INSTRUCTIONS TO THE PROGRAMS FOR NONLINEARLY TESTING FOR A UNIT ROOT IN THE PRESENCE OF A BREAK IN THE MEAN

Developed by **Konstantin Gluschenko**
glu@nsu.ru
http://econom.nsu.ru/users/gluschenko
Version: 08.2005

0. Introduction

The programs deal with testing for a unit root, taking account of a (possible) structural break in the mean of a tested time series. The programs are those for EViews 4.1 and later versions. They are provided in ASCII format and are ready for use with EViews; you need only to copy programs to your computer. Two programs use the file *CriticalValues.xls* which is also provided. You need copy this file as well and place it to the default (for your EViews) directory. You may modify this file or create such a file by yourself. To do so, you should know its structure which is described in Section 3 of this document.

The programs are licensed to be available to users to download, copy, use, and modify (except for a commercial use). I hope that in doing so you will acknowledge me as the original creator.

There are two main programs; their functions are as follows.

AR(1) with break.prg estimates nonlinear and Perror-type model allowing for break and perform the respective unit root tests as well as the Dickey-Fuller and Phillips-Perron tests. The program uses file *CriticalValues.xls*.

CDFs of UR statistics with break.prg estimates cumulative distribution functions (CDF), that is, critical values, of the nonlinear and Perron test statistics under the null hypothesis. These results are used to fill file *CriticalValues.xls*.

Besides, there are two additional programs for exploring properties of the nonlinear test.

Experimenting with parameters.prg also estimates CDFs of the above statistics, but in contrast to *CDFs of UR statistics with break* it allows specifying a wide range of parameters of generated time series. And so, this program can estimate CDFs both for the unit root case and for various alternatives.

Power.prg estimates power of the unit root tests allowing for break and that of the Dickey-Fuller test under various alternatives. This program uses file *CriticalValues.xls*.

Instructions for these two programs are provided in Appendix.

1. Models and statistics dealt with

The starting point is an AR(1) process with a break which changes the mean of the process from μ_0 to μ_1 at $t = \theta + 1$:

$$(0) \qquad y_t = \mu_0 + (\mu_1 - \mu_0)B_{\theta t} + \nu_t \ \ (t = 0, 1, \ldots, T),\ \nu_t = (\lambda+1)\nu_{t-1} + \varepsilon_t \ \ (t = 1, \ldots, T),\ \nu_0 = \xi,$$

where $\varepsilon_t \sim$ iid N(0, σ^2), ξ is either a constant or a random variable, and $B_{\theta t}$ is a step dummy such that $B_{\theta t} = 0$ if $t \le \theta$ and $B_{\theta t} = 0$ if $t > \theta$. The first difference $\Delta B_{\theta t} \equiv B_{\theta t} - B_{\theta,t-1}$ can be interpreted as a pulse dummy that takes the value of 1 if $t = \theta + 1$ and 0 otherwise. An alternative way of characterizing the break, which is referred to as "reversed break," is dummy $B'_{\theta t}$ such that $B'_{\theta t} = 1$ if $t \le \theta$ and

$B'_{\theta t} = 0$ if $t > \theta$; note that $B'_{\theta t} = 1 - B_{\theta t}$. (Then the mean of the process changes from μ_1 to μ_0 at $t = \theta + 1$.) The pre-break period θ is referred to as "break point"; $0 < \theta < T - 1$. It can be also represented as the fraction of the sample size: $\Theta = \theta/T$; $0 < \Theta < 1$. To distinguish between them, θ is referred to as "absolute break point," while Θ is referred to as "relative break point." The actual point in time when the break occurs, $\theta + 1$, is referred to as "break date."

Let $\alpha \equiv -\lambda\mu_0$ and $\gamma \equiv \mu_1 - \mu_0$ (the "break height"); from now on, $t = 1,\ldots,T$. Rearranging (0), nonlinear – with respect to coefficients – models are arrived at:

(1a) $\Delta y_t = \lambda y_{t-1} + \gamma B_{\theta t} - \gamma(\lambda+1)B_{\theta,t-1} + \varepsilon_t$;

(1b) $\Delta y_t = \alpha + \lambda y_{t-1} + \gamma B_{\theta t} - \gamma(\lambda+1)B_{\theta,t-1} + \varepsilon_t$.

The first model correspond to the case of $\mu_0 = 0$, the second one is for $\mu_0 \neq 0$.

Using the above notations, the respective specifications of the AR(1) model allowing for break in the spirit of Perron (1990) – the Perron-type models – look like

(2a) $\Delta y_t = \lambda y_{t-1} + \gamma B_{\theta t} - \delta B_{\theta,t-1} + \varepsilon_t$;

(2b) $\Delta y_t = \alpha + \lambda y_{t-1} + \gamma B_{\theta t} - \delta B_{\theta,t-1} + \varepsilon_t$.

Equations (2a) and (2b) are derived with $\gamma \equiv \psi + \delta$ from forms similar to a specification used by Perron (1990):

(3a) $\Delta y_t = \lambda y_{t-1} + \psi B_{\theta t} + \delta\Delta B_{\theta t} + \varepsilon_t$;

(3b) $\Delta y_t = \alpha + \lambda y_{t-1} + \psi B_{\theta t} + \delta\Delta B_{\theta t} + \varepsilon_t$.

The program *AR(1) with break* can estimate the Perron-type models in both forms, so that you can compare your estimates with those reported by other authors, since they commonly use specifications similar to (3a) or/and (3b). (See, e.g., Enders, 1995:248.) To distinguish between these two forms, the program labels (2a) and (2b) as "Perron-type equation," while (3a) and (3b) are labeled as "Original Perron equation" (although they are not, in fact, true original ones).

The programs also use the Dickey-Fuller test equations:

(4a) $\Delta y_t = \lambda y_{t-1} + \varepsilon_t$;

(4b) $\Delta y_t = \alpha + \lambda y_{t-1} + \varepsilon_t$.

The interest is to distinguish between hypotheses H_0: $\lambda = 0$ against H_1: $\lambda < 0$. (In the literature, $\rho \equiv \lambda + 1$ is frequently used instead; then the hypotheses are H_0: $\rho = 1$, and H_1: $\rho < 1$.) In doing so, the t-ratio of λ is used as the test statistic, denoting it by τ (with a subscript which indicates belonging to a particular test) in order to underline that it has a distribution differing from the standard t distribution. The notation is as follows: τ_{0NL} for (1a), $\tau_{\mu NL}$ for (1b), τ_{0P} for (2a)/(3a), $\tau_{\mu P}$ for (2b)/(3b), τ_0 for (4a), and τ_μ for (4b). The parameterized null hypothesis is the same for models (1a) through (4b); it looks like

(5) $\Delta y_t = \lambda y_{t-1} + \gamma\Delta B_{\theta t} + \varepsilon_t$.

For the case of reversed break, equations (1a) through (3b) are the same, except for $B_{\theta t}$ is replaced by $B'_{\theta t}$, and $\Delta B_{\theta t}$ is replaced by $\Delta B'_{\theta t}$. The test statistics for models with no constant are denoted by τ'_{0NL} and τ'_{0P}. The models with constant share the same test statistics: $\tau'_{\mu NL} = \tau_{\mu NL}$ and $\tau'_{\mu P} = \tau_{\mu P}$.

For more details, see Gluschenko (2005).

2. Instructions to the programs

2.0 Executing a Program

To run a program, a standard way is used; see Quantitative Micro Software (2001, 2004). To load a program previously saved on disk, click on `File/Open/Program...`, navigate to the appropriate directory, and click on the desired name. To execute the program, push the `Run` button on a program window. The `Run Program` dialog opens, where you should supply arguments (see subsections below for arguments needed for a particular program). Choose `Quiet` mode.

Set the `Maximum errors before halting` to a number more than 1. Then EViews will continue to execute the program until this number of errors is reached. For example, before creating a new EViews workfile, the program *CDFs of UR statistics with break* (as well as *Experimenting with parameters* and *Power*) checks whether it already exists on disk, trying to load it. When the file does not exist, EViews interprets this as the error; and so, the program would halt, be the maximum number of errors 1. (However, if there is a serious error so that it is impractical for EViews to continue, the program will halt even if the maximum number of errors is not reached.) After all that push `OK` button on the Run dialog window. Figure 1 demonstrates an example of starting the program *CDFs of UR statistics with break* in EViews 4.1.

