VAR, SVAR and VECM Using the vars package

Henri Makika

May 22, 2019

Contents

Introduction

Depuis la critique de Sims (1980) au début des années quatre-vingt du siècle dernier, l'analyse VAR est devenue un instrument standard en économétrie pour l'analyse de séries chronologiques à plusieurs variables. Les tests statistiques étant très utilisés pour déterminer les interdépendances et les relations dynamiques entre variables, il est rapidement devenu évident qu'enrichir cette méthodologie en incorporant des informations a priori non statistiques a conduit à l'évolution des modèles SVAR qui tentent de contourner ces lacunes. Sims a mis en péril le paradigme des multiples modèles d'équations structurelles proposés par la *Cowles Foundation* dans les années 1930 et 1950 du siècle dernier. Granger (1981) et Engle & Granger (1987) ont doté les économétriciens d'un puissant outil de modélisation et de tester les relations économiques, à savoir le concept d'intégration et de cointégration. De nos jours, ces traces de recherche sont unifiées sous la forme de modèles de correction d'erreur vectorielle et de correction d'erreur vectorielle de structure. Bien que ces derniers sujets soient laissés de côté dans cette vignette, le lecteur intéressé est renvoyé aux monographies de Lütkepohl (2006), Hendry (1995), Johansen (1995), Hamilton (1994), Banerjee, Dolado, Galbraith & Hendry (1993) et Pfaff (2006) pour une exposition sur les tests de racine unitaire et l'analyse de co-intégration en utilisant le langage R.

À la connaissance de l'auteur, seules les fonctions actuelles sont disponibles dans la distribution de base de R et dans les *packages CRAN dse* (package, voir Gilbert 1993, 1995, 2000) et *fMultivar* (Würtz 2006) pour estimer les modèles de séries chronologiques ARIMA et VARIMA. . Bien que le paquetage CRAN MSBVAR (Brandt 2006) fournisse des méthodes d'estimation des modèles BVR (Bayesian Autoregression Vector Frequentist), les méthodes et fonctions fournies dans les paquetages essayent de combler une lacune dans le paysage des méthodes économétriques de R en fournissant des outils «standard» dans le contexte de l'analyse VAR et SVAR.

1. VAR : Vector autoregressive models

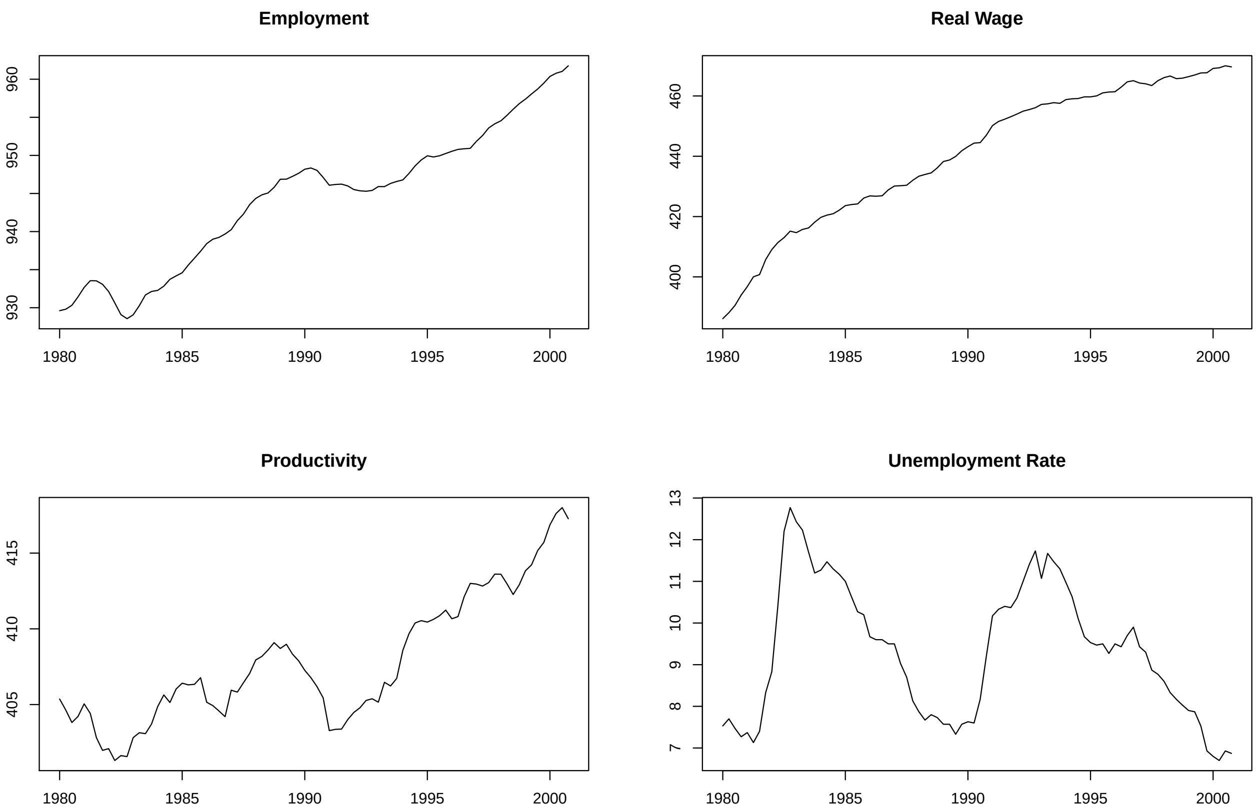
Estimation

Pour la description de données, voir Lütkepohl et al. (2004).

```r
library(vars)
data("Canada")
head(Canada)
```

```
## [1] 929.6105 929.8040 930.3184 931.4277 932.6620 933.5509
```

```r
layout(matrix(1:4, nrow = 2, ncol = 2))

plot.ts(Canada[,1], main = "Employment", ylab = "", xlab = "")
plot.ts(Canada[,2], main = "Productivity", ylab = "", xlab = "")
plot.ts(Canada[,3], main = "Real Wage", ylab = "", xlab = "")
plot.ts(Canada[,4], main = "Unemployment Rate", ylab = "", xlab = "")
```



Explication

La variable e est utilisée pour l'emploi; *prod* est une mesure de la productivité du travail; *rw* assigne le salaire réel et finalement U est le taux de chômage. La fonction d'estimation d'un VAR sur R est *VAR()*. Il se compose de trois arguments: l'objet de matrice de données y (ou un objet pouvant être forcé dans une matrice), l'ordre de décalage entier p et le type de régresseurs déterministes à inclure dans le *VAR(p)*. Un ordre de retard optimal peut être déterminé en fonction d'un critère d'information ou de l'erreur de prédiction finale d'un *VAR(p)* avec la fonction *VARselect()*.

```r
args(VAR)
```

```
## function (y, p = 1, type = c("const", "trend", "both", "none"),
##     season = NULL, exogen = NULL, lag.max = NULL, ic = c("AIC",
##         "HQ", "SC", "FPE"))
## NULL
```

```r
args(VARselect)
```

```
## function (y, lag.max = 10, type = c("const", "trend", "both",
##     "none"), season = NULL, exogen = NULL)
## NULL
```

```r
VARselect(Canada, lag.max = 5, type = "const")
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      3      2      2      3
##
## $criteria
##                   1           2            3            4            5
## AIC(n) -5.817851996 -6.35093701 -6.397756084 -6.145942174 -5.926500201
## HQ(n)  -5.577529641 -5.91835677 -5.772917961 -5.328846166 -4.917146309
## SC(n)  -5.217991781 -5.27118862 -4.838119523 -4.106417440 -3.407087295
## FPE(n)  0.002976003  0.00175206  0.001685528  0.002201523  0.002811116
```

VARselect() permet à l'utilisateur de déterminer une longueur de décalage optimale en fonction d'un critère d'information ou de l'erreur de prédiction finale d'un processus *VAR(p)* empirique. Chacune de ces mesures est définie dans le fichier d'aide de la fonction. La fonction renvoie un objet de liste avec l'ordre de décalage optimal en fonction de chacun des critères, ainsi qu'une matrice contenant les valeurs de tous les décalages jusqu'à lag.max. Selon les critères plus conservateurs de *SC(n)* et *HQ(n)*, l'ordre de latence optimal empirique est de 2. Veuillez noter que le calcul de ces critères est basé sur la même taille d'échantillon et que, par conséquent, les critères pourraient prendre légèrement différentes les valeurs permettant d'estimer un VAR pour l'ordre choisi.

Dans une étape suivante, le VAR (2) est estimé avec la fonction *VAR()* et, en tant que régresseurs déterministes, une constante est incluse.

```r
var2c <- VAR(Canada, p = 2, type = "const")
names(var2c)
```

```
##  [1] "varresult"    "datamat"      "y"            "type"
##  [5] "p"            "K"            "obs"          "totobs"
##  [9] "restrictions" "call"
```

Avec la fonction *summary*, on a tout les détails de notre estimation et la fonction *plot* pour le graphique du modèle VAR(2) estimé.

```r
summary(var2c)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: e, prod, rw, U
## Deterministic variables: const
## Sample size: 82
## Log Likelihood: -175.819
## Roots of the characteristic polynomial:
## 0.995 0.9081 0.9081 0.7381 0.7381 0.1856 0.1429 0.1429
## Call:
## VAR(y = Canada, p = 2, type = "const")
```

```
##
##
## Estimation results for equation e:
## ==================================
## e = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + const
##
##           Estimate Std. Error t value Pr(>|t|)
## e.l1     1.638e+00  1.500e-01  10.918  < 2e-16 ***
## prod.l1  1.673e-01  6.114e-02   2.736  0.00780 **
## rw.l1   -6.312e-02  5.524e-02  -1.143  0.25692
## U.l1     2.656e-01  2.028e-01   1.310  0.19444
## e.l2    -4.971e-01  1.595e-01  -3.116  0.00262 **
## prod.l2 -1.017e-01  6.607e-02  -1.539  0.12824
## rw.l2    3.844e-03  5.552e-02   0.069  0.94499
## U.l2     1.327e-01  2.073e-01   0.640  0.52418
## const   -1.370e+02  5.585e+01  -2.453  0.01655 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.3628 on 73 degrees of freedom
## Multiple R-Squared: 0.9985,  Adjusted R-squared: 0.9984
## F-statistic:  6189 on 8 and 73 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation prod:
## =====================================
## prod = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + const
##
##           Estimate Std. Error t value Pr(>|t|)
## e.l1      -0.17277    0.26977  -0.640  0.52390
## prod.l1    1.15043    0.10995  10.464 3.57e-16 ***
## rw.l1      0.05130    0.09934   0.516  0.60710
## U.l1      -0.47850    0.36470  -1.312  0.19362
## e.l2       0.38526    0.28688   1.343  0.18346
## prod.l2   -0.17241    0.11881  -1.451  0.15104
## rw.l2     -0.11885    0.09985  -1.190  0.23778
## U.l2       1.01592    0.37285   2.725  0.00805 **
## const   -166.77552  100.43388  -1.661  0.10109
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.6525 on 73 degrees of freedom
## Multiple R-Squared: 0.9787,  Adjusted R-squared: 0.9764
## F-statistic: 419.3 on 8 and 73 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation rw:
## ===================================
## rw = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + const
##
##           Estimate Std. Error t value Pr(>|t|)
## e.l1     -0.268833   0.322619  -0.833    0.407
```

```
## prod.l1  -0.081065   0.131487  -0.617    0.539
## rw.l1     0.895478   0.118800   7.538 1.04e-10 ***
## U.l1      0.012130   0.436149   0.028    0.978
## e.l2      0.367849   0.343087   1.072    0.287
## prod.l2  -0.005181   0.142093  -0.036    0.971
## rw.l2     0.052677   0.119410   0.441    0.660
## U.l2     -0.127708   0.445892  -0.286    0.775
## const   -33.188339 120.110525  -0.276    0.783
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.7803 on 73 degrees of freedom
## Multiple R-Squared: 0.9989,  Adjusted R-squared: 0.9987
## F-statistic:  8009 on 8 and 73 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation U:
## ==================================
## U = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + const
##
##          Estimate Std. Error t value Pr(>|t|)
## e.l1     -0.58076    0.11563  -5.023 3.49e-06 ***
## prod.l1  -0.07812    0.04713  -1.658 0.101682
## rw.l1     0.01866    0.04258   0.438 0.662463
## U.l1      0.61893    0.15632   3.959 0.000173 ***
## e.l2      0.40982    0.12296   3.333 0.001352 **
## prod.l2   0.05212    0.05093   1.023 0.309513
## rw.l2     0.04180    0.04280   0.977 0.331928
## U.l2     -0.07117    0.15981  -0.445 0.657395
## const   149.78056   43.04810   3.479 0.000851 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.2797 on 73 degrees of freedom
## Multiple R-Squared: 0.9726,  Adjusted R-squared: 0.9696
## F-statistic:   324 on 8 and 73 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##              e      prod       rw        U
## e     0.131635 -0.007469 -0.04210 -0.06909
## prod -0.007469  0.425711  0.06461  0.01392
## rw   -0.042099  0.064613  0.60886  0.03422
## U    -0.069087  0.013923  0.03422  0.07821
##
## Correlation matrix of residuals:
##             e     prod      rw       U
## e     1.00000 -0.03155 -0.1487 -0.6809
## prod -0.03155  1.00000  0.1269  0.0763
## rw   -0.14870  0.12691  1.0000  0.1568
## U    -0.68090  0.07630  0.1568  1.0000
```

Comme on a prit le *lag* 2, c'est à dire VAR(2), on remarque que toute les variables leurs coefficients ne sont pas statistiquement significatifs.

```r
par(mar = rep(2, 4))
plot(var2c)
```

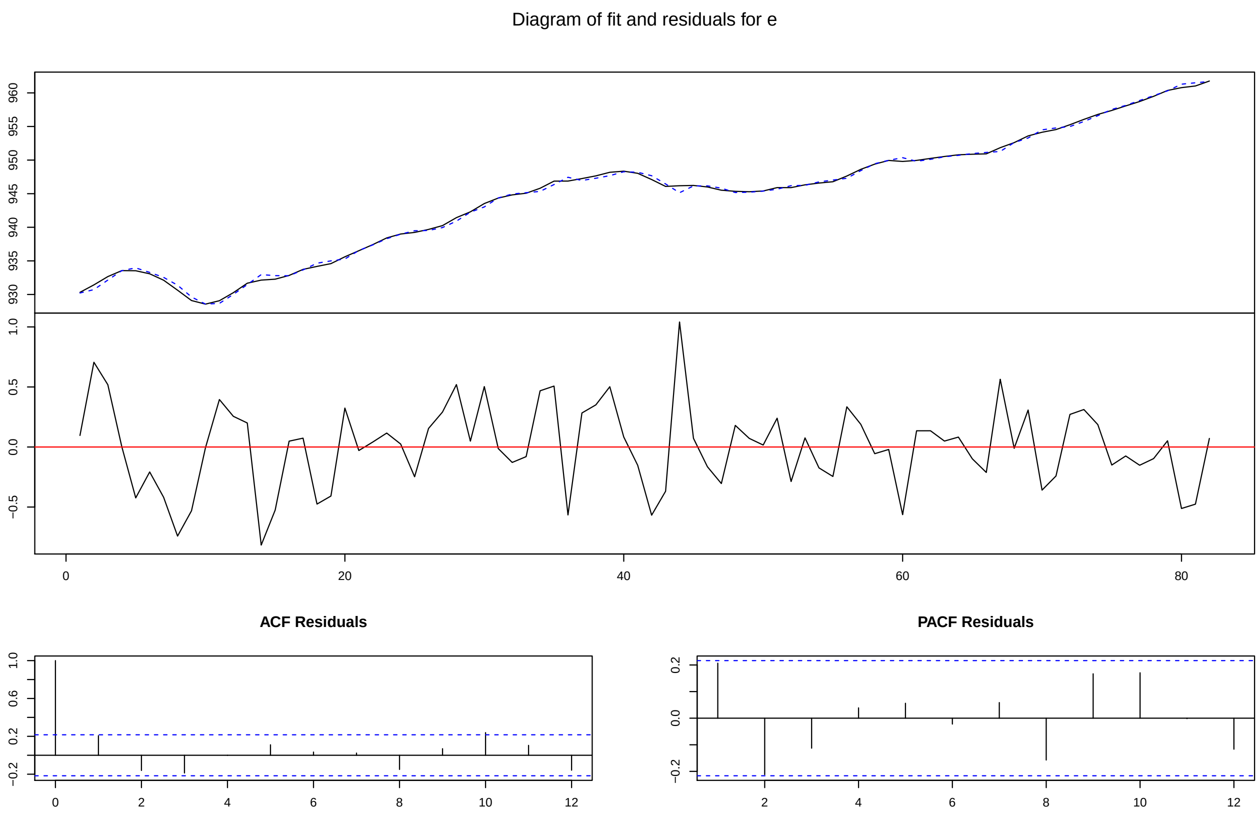


Diagram of fit and residuals for prod

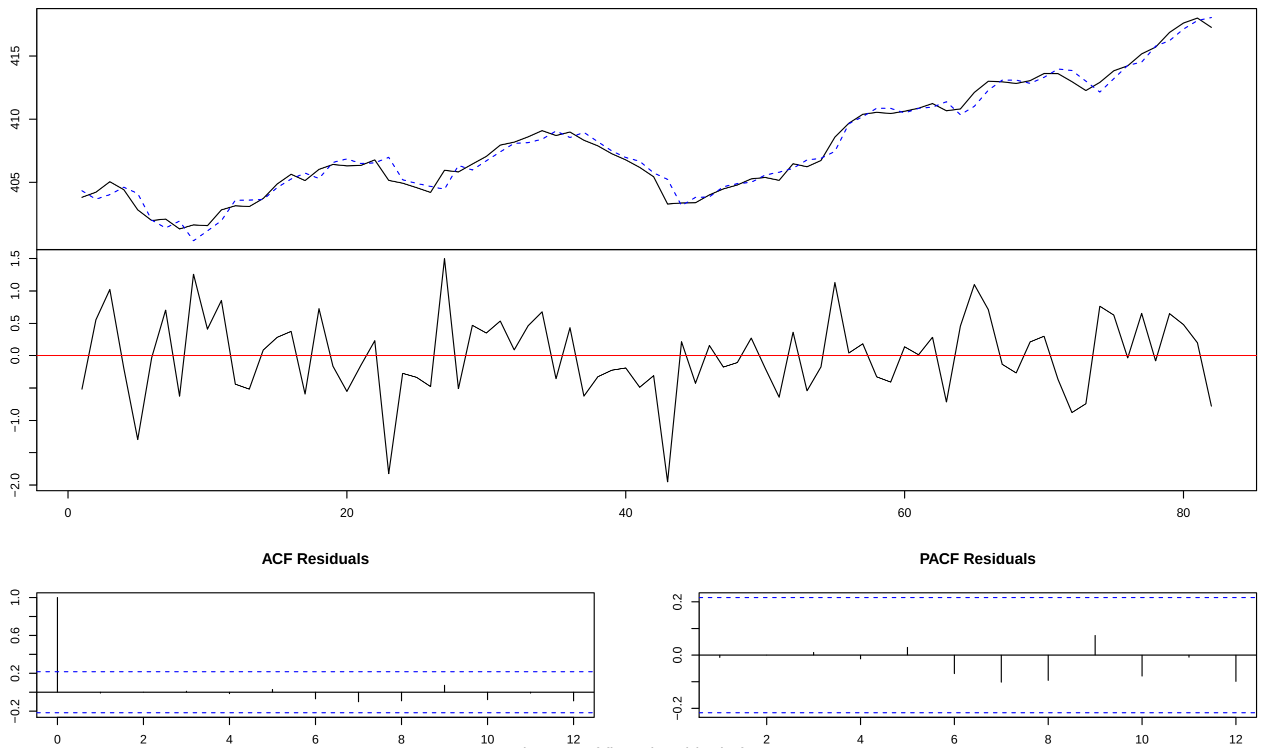
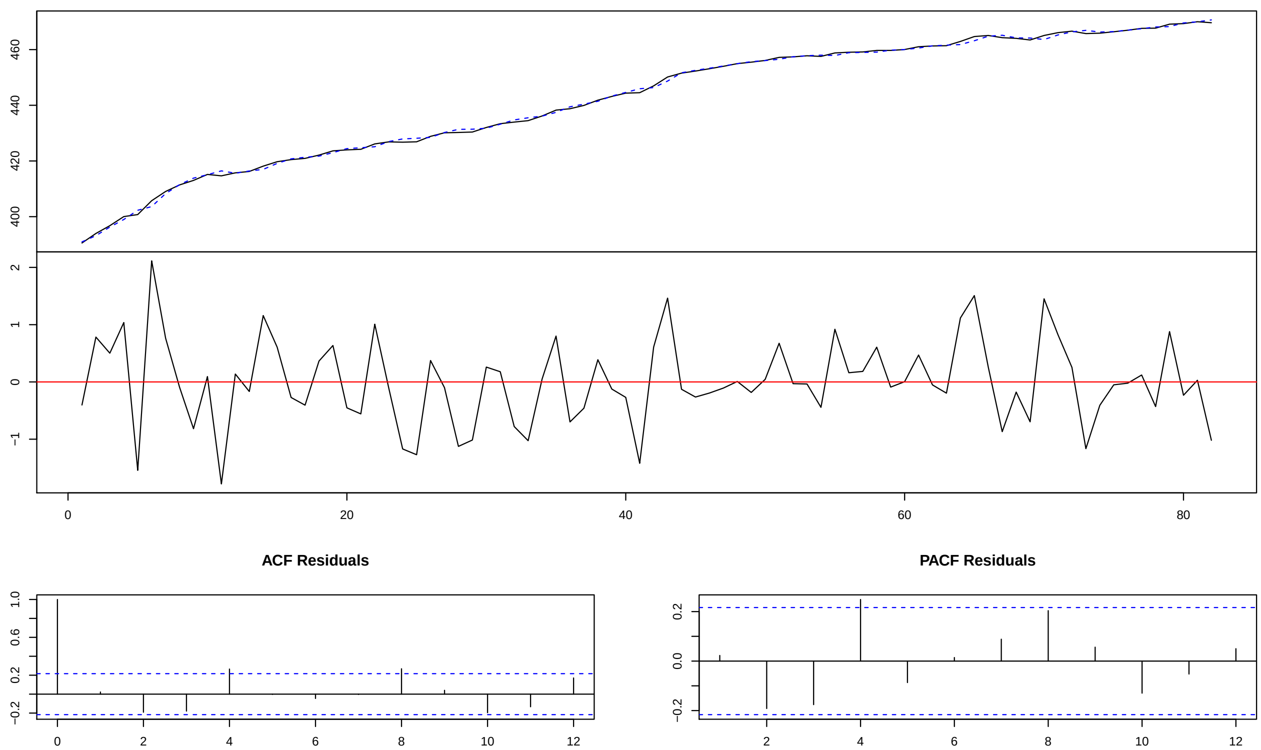
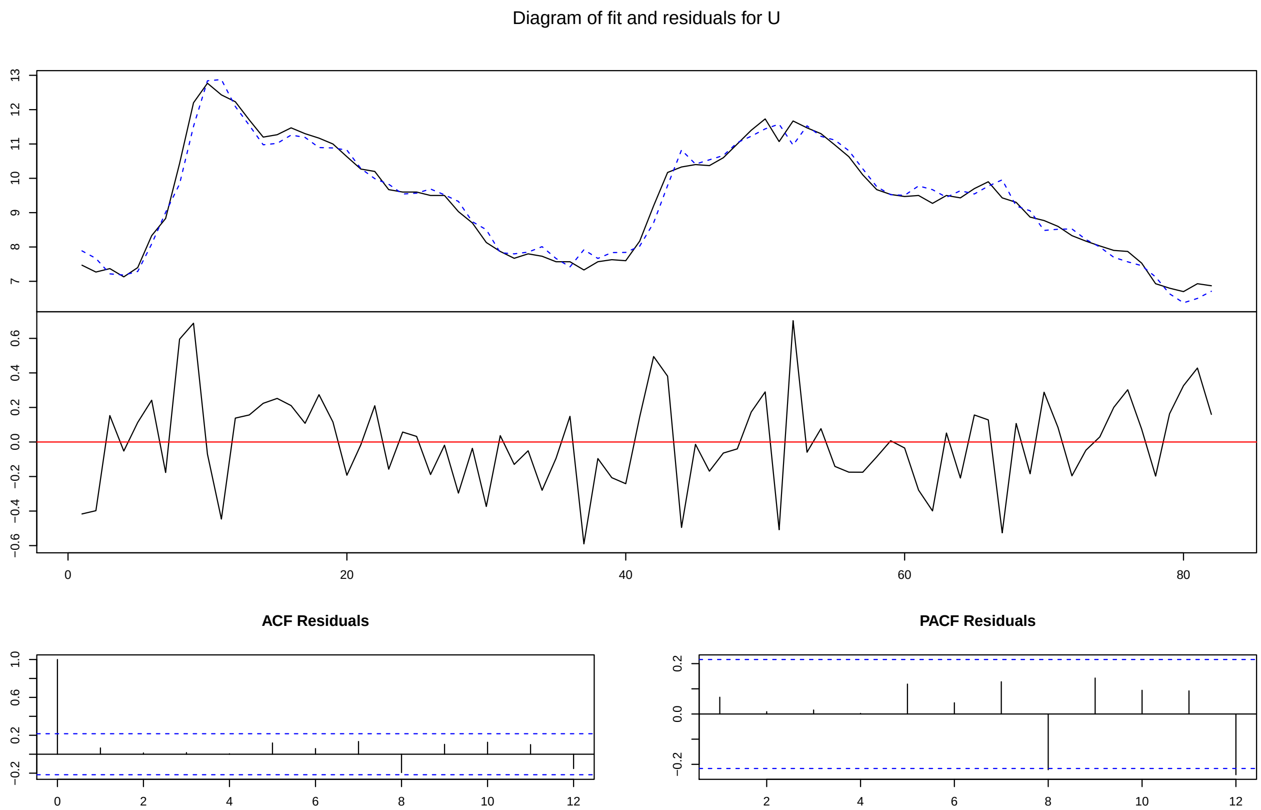


