Chapter 8: Unit roots and cointegration

GROCER contains the most popular tests of unit roots: augmented Dickey-Fuller, Phillips-Perron, KPSS, (°)Elliott-Rothenberg-Stock(°°) and Schmidt-Philipps (part 1). It contains also the 2 main cointegration methods: the Engle and Granger two-steps methods and Johansen likelihood one (part 2).

1. Unit root tests.

GROCER provides 5 among the most popular unit root tests: the (augmented) dickey-Fuller test (function adf); the Phillips-Perron test (function phil_perr); (°)the Elliott-Rothenberg-Stock test (function ers)(°°); the Schmidt-Phillips test (function schmiphi) and the Kwiatkowski, Phillips, Schmidt, Shin stationarity test (function kpss). The first three take as null hypothesis the existence of a unit root, while the fourth takes as null hypothesis the (trend) stationarity. (°)For the Elliott-Rothenberg-Stock test, approximate p-values are reported, built upon a response surface technology borrowed from Y-W Cheung and K.S Lai (1995) ("Lag Order and Critical Values of a modified Dickey-Fuller test", Oxford Bulletin of Economics and Statistics, vol50, n°3, pp. 411-419., see annex for details).(°°)

All these tests take as first argument the name of the variable. As in other regression functions, its type can be: a ts, a (nx1) vector or a string, equal to the name of a ts or a vector between quotes. If these tests applies to ts, then it is highly recommended, although not compulsory, to set bounds before.

The second argument is always the order of the time trend polynomial: 0 (a constant) and 1 (a time trend) are always allowed. Functions adf and schmiphi allow higher orders (2 to 4). Functions adf and phil_perr also allow order -1, which means the absence of even a constant term.

The third (optional) argument differs among functions. In adf, it corresponds to the lag on the differenced variable in the Augmented Dickey-Fuller regression. If it is set to 0, then ones obtain the Dickey-Fuller original test. In the other 3 functions, it corresponds to the length of the Newey-West window used to correct the estimated variance of residuals.

Lastly, the fourth (optional) argument - the third argument if the user does not want to enter the preceding argument - is 'noprint' if the user does not want to print the results of the regression. Note that if the results of the regression are saved (by typing, for instance, r=adf(…)) then they can be printed later by typing prtunitr(r) or prtres(r) (see chapter 21).

As an example, here are the results of these tests when applied on variable lm1, taken from data base hendryericsson.dat:

```
-->load('SCI/macros/grocer/db/bdhenderic.dat') ;
```
// load Hendry and Ericsson data base

```
-->bounds('1964q3','1989q2')
```
// set bounds

```
-->adf('lm1',1,4)
```
// compute ADF test with a trend

```
ADF estimation results for dependent variable: del(lm1)
estimation period: 1964q3-1989q2
```

```
number of observations: 100
number of variables: 7
R² =  .3355932      ajusted R² = .2927282
Overall F test: F(6,93) = 7.8290795       p-value = 7.805E-07
standard error of the regression:  .0175183
sum of squared residuals:  .0285407
DW(0) =2.0023544
Belsley, Kuh, Welsch Condition index: 4449

variable      coeff      t-statistic p value
lm1(t-1)       -.0180312 -1.3701921  °
del(lm1(t-1))  .1229803  1.1996076    .2333390
del(lm1(t-2))  .2377592  2.3105973    .0230659
del(lm1(t-3))  -.0955856  -.9233646   .3582067
del(lm1(t-4))  .0501985   .4825796    .6305283
cte            .1574329  1.4066493    .1628643
t              .0007295  2.0596255    .0422292


                        *
                    *     *



.......................................
° t-value for variable lm1(t-1) should be compared to the following values:
1% level 5% level 10% level
-4.00519 -3.4611  -3.15517

conclusion: the null hypothesis of a unit root is accepted at a 10% level

                        *
                    *     *


phil_perr('lm1',1)
```