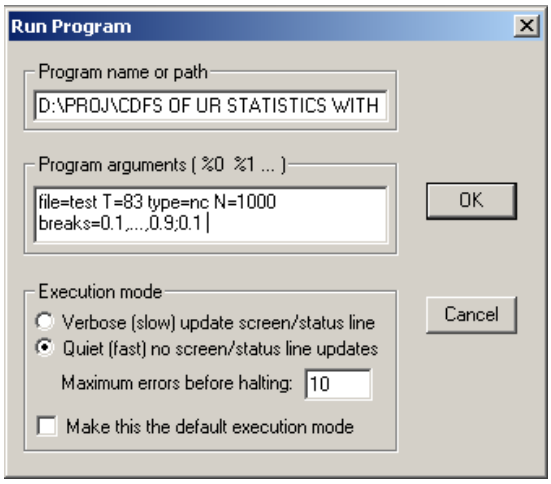


Figure 1. Starting the program *CDFs of UR statistics with break.*

The program *CDFs of UR statistics with break* (as well as *Experimenting with parameter* and *Power*) creates itself an EViews workfile. However, the program *AR(1) with break* needs a workfile with time series to be analyzed. Therefore, you should have such a file opened before starting this program.

CDFs of UR statistics with break (and *Experimenting with parameter* and *Power*, too) can take much time to execute, several hours or even days, depending on a specified number of replications, as well as on the power of computer, operating system, and version of EViews. (While executing, these programs display estimated time left to complete the work in the status line.) Therefore, each of them saves the Evies workfile with its results every hour. Thus, if some problem occurs with your computer, you lose, at the worst, a work of one hour. To continue the work, start the program with the name of the workfile with incomplete work as the (only) argument. You may do the work in parts yourself. Push `F1` or `Esc` key to halt the program and save the relevant workfile to disc. You can continue the interrupted work at any time as above.

All the programs check arguments specified, halting the program and reporting the error found in the status line. However, I cannot guarantee that all possible types of user's errors are foreseen. An unforeseen mistake in arguments can cause some (*a priory* unknown) error while executing the program. In such a case, if you cannot find a mistake in your arguments, please do not hesitate to contact me (as well as when you need some other advice).

It may happen that you make mistake in a parameter, while formally it is correct. In order that you can check correctness of your parameters, the programs show a window, table `Description`, that contains interpreted values of parameters. (In the program *Power*, this table is called `Results`, containing both specified parameters and results of execution.) When you find a mistake, you can halt the program (pushing `F1` or `Esc`), and rerun it with correct arguments. If the program has already saved its results on disc (i.e., if it has worked more than one hour), delete this file from disc. (This does not concern the program *AR(1) with break* which works fast. Nevertheless, it also creates the table `Description` so that you can check whether specified arguments are correct.)

2.1 AR(1) with break

Function. This program estimates models (1a) through (4b) – depending on specification – on a user-specified time series and performs the nonlinear unit root test allowing for break, the Perron test (if specified), and the Dickey-Fuller and Phillips-Perron tests (if specified). The break date can be either user-specified or estimated by the program. The program uses a user-provided EViews workfile with time series to be analyzed as well as the file *CriticalValues.xls* that contains critical values of the above tests. (These values are obtained with the use of the program *CDFs of UR statistics with break*.) See Section 3 of this document for the description of this file. (It is possible to use a different file with critical values; it should have a certain structure described in Section 3 of this document.)

Algorithm. At first, the program finds critical values of the nonlinear and Perron tests for T of the specified time series or a user-specified sample size, and for user-specified break date(s) in the file of critical values. If the file does not contain information for a given sample size, the program reports the nearest sample sizes in the file and halts. (In the next run, you may specify one of these sample sizes for unit root testing.) If the break dates are not specified, the program uses all those from the file of critical values for a given sample size. In the case that the file does not contain critical values for specified break dates, the program uses those for the nearest relative break points, Θ. Depending on the type of model, the program estimates model (1a)/(1b); if specified, model (2a)/(2b) or (3a)/(3b) is estimated; again, if estimated, the program performs the Dickey-Fuller and Phillips-Perron tests, using model (4a)/(4b). If a range of break dates is specified, the program estimates model(s) for each break date from this range and then chooses the date yielding the minimal sum of squared residuals, so estimating the break date. (The break dates for the nonlinear model and the Perron-type model may differ.) Using data from the file of critical values, the program performs the nonlinear unit root test, as well as the Perron test, if specified (i.e., finds a p-value corresponding to estimated value of a statistic), using both estimated τ (τ_{0NL}/$\tau_{\mu NL}$ and, maybe, τ_{0P}/$\tau_{\mu P}$) and $Z_t(\tau)$ which is the Phillips (1987) transformation of τ. (Thus, the latter is an analogue of the Phillips-Perron test.[1]) The Newey-West (1994) automatic bandwidth selection method with the Bartlett spectral kernel is used for this transformation.

[1] Note that for models (1·) and (2·)/(3·), the validity of such a testing is not proved either theoretically or empirically. However, a (moderate) number of experiments give an impression that it does work.

Results. The program generates a number of EViews objects with outputs and saves them to the user's workfile. (There is a special mode of the program that cleans the file from these objects.) These outputs are displayed on the screen as well. Depending on arguments specified for a particular run, the set of outputs can differ. The full set is the following.

Table `Description` which consists of two parts. The first part reports arguments as they are specified and as they are used by the program (including values taken by default of from the previous run). The second part reports the sample size of analysed series and the sample size used for testing, and lists the break dates (across which the break date is estimated) and the respective absolute and relative break points. (If the break date is specified by user, the list contains the only item.) Two values are reported; the first one is as specified (either by user or by default); the second value is that for which the unit root test statistics are available. If the file of critical values contains all needed data, these two values coincide. If it does not, the program takes statistics for available Θ that is the nearest to the specified one. It may be the third part of `Description` that reports non-critical errors, if they occur. Figure 2 provides an example of `Description`.

Table: DESCRIPTION Workfile: EMPIRICAL DATA

View | Procs | Objects | Print | Name | Edit+/- | Font | InsDel | Width | Numbers | Justify | Lines | Grid+- | Title

	A	B	C	D	E	F	G	H	I
1	ARGUMENTS		SERIES=_B TYPE=-NC PERRON=YES						
2	series=	_B							
3	type=	-NC							
4	break=	1998:08-1999:02;1	(by default)						
5	CVfile=	CriticalValues.xls	(by default)						
6	T=	83	(by default)						
7	Perron=	yes,pt							
8	DF=	yes	(by default)						
9	keep=	yes	(by default)						
10									
11	BREAK POINTS:								
12	SAMPLE: t=0,...,83; TEST STATISTICS FOR: t=0,...,83								
13	Date:	specified	1998:08	1998:09	1998:10	1998:11	1998:12	1999:01	1999:02
14		test statistics for	1998:08	1998:09	1998:10	1998:11	1998:12	1999:01	1999:02
15	Theta:	specified	54	55	56	57	58	59	60
16		test statistics for	54	55	56	57	58	59	60
17	Theta/T:	specified	0.651	0.663	0.675	0.687	0.699	0.711	0.723
18		test statistics for	0.651	0.663	0.675	0.687	0.699	0.711	0.723
19									

Figure 2. `Description` reported by the program *AR(1) with break.*

If you use statistics for, say, $T = 100$ rather than for $T = 83$, the lower panel of the table would look like

BREAK POINTS:				
SAMPLE: t=0,...,83; TEST STATISTICS FOR: t=0,...,100				
Date:	specified	1998:08	1998:09	
	test statistics for	1998:12	1998:12	
Theta:	specified	54	55	
	test statistics for	58	58	
Theta/T:	specified	0.651	0.663	
	test statistics for	0.700	0.700	etc.

That is, an available unit root test statistic for Θ nearest to 0.651, 0.663, etc., is that for $\Theta = 0.7$. (Currently, there are statistics for Θ = 0.1, 0.2,…, 0.9 in *CriticalValues.xls* for T =100.) The corresponding absolute break points are calculated as $\theta = 0.7 \cdot 83$ rounded (where 83 is the sample size of the analyzed series), which is equivalent to the break date 1998:12.

Equation `Nonlinear` is a standard EViews output for regression (1a)/(1b). It contains parameter

estimates, a number of standard statistics, and some additional information. In this output, `C(1)` is λ, `C(2)` is α, and `C(3)` is γ. `C(2)` does not appear if estimated equation is (1a).

Equations `Perron_type` and `Original_Perron` are standard EViews outputs for regressions (2a)/(2b) and (3a)/(3b), respectively. Only one of them, if any, appears, depending on specification. In the first output, `C(1)` is λ, `C(2)` is α, `C(3)` is γ, and `C(4)` is δ. In the second output, `C(1)` is λ, `C(2)` is α, `C(4)` is δ, and `C(5)` is ψ. `C(2)` does not appear if estimated equation is (2a) or (3a).

Table `UR_tests` reports results of testing for a unit root. These are: estimated/user-specified break date, τ-statistic and the relevant p-value, adjusted τ-statistic $Z_t(\tau)$ and the relevant p-value, and the number of lags found (with the use of the Newey-West method) while performing the Phillips transformation. If estimation of the Perron-type model is not specified, the table reports results for the nonlinear test only. Figure 3 provides an example.

