VAR\_External\_Instrument\_ECB

MandarPriya\_Phatak

2024-09-21

Table of Contents

knitr::opts\_chunk$set(cache = TRUE, cache.lazy = FALSE, warning = FALSE,  
 message = FALSE,echo = TRUE, dpi = 360, warning = FALSE,  
 fig.width = 10, fig.height = 9)

# prequisite libraries

library(tidyquant)  
library(tidyverse)  
library(Matrix)  
library(readxl)  
library(dplyr)  
library(broom)  
library(modeltime)  
library(readr)  
library(vars)  
library(svars)

# reading the data

data <- read\_csv("data\_final.csv")  
data |> head()

## # A tibble: 6 × 7  
## date IP HICP de1y eurostoxx50 bbb\_spread ois\_1y  
## <date> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1999-01-01 4.44 4.30 2.86 8.16 7.07 -0.250  
## 2 1999-02-01 4.43 4.31 2.99 8.15 7.53 0   
## 3 1999-03-01 4.44 4.31 2.84 8.17 7.35 1.00   
## 4 1999-04-01 4.44 4.31 2.60 8.21 6.99 -0.900  
## 5 1999-05-01 4.44 4.31 2.65 8.21 6.83 0.700  
## 6 1999-06-01 4.45 4.31 2.90 8.23 6.61 -0.500

## Data exploration using skimr

library(skimr)  
  
data\_var <- data |>   
 dplyr::select(-date,-ois\_1y)  
  
skim(data = data\_var)

Data summary

|  |  |
| --- | --- |
| Name | data\_var |
| Number of rows | 261 |
| Number of columns | 5 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| numeric | 5 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IP | 0 | 1 | 4.55 | 0.05 | 4.27 | 4.51 | 4.54 | 4.59 | 4.65 | ▁▁▂▇▅ |
| HICP | 0 | 1 | 4.52 | 0.12 | 4.30 | 4.42 | 4.53 | 4.60 | 4.82 | ▅▅▇▃▁ |
| de1y | 0 | 1 | 1.58 | 1.83 | -0.85 | -0.04 | 1.32 | 3.20 | 5.17 | ▇▃▃▅▂ |
| eurostoxx50 | 0 | 1 | 8.09 | 0.21 | 7.60 | 7.94 | 8.09 | 8.23 | 8.58 | ▂▇▇▆▂ |
| bbb\_spread | 0 | 1 | 6.09 | 3.67 | 1.83 | 3.55 | 4.92 | 7.34 | 21.79 | ▇▃▂▁▁ |

library(DataExplorer)  
DataExplorer::create\_report(data = data\_var)

## | | | 0% | |. | 2% | |.. | 5% [global\_options] | |... | 7% | |.... | 10% [introduce] | |.... | 12% | |..... | 14% [plot\_intro] | |...... | 17% | |....... | 19% [data\_structure] | |........ | 21% | |......... | 24% [missing\_profile] | |.......... | 26% | |........... | 29% [univariate\_distribution\_header] | |........... | 31% | |............ | 33% [plot\_histogram] | |............. | 36% | |.............. | 38% [plot\_density] | |............... | 40% | |................ | 43% [plot\_frequency\_bar] | |................. | 45% | |.................. | 48% [plot\_response\_bar] | |.................. | 50% | |................... | 52% [plot\_with\_bar] | |.................... | 55% | |..................... | 57% [plot\_normal\_qq] | |...................... | 60% | |....................... | 62% [plot\_response\_qq] | |........................ | 64% | |......................... | 67% [plot\_by\_qq] | |.......................... | 69% | |.......................... | 71% [correlation\_analysis] | |........................... | 74% | |............................ | 76% [principal\_component\_analysis] | |............................. | 79% | |.............................. | 81% [bivariate\_distribution\_header] | |............................... | 83% | |................................ | 86% [plot\_response\_boxplot] | |................................. | 88% | |................................. | 90% [plot\_by\_boxplot] | |.................................. | 93% | |................................... | 95% [plot\_response\_scatterplot] | |.................................... | 98% | |.....................................| 100% [plot\_by\_scatterplot]   
## /Applications/RStudio.app/Contents/Resources/app/quarto/bin/tools/aarch64/pandoc +RTS -K512m -RTS /Users/mandarphatak/Downloads/report.knit.md --to html4 --from markdown+autolink\_bare\_uris+tex\_math\_single\_backslash --output /Users/mandarphatak/Downloads/report.html --lua-filter /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/rmarkdown/rmarkdown/lua/pagebreak.lua --lua-filter /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/rmarkdown/rmarkdown/lua/latex-div.lua --embed-resources --standalone --variable bs3=TRUE --section-divs --table-of-contents --toc-depth 6 --template /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/rmarkdown/rmd/h/default.html --no-highlight --variable highlightjs=1 --variable theme=yeti --mathjax --variable 'mathjax-url=https://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML\_HTMLorMML' --include-in-header /var/folders/6w/k13p0nq95dggbfm0scx2d38r0000gn/T//RtmpIxTj1L/rmarkdown-str731626939b10.html