// computes Phillips-Perron test with a trend

```
phillips-perron unit root test for variable: lm1
with a time term of order 1
and estimation period: 1964q3-1989q2
rho test is equal to: -2.5026602

this values should be compared to the following critical values:
1% level 5% level 10% level
-28.612  -20.664  -17.472

conclusion: the null hypothesis of a unit root is accepted at a 10% level

tstat test is equal to: -1.504503

this values should be compared to the following critical values:
1% level 5% level 10% level
-4.0444  -3.452   -3.1512

conclusion: the null hypothesis of a unit root is accepted at a 10% level

                        *
                    *     *


-->schmiphi('lm1',1)
```

// Computes Schmidt-Phillips unit root test

```
Schmidt-Phillips unit root test for variable: lm1
with a time term of order 1
and estimation period: 1964q3-1989q2
rho test is equal to: -3.2413162

this values should be compared to the following critical values:
1% level 5% level 10% level
-23.8    -17.5    -14.6

conclusion: the null hypoyhesis of a unit root is accepted at a 10% level

                        *
                    *     *


-->kpss('lm1',1)
```

// Computes Kwiatkowski, Phillips, Schmidt, Shin stationarity test

```
KPSS stationarity test for variable: lm1
with a time term
and estimation period: 1964q3-1989q2
is equal to:  .1794712
this values should be compared to the following critical values:
1% level 5% level 10% level
 .216     .146     .119

conclusion: the null hypoyhesis of stationarity is accepted at a 1% level, but rejected
at a 5% level

                        *
                    *     *
      (°)
-->ers('lm1',1,4)
```

// Computes Elliott-Rothenberg-Stock test with 4 lags of the differentiated series

```
ers estimation results for dependent variable: lm1
model with constant + trend
estimation period: 1964q3-1989q2
number of observations: 99
number of variables: 5
standard error of the regression: 0.0183953
sum of squared residuals: 0.0318084
DW(0) =2.0331981
Belsley, Kuh, Welsch Condition index: 2

variable     coeff      t-statistic p value
lm1[-1]      -0.0074541 -0.6941866  °
del(lm1[-1]) 0.2336509  2.287119    0.0244325
del(lm1[-2]) 0.3181879  3.0306763   0.0031519
del(lm1[-3]) -0.0244632 -0.2307387  0.8180193
del(lm1[-4]) 0.1556673  1.5004512   0.1368490


                        *
                    *     *


.............................................
° approximate p-value for ERS test is: 0.9699
```

```
conclusion: the null hypothesis of non stationarity is accepted at a 10% level
      (°°)
```

2. Cointegration tests.

GROCER provides 2 cointegration tests. The first one is the ADF tests on residuals, implemented in function cadf, the second one is Johansen method, implemented in function johansen. In each case, the first argument is then the order of the time trend: 0 for a constant term, 1 for a simple time trend and -1 if there is no deterministic part. The second argument is the number of lags of the differentiated residuals in the auxiliary regression. In cadf, the third argument is the name of the endogenous variable, which can be as with all other regression functions, a ts, a (nx1) vector or a string, equal to the name of a ts or a vector between quotes.

The following arguments in cadf are the name of the exogenous (or, more correctly, right hand side) variables. These can be, again as with all other regression functions, a ts, a (nx1) vector or a string, equal to the name of a ts or a vector between quotes, or even a (nxp) matrix.

The following arguments in johansen are the names of all endogenous variables, which can be ts, (nx1) vectors or strings, equal to the name of a ts or a vector between quotes, or even (nxp) matrices. As with other regression functions, if cadf or johansen applies to ts, then it is highly recommended, although not compulsory, to set bounds before.

