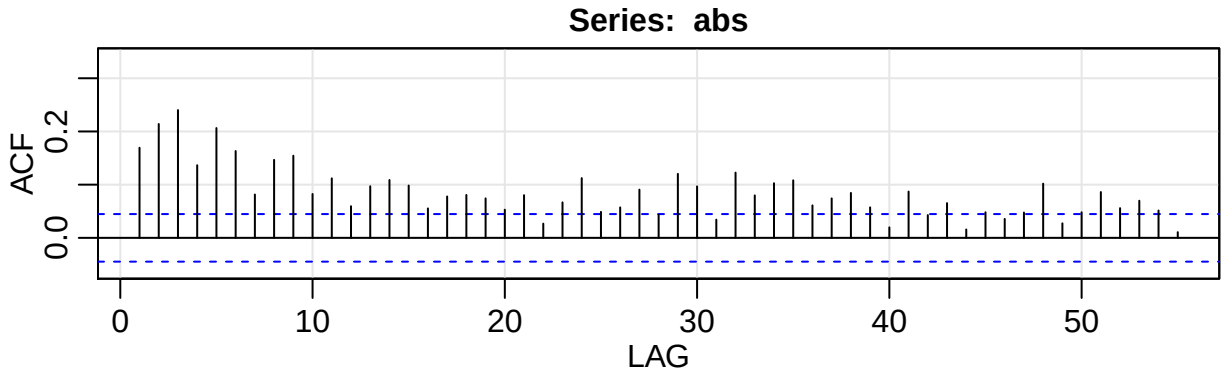
Assignment 4

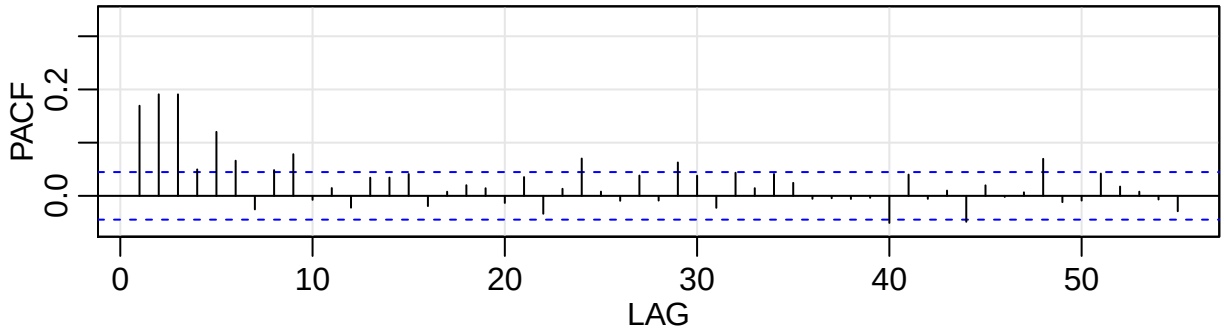
Liam Fruzyna

1) Consider the absolute values of the nyse returns data

a) Check using ACF whether abs(nyse) follows long memory or short memory

```
acf2(abs)
```





Long memory, doesn't cut off or exponentially decay

b) Fit ARFIMA with appropriate order. Make sure to test the residuals.

```
anyse = abs - mean(abs)
nyse.fd = fracdiff(anyse, nar=22, nma=6, M=30)
```

```
## Warning: C fracdf() optimization failure
```

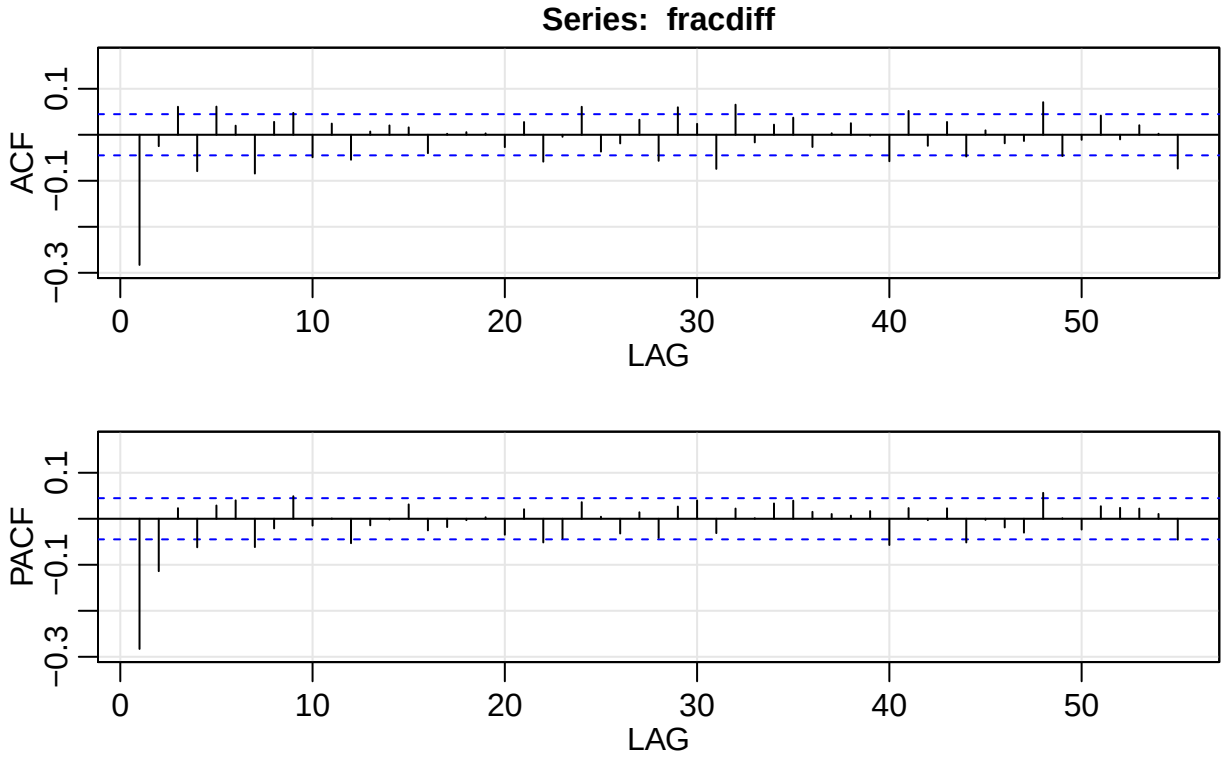
```
## Warning: unable to compute correlation matrix; maybe change 'h'
```

```
nyse.fd
```

```
##
## Call:
##   fracdiff(x = anyse, nar = 22, nma = 6, M = 30)
##
## *** Warning during (fracdf) fit: C fracdf() optimization failure
##
## *** Warning during (fdcov) fit: unable to compute correlation matrix; maybe change 'h'
```

```
##
## Coefficients:
##             d           ar1           ar2           ar3           ar4
##  0.3824180732 -0.3034303187 -0.3046372000  0.0672796709  0.0601057521
##           ar5           ar6           ar7           ar8           ar9
##  0.4155211781  0.7794009851  0.1722748448  0.0379186868  0.0139432970
##          ar10          ar11          ar12          ar13          ar14
## -0.0258320409 -0.0481545484 -0.0656091017  0.0334705573  0.0009811607
##          ar15          ar16          ar17          ar18          ar19
##  0.0206356537  0.0196294632  0.0557346380  0.0663827230  0.0359818159
##          ar20          ar21          ar22           ma1           ma2
## -0.0103284995  0.0153593455 -0.0404592183  0.0064518613 -0.2035448544
##           ma3           ma4           ma5           ma6
##  0.1291237516  0.0914974397  0.3481993869  0.6811187462
## sigma[eps] = 0.007010013
## a list with components:
##  [1] "log.likelihood"  "n"               "msg"
##  [4] "d"               "ar"              "ma"
##  [7] "covariance.dpq"  "fnormMin"        "sigma"
## [10] "stderror.dpq"    "correlation.dpq" "h"
## [13] "d.tol"           "M"               "hessian.dpq"
## [16] "length.w"        "call"
```

```
fracdiff = diffseries(anyse, nyse.fd$d)
acf2(fracdiff)
```




```
Box.test(fracdiff, type='Ljung', lag=20)
```