## Selecting the number of lags, and no intercept  
VARselect(data\_var, lag.max = 12,type = "none")

## $selection  
## AIC(n) HQ(n) SC(n) FPE(n)   
## 6 3 2 6   
##   
## $criteria  
## 1 2 3 4 5  
## AIC(n) -2.926275e+01 -2.978548e+01 -3.000175e+01 -2.992854e+01 -2.998634e+01  
## HQ(n) -2.912060e+01 -2.950118e+01 -2.957530e+01 -2.935994e+01 -2.927558e+01  
## SC(n) -2.890959e+01 -2.907917e+01 -2.894228e+01 -2.851591e+01 -2.822055e+01  
## FPE(n) 1.955962e-13 1.159905e-13 9.348041e-14 1.006820e-13 9.518607e-14  
## 6 7 8 9 10  
## AIC(n) -3.006240e+01 -3.004426e+01 -2.996466e+01 -2.989539e+01 -2.985865e+01  
## HQ(n) -2.920949e+01 -2.904919e+01 -2.882744e+01 -2.861603e+01 -2.843713e+01  
## SC(n) -2.794346e+01 -2.757215e+01 -2.713939e+01 -2.671697e+01 -2.632707e+01  
## FPE(n) 8.843366e-14 9.036774e-14 9.831287e-14 1.060016e-13 1.108073e-13  
## 11 12  
## AIC(n) -2.985071e+01 -2.975175e+01  
## HQ(n) -2.828704e+01 -2.804593e+01  
## SC(n) -2.596598e+01 -2.551385e+01  
## FPE(n) 1.127391e-13 1.258839e-13

## We can see that number of lags based on bic is 2

# VAR model without constant  
var <- vars::VAR(data\_var,lag.max = 12,ic = "SC",type = "none")  
  
  
  
## Residual   
  
res = data.frame(residuals(var))  
  
  
## Instrument   
  
instrument <-(data[,"ois\_1y"])  
  
  
instrument <- instrument[3:nrow(instrument), , drop = FALSE]  
  
## Aligining the dependent based on ordering  
  
  
# Extract column names from the residuals dataframe  
seriesnames <- colnames(res)  
origorder <- seriesnames  
  
# Check if the dependent variable (assuming "ffr") is in the residuals  
dependent <- "de1y"  
if (dependent %in% seriesnames) {  
 # Reorder columns to put dependent first  
 seriesnames <- seriesnames[seriesnames != dependent]  
 seriesnames <- c(dependent, seriesnames)  
   
 # Reorder the columns in res  
 res <- res[, seriesnames]  
} else {  
 stop(paste("The series you are trying to instrument (", dependent, ") is not a series in the residual dataframe.", sep =""))  
}  
  
## combining the instrument with the residual  
  
  
res[,"instrument"] <- instrument  
  
# Create matrix u from the reordered residuals  
u <- as.matrix(res[, seriesnames])  
  
# Remove rows with NA values in the instrument column  
u <- u[!is.na(res$instrument), ]  
  
print("Step 2 completed: Residual matrix created.")

## [1] "Step 2 completed: Residual matrix created."

print(dim(u))

## [1] 259 5

## Intermediate steps   
  
# Useful constants  
 T <- nrow(u)  
 k <- ncol(u)  
 p <- 2  
  
 # Some necessary parts of the covariance matrix  
 gamma <- (1 / (T - k\*p - 1)) \* t(u) %\*% u  
 gamma\_11 <- gamma[1,1]  
 gamma\_21 <- matrix(gamma[2:nrow(gamma), 1], c(k-1,1))  
 gamma\_22 <- matrix(gamma[2:nrow(gamma), 2:nrow(gamma)], c(k-1,k-1))  
  
  
## First and Second stage regression   
  
# First stage regression  
firststage <- lm(de1y ~ instrument, data = res)  
res[names(predict(firststage)), "fs"] <- predict(firststage)  
  
print("Step 4 completed: First stage regression performed.")

