6104640427_Exam.R

user

2021-06-06

```
setwd("/Users/user/Desktop/Arm/EE435 R")
cat(rep("\n",50))

#install.packages("vars")
library(vars)

## Warning: package 'vars' was built under R version 4.0.5

## Loading required package: MASS

## Loading required package: strucchange

## Warning: package 'strucchange' was built under R version 4.0.5

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

## Loading required package: urca

## Loading required package: lmtest

#library(MTS)
library(quantmod)

## Loading required package: xts

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

library(fGarch)

## Warning: package 'fGarch' was built under R version 4.0.5

## Loading required package: timeDate
```

```
## Loading required package: timeSeries

##
## Attaching package: 'timeSeries'

## The following object is masked from 'package:zoo':
##
##     time<-

## Loading required package: fBasics

##
## Attaching package: 'fBasics'

## The following object is masked from 'package:TTR':
##
##     volatility

library(quantmod)
library(fBasics)
library(sn)

## Loading required package: stats4

##
## Attaching package: 'sn'

## The following object is masked from 'package:fBasics':
##
##     vech

## The following object is masked from 'package:stats':
##
##     sd

library(PerformanceAnalytics)

##
## Attaching package: 'PerformanceAnalytics'

## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness

## The following object is masked from 'package:graphics':
##
##     legend

library(car)

## Loading required package: carData

##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:fBasics':
##
##     densityPlot

library(tseries)
library(forecast)
library(fUnitRoots)

## Warning: package 'fUnitRoots' was built under R version 4.0.5

##
## Attaching package: 'fUnitRoots'

## The following objects are masked from 'package:urca':
##
##     punitroot, qunitroot, unitrootTable

getSymbols("IPDCONGD", src="FRED")

## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.

## [1] "IPDCONGD"

dim(IPDCONGD)

## [1] 892   1

tail(IPDCONGD)

##            IPDCONGD
## 2020-11-01 104.8970
## 2020-12-01 105.3158
## 2021-01-01 107.6347
## 2021-02-01 100.0964
## 2021-03-01 102.5666
## 2021-04-01 100.7979

getSymbols("IPNCONGD", src="FRED")

## [1] "IPNCONGD"

dim(IPNCONGD)

## [1] 892   1

tail(IPNCONGD)
```

```
##            IPNCONGD
## 2020-11-01  96.4210
## 2020-12-01  99.0397
## 2021-01-01  98.8692
## 2021-02-01  97.7272
## 2021-03-01  97.9114
## 2021-04-01  99.0269
```

```
getSymbols("IPBUSEQ", src="FRED")
```

```
## [1] "IPBUSEQ"
```

```
dim(IPBUSEQ)
```

```
## [1] 892   1
```

```
tail(IPBUSEQ)
```

```
##            IPBUSEQ
## 2020-11-01 91.1940
## 2020-12-01 91.5631
## 2021-01-01 93.3570
## 2021-02-01 91.4516
## 2021-03-01 94.1449
## 2021-04-01 94.1154
```

```
getSymbols("IPMAT", src="FRED")
```

```
## [1] "IPMAT"
```

```
dim(IPMAT)
```

```
## [1] 988   1
```

```
tail(IPMAT)
```

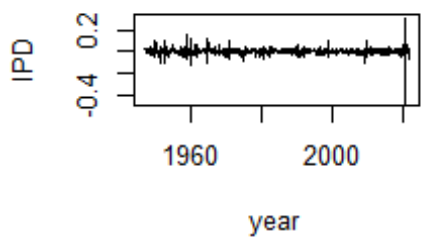
```
##              IPMAT
## 2020-11-01 97.7184
## 2020-12-01 98.5967
## 2021-01-01 99.7727
## 2021-02-01 95.5421
## 2021-03-01 98.5340
## 2021-04-01 99.4635
```

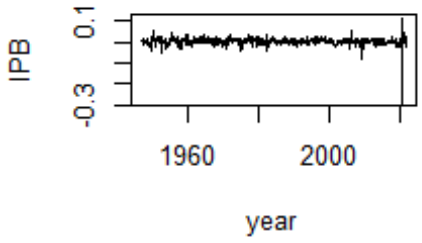
```
IP =
cbind(as.numeric(IPDCONGD),as.numeric(IPNCONGD),as.numeric(IPBUSEQ),as.numeri
c(IPMAT[1:96]))
```

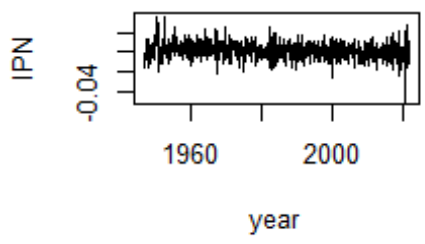
```
## Warning in cbind(as.numeric(IPDCONGD), as.numeric(IPNCONGD),
## as.numeric(IPBUSEQ), : number of rows of result is not a multiple of
vector
## length (arg 4)
```

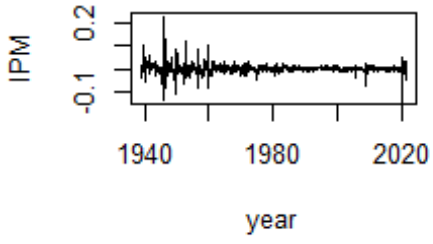
```
dim(IP)

## [1] 892   4

colnames(IP) <- c("IPD","IPN","IPB","IPM")


IP = log(IP)
zt = diff(IP)*100

#library(MTS)
#VARorder(IP)

library(vars)
tdx = c(1:892)/12+1947
tdx1 = c(1:988)/12+1939

par (mfcol=c(2,2))
plot(tdx ,diff(log(IPDCONGD)), xlab='year',ylab= 'IPD',type = 'l')

plot(tdx ,diff(log(IPNCONGD)), xlab='year',ylab= 'IPN',type = 'l')

plot(tdx ,diff(log(IPBUSEQ)), xlab='year',ylab= 'IPB',type = 'l')

plot(tdx1 ,diff(log(IPMAT)), xlab='year',ylab= 'IPM',type = 'l')
```

1.1) All indicators was hit hard by covid pandemic, dumped the production down.









we can see that all of the monthly industrial production index are fluctuate around mean of 0.