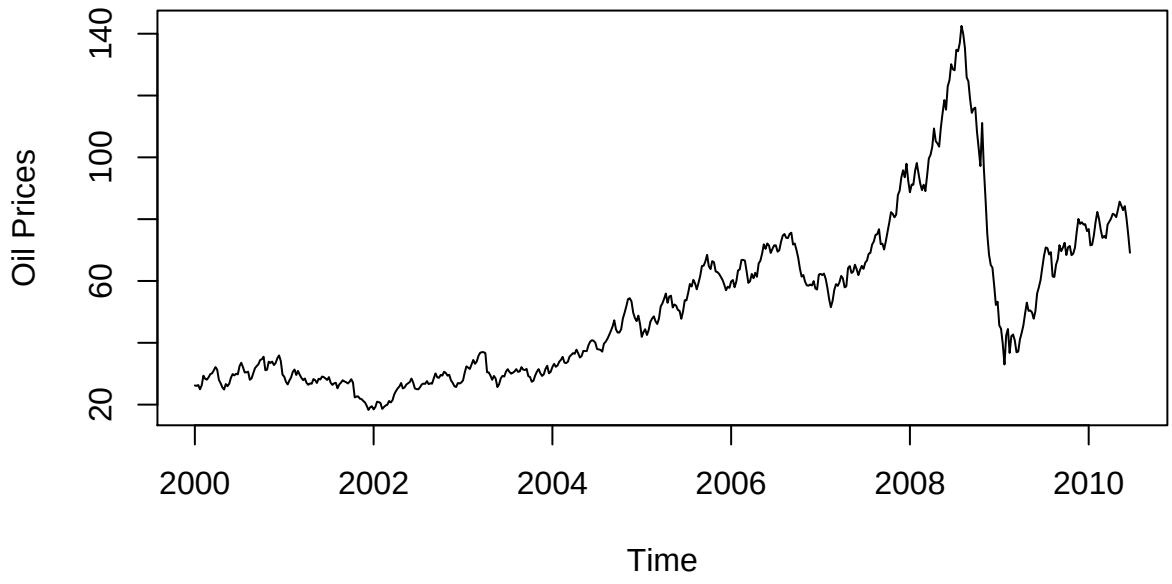
```
##
##  Box-Ljung test
```

```
##
## data:  fracdiff
## X-squared = 228.4, df = 20, p-value < 2.2e-16
```

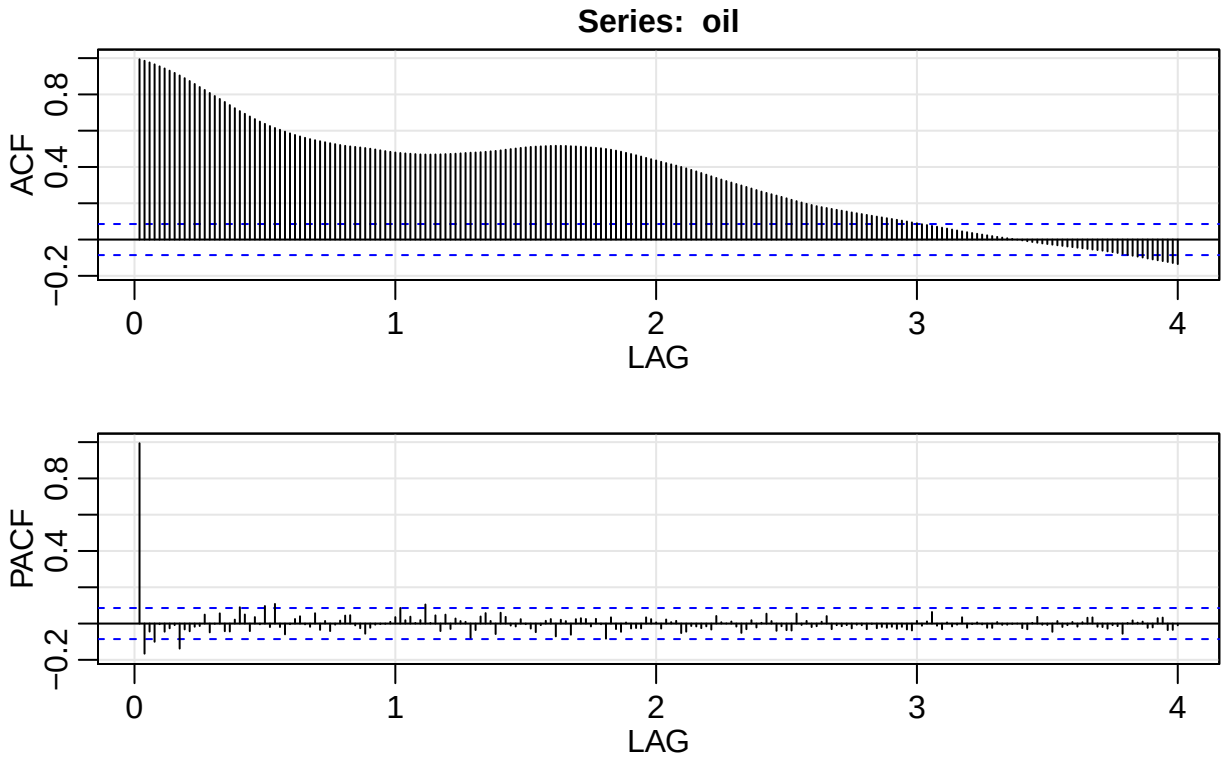
2) Weekly crude oil spot prices in dollars per barrel are in oil

a Investigate whether the growth rate of the weekly oil prices exhibit GARCH behavior

```
plot.ts(oil, ylab='Oil Prices')
```




```
acf2(oil)
```



b) Is the weekly growth rate white noise?

No, not white noise.