## [1] "Step 4 completed: First stage regression performed."

print(summary(firststage))

##   
## Call:  
## lm(formula = de1y ~ instrument, data = res)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.90120 -0.08714 -0.00223 0.09165 0.70905   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.002036 0.010363 -0.196 0.844   
## instrument 0.009984 0.002176 4.588 7.02e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1666 on 257 degrees of freedom  
## Multiple R-squared: 0.07569, Adjusted R-squared: 0.07209   
## F-statistic: 21.05 on 1 and 257 DF, p-value: 7.016e-06

# Now get the second-stage coefficients  
coefs <- rep(0, k)  
names(coefs) <- seriesnames  
for (i in 1:k) {  
 s <- seriesnames[i]  
 if (s != "de1y") { # Using "d" as the instrumented variable  
 secondstage <- lm(as.formula(paste(s, " ~ fs")), res)  
 coefs[i] <- secondstage$coefficients["fs"]  
 } else {  
 coefs[i] <- 1  
 }  
}  
  
print("Step 5 completed: Second stage regression performed.")

## [1] "Step 5 completed: Second stage regression performed."

print(coefs)

## de1y IP HICP eurostoxx50 bbb\_spread   
## 1.0000000000 -0.0271950216 0.0005543646 0.0732696012 -0.4361289417

## Identitfying the factor   
  
s21\_on\_s11 <- matrix(coefs[2:k], c(k-1,1))  
  
Q <- (s21\_on\_s11 \* gamma\_11) %\*% t(s21\_on\_s11) - (gamma\_21 %\*% t(s21\_on\_s11) + s21\_on\_s11 %\*% t(gamma\_21)) + gamma\_22  
  
s12s12 <- t(gamma\_21 - s21\_on\_s11 \* gamma\_11) %\*% solve(Q) %\*% (gamma\_21 - s21\_on\_s11 \* gamma\_11)  
  
s11\_squared <- gamma\_11 - s12s12  
  
sp <- as.numeric(sqrt(s11\_squared))  
  
print("Step 6 completed: Intermediate matrices calculated.")

## [1] "Step 6 completed: Intermediate matrices calculated."

print("sp value:")

## [1] "sp value:"

print(sp)

## [1] 0.1636735

result <- sp \* coefs[origorder]  
print("Step 7 completed: Final result calculated.")

## [1] "Step 7 completed: Final result calculated."

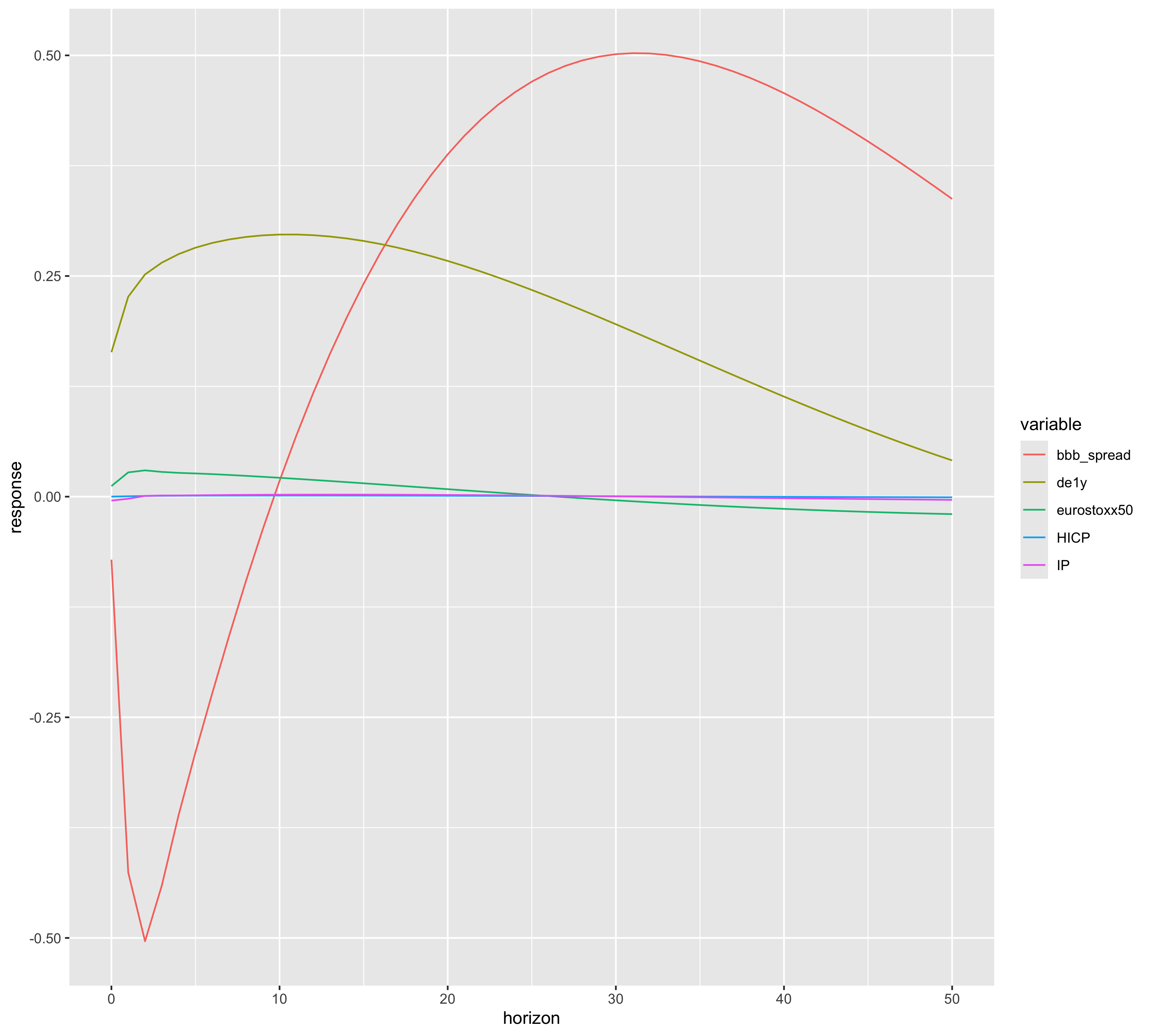
print("Final result:")