- The durable goods will be less volatile due to the needs of econ.
- Non-durable goods are always volatile as it is not always essential for living. The high fluctuation reflect ppl. adjustment overtime.
- business equivalent also in the mean clustering.
- IPM for materials, is much volatile in the time of great depression, where it link to inflation.

```
#////////////////////////////////////////////////////////////////#

varfit=VAR(zt,p=2)
summary(varfit)

##
## VAR Estimation Results:
## =========================
## Endogenous variables: IPD, IPN, IPB, IPM
## Deterministic variables: const
## Sample size: 889
## Log Likelihood: -7810.545
## Roots of the characteristic polynomial:
## 0.6124 0.4336 0.4063 0.4063 0.2519 0.2519 0.185 0.185
## Call:
## VAR(y = zt, p = 2)
##
##
## Estimation results for equation IPD:
## ====================================
## IPD = IPD.l1 + IPN.l1 + IPB.l1 + IPM.l1 + IPD.l2 + IPN.l2 + IPB.l2 +
IPM.l2 + const
##
##          Estimate Std. Error t value Pr(>|t|)
## IPD.l1  0.1592888  0.0469596   3.392 0.000725 ***✓
## IPN.l1  0.2620278  0.1477685   1.773 0.076536 .
## IPB.l1 -0.0010868  0.0909152  -0.012 0.990465
## IPM.l1  0.0017152  0.0129585   0.132 0.894727
## IPD.l2 -0.1381939  0.0468288  -2.951 0.003251 **✓
## IPN.l2  0.2452490  0.1473544   1.664 0.096399 .
## IPB.l2 -0.1565839  0.0908259  -1.724 0.085059 .
## IPM.l2 -0.0007519  0.0129137  -0.058 0.953584
## const   0.1982092  0.1144743   1.731 0.083718 .✓
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 3.219 on 880 degrees of freedom
## Multiple R-Squared: 0.06191, Adjusted R-squared: 0.05338
## F-statistic:  7.26 on 8 and 880 DF,  p-value: 2.322e-09
##
##
## Estimation results for equation IPN:
## ====================================
## IPN = IPD.l1 + IPN.l1 + IPB.l1 + IPM.l1 + IPD.l2 + IPN.l2 + IPB.l2 +
IPM.l2 + const
##
##          Estimate Std. Error t value Pr(>|t|)
```

1.2

$\widehat{IPD}_t = 0.198 + 0.159\ IPD_{t-1} + 0.138\ IPD_{t-2}$
(0.114) (0.047) (0.047)

```
## IPD.l1  3.161e-02  1.148e-02   2.754  0.00601 **
## IPN.l1 -1.654e-01  3.612e-02  -4.578 5.36e-06 ***
## IPB.l1  1.322e-02  2.223e-02   0.595  0.55209
## IPM.l1  1.164e-03  3.168e-03   0.367  0.71349
## IPD.l2  7.699e-04  1.145e-02   0.067  0.94639
## IPN.l2 -4.182e-02  3.602e-02  -1.161  0.24603
## IPB.l2 -8.796e-06  2.220e-02   0.000  0.99968
## IPM.l2 -2.073e-03  3.157e-03  -0.657  0.51154
## const   1.819e-01  2.798e-02   6.501 1.34e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.787 on 880 degrees of freedom
## Multiple R-Squared: 0.03292, Adjusted R-squared: 0.02413
## F-statistic: 3.744 on 8 and 880 DF,  p-value: 0.0002532
##
##
## Estimation results for equation IPB:
## ====================================
## IPB = IPD.l1 + IPN.l1 + IPB.l1 + IPM.l1 + IPD.l2 + IPN.l2 + IPB.l2 +
IPM.l2 + const
##
##         Estimate Std. Error t value Pr(>|t|)
## IPD.l1  0.025337   0.024370   1.040  0.29877
## IPN.l1  0.073587   0.076684   0.960  0.33752
## IPB.l1  0.185487   0.047180   3.931 9.11e-05 ***
## IPM.l1 -0.002311   0.006725  -0.344  0.73114
## IPD.l2 -0.126909   0.024302  -5.222 2.21e-07 ***
## IPN.l2  0.058110   0.076469   0.760  0.44751
## IPB.l2  0.217536   0.047134   4.615 4.51e-06 ***
## IPM.l2  0.003214   0.006702   0.480  0.63164
## const   0.192802   0.059406   3.245  0.00122 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 1.671 on 880 degrees of freedom
## Multiple R-Squared: 0.09136, Adjusted R-squared: 0.0831
## F-statistic: 11.06 on 8 and 880 DF,  p-value: 5.781e-15
##
##
## Estimation results for equation IPM:
## ====================================
## IPM = IPD.l1 + IPN.l1 + IPB.l1 + IPM.l1 + IPD.l2 + IPN.l2 + IPB.l2 +
IPM.l2 + const
##
##         Estimate Std. Error t value Pr(>|t|)
## IPD.l1 -0.253340   0.122193  -2.073   0.0384 *
## IPN.l1 -0.448470   0.384506  -1.166   0.2438
## IPB.l1  0.486339   0.236569   2.056   0.0401 *
## IPM.l1  0.004424   0.033719   0.131   0.8956
## IPD.l2 -0.107171   0.121852  -0.880   0.3794
## IPN.l2  0.064195   0.383428   0.167   0.8671
## IPB.l2  0.038803   0.236337   0.164   0.8696
## IPM.l2 -0.048159   0.033602  -1.433   0.1522
## const   0.033270   0.297872   0.112   0.9111
## ---
```

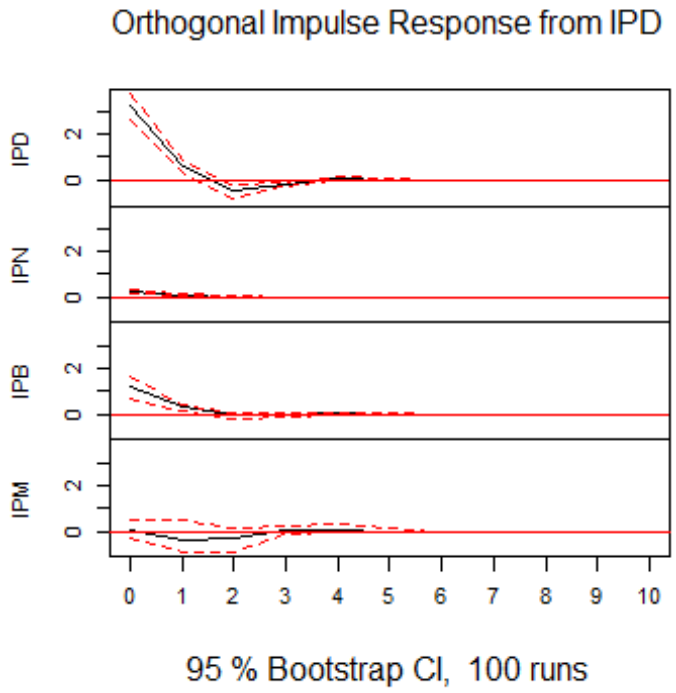
$\widehat{IPN}_t = 0.182 + 0.032\ IPD_{t-1} - 0.165\ IPN_{t-1}$
(0.028) (0.011) (0.036)

$\widehat{IPB}_t = 0.193 + 0.185\ IPB_{t-1} - 0.127\ IPD_{t-2} + 0.218\ IPB_{t-2}$
(0.059) (0.047) (0.024) (0.047)