c) Fit an appropriate GARCH model

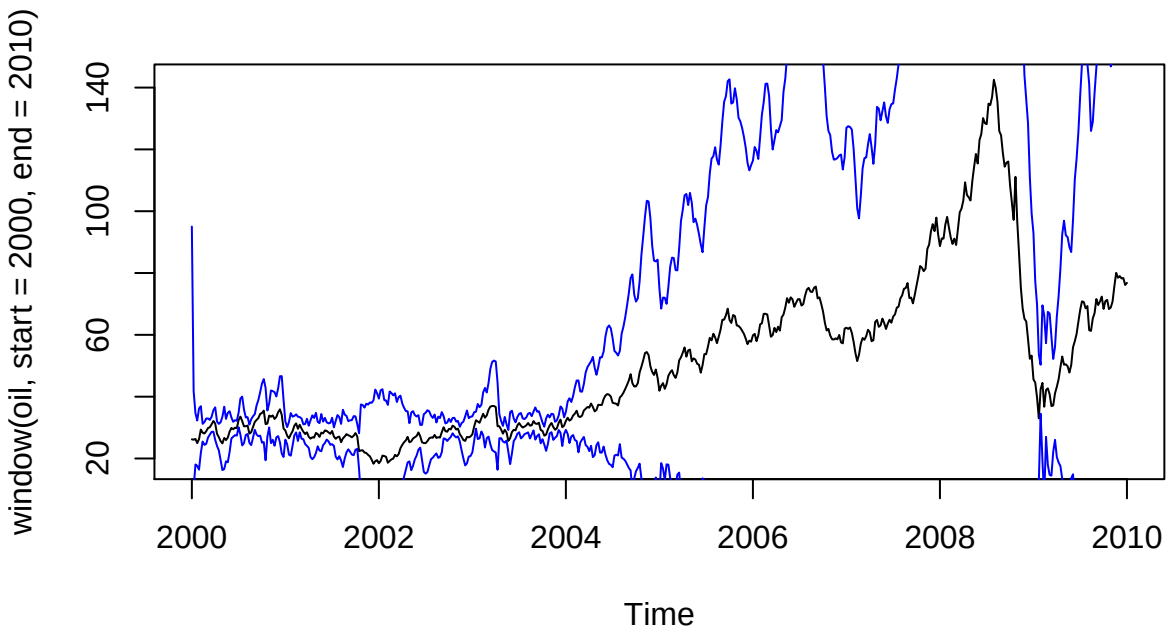
```
fit = garchFit(~garch(1, 1), oil)
```

```
##
## Series Initialization:
##  ARMA Model:                 arma
##  Formula Mean:               ~ arma(0, 0)
##  GARCH Model:                garch
##  Formula Variance:           ~ garch(1, 1)
##  ARMA Order:                 0 0
##  Max ARMA Order:             0
##  GARCH Order:                1 1
##  Max GARCH Order:            1
##  Maximum Order:              1
##  Conditional Dist:           norm
##  h.start:                    2
##  llh.start:                  1
##  Length of Series:           545
##  Recursion Init:             mci
##  Series Scale:               25.90377
##
## Parameter Initialization:
##  Initial Parameters:           $params
##  Limits of Transformations:    $U, $V
```

```
##  Which Parameters are Fixed?  $includes
##  Parameter Matrix:
##                        U         V   params includes
##     mu     -20.09484035  20.09484 2.009484     TRUE
##     omega    0.00000100 100.00000 0.100000     TRUE
##     alpha1   0.00000001   1.00000 0.100000     TRUE
##     gamma1  -0.99999999   1.00000 0.100000    FALSE
##     beta1    0.00000001   1.00000 0.800000     TRUE
##     delta    0.00000000   2.00000 2.000000    FALSE
##     skew     0.10000000  10.00000 1.000000    FALSE
##     shape    1.00000000  10.00000 4.000000    FALSE
##  Index List of Parameters to be Optimized:
##     mu  omega alpha1  beta1
##      1      2      3      5
##  Persistence:                  0.9
##
##
## --- START OF TRACE ---
## Selected Algorithm: nlminb
##
## R coded nlminb Solver:
##
##   0:     665.20476:  2.00948 0.100000 0.100000 0.800000
##   1:     638.25108:  1.99360 0.0651184 0.0996910 0.779371
##   2:     555.36438:  1.73499 1.00000e-06 0.318869 0.694026
##   3:     554.56015:  1.73447 0.00322690 0.318877 0.693914
##   4:     507.69015:  1.49648 0.00146283 0.467194 0.610677
##   5:     439.87631:  1.30955 1.00000e-06 0.580146 0.546762
##   6:     388.05355:  1.15504 1.00000e-06 0.574531 0.536482
##   7:     386.90914:  1.15508 0.00164968 0.574530 0.536481
##   8:     385.70263:  1.15550 0.00146549 0.575253 0.536198
##   9:     381.82210:  1.15551 0.000297944 0.575237 0.536161
##  10:     381.33901:  1.15553 0.000536269 0.575234 0.536156
##  11:     381.31453:  1.15566 0.000504135 0.575444 0.536069
##  12:     381.30490:  1.15580 0.000479587 0.575649 0.535973
##  13:     381.29145:  1.15607 0.000500116 0.576070 0.535794
##  14:     381.27143:  1.15658 0.000472748 0.576924 0.535453
##  15:     381.24605:  1.15757 0.000500195 0.578646 0.534795
##  16:     381.23326:  1.15853 0.000448384 0.580375 0.534148
##  17:     381.22458:  1.15948 0.000487325 0.582111 0.533512
##  18:     381.20250:  1.16021 0.000450354 0.583806 0.532715
##  19:     381.15442:  1.15744 0.000465217 0.579389 0.533848
##  20:     381.15245:  1.15746 0.000507517 0.579381 0.533831
##  21:     381.14485:  1.15747 0.000485536 0.579377 0.533824
##  22:     381.12533:  1.15764 0.000443043 0.579309 0.533674
##  23:     381.08725:  1.15764 0.000486137 0.579585 0.533415
##  24:     370.90435:  1.14457 0.000975175 0.815523 0.319590
##  25:     369.13925:  1.15596 0.00107231 0.865500 0.287077
##  26:     368.80583:  1.15959 0.000824314 0.864528 0.282582
##  27:     368.47823:  1.15959 0.00106830 0.859858 0.284196
##  28:     367.49107:  1.15731 0.00102479 0.647570 0.359242
##  29:     366.15946:  1.16817 0.000986685 0.706252 0.308003
##  30:     363.10455:  1.15899 0.00143608 0.904251 0.147534
##  31:     362.44135:  1.15716 0.00170626 0.959526 0.0677701
```

```
##  32:     362.30843:  1.15365 0.00183465 0.955167 0.0592314
##  33:     362.27076:  1.15076 0.00194149 0.962184 0.0483562
##  34:     362.26582:  1.14970 0.00197375 0.968710 0.0436907
##  35:     362.26491:  1.14928 0.00198227 0.973015 0.0411625
##  36:     362.26478:  1.14918 0.00198080 0.974067 0.0406649
##  37:     362.26474:  1.14914 0.00197879 0.974353 0.0404557
##  38:     362.26474:  1.14913 0.00197792 0.974232 0.0404700
##  39:     362.26474:  1.14913 0.00197791 0.974172 0.0404739
##  40:     362.26474:  1.14913 0.00197794 0.974161 0.0404782
##
## Final Estimate of the Negative LLH:
##  LLH:  2135.906    norm LLH:  3.919094
##          mu       omega      alpha1       beta1
## 29.76674823  1.32720694  0.97416132  0.04047824
##
## R-optimhess Difference Approximated Hessian Matrix:
##               mu       omega      alpha1      beta1
## mu     -9.701374   -2.529776    2.716553   29.24605
## omega  -2.529776 -575.034808  -12.896778  -42.74932
## alpha1  2.716553  -12.896778 -236.409042 -298.77724
## beta1  29.246051  -42.749318 -298.777243 -591.98502
## attr(,"time")
## Time difference of 0.01342463 secs
##
## --- END OF TRACE ---
##
##
## Time to Estimate Parameters:
##  Time difference of 0.1212881 secs
```

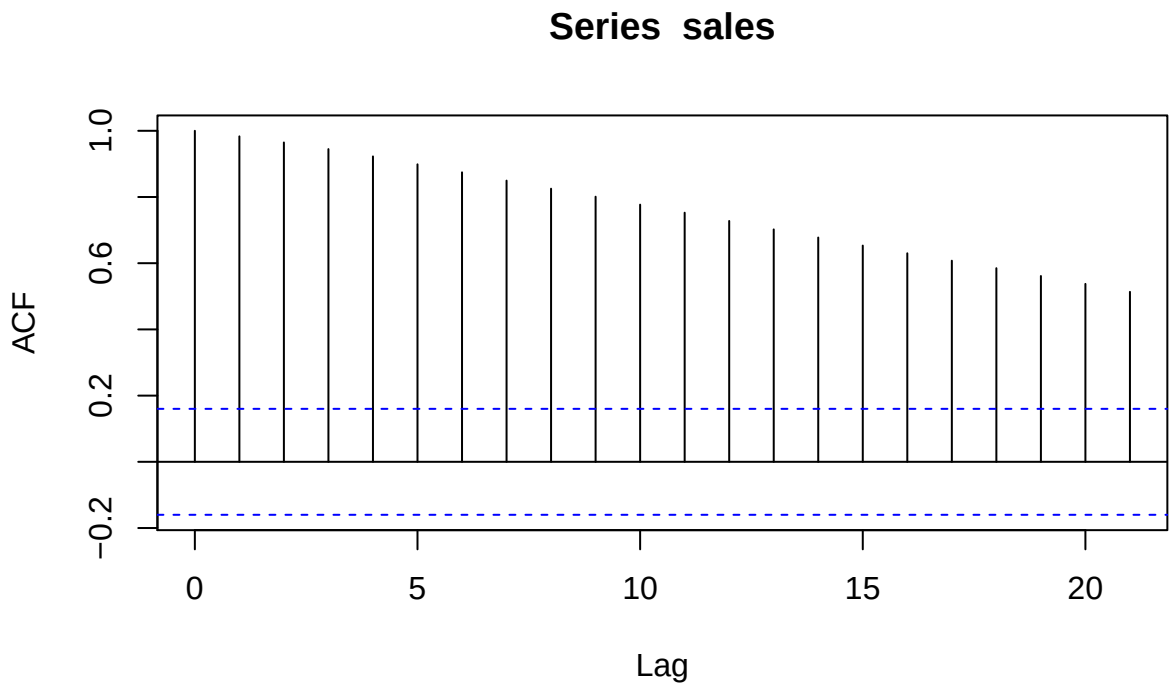
```
y = fit@sigma.t
plot(window(oil, start=2000, end=2010))
lines(window(oil-2*y, start=2000, end=2010), col=4)
lines(window(oil+2*y, start=2000, end=2010), col=4)
```