## [1] "Final result:"

print(result)

## IP HICP de1y eurostoxx50 bbb\_spread   
## -4.451104e-03 9.073478e-05 1.636735e-01 1.199229e-02 -7.138274e-02

## IRFs plot  
  
ma\_representation <- Phi(var, 50)  
irfs <- apply(ma\_representation, 3, function(x) x %\*% result)  
irfs <- as.data.frame(t(irfs))  
colnames(irfs) <- names(result)  
irfs <- mutate(irfs, horizon = 0:50)  
irfs <- gather(irfs, key = variable, value = response, -horizon)  
ggplot(irfs, aes(x = horizon, y = response, group = variable, color = variable)) + geom\_line()

 ### Checking with Robust Errors:- using vcovHAC, waldtest

library(sandwich)  
library(lmtest)  
  
var <- vars::VAR(data\_var, lag.max = 12, ic = "SC", type = "none")  
  
res = data.frame(residuals(var))

instrument <- (data[,"ois\_1y"])  
instrument <- instrument[3:nrow(instrument), , drop = FALSE]

# Extract column names from the residuals dataframe  
seriesnames <- colnames(res)  
origorder <- seriesnames  
  
# Check if the dependent variable (assuming "ffr") is in the residuals  
dependent <- "de1y"  
if (dependent %in% seriesnames) {  
 # Reorder columns to put dependent first  
 seriesnames <- seriesnames[seriesnames != dependent]  
 seriesnames <- c(dependent, seriesnames)  
   
 # Reorder the columns in res  
 res <- res[, seriesnames]  
} else {  
 stop(paste("The series you are trying to instrument (", dependent, ") is not a series in the residual dataframe.", sep =""))  
}  
  
res[,"instrument"] <- instrument  
  
# Create matrix u from the reordered residuals  
u <- as.matrix(res[, seriesnames])  
  
# Remove rows with NA values in the instrument column  
u <- u[!is.na(res$instrument), ]  
  
print("Step 2 completed: Residual matrix created.")

## [1] "Step 2 completed: Residual matrix created."

print(dim(u))

## [1] 259 5

# Useful constants  
T <- nrow(u)  
k <- ncol(u)  
p <- 2  
  
# Some necessary parts of the covariance matrix  
gamma <- (1 / (T - k\*p - 1)) \* t(u) %\*% u  
gamma\_11 <- gamma[1,1]  
gamma\_21 <- matrix(gamma[2:nrow(gamma), 1], c(k-1,1))  
gamma\_22 <- matrix(gamma[2:nrow(gamma), 2:nrow(gamma)], c(k-1,k-1))  
  
# First stage regression  
firststage <- lm(de1y ~ instrument, data = res)  
res[names(predict(firststage)), "fs"] <- predict(firststage)  
  
print("Step 4 completed: First stage regression performed.")

