PS8 R Solutions

Matthew Alampay Davis

December 12, 2021

Parts a and b

```
Quandl.api_key('xHu2y3xExQ6bGkGqcYEi') # This is my personal API key
gdp <- Quandl('FRED/GDPC1')
m2 <- Quandl('FRED/M2REAL')
gov <- Quandl('FRED/GCEC1')
head(gdp)
```

```
##         Date    Value
## 1 2021-07-01 19469.40
## 2 2021-04-01 19368.31
## 3 2021-01-01 19055.65
## 4 2020-10-01 18767.78
## 5 2020-07-01 18560.77
## 6 2020-04-01 17258.21
```

```
head(m2)
```

```
##         Date  Value
## 1 2021-10-01 7656.4
## 2 2021-09-01 7658.3
## 3 2021-08-01 7630.3
## 4 2021-07-01 7558.1
## 5 2021-06-01 7538.8
## 6 2021-05-01 7597.1
```

```
head(gov)
```

```
##         Date    Value
## 1 2021-07-01 3381.311
## 2 2021-04-01 3373.765
## 3 2021-01-01 3390.921
## 4 2020-10-01 3356.030
## 5 2020-07-01 3360.238
## 6 2020-04-01 3378.132
```

Let's rename the variables for clarity:

```
names(gdp)[2] <- 'gdp'
names(m2)[2] <- 'm2'
names(gov)[2] <- 'gov'
```

Since M2 is a stock variable, you can take the value of this series corresponding to the first month of each quarter to get the quarterly values of this variable. The function *Quandl* already has an argument called *collapse* that lets you do this, but it changes the Date values for some reason. So instead, we will use our old friend *filter*:

```
# Choose observations for the months that begin each quarter
m2 %<>% filter(month(Date) %in% c(1,4,7,10))
```

Now let's merge all the variables into one dataset. Note that m2 data goes all the way to January 2021 but the GDP and gov data only goes to October 2020. So we cannot just bind them together:

```
q3.df <- merge(gdp, m2, by = c('Date')) %>%
  merge(gov, by = c('Date'))
head(q3.df, 10)
```

```
##          Date      gdp     m2      gov
## 1  1959-01-01 3123.978  987.9 1067.197
## 2  1959-04-01 3194.429 1001.0 1080.416
## 3  1959-07-01 3196.683 1012.7 1089.655
## 4  1959-10-01 3205.790 1010.2 1080.744
## 5  1960-01-01 3277.847 1015.3 1062.312
## 6  1960-04-01 3260.177 1015.9 1074.240
## 7  1960-07-01 3276.133 1029.1 1100.511
## 8  1960-10-01 3234.087 1040.3 1108.142
## 9  1961-01-01 3255.914 1052.6 1124.596
## 10 1961-04-01 3311.181 1073.1 1127.411
```

Now produce growth rates for all three variables. For a variable X_t, we'll compute the growth rate x_t using the following transformation:

$$x_t = \log(X_t) - \log(X_{t-1})$$

```
q3.df %<>% mutate(Y = log(gdp),
                  M = log(m2),
                  G = log(gov),
                  # Create lagged versions
                  Y.lag = lag(Y),
                  M.lag = lag(M),
                  G.lag = lag(G),
                  # Create the growth rates
                  y.growth = Y-Y.lag,
                  m.growth = M-M.lag,
                  g.growth = G-G.lag)
```

An identical way of doing this is:

```
q3.df %<>% mutate(Y = log(gdp),
                  M = log(m2),
                  G = log(gov),
                  # Create the growth rates by differencing
                  y.growth = c(NA, diff(Y)),
                  m.growth = c(NA, diff(M)),
                  g.growth = c(NA, diff(G)))
```

Let's add transformations that we'll need for later questions. Doing them here is best because we'll be filtering by Date below and that way we avoid producing unnecessary NAs when we produce lag and differenced terms:

```
# For part h
q3.df %<>% mutate(dmg = c(NA, diff(m.growth)),
                  dgg = c(NA, diff(g.growth)),
                  dmg1 = lag(dmg),
                  dgg1 = lag(dgg),
                  dmg2 = lag(dmg, 2),
                  dgg2 = lag(dgg, 2),
                  dmg3 = lag(dmg, 3),
                  dgg3 = lag(dgg, 3),
                  # Careful: the last lagged term is the non-differenced fourth lag
                  mg4 = lag(m.growth, 4),
                  gg4 = lag(g.growth, 4))
```