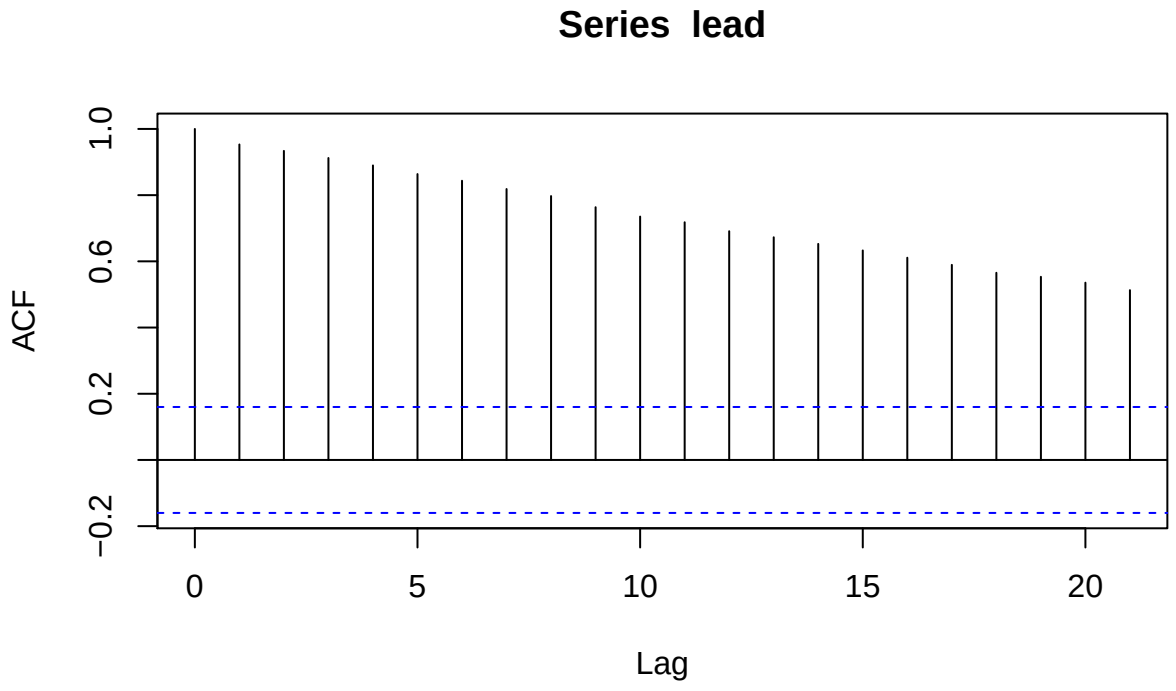
3) Let S_t represent the monthly sales data, sales, and L_t be a leading indicator lead

a) Plot the autocorrelation, acfs, or both sales and lead

```
acf(sales)
```

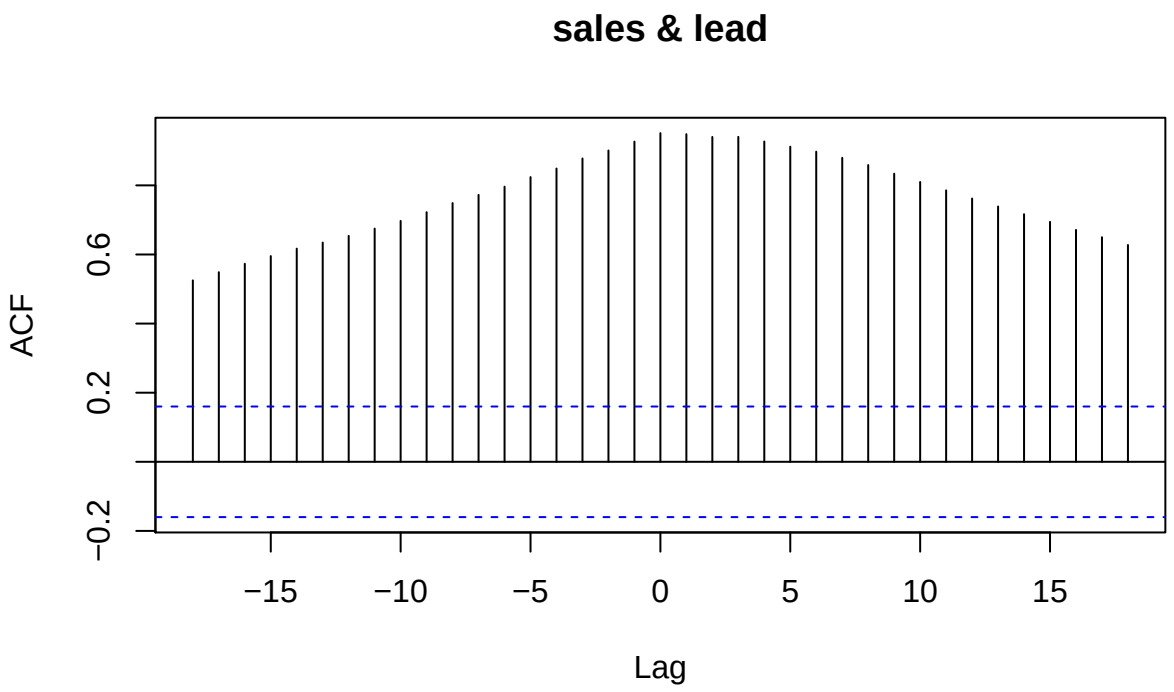



```
acf(lead)
```



Also plot the cross-correlation ccf between sales and leads. Observer their behaviors.

```
ccf(sales, lead)
```

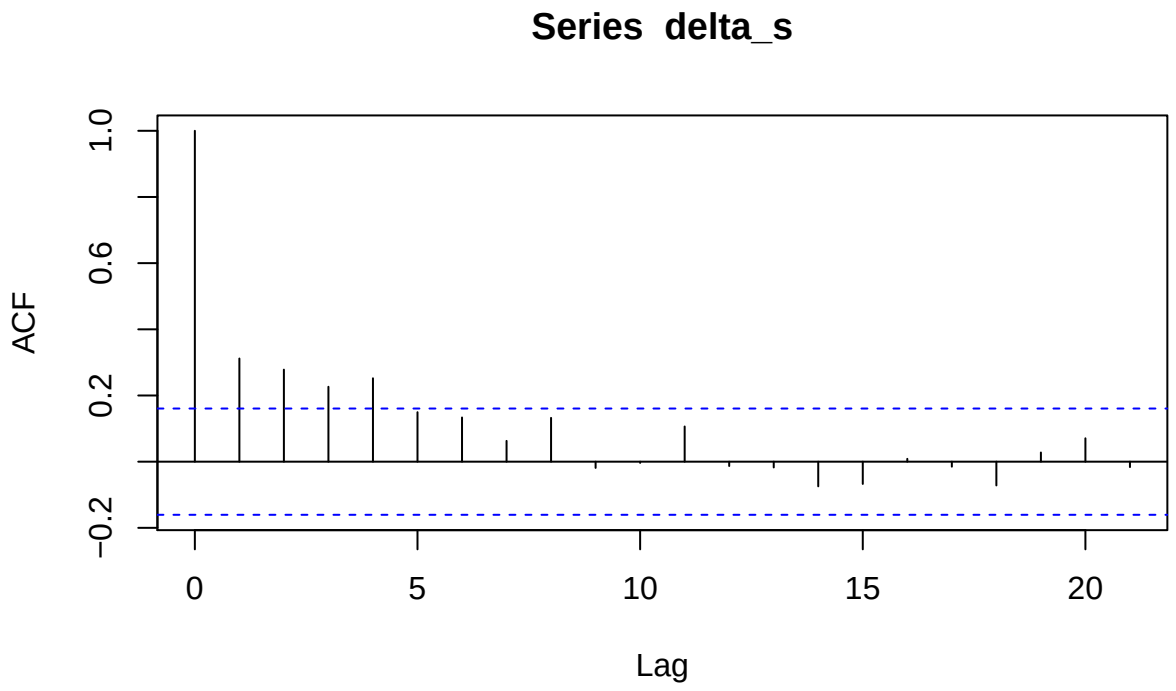


b) Compute the difference of sales ΔS_t and difference of lead ΔL_t.

