Modelo estimado utilizando R

Gabriel Petrini da Silveira

2019

Carregando pacotes

```
library(zoo)
library(xts)
library(tsDyn)
library(urca)
library(vars)
library(dplyr)
library(stargazer)
library(lmtest)
```

Carregando dados

```
df <- read.csv(
  "/dados/Dissertacao/Modelo/SeriesTemporais/Dados_completos.csv",
  encoding="UTF-8",
  stringsAsFactors=FALSE
  )
df <- ts(data = df, start = c(1987,01), frequency = 4)
#df <- as.xts(df)
df <- zoo::na.locf0(df)
```

Quebra estrutural

Taxa de crescimento do investimento residencial

```
result = breakpoints(gZ~1, data=df)
result$breakpoints %>% unique() %>% na.omit() %>% c() -> breaks

for(i in breaks){
  print(paste0("Testando para i = ", index(df)[i]))
  strucchange::sctest(gZ~1, data=df, point=i, type="Chow") %>% print()
}
```

```
## [1] "Testando para i = 1991.5"
##
##  Chow test
##
## data:  gZ ~ 1
## F = 5.1087, p-value = 0.02548
##
## [1] "Testando para i = 2005.75"
##
##  Chow test
##
## data:  gZ ~ 1
## F = 7.3106, p-value = 0.007779
##
## [1] "Testando para i = 2010.5"
##
##  Chow test
##
## data:  gZ ~ 1
## F = 6.073, p-value = 0.01504
```

Taxa Própria

```
result = breakpoints(Taxa.Própria~1, data=df)
result$breakpoints %>% unique() %>% na.omit() %>% c() -> breaks

for(i in breaks){
  print(paste0("Testando para i = ", index(df)[i]))
  strucchange::sctest(Taxa.Própria~1, data=df, point=i, type="Chow") %>% print()
}
```

```
## [1] "Testando para i = 1991.75"
##
##  Chow test
##
## data:  Taxa.Própria ~ 1
## F = 63.194, p-value = 8.177e-13
##
## [1] "Testando para i = 1996.5"
##
##  Chow test
##
## data:  Taxa.Própria ~ 1
## F = 107.12, p-value < 2.2e-16
##
## [1] "Testando para i = 2001.25"
##
##  Chow test
##
## data:  Taxa.Própria ~ 1
## F = 78.179, p-value = 5.995e-15
##
```

```
## [1] "Testando para i = 2006"
##
##  Chow test
##
## data:  Taxa.Própria ~ 1
## F = 20.637, p-value = 1.26e-05
##
## [1] "Testando para i = 2011"
##
##  Chow test
##
## data:  Taxa.Própria ~ 1
## F = 78.824, p-value = 4.885e-15
```

Taxa de juros

```r
result = breakpoints(Taxa.de.juros~1, data=df)
result$breakpoints %>% unique() %>% na.omit() %>% c() -> breaks

for(i in breaks){
  print(paste0("Testando para i = ", index(df)[i]))
  strucchange::sctest(Taxa.de.juros~1, data=df, point=i, type="Chow") %>% print()
}
```

```
## [1] "Testando para i = 1991.5"
##
##  Chow test
##
## data:  Taxa.de.juros ~ 1
## F = 124.35, p-value < 2.2e-16
##
## [1] "Testando para i = 1997"
##
##  Chow test
##
## data:  Taxa.de.juros ~ 1
## F = 199.25, p-value < 2.2e-16
##
## [1] "Testando para i = 2002"
##
##  Chow test
##
## data:  Taxa.de.juros ~ 1
## F = 301.18, p-value < 2.2e-16
##
## [1] "Testando para i = 2009.75"
##
##  Chow test
##
## data:  Taxa.de.juros ~ 1
## F = 172.97, p-value < 2.2e-16
```

Inflação

```
result = breakpoints(Inflação~1, data=df)
result$breakpoints %>% unique() %>% na.omit() %>% c() -> breaks

for(i in breaks){
  print(paste0("Testando para i = ", index(df)[i]))
  strucchange::sctest(Inflação~1, data=df, point=i, type="Chow") %>% print()
}
```

```
## [1] "Testando para i = 1997.5"
##
##  Chow test
##
## data:  Inflação ~ 1
## F = 1.5508, p-value = 0.2153
##
## [1] "Testando para i = 2005.75"
##
##  Chow test
##
## data:  Inflação ~ 1
## F = 23.49, p-value = 3.569e-06
##
## [1] "Testando para i = 2011.5"
##
##  Chow test
##
## data:  Inflação ~ 1
## F = 4.4981, p-value = 0.03586
```

Teste de Johansen

gZ e Taxa Própria

```
vars::VARselect(
  y = df[,c("gZ", "Taxa.Própria")] %>%  na.omit(),
  type="both"
)$selection[1] %>% as.numeric() -> p
urca::ca.jo(
  x = df[,c("gZ", "Taxa.Própria")],
  ecdet = "const",
  #ecdet = "trend",
  K = p-1,
  spec = "longrun",
  type = "trace"
) %>% summary()
```

```
##
## ######################
```

```
## # Johansen-Procedure #
## ######################
##
## Test type: trace statistic , without linear trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1]  1.430140e-01  2.264498e-02 -1.394865e-17
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 1 |  2.91  7.52  9.24 12.97
## r = 0  | 22.51 17.85 19.96 24.60
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                       gZ.l4 Taxa.Própria.l4  constant
## gZ.l4            1.00000000       1.0000000  1.000000
## Taxa.Própria.l4  0.16786715       2.2242844  1.171986
## constant        -0.02323547      -0.1388002 -0.163997
##
## Weights W:
## (This is the loading matrix)
##
##                      gZ.l4 Taxa.Própria.l4      constant
## gZ.d           -0.35519700     0.009248891  1.075133e-16
## Taxa.Própria.d  0.03814169    -0.022363632 -2.793648e-17
```