Table: UR_TESTS Workfile: EMPIRICAL DATA

View | Procs | Objects | Print | Name | Edit+/- | Font | InsDel | Width | Numbers | Justify | Lines | Grid+- | Title

	A	B	C	D	E	F	G
1	TEST ON _B FOR A UNIT ROOT SUBJECT TO BREAK						
2							
3		Nonlinear model		Perron-type model			
4	Break at:	1998:12		1998:12			
5		statistic	p-value	statistic	p-value		
6	t-stat	-2.058	0.052	-2.609	0.062		
7	Z(t-stat)	-2.386	0.029	-2.653	0.057		
8	Bandwidth	4		2			
9							
10	't-stat' is the t-ratio of C(1) (lambda).						
11	'Z(t-stat)' is its Pillips transformation, bandwidth beeing the number of lags used.						
12	In the regression outputs, C(1) stands for lambda (autoregression coefficient - 1),						
13	C(2) stands for alfa (constant),						
14	C(3) stands for gamma (break height),						
15	C(4) stands for delta (in both versions of Perron"s regression),						
16	and C(5) stands for psi (in the original Perron"s regression).						
17							

Figure 3. Table `UR_tests` reported by the program *AR(1) with break*.

Tables `Dickey-Fuller_test` and `Phillips-Perron_test` are standard EViews outputs for these tests. They contain both results of testing and estimates of (4a)/(4b).

The program also creates not displayed table `Settings` which contains arguments used in the current run of the program. In the next run(s), the program takes unspecified arguments from this table. And so, you need specify only arguments that have other values. Its contents looks, for example, like

SETTINGS:	
series=	_B
type=	-nc
break=	1998:08-1999:02;1
CVfile=	CriticalValues.xls
T=	100
Perron=	yes,pt
DF=	yes
keep=	yes

Besides, the program stores a number of series `Break`θ, where θ is some number (the break point). These are break dummies for each break point used in the current and earlier estimations. You might need them for some additional analyses. While program processes the arguments, it creates a table `Arguments`. If there is an error in the argument list, the program halts and displays this table; otherwise the program deletes it.

Arguments. Arguments of the program may follow in any order. All arguments are optional. If an argument is omitted, the program uses either a default value or a previous user-specified value from `Settings`. However, you should specify at least the series to be analyzed if there is no `Settings` in your EViews workfile.

Hereafter, the following notations are used in the description of arguments: <...> means some value of argument or its part, [...] means an optional part of argument, | means "or", and `and so on` means replication of a previous expression; keywords are in thick print. Except for the filename, no blanks can be tolerated within arguments.

The program uses the following 8 arguments:

`series=`<*series name*> is the name of a series (from your Eviews workfile) to be analyzed.

`type=``[-]nc|c` is the type of equations to be estimated: `nc` means equations with no constant, (1a), (2a)/(3a), and (4a); `c` means equations with constant, (1b), (2b)/(3b), and (4b). Minus means a break to be reversed. By default, `type=c`.

`break=`<$date_1$>[-<$date_2$>[;<*step*>]] is a break date, $\theta + 1$, or a range of break dates; <$date_1$> and <$date_2$> should be specified as dates (according to the frequency in your workfile) rather than the observation numbers. If range <$date_1$>-<$date_2$> is specified, the program runs estimations for break points from <$date_1$> through <$date_2$> with increment <*step*>; then it takes the date that yields the minimal sum of squared residuals. If <$date_2$> is omitted, the only break point <$date_1$> is used. If the argument is omitted, the first and last break points available in the file of critical values, θ_a and θ_b, are used; then <$date_1$> = $\theta_a + 1$ and <$date_2$> = $\theta_b + 1$. The default value of <*step*> is 1.

`CVfile=`[<*path*>]<*file name*>] is the name of an (Excel) file with critical values of the tests (if there are blanks in the path or/and file name, enclose the full name in quotes). By default, `CVfile=CriticalValues.xls`. (It should be placed to the default – for your EViews – directory; otherwise specify it.)

`T=`<*number*> is the sample size for test statistics. (Note that $t = 0,...,T$.) This argument is necessary if the file of critical values does not contain the sample size that the processed series has. Then you should specify a sample size you wish to use instead for testing. `T=10000` stands for asymptotic values. By default, the sample size in your workfile (minus one) is taken.

`Perron=``yes[,pt|,or]|no` enables/disables estimating models (2a)/(2b) or (3a)/(3b); `pt` means (2a)/(2b), while `or` stands for (3a)/(3b). By default, `Perron=no`; in the `yes` option, the default is `pt`.

`DF=``yes|no` enables/disables the Dickey-Fuller and Phillips-Perron tests. By default, `DF=yes`.

`keep=``yes|no` is the indication whether to keep the program objects in your workfile. If yes, the settings are used instead of default values unless replaced by new arguments. If `keep=no`, the program removes its objects (including outputs) from your workfile. The only use of this is to clean your workfile (running the program with the only argument `keep=no`) when you complete your work. By default, `keep=yes`.

Examples. Let you have a file with time series `_a`, `_b`, and `_c` spanning 1994:01 through 2000:12 (then $T = 83$). Arguments

```
series=_a type=-nc break=1998:08-1999:02;1 Perron=yes,pt DF=yes
```

imply that series `_a` should be analyzed; in doing so, model (1a) with the reversed break should be used; the break date should be searched in the interval of 1998:08 through 1999:02 with the one-month step; model (2a) should be estimated as well; and the Dickey-Fuller and Phillips-Perron tests should be performed. Omitted arguments are, by default, the following: `CVfile=CriticalValues.xls`

`T=83 keep=yes`. That is, *p*-values of the nonlinear and Perron tests should be taken for $T = 83$ from file `CriticalValues.xls`; results and some additional program objects should be kept.

If you specify

```
series=_b break=1998:10 Perron=yes,or DF=no  ,
```

then the program will analyze series `_b`, provided that the break date is 1998:10; by default, (`type=c`). The respective nonlinear model is (1b); the Perron-type model should be estimated in form (3b). The Dickey-Fuller and Phillips-Perron tests will not be performed. Omitted arguments are, by default, the following: `type=c CVfile=CriticalValues.xls T=83 keep=yes` (Hence, according to the default specification, the models with intercept will be estimated.)

With arguments

```
series=_c type=-c T=100 CVfile="c:\New stats\cv.xls" Perron=no DF=no  ,
```

series `_c` will be analyzed, using equation (1b) with reversed break (in fact, provided that constant is included, results do not depend on whether ordinary or reversed break is used); no estimation of a Perron-type equation and no ordinary unit root tests will be performed. Critical values for the nonlinear test will be taken for $T = 100$ from file `cv.xls`. Since the break argument is not specified, the date of the break will be searched within interval from the first through the last break date available in this file; the search will step across break dates for which the statistic is available.

2.2 CDFs of UR statistics with break

Function. This program estimates cumulative distribution functions (CDF) of the nonlinear and Perron test statistics under the null hypothesis for a specified set of break points, as well as CDFs of the Dickey-Fuller statistics τ_0 and τ_μ. The number of replications and the type of equations are user-specified.

Algorithm. In each replication, the program generates $\{\varepsilon_t\}_{t=1,\dots,T} \sim$ iid N(0,1) and construct a random walk $y_t = y_{t-1} + \varepsilon_t$ (t=1,…,T) with $y_0 = 0$. Then it estimates (4a) and (4b), and, depending of a specified type of equations, either (1a) and (2a) or (1b) and (2b) for each of specified break points. The breaks can be mixed. That is, in some of the break points, the breaks may be specified as ordinary ones, while they may be specified as reversed in other break points. Having reached the specified number of replications, the program computes the CDFs, using 1000 quantiles plus one more for probability 0. (The program does not check accordance between the numbers of quantiles and replications! Thus, having specified a small number of replications, you can obtain unsatisfactory results.)

Results. The program generates and saves to disc an EViews workfile with a user-specified name. The content of this file is the following (N denotes the number of replications, M denotes the number of break points).

Table `Description` which reports conditions of a given estimation, namely, the type of models (with constant/with no constant), sample size, number of replications (if the work is not completed, the number of replications done is reported as well), and the list of break points. Besides, it reports the time when program has started and finished, and elapsed time, as well as the list of arguments as they have been specified. Figure 4 provides an example.

The main results are contained in the following matrices.

Matrix `cdf_t_nl`, 1001 × (M +1), is the CDFs of τ_{0NL} or $\tau_{\mu NL}$ for each specified break point.

Matrix `cdf_t_p`, 1001 × (M +1), is the CDFs of τ_{0P} or $\tau_{\mu P}$ for each specified break point.

Matrix `cdf_df_nc`, 1001 × 2, is the CDFs of τ_0.

Matrix `cdf_df_c`, 1001 × 2, is the CDFs of τ_μ.

The first column of these matrices is *p*-values from 0 through 1 with increment 0.001. The rest columns are CDFs themselves. The number of a column corresponds to the number of a break point in the `Description` plus one.