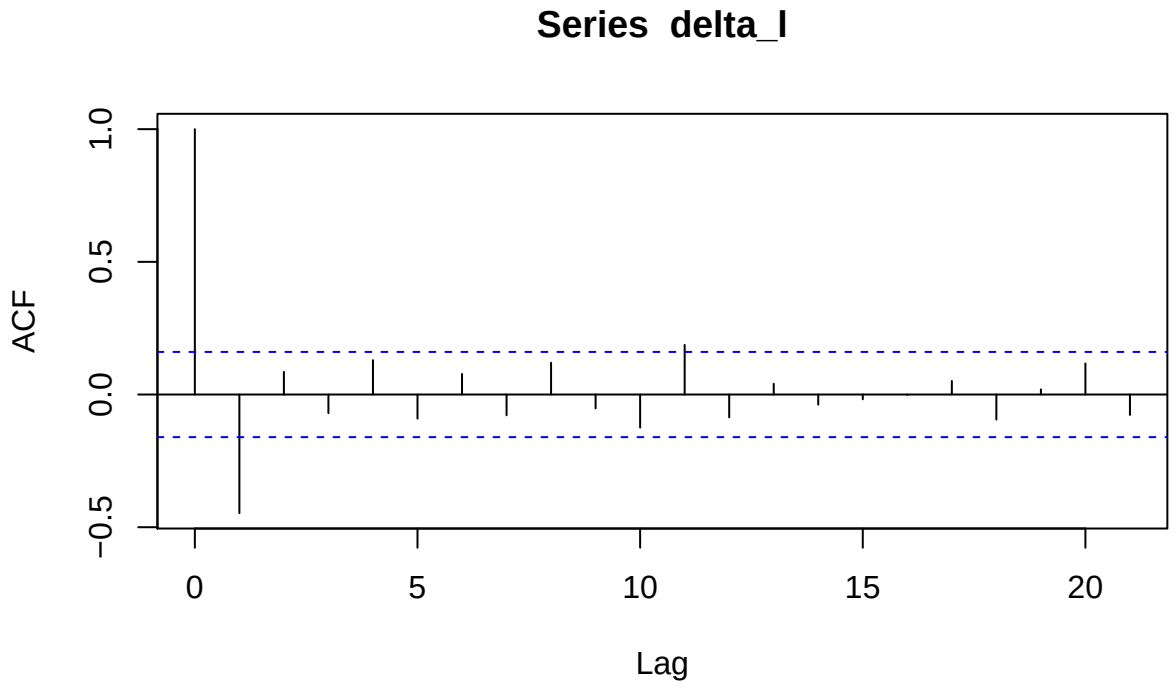
```
delta_s = diff(sales)
delta_l = diff(lead)
```

Repeat a for ΔS_t and ΔL_t.

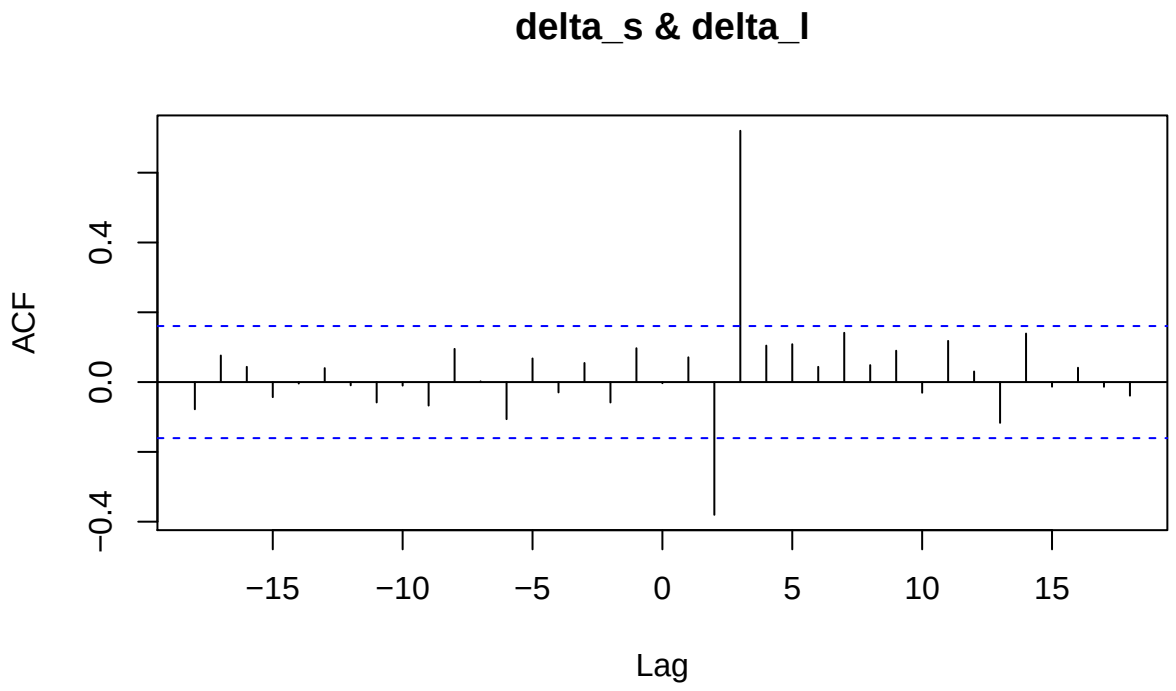
```
acf(delta_s)
```




```
acf(delta_l)
```




```
ccf(delta_s, delta_l)
```



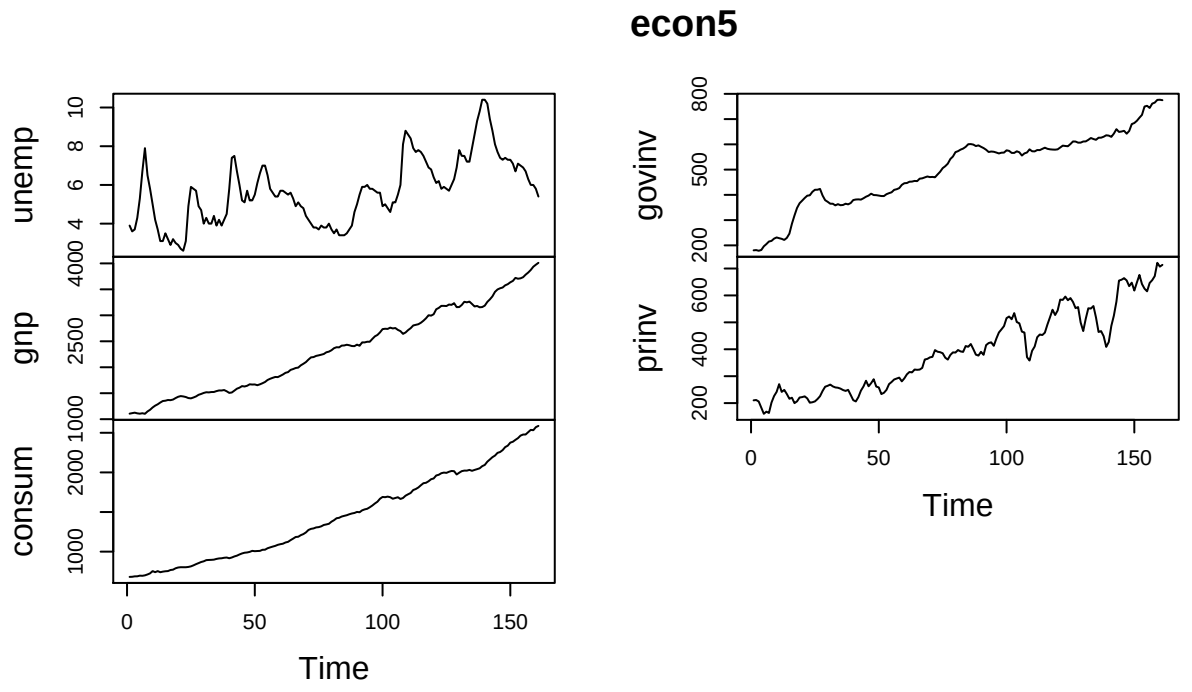
c) Fit the model $\Delta S_t = \beta_0 + \beta_1 \Delta L_{t-2} + \beta_2 \Delta L_{t-3} + x_t$, where x_t is an ARMA process.

```
fit = arima(delta_s, order=c(3,0,0), xreg=delta_l)
```