As an example, here are the results of these two cointegration tests of lm1 on ly1, lp and rnet, from Hendry and Ericsson data base:

```
-->cadf(1,4,'lm1','ly','lp','rnet')
```
// Augmented Dickey-Fuller cointegration test on residuals

```
cadf estimation results for second stage regression of: lm1
on: ly, lp, rnet, cte, t
res = residuals from first stage regression

estimation period: 1965q4-1989q2
number of observations: 95
number of variables: 5
standard error of the regression:  .0453840
sum of squared residuals:  .1853737
DW(0) =1.9720477
Belsley, Kuh, Welsch Condition index: 3

variable     coeff       t-statistic p value
res(-1)       -.1104030 -1.733702    °
del(res(-1))  .0102281   .0917168    .9271269
del(res(-2))  -.0077212  -.0714427   .9432040
del(res(-3))  -.0909822  -.8503629   .3973800
del(res(-4))  .0038596   .0361411    .9712499


                         *
                      *     *


° t-value for variable res(-1): -1.733702
should be compared to the following values:
1% level 5% level 10% level
-5.20113 -4.56724 -4.27167
```

```
conclusion: the null hypothesis of no cointegration is accepted at a 10% level

                          *
                      *     *

-->johansen(1,2,'lm1','ly','lp','rnet')
```
// Johansen cointegration test

```
Johansen estimation results for variables:
lm1, ly, lp, rnet
time order: 1
# of lags: 2

NULL:    Trace Statistic Crit 90% Crit 95%  Crit 99%
r <= 0   56.150759       51.6492  55.2459   62.5202
r <= 1   21.664992       32.0645  35.0116   41.0815
r <= 2   9.135994        16.1619  18.3985   23.1485
r <= 3    .2887852       2.7055   3.8415    6.6349

conclusions from the trace statistics:
at a 10% level, there are 1 cointegration relation(s)
at a 5% level, there are 1 cointegration relation(s)
at a 1% level, there are 0 cointegration relation(s)

NULL:    Max Eigenvalues Statistic  Crit 90%  Crit 95%  Crit 99%
l <= 0   34.485767                  28.2398   30.8151   36.193
l <= 1   12.528998                  21.8731   24.2522   29.2631
l <= 2   8.8472088                  15.0006   17.1481   21.7465
l <= 3    .2887852                  2.7055    3.8415    6.6349

conclusions from the maximal eigenvalues statistics:
at a 10% level, there are 1 cointegration relation(s)
at a 5% level, there are 1 cointegration relation(s)
at a 1% level, there are 0 cointegration relation(s)

                          *
                      *     *
```

3. The unit roots and cointegration functions and their specifications.

adf_____ADF unit root test

CALLING SEQUENCE
[resadf]=adf(namey,p,l,np)

PARAMETERS

INPUT:
* namey: a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* p: order of time polynomial in the null-hypothesis
  - p = -1, no deterministic part
  - p = 0, for constant term
  - p = 1, for constant plus time-trend

p > 1 returns no critical values
* l: # of lags in the adf regression
* np: 'noprint' if the user doesn't want to print the results of the regression
---
OUTPUT:
* resadf: a results tlist with
. resadf('meth') = 'adf'
. resadf('y') = y data vector of the auxiliary regression
. resadf('x') = x data matrix of the auxiliary regression
. resadf('nobs') = # observations
. resadf('nvar') = # variables
. resadf('beta') = bhat
. resadf('yhat') = yhat
. resadf('resid') = residuals of the auxilliary regression
. resadf('vcovar') = estimated variance-covariance matrix of beta
. resadf('sige') = estimated variance of the residuals
. resadf('sigu') = sum of squared residuals
. resadf('ser') = standard error of the regression
. resadf('tstat') = t-stats
. resadf('pvalue') = pvalue of the betas
. resadf('dw') = Durbin-Watson Statistic
. resadf('condindex') = multicolinearity cond index
. resadf('prescte') = boolean indicating the presence or absence of a constant in the regression
. resadf('rsqr') = rsquared
. resadf('rbar') = rbar-squared
. resadf('f') = F-stat for the nullity of coefficients other than the constant
. resadf('pvaluef') = its significance level
. resadf('prests') = boolean indicating the presence or absence of a time series in the regression
. resadf('namey') = name of the y variable of the auxiliary regression
. resadf('namex') = name of the x variables of the auxiliary regression
. resadf('bounds') = if there is a timeseries in the regression, the bounds of the regression
. resadf('like') = log-likelihood of the regression
. resadf('1% level') = 1% critical level
. resadf('5% level') = 5% critical level
. resadf('10% level') = 10% critical level