Diagram of fit and residuals for rw





Avant de procéder à l'estimation des VAR restreints, examinons d'abord la méthode de tracé pour les objets avec attribut de classe varest et la fonction roots () pour vérifier la stabilité du VAR, brièvement mentionnés à la fin de la section précédente. Pour chaque équation dans un VAR, un graphique constitué d'un diagramme d'ajustement, d'un graphique de résidus, de la fonction d'autocorrélation et d'une autocorrélation partielle des résidus est présenté.

```
roots(var2c)
```

```
## [1] 0.9950338 0.9081062 0.9081062 0.7380565 0.7380565 0.1856381 0.1428889
## [8] 0.1428889
```

Bien que la première valeur propre soit assez proche de l'unité, pour simplifier, nous supposons un processus VAR (2) stable avec une constante comme régresseur déterministe.

Restricted VARs

```
args(restrict)
```

```
## function (x, method = c("ser", "manual"), thresh = 2, resmat = NULL)
## NULL
```

```
var2c.ser <- restrict(var2c, method = "ser", thresh = 2)
var2c.ser$restrictions
```

```
##      e.l1 prod.l1 rw.l1 U.l1 e.l2 prod.l2 rw.l2 U.l2 const
## e       1       1     1    1    1       0     0    0     1
## prod    0       1     0    0    1       0     1    1     1
## rw      0       1     1    0    1       0     0    1     0
## U       1       0     0    1    1       0     1    0     1
```

```r
Acoef(var2c.ser)
```

```
## [[1]]
##            e.l1     prod.l1       rw.l1      U.l1
## e     1.7245893  0.07872263 -0.05370603 0.3915061
## prod  0.0000000  1.00809918  0.00000000 0.0000000
## rw    0.0000000 -0.11816412  0.96382332 0.0000000
## U    -0.6954549  0.00000000  0.00000000 0.5600228
##
## [[2]]
##             e.l2 prod.l2       rw.l2       U.l2
## e    -0.60070476       0  0.00000000  0.0000000
## prod  0.28943990       0 -0.09728839  0.7503647
## rw    0.07092345       0  0.00000000 -0.1930013
## U     0.52253799       0  0.05670873  0.0000000
```

```r
plot(var2c.ser)
```

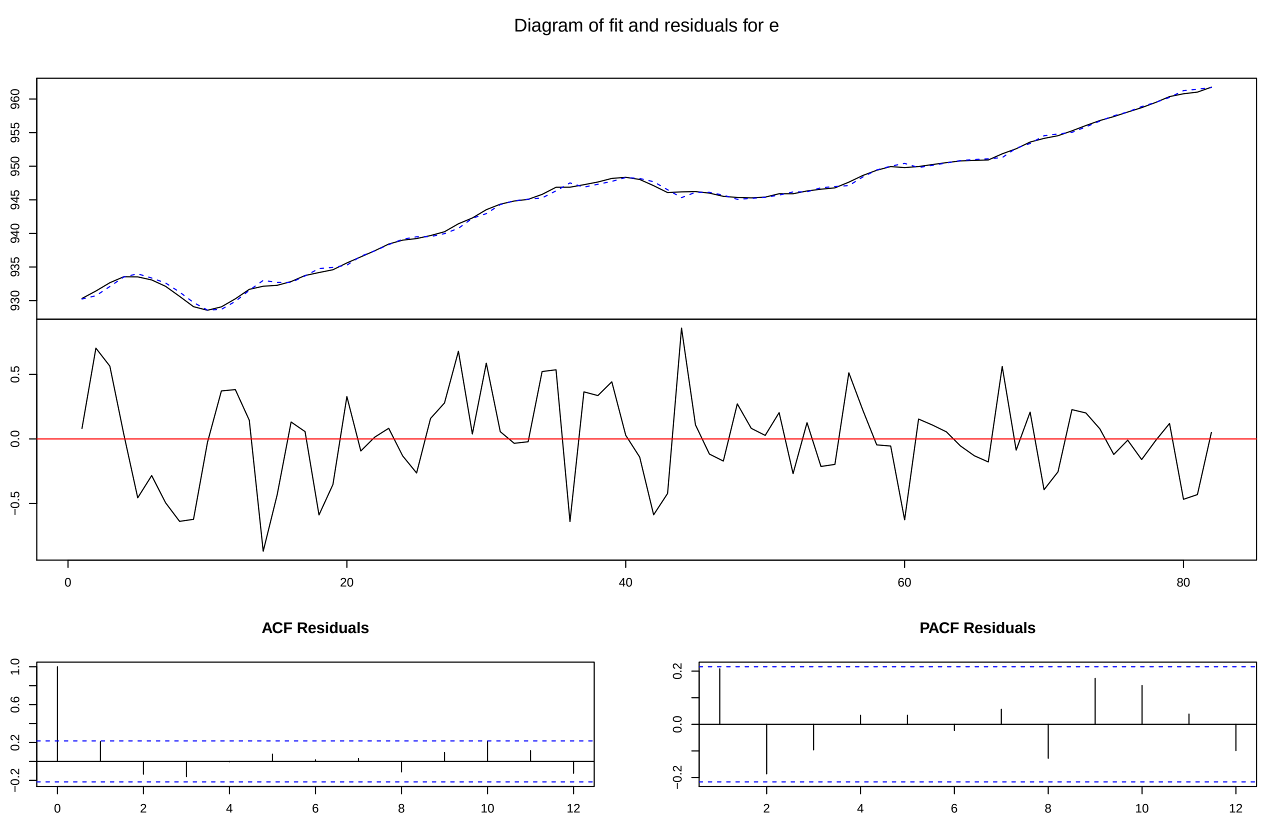


Diagram of fit and residuals for prod

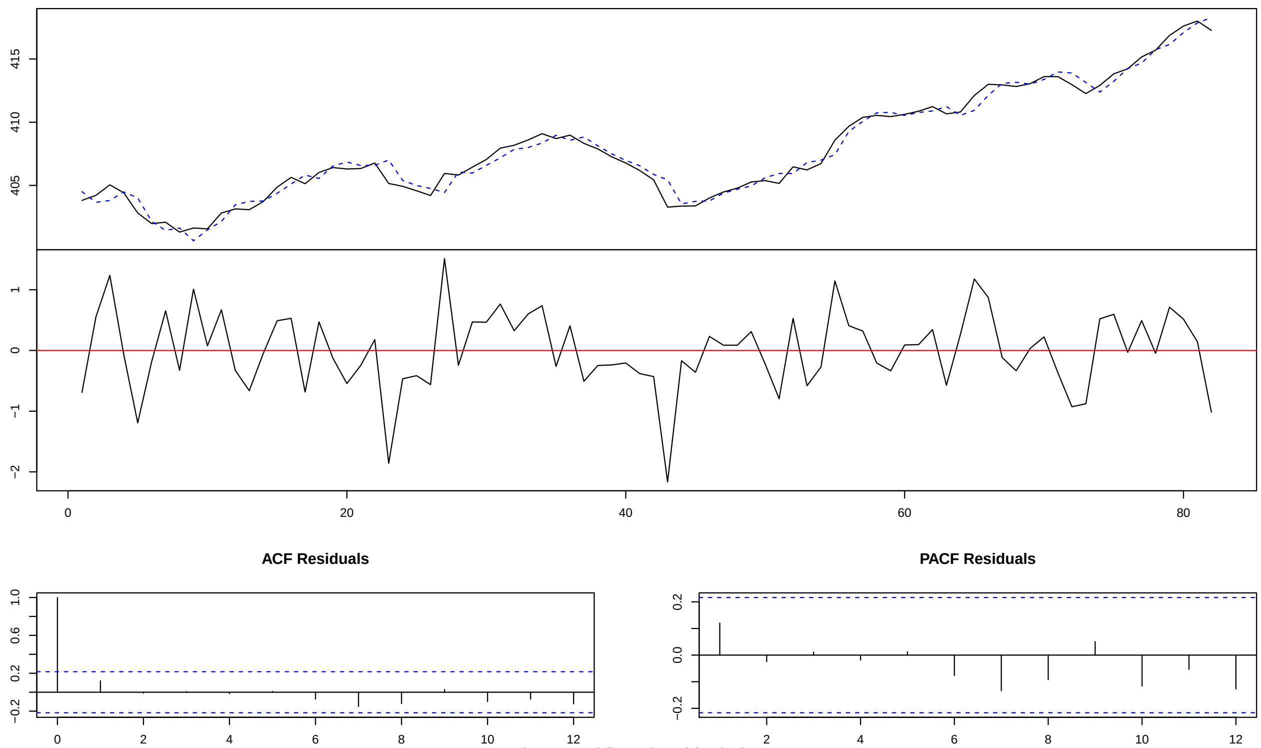
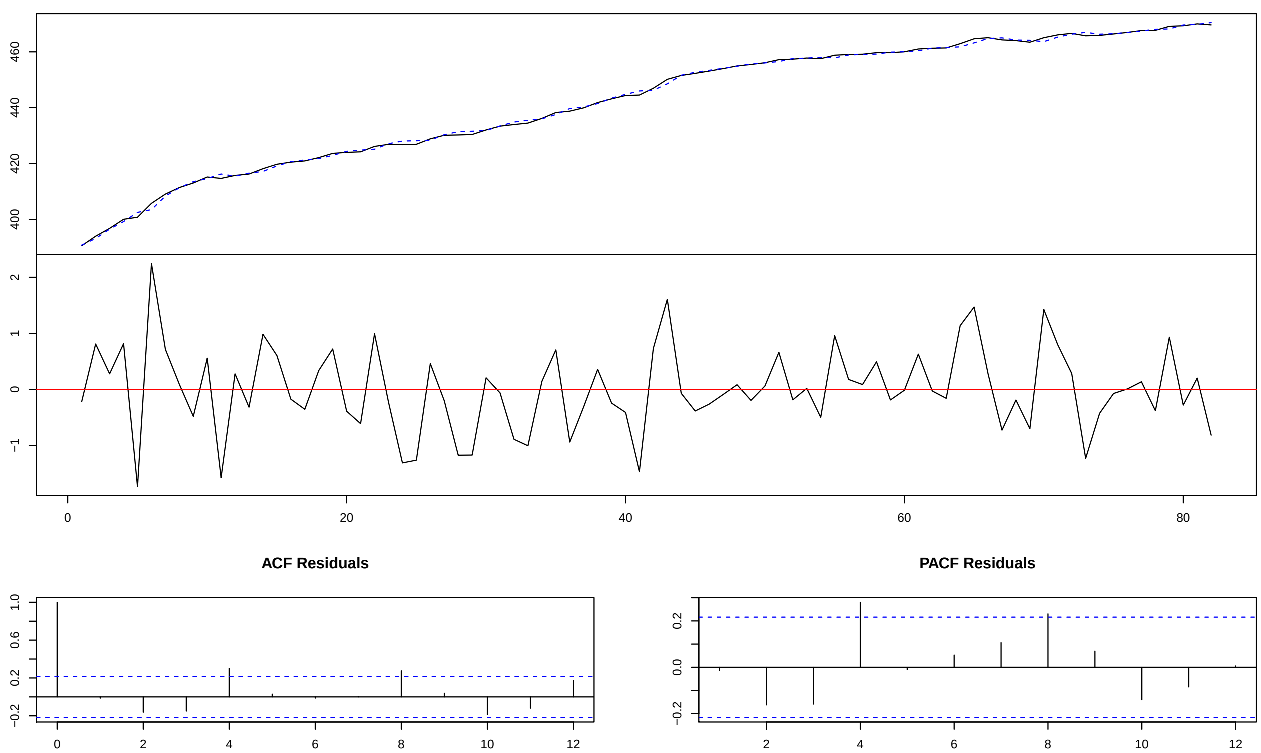
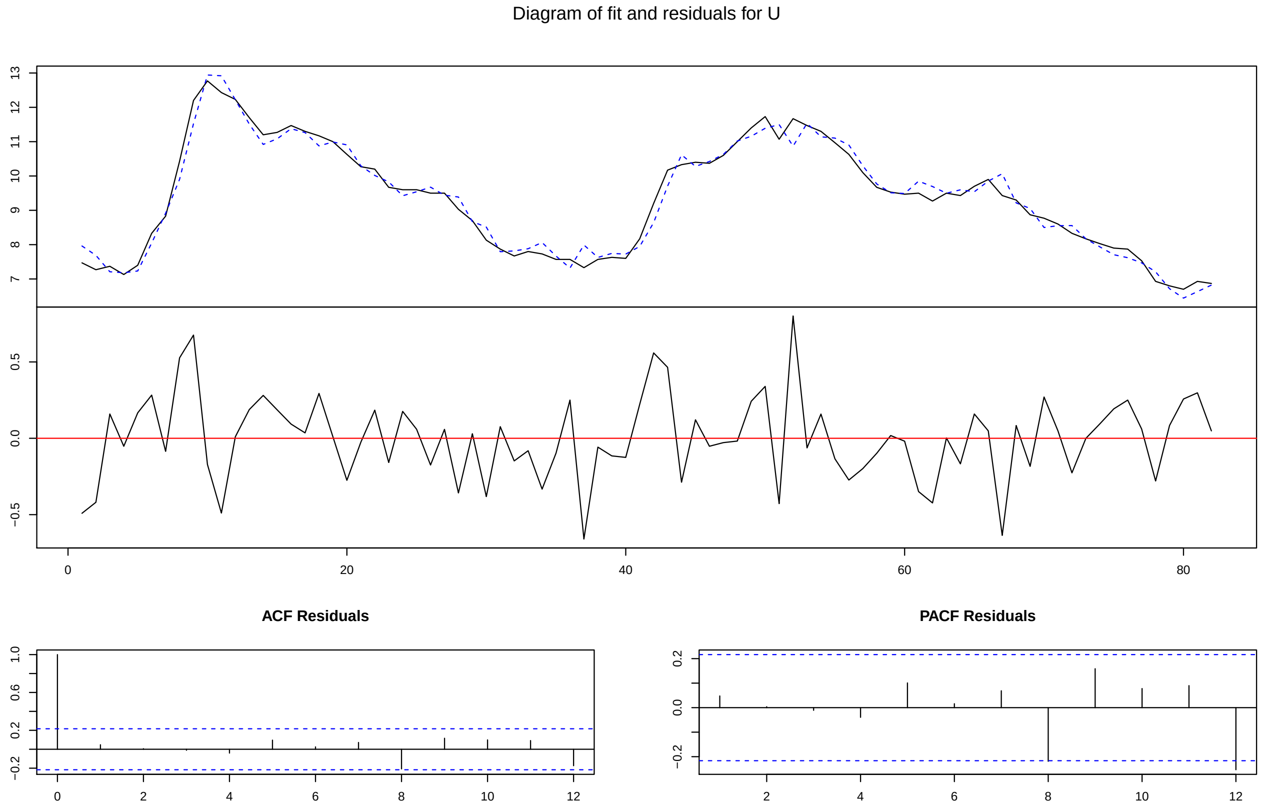


Diagram of fit and residuals for rw





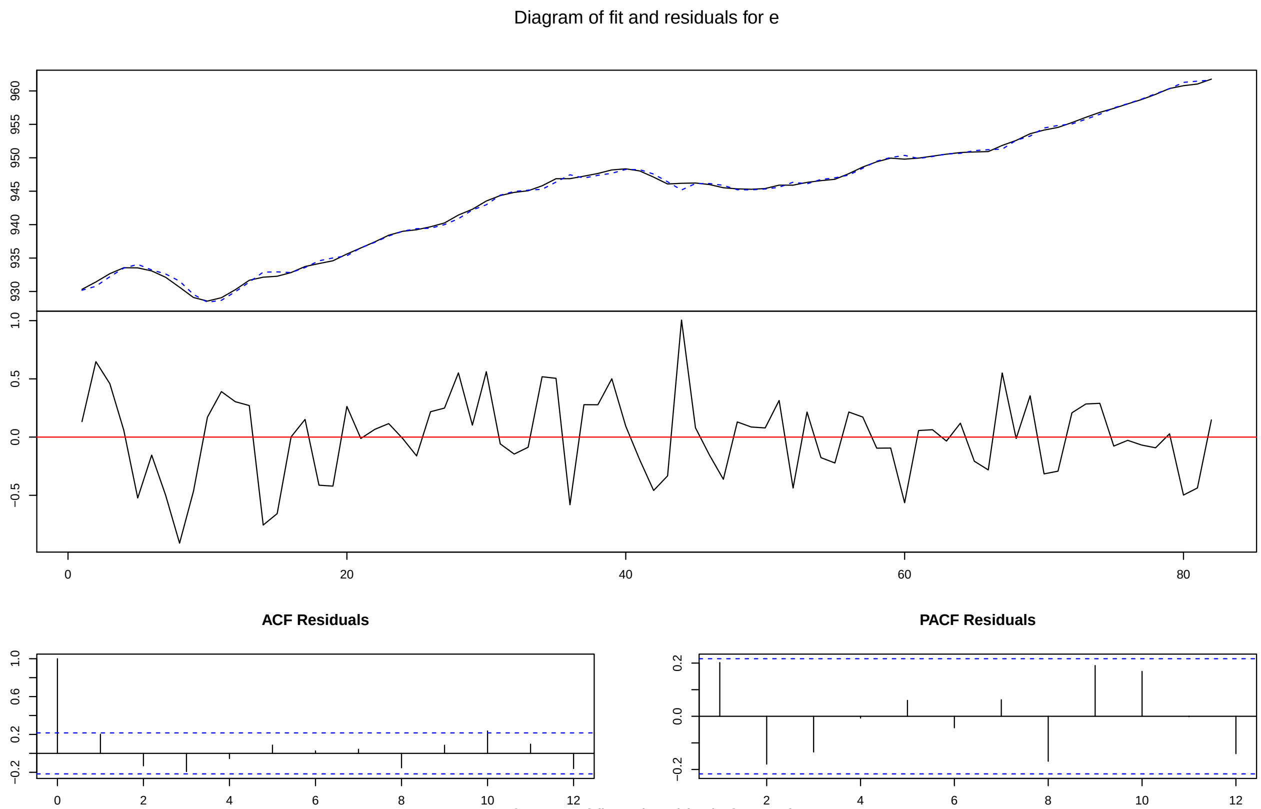

```r
res <- matrix(rep(1, 36), nrow = 4, ncol = 9)
res[1, 3] <- 0
res[1, 4] <- 0

var2c.man <- restrict(var2c, method = "manual", resmat = res)
var2c.man$restrictions
```

```
##      e.l1 prod.l1 rw.l1 U.l1 e.l2 prod.l2 rw.l2 U.l2 const
## e       1       1     0    0    1       1     1    1     1
## prod    1       1     1    1    1       1     1    1     1
## rw      1       1     1    1    1       1     1    1     1
## U       1       1     1    1    1       1     1    1     1
```

```r
Acoef(var2c.man)
```

```
## [[1]]
##            e.l1     prod.l1      rw.l1       U.l1
## e     1.5079689  0.16584386 0.00000000  0.00000000
## prod -0.1727658  1.15042820 0.05130390 -0.47850131
## rw   -0.2688329 -0.08106500 0.89547833  0.01213003
## U    -0.5807638 -0.07811707 0.01866214  0.61893150
##
## [[2]]
##            e.l2      prod.l2       rw.l2        U.l2
## e    -0.4043821 -0.095383974 -0.04593897  0.33088442
## prod  0.3852589 -0.172411873 -0.11885104  1.01591801
## rw    0.3678489 -0.005180947  0.05267656 -0.12770826
## U     0.4098182  0.052116684  0.04180115 -0.07116885
```

```r
plot(var2c.man)
```