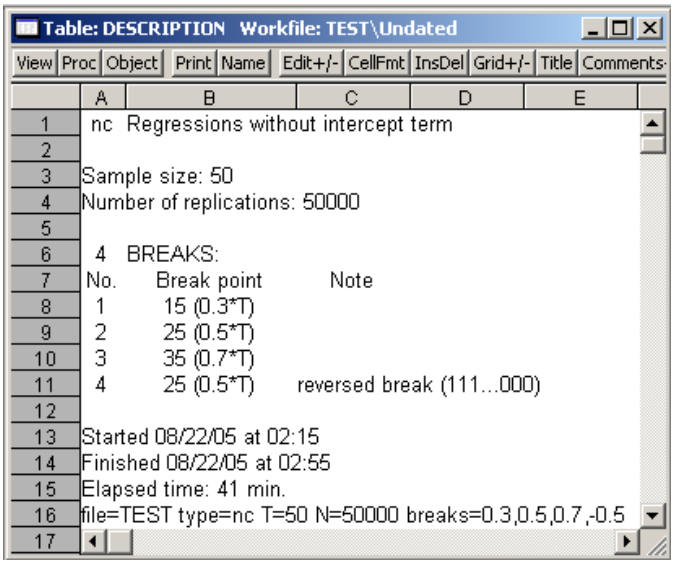


Figure 4. `Description` reported by the program *CDFs of UR statistics with break.*

The full sets of estimated statistics are saved in the following matrices. (This information may be useful for some analytical work, e.g., to plot the probability density function of a statistic.)
Matrix `t_nl`, $N \times M$, is the full set of estimated τ_{0NL} or $\tau_{\mu NL}$ for each specified break point.
Matrix `t_p`, $N \times M$, is the full set of estimated τ_{0P} or $\tau_{\mu P}$ for each specified break point.
Matrix `t_df_nc`, $N \times 1$, is the full set of estimated τ_0.
Matrix `t_df_c`, $N \times 1$, is the full set of estimated τ_μ.
The number of a column corresponds to the number of a break point in the `Description`.

Until the program completes, it saves in the workfile additional objects that are needed to continue the work if it has been interrupted. Do not remove or change these objects.

Arguments. Arguments of the program may follow in any order, but all 5 of them should be provided except for the case when the workfile already exists on disc (e.g., estimations are processing in parts). Only the filename should be provided in such a case (then keyword **`file=`** may be omitted).

The program uses the following 5 arguments:

`file=`[<*path*>]<*file name*> is the workfile name (if there are blanks in the path or/and file name, enclose the full name in quotes, e.g., `file="c:\my path\my file.wf1"`).
`T=`<*number*> is the sample size ($t = 0,...,T$). (In fact, in the workfile and program, it will be $t = 1,...,T+1$.)
`N=`<*number*> is the number of replications.

`type=``nc|c` is the type of equations to be estimated: `nc` means equations with no constant, (1a) and (2a); `c` means equations with constant, (1b) and (2b).
`breaks=`[`-`]<*number A*>`,...,`<*number B*>`;`<*number C*>

| [−]<*number*$_1$>[,[−]<*number*$_2$>][,[−]<*number*$_3$>[,`and so on`]]]

is the list of break points. The upper form specifies them as a sequence from <*number A*> through <*number B*> with increment <*number C*>; the lower one lists any number of specific points. The break points may be specified either as relative or as absolute. However, these types should not be mixed in the same list. Minus means a break(s) to be reversed. In the upper form of the argument, − concerns all break points; in the lower one, it concerns the given break point. (Note that <*number C*> may also be negative; this implies just a negative increment. Then <*number A*> should be greater than <*number B*>.)

For example, `breaks=10,…,90;10` specifies 9 break points: θ = 10, 20, 30, 40, 50, 60, 70, 80, 90. With T = 100, an equivalent specifications are `breaks=0.1,…,0.9;0.1` and `breaks=0.9,…,0.1;-0.1`. The same for reversed breaks is `breaks=-0.1,…,0.9;0.1` and `breaks=-0.9,…,0.1;-0.1`. Alternatively, you may list all the break points: `breaks=-0.1,-0.2,-0.3,-0.4,-0.5,-0.6,-0.7,-0.8,-0.9`. Figure 4 provides an example of mixed breaks, `breaks=0.3,0.5,0.7,-0.5`, which means that the statistics will be estimated with $B_{\theta t}$ for Θ = 0.3, 0.5, and 0.7, and again for Θ = 0.5 with $B'_{\theta t}$.

Examples. Arguments

```
file=50c type=c T=50 N=200000 breaks=0.1,…,0.9;0.1
```

imply that the critical values of the tests including constant term will be estimated for sample of size 50 with 9 break points: θ = 5, 10, 15,…, 40, 45, using 200,000 replications. The results will be saved to new EViews workfile named *50c*.

If you specify

```
file="test 1" type=nc breaks=-75,25  N=10000 T=100   ,
```

then the program estimates critical values of the tests with no constant for sample of size 100 and two break points: the first is θ = 75 (Θ = 0.75) with reversed break, and the second is θ = 25 (Θ = 0.25) with ordinary break; in doing so, the program uses 10,000 replications.

Specification

```
file="D:\Projects\UR statistics.wf1"
```

means that the file *UR statistics.wf1* already exists on disk and contains incomplete work on estimation of critical values (parameters of estimation as well as the number of replications performed being stored in this file); the program will continue estimation until reaches *N* specified at the very first start of this work.

3. The File of Critical Values

This file contains cumulative distribution functions (critical values) of the unit root test statistics τ_{0NL}, τ'_{0NL}, $\tau_{\mu NL}$, τ_{0P}, τ'_{0P}, $\tau_{\mu P}$, τ_0, and τ_μ for a certain number of sample sizes and for a number of break points. The statistics are obtained with the use of program *CDFs of UR statistics with break*. This file is used by program *AR(1) with break* (as well as by *Power*) to find *p*-values of unit root test, namely, τ_{0NL}/τ'_{0NL}, $\tau_{\mu NL}$, τ_{0P}/τ'_{0P}, and $\tau_{\mu P}$ (program *Power* uses τ_{0NL}/τ'_{0NL}, $\tau_{\mu NL}$, τ_{0P}/τ'_{0P}, $\tau_{\mu P}$, τ_0, and τ_μ).

The structure of the file is not rigid. Therefore, it contains a required sheet `Map` with a description of the file organization. The names of the sheets with statistics are fixed. They are: `Nonlin. tau-0` for τ_{0NL}, `Nonlin. tau-0(r)` for τ'_{0NL}, `Nonlin. tau-mu` for $\tau_{\mu NL}$, `Perron tau-0` for τ_{0P}, `Perron tau-0(r)` for τ'_{0P}, `Perron tau-mu` for $\tau_{\mu P}$, `DF tau-0` for τ_0, and `DF tau-mu` for τ_μ. (Some of these sheets may be absent if you do not need respective statistics.)

The description of the file organization is the following.

Cell B2 of Map indicates the number of column with *p*-values. (For some reasons, the numbers of columns is used rather than their letterings. The provided file *CriticalValues.xls* contains in its sheet Map a transition table between lettering of columns and their numbers.) The *p*-values should be in the same column across all sheets with the statistics.

Cell B3 indicates the number of raw with break points. Break points should be in the same raw across all sheets with the statistics.

Cell B4 indicates the total number of sample sizes for which statistics are available in the file. This does not imply that there should be the full set of the statistics for each sample size. There may be different sets of selected statistics for different sample sizes.

Cell B5 refers to the first raw of the statistics; and

cell C6 refers to their last raw. The statistics should begin from the same raw across all sheets with the statistics. However, the statistics need not end in the same raw. Its number in cell C6 is the maximal one; the length of columns with the statistics may vary across sheets. (The actual indication of the end for the programs is *p*-value equaling 1.)

At last, cell B7 refers to the first row of the data allocation table. (This is reference to the first raw with data, and not to the heading of the table.)

While the first raw of the data allocation table may be arbitrary, its first column is fixed. It is column A. In this column of the table, the sample sizes – for which statistics are stored – are listed. The subsequent columns report the number of the first and last columns of a given statistic for each of the 6 statistics associated with breaks, and the number of column for the Dickey-Fuller statistics. (The number of columns with a particular statistic, last column – first column + 1, is the number of break points for which this statistic is available. This figure as well as the break points themselves may vary across sample sizes and across different statistics with the same sample size. However, when break points for nonlinear and Perron statistics does not coincide in number or in composition, the program *AR(1) with break* skips the Perron test.) Namely,

columns B and C contain the number of the first and last columns with τ_{0NL} (i.e., in the sheet Nonlin. tau-0),

columns D and E contain the number of the first and last columns with τ'_{0NL} (i.e., in the sheet Nonlin. tau-0(r)),

columns F and G contain the number of the first and last columns with $\tau_{\mu NL}$ (i.e., in the sheet Nonlin. tau-mu),

columns H and I contain the number of the first and last columns with τ_{0P} (i.e., in the sheet Perron tau-0),

columns J and K contain the number of the first and last columns with τ'_{0P} (i.e., in the sheet Perron tau-0(r)),

columns L and M contain the number of the first and last columns with $\tau_{\mu P}$ (i.e., in the sheet Perron tau-mu),

column N contains the number a column with τ_0 (i.e., in the sheet DF tau-0),

and column O contains the number a column with τ_μ (i.e., in the sheet DF tau-mu).