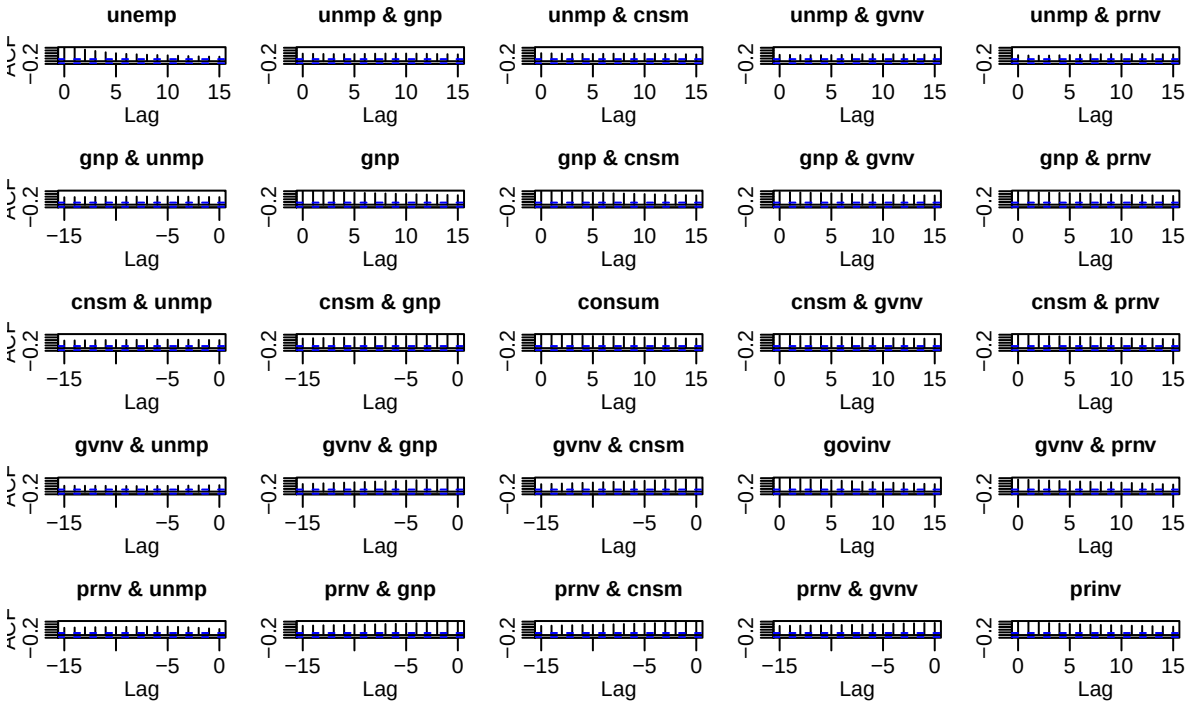
4) Consider the data set econ5. Conceptrate only on the unemployment, gnp, and consumption.

a) Fit an appropriate vector ARMA (VAR) to $x_t = (x_1 t, x_2 t, x_3 t)$, where $x_1 t = log(U_t) - \beta_0 - \beta_1 t$, $x_2 t = G_t$, and $x_3 t = C_t$. Meake sure to test for the residuals.

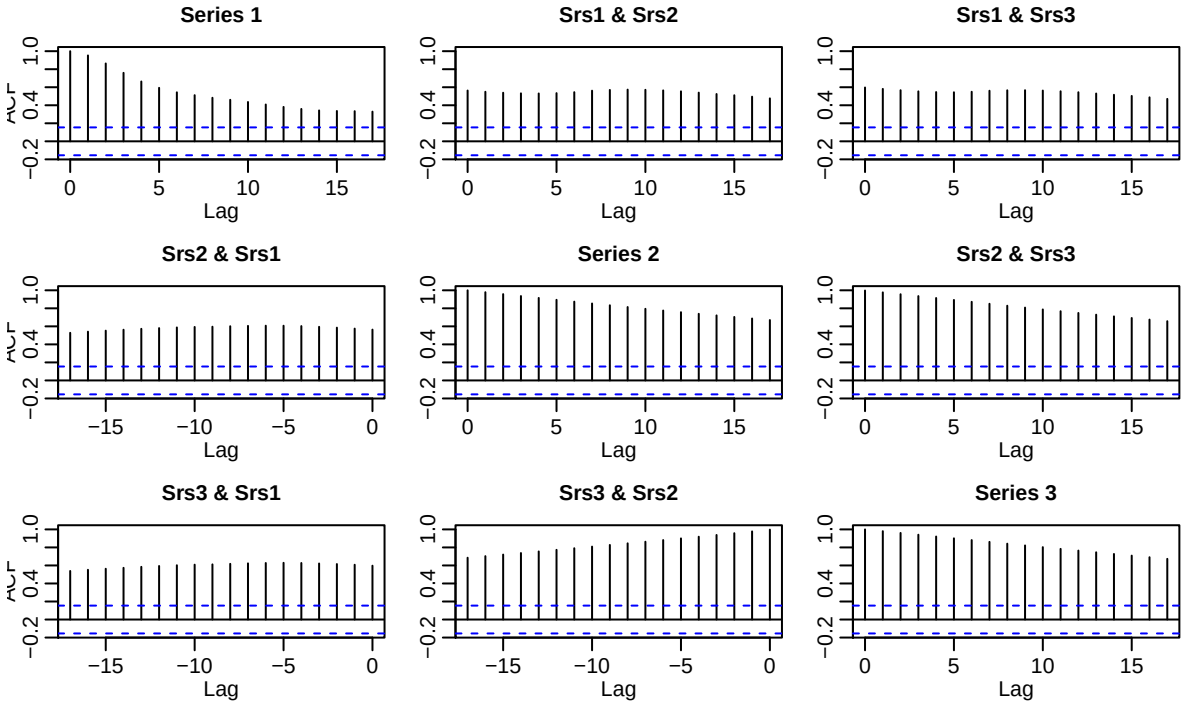
```
plot.ts(econ5)
```



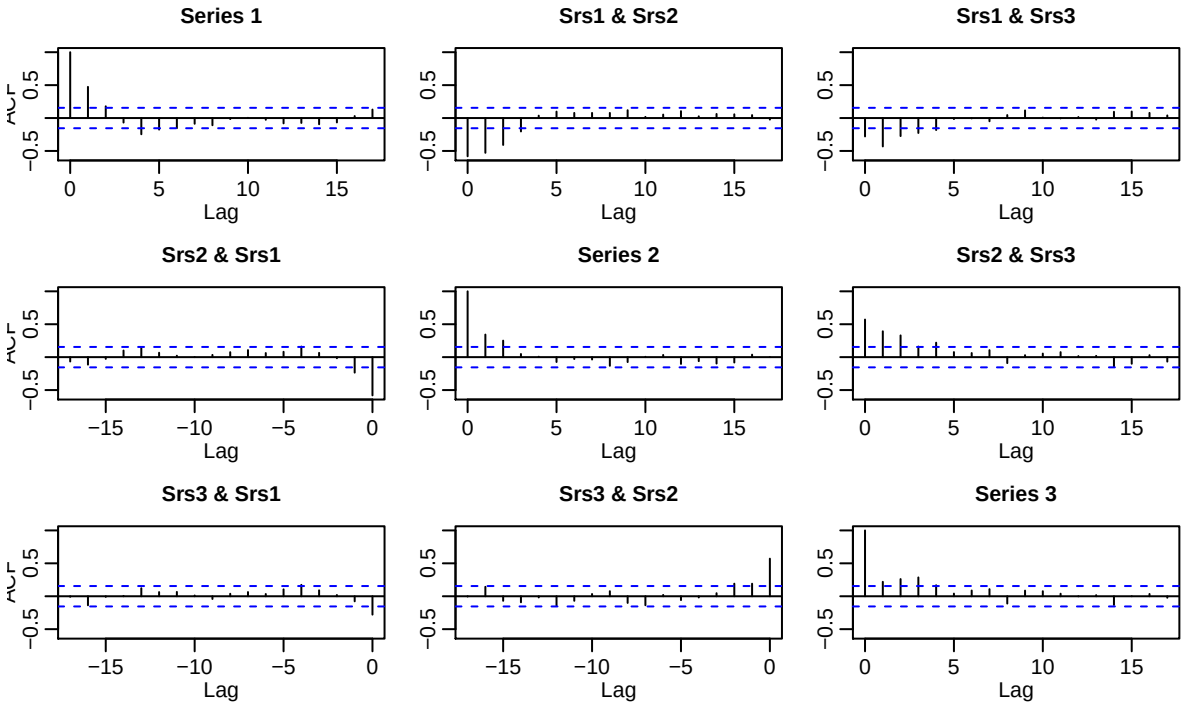

```
acf(econ5)
```




```
x=cbind(econ5$unemp, econ5$gnp, econ5$consum)
acf(x)
```




```
xnew = diff(x)
acf(xnew)
```



```r
fit <- VAR(xnew, p=1)
```