DESCRIPTION
Computes augmented Dickey-Fuller tests on a time-series vector

Example
1) r = adf('lm1',1,4)
2) r = adf('lm1',0,4)
Examples taken from function adf_d. Example 1 computes the ADF test on lm1 with 4 lags and a time trend. Example 2 computes the ADF test on lm1 with 4 lags and a constant.

c_sja_____critical values for Johansen maximum eigenvalue statistic

CALLING SEQUENCE
[jc]=c_sja(n,p)

PARAMETERS

INPUT:
* n: dimension of the VAR system
* p: order of time polynomial in the null-hypothesis
  p = -1, no deterministic part
  p = 0, for constant term
  p = 1, for constant plus time-trend
  p > 1 returns no critical values

OUTPUT:
jc: a (3x1) vector of percentiles for the maximum eigenvalue statistic for: [90% 95% 99%]

DESCRIPTION
Finds critical values for Johansen maximum eigenvalue statistic.

Example
cvm(i,:) = c_sja(m-i+1,grocer_p)

Example taken from function johansen. Should not have many other uses.

c_sjt_____critical values for Johansen trace statistic

CALLING SEQUENCE
[jc]=c_sjt(n,p)

PARAMETERS

INPUT:
* n: dimension of the VAR system
* p: order of time polynomial in the null-hypothesis
  p = -1, no deterministic part
  p = 0, for constant term
  p = 1, for constant plus time-trend
  p > 1 returns no critical values

OUTPUT:
jc: a (3x1) vector of percentiles for the trace statistic for [90% 95% 99%]

DESCRIPTION
Finds critical values for Johansen trace statistic

Example
cvt(i,:) = c_sjt(m-i+1,grocer_p);

Example taken from function johansen. Should not have many other uses.

cadf_____ ADF statistic for residuals from a cointegrating regression

CALLING SEQUENCE
[rescadf]=cadf(p,l,namey,arg1,...,argn)

PARAMETERS

INPUT:
* p: order of time polynomial in the null-hypothesis
  p = -1, no deterministic part
  p = 0, for constant term
  p = 1, for constant plus time-trend
  p > 1 returns no critical values
* l: # of lagged changes of the residuals to include in regression
* namey: a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* argi: arguments which can be:
  . a time series
  . a real (nx1) vector
  . a string equal to the name of a time series or a (nx1) real vector between quotes
  . the string 'noprint' if the user doesn't want to print the results of the regression
---
OUTPUT:
rcadf: a tlist with
  . all of the arguments of the second stage regression
    and
  . rcadf('cointrel') = tlist with all the arguments of the first stage regression (see ols() for a description of all these arguments)

DESCRIPTION
Computes augmented Dickey-Fuller statistic for residuals from a cointegrating regression, allowing for deterministic polynomial trends

Examples:
1) r = cadf(0,6,'illinos','indiana')
2) r = cadf(1,4,illinos,indiana)
Example 1 is taken from function cadf_d. Illinos and indiania are the names of two variables taken from data base jpl.dat and these names will be used for printings. First entry is set to 0, which means that no trend is imposed in the cointegrating regression. Second entry is set to 6, which means that there are 6 lags to the residuals in the second stage regression. Example 2 imposes a trend in the cointegrating regression, 4 lags on residuals in the second stage regression and the variables are named ‘endogenous’ and ‘exogenous’ respectively when the results are printed.

(°)

ers_____Elliott-Rothenberg-Stock unit root test

CALLING SEQUENCE
resers=ers(grocer_namey,grocer_p,grocer_l,grocer_np)