$\widehat{IPM}_t = 0.033 - 0.253\ IPD_{t-1} + 0.486\ IPB_{t-1}$
(0.298) (0.122) (0.237)

```
## IPB.l1  0.486339   0.236569   2.056   0.0401 *
## IPM.l1  0.004424   0.033719   0.131   0.8956
## IPD.l2 -0.107171   0.121852  -0.880   0.3794
## IPN.l2  0.064195   0.383428   0.167   0.8671
## IPB.l2  0.038803   0.236337   0.164   0.8696
## IPM.l2 -0.048159   0.033602  -1.433   0.1522
## const   0.033270   0.297872   0.112   0.9111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 8.377 on 880 degrees of freedom
## Multiple R-Squared: 0.01197, Adjusted R-squared: 0.002991
## F-statistic: 1.333 on 8 and 880 DF,  p-value: 0.2232
##
##
##
## Covariance matrix of residuals:
##         IPD    IPN    IPB     IPM
## IPD 10.3649 0.8522 3.8672  0.3240
## IPN  0.8522 0.6194 0.4514  0.1593
## IPB  3.8672 0.4514 2.7913  0.7414
## IPM  0.3240 0.1593 0.7414 70.1789
##
## Correlation matrix of residuals:
##         IPD     IPN     IPB     IPM
## IPD 1.00000 0.33631 0.71897 0.01201
## IPN 0.33631 1.00000 0.34329 0.02415
## IPB 0.71897 0.34329 1.00000 0.05297
## IPM 0.01201 0.02415 0.05297 1.00000

varfit.prd <- predict(varfit, n.ahead = 15, ci = 0.90)


impresp =irf(varfit)
plot(impresp)
```

2.3)

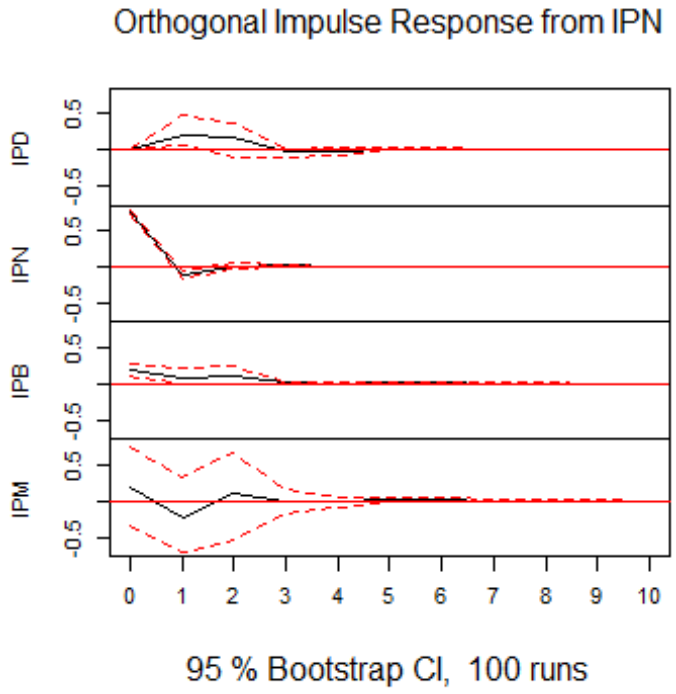


IPD (Durable goods)

The negative shock, on durable goods index create a ripple effects to the business equivalent the most where it goes the same sides

The durable goody index not likely to have the significant effect on the materials and non durable.

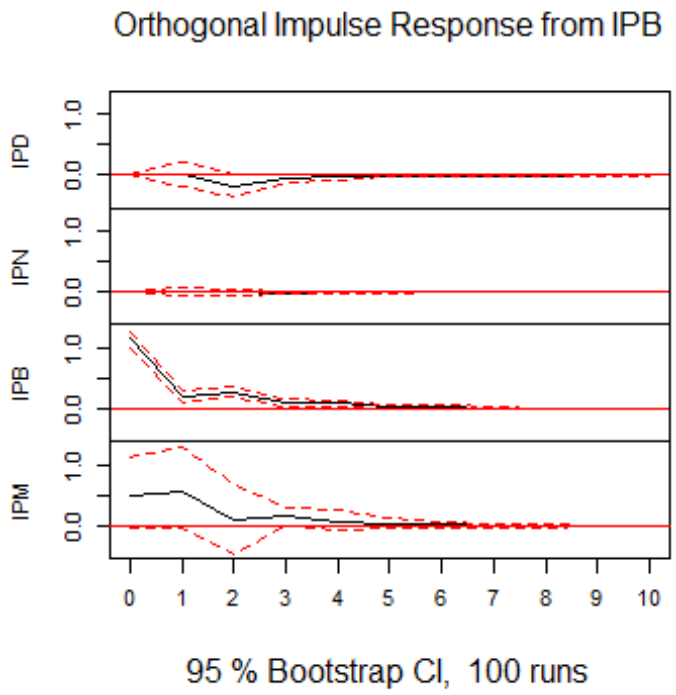
It is intuitively likely that business equivalent is the manufacturing for durable goods.



Shock or

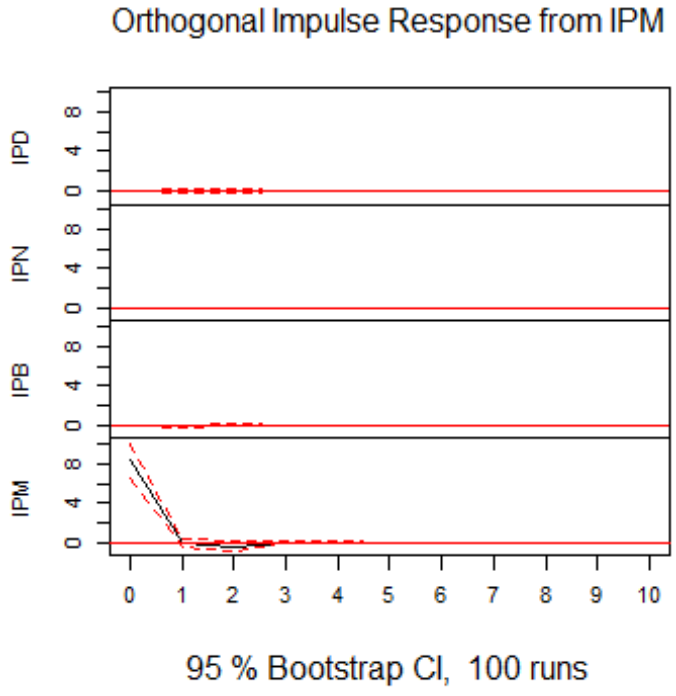
IPN create more impulse effect on the others

When the first hit of any shock to the economy, ppl. adjust their spending, start from the consumption of non-durable goods (as followed business cycles) the negative shock on IPN likely to create ⊖ shock to other economic units, however, the durable goods is quite stable with the shock.



IPB - business equivalent

create most impact to the materials, it is the first upstream. When there is business shock (-), the manufact production would be decrease, the lag of 1 period might be from the futures order



Shock on.
(Materials (likely to cause inflation)
this not affecting any of the other variables, it is the case that the price & cost & wages might adjust well enough for the shock in material price).

