Dissertation

Samid Ali

Contents

Chapter 1

Introduction

Chapter 2

Literature Review

Chapter 3

Data

Discuss Empirical model, lag structure, data sources, potentially features of the data

Chapter 4

Identification

How we recover the structural shocks to conduct meaningful analysis. Recall The reduced form residuals are correlated and lack an economic interpretation. Identification measure requires economic justification. Use short run restrictions with a recursive structure for this. Following Fernandez (2006)

```
## Classes 'data.table' and 'data.frame':   152 obs. of  6 variables:
##  $ CDID    : chr  "1987 Q1" "1987 Q2" "1987 Q3" "1987 Q4" ...
##  $ Deflator: num  34.8 35.3 36 36.3 36.7 ...
##  $ GDP     : num  314804 319478 327346 331075 336442 ...
##  $ Year    : num  1987 1987 1987 1987 1988 ...
##  $ Quarter : chr  "Q1" "Q2" "Q3" "Q4" ...
##  $ Q       : num  1 2 3 4 1 2 3 4 1 2 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

Identification:

Chapter 5

Recovering Structural Shocks

Let Y_t be the vector of variables:

$$Y_t = \begin{pmatrix} G_t \\ R_t \\ GDP_t \\ T_t \\ P_t \end{pmatrix}$$

The reduced-form VAR model can be written as:

$$Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \cdots + A_p Y_{t-p} + \epsilon_t$$

where ϵ_t is the vector of reduced-form residuals. To recover the structural shocks u_t, we assume:

$$\epsilon_t = B u_t$$

where B is a lower triangular matrix. The structural shocks u_t are assumed to be uncorrelated and have unit variance.

The matrix B can be obtained using Cholesky decomposition of the covariance matrix of the reduced-form residuals Σ_ϵ:

$$\Sigma_\epsilon = E[\epsilon_t \epsilon_t'] = BB'$$

Given the recursive ordering (G, R, GDP, T, P), the matrix B has the form:

$$B = \begin{pmatrix} b_{11} & 0 & 0 & 0 & 0 \\ b_{21} & b_{22} & 0 & 0 & 0 \\ b_{31} & b_{32} & b_{33} & 0 & 0 \\ b_{41} & b_{42} & b_{43} & b_{44} & 0 \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{pmatrix}$$

Thus, the structural shocks u_t can be recovered as:

$$u_t = B^{-1}\epsilon_t$$

5.0.1 Appendix

```
library(ggplot2)
library(knitr)
library(ivreg)
library(ggdag)
library(data.table)
library(dplyr)
library(tidyr)
library(stargazer)
library(clipr)
library(tibble)
library(lubridate)
#library(kableExtra)

knitr::opts_chunk$set(echo = FALSE)



df <- fread("D:/Samid work/University/KCL - Econ and Policy/Dissertation/Data/GDP.csv", skip


# Filter the data frame to exclude rows where the column 'title' matches any of the specifi
filtered_df <- df %>%
  # Only keep the quarterly data
  subset(nchar(CDID) == 7 & substr(CDID,6,6) == "Q") %>%
  select(CDID,Deflator = L8GG, GDP = ABMI) %>%
  mutate(Year = as.numeric(substr(CDID,1,4)),
         Quarter = substr(CDID,6,7),
         Q  = as.numeric(substr(CDID,7,7))) %>%
  # For testing purposes, can change later
  subset(Year >= 1987)  %>%
```

```r
  mutate(Deflator = as.numeric(Deflator),
         GDP = as.numeric(GDP))


# View the filtered data frame
# head(filtered_df)

str(filtered_df)


# L8GG


fiscal_raw <- fread("D:/Samid work/University/KCL - Econ and Policy/Dissertation/Data/Fiscal

fiscal_proc <- fiscal_raw %>%
  select(Date_ID = Transaction, Revenue = OTR, Expenditure = OTE) %>%
  subset(Date_ID !=  "Dataset identifier code" & Date_ID != "Identifier") %>%
  mutate(Year = as.numeric(gsub("\\D", "", Date_ID)),
         Period = gsub("\\d{4}", "", Date_ID))  %>%
  mutate(
    Q = case_when(
      Period == "Jan to Mar " ~ 1,
      Period == "Apr to Jun " ~ 2,
      Period == "Jul to Sep " ~ 3,
      Period == "Oct to Dec " ~ 4
      ),
    Unique_Period = Year +(Q/4)

  ) %>%
  # Convert to numeric and multiply by 1 million so values as these will later be made into
  mutate(Revenue = as.numeric(gsub(",", "", Revenue) ),
         Expenditure = as.numeric(gsub(",", "", Expenditure )))

# join GDP deflator and GDP data

population <- fread("D:/Samid work/University/KCL - Econ and Policy/Dissertation/Data/Popula
                    skip = 4,
                    header = TRUE) %>%
  subset(`Country Name` == "United Kingdom") %>%
  t() %>%
  as.data.frame() %>%
  rownames_to_column(var = "Year") %>%
  rename(Population = V1 ) %>%
  filter(grepl("^\\d{4}$", Year)) %>%
```

```r
  mutate(Year = as.numeric(Year),
         Population = as.numeric(Population))

Interest_SR <- fread("D:/Samid work/University/KCL - Econ and Policy/Dissertation/Data/3 Mon
  mutate(Date = dmy(Date),
         month = month(Date),
         Year = year(Date),
         Q = case_when(
           month %in% 1:3 ~ 1,
           month %in% 4:6 ~ 2,
           month %in% 7:9 ~ 3,
           month %in% 10:12 ~ 4
         )) %>%
  group_by(Year, Q) %>%
  summarize(mean_SR_Rate = mean(SR_Rate, na.rm = TRUE))


data <- fiscal_proc %>%
  left_join(filtered_df, by = c("Q" = "Q", "Year" = "Year")) %>%
  left_join(Interest_SR, by = c("Q" = "Q", "Year" = "Year")) %>%
  left_join(population, by = c("Year" = "Year")) %>%
# Convert variables to per capita, note revenue, expenditure, and GDP are in £ million so ne
  mutate(RevenuePerCapita = (Revenue *10^6) /Population,
         ExpenditurePerCapita = (Expenditure *10^6) /Population,
         GDPPerCapita = (GDP *10^6) /Population) %>%
# Convert variables (except interest rate) to logs
  mutate(log_revenue = log(Revenue *10^6),
         log_expenditure = log(Expenditure *10^6),
         log_GDP = log(GDP *10^6),
         log_deflator = log(Deflator))


model_data <- data %>%
  select(CDID, log_revenue, log_expenditure, log_GDP, log_deflator, mean_SR_Rate)





# ggplot(fiscal_proc, aes(x = Year2)) +
#   geom_line(aes(y = Revenue, color = "Revenue"), size = 1) +
#   geom_line(aes(y = Expenditure, color = "Expenditure"), size = 1) +
#   labs(
#     x = "Date ID",
#     y = "Amount (? in millions)",
```

```
#     title = "Revenue and Expenditure Over Time",
#     color = "Legend"
#   ) +
#   scale_color_manual(values = c("Revenue" = "blue", "Expenditure" = "red")) +
#   theme_minimal(base_size = 15) +
#   theme(
#     axis.text.x = element_text(angle = 45, hjust = 1),
#     plot.title = element_text(hjust = 0.5, face = "bold"),
#     axis.title.x = element_text(face = "bold"),
#     axis.title.y = element_text(face = "bold"),
#     legend.position = "bottom",
#     legend.title = element_text(face = "bold")
#   )
```