Empty number of the first column means that the respective statistic for a given sample size is not available.

Figure 5 provides an example of sheet Map. The programs use data only from cells marked with yellow. All the rest is optional comments. You may insert your own comments and store other information in this sheet.

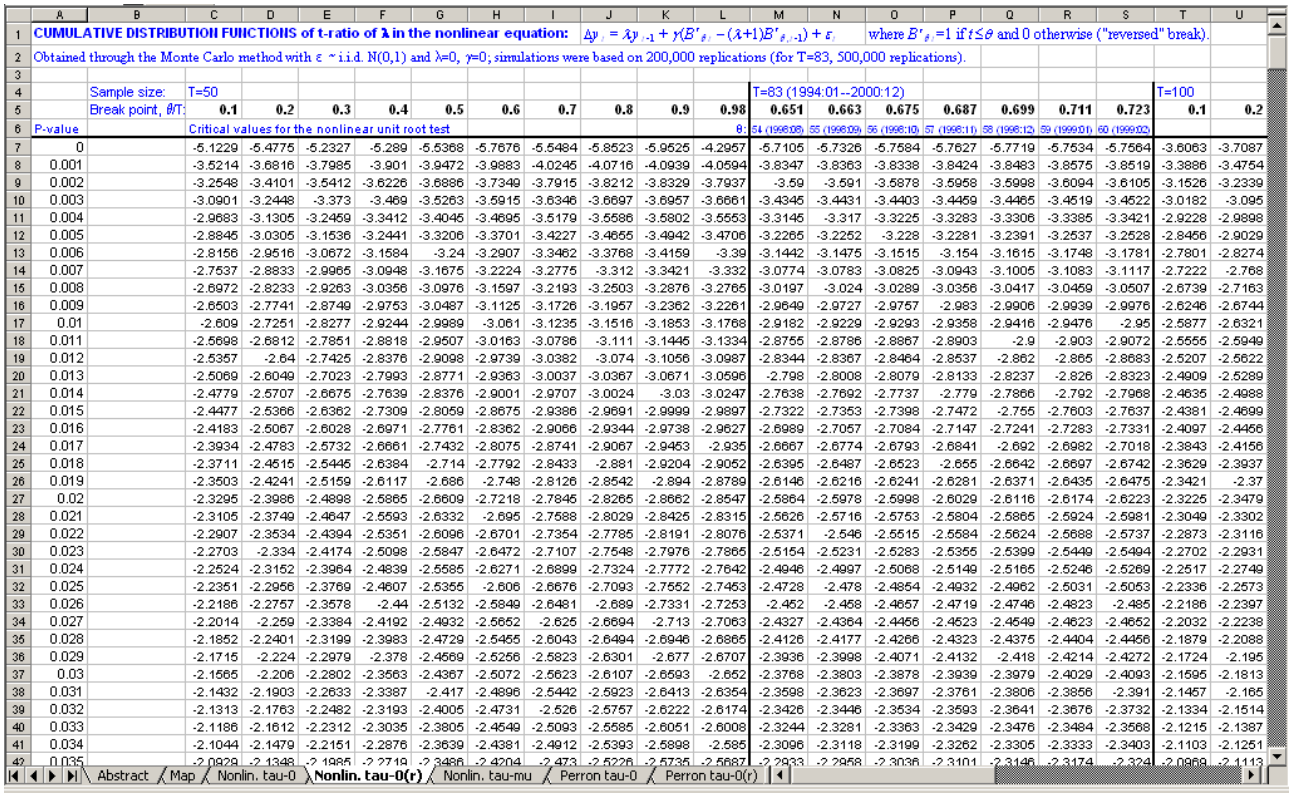
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	THE MAP OF THE FILE														
2	Column of p-values:	1													
3	Row of break points:	5													
4	Total number of samle sizes:	7													
5	First row of the data:	7													
6	Last row of the data	1007													
7	First row of the allocation table	13													
8															
9	**Data allocation table** (columns of critical values for a given sample size)														
10		Nonlinear test						Perron test						Dickey-Fuller test	
11	Sample size	no const		no const, reversed break		const		no const		no const, reversed break		const		no const	const
12		first	last	first	last	first	last	first	last	first	last	first	last		
13	**26**					3	11					3	11	3	3
14	**50**	3	12	3	12	12	20	3	12	3	12	12	20	4	4
15	**83**			13	19					13	19			5	5
16	**100**	13	27	20	28	21	29	13	27	20	28	21	29	6	6
17	**150**	28	36	29	37	30	38	28	36	29	37	30	38	7	7
18	**200**	37	45	38	46	39	47	37	45	38	46	39	47	8	8
19	**10000**	46	54			48	56	46	54			48	56	9	9
20															
21	Sheet name	Nonlin. tau-0		Nonlin. tau-0(r)		Nonlin. tau-mu		Perron tau-0		Perron tau-0(r)		Perron tau-mu		DF tau-0	DF tau-mu
22															

Figure 5. Description of the file organization.

As seen from the figure, the file does not contain τ_{0NL}, τ'_{0NL}, τ_{0P}, and τ'_{0P} for sample of size 26; there is no τ_{0NL}, $\tau_{\mu NL}$, τ_{0P}, and $\tau_{\mu P}$ for sample of size 83; and τ'_{0NL} and τ'_{0P} are not available for $T =$ 10000. For a given sample size, the allocation of data sometimes varies across sheets (because of different sets of break points).

As follows from the preceding, each particular statistic should be a continuous block for each sample size in the statistic sheet. These blocks may follow in any order by sample size (i.e., you need not sort them by sample size). Blocks may be separated with an arbitrary number of columns, empty or filled with some your additional data. But there should not be any additional columns or/and rows within blocks. It is desirable that the break points for each sample size be arranged in ascending order. (Generally, it does not matter; however, it is possible that some particular cases exist that would cause errors, be the break points not sorted.) Rows of the statistics should follow in ascending order of *p*-values. The data in the statistic sheets should be organized in accordance with the description provided in the sheet `Map`.