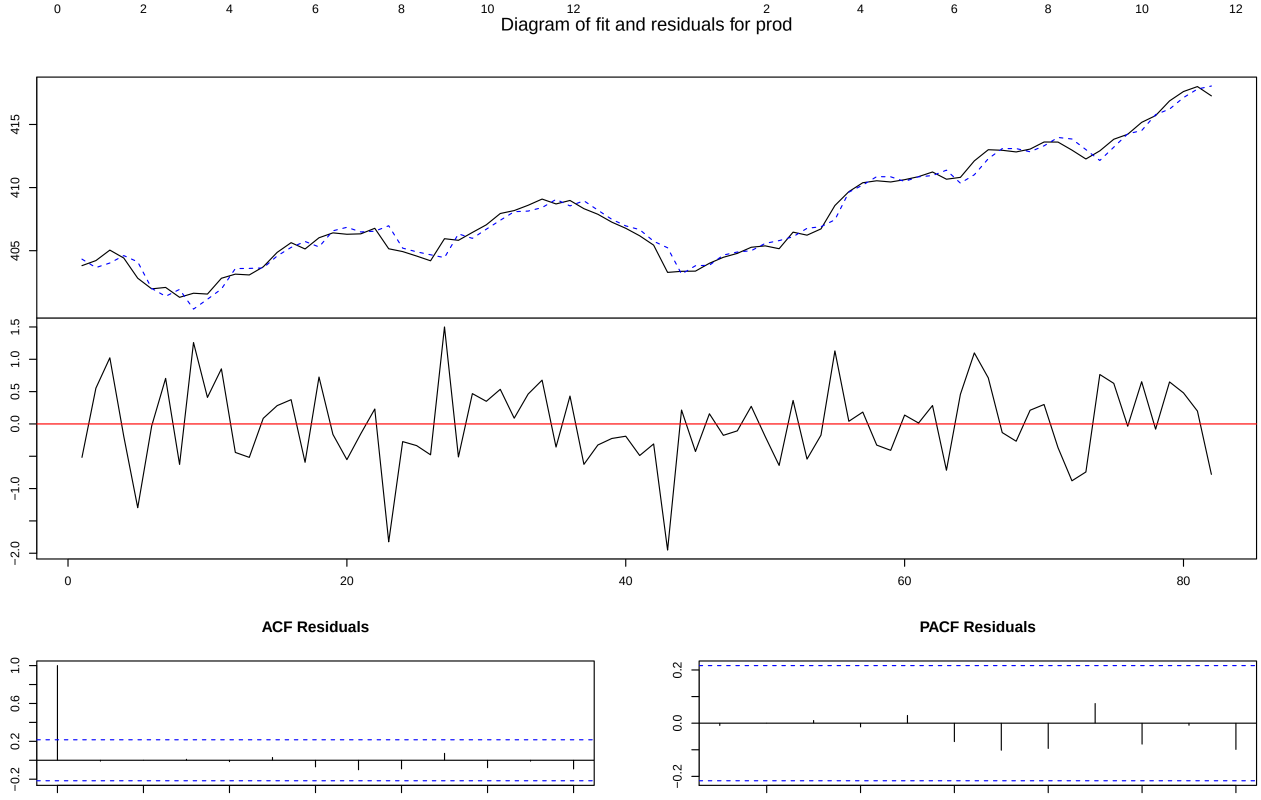


Diagram of fit and residuals for rw

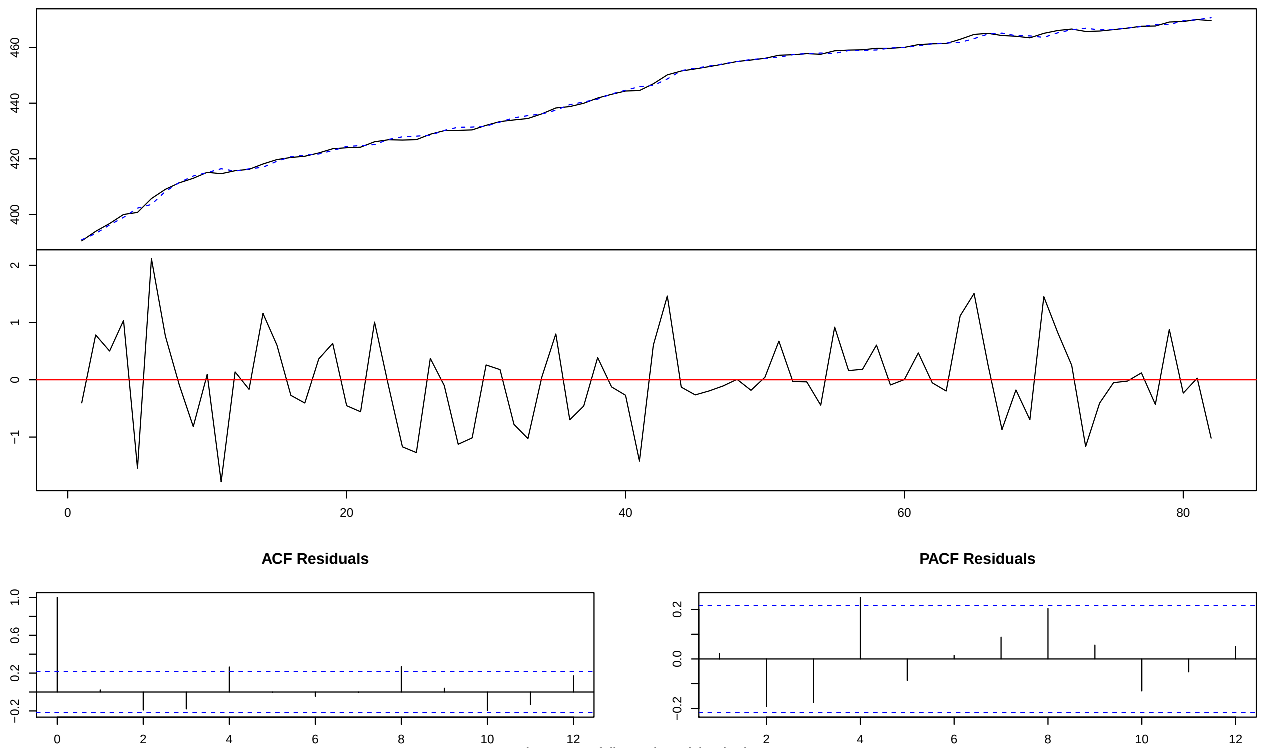
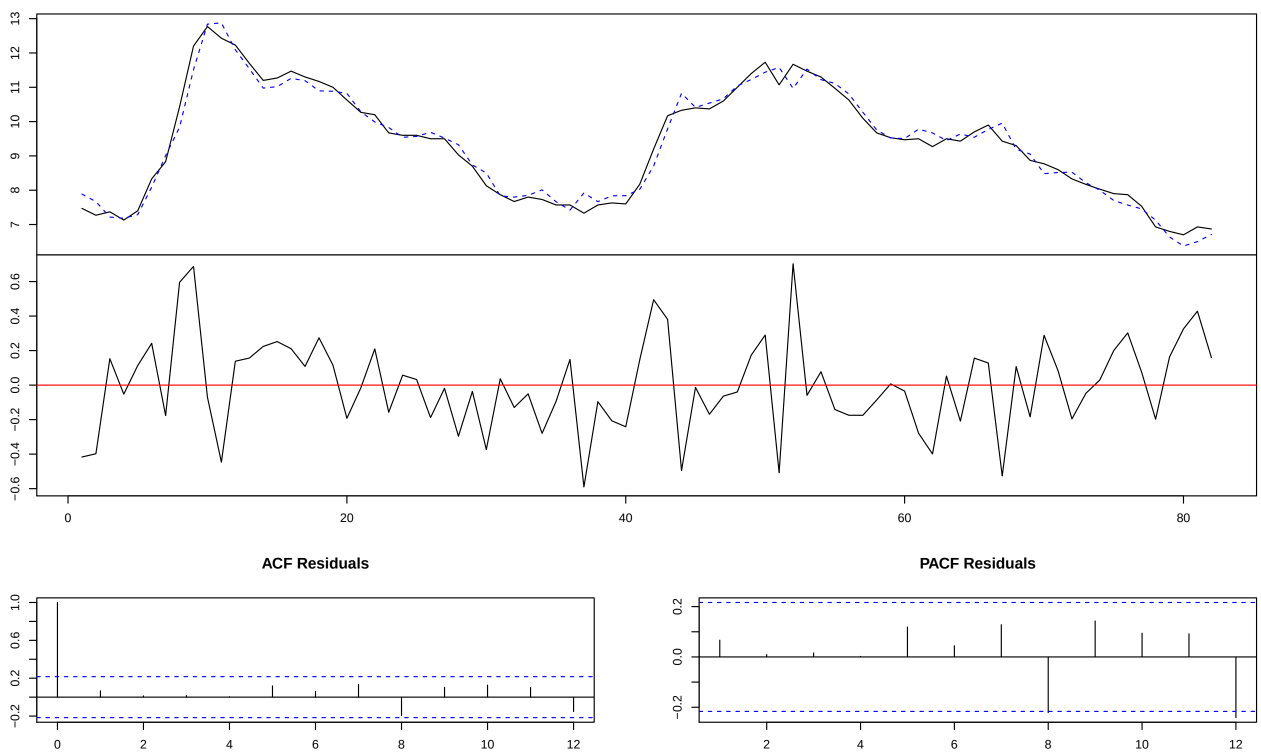


Diagram of fit and residuals for U



Diagnostic testing

```r
args(arch.test)
```

```
## function (x, lags.single = 16, lags.multi = 5, multivariate.only = TRUE)
## NULL
```

```r
var2c.arch.test <- arch.test(var2c)
names(var2c.arch.test)
```

```
## [1] "resid"    "arch.mul"
```

```r
var2c.arch.test
```

```
##
##  ARCH (multivariate)
##
## data:  Residuals of VAR object var2c
## Chi-squared = 538.89, df = 500, p-value = 0.1112
```

Les tests de normalité de Jarque-Bera pour les séries univariée et multivariée sont mis en œuvre et appliqués aux résidus d'un VAR tests séparés pour l'asymétrie multivariée et le kurtosis (voir Bera & Jarque [1980], [1981] et Jarque & Bera [1987] et Lütkepohl [2006]). Les versions univoques du test de Jarque-Bera sont appliquées aux résidus de chaque équation. Une version multivariée de ce test peut être calculée en utilisant les résidus normalisés par une décomposition de Choleski de la matrice de variance-covariance pour les résidus centrés. Veuillez noter que dans ce cas, le résultat du test dépend de l'ordre des variables.

```r
var2c.norm <- normality.test(var2c, multivariate.only = TRUE)
names(var2c.norm)
```

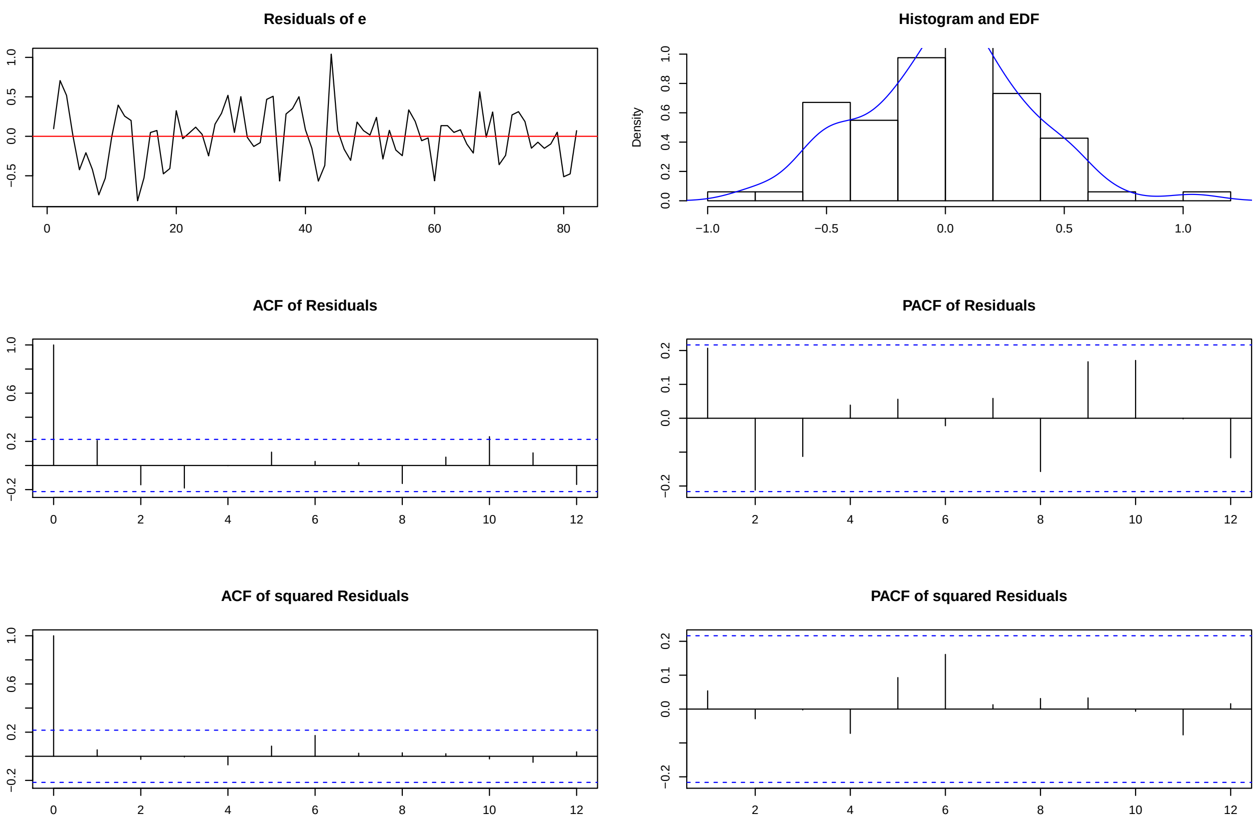
```
## [1] "resid"  "jb.mul"
```

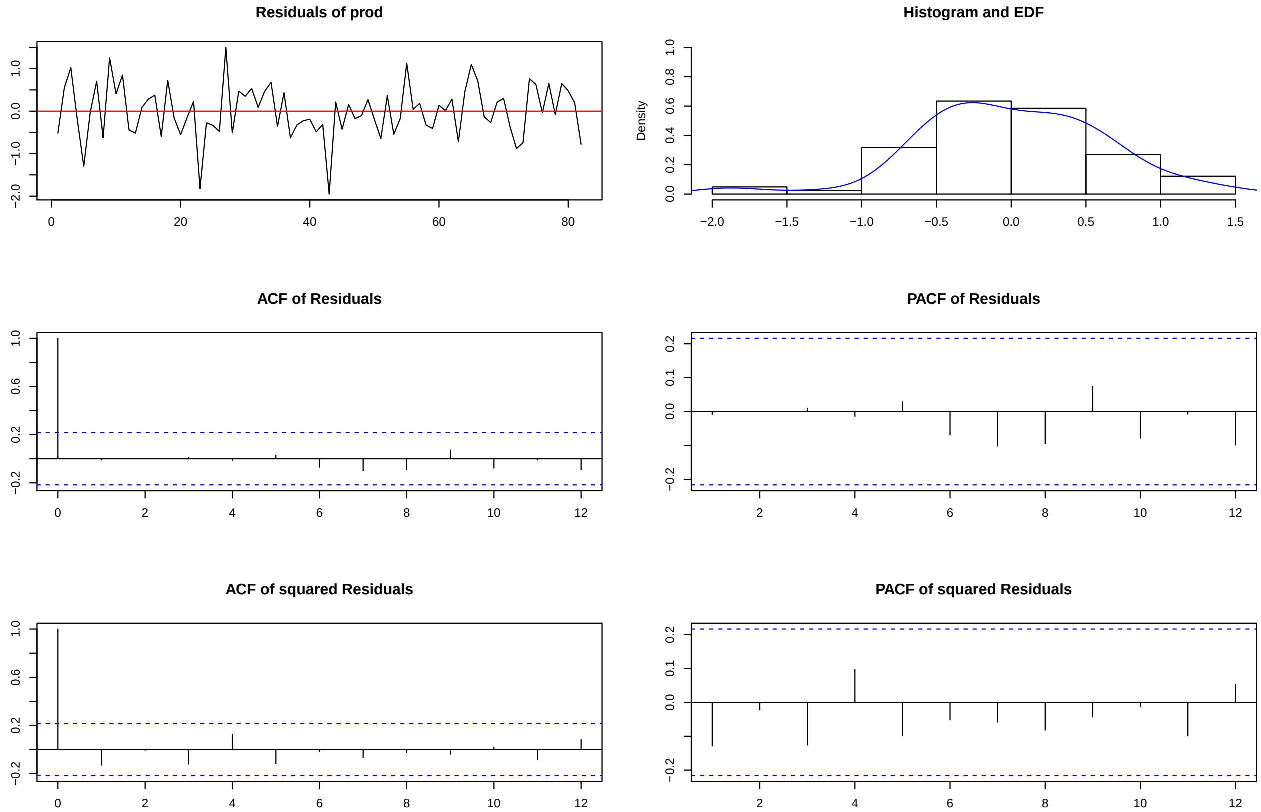
```r
var2c.norm
```

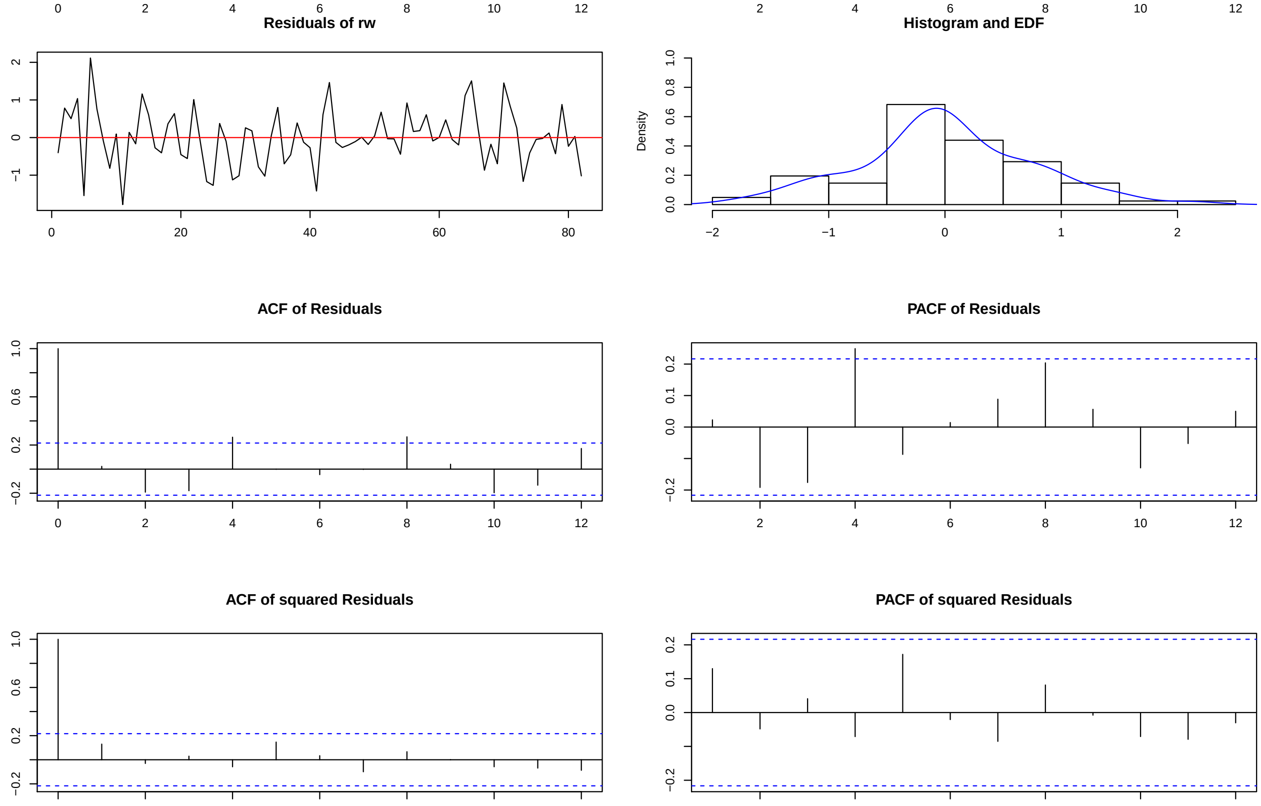
```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var2c
## Chi-squared = 5.094, df = 8, p-value = 0.7475
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var2c
## Chi-squared = 1.7761, df = 4, p-value = 0.7769
##
##
```

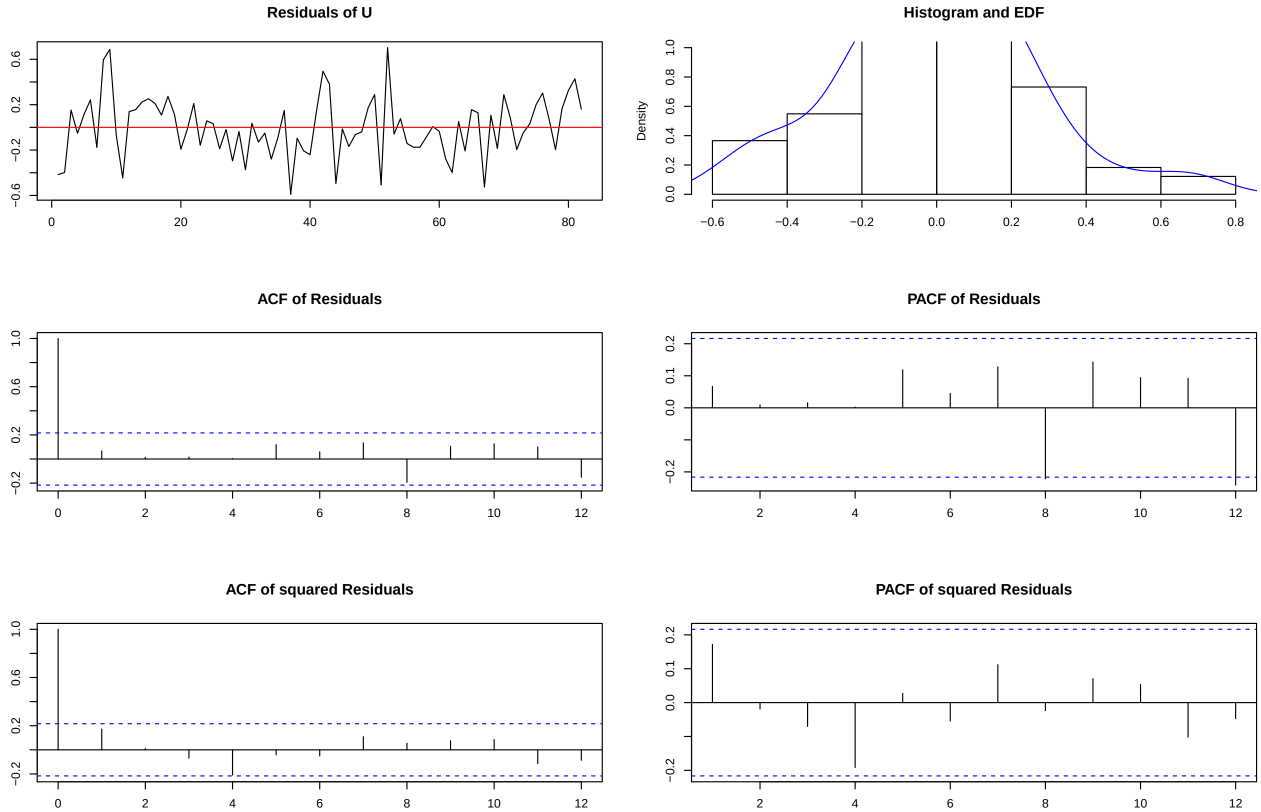
```
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var2c
## Chi-squared = 3.3179, df = 4, p-value = 0.5061
```

```r
plot(var2c.norm)
```




