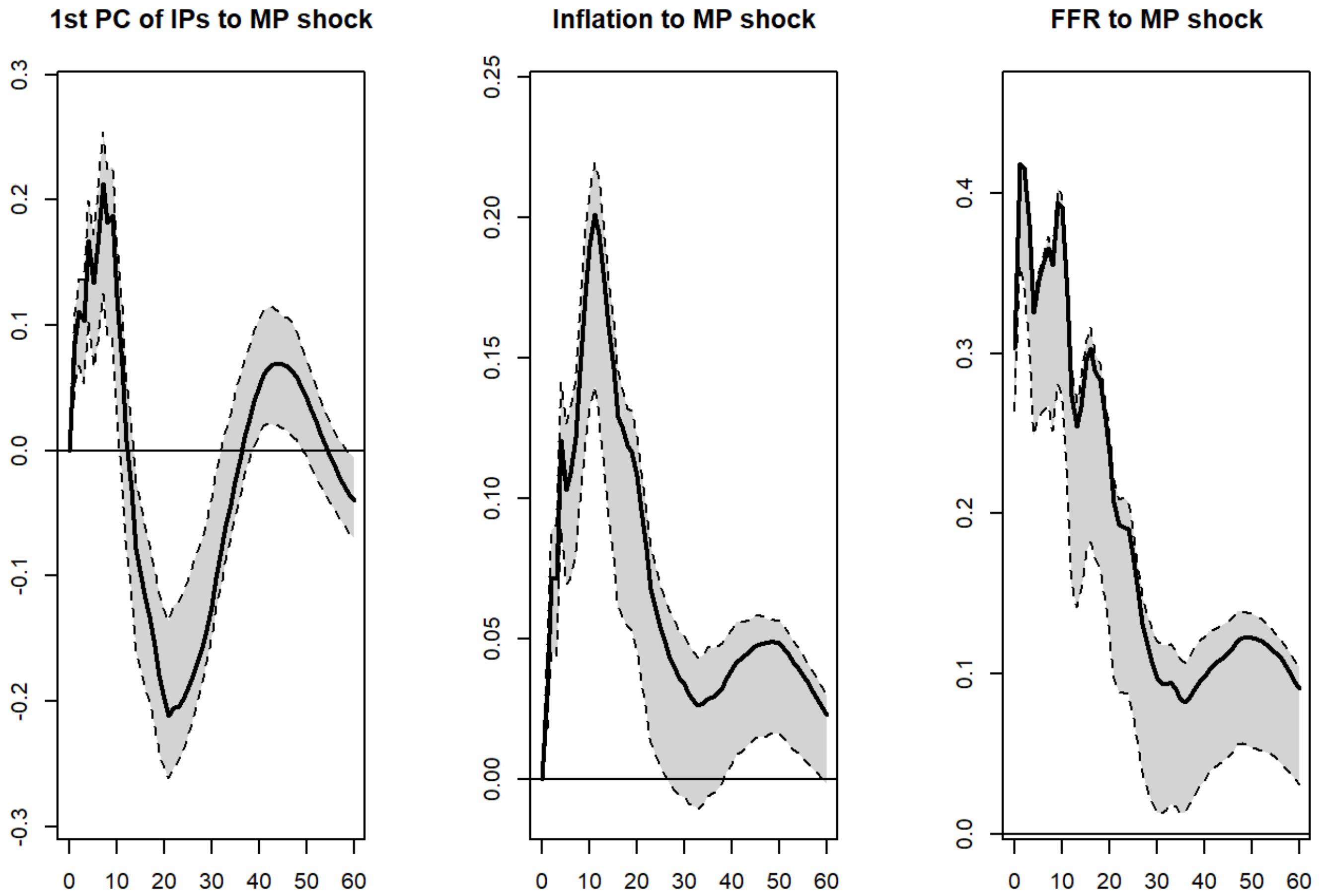
Again, we will use impulse response functions (IRFs) to analyse the dynamic response over time of endogenous variables to a monetary policy shock. For identification we will use Cholesky, we will impose a **positive one standard deviation shock** and consider a **maximum horizon of 60 months (i.e., 5 years)**.

```
# Obtain impulse responses
nhor <- 60 # Maximum horizon of impulse responses
irf.favar <- irf(fa.var,                   # VAR object
                 impulse = "FFR",          # Impulse variable (in this case a "monetar
y policy shock")
                 response =  var.order, # Response variables
                 n.ahead = nhor,
                 ci = 0.67,                # Confidence interval (67% corresponds to
+/- one standard deviation)
                 runs = 1000)              # Number of bootstrapping runs to obtain co
nfidence intervals
```

To plot impulse responses, we could simply use `plot()` , but these default plots typically do not look nice.