Figure 6 provides an example of a statistic sheet. (It corresponds the description displayed in Figure 5.) The data used by the programs are in black print. Blue font marks optional comments. You may insert your own comments and store other information in this sheet outside the statistic blocks. Graphs are also allowable.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	**CUMULATIVE DISTRIBUTION FUNCTIONS of t-ratio of λ in the nonlinear equation:**									$\Delta y_t = \lambda y_{t-1} + \gamma(B'_{\theta,t} - (\lambda+1)B'_{\theta,t-1}) + \varepsilon_t$			where $B'_{\theta,t}$=1 if $t \le \theta$ and 0 otherwise ("reversed" break).								
2	Obtained through the Monte Carlo method with ε ~ i.i.d. N(0,1) and λ=0, γ=0; simulations were based on 200,000 replications (for T=83, 500,000 replications).																				
3																					
4		Sample size:	T=50										T=83 (1994:01--2000:12)							T=100	
5		Break point, θ/T:	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**0.98**	**0.651**	**0.663**	**0.675**	**0.687**	**0.699**	**0.711**	**0.723**	**0.1**	**0.2**
6	P-value		Critical values for the nonlinear unit root test									θ:	54 (1998:06)	55 (1998:09)	56 (1998:10)	57 (1998:11)	58 (1998:12)	59 (1999:01)	60 (1999:02)		
7	0		-5.1229	-5.4775	-5.2327	-5.289	-5.5368	-5.7676	-5.5484	-5.8523	-5.9525	-4.2957	-5.7105	-5.7326	-5.7584	-5.7627	-5.7719	-5.7534	-5.7564	-3.6063	-3.7087
8	0.001		-3.5214	-3.6816	-3.7985	-3.901	-3.9472	-3.9883	-4.0245	-4.0716	-4.0939	-4.0594	-3.8347	-3.8363	-3.8338	-3.8424	-3.8483	-3.8575	-3.8519	-3.3886	-3.4754
9	0.002		-3.2548	-3.4101	-3.5412	-3.6226	-3.6886	-3.7349	-3.7915	-3.8212	-3.8329	-3.7937	-3.59	-3.591	-3.5878	-3.5958	-3.5998	-3.6094	-3.6105	-3.1526	-3.2339
10	0.003		-3.0901	-3.2448	-3.373	-3.469	-3.5263	-3.5915	-3.6346	-3.6697	-3.6957	-3.6661	-3.4345	-3.4431	-3.4403	-3.4459	-3.4465	-3.4519	-3.4522	-3.0182	-3.095
11	0.004		-2.9683	-3.1305	-3.2459	-3.3412	-3.4045	-3.4695	-3.5179	-3.5586	-3.5802	-3.5553	-3.3145	-3.317	-3.3225	-3.3283	-3.3306	-3.3385	-3.3421	-2.9228	-2.9898
12	0.005		-2.8845	-3.0305	-3.1536	-3.2441	-3.3206	-3.3701	-3.4227	-3.4655	-3.4942	-3.4706	-3.2265	-3.2252	-3.228	-3.2281	-3.2391	-3.2537	-3.2528	-2.8456	-2.9029
13	0.006		-2.8156	-2.9516	-3.0672	-3.1584	-3.24	-3.2907	-3.3462	-3.3768	-3.4159	-3.39	-3.1442	-3.1475	-3.1515	-3.154	-3.1615	-3.1748	-3.1781	-2.7801	-2.8274
14	0.007		-2.7537	-2.8833	-2.9965	-3.0948	-3.1675	-3.2224	-3.2775	-3.312	-3.3421	-3.332	-3.0774	-3.0783	-3.0825	-3.0943	-3.1005	-3.1083	-3.1117	-2.7222	-2.768
15	0.008		-2.6972	-2.8233	-2.9263	-3.0356	-3.0976	-3.1597	-3.2193	-3.2503	-3.2876	-3.2765	-3.0197	-3.024	-3.0289	-3.0356	-3.0417	-3.0459	-3.0507	-2.6739	-2.7163
16	0.009		-2.6503	-2.7741	-2.8749	-2.9753	-3.0487	-3.1125	-3.1726	-3.1957	-3.2362	-3.2261	-2.9649	-2.9727	-2.9757	-2.983	-2.9906	-2.9939	-2.9976	-2.6246	-2.6744
17	0.01		-2.609	-2.7251	-2.8277	-2.9244	-2.9989	-3.061	-3.1235	-3.1516	-3.1853	-3.1768	-2.9182	-2.9229	-2.9293	-2.9358	-2.9416	-2.9476	-2.95	-2.5877	-2.6321
18	0.011		-2.5698	-2.6812	-2.7851	-2.8818	-2.9507	-3.0163	-3.0786	-3.111	-3.1445	-3.1334	-2.8755	-2.8786	-2.8867	-2.8903	-2.9	-2.903	-2.9072	-2.5555	-2.5949
19	0.012		-2.5357	-2.64	-2.7425	-2.8376	-2.9098	-2.9739	-3.0382	-3.074	-3.1056	-3.0987	-2.8344	-2.8367	-2.8464	-2.8537	-2.862	-2.865	-2.8683	-2.5207	-2.5622
20	0.013		-2.5069	-2.6049	-2.7023	-2.7993	-2.8771	-2.9363	-3.0037	-3.0367	-3.0671	-3.0596	-2.798	-2.8008	-2.8079	-2.8133	-2.8237	-2.826	-2.8323	-2.4909	-2.5289
21	0.014		-2.4779	-2.5707	-2.6675	-2.7639	-2.8376	-2.9001	-2.9707	-3.0024	-3.03	-3.0247	-2.7638	-2.7692	-2.7737	-2.779	-2.7866	-2.792	-2.7968	-2.4635	-2.4988
22	0.015		-2.4477	-2.5366	-2.6362	-2.7309	-2.8059	-2.8675	-2.9386	-2.9691	-2.9999	-2.9897	-2.7322	-2.7353	-2.7398	-2.7472	-2.755	-2.7603	-2.7637	-2.4381	-2.4699
23	0.016		-2.4183	-2.5067	-2.6028	-2.6971	-2.7761	-2.8362	-2.9066	-2.9344	-2.9738	-2.9627	-2.6989	-2.7057	-2.7084	-2.7147	-2.7241	-2.7283	-2.7331	-2.4097	-2.4456
24	0.017		-2.3934	-2.4783	-2.5732	-2.6661	-2.7432	-2.8075	-2.8741	-2.9067	-2.9453	-2.935	-2.6667	-2.6774	-2.6793	-2.6841	-2.692	-2.6982	-2.7018	-2.3843	-2.4156
25	0.018		-2.3711	-2.4515	-2.5445	-2.6384	-2.714	-2.7792	-2.8433	-2.881	-2.9204	-2.9052	-2.6395	-2.6487	-2.6523	-2.655	-2.6642	-2.6697	-2.6742	-2.3629	-2.3937
26	0.019		-2.3503	-2.4241	-2.5159	-2.6117	-2.686	-2.748	-2.8126	-2.8542	-2.894	-2.8789	-2.6146	-2.6216	-2.6241	-2.6281	-2.6371	-2.6435	-2.6475	-2.3421	-2.37
27	0.02		-2.3295	-2.3986	-2.4898	-2.5865	-2.6609	-2.7218	-2.7845	-2.8265	-2.8662	-2.8547	-2.5864	-2.5978	-2.5998	-2.6029	-2.6116	-2.6174	-2.6223	-2.3225	-2.3479
28	0.021		-2.3105	-2.3749	-2.4647	-2.5593	-2.6332	-2.695	-2.7588	-2.8029	-2.8425	-2.8315	-2.5626	-2.5716	-2.5753	-2.5804	-2.5865	-2.5924	-2.5981	-2.3049	-2.3302
29	0.022		-2.2907	-2.3534	-2.4394	-2.5351	-2.6096	-2.6701	-2.7354	-2.7785	-2.8191	-2.8076	-2.5371	-2.546	-2.5515	-2.5584	-2.5624	-2.5688	-2.5737	-2.2873	-2.3116
30	0.023		-2.2703	-2.334	-2.4174	-2.5098	-2.5847	-2.6472	-2.7107	-2.7548	-2.7976	-2.7865	-2.5154	-2.5231	-2.5283	-2.5355	-2.5399	-2.5449	-2.5494	-2.2702	-2.2931
31	0.024		-2.2524	-2.3152	-2.3964	-2.4839	-2.5585	-2.6271	-2.6899	-2.7324	-2.7772	-2.7642	-2.4946	-2.4997	-2.5068	-2.5149	-2.5165	-2.5246	-2.5269	-2.2517	-2.2749
32	0.025		-2.2351	-2.2956	-2.3769	-2.4607	-2.5355	-2.606	-2.6676	-2.7093	-2.7552	-2.7453	-2.4728	-2.478	-2.4854	-2.4932	-2.4962	-2.5031	-2.5053	-2.2336	-2.2573
33	0.026		-2.2186	-2.2757	-2.3578	-2.44	-2.5132	-2.5849	-2.6481	-2.689	-2.7331	-2.7253	-2.452	-2.458	-2.4657	-2.4719	-2.4746	-2.4823	-2.485	-2.2186	-2.2397
34	0.027		-2.2014	-2.259	-2.3384	-2.4192	-2.4932	-2.5652	-2.625	-2.6694	-2.713	-2.7063	-2.4327	-2.4364	-2.4456	-2.4523	-2.4549	-2.4623	-2.4652	-2.2032	-2.2236
35	0.028		-2.1852	-2.2401	-2.3199	-2.3983	-2.4729	-2.5455	-2.6043	-2.6494	-2.6946	-2.6865	-2.4126	-2.4177	-2.4266	-2.4323	-2.4375	-2.4404	-2.4456	-2.1879	-2.2088
36	0.029		-2.1715	-2.224	-2.2979	-2.378	-2.4569	-2.5256	-2.5823	-2.6301	-2.677	-2.6707	-2.3936	-2.3998	-2.4071	-2.4132	-2.418	-2.4214	-2.4272	-2.1724	-2.195
37	0.03		-2.1565	-2.206	-2.2802	-2.3563	-2.4367	-2.5072	-2.5623	-2.6107	-2.6593	-2.652	-2.3768	-2.3803	-2.3878	-2.3939	-2.3979	-2.4029	-2.4093	-2.1595	-2.1813
38	0.031		-2.1432	-2.1903	-2.2633	-2.3387	-2.417	-2.4896	-2.5442	-2.5923	-2.6413	-2.6354	-2.3598	-2.3623	-2.3697	-2.3761	-2.3806	-2.3856	-2.391	-2.1457	-2.165
39	0.032		-2.1313	-2.1763	-2.2482	-2.3193	-2.4005	-2.4731	-2.526	-2.5757	-2.6222	-2.6174	-2.3426	-2.3446	-2.3534	-2.3593	-2.3641	-2.3676	-2.3732	-2.1334	-2.1514
40	0.033		-2.1186	-2.1612	-2.2312	-2.3035	-2.3805	-2.4549	-2.5093	-2.5585	-2.6051	-2.6008	-2.3244	-2.3281	-2.3363	-2.3429	-2.3476	-2.3484	-2.3568	-2.1215	-2.1387
41	0.034		-2.1044	-2.1479	-2.2151	-2.2876	-2.3639	-2.4381	-2.4912	-2.5393	-2.5898	-2.585	-2.3096	-2.3118	-2.3199	-2.3262	-2.3305	-2.3333	-2.3403	-2.1103	-2.1251
42	0.035		-2.0929	-2.1348	-2.1965	-2.2719	-2.3486	-2.4204	-2.473	-2.5226	-2.5735	-2.5687	-2.2933	-2.2958	-2.3036	-2.3101	-2.3146	-2.3174	-2.324	-2.0969	-2.1113

Abstract | Map | Nonlin. tau-0 | **Nonlin. tau-0(r)** | Nonlin. tau-mu | Perron tau-0 | Perron tau-0(r)

Figure 6. Example of a statistic sheet in the file of critical values.