PARAMETERS

INPUT:
* namey: a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* p: order of time polynomial in the null-hypothesis
    . grocer_p = 0, for constant term
    . grocer_p = 1, for constant plus time-trend
* l: # of lagged changes of the residuals to include in regression
* np: 'noprint' if the user doesn't want to print the results of the regression
---
OUTPUT:
resers = a results tlist with
. resers('meth') = 'ers'
. resers('y') = y data vector of the auxiliary regression
. resers('x') = x data matrix of the auxiliary regression
. resers('nobs') = # observations
. resers('nvar') = # variables
. resers('beta') = bhat
. resers('yhat') = yhat
. resers('resid') = residuals of the auxiliary regression
. resers('vcovar') = estimated variance-covariance matrix of beta
. resers('sige') = estimated variance of the residuals
. resers('sigu') = sum of squared residuals
. resers('ser') = standard error of the regression
. resers('tstat') = t-stats
. resers('pvalue') = pvalue of the betas
. resers('dw') = Durbin-Watson Statistic
. resers('condindex') = multicolinearity cond index
. resers('prescte') = boolean indicating the absence of a constant in the regression
. resers('test p-value') = the (approximate) p-value of the test
. resers('prests') = boolean indicating the presence or absence of a time series in the regression
. resers('namey') = name of the y variable of the auxiliary regression
. resers('namex') = name of the x variables of the auxiliary regression
. resers('bounds') = if there is a timeseries in the regression, the bounds of the regression
. resers('like') = log-likelihood of the regression

DESCRIPTION
Computes Elliott-Rothenberg-Stock (ERS) test on a time-series vector. The p-value have been tabulated using an extension of Y-W Cheung and K.S Lai (1995) ("Lag Order and Critical Values of a modified Dickey-Fuller test", Oxford Bulletin of Economics and Statistics, vol50, n°3, pp. 411-419.) method (see annex for details).

Examples:
```
load('SCI/macros/grocer/db/bdhenderic.dat')                    ;
bounds('1964q1','1989q2')
ers('ly',1,4)
ers('delts(ly)',0,4)
```

Example taken from function ers_d. The nature of series ly is asserted, first by testing if it is trend-stationary (the result is negative), then if its first difference is level-stationary (the result is positive). 4 lags of the first difference of the tested variable are introduced.

(° °)

fuller_tab_____critical values for the Dickey-Fuller statistics

CALLING SEQUENCE
[critical1,critical2]=fuller_tab(p,T)

PARAMETERS

INPUT:
* p : order of the polynomial trend
* T : # of observations

OUTPUT:
* critical1 : (1% - 5% -10%) critical values of the rho test
* critical2 : (1% - 5% -10%) critical values of the tau test

DESCRIPTION
Returns critical values for the Dickey-Fuller statistics

EXAMPLE
1) [critical1,critical2]=fuller_tab(grocer_t,nobs)
2) [c1,c2]=fuller,100)

Example 1 is taken from function adf... Example 2 provides Fuller table for a model with a simple trend and 100 observations.

johansen_____ Johansen cointegration tests

CALLING SEQUENCE
[result]=johansen(p,k,arg1,...,argn)

PARAMETERS

INPUT:
* p: order of time polynomial in the null-hypothesis
  p = -1, no deterministic part
  p = 0, for constant term
  p = 1, for constant plus time-trend
  p > 1 returns no critical values
* k: number of lagged difference terms used when computing the estimator
* namey: a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* argi: arguments which can be:
  . a time series
  . a real (nx1) vector

. a string equal to the name of a time series or a (nx1) real vector between quotes
. the string 'noprint' if the user doesn't want to print the results of the regression

---

OUTPUT:
result =a results tlist:
- result('meth') = 'johansen'
- result('namex') = the name of the variables (m x 1)
- result('x') = matrix of values for the variables (m x 1)
- result('nobs') = # of observations
- result('nvar') = # of variables
- result('nlags') = k
- result('trend') = p
- result('eig') = eigenvalues (m x 1)
- result('evec') = eigenvectors (m x m), where first r columns are normalized coint vectors
- result('lr1') = likelihood ratio trace statistic for r=0 to m-1 (m x 1) vector
- result('lr2') = maximum eigenvalue statistic for r=0 to m-1 (m x 1) vector
- result('cvt') = critical values for trace statistic (m x 3) vector [90% 95% 99%]
- result('cvm') = critical values for max eigen value statistic (m x 3) vector [90% 95% 99%]
- result('ind') = index of co-integrating variables ordered by size of the eigenvalues from large to small

DESCRIPTION
Performs Johansen cointegration tests

Example
result = johansen(0,9,'illinos','indiana','kentucky','michigan','ohio','pennsyvlania','tennesse','westvirginia')