Residuals of prod
Histogram and EDF
Density
ACF of Residuals
PACF of Residuals
ACF of squared Residuals
PACF of squared Residuals


Residuals of rw
Histogram and EDF
Density
ACF of Residuals
PACF of Residuals
ACF of squared Residuals
PACF of squared Residuals



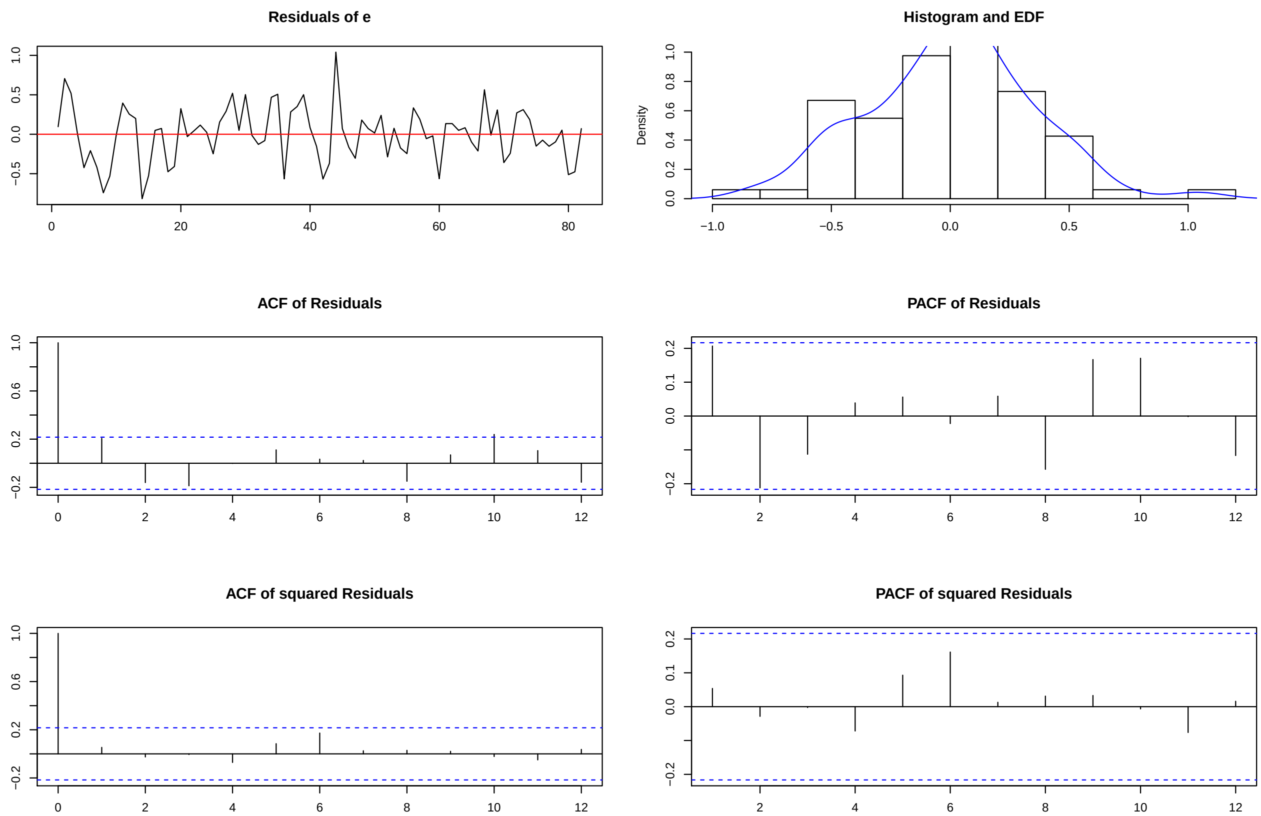

```r
var2c.pt.asy <- serial.test(var2c, lags.pt = 16,
                            type = "PT.asymptotic")
var2c.pt.asy
```

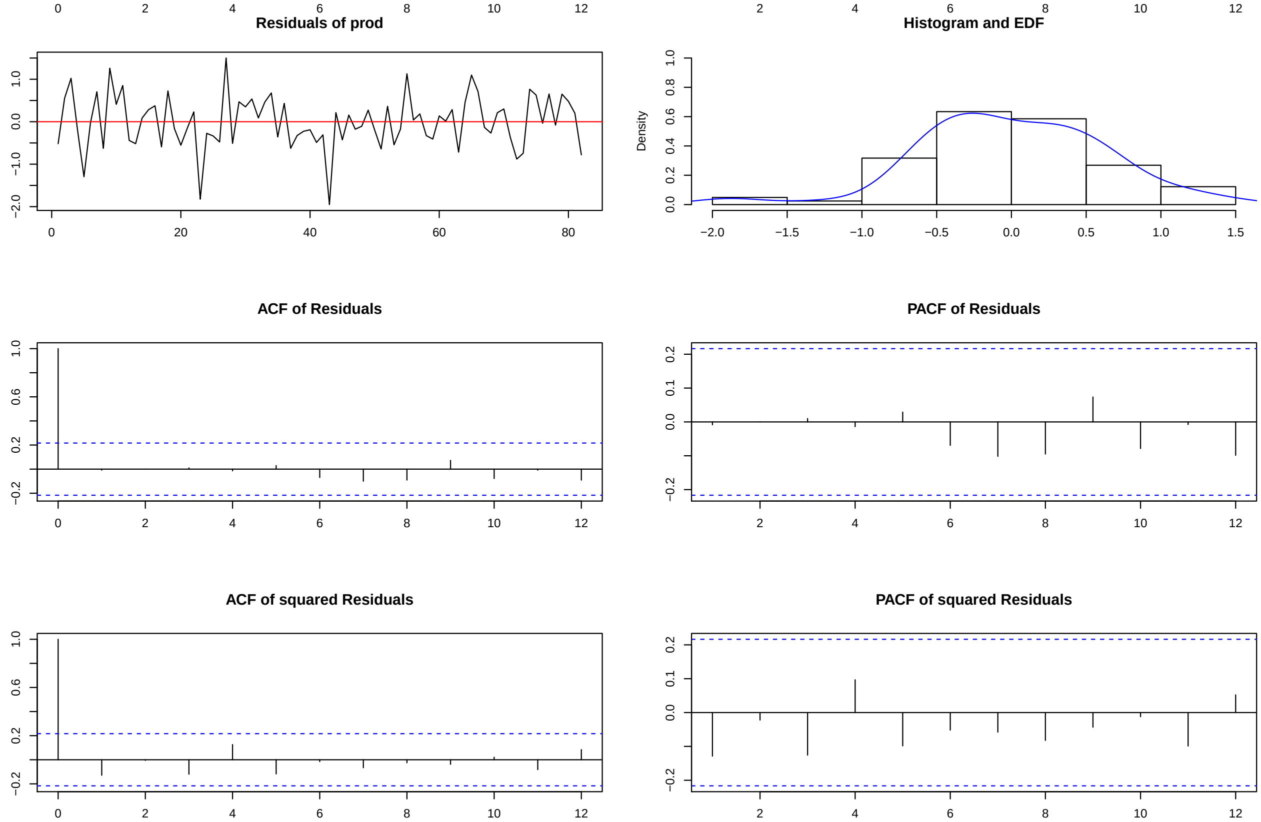
```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var2c
## Chi-squared = 205.35, df = 224, p-value = 0.8092
```

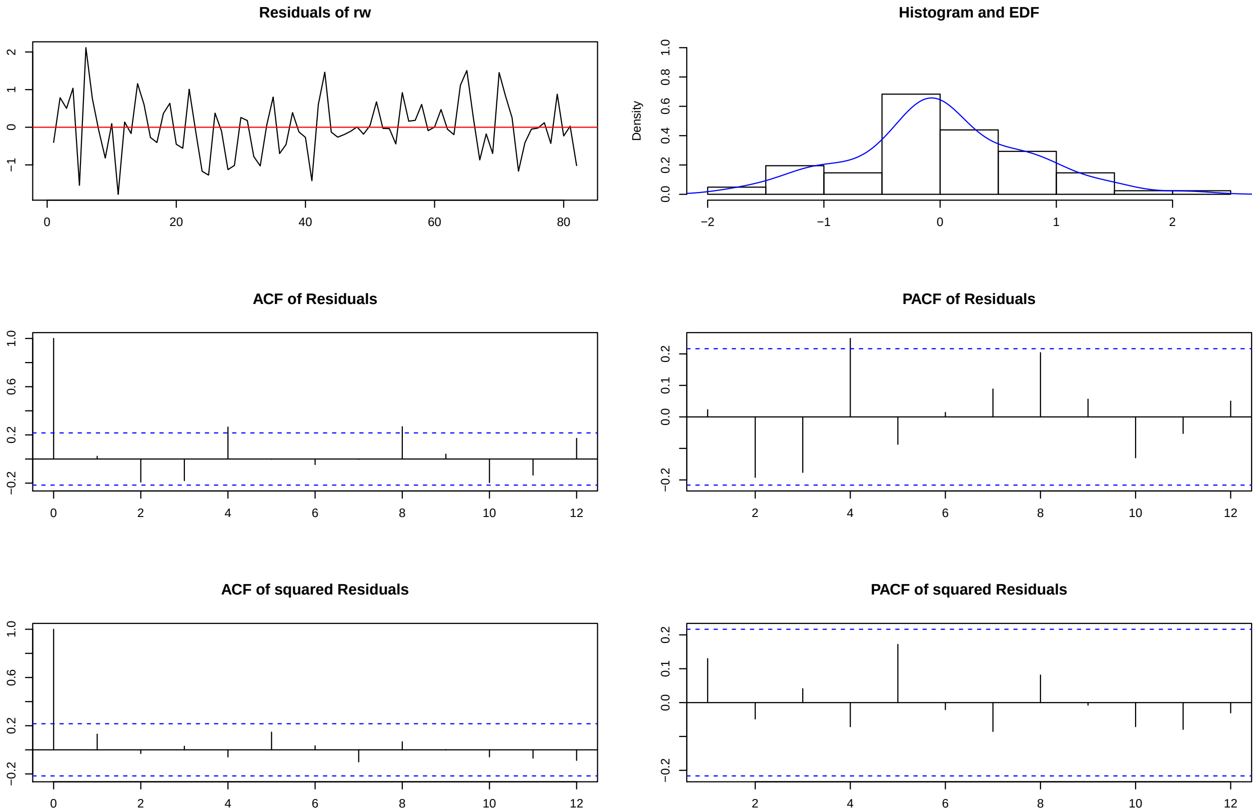
```r
var2c.pt.adj <- serial.test(var2c, lags.pt = 16,
                       type = "PT.adjusted")
var2c.pt.adj
```

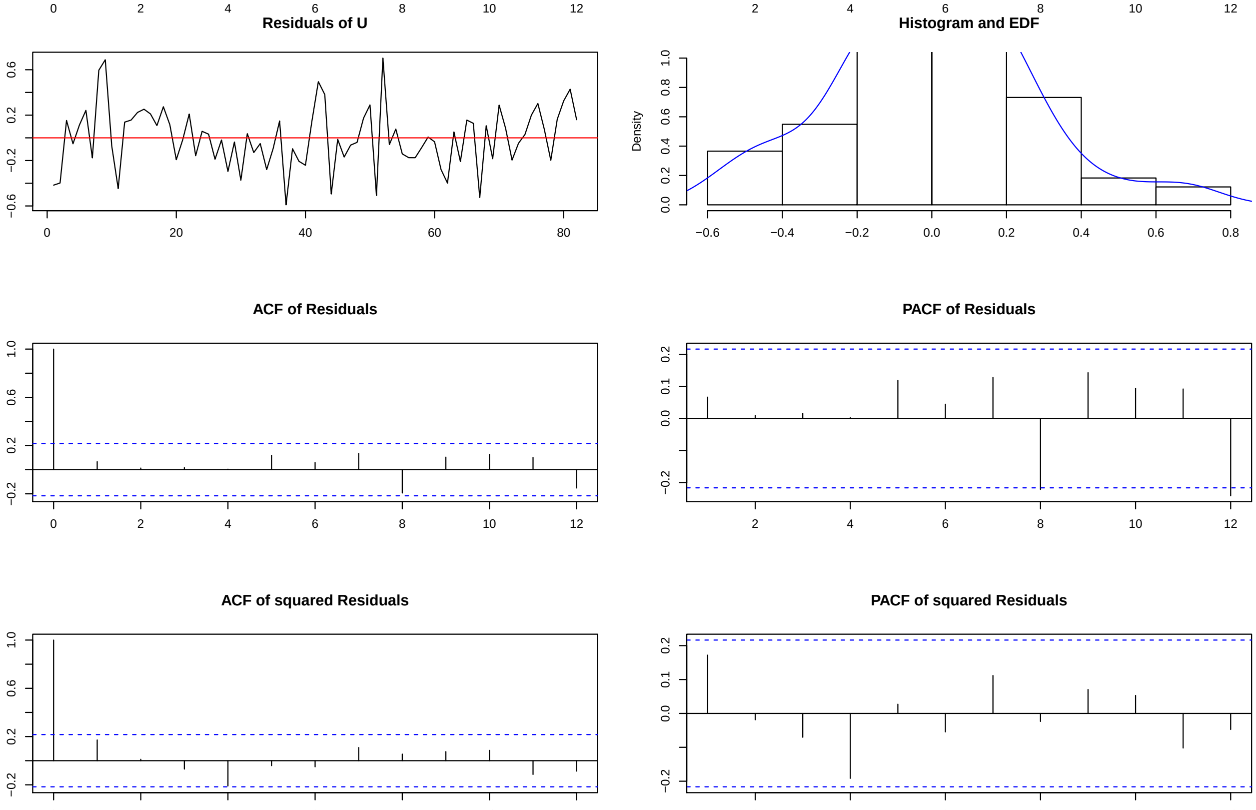
```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var2c
## Chi-squared = 231.59, df = 224, p-value = 0.3497
```

```r
plot(var2c.pt.asy)
```

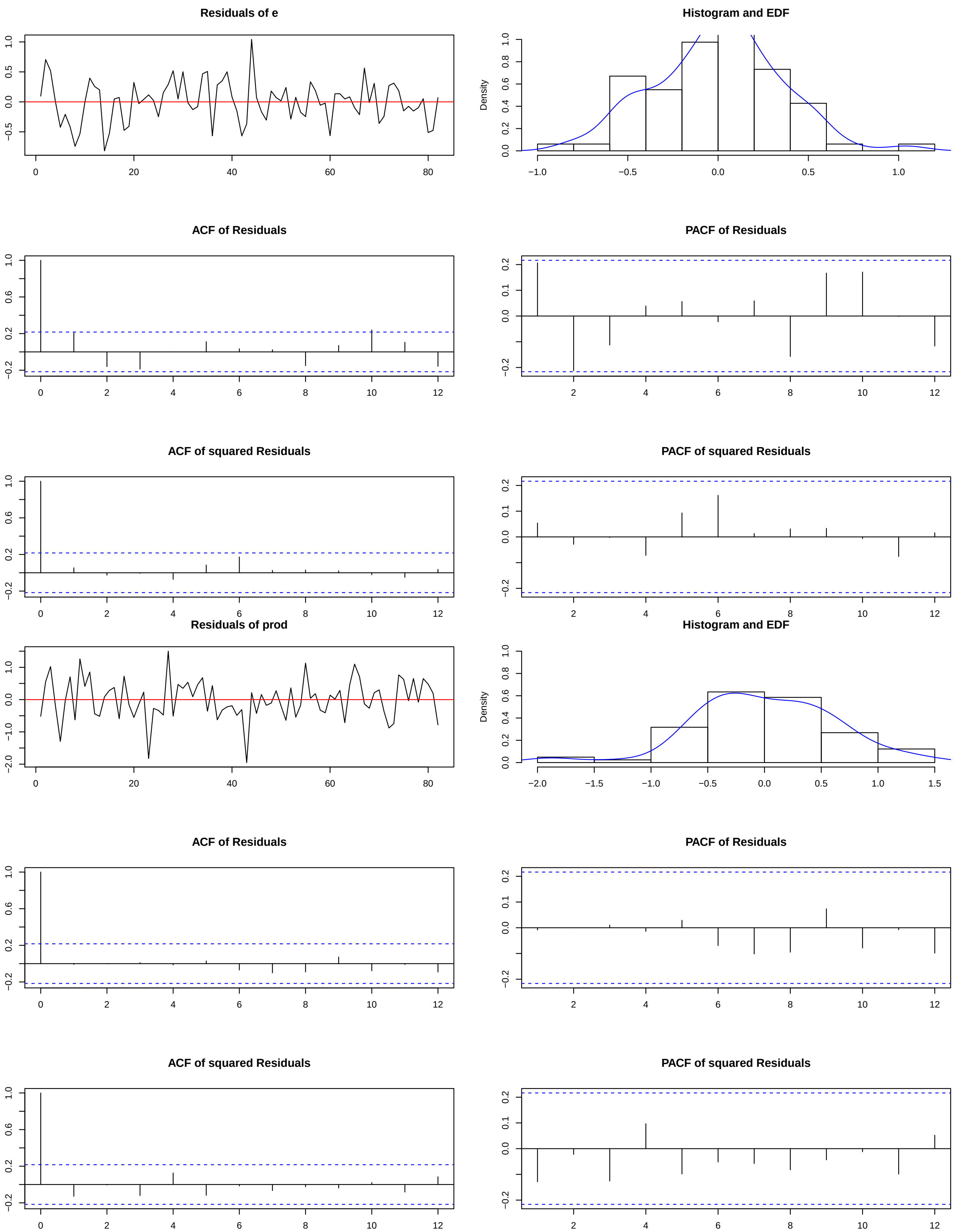

Residuals of e
Histogram and EDF
Density
ACF of Residuals
PACF of Residuals
ACF of squared Residuals
PACF of squared Residuals

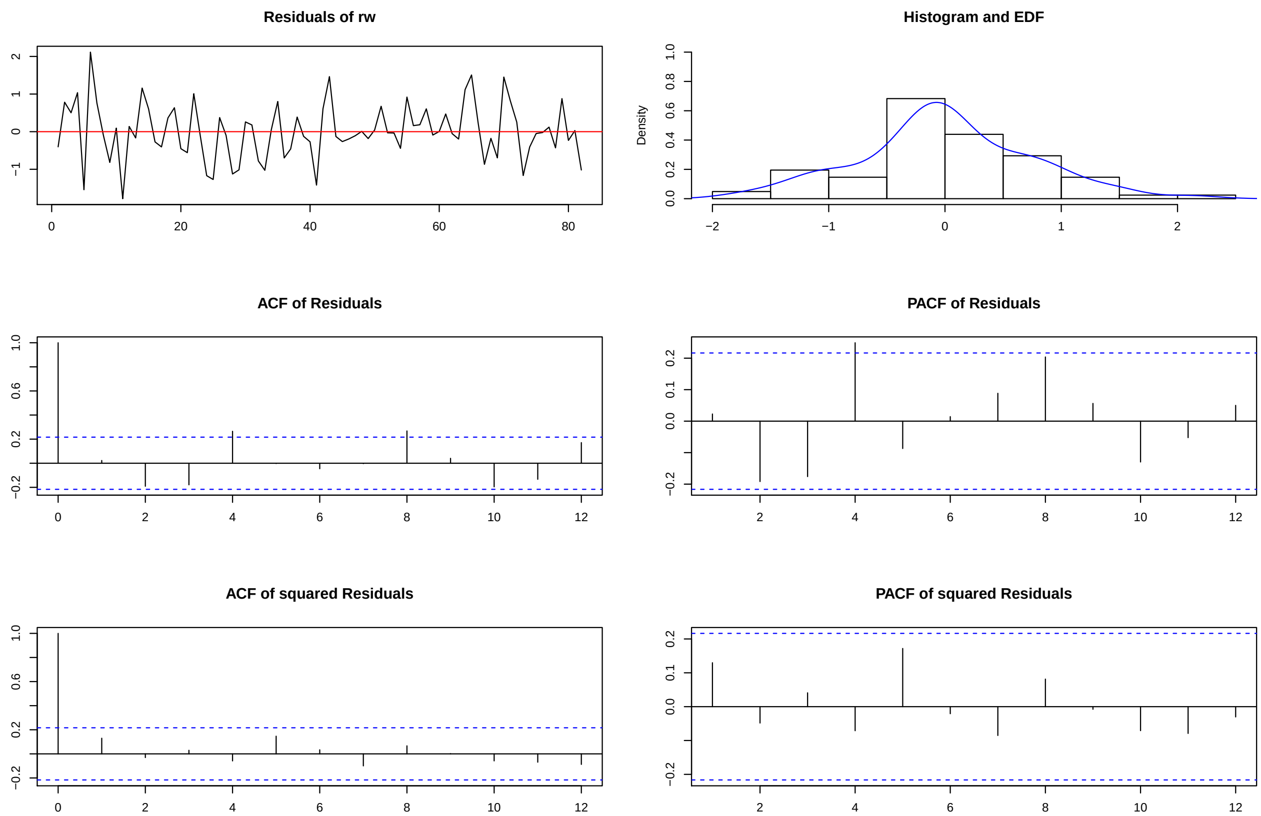

Residuals of prod
Histogram and EDF
Density
ACF of Residuals
PACF of Residuals
ACF of squared Residuals
PACF of squared Residuals