1.4)

```
fevd(varfit, start = 879 , n.ahead = 6)
```

```
## $IPD
##            IPD         IPN          IPB          IPM
## [1,] 1.0000000 0.000000000 0.000000e+00 0.000000e+00
## [2,] 0.9964651 0.003515788 1.254705e-08 1.913970e-05
## [3,] 0.9916473 0.005543804 2.790036e-03 1.885246e-05
## [4,] 0.9911782 0.005720795 3.081669e-03 1.933317e-05
## [5,] 0.9909328 0.005873537 3.169082e-03 2.457529e-05
## [6,] 0.9909176 0.005873036 3.184744e-03 2.457831e-05
##
## $IPN
##            IPD       IPN          IPB          IPM
## [1,] 0.1131069 0.8868931 0.0000000000 0.0000000000
## [2,] 0.1180915 0.8813721 0.0003884815 0.0001478934
## [3,] 0.1180252 0.8808924 0.0003883480 0.0006940446
## [4,] 0.1184195 0.8804724 0.0004055296 0.0007025819
## [5,] 0.1184288 0.8804609 0.0004055529 0.0007047158
## [6,] 0.1184388 0.8804505 0.0004056304 0.0007051104
##
## $IPB
##            IPD        IPN       IPB          IPM
## [1,] 0.5169170 0.01161371 0.4714693 0.0000000000
## [2,] 0.5248171 0.01359103 0.4614653 0.0001265971
## [3,] 0.5090126 0.01616755 0.4745038 0.0003160843
## [4,] 0.5074718 0.01613368 0.4760789 0.0003155765
## [5,] 0.5059643 0.01607454 0.4776441 0.0003170436
## [6,] 0.5056978 0.01607784 0.4779075 0.0003168628
##
## $IPM
##              IPD          IPN         IPB       IPM
## [1,] 0.000144345 0.0004561856 0.003747381 0.9956521
## [2,] 0.001873442 0.0012956491 0.008161195 0.9886697
## [3,] 0.003226897 0.0014095731 0.008324195 0.9870393
## [4,] 0.003278165 0.0014090595 0.008720230 0.9865925
## [5,] 0.003439563 0.0014094484 0.008828873 0.9863221
## [6,] 0.003448054 0.0014147158 0.008878252 0.9862590
```

for IPD & IPN, the shock from itself will be highest degree of determining the contribution in explaining itself.

⇒ the IPD have the contribution of shock from mostly IPD then IPB itself

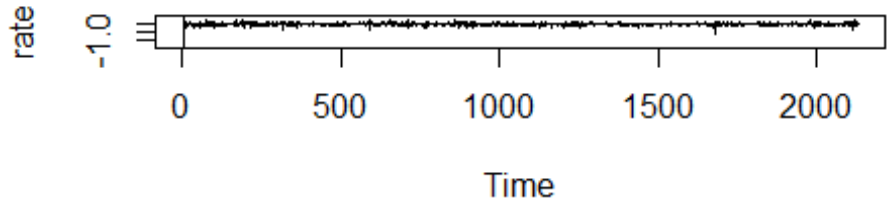
⇒ IPN is more likely determining itself from the past shock to future shock.

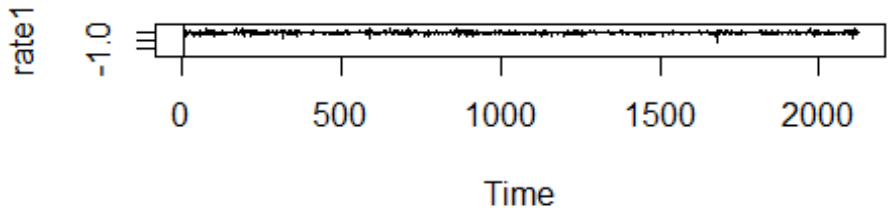
```
#////////////////////////////////////////////////////////////////////#
```

(2)

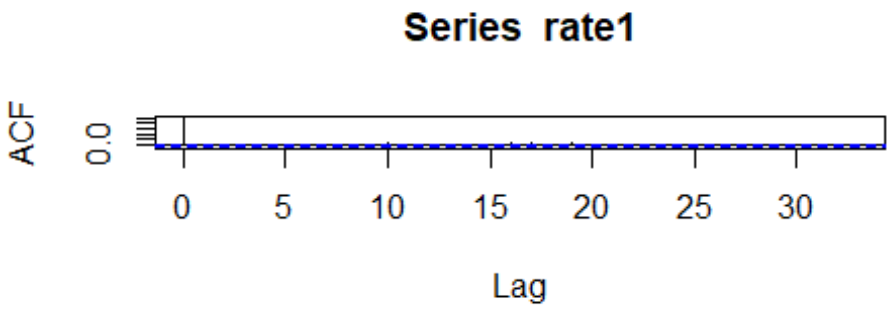
```
getSymbols('ETH-USD',src='yahoo')

## Warning: ETH-USD contains missing values. Some functions will not work if
## objects contain missing values in the middle of the series. Consider using
## na.omit(), na.approx(), na.fill(), etc to remove or replace them.

## [1] "ETH-USD"

par (mfcol=c(2,1))
rate = (`ETH-USD`[,6])
rate = diff(log(as.numeric(rate)))
ts.plot(rate)
rate1 = na.omit(rate)
ts.plot(rate1)
```