Example is taken from fucntion johansen_d. It provides Johansen cointegration test for the 9 variables in data base jpl.dat. There is no trend (first entry set to 0) and 9 lags of each endogenous variable.

kpss_____KPSS unit root test

CALLING SEQUENCE
[result]=kpss(namey,t,l)

PARAMETERS

INPUT:
* namey: a time series, a real (nx1) vector or a string equal to the
name of a time series or a (nx1) real vector between quotes
* p order of time polynomial in the null-hypothesis
p = 0, for constant term
p = 1, for constant plus time-trend
* l: (optional) truncation lag of the Newey-West window (default: l = floor(5*nobs^0.25))
* np: 'noprint' if the user doesn't want to print the results of the regression

---

OUTPUT:
results tlist with:
- result('meth') = 'kpss'
- result('namey') = namey

- result('y') = y
- result('t') = t
- result('lag(N_W)') = l
- result('test_value') = lm
- result('prests') = prests
- result('1% level') = 1% critical value
- result('5% level') = 5% critical value
- result('10% level') = 10% critical value
- result('bounds') = bounds (if any)

DESCRIPTION
Computes Kwiatkowski, Phillips, Schmidt, Shin stationarity test

Example
1) kpss('lm1',0)
2) kpss('lm1',1)

Examples taken from function kpss_d. Provide KPSS test for variable lm1 taken from data base hendryericsson, with no time trend and a simple time trend respectively

phil_perr_____Phillips-Perron unit-root test

CALLING SEQUENCE
[result]=phil_perr(namey,p,l,np)

PARAMETERS

INPUT:
* namey: a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* p: order of time polynomial in the null-hypothesis
  p = 0, for constant term
  p = 1, for constant plus time-trend
* l: (optional) truncation lag of the Newey-West window (default: l= floor(5*nobs^0.25)
* np (optional) = 'noprint' if the user doesn't want to print the results of the regression
---
OUTPUT:
result tlist with:
- result('meth') = 'phillips-perron'
- result('y') = y data vector
- result('x') = x data matrix
- result('nobs') = nobs
- result('nvar') = nvars
- result('beta') = bhat
- result('yhat') = yhat
- result('resid') = residuals
- result('vcovar') = estimated variance-covariance matrix of beta
- result('sige') = estimated variance of the residuals
- result('sige') = estimated variance of the residuals

- result('ser') = standard error of the regression
- result('tstat') = t-stats
- result('pvalue') = pvalue of the betas
- result('dw') = Durbin-Watson Statistic
- result('condindex') = multicolinearity cond index
- result('prescte') = boolean indicating the presence or absence of a constant in the regression
- result('rsqr') = rsquared
- result('rbar') = rbar-squared
- result('f') = F-stat for the nullity of coefficients other than the constant
- result('pvaluef') = its significance level
- result('rho') = Phillips-perron statistic, autocorrelation/heteroskedasticity corrected value for the unit-root coefficient based on a Dickey-Fuller regression
- result('tstat') = Phillips-perron statistic, autocorrelation/heteroskedasticity corrected t-ratio for the unit-root coefficient based on a Dickey-Fuller regression
- result('prests') = a boolean indicating if the tested variable is a ts or not
- result('cv_rho_1%') = 1% critical values for rho
- result('cv_rho_5%') = 5% critical values for rho
- result('cv_rho_1%') = 10% critical values for rho
- result('cv_tstat_1%') = 1% critical values for tstat
- result('cv_tstat_5%') = 5% critical values for tstat
- result('cv_tstat_10%') = 10% critical values for tstat
- result('bounds') = bounds of the regression (if any)

DESCRIPTION
Computes Phillips-Perron test of the unit-root hypothesis based on a Dickey-Fuller/ Augmented Dickey-Fuller regression.