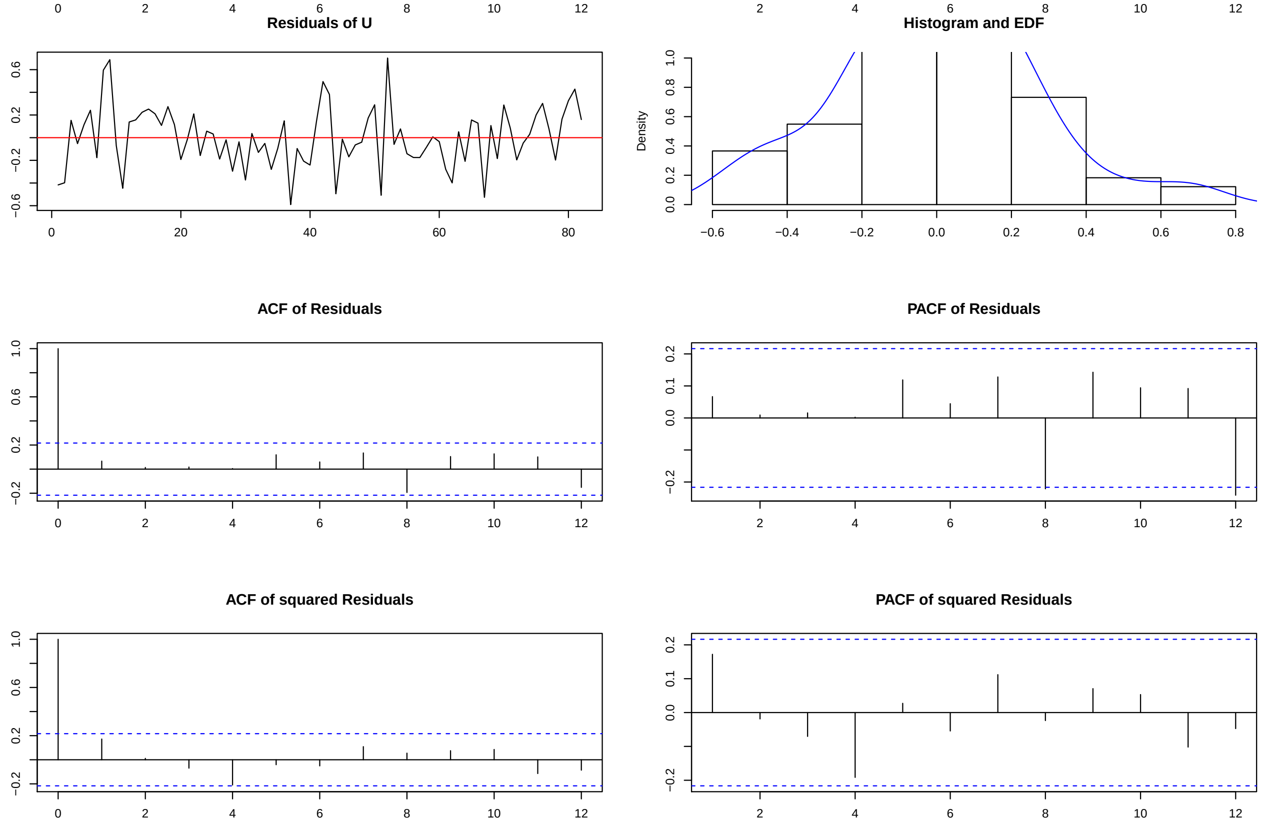

Residuals of rw
Histogram and EDF
Density
ACF of Residuals
PACF of Residuals
ACF of squared Residuals
PACF of squared Residuals


Residuals of U
Histogram and EDF
Density
ACF of Residuals
PACF of Residuals
ACF of squared Residuals
PACF of squared Residuals

```r
plot(var2c.pt.adj)
```








```
var2c.BG <- serial.test(var2c, lags.pt = 16, type = "BG")
var2c.BG
```

```
##
##  Breusch-Godfrey LM test
##
## data:  Residuals of VAR object var2c
## Chi-squared = 92.628, df = 80, p-value = 0.1581
```

```
var2c.ES <- serial.test(var2c, lags.pt = 16, type = "ES")
var2c.ES
```

```
##
##  Edgerton-Shukur F test
##
## data:  Residuals of VAR object var2c
## F statistic = 1.1186, df1 = 80, df2 = 199, p-value = 0.2648
```

```
args(stability)
```

```
## function (x, ...)
## NULL
```

```
var2c.stab <- stability(var2c, type = "OLS-CUSUM")
names(var2c.stab)
```

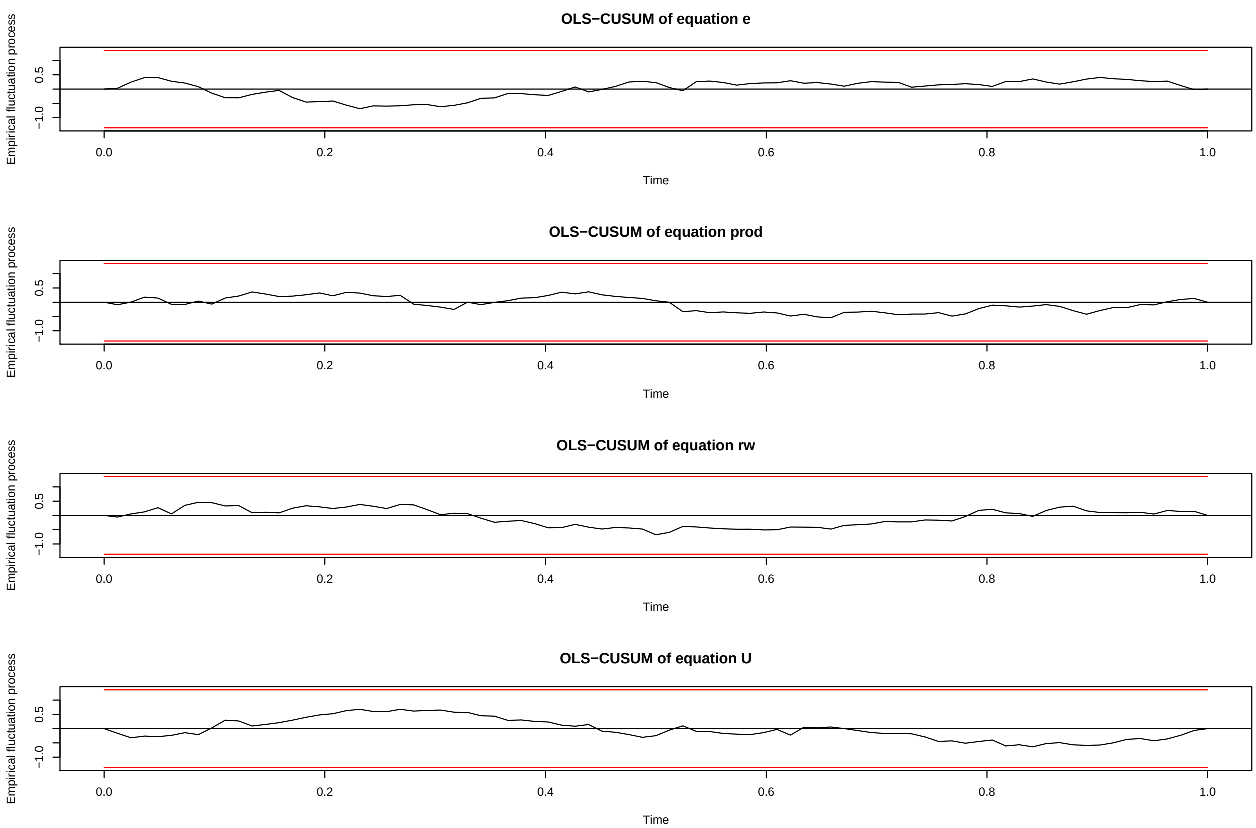
```
## [1] "stability" "names"     "K"
```

```
var2c.stab
```

```
## $e
##
## Empirical Fluctuation Process: OLS-based CUSUM test
##
## Call: efp(formula = formula, data = data, type = type, h = h, dynamic = dynamic,
##     rescale = rescale)
##
##
## $prod
##
## Empirical Fluctuation Process: OLS-based CUSUM test
##
## Call: efp(formula = formula, data = data, type = type, h = h, dynamic = dynamic,
##     rescale = rescale)
##
##
## $rw
##
## Empirical Fluctuation Process: OLS-based CUSUM test
##
## Call: efp(formula = formula, data = data, type = type, h = h, dynamic = dynamic,
##     rescale = rescale)
##
##
```

```
## $U
##
## Empirical Fluctuation Process: OLS-based CUSUM test
##
## Call: efp(formula = formula, data = data, type = type, h = h, dynamic = dynamic,
##     rescale = rescale)
```

```r
plot(var2c.stab)
```



Causality Analysis

Les chercheurs s'intéressent souvent à la détection des liens de causalité entre les variables. Le plus commun est le test de *Granger-Causality* (Granger, 1969). Incidemment, ce test ne convient pas pour vérifier les relations de causalité au sens strict, car la possibilité d'une erreur *post hoc ergo propter hoc* ne peut être exclue. Cela est vrai pour tout prétendu *test de causalité* en économétrie. Il est donc courant de dire que la variable x ne cause pas la variable cause au sens de granger y si la variable x permet de prédire la variable y. Outre ce test, dans la fonction *causality()*, un test de causalité instantanée de type *Wald* est également implémenté. Il est caractérisé par des tests de corrélation non nulle entre les processus d'erreur des variables de cause à effet.

```r
args(causality)
```

```
## function (x, cause = NULL, vcov. = NULL, boot = FALSE, boot.runs = 100)
## NULL
```

La fonction *causality()* a deux arguments importants. Le premier argument, x, est un objet de la classe *varest* et le second, *cause*, est un vecteur de caractères des noms de variable, supposés être causaux pour les variables restantes d'un processus *VAR(p)*.

La fonction *causality()* est maintenant utilisée pour rechercher si le salaire réel (rw) et la productivité (prod) sont des causes de l'emploi (e) et du chômage (U).

```
causality(var2c, cause = c("rw", "prod"))
```

```
## $Granger
##
##  Granger causality H0: prod rw do not Granger-cause e U
##
## data:  VAR object var2c
## F-Test = 3.4529, df1 = 8, df2 = 292, p-value = 0.0008086
##
##
## $Instant
##
##  H0: No instantaneous causality between: prod rw and e U
##
## data:  VAR object var2c
## Chi-squared = 2.5822, df = 4, p-value = 0.63
```

L'hypothèse nulle d'absence de causalité de Granger du salaire réel et de la productivité du travail à l'emploi et au chômage doit être rejetée; alors que l'hypothèse nulle d'une causalité non instantanée ne peut être rejetée. Ce résultat de test est économiquement plausible, compte tenu des frictions observées sur les marchés du travail.

Prévision (Forecasting)

Une méthode de prédiction pour les objets avec l'attribut de classe *varest* est disponible. Les prévisions *n.ahead* sont calculées de manière récursive pour le VAR estimé, en commençant par $h = 1, 2, ..., n.ahead$:

$$Y_{T+1|T} = A_1 Y_T + ... A_p Y_{T+1-p} + CD_{T+1}$$

```
var.f10 = predict(var2c, n.ahead = 10, ci = 0.95)
names(var.f10)
```

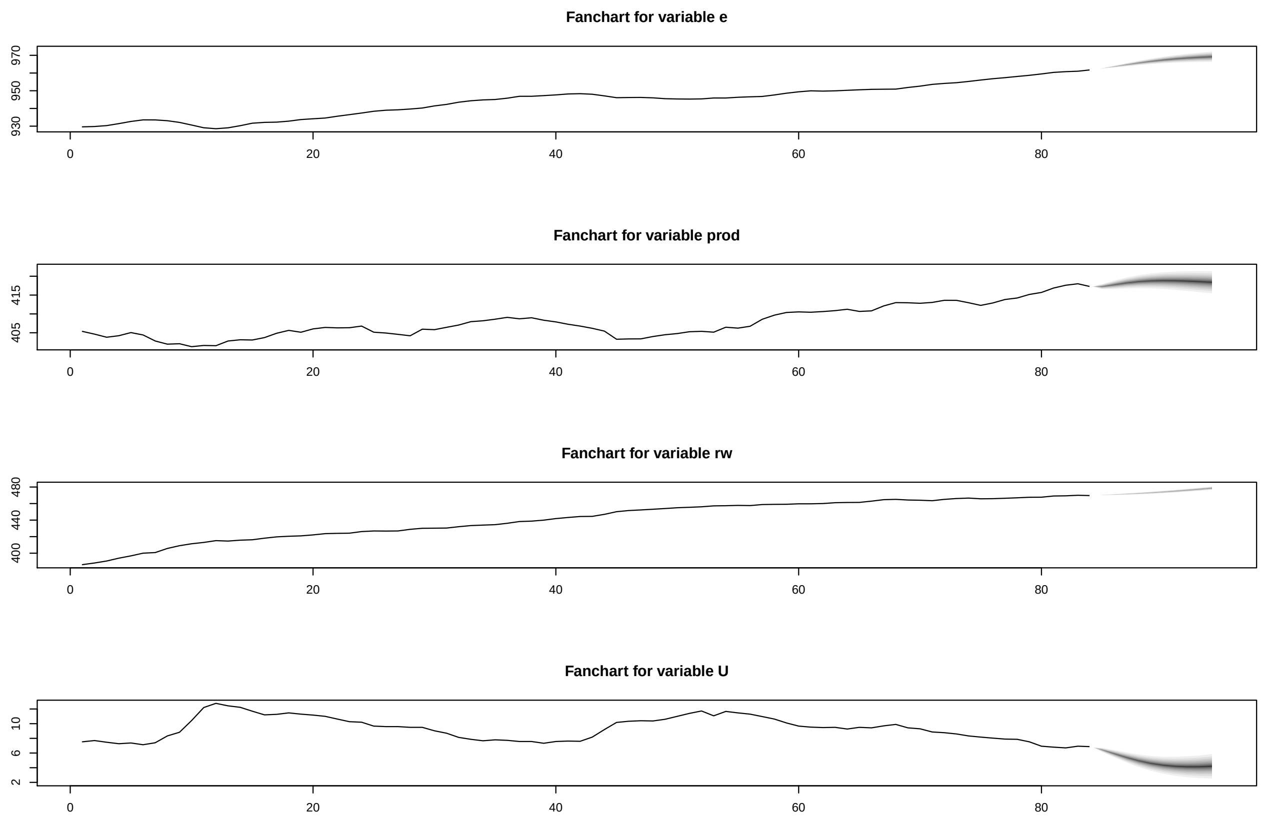
```
## [1] "fcst"     "endog"    "model"    "exo.fcst"
```

Outre les arguments de la fonction pour l'objet *varest* et les étapes de prévision *n.ahead*, une valeur pour l'intervalle de confiance de prévision peut également être fournie. Sa valeur par défaut est *0.95*.

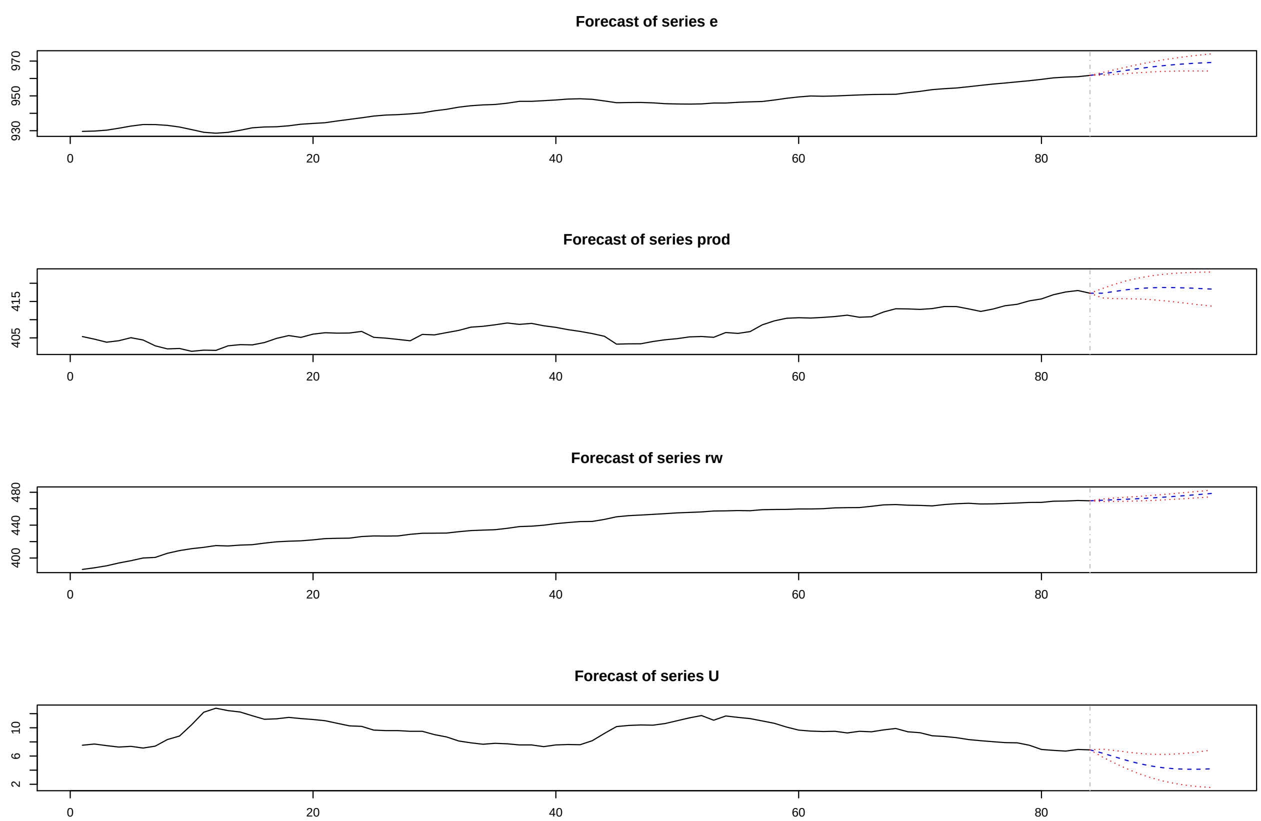
```
class(var.f10)
```

```
## [1] "varprd"
```

```
fanchart(var.f10)
```




```r
plot(var.f10)
```



```r
args(fanchart)
```

```
## function (x, colors = NULL, cis = NULL, names = NULL, main = NULL,
##     ylab = NULL, xlab = NULL, col.y = NULL, nc, plot.type = c("multiple",
##         "single"), mar = par("mar"), oma = par("oma"), ...)
## NULL
```

Impulse response analysis (Analyse de la réponse impulsionnelle)

L'analyse de réponse impulsionnelle est basée sur la représentation de moyenne mobile d'un processus $VAR(p)$. Il est utilisé pour étudier les interactions dynamiques entre les variables endogènes.

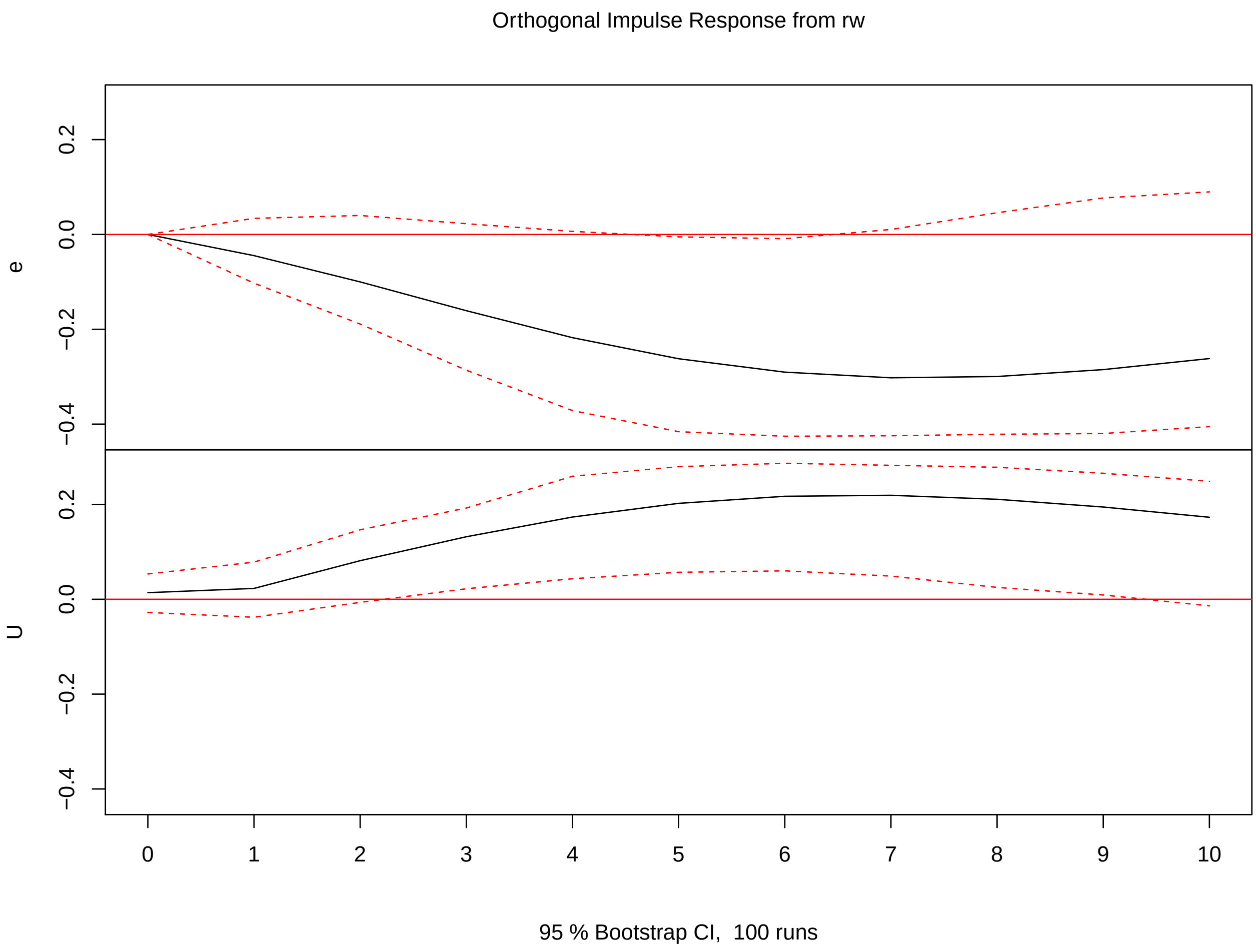
```r
args(irf)
```

```
## function (x, impulse = NULL, response = NULL, n.ahead = 10, ortho = TRUE,
##     cumulative = FALSE, boot = TRUE, ci = 0.95, runs = 100, seed = NULL,
##     ...)
## NULL
```

```r
var.2c.alt <- VAR(Canada, p = 2, type = "const")
irf.rw.eU <- irf(var.2c.alt, impulse = "rw", response = c("e", "U"), boot = TRUE)
names(irf.rw.eU)
```

```
##  [1] "irf"        "Lower"      "Upper"      "response"   "impulse"
##  [6] "ortho"      "cumulative" "runs"       "ci"         "boot"
## [11] "model"
```

```r
plot(irf.rw.eU)
```



Réponses impulsionnelles orthogonales du salaire réel à l'emploi et au chômage (avec un interval de confiance à 95%, avec 100 répétitions).