```
# Plot impulse responses
irf.mp   <- data.frame(irf.favar$irf$FFR, irf.favar$Lower$FFR, irf.favar$Upper$FFR)
colnames(irf.mp) <- c(paste0(var.order, ".m"),
                      paste0(var.order, ".l"),
                      paste0(var.order, ".u"))

par(mfrow=c(1,3)) # Divides the plotting window into three separate panels (3 colum
ns)

# 1.) IRF of 1st PC of IPs to MP shock
plot(x = 0:nhor,
     y = irf.mp$IP.PC1.m, type = "l", main = "1st PC of IPs to MP shock",
     xlab = '', ylab = '',
     ylim = c(min(irf.mp$IP.PC1.l), max(irf.mp$IP.PC1.u))*1.1)
polygon(x = c(0:nhor, rev(0:nhor)),
        y = c(irf.mp$IP.PC1.u, rev(irf.mp$IP.PC1.l)),
        col = 'lightgray',
        lty = 0)
lines(x = 0:nhor, y = irf.mp$IP.PC1.m, lty=1, col = "black", lwd = 2) # Solid line
for median response
lines(x = 0:nhor, y = irf.mp$IP.PC1.l,  lty=2, col = "black", lwd = 1) # Dashed lin
es for bounds
lines(x = 0:nhor, y = irf.mp$IP.PC1.u,  lty=2, col = "black", lwd = 1)
abline(h=0) # Horizontal zero line

# 2.) IRF of Inflation to MP shock
plot(x = 0:nhor,
     y = irf.mp$CPI.m, type = "l", main = "Inflation to MP shock",
     xlab = '', ylab = '',
```

```
     ylim = c(min(irf.mp$CPI.l), max(irf.mp$CPI.u))*1.1)
polygon(x = c(0:nhor, rev(0:nhor)),
        y = c(irf.mp$CPI.u, rev(irf.mp$CPI.l)),
        col = 'lightgray',
        lty = 0)
lines(x = 0:nhor, y = irf.mp$CPI.m, lty=1, col = "black", lwd = 2) # Solid line for
median response
lines(x = 0:nhor, y = irf.mp$CPI.l,  lty=2, col = "black", lwd = 1) # Dashed lines
for bounds
lines(x = 0:nhor, y = irf.mp$CPI.u,  lty=2, col = "black", lwd = 1)
abline(h=0) # Horizontal zero line

# 3.) IRF of FFR to MP shock
plot(x = 0:nhor,
     y = irf.mp$FFR.m, type = "l", main = "FFR to MP shock",
     xlab = '', ylab = '',
     ylim = c(min(irf.mp$FFR.l), max(irf.mp$FFR.u))*1.1)
polygon(x = c(0:nhor, rev(0:nhor)),
        y = c(irf.mp$FFR.u, rev(irf.mp$FFR.l)),
        col = 'lightgray',
        lty = 0)
lines(x = 0:nhor, y = irf.mp$FFR.m, lty=1, col = "black", lwd = 2) # Solid line for
median response
lines(x = 0:nhor, y = irf.mp$FFR.l,  lty=2, col = "black", lwd = 1) # Dashed lines
for bounds
lines(x = 0:nhor, y = irf.mp$FFR.u,  lty=2, col = "black", lwd = 1)
abline(h=0) # Horizontal zero line
```




FA-VAR is a powerful tool. It allows now to obtain also series-specific impulse responses for each IP measure. [...]

```
# Impulse responses to each component (point estimate only)
# Duplicate the response from the common component for each series
irf.IPs <- matrix(rep(irf.mp$IP.PC1.m, each = ncol(IP.series)), ncol(IP.series), nh
or)
```

```
## Warning in matrix(rep(irf.mp$IP.PC1.m, each = ncol(IP.series)),
## ncol(IP.series), : data length [427] is not a sub-multiple or multiple of the
## number of columns [60]
```

```r
rownames(irf.IPs) <- colnames(IP.series)

# Scale them with the associated loading of the 1st component
irf.IPs <- t(irf.IPs*IP.load[,1])
# Define them as a time series
irf.ip <- ts(irf.IPs, start = 0)

# Plot impulse responses of each IP series
dev.off()
ts.plot(irf.IPs,
        main = "IRFs of each IP series to MP shock",   # Title of plot
        ylab = "", xlab = "",
        col  =  c(gray(seq(0.7, 0.1,length.out = 7))), # Define colours
        lty  = "solid",                                # Define line types
        lwd  = 1.5                                     # Define line width
)
abline(h = 0)
```