Example
1) r=phil_perr('lm1',0,15)
2) r=phil_perr('lm1',1)

These Example are taken from fucntion phil_perr_d. Example 1 provides the Phillips-Perron test for variable lm1, whitout trend and with a 15 length for the Newey-West window. Example 2 provides the Phillips-Perron test for variable lm1, whit trend and with a default length for the Newey-West window.

rztcrit_____critical values for the cadf Zt statistic

CALLING SEQUENCE
[crit]=rztcrit(nobs,k,p)

PARAMETERS

INPUT:
* nobs: # of observations
* k: # of variables in x-matrix from cadf()
* p: order of time polynomial in the null-hypothesis
  p = 0, for constant term
  p = 1, for constant plus time-trend
  p = 2, 3 or 4 for higher order time trend polynomial

OUTPUT:
crit: a (6 x 1) vector of critical values: [1% 5% 10% 90% 95% 99%] quintiles

DESCRIPTION
Computes critical values for the Zt statistic used in cadf()

Example
1) rescadf('crit')=rztcrit(rescadf('nobs'),size(rols('x'),2)-grocer_p-1,grocer_p)
2) crit=rzcrit(100,4,1)

Example 1 is a somehow complex example taken from cadf. Example 2 provides the critical values for a cointegrating relation with 100 observations, 4 variables and a simple time trend.

schmiphi_____computes Schmidt-Phillips test

CALLING SEQUENCE
[resulsp]=schmiphi(namey,t,varargin)

PARAMETERS

INPUT:
* namey: a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* t: order of time polynomial in the null-hypothesis
  t = 0, for constant term
  t = 1, for constant plus time-trend
  t = 2,3 or 4 for higher order time trend polynomial

OUTPUT:
* result: results tlist with:
  - result('meth') = 'schmiphi'
  - result('namey') = name of the tested variable
  - result('y') = (nobsx1) vector of endogenous variables
  - result('namey') = name of the tested variable
  - result('nobs') = # of observations
  - result('t') = order of the polynomial trend
  - result('lag(NW)') = # of lags of the Newey-West window
  - result('phi') = value of the phi test
  - result('rho') = rho statistics
  - result('tau') = tau statisctics
  - result('v_rho_1%') = critical value of the rho-test at the 1% level
  - result('v_rho_5%') = critical value of the rho-test at the 5% level
  - result('v_rho_10%') = critical value of the rho-test at the 10% level
  - result('v_tau_1%') = critical value of the tau-test at the 1% level
  - result('v_tau_5%') = critical value of the tau-test at the 5% level
  - result('v_tau_10%') = critical value of the tau-test at the 10% level
  - result('prests') = boolean indicating the presence or absence of a time series in the regression
  - result('bounds') = if there is a timeseries in the regression, the bounds of the regression

DESCRIPTION
Computes Schmidt-Phillips test

Example:
r=schmiphi('lm1',1)

Example taken from function schmiphi_d. Tests if variable lm1 from data base hendryericsson() is trend stationary or has a unit root.

schmiphi_tab_____critical values for the Schmidt-Phillips statistic

CALLING SEQUENCE
[critical1,critical2]=schmiphi_tab(p,T)

PARAMETERS

INPUT:
* p: order of the polynomial trend
* T: # of observations

OUTPUT:
* critical1: (1% - 2.5% - 5% -10%) critical values of the rho test
* critical2: (1% - 2.5% - 5% -10%) critical values of the tau test

DESCRIPTION
Returns critical values for the Schmidt-Phillips statistic.

EXAMPLE
1) [critical1,critical2]=schmiphi_tab(grocer_t,nobs)
2) [c1,c2]=schmiphi_tab(1,100)

Example 1 is taken from function schmiphi... Example 2 provides Schmidt and Phillips table for a model with a simple trend and 100 observations.

ztcrit_____critical values for the ADF Zt statistic

CALLING SEQUENCE
[crit]=ztcrit(nobs,p)

PARAMETERS

INPUT:
* nobs: # of observations
* p: order of time polynomial in the null-hypothesis
p = -1, no deterministic part
p = 0, for constant term
p = 1, for constant plus time-trend
p > 1, for higher order polynomial

OUTPUT:
crit = a (6 x 1) vector of critical values: [1% 5% 10% 90% 95% 99%] quintiles

DESCRIPTION
Returns critical values for the Zt statistic used in adf()

Example:
crit=ztcrit(nobs,p)

This Example is taken from function adf...