Décomposition de la variance d'erreur prévue

La décomposition de la variance d'erreur de prévision (dorénavant: FEVD) est basée sur les matrices de coefficient de réponse impulsionnelle orthogonalisées.

```
args(fevd)
```

```
## function (x, n.ahead = 10, ...)
## NULL
```

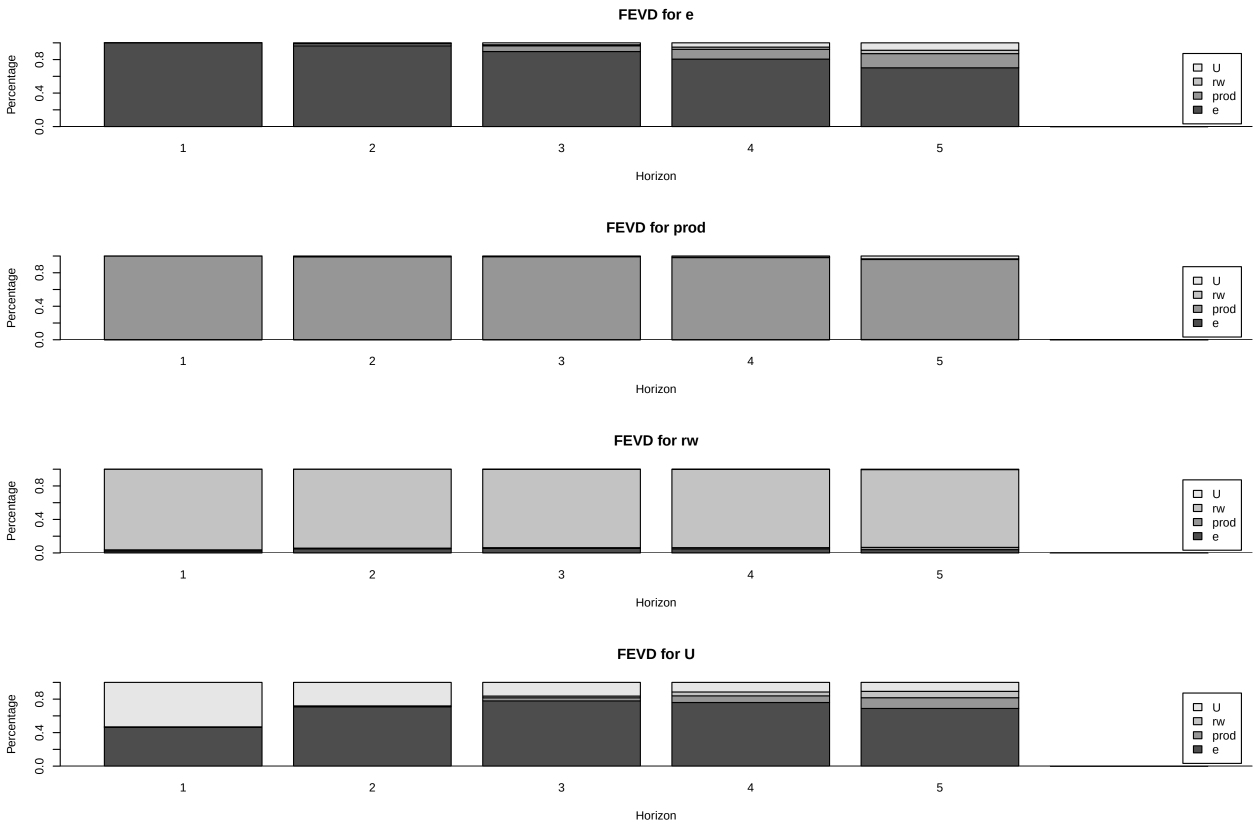
```
var2c.fevd <- fevd(var2c, n.ahead = 5)
class(var2c.fevd)
```

```
## [1] "varfevd"
```

```
names(var2c.fevd)
```

```
## [1] "e"    "prod" "rw"   "U"
```

```r
plot(var2c.fevd)
```



2. SVAR: Structural vector autoregressive models

Estimation

Nous aurons besoin ici de la fonction *SVAR*.

```r
args(SVAR)
```

```
## function (x, estmethod = c("scoring", "direct"), Amat = NULL,
##     Bmat = NULL, start = NULL, max.iter = 100, conv.crit = 1e-07,
##     maxls = 1, lrtest = TRUE, ...)
## NULL
```

```r
amor <- diag(4)
diag(amor) = NA
amor[1, 2] = NA
amor[1, 3] = NA
amor[3, 2] = NA
amor[4, 1] = NA
amor
```

```
##      [,1] [,2] [,3] [,4]
```

```
## [1,]   NA   NA   NA    0
## [2,]    0   NA    0    0
## [3,]    0   NA   NA    0
## [4,]   NA    0    0   NA
```

Les paramètres sont estimés en minimisant le négatif de la fonction de log-vraisemblance concentrée :

$$lnL_c = -\frac{KT}{2}ln(2\pi) + \frac{T}{2}ln|A|^2 - \frac{T}{2}ln|B|^2 - \frac{T}{2}tr(A'B'^{-1}B^{-1}A\sum_{\mu})$$

Sur R on utilise l'argument *optim* :

```
args(optim)
```

```
## function (par, fn, gr = NULL, ..., method = c("Nelder-Mead",
##     "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"), lower = -Inf,
##     upper = Inf, control = list(), hessian = FALSE)
## NULL
```

```
svar2c.A = SVAR(var2c, estmethod = "scoring", Amat = amor, Bmat = NULL,
                hessian = TRUE, method = "BFGS")
svar2c.A
```

```
##
## SVAR Estimation Results:
## ========================
##
##
## Estimated A matrix:
##          e     prod     rw     U
## e    2.787  0.01997 0.1906 0.000
## prod 0.000  1.53265 0.0000 0.000
## rw   0.000 -0.19610 1.2920 0.000
## U    2.562  0.00000 0.0000 4.882
```

On peut aussi estimer par la méthode *direct* :

```
svar2c.A2 = SVAR(var2c, estmethod = "direct", Amat = amor, Bmat = NULL,
                hessian = TRUE, method = "BFGS")
svar2c.A2
```

```
##
## SVAR Estimation Results:
## ========================
##
##
## Estimated A matrix:
##          e     prod     rw     U
## e    2.788  0.01999 0.1906 0.000
## prod 0.000  1.53260 0.0000 0.000
## rw   0.000 -0.19609 1.2920 0.000
## U    2.563  0.00000 0.0000 4.883
```

```r
class(svar2c.A)
```

```
## [1] "svarest"
```

```r
names(svar2c.A)
```

```
##  [1] "A"       "Ase"     "B"       "Bse"     "LRIM"    "Sigma.U" "LR"
##  [8] "opt"     "start"   "type"    "var"     "iter"    "call"
```

Le modèle de Blanchard & Quah est implémenté sur R sous la fonction $BQ()$. Il a un argument x, qui est un objet avec l'attribut de classe *varest*. La fonction renvoie un objet de liste avec l'attribut de classe *svarest*. Vous trouverez ci-dessous un exemple trivial d'application d'un SVAR de type Blanchard & Quah (1989) à la *var2c*:

```r
BQ(var2c)
```

```
##
## SVAR Estimation Results:
## ========================
##
##
## Estimated contemporaneous impact matrix:
##             e     prod       rw        U
## e    -0.007644 -0.28470  0.07374 -0.21234
## prod  0.543663  0.21658 -0.03379 -0.28652
## rw    0.082112  0.28588  0.71874  0.06162
## U     0.129451  0.05668 -0.01039  0.24111
##
## Estimated identified long run impact matrix:
##           e    prod    rw      U
## e    104.37  0.0000  0.00 0.0000
## prod  45.35  5.1971  0.00 0.0000
## rw   168.41 -2.1145 10.72 0.0000
## U    -19.26 -0.4562  1.41 0.5331
```

La matrice d'impact simultanée est stockée en tant qu'élément de liste B et la matrice d'impact à long terme en tant qu'élément de liste LRIM dans l'objet renvoyé.

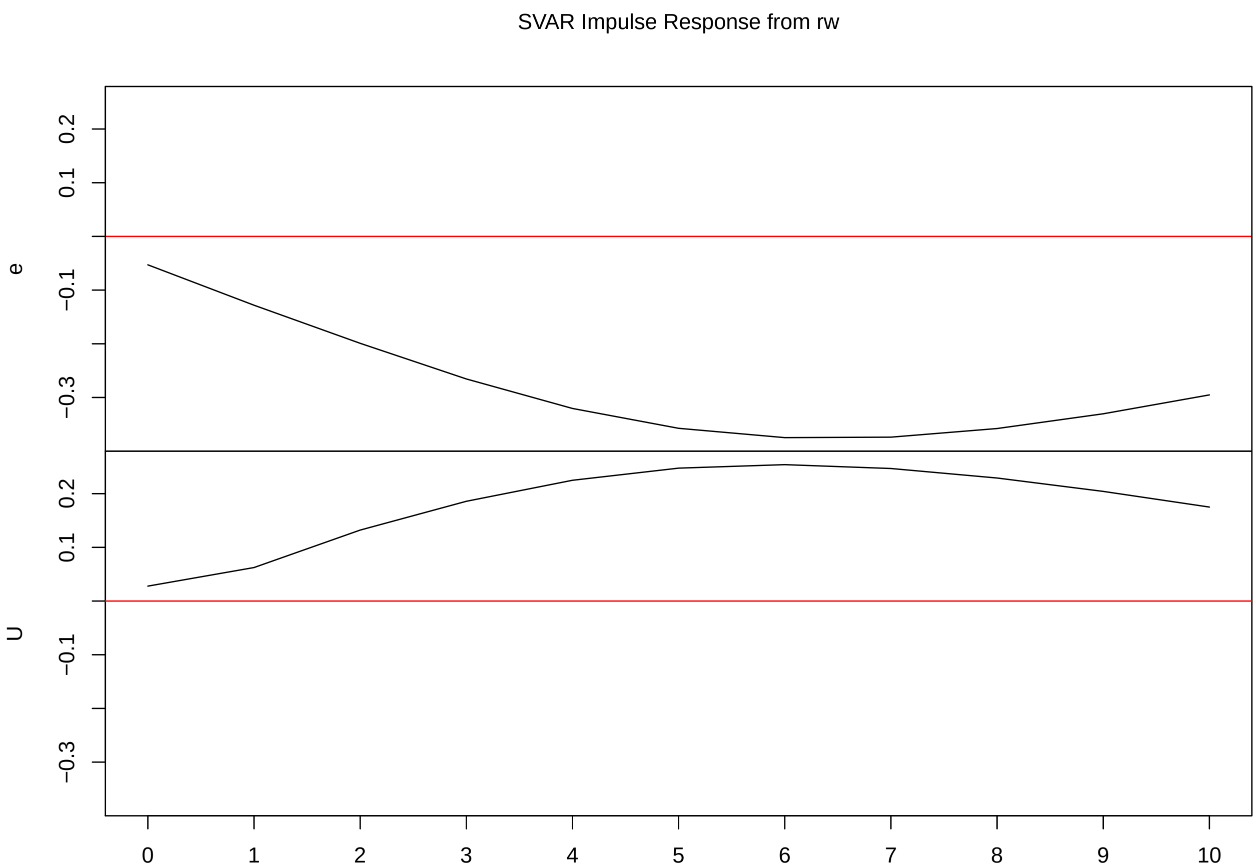
Impulse response analysis

```r
svar2cA.ira <- irf(svar2c.A, impulse = "rw", response = c("e", "U"), boot = FALSE)
svar2cA.ira
```

```
##
## Impulse response coefficients
## $rw
##                 e          U
##  [1,] -0.05292759 0.02777855
##  [2,] -0.12816116 0.06237565
```

```
##  [3,] -0.19908314 0.13215181
##  [4,] -0.26547817 0.18579008
##  [5,] -0.32043171 0.22492367
##  [6,] -0.35744006 0.24754066
##  [7,] -0.37478948 0.25405349
##  [8,] -0.37389527 0.24686436
##  [9,] -0.35784342 0.22915830
## [10,] -0.33037072 0.20422398
## [11,] -0.29519964 0.17505146
```

```r
plot(svar2cA.ira)
```



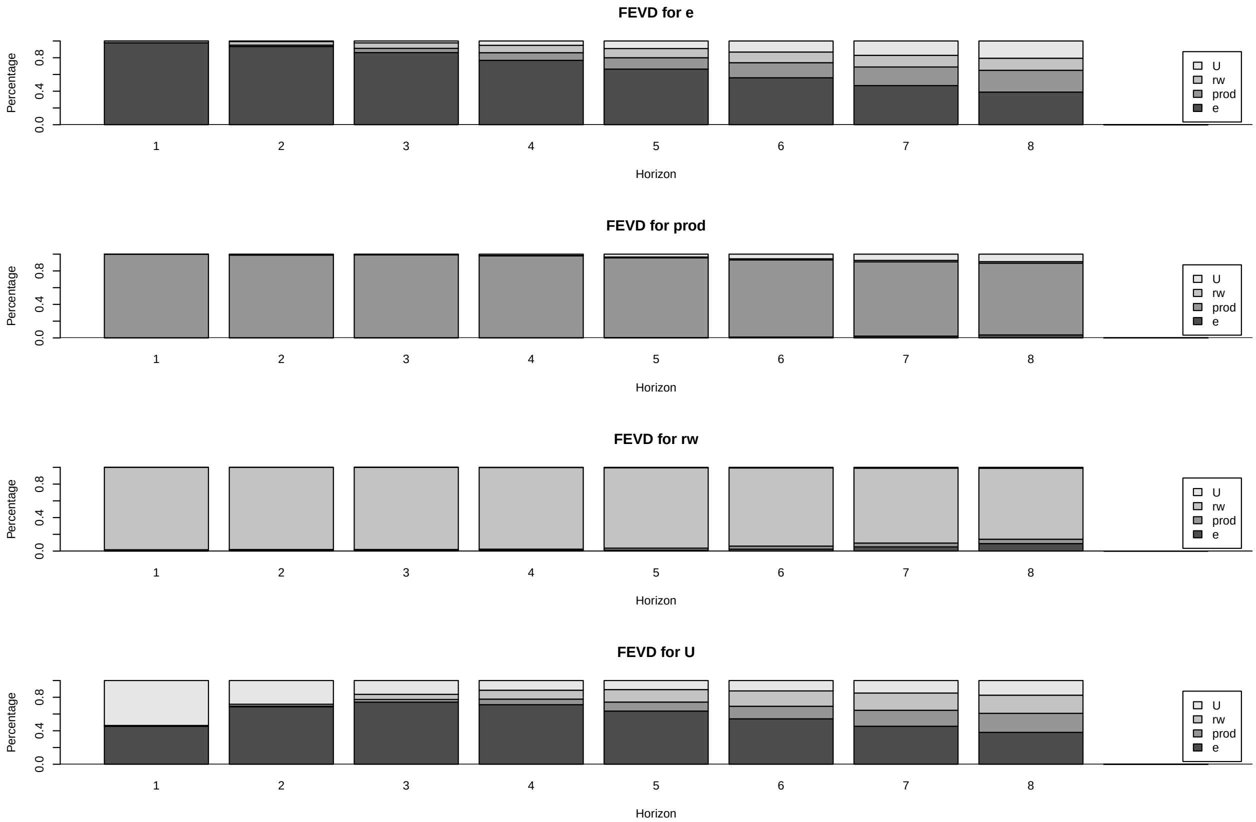
Forecast error variance decomposition (Décomposition de la variance d'erreur prévue)

```r
svar2cA.fevd = fevd(svar2c.A, n.ahead = 8)
svar2cA.fevd
```

```
## $e
##              e         prod         rw           U
## [1,] 0.9777235 0.0009954282 0.02128108 0.000000000
## [2,] 0.9336814 0.0167246253 0.04297944 0.006614517
## [3,] 0.8619288 0.0500730692 0.06506141 0.022936768
```

```
## [4,] 0.7687697 0.0911061886 0.08828845 0.051835690
## [5,] 0.6644015 0.1353599400 0.11009035 0.090148250
## [6,] 0.5609862 0.1796029449 0.12745193 0.131958965
## [7,] 0.4678657 0.2215432264 0.13883208 0.171758991
## [8,] 0.3899084 0.2598661398 0.14424591 0.205979578
##
## $prod
##                 e      prod           rw           U
## [1,] 0.0000000000 1.0000000 0.0000000000 0.000000000
## [2,] 0.0007849282 0.9884220 0.0012556396 0.009537435
## [3,] 0.0013622579 0.9911861 0.0009396943 0.006511990
## [4,] 0.0010584313 0.9791939 0.0042079409 0.015539736
## [5,] 0.0033137858 0.9530684 0.0095487984 0.034068983
## [6,] 0.0101073089 0.9203052 0.0142980234 0.055289505
## [7,] 0.0212372885 0.8872371 0.0172159553 0.074309703
## [8,] 0.0352836374 0.8574172 0.0182751514 0.089024054
##
## $rw
##                e        prod        rw            U
## [1,] 0.000000000 0.016106922 0.9838931 0.0000000000
## [2,] 0.008695900 0.010101636 0.9811970 0.0000055067
## [3,] 0.010417813 0.008836712 0.9802130 0.0005324700
## [4,] 0.008203103 0.014845109 0.9748297 0.0021220772
## [5,] 0.010273263 0.025147118 0.9597723 0.0048072841
## [6,] 0.023338212 0.036326587 0.9326378 0.0076973774
## [7,] 0.050131223 0.045769592 0.8944162 0.0096829955
## [8,] 0.089437915 0.052012323 0.8483264 0.0102233872
##
## $U
##              e         prod          rw         U
## [1,] 0.4532932 0.0004615015 0.009866359 0.5363789
## [2,] 0.6870704 0.0074941451 0.022718445 0.2827170
## [3,] 0.7424808 0.0313639523 0.061213946 0.1649413
## [4,] 0.7117914 0.0662709011 0.106115156 0.1158225
## [5,] 0.6353394 0.1073181982 0.149107946 0.1082345
## [6,] 0.5425133 0.1499584669 0.183443049 0.1240851
## [7,] 0.4540476 0.1906359927 0.205905269 0.1494112
## [8,] 0.3808673 0.2272736449 0.216857682 0.1750013
```

```r
plot(svar2cA.fevd)
```



3. VECM to VAR (VECM à VAR)

```r
library(urca)
library(timeSeries)

data("finland")

hen = finland
```

```r
args(ca.jo)
```

```
## function (x, type = c("eigen", "trace"), ecdet = c("none", "const",
##     "trend"), K = 2, spec = c("longrun", "transitory"), season = NULL,
##     dumvar = NULL)
## NULL
```

```r
hen.vecm = ca.jo(hen, type = "eigen", ecdet = "const", K = 2, spec = "longrun",
                 season = 4)
summary(hen.vecm)
```

```
##
## ######################
## # Johansen-Procedure #
```

```
## ######################
##
## Test type: maximal eigenvalue statistic (lambda max) , without linear trend and constant in cointegra
##
## Eigenvalues (lambda):
## [1]  3.922735e-01  2.465575e-01  1.258139e-01  7.304446e-02 -6.179556e-16
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 3 |  7.89  7.52  9.24 12.97
## r <= 2 | 13.98 13.75 15.67 20.20
## r <= 1 | 29.44 19.77 22.00 26.81
## r = 0  | 51.80 25.56 28.14 33.24
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##              lrm1.l2    lny.l2     lnmr.l2    difp.l2   constant
## lrm1.l2    1.0000000  1.000000       1.000  1.0000000  1.0000000
## lny.l2    -0.9300137 -1.180848    4611.559 -0.9285116  1.0076173
## lnmr.l2  -12.6865176 -3.312985  -11702.809  0.2838535  0.5013506
## difp.l2  -34.0243950 14.192182 -140993.071 -1.6885065 -0.8357096
## constant   3.8398216  2.061620  -17832.599  0.7994953 -7.7424043
##
## Weights W:
## (This is the loading matrix)
##
##            lrm1.l2       lny.l2       lnmr.l2      difp.l2      constant
## lrm1.d 0.018027124 -0.008910894 -7.413308e-07 -0.128563428 -2.872450e-16
## lny.d  0.016855021 -0.009134709 -3.661279e-06  0.015794409 -6.718564e-17
## lnmr.d 0.012164112  0.091498395  1.625688e-06 -0.008477157  7.469079e-16
## difp.d 0.001893935 -0.011243307  1.681968e-06  0.013640791 -7.263298e-18
```

```r
args(vec2var)
```

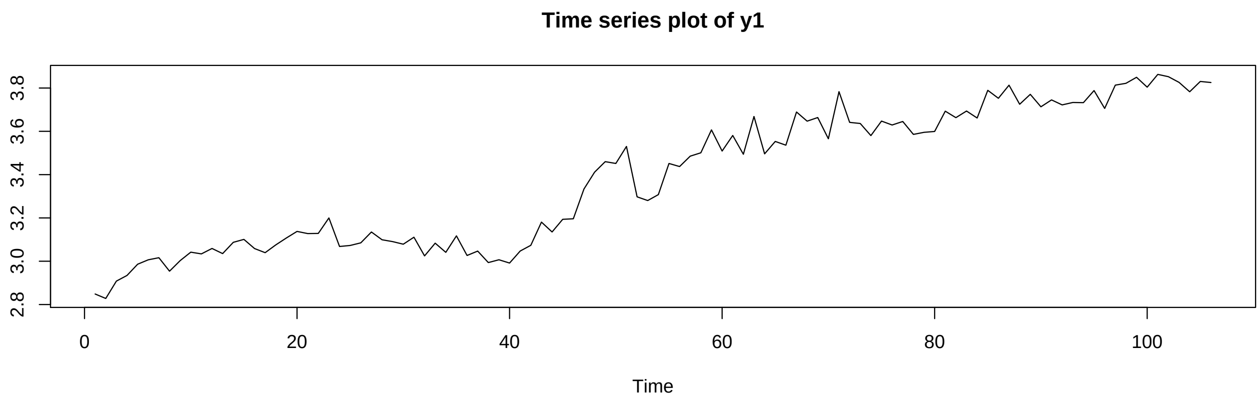
```
## function (z, r = 1)
## NULL
```

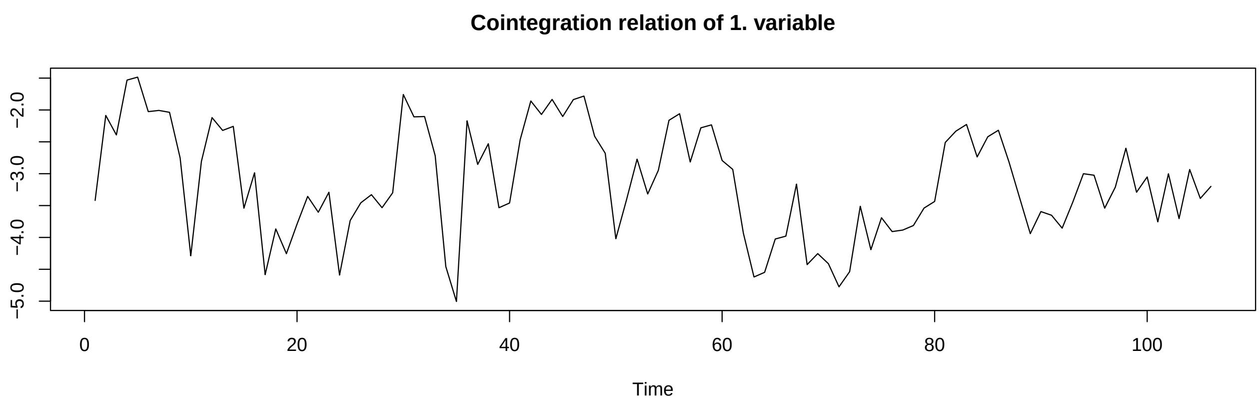
```r
hen.var = vec2var(hen.vecm, r = 2)
print(hen.var)
```