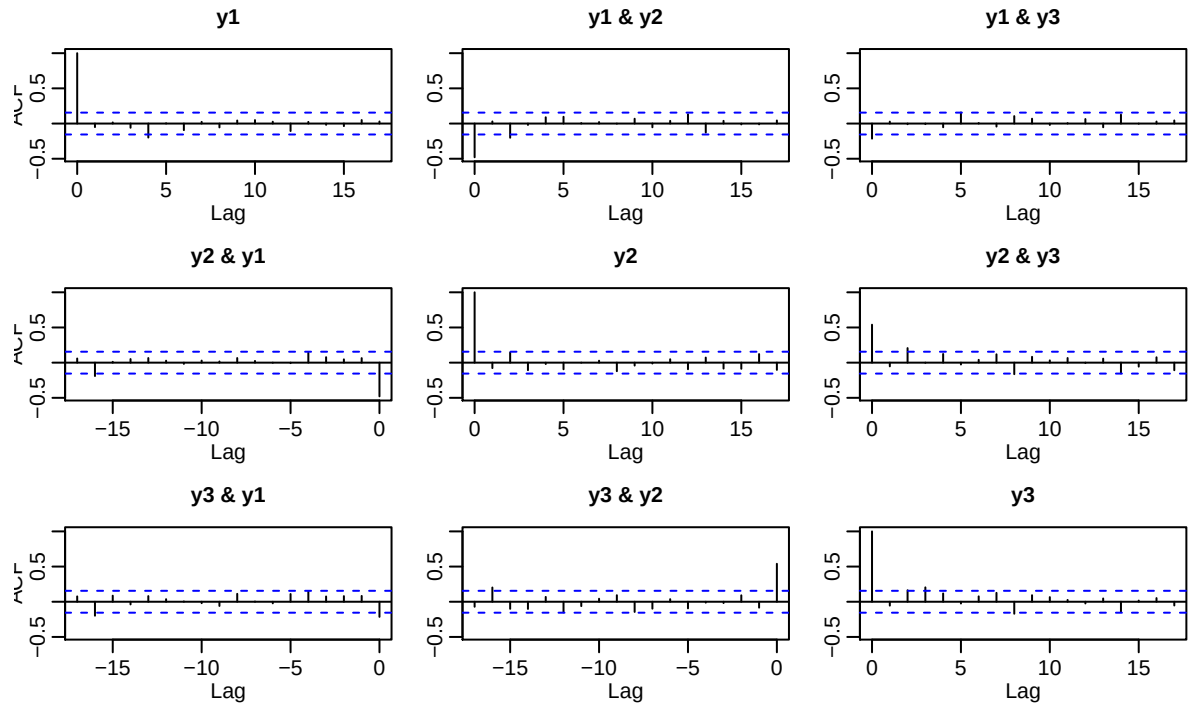
```
## Warning in VAR(xnew, p = 1): No column names supplied in y, using: y1, y2, y3 , instead.
```

```r
summary(fit)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: y1, y2, y3
## Deterministic variables: const
## Sample size: 159
## Log Likelihood: -1378.536
## Roots of the characteristic polynomial:
## 0.3692 0.2636 0.07396
## Call:
## VAR(y = xnew, p = 1)
##
##
## Estimation results for equation y1:
## ===================================
## y1 = y1.l1 + y2.l1 + y3.l1 + const
##
##         Estimate Std. Error t value Pr(>|t|)
## y1.l1  0.270336   0.079524   3.399 0.000859 ***
## y2.l1 -0.004906   0.001814  -2.705 0.007592 **
## y3.l1 -0.008856   0.003271  -2.707 0.007541 **
## const  0.201847   0.048359   4.174 4.97e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.4104 on 155 degrees of freedom
## Multiple R-Squared: 0.3538,  Adjusted R-squared: 0.3413
## F-statistic: 28.29 on 3 and 155 DF,  p-value: 1.207e-14
##
##
## Estimation results for equation y2:
## ===================================
## y2 = y1.l1 + y2.l1 + y3.l1 + const
##
##        Estimate Std. Error t value Pr(>|t|)
## y1.l1  -4.0529     4.5969  -0.882 0.379322
## y2.l1   0.1275     0.1048   1.216 0.225695
## y3.l1   0.6352     0.1891   3.359 0.000984 ***
## const   8.4038     2.7954   3.006 0.003086 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 23.72 on 155 degrees of freedom
## Multiple R-Squared: 0.1797,  Adjusted R-squared: 0.1638
## F-statistic: 11.32 on 3 and 155 DF,  p-value: 9.351e-07
##
```

```
##
## Estimation results for equation y3:
## ===================================
## y3 = y1.l1 + y2.l1 + y3.l1 + const
##
##       Estimate Std. Error t value Pr(>|t|)
## y1.l1  0.91054    2.31331   0.394   0.6944
## y2.l1  0.05696    0.05276   1.080   0.2820
## y3.l1  0.16096    0.09516   1.692   0.0927 .
## const  9.02289    1.40674   6.414 1.63e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 11.94 on 155 degrees of freedom
## Multiple R-Squared: 0.05578, Adjusted R-squared: 0.0375
## F-statistic: 3.052 on 3 and 155 DF,  p-value: 0.03031
##
##
##
## Covariance matrix of residuals:
##         y1      y2      y3
## y1  0.1684  -4.664  -1.051
## y2 -4.6640 562.783 152.798
## y3 -1.0515 152.798 142.522
##
## Correlation matrix of residuals:
##         y1      y2      y3
## y1  1.0000 -0.4791 -0.2146
## y2 -0.4791  1.0000  0.5395
## y3 -0.2146  0.5395  1.0000
```

```
acf(resid(fit))
```




```
serial.test(fit, lags.pt=20)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object fit
## Chi-squared = 169.49, df = 171, p-value = 0.5184
```

```
VARselect(xnew, lag.max=10)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      3      1      1      3
##
## $criteria
##                  1           2           3           4           5
## AIC(n)    8.789060    8.731037    8.702722    8.737190    8.813330
## HQ(n)     8.886910    8.902275    8.947347    9.055202    9.204730
## SC(n)     9.029911    9.152526    9.304849    9.519955    9.776733
## FPE(n) 6562.311024 6193.408922 6022.851768 6238.514417 6739.719001
##                  6           7           8           9          10
## AIC(n)    8.867027    8.912589    8.937385    8.98101    9.036221
## HQ(n)     9.331814    9.450764    9.548948    9.66596    9.794559
## SC(n)    10.011068   10.237269   10.442703   10.66697   10.902815
## FPE(n) 7123.332111 7472.476593 7683.282494 8057.00475 8555.606632
```

```
fit <- VAR(xnew, p=3)
```