gZ, Inflação e Taxa de juros

```
vars::VARselect(
  y = df[,c("gZ", "Inflação", "Taxa.de.juros")] %>%  na.omit(),
  type="both"
)$selection[1] %>% as.numeric() -> p

urca::ca.jo(
  x = df[,c("gZ", "Inflação", "Taxa.de.juros")],
  ecdet = "const",
  #ecdet = "trend",
  K = p-1,
  spec = "longrun",
  type = "trace"
) %>% summary()
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: trace statistic , without linear trend and constant in cointegration
##
```

```
## Eigenvalues (lambda):
## [1] 2.178892e-01 6.632213e-02 4.980109e-02 3.274353e-17
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 2 |  6.44  7.52  9.24 12.97
## r <= 1 | 15.08 17.85 19.96 24.60
## r = 0  | 46.05 32.00 34.91 41.07
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                         gZ.l4 Inflação.l4 Taxa.de.juros.l4   constant
## gZ.l4             1.000000000   1.0000000        1.0000000  1.0000000
## Inflação.l4      -1.394466726 -96.2890267      -12.6243068 -1.0154784
## Taxa.de.juros.l4  0.065581748  18.0791283       -8.3286175  3.9018595
## constant         -0.004284139   0.1820761        0.3509467 -0.3056512
##
## Weights W:
## (This is the loading matrix)
##
##                        gZ.l4   Inflação.l4 Taxa.de.juros.l4      constant
## gZ.d            -0.375648992  0.0012899011      0.009290987 -7.037504e-17
## Inflação.d       0.072662883  0.0005128200      0.003264098 -2.840004e-17
## Taxa.de.juros.d  0.002156497 -0.0006490703      0.001206574  1.348445e-18
```

gZ, Inflação (Taxa de juros exog)

```r
df <- df[,c("gZ", "Inflação", "Taxa.de.juros")] %>% na.omit()

vars::VARselect(
  y = df[,c("gZ", "Inflação")],
  type="both",
  exogen = df[,c("Taxa.de.juros")]
)$selection[1] %>% as.numeric() -> p
urca::ca.jo(
  x = df[,c("gZ", "Inflação")],
  ecdet = "const",
  #ecdet = "trend",
  K = p-1,
  spec = "longrun",
  #dumvar = df[,c("Taxa.de.juros")],
  type = "trace"
) %>% summary()
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: trace statistic , without linear trend and constant in cointegration
```

```
##
## Eigenvalues (lambda):
## [1] 2.056295e-01 6.055757e-02 8.326673e-17
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 1 |  7.87  7.52  9.24 12.97
## r = 0  | 36.88 17.85 19.96 24.60
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                     gZ.l4  Inflação.l4   constant
## gZ.l4        1.000000000   1.00000000  1.0000000
## Inflação.l4 -1.287694762 -12.74587081 -2.1282945
## constant    -0.004275919   0.09525539  0.2556235
##
## Weights W:
## (This is the loading matrix)
##
##                   gZ.l4 Inflação.l4      constant
## gZ.d       -0.41691123  0.03275983 -1.071008e-17
## Inflação.d  0.06182621  0.01037229 -3.438307e-19
```

VECM Infla

```
df <- read.csv("./Dados_yeojohnson.csv", encoding="UTF-8")
names(df) <- c("Time", "Infla", "gZ", "TaxaP", "Juros")
df <- na.omit(df[,c("Infla", "gZ", "TaxaP", "Juros")])
df <- ts(data = df, start = c(1992,03), frequency = 4)
model <- tsDyn::VECM(data = df[,c("Infla","gZ")], lag = 5, r = 1, exogen = coredata(df[,"Juros"]))
fevd_gz = data.frame(tsDyn::fevd(model, 20)$gZ)
fevd_tx = data.frame(tsDyn::fevd(model, 20)$TaxaP)
model %>%  summary()
```

```
## #############
## ###Model VECM
## #############
## Full sample size: 110    End sample size: 104
## Number of variables: 2   Number of estimated slope parameters 26
## AIC -1592.069    BIC -1520.67    SSR 0.3107741
## Cointegrating vector (estimated by 2OLS):
##    Infla        gZ
## r1     1 -0.283771
##
##
##                ECT                 Intercept          Infla -1
## Equation Infla -0.0543(0.0238)*    0.0007(0.0031)     0.8325(0.1092)***
## Equation gZ    -0.0385(0.1720)     0.0201(0.0226)     0.1552(0.7896)
##                gZ -1              Infla -2           gZ -2
```

```
## Equation Infla 0.0013(0.0169)      -0.1536(0.1427)      0.0084(0.0142)
## Equation gZ     0.1609(0.1220)      1.9672(1.0323).      -0.1250(0.1027)
##                 Infla -3             gZ -3               Infla -4
## Equation Infla 0.0664(0.1479)       0.0025(0.0142)      -0.2430(0.1442).
## Equation gZ     -1.1045(1.0699)      0.1196(0.1025)      -0.4739(1.0425)
##                 gZ -4                Infla -5            gZ -5
## Equation Infla -0.0109(0.0136)      0.1332(0.1067)      0.0237(0.0150)
## Equation gZ     -0.4693(0.0981)***   0.1339(0.7714)      0.0716(0.1085)
##                 exo_1
## Equation Infla -0.0043(0.0523)
## Equation gZ     -0.3797(0.3784)
```

```r
fevd(model, 20) %>% plot()
```