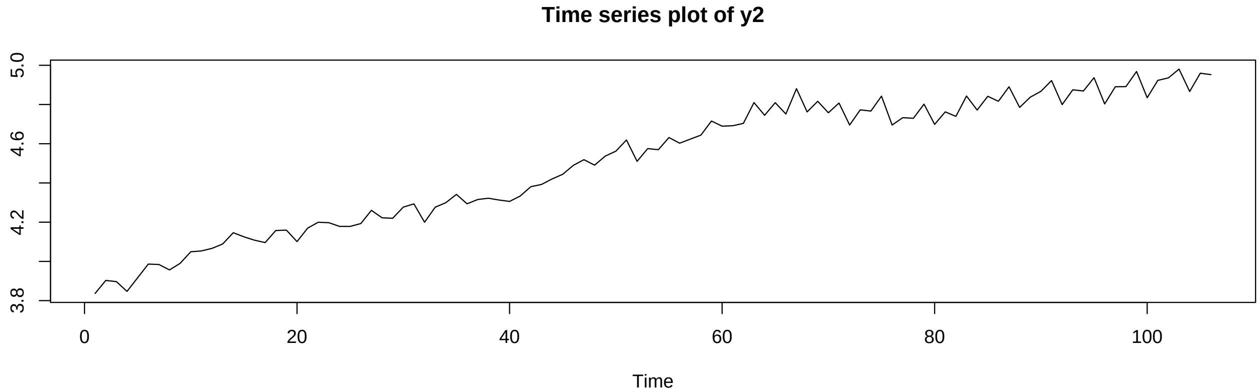
```
##
## Coefficient matrix of lagged endogenous variables:
##
## A1:
##           lrm1.l1      lny.l1     lnmr.l1    difp.l1
## lrm1  0.852554992 -0.28278842 -0.09205914 -0.1969425
## lny   0.023855041  0.34766934 -0.07437779 -0.4020086
## lnmr -0.151091223 -0.01945451  0.77246215  0.5220277
## difp  0.007163811  0.01836939  0.02825459  0.3116453
##
##
```

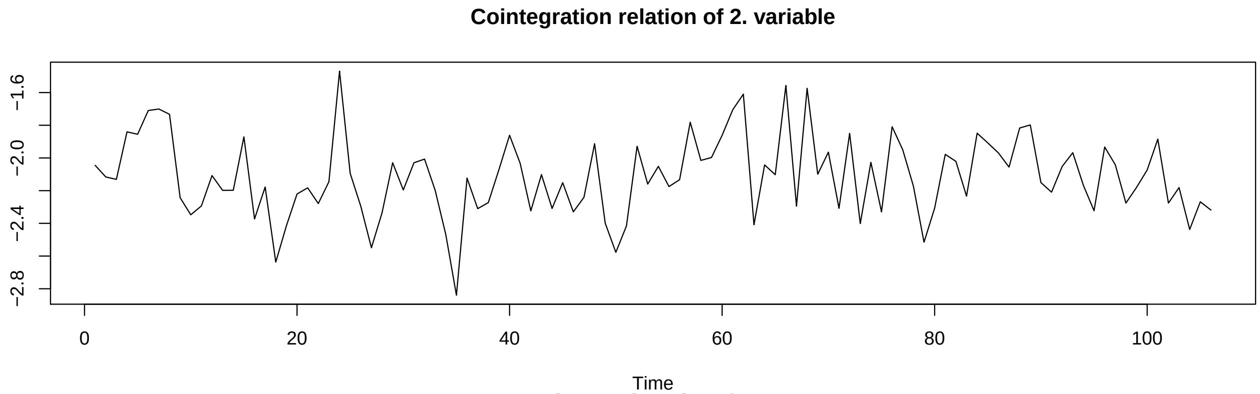
```
## A2:
##          lrm1.l2       lny.l2     lnmr.l2    difp.l2
## lrm1  0.15656124  0.276545367 -0.10712062 -0.5428845
## lny  -0.01613473  0.647441966 -0.10919058 -0.3011147
## lnmr  0.25475373 -0.099903996 -0.22991519  0.3626576
## difp -0.01651318 -0.006854137 -0.01503313  0.4643476
##
##
## Coefficient matrix of deterministic regressor(s).
##
##         constant           sd1          sd2          sd3
## lrm1  0.05085006 0.0397880457  0.037369011  0.100913516
## lny   0.04588797 0.0455778987  0.084628551  0.096385625
## lnmr  0.23534299 0.0078896488  0.011571987  0.019700923
## difp -0.01590706 0.0007442968 -0.008486073 -0.008839663
```

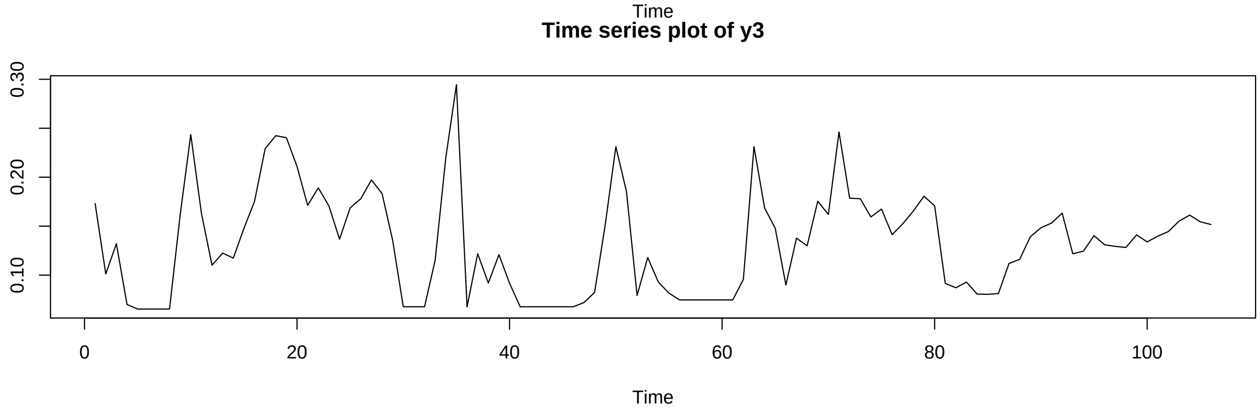
```r
plot(hen.vecm)
```

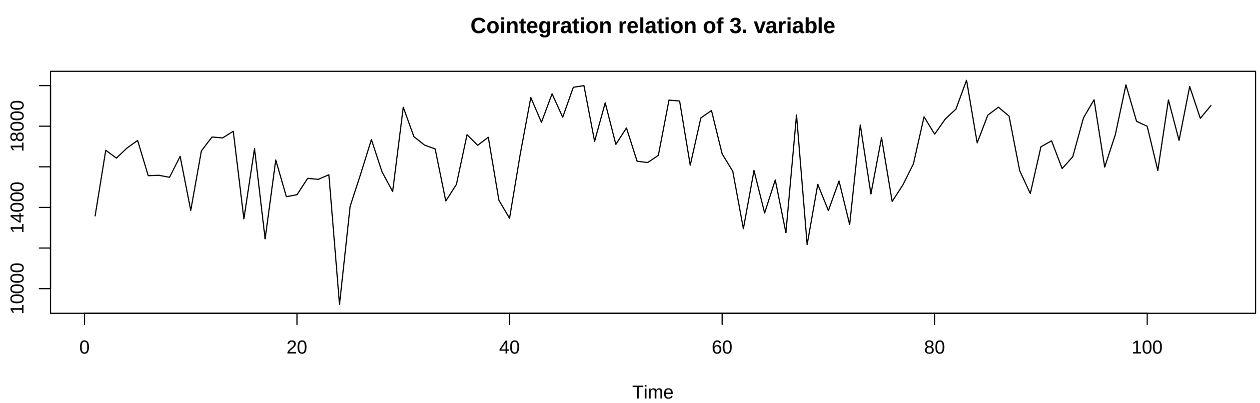


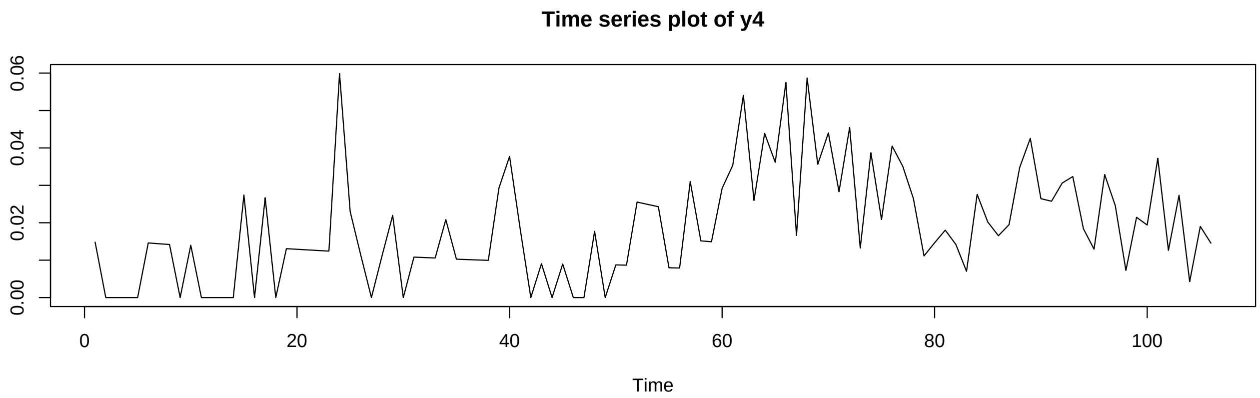


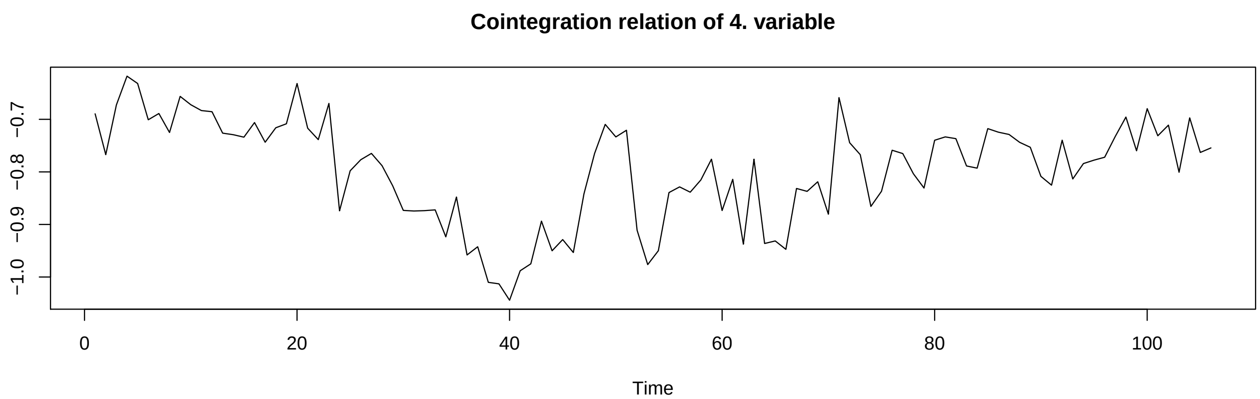

Time series plot of y2
3.8
4.2
4.6
5.0
0
20
40
60
80
100
Time


Cointegration relation of 2. variable
−2.8
−2.4
−2.0
−1.6
0
20
40
60
80
100
Time


Time series plot of y3
0.10
0.20
0.30
0
20
40
60
80
100
Time


Cointegration relation of 3. variable
10000
14000
18000
0
20
40
60
80
100
Time






```
vecm = ca.jo(Canada[, c("prod", "e", "U", "rw")], type = "trace", ecdet = "trend",
             K = 3, spec = "transitory")
SR = matrix(NA, nrow = 4, ncol = 4)
SR[4, 2] = 0
SR
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   NA   NA   NA   NA
## [2,]   NA   NA   NA   NA
## [3,]   NA   NA   NA   NA
## [4,]   NA    0   NA   NA
```

```
LR = matrix(NA, nrow = 4, ncol = 4)
LR[1, 2:4] = 0
LR[2:4, 4] = 0
LR
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   NA    0    0    0
## [2,]   NA   NA   NA    0
## [3,]   NA   NA   NA    0
## [4,]   NA   NA   NA    0
```

```
svec = SVEC(vecm, LR = LR, SR = SR, r = 1, lrtest = FALSE, boot = TRUE, runs = 100)
svec
```

```
##
## SVEC Estimation Results:
## ========================
##
##
## Estimated contemporaneous impact matrix:
##          prod        e         U      rw
## prod  0.58402  0.07434 -0.152578 0.06900
## e    -0.12029  0.26144 -0.155096 0.08978
## U     0.02526 -0.26720  0.005488 0.04982
## rw    0.11170  0.00000  0.483771 0.48791
##
## Estimated long run impact matrix:
##         prod       e       U rw
## prod  0.7910  0.0000  0.0000  0
## e     0.2024  0.5769 -0.4923  0
## U    -0.1592 -0.3409  0.1408  0
## rw   -0.1535  0.5961 -0.2495  0
```

```r
svec.irf = irf(svec, response = "U", n.ahead = 48, boot = TRUE)
svec.irf
```

```
##
## Impulse response coefficients
## $prod
##                  U
##  [1,]  0.025256953
##  [2,]  0.014925222
##  [3,] -0.005309212
##  [4,] -0.074625104
##  [5,] -0.151124187
##  [6,] -0.189055852
##  [7,] -0.201781154
##  [8,] -0.205710310
##  [9,] -0.202799768
## [10,] -0.195592001
## [11,] -0.188131379
## [12,] -0.182010221
## [13,] -0.177439496
## [14,] -0.174329599
## [15,] -0.172247391
## [16,] -0.170750317
## [17,] -0.169608943
## [18,] -0.168669251
## [19,] -0.167806583
## [20,] -0.166976400
## [21,] -0.166187224
## [22,] -0.165450113
## [23,] -0.164770717
## [24,] -0.164154312
## [25,] -0.163602950
## [26,] -0.163113794
## [27,] -0.162681284
## [28,] -0.162299061
```

```
## [29,] -0.161960852
## [30,] -0.161660995
## [31,] -0.161394620
## [32,] -0.161157599
## [33,] -0.160946471
## [34,] -0.160758324
## [35,] -0.160590650
## [36,] -0.160441247
## [37,] -0.160308158
## [38,] -0.160189632
## [39,] -0.160084098
## [40,] -0.159990144
## [41,] -0.159906503
## [42,] -0.159832045
## [43,] -0.159765758
## [44,] -0.159706746
## [45,] -0.159654207
## [46,] -0.159607430
## [47,] -0.159565783
## [48,] -0.159528702
## [49,] -0.159495688
##
## $e
##                U
##  [1,] -0.2671973
##  [2,] -0.3918931
##  [3,] -0.4828716
##  [4,] -0.5543595
##  [5,] -0.5670182
##  [6,] -0.5473367
##  [7,] -0.5208727
##  [8,] -0.4924546
##  [9,] -0.4665115
## [10,] -0.4463974
## [11,] -0.4310505
## [12,] -0.4191156
## [13,] -0.4100225
## [14,] -0.4028036
## [15,] -0.3965443
## [16,] -0.3909067
## [17,] -0.3857913
## [18,] -0.3811000
## [19,] -0.3767921
## [20,] -0.3728768
## [21,] -0.3693539
## [22,] -0.3662040
## [23,] -0.3634006
## [24,] -0.3609123
## [25,] -0.3587052
## [26,] -0.3567469
## [27,] -0.3550077
## [28,] -0.3534615
## [29,] -0.3520856
## [30,] -0.3508604
```

```
## [31,] -0.3497692
## [32,] -0.3487972
## [33,] -0.3479314
## [34,] -0.3471603
## [35,] -0.3464736
## [36,] -0.3458623
## [37,] -0.3453179
## [38,] -0.3448333
## [39,] -0.3444019
## [40,] -0.3440178
## [41,] -0.3436758
## [42,] -0.3433714
## [43,] -0.3431003
## [44,] -0.3428590
## [45,] -0.3426441
## [46,] -0.3424528
## [47,] -0.3422825
## [48,] -0.3421309
## [49,] -0.3419958
##
## $U
##                 U
##  [1,] 0.005488222
##  [2,] 0.130740116
##  [3,] 0.274634970
##  [4,] 0.334965264
##  [5,] 0.349882490
##  [6,] 0.349878871
##  [7,] 0.333701118
##  [8,] 0.307824632
##  [9,] 0.282543856
## [10,] 0.261014947
## [11,] 0.243366684
## [12,] 0.229555542
## [13,] 0.218752275
## [14,] 0.209954155
## [15,] 0.202577181
## [16,] 0.196200811
## [17,] 0.190484400
## [18,] 0.185275491
## [19,] 0.180538132
## [20,] 0.176245725
## [21,] 0.172371112
## [22,] 0.168894561
## [23,] 0.165792406
## [24,] 0.163032911
## [25,] 0.160581662
## [26,] 0.158405120
## [27,] 0.156471828
## [28,] 0.154753387
## [29,] 0.153224807
## [30,] 0.151864269
## [31,] 0.150652794
## [32,] 0.149573846
```

```
## [33,] 0.148612896
## [34,] 0.147757077
## [35,] 0.146994955
## [36,] 0.146316338
## [37,] 0.145712124
## [38,] 0.145174183
## [39,] 0.144695255
## [40,] 0.144268870
## [41,] 0.143889261
## [42,] 0.143551292
## [43,] 0.143250392
## [44,] 0.142982493
## [45,] 0.142743973
## [46,] 0.142531610
## [47,] 0.142342536
## [48,] 0.142174197
## [49,] 0.142024319
##
## $rw
##                   U
##  [1,]  0.0498174131
##  [2,] -0.0242232048
##  [3,] -0.0083672735
##  [4,]  0.0515879812
##  [5,]  0.0708593994
##  [6,]  0.0773943101
##  [7,]  0.0852478401
##  [8,]  0.0812862559
##  [9,]  0.0702674342
## [10,]  0.0610999821
## [11,]  0.0531969649
## [12,]  0.0454309885
## [13,]  0.0390519169
## [14,]  0.0342470678
## [15,]  0.0303043716
## [16,]  0.0269374542
## [17,]  0.0240737319
## [18,]  0.0215585555
## [19,]  0.0192887182
## [20,]  0.0172372414
## [21,]  0.0153832484
## [22,]  0.0137083648
## [23,]  0.0122042973
## [24,]  0.0108608236
## [25,]  0.0096633451
## [26,]  0.0085978794
## [27,]  0.0076512881
## [28,]  0.0068103537
## [29,]  0.0060627878
## [30,]  0.0053978787
## [31,]  0.0048062201
## [32,]  0.0042795084
## [33,]  0.0038104858
## [34,]  0.0033927946
```

```
## [35,]  0.0030208155
## [36,]  0.0026895622
## [37,]  0.0023945984
## [38,]  0.0021319666
## [39,]  0.0018981343
## [40,]  0.0016899502
## [41,]  0.0015046037
## [42,]  0.0013395895
## [43,]  0.0011926759
## [44,]  0.0010618764
## [45,]  0.0009454225
## [46,]  0.0008417401
## [47,]  0.0007494282
## [48,]  0.0006672398
## [49,]  0.0005940645
##
##
## Lower Band, CI= 0.95
## $prod
##                U
##  [1,] -0.07916811
##  [2,] -0.16508904
##  [3,] -0.25929628
##  [4,] -0.38644150
##  [5,] -0.51203492
##  [6,] -0.56116493
##  [7,] -0.56545000
##  [8,] -0.55779762
##  [9,] -0.53880604
## [10,] -0.51742206
## [11,] -0.50048263
## [12,] -0.49757778
## [13,] -0.48410462
## [14,] -0.49058322
## [15,] -0.49550270
## [16,] -0.49851112
## [17,] -0.49757766
## [18,] -0.49523288
## [19,] -0.49251258
## [20,] -0.48968268
## [21,] -0.48680028
## [22,] -0.48394746
## [23,] -0.48117157
## [24,] -0.47847672
## [25,] -0.47590015
## [26,] -0.47346366
## [27,] -0.47116640
## [28,] -0.46901881
## [29,] -0.46702863
## [30,] -0.46675388
## [31,] -0.46669906
## [32,] -0.46665159
## [33,] -0.46660981
## [34,] -0.46657325
```

```
## [35,] -0.46654159
## [36,] -0.46651395
## [37,] -0.46648987
## [38,] -0.46646896
## [39,] -0.46645077
## [40,] -0.46643498
## [41,] -0.46642131
## [42,] -0.46640946
## [43,] -0.46639921
## [44,] -0.46639036
## [45,] -0.46638272
## [46,] -0.46637612
## [47,] -0.46637042
## [48,] -0.46636552
## [49,] -0.46636128
##
## $e
##                U
##  [1,] -0.2811554
##  [2,] -0.4630211
##  [3,] -0.5714115
##  [4,] -0.6439410
##  [5,] -0.6470449
##  [6,] -0.6327614
##  [7,] -0.5977278
##  [8,] -0.5574146
##  [9,] -0.5278927
## [10,] -0.5102261
## [11,] -0.4991048
## [12,] -0.4919288
## [13,] -0.4792290
## [14,] -0.4687839
## [15,] -0.4608807
## [16,] -0.4538126
## [17,] -0.4476129
## [18,] -0.4420313
## [19,] -0.4371039
## [20,] -0.4330092
## [21,] -0.4294405
## [22,] -0.4273060
## [23,] -0.4266437
## [24,] -0.4261060
## [25,] -0.4256607
## [26,] -0.4252861
## [27,] -0.4249730
## [28,] -0.4247109
## [29,] -0.4244889
## [30,] -0.4243014
## [31,] -0.4241440
## [32,] -0.4240114
## [33,] -0.4238992
## [34,] -0.4238045
## [35,] -0.4237246
## [36,] -0.4236567
```

```
## [37,] -0.4235991
## [38,] -0.4235502
## [39,] -0.4235085
## [40,] -0.4234730
## [41,] -0.4234426
## [42,] -0.4234167
## [43,] -0.4233945
## [44,] -0.4233755
## [45,] -0.4233592
## [46,] -0.4233452
## [47,] -0.4233332
## [48,] -0.4233228
## [49,] -0.4233139
##
## $U
##                  U
##  [1,]  0.002368871
##  [2,] -0.106630847
##  [3,] -0.264812737
##  [4,] -0.355502852
##  [5,] -0.368693936
##  [6,] -0.370960714
##  [7,] -0.351455285
##  [8,] -0.315193463
##  [9,] -0.265809185
## [10,] -0.213860152
## [11,] -0.197409730
## [12,] -0.196367657
## [13,] -0.196213979
## [14,] -0.196420038
## [15,] -0.195930924
## [16,] -0.194674666
## [17,] -0.193102861
## [18,] -0.191235120
## [19,] -0.189527334
## [20,] -0.187845102
## [21,] -0.186136832
## [22,] -0.184488264
## [23,] -0.183008947
## [24,] -0.181721916
## [25,] -0.180595641
## [26,] -0.179607487
## [27,] -0.178746456
## [28,] -0.177992940
## [29,] -0.177322079
## [30,] -0.176717221
## [31,] -0.176170692
## [32,] -0.175677034
## [33,] -0.175230699
## [34,] -0.174827516
## [35,] -0.174464737
## [36,] -0.174139589
## [37,] -0.173848720
## [38,] -0.173588692
```

```
## [39,] -0.173356355
## [40,] -0.173148781
## [41,] -0.172963207
## [42,] -0.172797138
## [43,] -0.172648421
## [44,] -0.172515198
## [45,] -0.172395834
## [46,] -0.172288880
## [47,] -0.172193057
## [48,] -0.172107230
## [49,] -0.172030375
##
## $rw
##                  U
##  [1,] -2.505511e-02
##  [2,] -8.402177e-02
##  [3,] -6.461270e-02
##  [4,] -2.220798e-02
##  [5,]  1.352681e-03
##  [6,]  1.360884e-02
##  [7,]  2.575382e-02
##  [8,]  2.444480e-02
##  [9,]  1.216857e-02
## [10,]  1.172589e-02
## [11,]  9.003558e-03
## [12,]  6.995738e-03
## [13,]  5.906098e-03
## [14,]  4.002505e-03
## [15,]  3.622165e-03
## [16,]  3.060153e-03
## [17,]  2.026452e-03
## [18,]  1.125446e-03
## [19,]  1.181241e-04
## [20,]  3.364319e-04
## [21,]  4.386181e-04
## [22,] -3.996162e-05
## [23,] -4.765075e-04
## [24,] -9.776262e-04
## [25,] -1.035097e-03
## [26,] -1.038754e-03
## [27,] -9.909078e-04
## [28,] -1.020952e-03
## [29,] -9.945842e-04
## [30,] -9.240660e-04
## [31,] -8.179480e-04
## [32,] -6.978394e-04
## [33,] -5.819281e-04
## [34,] -4.672096e-04
## [35,] -2.691548e-04
## [36,] -1.862364e-04
## [37,] -1.418611e-04
## [38,] -9.879063e-05
## [39,] -6.431435e-05
## [40,] -3.742851e-05
```

```
## [41,] -1.844580e-05
## [42,] -1.208512e-05
## [43,] -7.918276e-06
## [44,] -6.369435e-07
## [45,]  2.302152e-07
## [46,]  1.010143e-08
## [47,] -1.120643e-07
## [48,] -2.385427e-07
## [49,] -1.381189e-07
##
##
## Upper Band, CI= 0.95
## $prod
##                U
##  [1,] 0.15996121
##  [2,] 0.22503794
##  [3,] 0.30878979
##  [4,] 0.31957474
##  [5,] 0.26146116
##  [6,] 0.22128314
##  [7,] 0.18916640
##  [8,] 0.15240523
##  [9,] 0.12697492
## [10,] 0.11253259
## [11,] 0.10364183
## [12,] 0.09680636
## [13,] 0.08831749
## [14,] 0.08042285
## [15,] 0.07799467
## [16,] 0.07574282
## [17,] 0.07353731
## [18,] 0.07217651
## [19,] 0.07198874
## [20,] 0.07284943
## [21,] 0.07440658
## [22,] 0.07629473
## [23,] 0.07818437
## [24,] 0.07979441
## [25,] 0.08095269
## [26,] 0.08162831
## [27,] 0.08190232
## [28,] 0.08191137
## [29,] 0.08179452
## [30,] 0.08166100
## [31,] 0.08158122
## [32,] 0.08158814
## [33,] 0.08168154
## [34,] 0.08183720
## [35,] 0.08202000
## [36,] 0.08219618
## [37,] 0.08234145
## [38,] 0.08244397
## [39,] 0.08250312
## [40,] 0.08252621
```