Annex: calculation of the p-value of the ERS test

The method is borrowed from Y-W Cheung and K.S Lai (1995) ("Lag Order and Critical Values of a modified Dickey-Fuller test", Oxford Bulletin of Economics and Statistics, vol. 50, n°3, pp. 411-419.).

The tables of p-values for the test with constant and the test with a constant and a time trend have been built independently, applying the same method.

For T=28, 30, 31, 33, 34, 36, 37, 39, 40, 42, 43, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 125, 150, 175, 200, 225, 250, 275, 300, 325, 350, 375, 400, 425, 450, 475, 500, 600 and p=0,1,2,3,4,5,6,8, 100 000 draws of a random walk $\{x_t\}_{t=1,\dots,T}$ have been made (T+50 draws with the 50 first values discarded).

For each draw (i=1 to 100 000) of a given T and a given p, the following procedure is followed:
a) quasi-differentiate x_t, which provides a variable x_t*:
$x_t^* = x_t - (1-7/T)\ x_{t-1}$ (case of a constant)
or:
$x_t^* = x_t - (1-13.5/T)\ x_{t-1}$ (case of a constant+trend)

(with x_0=0)

b) create z_t=1 (case of a constant)
or:
z_t = [1 t] (case of a constant+trend)

c) quasi-diffrentiate z_t as x_t, which provides a variable z_t*

d) regress x_t* on zt*, which provides a variable x_t**

e) then estimate the following model:

$$\Delta x_t^{**} = a_0\ \mathbf{x_t^{**}} + \sum_{i=1}^{p} b_i \cdot \Delta x_{t-i}^{**}$$

f) store in row (t-1)*8+p and column i of matrix M the Student's statistic associated to coefficient a_0.

Once the simulations have been performed, then sort matrix M row by row (each row corresponds to 100 000 simulations of a regression on T periods and p lags of Δx_t^{**}. Then for each p-value π, take the π.100 000 th column of matrix M and perform the following regression:

- t_j = [1 t^(-0.25) t^(-0.5) 1/t (1/t)^2 (p/t)^0.25 (p/t)^0.5 (p/t) (p/t)^2] b + ε_t

(model with constant)

- t_j = [1 t^(-0.25) t^(-0.5) 1/t (p/t)^0.25 (p/t)^0.5 (p/t) (p/t)^2]' b + ε_t

(model with constant+trend)

There are some differences with the regressors used by Y-W Cheung and K.S Lai (1995): in particular, I have introduced lower powers of (1/t) and p/t, because in most cases they did a better job at capturing the shape of the response surface.

The differences with their results can be assessed with a comparison between table 2 in their paper and the one below built from these simulations. They are generally small. In particular, the 2 procedures lead to results that are in both cases very close to those of Elliott-Rothenberg-Stock when they exist. The values tabulated here fluctuate a little less with the number of lags p than Y-W Cheung and K.S Lai's tabulated ones.

Lag order and Finite-Sample Critical Values

Sample Size	Sig. Level	ERS estimate	Response Surface Estimates of Critical Values p=0	p=4	p=6	p=8
50	10%	-2.89	-2.888	-2.744	-2.652	-2.558
	5%	-3.19	-3.193	-3.055	-2.978	-2.910
75	10%	N/A	-2.791	-2.706	-2.685	-2.586
	5%	N/A	-3.086	-3.001	-2.948	-2.896
100	10%	-2.74	-2.737	-2.679	-2.638	-2.593
	5%	-3.03	-3.028	-2.970	-2.930	-2.890
150	10%	N/A	-2.679	-2.647	-2.621	-2.594
	5%	N/A	-2.968	-2.935	-2.910	-2.883
200	10%	-2.64	-2.648	-2.627	-2.610	-2.591
	5%	-2.93	-2.936	-2.915	-2.897	-2.878
500	10%	-2.59	-2.593	-2.588	-2.584	-2.578
	5%	-2.89	-2.881	-2.875	-2.871	-2.865