The programs check the data description. However, they do not check whether the data is actually organized in accordance with it. A discrepancy can cause some (*a priory* unknown) error while executing a program. In such a case, if you cannot find a mistake in your file of critical values, please do not hesitate to contact me, providing the file and the program arguments used when the error has occurred.

References

Enders, W. (1995). *Applied Econometric Time Series.* New York – Chichester – Brisbane – Toronto – Singapore: John Wiley & Sons.

Gluschenko, K. (2005). Nonlinearly Testing for a Unit Root in the Presence of a Break in the Mean. Institute of Economics and Industrial Engineering. (Mimeo).
Available on http://econom.nsu.ru/users/gluschenko/research/papers/Gluschenko 2005 (mimeo).pdf.

Newey, W. K., and West, K. D. (1994). Automatic Lag Selection in Covariance Matrix Estimation. *Review of Economic Studies*, 61, 631-653.

Perron, P. (1990). Testing for a Unit Root in a Time Series with a Changing Mean. *Journal of Business and Economics Statistics*, 8, 153-162.

Phillips, P.C.B. (1987). Time Series Regression with a Unit Root. *Econometrica*, 55, 277-301.

Quantitative Micro Software, LLC (2001). *EViews 4 User's Guide*.

Quantitative Micro Software, LLC (2004). *EViews 5 User's Guide*.

Appendix: Additional Programs

A.1 Experimenting with parameters

Function. This program estimates cumulative distribution functions (CDF) of the nonlinear and Perron test statistics under the null hypothesis for a specified set of break points, as well as those of the Dickey-Fuller statistics τ_0 and τ_{μ}, under the null hypothesis or under a specified alternative. In contrast to the program *CDFs of UR statistics with break*, this program allows to specify many parameters of the data generating process, so yielding various processes other than pure random walk.

Algorithm. In each replication, the program generates $\{\varepsilon_t\}_{t=1,\dots,T} \sim$ iid N$(0,\sigma^2)$ and construct a time series $y_t = \alpha + (\lambda+1)y_{t-1} + \gamma B_{\theta t} - \gamma(\lambda+1)B_{\theta,t-1} + \varepsilon_t$ with $y_0 \sim$ N(c, ζ^2). Parameters α, λ, γ, θ, σ, c, and ζ^2 are user-specified (a set of values may be specified, but only for one of them). It may be specified $\zeta^2 = 0$; this means y_0 to be fixed rather than random. Then the program estimates (4a) and (4b), and, depending of a specified type of equations, either (1a) and (2a) or (1b) and (2b) for each value of a parameter specified with a set of its values. Having reached the specified number of replications, the program computes the CDFs, using 1000 quantiles plus one more for probability 0. (The program does not check accordance between the numbers of quantiles and replications! Thus, having specified a small number of replications, you can obtain unsatisfactory results.)

Results. The program generates and saves to disc an EViews workfile with a user-specified name. The content of this file is the same as that of the file created by the program *CDFs of UR statistics with break*; see Subsection 2.2 of this document. The difference is in the structure of the table `Description`. Figure A1 provides an example of this table.

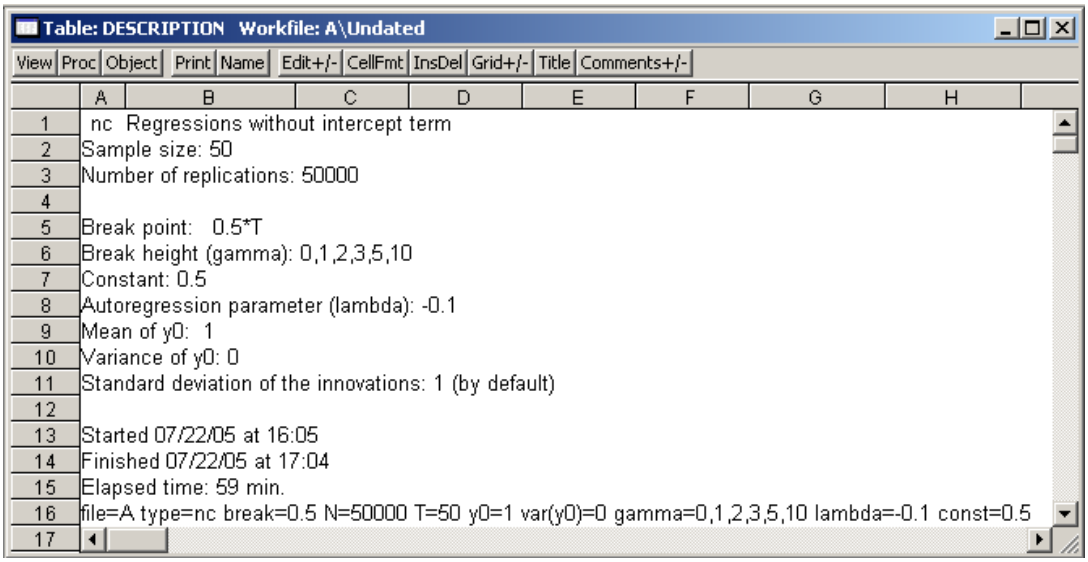


Figure A1. `Description` reported by the program *Experimenting with parameters.*

One more difference is in that M is the number of values in the value set specified for some parameter (in Figure A1, such a parameter is γ, the number of its values is 6). Matrices `cdf_df_nc` and `cdf_df_c` are now $1001 \times (M+1)$; matrices `t_df_nc` and `t_df_c` are $N \times M$. The number of a column in `t_nl`, `t_p`, `t_df_nc`, and `t_df_c` corresponds to the number of value in the set of values of a parameter (say, of γ, like in Figure A1); for `cdf_t_nl`, cdf_t_p, `cdf_df_nc`, and

`cdf_df_c`, one should be added to the latter number, as the first column of these matrices is *p*-values.

Arguments. Arguments of the program may follow in any order. Only 3 of all the arguments are optional, having default values, namely `y0`, `var(y0)`, and `sigma`. The rest arguments must be specified except the case when the workfile already exists on disc (e.g., estimations are processing in parts). Only the filename should be provided in such a case (then keyword **`file=`** may be omitted).

The program uses the following 11 arguments:

`file=`[<*path*>]<*file name*> is the workfile name (if there are blanks in the path or/and file name, enclose the full name in quotes, e.g., `file="c:\my path\my file.wf1"`).

`T=`<*number*> is the sample size ($t = 0,...,T$). (In fact, in the workfile and program, it will be $t = 1,...,T+1$.)

`N=`<*number*> is the number of replications.

`type=``nc|c` is the type of equations to be estimated: `nc` means equations with no constant, (1a) and (2a); `c` means equations with constant, (1b) and (2b).

`break=`[−]<*number*$_1$>[`,`[−]<*number*$_2$>][`,`[−]<*number*$_3$>[`,and so on`]]] is the relative break point(s), Θ; minus means a break to be reversed.

[**`y0=`**<*number*$_1$>[`,`<*number*$_2$>][`,`<*number*$_3$>[`,and so on`]]]] is the mean(s) of the initial value, c. This argument is optional; by default, `y0=0`.

[**`var(y0)=`**<*number*$_1$>[`,`<*number*$_2$>][`,`<*number*$_3$>[`,and so on`]]]] is the variance(s) of the y_0, ζ^2. Zero value of it implies that y_0 is a fixed constant (specified by the above argument) rather than random. This argument is optional; by default, `var(y0)=0`.

`gamma=`<*number*$_1$>[`,`<*number*$_2$>][`,`<*number*$_3$>[`,and so on`]]]] is the value(s) of the break height, γ.

`const=`<*number*$_1$>[`,`<*number*$_2$>][`,`<*number*$_3$>[`,and so on`]]]] is the value(s) of the constant, α.

`lambda=`<*number*$_1$>[`,`<*number*$_2$>][`,`<*number*$_3$>[`,and so on`]]]] is the value(s) of λ.

[**`sigma=`**<*number*$_1$>[`,`<*number*$_2$>][`,`<*number*$_3$>[`,and so on`]]]] is the standard deviation(s) of the innovations, σ. This argument is optional; by default, `sigma=1`, i.e., $\varepsilon_t \sim$ iid N(0,1).

Examples. Arguments

```
file="50nc,across y0" type=nc T=50 N=200000 break=0.5 y0=0,1,3,5,10 var(y0)=0
    const=0 gamma=0 lambda=0 sigma=1
```

imply the data generating process to be $y_t = y_{t-1} + \varepsilon_t$ with $t = 0,\ldots,50$, $\{\varepsilon_t\} \sim$ iid N(0,1), and $y_0 = 0$, 1, 3, 5, and 10 (as it is specified $\alpha = 0$, $\lambda = 0$, $\gamma = 0$, $\zeta^2 = 0$, and $\sigma = 1$, and a set of values is provided for y_0); the program will estimate CDFs of the *t*-ratio of λ in no-constant type equations (1a) and (2a) with the break dummy switching from 0 to 1 at $t = 50 \cdot 0.5 + 1 = 26$, using 200,000 replications. The results will be saved to file named *50nc,across y0.wf1* .