## [1] "Step 4 completed: First stage regression performed."

print(summary(firststage))

##   
## Call:  
## lm(formula = de1y ~ instrument, data = res)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.90120 -0.08714 -0.00223 0.09165 0.70905   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.002036 0.010363 -0.196 0.844   
## instrument 0.009984 0.002176 4.588 7.02e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1666 on 257 degrees of freedom  
## Multiple R-squared: 0.07569, Adjusted R-squared: 0.07209   
## F-statistic: 21.05 on 1 and 257 DF, p-value: 7.016e-06

# Calculate robust standard errors for first stage  
robust\_se\_first <- sqrt(diag(vcovHAC(firststage)))  
print("Robust standard errors for first stage:")

## [1] "Robust standard errors for first stage:"

print(robust\_se\_first)

## (Intercept) instrument   
## 0.01008419 0.00257983

# Calculate robust F-statistic  
robust\_f <- waldtest(firststage, vcov = vcovHAC(firststage))  
print("Robust F-statistic:")

## [1] "Robust F-statistic:"

print(robust\_f)

## Wald test  
##   
## Model 1: de1y ~ instrument  
## Model 2: de1y ~ 1  
## Res.Df Df F Pr(>F)   
## 1 257   
## 2 258 -1 14.978 0.000138 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# Now get the second-stage coefficients  
coefs <- rep(0, k)  
names(coefs) <- seriesnames  
robust\_se\_second <- list()  
for (i in 1:k) {  
 s <- seriesnames[i]  
 if (s != "de1y") { # Using "d" as the instrumented variable  
 secondstage <- lm(as.formula(paste(s, " ~ fs")), res)  
 coefs[i] <- secondstage$coefficients["fs"]  
   
 # Calculate robust standard errors for second stage  
 robust\_se\_second[[s]] <- sqrt(diag(vcovHC(secondstage, type = "HC1")))  
 print(paste("Robust standard errors for", s, "in second stage:"))  
 print(robust\_se\_second[[s]])  
 } else {  
 coefs[i] <- 1  
 }  
}

## [1] "Robust standard errors for IP in second stage:"  
## (Intercept) fs   
## 0.001340468 0.024550264   
## [1] "Robust standard errors for HICP in second stage:"  
## (Intercept) fs   
## 0.0001649584 0.0047209068   
## [1] "Robust standard errors for eurostoxx50 in second stage:"  
## (Intercept) fs   
## 0.002938334 0.095242589   
## [1] "Robust standard errors for bbb\_spread in second stage:"  
## (Intercept) fs   
## 0.05008521 1.50114509

print("Step 5 completed: Second stage regression performed.")

## [1] "Step 5 completed: Second stage regression performed."

print(coefs)

## de1y IP HICP eurostoxx50 bbb\_spread   
## 1.0000000000 -0.0271950216 0.0005543646 0.0732696012 -0.4361289417

s21\_on\_s11 <- matrix(coefs[2:k], c(k-1,1))  
  
Q <- (s21\_on\_s11 \* gamma\_11) %\*% t(s21\_on\_s11) - (gamma\_21 %\*% t(s21\_on\_s11) + s21\_on\_s11 %\*% t(gamma\_21)) + gamma\_22  
  
s12s12 <- t(gamma\_21 - s21\_on\_s11 \* gamma\_11) %\*% solve(Q) %\*% (gamma\_21 - s21\_on\_s11 \* gamma\_11)  
  
s11\_squared <- gamma\_11 - s12s12  
  
sp <- as.numeric(sqrt(s11\_squared))  
  
print("Step 6 completed: Intermediate matrices calculated.")

## [1] "Step 6 completed: Intermediate matrices calculated."

print("sp value:")

## [1] "sp value:"

print(sp)

## [1] 0.1636735

result <- sp \* coefs[origorder]  
print("Step 7 completed: Final result calculated.")

## [1] "Step 7 completed: Final result calculated."

print("Final result:")

## [1] "Final result:"

print(result)

## IP HICP de1y eurostoxx50 bbb\_spread   
## -4.451104e-03 9.073478e-05 1.636735e-01 1.199229e-02 -7.138274e-02

ma\_representation <- Phi(var, 50)  
irfs <- apply(ma\_representation, 3, function(x) x %\*% result)  
irfs <- as.data.frame(t(irfs))  
colnames(irfs) <- names(result)  
irfs <- mutate(irfs, horizon = 0:50)  
irfs <- gather(irfs, key = variable, value = response, -horizon)  
ggplot(irfs, aes(x = horizon, y = response, group = variable, color = variable)) + geom\_line()