Finally, let's filter to only include dates from 1960q1 to 2019q4

```
q3.df %<>% filter(Date >= '1960-01-01' & Date <= '2019-10-01')
```

You can ignore this next chunk, but I will save this as a csv so that I can compare results to Stata. To do what we've just done above in Stata is stupidly inconvenient so I'm processing it in R before exporting it to Stata.

```
write.csv(q3.df, file = 'q3df.csv')
```

We want to run a series of distributed lag models using the dependent variable *y.growth*

Need to determine the truncation parameter:

```
m.trunc <- 0.75*nrow(q3.df)^(1/3)
m.trunc
```

```
## [1] 4.660849
```

```
m.trunc <- ceiling(m.trunc)
```

Part c: y.growth on m.growth

Then the regression we want to run is

```
q3c.mod <- lm(y.growth ~ m.growth, q3.df)
nw.se <- NeweyWest(q3c.mod,
                   lag = m.trunc,
                   prewhite = F,
                   adjust = T)
coeftest(q3c.mod, vcov = nw.se)
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 0.00663345 0.00075673  8.7659 3.579e-16 ***
## m.growth    0.11235652 0.07811470  1.4384    0.1516
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Part d: y.growth on g.growth

Then the regression we want to run is

```
q3d.mod <- lm(y.growth ~ g.growth, q3.df)
nw.se <- NeweyWest(q3d.mod,
                   lag = m.trunc,
                   prewhite = F,
                   adjust = T)
coeftest(q3d.mod, vcov = nw.se)
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 0.00654416 0.00067622  9.6775 < 2.2e-16 ***
## g.growth    0.19575106 0.05767480  3.3940 0.0008066 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Part e: y.growth on current and next-quarter monetary policy

```
q3e.mod <- lm(y.growth ~ m.growth + lag(m.growth), q3.df)
nw.se <- NeweyWest(q3e.mod,
                   lag = m.trunc,
                   prewhite = F,
                   adjust = T)
coeftest(q3e.mod, vcov = nw.se)
```

```
##
## t test of coefficients:
##
##                 Estimate Std. Error t value  Pr(>|t|)
```

```
## (Intercept)   0.00574539 0.00080015  7.1804 9.003e-12 ***
## m.growth      0.01944254 0.06310140  0.3081  0.758266
## lag(m.growth) 0.20616909 0.06482558  3.1804  0.001668 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Part f: y.growth on current and next-quarter fiscal policy

```
q3f.mod <- lm(y.growth ~ g.growth + lag(g.growth), q3.df)
nw.se <- NeweyWest(q3f.mod,
                   lag = m.trunc,
                   prewhite = F,
                   adjust = T)
coeftest(q3f.mod, vcov = nw.se)
```

```
##
## t test of coefficients:
##
##                  Estimate  Std. Error t value  Pr(>|t|)
## (Intercept)    0.00654381  0.00065308 10.0200 < 2.2e-16 ***
## g.growth       0.22352356  0.05201833  4.2970 2.532e-05 ***
## lag(g.growth) -0.04541766  0.04364314 -1.0407    0.2991
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Part g: y.growth on the first difference in current and four lags of m.growth and g.growth