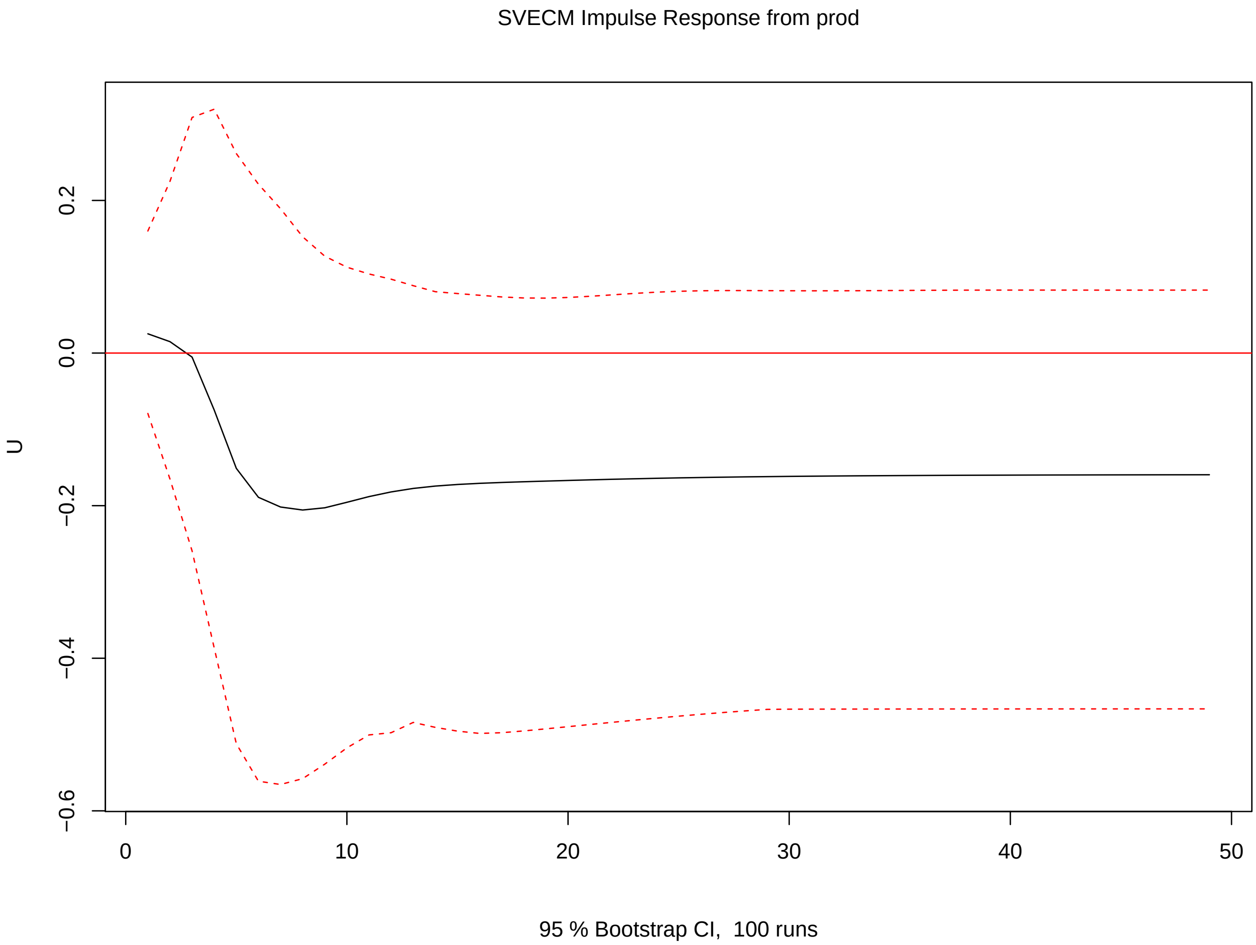
```
## [41,] 0.08252476
## [42,] 0.08251105
## [43,] 0.08249544
## [44,] 0.08248473
## [45,] 0.08248191
## [46,] 0.08248674
## [47,] 0.08249698
## [48,] 0.08250955
## [49,] 0.08252154
##
## $e
##                 U
##  [1,] -0.17519955
##  [2,] -0.19847484
##  [3,] -0.21035382
##  [4,] -0.24352822
##  [5,] -0.23237617
##  [6,] -0.20218922
##  [7,] -0.18870006
##  [8,] -0.18494427
##  [9,] -0.18501214
## [10,] -0.17753330
## [11,] -0.17726587
## [12,] -0.16405511
## [13,] -0.15126673
## [14,] -0.14181791
## [15,] -0.13419912
## [16,] -0.12783645
## [17,] -0.12236486
## [18,] -0.11789776
## [19,] -0.11419452
## [20,] -0.11089204
## [21,] -0.10804140
## [22,] -0.10567799
## [23,] -0.10367068
## [24,] -0.10195132
## [25,] -0.10049684
## [26,] -0.09927139
## [27,] -0.09824369
## [28,] -0.09737937
## [29,] -0.09664818
## [30,] -0.09603356
## [31,] -0.09551849
## [32,] -0.09508406
## [33,] -0.09471707
## [34,] -0.09440763
## [35,] -0.09414646
## [36,] -0.09392579
## [37,] -0.09373919
## [38,] -0.09358131
## [39,] -0.09344784
## [40,] -0.09333500
## [41,] -0.09323950
## [42,] -0.09315870
```

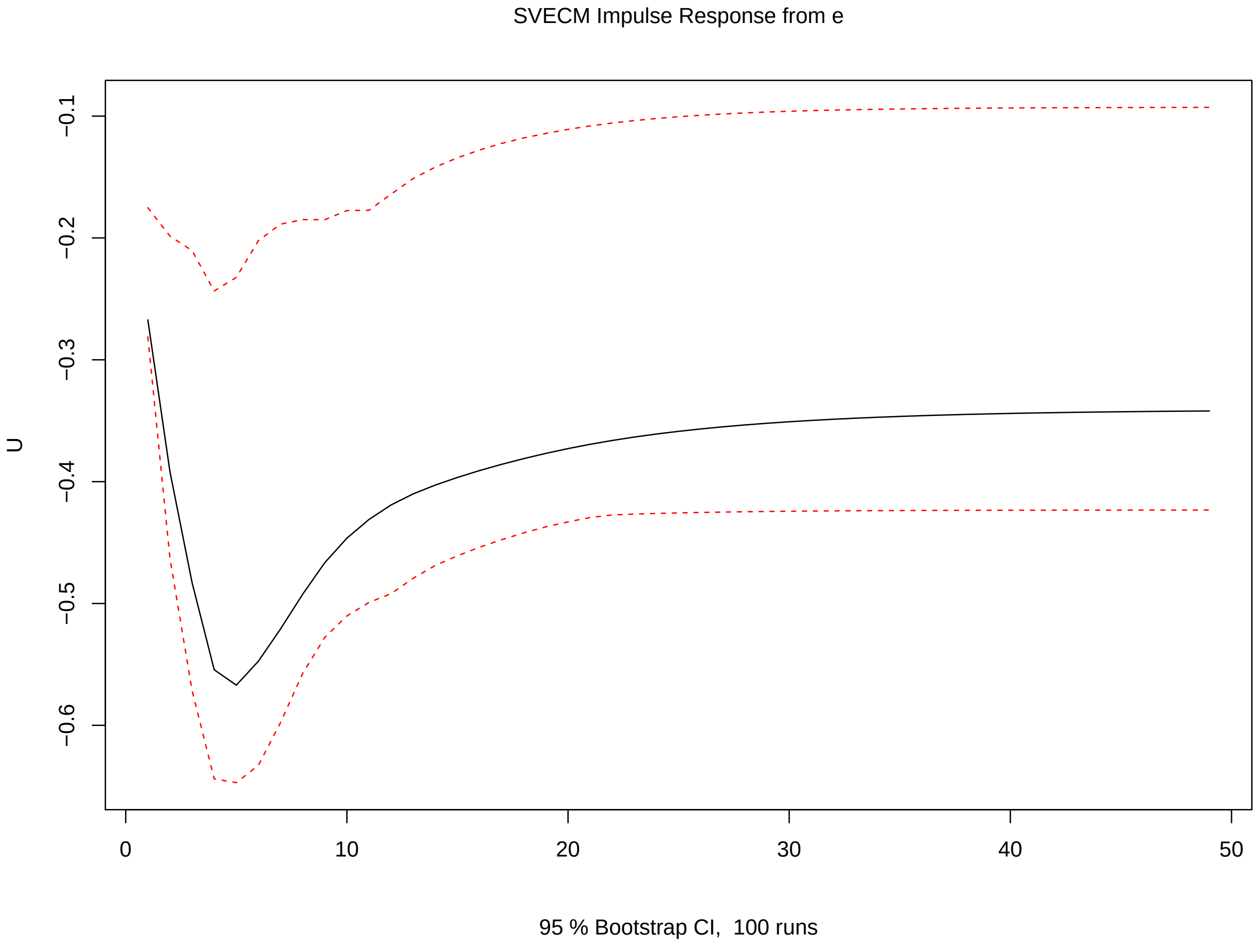
```
## [43,] -0.09309035
## [44,] -0.09303252
## [45,] -0.09298356
## [46,] -0.09294210
## [47,] -0.09290700
## [48,] -0.09287726
## [49,] -0.09285206
##
## $U
##               U
##  [1,] 0.1364445
##  [2,] 0.2922922
##  [3,] 0.4682270
##  [4,] 0.5611019
##  [5,] 0.5970659
##  [6,] 0.5726682
##  [7,] 0.5472688
##  [8,] 0.5200603
##  [9,] 0.4706195
## [10,] 0.4441134
## [11,] 0.4251272
## [12,] 0.4128706
## [13,] 0.4030959
## [14,] 0.3974295
## [15,] 0.3927874
## [16,] 0.3883962
## [17,] 0.3845218
## [18,] 0.3806072
## [19,] 0.3792404
## [20,] 0.3786174
## [21,] 0.3780900
## [22,] 0.3776032
## [23,] 0.3771222
## [24,] 0.3766852
## [25,] 0.3763011
## [26,] 0.3759466
## [27,] 0.3717667
## [28,] 0.3672150
## [29,] 0.3630813
## [30,] 0.3601789
## [31,] 0.3597997
## [32,] 0.3594597
## [33,] 0.3591606
## [34,] 0.3588955
## [35,] 0.3586570
## [36,] 0.3584424
## [37,] 0.3582490
## [38,] 0.3580743
## [39,] 0.3579163
## [40,] 0.3577733
## [41,] 0.3576437
## [42,] 0.3575260
## [43,] 0.3574191
## [44,] 0.3573219
```

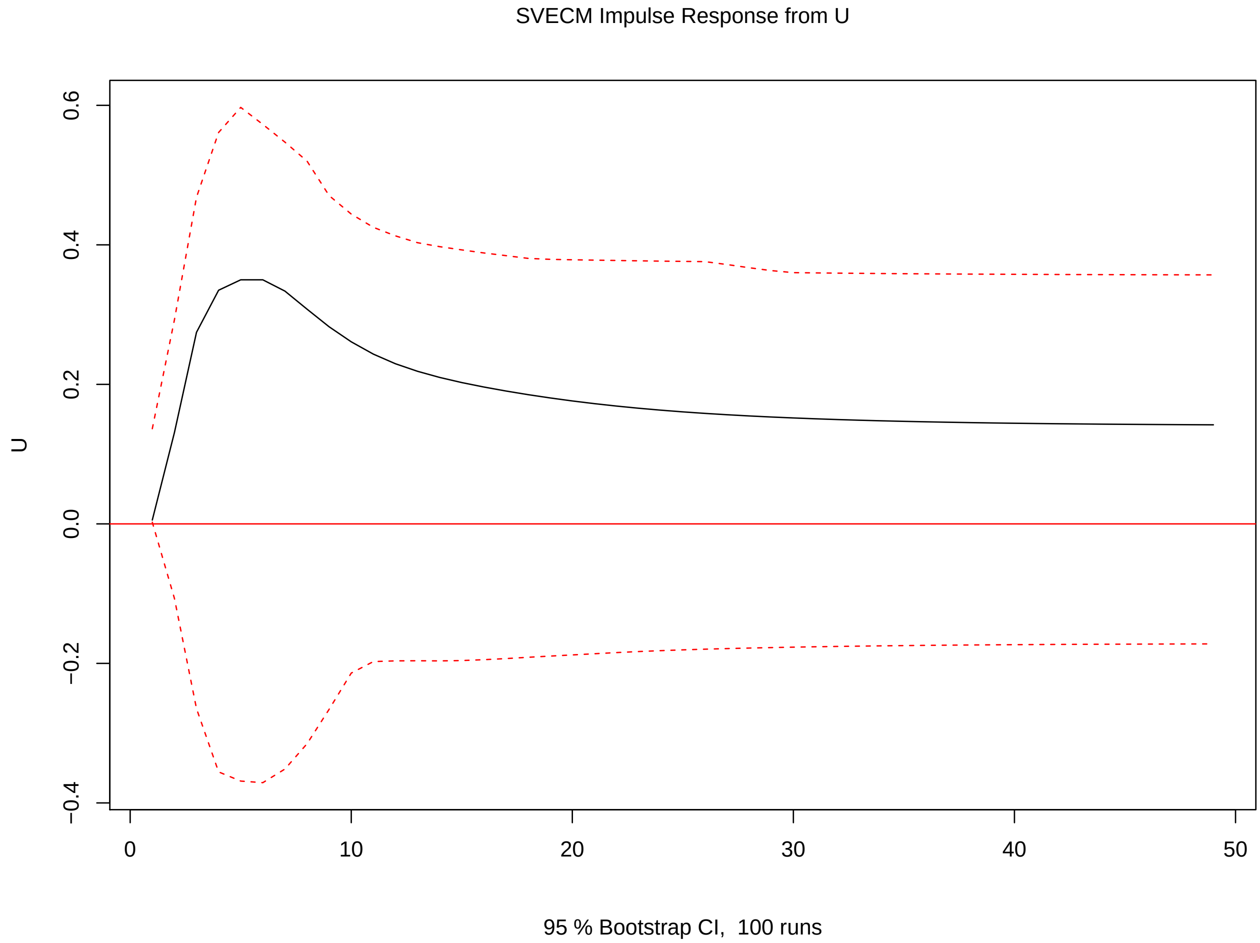
```
## [45,] 0.3572334
## [46,] 0.3571527
## [47,] 0.3570792
## [48,] 0.3570120
## [49,] 0.3569507
##
## $rw
##                 U
##  [1,] 0.101860197
##  [2,] 0.063458607
##  [3,] 0.080203857
##  [4,] 0.137268808
##  [5,] 0.137776625
##  [6,] 0.131908966
##  [7,] 0.139124865
##  [8,] 0.121902379
##  [9,] 0.098902645
## [10,] 0.085054644
## [11,] 0.070919773
## [12,] 0.060232433
## [13,] 0.052143188
## [14,] 0.040760593
## [15,] 0.035124901
## [16,] 0.033698686
## [17,] 0.032251070
## [18,] 0.029459619
## [19,] 0.025984467
## [20,] 0.023980786
## [21,] 0.022005730
## [22,] 0.019721905
## [23,] 0.017348226
## [24,] 0.016688371
## [25,] 0.016104725
## [26,] 0.015323506
## [27,] 0.014012498
## [28,] 0.012460765
## [29,] 0.011054964
## [30,] 0.009786810
## [31,] 0.008647078
## [32,] 0.007624962
## [33,] 0.006784450
## [34,] 0.006208769
## [35,] 0.005683177
## [36,] 0.005203237
## [37,] 0.004814679
## [38,] 0.004483053
## [39,] 0.004225435
## [40,] 0.003982595
## [41,] 0.003754528
## [42,] 0.003545620
## [43,] 0.003349554
## [44,] 0.003165633
## [45,] 0.002992959
## [46,] 0.002830598
```

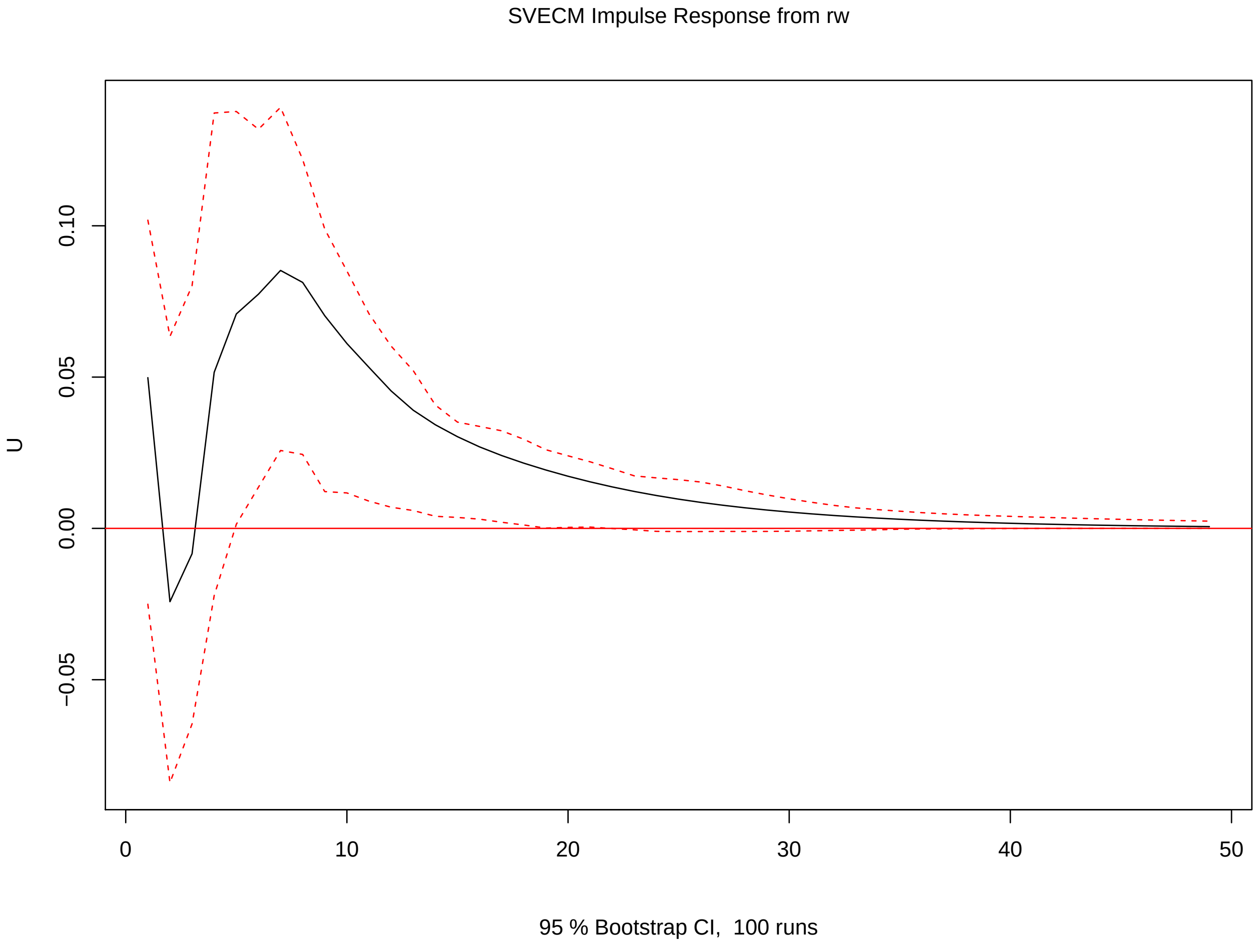
```
## [47,] 0.002677705
## [48,] 0.002533605
## [49,] 0.002397785
```

```r
plot(svec.irf)
```




SVECM Impulse Response from e
U
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
0
10
20
30
40
50
95 % Bootstrap CI, 100 runs


SVECM Impulse Response from U
U
0.6
0.4
0.2
0.0
−0.2
−0.4
0
10
20
30
40
50
95 % Bootstrap CI, 100 runs



SVECM Impulse Response from rw
U
0.10
0.05
0.00
−0.05
0
10
20
30
40
50
95 % Bootstrap CI, 100 runs