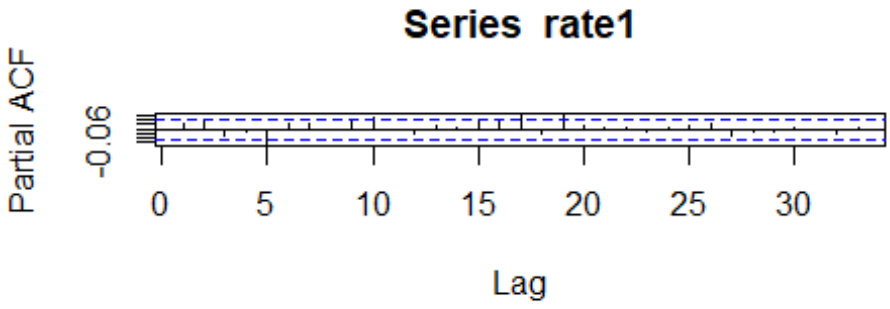
```
> acf(rate1^2)
> Box.test(rate^2, lag=10,type='Ljung')

        Box-Ljung test

data:  rate^2
X-squared = 43.307, df = 10, p-value = 4.382e-06
```






```
acf(rate1)
pacf(rate1)
```






```
auto.arima(rate1)

## Series: rate1
## ARIMA(5,0,5) with non-zero mean
##
## Coefficients:
##           ar1      ar2     ar3      ar4      ar5     ma1     ma2      ma3
##       -0.1803  -0.3006  0.0280  -0.4563  -0.7101  0.2024  0.3518  -0.0498
## s.e.   0.1176   0.0785  0.0954   0.0837   0.1169  0.1254  0.0814   0.1019
##          ma4     ma5    mean
##       0.4482  0.6228  0.0032
## s.e.  0.0849  0.1250  0.0014
##
## sigma^2 estimated as 0.004607:  log likelihood=2703.95
## AIC=-5383.91   AICc=-5383.76   BIC=-5315.98

m1 <- arima(rate1,order = c(5,0,5))

## Warning in arima(rate1, order = c(5, 0, 5)): possible convergence problem:
optim
## gave code = 1

acf(m1$residuals^2)
Box.test(m1$residuals^2,lag=10,type = 'Ljung')

##
##  Box-Ljung test
```

```
##
## data:  m1$residuals^2
## X-squared = 14.866, df = 10, p-value = 0.137
```

```
summary(m1)
```

```
##
## Call:
## arima(x = rate1, order = c(5, 0, 5))
##
## Coefficients:
##           ar1      ar2     ar3      ar4      ar5     ma1     ma2      ma3
##       -0.1803  -0.3006  0.0280  -0.4563  -0.7101  0.2024  0.3518  -0.0498
## s.e.   0.1176   0.0785  0.0954   0.0837   0.1169  0.1254  0.0814   0.1019
##          ma4     ma5  intercept
##       0.4482  0.6228     0.0032
## s.e.  0.0849  0.1250     0.0014
##
## sigma^2 estimated as 0.004583:  log likelihood = 2703.95,  aic = -5383.91
##
## Training set error measures:
##                         ME       RMSE       MAE      MPE     MAPE      MASE
## Training set 1.060009e-05 0.06769734 0.0421659 80.44417 201.5599 0.6935251
##                      ACF1
## Training set -0.002804635
```

```
predict(m1,6)
```

```
## $pred
## Time Series:
## Start = 2124
## End = 2129
## Frequency = 1
## [1] -0.015437331  0.001159058  0.031950677  0.019867885  0.007028348
## [6]  0.012371424
##
## $se
## Time Series:
## Start = 2124
## End = 2129
## Frequency = 1
## [1] 0.06769734 0.06771382 0.06778944 0.06783556 0.06784325 0.06807148
```

```
m2 <- garchFit(~garch(1,1), data=rate1, trace = FALSE)
```

```
## Warning: Using formula(x) is deprecated when x is a character vector of
length > 1.
##   Consider formula(paste(x, collapse = " ")) instead.
```

```
summary(m2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(1, 1), data = rate1, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(1, 1)
## <environment: 0x000000001c6e6bf8>
##  [data = rate1]
##
## Conditional Distribution:
##  norm
##
```

```
## Coefficient(s):
##         mu       omega      alpha1       beta1
## 0.00182168  0.00031083  0.22444426  0.72912838
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     0.0018217   0.0011439    1.592    0.111
## omega  0.0003108   0.0000524    5.931 3.00e-09 ***
## alpha1 0.2244443   0.0327325    6.857 7.04e-12 ***
## beta1  0.7291284   0.0303551   24.020  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  2937.329    normalized:  1.383574
##
## Description:
##  Sun Jun 06 14:29:06 2021 by user: user
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  236591.4  0
##  Shapiro-Wilk Test  R    W      0.8442512 0
##  Ljung-Box Test     R    Q(10)  18.0503   0.05412109
##  Ljung-Box Test     R    Q(15)  22.24218  0.1016091
##  Ljung-Box Test     R    Q(20)  37.33219  0.01067344
##  Ljung-Box Test     R^2  Q(10)  0.1903174 0.9999999
##  Ljung-Box Test     R^2  Q(15)  0.2903802 1
##  Ljung-Box Test     R^2  Q(20)  0.3754667 1
##  LM Arch Test       R    TR^2   8.732661  0.7255792
##
## Information Criterion Statistics:
##       AIC       BIC       SIC      HQIC
## -2.763381 -2.752715 -2.763388 -2.759476
```

```
predict(m2,6)
```

```
##   meanForecast  meanError standardDeviation
## 1  0.001821681 0.05765823        0.05765823
## 2  0.001821681 0.05899958        0.05899958
## 3  0.001821681 0.06025085        0.06025085
## 4  0.001821681 0.06142028        0.06142028
## 5  0.001821681 0.06251505        0.06251505
## 6  0.001821681 0.06354142        0.06354142
```

```
m3 <- garchFit(~arma(5,5)+garch(1,1), data=rate1, trace = FALSE)
```

```
## Warning in arima(.series$x, order = c(u, 0, v), include.mean =
include.mean):
## possible convergence problem: optim gave code = 1
```

```
## Warning: Using formula(x) is deprecated when x is a character vector of
length > 1.
##   Consider formula(paste(x, collapse = " ")) instead.
```

```
summary(m3)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~arma(5, 5) + garch(1, 1), data = rate1, trace =
FALSE)
##
## Mean and Variance Equation:
##  data ~ arma(5, 5) + garch(1, 1)
## <environment: 0x000000001e16b8d0>
##  [data = rate1]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##          mu          ar1          ar2          ar3          ar4
ar5
##  0.00440964  -0.23424935  -0.07998447   0.11238981  -0.37851164  -
0.34767937
##         ma1          ma2          ma3          ma4          ma5
omega
##  0.25614780   0.11708857  -0.12106321   0.41241581   0.36292308
```

```
0.00028252
##      alpha1        beta1
##  0.18481453   0.75315762
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu      4.410e-03   2.212e-03    1.994 0.046161 *
## ar1    -2.342e-01   6.368e-02   -3.678 0.000235 ***
## ar2    -7.998e-02   1.146e-01   -0.698 0.485306
## ar3     1.124e-01   6.677e-02    1.683 0.092337 .
## ar4    -3.785e-01   6.493e-02   -5.830 5.55e-09 ***
## ar5    -3.477e-01   6.104e-02   -5.696 1.23e-08 ***
## ma1     2.561e-01   6.920e-02    3.702 0.000214 ***
## ma2     1.171e-01   1.101e-01    1.063 0.287652
## ma3    -1.211e-01   6.897e-02   -1.755 0.079220 .
## ma4     4.124e-01   5.678e-02    7.263 3.78e-13 ***
## ma5     3.629e-01   5.586e-02    6.497 8.21e-11 ***
## omega   2.825e-04   5.128e-05    5.510 3.60e-08 ***
## alpha1  1.848e-01   2.466e-02    7.495 6.62e-14 ***
## beta1   7.532e-01   2.974e-02   25.328  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  3144.441    normalized:  1.481131
##
## Description:
##  Sun Jun 06 14:29:09 2021 by user: user
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  2819.519  0
##  Shapiro-Wilk Test  R    W      0.941982  0
##  Ljung-Box Test     R    Q(10)  20.78912  0.02261297
##  Ljung-Box Test     R    Q(15)  27.4493   0.02528179
##  Ljung-Box Test     R    Q(20)  42.46562  0.002403851
##  Ljung-Box Test     R^2  Q(10)  8.994997  0.5325784
##  Ljung-Box Test     R^2  Q(15)  11.43751  0.7210084
##  Ljung-Box Test     R^2  Q(20)  14.91342  0.7813399
##  LM Arch Test       R    TR^2   9.051904  0.6984909
##
## Information Criterion Statistics:
##       AIC       BIC       SIC      HQIC
## -2.949073 -2.911744 -2.949159 -2.935408

#////////////////////////////////////////////////////////#
```

Study to propose the model for ETH return :

1) We start with pulling the data from yahoo and omit the n.a. variable a/f. turn to the log return. (+Basicstats)

2) We now check for ARCH effect in the series for picking model. (Pv < 0.05 then) now we have ARCH in the series.