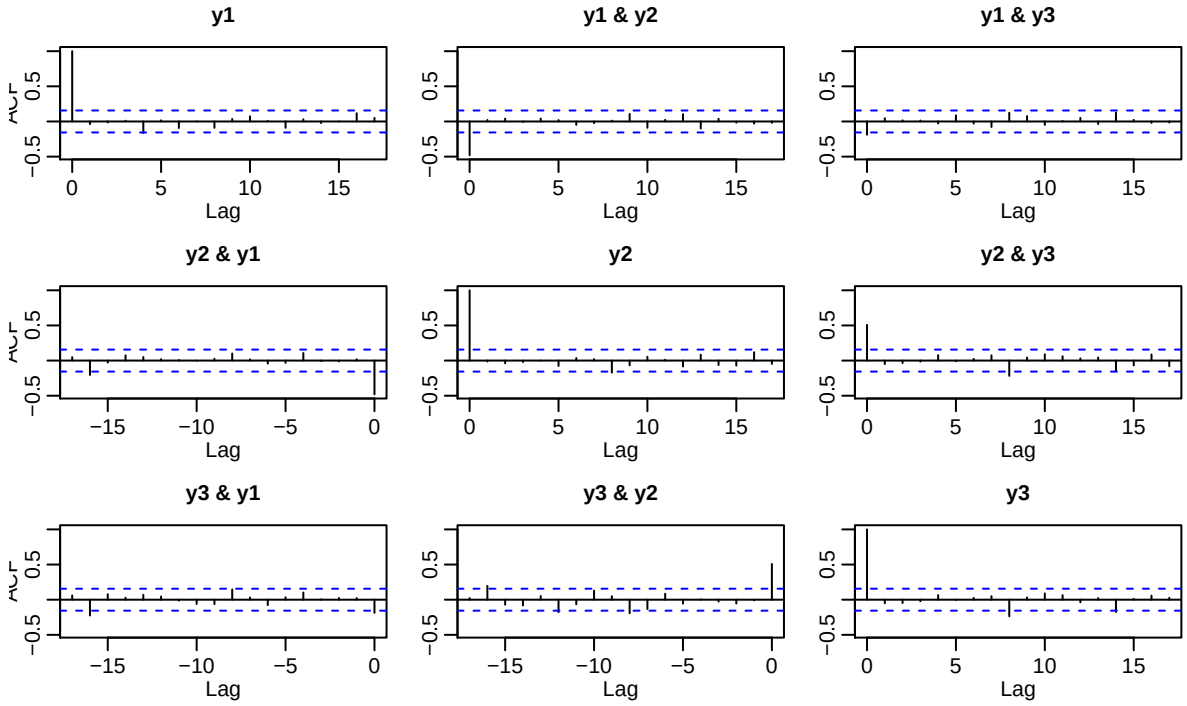
```
## Warning in VAR(xnew, p = 3): No column names supplied in y, using: y1, y2, y3 , instead.
```

```r
summary(fit)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: y1, y2, y3
## Deterministic variables: const
## Sample size: 157
## Log Likelihood: -1337.77
## Roots of the characteristic polynomial:
## 0.6998 0.6466 0.6466 0.576 0.576 0.5308 0.4274 0.4274 0.3814
## Call:
## VAR(y = xnew, p = 3)
##
##
## Estimation results for equation y1:
## ===================================
## y1 = y1.l1 + y2.l1 + y3.l1 + y1.l2 + y2.l2 + y3.l2 + y1.l3 + y2.l3 + y3.l3 + const
##
##        Estimate Std. Error t value Pr(>|t|)
## y1.l1  0.193154   0.092270   2.093 0.038034 *
## y2.l1 -0.004864   0.001817  -2.677 0.008273 **
## y3.l1 -0.007673   0.003221  -2.382 0.018484 *
## y1.l2 -0.098558   0.091168  -1.081 0.281442
## y2.l2 -0.006215   0.001829  -3.398 0.000875 ***
## y3.l2  0.004085   0.003349   1.220 0.224472
## y1.l3 -0.188359   0.082143  -2.293 0.023262 *
## y2.l3 -0.002556   0.001943  -1.315 0.190402
## y3.l3  0.001868   0.003588   0.521 0.603418
## const  0.275169   0.064909   4.239 3.94e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.3973 on 147 degrees of freedom
## Multiple R-Squared: 0.4206,  Adjusted R-squared: 0.3851
## F-statistic: 11.86 on 9 and 147 DF,  p-value: 6.378e-14
##
##
## Estimation results for equation y2:
## ===================================
## y2 = y1.l1 + y2.l1 + y3.l1 + y1.l2 + y2.l2 + y3.l2 + y1.l3 + y2.l3 + y3.l3 + const
##
##       Estimate Std. Error t value Pr(>|t|)
## y1.l1 -2.92867    5.40147  -0.542  0.58850
## y2.l1  0.06108    0.10635   0.574  0.56661
## y3.l1  0.59353    0.18855   3.148  0.00199 **
## y1.l2  7.85573    5.33700   1.472  0.14318
## y2.l2  0.12887    0.10709   1.203  0.23076
## y3.l2  0.40726    0.19605   2.077  0.03952 *
## y1.l3  1.40380    4.80863   0.292  0.77075
## y2.l3 -0.05910    0.11374  -0.520  0.60408
## y3.l3  0.13738    0.21005   0.654  0.51409
## const  2.44238    3.79978   0.643  0.52137
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 23.26 on 147 degrees of freedom
## Multiple R-Squared: 0.2447,  Adjusted R-squared: 0.1985
## F-statistic: 5.292 on 9 and 147 DF,  p-value: 3.011e-06
##
##
## Estimation results for equation y3:
## ===================================
## y3 = y1.l1 + y2.l1 + y3.l1 + y1.l2 + y2.l2 + y3.l2 + y1.l3 + y2.l3 + y3.l3 + const
##
##         Estimate Std. Error t value Pr(>|t|)
## y1.l1  1.228994   2.618434   0.469 0.639506
## y2.l1 -0.003302   0.051557  -0.064 0.949013
## y3.l1  0.106798   0.091404   1.168 0.244530
## y1.l2  4.355152   2.587183   1.683 0.094429 .
## y2.l2  0.049287   0.051913   0.949 0.343964
## y3.l2  0.194420   0.095040   2.046 0.042572 *
## y1.l3  2.014639   2.331048   0.864 0.388851
## y2.l3 -0.035565   0.055136  -0.645 0.519906
## y3.l3  0.345610   0.101823   3.394 0.000885 ***
## const  4.127006   1.841992   2.241 0.026557 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 11.27 on 147 degrees of freedom
## Multiple R-Squared: 0.1952,  Adjusted R-squared: 0.1459
## F-statistic: 3.961 on 9 and 147 DF,  p-value: 0.0001528
##
##
##
## Covariance matrix of residuals:
##         y1      y2       y3
## y1  0.1578  -4.439  -0.8421
## y2 -4.4394 540.893 133.2176
## y3 -0.8421 133.218 127.1075
##
## Correlation matrix of residuals:
##         y1      y2      y3
## y1  1.0000 -0.4805 -0.1880
## y2 -0.4805  1.0000  0.5081
## y3 -0.1880  0.5081  1.0000
```

```
acf(resid(fit))
```

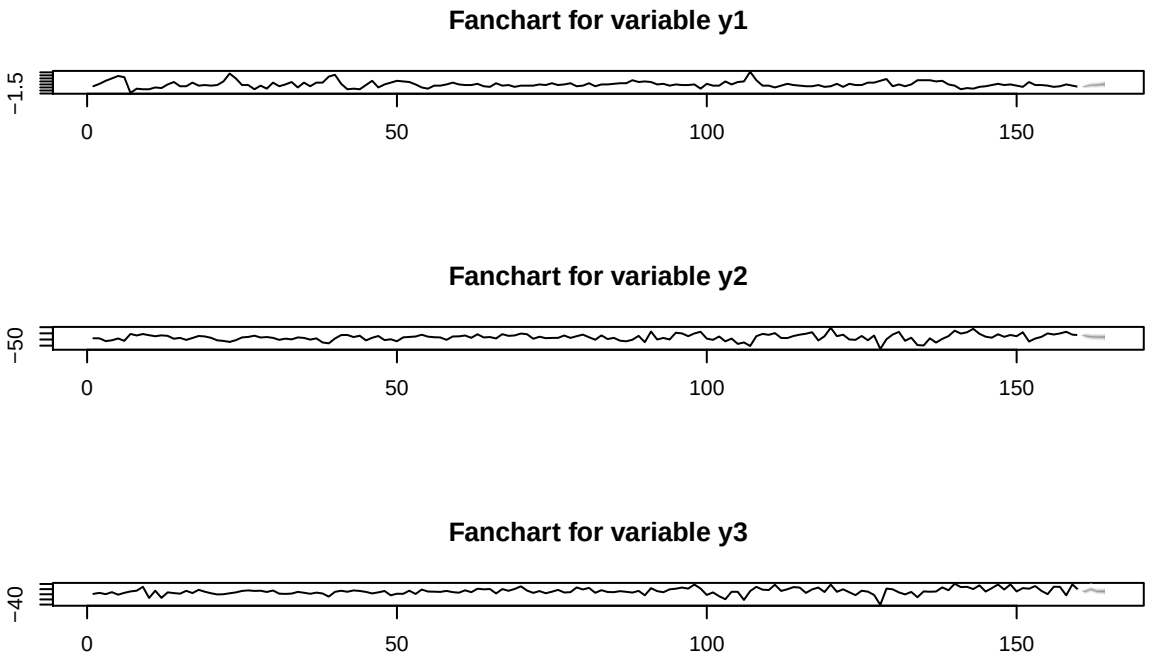



```
serial.test(fit, lags.pt=20)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object fit
## Chi-squared = 122.74, df = 153, p-value = 0.9656
```

b) Predict the unemployment of the next four quarters with 95% confidence intervals.

```
fit.predict = predict(fit, n.ahead=4, ci=0.95)
fanchart(fit.predict)
```


Fanchart for variable y1
−1.5
0
50
100
150
Fanchart for variable y2
−50
0
50
100
150
Fanchart for variable y3
−40
0
50
100
150