With arguments

```
file=test type=c T=150 N=20000 break=0.5 lambda=-0.1 gamma=2 const=0.5 sigma=0.5
    y0=1 var(y0)=0,1,4,10,25,100 ,
```

the data generating process will be $y_t = 0.5 + 0.9y_{t-1} + 2(B_{\theta t} - 0.9B_{\theta,t-1}) + \varepsilon_t$ with $t = 0,\ldots,150$ and $\{\varepsilon_t\} \sim$ iid N(0,0.5^2). The CDFs will be estimated for equations (1b) and (2b) with $B_{\theta t}$ switching from 0 to 1 at $t = 150 \cdot 0.5 + 1 = 76$, using 20,000 replications. For each statistic, there will be 6 CDFs corresponding to $y_0 = 1$ (i.e., $y_0 \sim$ N(1, 0)), $y_0 \sim$ N(1, 1), $y_0 \sim$ N(1, 4),…, $y_0 \sim$ N(1, 100). The results will be saved to file named *test.wf1* .

A.2 Power

Function. This program estimates power of the nonlinear and Perron test as well as that of the Dickey-Fuller test (for a given test size) under a user-specified alternative for different values of λ. The program uses file *CriticalValues.xls* that contains critical values of the above tests. (These values are obtained with the use of the program *CDFs of UR statistics with break*.) See Section 3 of this document for the description of this file. (It is possible to use a different file with critical values; it should have a certain structure described in Section 3 of this document.)

Algorithm. At first, the program finds critical values of the tests for a given sample size, test size, and break point in the file of critical values. The absence of some (or even all) data does not prevent program from execution; in such a case, the respective powers just will not be estimated. Then the powers are estimated. In each replication, the program generates $\{\varepsilon_t\}_{t=1,\dots,T} \sim$ iid N$(0,\sigma^2)$ and construct a time series $y_t = \alpha + (\lambda+1)y_{t-1} + \gamma B_{\theta t} - \gamma(\lambda+1)B_{\theta,t-1} + \varepsilon_t$ with $y_0 \sim$ N(c, ζ^2). Parameters α, λ, γ, θ, σ, c, and ζ^2 are user-specified; a set of values may be specified for λ. It may be specified $\zeta^2 = 0$; this means y_0 to be fixed rather than random. Then the program estimates regressions (4a), (4b), (1a), (1b), (2a), and (2b) for each specified value of a λ. Estimated t-ratios of λ are compared with critical values, summing the cases of rejection the unit root hypothesis. Having reached the specified number of replications, the program computes the probabilities of rejection, dividing frequencies to the number of replications.

Results. The program generates and saves to disc an EViews workfile with a user-specified name. This file contains the table `Results` which reports both conditions of the estimation and the results obtained (as well as non-critical errors in the critical values file). Figure A2 provides an example. The estimated powers are also saved in matrix `power`.

Table: RESULTS Workfile: POWER(200,C=0,G=0,B=05)\Undated

View | Proc | Object | Print | Name | Edit+/- | CellFmt | InsDel | Grid+/- | Title | Comments+/-

	A	B	C	D	E	F	G	H	I
1	POWER EXPERIMENT								
2	Number of replications: 200000								
3									
4	Sample size, T: 200								
5	Break point: 0.5*T								
6	Break height, gamma: 0								
7	Constant: 0								
8	Initial condition: y(0) = 0								
9	Standard deviation of the innovations, sigma: 1 (by default)								
10									
11	SIZE of the tests: 0.05	Critical values from: CriticalValues.xls (by default)							
12	Test:	DF tau-0	DF tau-mu	Nonlin. tau-0	Nonlin. tau-mu	Perron tau-0	Perron tau-mu		
13									
14	5% c.v.:	-1.939	-2.882	-1.984	-2.911	-2.641	-3.367		
15									
16	lambda	ESTIMATED POWER:							
17	-0.2	1.000	1.000	1.000	1.000	1.000	0.999		
18	-0.1	0.998	0.856	0.993	0.840	0.934	0.661		
19	-0.05	0.761	0.309	0.728	0.319	0.435	0.212		
20	-0.01	0.119	0.067	0.119	0.071	0.087	0.068		
21	0	0.050	0.050	0.050	0.051	0.051	0.050		
22	Started 07/21/05 at 10:30								
23	Finished 07/21/05 at 21:23								
24	Elapsed time: 10:53 hours								
25	file="Power(200,c=0,g=0,b=05)" size=0.05 lambda=-0.2,-0.1,-0.05,-0.01,0 break=0.5 N=200000 T=200 y0=0 gamma=0 const=0								
26									

Figure A2. `Results` reported by the program *Power*.

In the table `Results`, Row 12 contains the names of the tests. "DF tau-0" stands for the Dickey-Fuller τ_0-test associated with model (4a); "DF tau-mu" stands for the Dickey-Fuller τ_μ-test associated with model (4b); "Nonlin. tau-0" stands for the nonlinear τ_{0NL}-test associated with model (1a); "Nonlin. tau-mu" stands for the nonlinear $\tau_{\mu NL}$-test associated with model (1b); "Perron tau-0" stands for the Perron τ_{0P}-test associated with model (2a); "Perron tau-mu" stands for the Perron $\tau_{\mu P}$-test associated with model (2b). Row 14 reports critical values of the respective tests for a given test size. Lines 17 through 21 (the number of the last line may differ in other estimations) tabulate the results of the power estimation.

Arguments. Arguments of the program may follow in any order. Only 4 of all the arguments are optional, having default values, namely `y0`, `gamma`, `sigma`, and `CVfile`. The rest arguments must be specified except the case when the workfile already exists on disc (e.g., estimations are processing in parts). Only the filename should be provided in such a case (then keyword **`file=`** may be omitted).

The program uses the following 11 arguments:

`file=`[<*path*>]<*file name*> is the workfile name (if there are blanks in the path or/and file name, enclose the full name in quotes, e.g., `file="c:\my path\my file.wf1"`).

`T=`<*number*> is the sample size ($t = 0,...,T$). (In fact, in the workfile and program, it will be $t = 1,...,T+1$.)

`N=`<*number*> is the number of replications.

`size=`<*number*> is the size of the tests.

`lambda=`<*number A*>**`,...,`**<*number B*>**`;`**<*number C*>
| <*number*$_1$>[**`,`**<*number*$_2$>][**`,`**<*number*$_3$>[**`,`**`and so on`]]]
is the list of the values of λ. The upper form specifies them as a sequence from <*number A*> through <*number B*> with increment <*number C*>; the lower one lists any number of specific values.

`const=`<*number*> is the value of the constant, α.

`break=`[−]<*number*> is the relative break point(s), Θ; minus means a break to be reversed.

[**`gamma=`**<*number*>] is the value of the break height, γ. This argument is optional; by default, `gamma=0`.

[**`y0=`**<*number*> | N(<*number*$_1$>**`,`**<*number*$_2$>) | N(<*number*$_1$>**`,`**T)] is the initial value, y_0. The first option specifies it as a constant equaling <*number*>. The other two specify y_0 as a normal random variable with the mean <*number*$_1$> and the variance <*number*$_2$> or equaling the sample size, T. This argument is optional; by default, `y0=0`.

[**`sigma=`**<*number*>] is the standard deviation of the innovations, σ. This argument is optional; by default, `sigma=1`, i.e., $\varepsilon_t \sim$ iid N(0,1).

[**`CVfile=`**[<*path*>]<*file name*>] is the name of an (Excel) file with critical values of the tests (if there are blanks in the path or/and file name, enclose the full name in quotes). This argument is optional; by default, `CVfile=CriticalValues.xls`. (It should be placed to the default – for your EViews – directory; otherwise specify it.)

Example. Arguments

```
file=c:\UR\Power100.wf1 size=0.05 N=200000 lambda=-0.2,-0.1,-0.05,-0.01,0 T=100
    break=0.5 y0=0 gamma=2 const=0.5 sigma=1
```

imply estimating power of the 5%-size tests under the alternative $y_t = 0.5 + \lambda y_{t-1} + 2(B_{\theta t} - (\lambda+1)B_{\theta,t-1}) + \varepsilon_t$ with $t = 0,\ldots,100$, break at $t = 51$, $y_0 = 0$, and $\{\varepsilon_t\} \sim$ iid N(0,1) across λ = -0.2, -0.1, -0.05, -0.01, and 0, using 200,000 replications. Critical values of the test will be drawn from *CriticalValues.xls*; results will be stored in file *Power100.wf1* written to folder *UR* of disc *C*.