3) The model to be proposed will selected between the GARCH(1,1) and ARMA(5,5) + GARCH(1,1) from AIC & BIC criteria

4) As we use the BIC, now the M3 model is more powerful predicting tools

The fitted model is mean eq. : $\hat{r}_t = \underset{(0.002)}{0.004} - \underset{(0.064)}{0.234}\, r_{t-1} - \underset{(0.065)}{0.379} r_{t-4} - \underset{(0.061)}{0.348}\, r_{t-5}$

$$+ \underset{(0.069)}{0.256}\, a_{t-1} + \underset{(0.057)}{0.412}\, a_{t-4} + \underset{(0.056)}{0.363}\, a_{t-5}$$

var equation = $\underset{(0.000)}{0.0003} + \underset{(0.025)}{0.185}\, a_{t-1}^2 + \underset{(0.029)}{0.753}\, \sigma_{t-1}^2$

5) • Now we test the rest of ARCH effect, whether the ARCH effect is left in the model or not. From the $\tilde{a}_t$ of residuals-square, we can not reciject H_0 as pv > 0.05, then the leftover is gone for the rest of residual (not square in mean equation) there exist the problem of ARCH effect leftover

• The normality test pv. also < 0.05 then. The model is not normally distributed.

6) The prediction and variance will be provide in R script.

(forecasting interval ⇒ mean ± 1.96 (S.D))

```r
getSymbols("CLVMNACSCAB1GQUK", src="FRED")
```

```
## [1] "CLVMNACSCAB1GQUK"
```

```r
dim(CLVMNACSCAB1GQUK)
```

```
## [1] 183   1
```

```r
tail(CLVMNACSCAB1GQUK)
```

```
##            CLVMNACSCAB1GQUK
## 2019-04-01         470665.5
## 2019-07-01         473006.0
## 2019-10-01         473087.6
## 2020-01-01         458804.7
## 2020-04-01         372766.5
## 2020-07-01         432422.1
```

```r
getSymbols("NAEXKP02CAQ189S", src="FRED")
```

```
## [1] "NAEXKP02CAQ189S"
```

```r
dim(NAEXKP02CAQ189S)
```

```
## [1] 160   1
```

```r
tail(NAEXKP02CAQ189S)
```

```
##            NAEXKP02CAQ189S
## 2019-07-01    304775250000
## 2019-10-01    306277750000
## 2020-01-01    300805500000
## 2020-04-01    258382750000
## 2020-07-01    291737500000
## 2020-10-01    291772000000
```

```r
getSymbols("GDPC1", src="FRED")
```

```
## [1] "GDPC1"
```

```r
dim(GDPC1)
```

```
## [1] 297   1
```

```r
tail(GDPC1)
```

```
##               GDPC1
## 2019-10-01 19253.96
## 2020-01-01 19010.85
## 2020-04-01 17302.51
## 2020-07-01 18596.52
```

```
## 2020-10-01 18794.43
## 2021-01-01 19088.06

z =
cbind(as.numeric(CLVMNACSCAB1GQUK[1:146]),as.numeric(NAEXKP02CAQ189S[1:146]),
as.numeric(GDPC1[1:146]))

dim(z)

## [1] 146   3

colnames(z) <- c("UK","CA","US")

zt = log(z)
zt = diff(zt) * 100


#library(MTS)
#VARorder(zt)

varfit=VAR(zt,p=4)
summary(varfit)

##
## VAR Estimation Results:
## =========================
## Endogenous variables: UK, CA, US
## Deterministic variables: const
## Sample size: 141
## Log Likelihood: -476.777
## Roots of the characteristic polynomial:
## 0.776 0.776 0.774 0.774 0.6641 0.6641 0.6157 0.5831 0.5831 0.2969 0.2969
0.1111
## Call:
## VAR(y = zt, p = 4)
##
##
## Estimation results for equation UK:
## ===================================
## UK = UK.l1 + CA.l1 + US.l1 + UK.l2 + CA.l2 + US.l2 + UK.l3 + CA.l3 + US.l3
+ UK.l4 + CA.l4 + US.l4 + const
##
##       Estimate Std. Error t value Pr(>|t|)
## UK.l1  0.11467    0.08614   1.331  0.18549
## CA.l1 -0.14947    0.11496  -1.300  0.19585
## US.l1  0.02942    0.06196   0.475  0.63578
## UK.l2  0.28607    0.08598   3.327  0.00115 **
## CA.l2 -0.02873    0.11273  -0.255  0.79925
## US.l2  0.10282    0.06369   1.615  0.10887
## UK.l3  0.18134    0.08580   2.113  0.03651 *
```

3.1)

$$\widehat{UK}_t = \underset{(0.156)}{0.291} + \underset{(0.086)}{0.286}\, UK_{t-2} + \underset{(0.086)}{0.181}\, UK_{t-3}$$

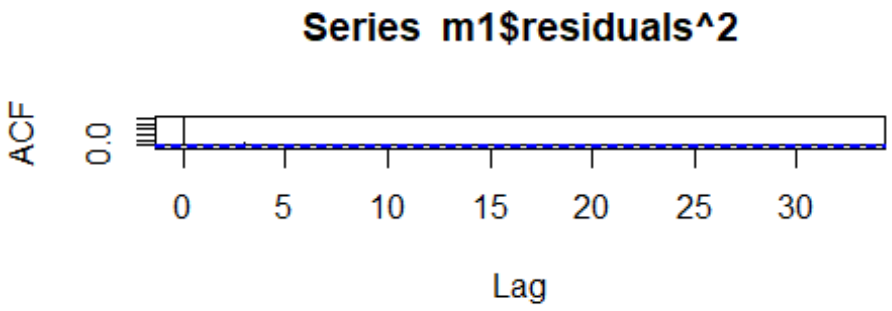
```
## CA.l3 -0.00527    0.11030  -0.048  0.96197
## US.l3  0.01480    0.06387   0.232  0.81715
## UK.l4 -0.09810    0.08275  -1.185  0.23802
## CA.l4  0.12120    0.11255   1.077  0.28355
## US.l4 -0.10053    0.06249  -1.609  0.11016
## const  0.29054    0.15626   1.859  0.06528 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.7843 on 128 degrees of freedom
## Multiple R-Squared: 0.2098,  Adjusted R-squared: 0.1358
## F-statistic: 2.833 on 12 and 128 DF,  p-value: 0.001779
##
##
## Estimation results for equation CA:
## ===================================
## CA = UK.l1 + CA.l1 + US.l1 + UK.l2 + CA.l2 + US.l2 + UK.l3 + CA.l3 + US.l3
+ UK.l4 + CA.l4 + US.l4 + const
##
##       Estimate Std. Error t value Pr(>|t|)
## UK.l1 -0.02850    0.06245  -0.456 0.648861
## CA.l1  0.13664    0.08335   1.639 0.103581
## US.l1  0.02213    0.04492   0.493 0.623108
## UK.l2  0.07197    0.06234   1.155 0.250438
## CA.l2  0.17731    0.08173   2.169 0.031905 *
## US.l2 -0.01578    0.04617  -0.342 0.733170
## UK.l3 -0.02827    0.06221  -0.454 0.650339
## CA.l3  0.22757    0.07997   2.846 0.005162 **
## US.l3  0.06305    0.04631   1.362 0.175723
## UK.l4 -0.03952    0.06000  -0.659 0.511226
## CA.l4 -0.24465    0.08160  -2.998 0.003263 **
## US.l4  0.04169    0.04531   0.920 0.359178
## const  0.41917    0.11329   3.700 0.000319 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.5686 on 128 degrees of freedom
## Multiple R-Squared: 0.2018,  Adjusted R-squared: 0.127
## F-statistic: 2.697 on 12 and 128 DF,  p-value: 0.002838
##
##
## Estimation results for equation US:
## ===================================
## US = UK.l1 + CA.l1 + US.l1 + UK.l2 + CA.l2 + US.l2 + UK.l3 + CA.l3 + US.l3
+ UK.l4 + CA.l4 + US.l4 + const
##
##         Estimate Std. Error t value Pr(>|t|)
## UK.l1  1.165e-01  1.200e-01   0.971 0.333546
```