Then we'll run the regression with four lags of each of *d.m.growth* and *d.g.growth*:

```
q3g.mod <- lm(y.growth ~ dmg + dmg1 + dmg2 + dmg3 + mg4 + dgg + dgg1 + dgg2 + dgg3 + gg4,
              data = q3.df)
nw.se <- NeweyWest(q3g.mod,
                   lag = m.trunc,
                   prewhite = F,
                   adjust = T)
test <- coeftest(q3g.mod, vcov = nw.se)
test
```

```
##
## t test of coefficients:
##
##                Estimate  Std. Error t value  Pr(>|t|)
## (Intercept)  0.00463575  0.00089094  5.2032 4.361e-07 ***
## dmg         -0.00969506  0.06395402 -0.1516 0.8796410
## dmg1         0.13874741  0.09659477  1.4364 0.1522631
## dmg2         0.21646125  0.09756735  2.2186 0.0275009 *
## dmg3         0.28519125  0.09110368  3.1304 0.0019738 **
## mg4          0.31323536  0.08327840  3.7613 0.0002149 ***
## dgg          0.20868931  0.05853726  3.5651 0.0004431 ***
```

```
## dgg1         0.13559736  0.08613928  1.5742 0.1168361
## dgg2         0.16952713  0.09342719  1.8145 0.0709093 .
## dgg3         0.08425752  0.07913311  1.0648 0.2881123
## gg4          0.09732794  0.07841499  1.2412 0.2158109
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To replicate the given subsample, you'd repeat the above filtering to years before 1981. Supposedly. I wasn't able to replicate those numbers in R nor Stata.

Part h

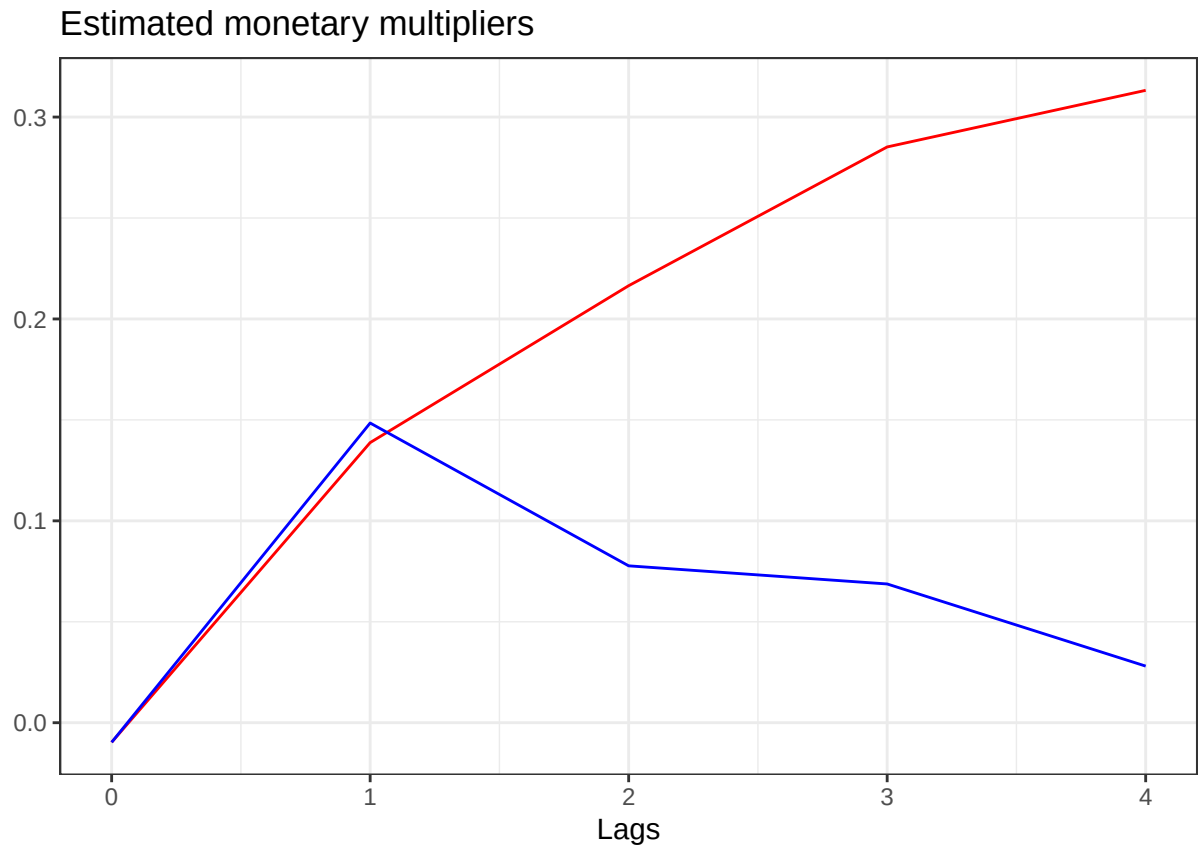
Assuming money and government expenditures are exogenous, the coefficients estimated represent the (monetary/fiscal) cumulative multipliers. If you differenced them out, they would represent the dynamic multipliers:

```
multipliers <- data.frame(lags = 0:4,
                          m.cum.mult = test[2:6],
                          f.cum.mult = test[7:11]) %>%
  mutate(m.dyn.mult = c(test[2], diff(test[2:6])),
         f.dyn.mult = c(test[7], diff(test[7:11])))
multipliers
```

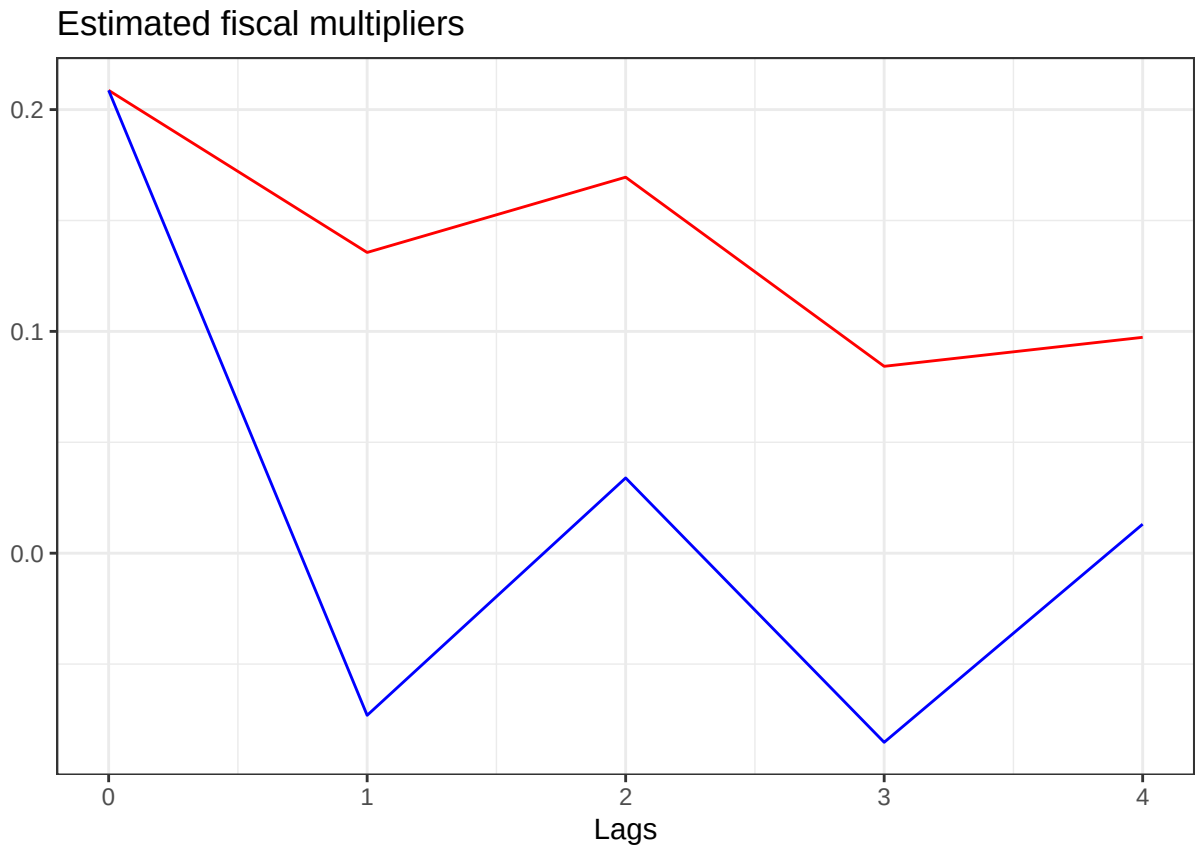
```
##   lags   m.cum.mult f.cum.mult   m.dyn.mult  f.dyn.mult
## 1    0 -0.009695062 0.20868931 -0.009695062  0.20868931
## 2    1  0.138747407 0.13559736  0.148442469 -0.07309195
## 3    2  0.216461254 0.16952713  0.077713847  0.03392977
## 4    3  0.285191248 0.08425752  0.068729995 -0.08526961
## 5    4  0.313235359 0.09732794  0.028044110  0.01307042
```

Part i

```
ggplot(multipliers, aes(x = lags)) +
  theme_bw() +
  geom_line(aes(y = m.cum.mult), col = 'red') +
  geom_line(aes(y = m.dyn.mult), col = 'blue') +
  ggtitle('Estimated monetary multipliers') +
  xlab('Lags') + ylab('')
```




```
ggplot(multipliers, aes(x = lags)) +
  theme_bw() +
  geom_line(aes(y = f.cum.mult), col = 'red') +
  geom_line(aes(y = f.dyn.mult), col = 'blue') +
  ggtitle('Estimated fiscal multipliers') +
  xlab('Lags') + ylab('')
```



Part j

There is little reason to believe that these government instruments are exogenous. Even if the monetary base and those components of government expenditures which do not respond to business cycle fluctuations had been chosen rather than the above regressors, then these instruments respond to changes in the growth rate of GDP. As a matter of fact, government reaction functions were also estimated at the time to capture how government instruments respond to changes in target variables. As a result, the regressors will be correlated with the error term, OLS estimation is inconsistent, and inference not dependable. It is hard to imagine how useable information can be retrieved from these numbers.