$$\widehat{CA}_t = \underset{(0.113)}{0.419} + \underset{(0.082)}{0.177}\, CA_{t-2} + \underset{(0.079)}{0.228}\, CA_{t-3} - \underset{(0.062)}{0.245}\, CA_{t-4}$$

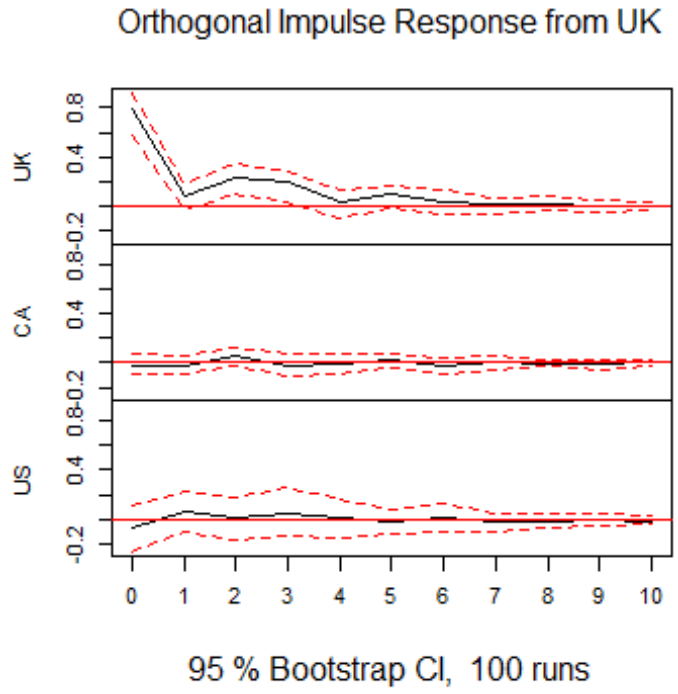
```
## CA.l1  2.241e-01  1.601e-01   1.399 0.164101
## US.l1  2.982e-01  8.631e-02   3.456 0.000746 ***
## UK.l2 -4.423e-05  1.198e-01   0.000 0.999706
## CA.l2  1.522e-01  1.570e-01   0.969 0.334254
## US.l2  1.095e-01  8.871e-02   1.234 0.219438
## UK.l3 -6.677e-03  1.195e-01  -0.056 0.955535
## CA.l3 -1.430e-01  1.536e-01  -0.931 0.353680
## US.l3 -1.440e-01  8.897e-02  -1.618 0.108038
## UK.l4 -3.404e-02  1.153e-01  -0.295 0.768236
## CA.l4  3.452e-01  1.568e-01   2.202 0.029445 *
## US.l4 -1.311e-01  8.705e-02  -1.506 0.134466
## const  3.148e-01  2.177e-01   1.446 0.150557
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 1.092 on 128 degrees of freedom
## Multiple R-Squared: 0.2139,  Adjusted R-squared: 0.1402
## F-statistic: 2.902 on 12 and 128 DF,  p-value: 0.001397
##
##
##
## Covariance matrix of residuals:
##          UK       CA       US
## UK  0.61509 -0.01565 -0.05452
## CA -0.01565  0.32332  0.08217
## US -0.05452  0.08217  1.19352
##
## Correlation matrix of residuals:
##          UK      CA       US
## UK  1.00000 -0.0351 -0.06363
## CA -0.03510  1.0000  0.13228
## US -0.06363  0.1323  1.00000
```

$$\widehat{US}_t = \underset{(0.218)}{0.315} + \underset{(0.086)}{0.298}\, US_{t-1} + \underset{(0.087)}{0.345}\, CA_{t-4}$$

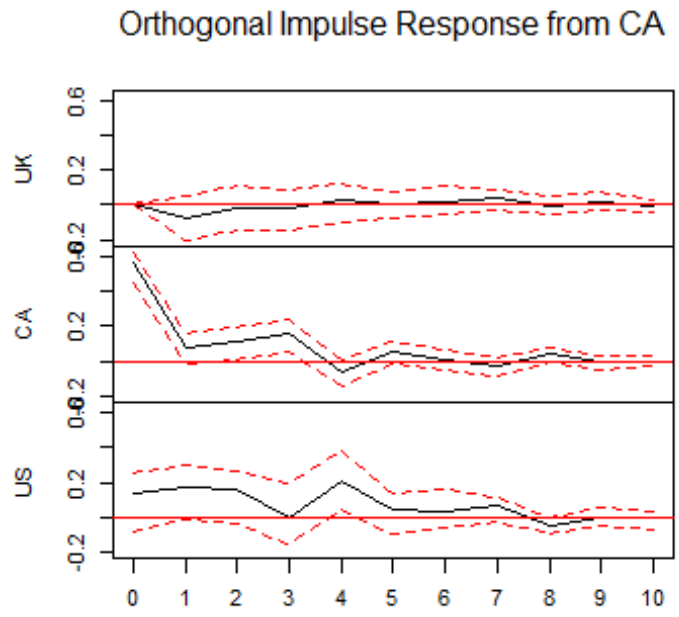
```
varfit.prd <- predict(varfit, n.ahead = 15, ci = 0.90)


impresp =irf(varfit)
plot(impresp)
```



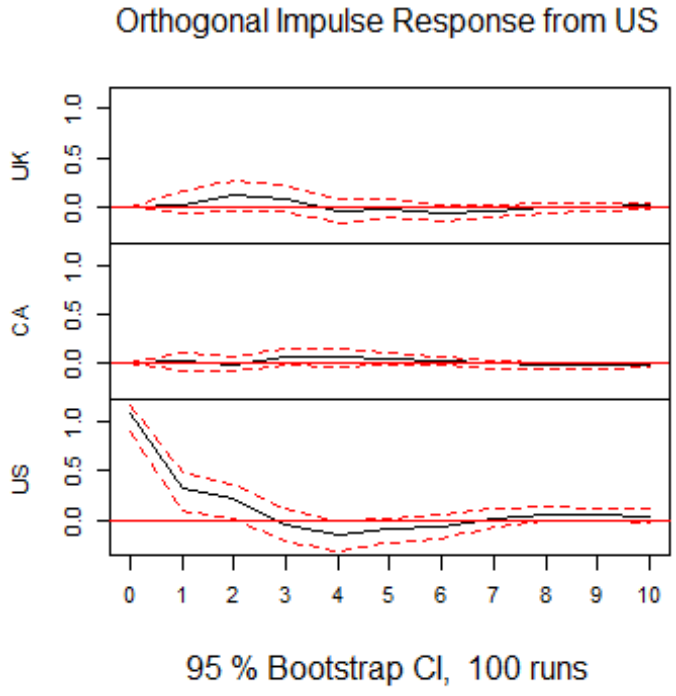
3.2



The shock to UK GDP will create very little shock to other 2 countries' GDP then we implies that UK is the quite mature country, the shock to UK solely will not create much ripple to other Developed country.



Shock to Canada GDP will create a ripple effect to the UK and US in small magnitude, as CA is also a developed country where its economy work closely with America, it also make the shock to spread, likely to be reverse trend with the US.



The shock in US will not likely to create the shock the other GDP.

Intuitively Us. is the big buyer of product from other countries, EM market namely.

However, other 2 developed countries tend to be the sames. So the shock in one country, like US, will not likely to create the shock to other developed chy.

```
fevd(varfit, n.ahead = 6)

## $UK
##              UK          CA          US
## [1,] 1.0000000 0.000000000 0.000000000
## [2,] 0.9880596 0.010337894 0.001602546
## [3,] 0.9679870 0.009774461 0.022238533
```

```
## [4,] 0.9607749 0.009926559 0.029298515
## [5,] 0.9578106 0.010576783 0.031612620
## [6,] 0.9580616 0.010500346 0.031438044
##
## $CA
##               UK        CA          US
## [1,] 0.001231813 0.9987682 0.000000000
## [2,] 0.003342759 0.9949290 0.001728261
## [3,] 0.010177347 0.9880088 0.001813843
## [4,] 0.011664672 0.9737453 0.014590033
## [5,] 0.012052980 0.9572492 0.030697853
## [6,] 0.012543535 0.9524844 0.034972061
##
## $US
##               UK         CA        US
## [1,] 0.004048800 0.01693180 0.9790194
## [2,] 0.006918754 0.03684019 0.9562411
## [3,] 0.006676604 0.05335640 0.9399670
## [4,] 0.008661201 0.05319655 0.9381422
## [5,] 0.008325875 0.08100256 0.9106716
## [6,] 0.008724282 0.08177757 0.9094981
```

```
zt <- as.data.frame(zt)
```

3.3

To check cointegration we need to check each variable(s) whether it is I(0) or I(1)

as we get the diff (log (data)) already then they are already in the I(1)

```
a = zt$UK
b = zt$CA
c = zt$US
```

```
m1 = adfTest(a , lag = 3, type = c("ct"),title = NULL ,description = NULL)
```

```
## Warning in adfTest(a, lag = 3, type = c("ct"), title = NULL, description =
## NULL): p-value smaller than printed p-value
```

```
m1@test$p.value
```

we no need to take any more manipulation to the data.

Δa, Δb, Δc $\sim$ I(1)

```
##
## 0.01
```

```
m2 = adfTest(b , lag = 3, type = c("ct"),title = NULL ,description = NULL)
```

```
## Warning in adfTest(b, lag = 3, type = c("ct"), title = NULL, description =
## NULL): p-value smaller than printed p-value
```

```
m2@test$p.value
```

∴ We do not have unit root now !

and the data are all now stationary.

```
##
## 0.01
```

```
m3 = adfTest(c , lag = 3, type = c("ct"),title = NULL ,description = NULL)
```

```
## Warning in adfTest(c, lag = 3, type = c("ct"), title = NULL, description =
## NULL): p-value smaller than printed p-value
```

```
m3@test$p.value

##
## 0.01

fit <- lm(a~b+c)
summary(fit)

##
## Call:
## lm(formula = a ~ b + c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7483 -0.3624  0.0823  0.4489  3.7579
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.58521    0.11385   5.140 8.93e-07 ***
## b           -0.07173    0.11286  -0.636    0.526
## c            0.02590    0.06183   0.419    0.676
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8654 on 142 degrees of freedom
## Multiple R-squared:  0.003745,   Adjusted R-squared:  -0.01029
## F-statistic: 0.2669 on 2 and 142 DF,  p-value: 0.7661

error = residuals(fit)

m4 = adfTest(error,lag = 3, type = c("ct"),title = NULL ,description = NULL)

## Warning in adfTest(error, lag = 3, type = c("ct"), title = NULL,
description =
## NULL): p-value smaller than printed p-value

m4@test$p.value

##
## 0.01

a.L.1 = Lag(a,k=1)
b.L.1 = Lag(b,k=1)
c.L.1 = Lag(c,k=1)
error.L.1=Lag(error,k=1)
fit <- lm(a~a.L.1+b.L.1+c.L.1 + error.L.1)
summary(fit)

##
## Call:
## lm(formula = a ~ a.L.1 + b.L.1 + c.L.1 + error.L.1)
##
```

after make the fitted model + after all data are stationary,
the cointegration will be cointegrated (as pv = 0.01 where < 0.1)
now we can reject Ho and now the model is corrected.

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7108 -0.3201  0.0205  0.3926  4.0134
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.48194    0.11712   4.115 6.58e-05 ***
## a.L.1        0.22925    0.07934   2.889  0.00447 **
## b.L.1       -0.15054    0.10699  -1.407  0.16163
## c.L.1        0.07573    0.05870   1.290  0.19914
## error.L.1         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8173 on 140 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.08127,    Adjusted R-squared:  0.06158
## F-statistic: 4.128 on 3 and 140 DF,  p-value: 0.007703
```

Δ UK CA US (GDP)

$$\widehat{\Delta UK\,GDP}_t = \underset{(0.117)}{0.482} + \underset{(0.079)}{0.229}\,\Delta UK_{t-1} - \underset{(0.107)}{0.151}\,\Delta CA_{t-1} + \underset{(0.059)}{0.076}\,\Delta US_{t-1} + \emptyset$$

Since we have no error term, we will interpret the fitted model.

- the change in UK GDP has the base 0.482 base intercept, it will be determined by last period of the shock

For the last period:

in UK last period by 0.229 percent, the CA GDP have negative impact to the shock on UK GDP. The US GDP will create the leftover shock by little amount.

It is likely that UK can grow by itself or not relying on other developed country that much.

Ly as it is in the EURO zone where An. they have BREXIT, the UK is really independent from the west pressure.