RTADF: TESTING FOR BUBBLES WITH EVIEWS

By

Itamar Caspi

November 6, 2016

RESEARCH INSTITUTE FOR ECONOMETRICS

DISCUSSION PAPER NO. 4-16

DEPARTMENT OF ECONOMICS
BAR-ILAN UNIVERSITY
RAMAT-GAN 5290002, ISRAEL

http://econ.biu.ac.il/en/node/2473

RTADF: TESTING FOR BUBBLES WITH EVIEWS*

Itamar Caspi†

Bar-Ilan University and Ban of Israel

November 6, 2016

Abstract

This paper presents Rtadf (**R**ight **T**ail **A**ugmented **D**ickey-**F**uller), an EViews Add-in that facilitates the performance of time series based tests that help detect, date-stamp and monitor asset price bubbles. Detection strategy is based on a right-tail variation of the standard Augmented Dickey–Fuller(ADF) test where the alternative hypothesis is of a mildly explosive process. Rejection of the null in each of these tests may serve as empirical evidence for an asset price bubble. The add-in implements four types of tests: Standard ADF, Rolling window ADF, supremum ADF (SADF) (Phillips et al., 2011) and generalized SADF (GSADF) (Phillips et al., 2015). This add-in calculates the test statistics for each of the above four tests, simulates the corresponding exact finite sample critical values and p-values via Monte Carlo and bootstrap methods, and produces a graphical display of the date stamping procedure.

*Version 2.0. This manuscript is an updated version of "Caspi, I. (forthcoming). Rtadf: Testing for bubbles with EViews. *Journal of Statistical Software*.". I thank Peter C.B. Phillips, Shuping Shi, Jun Yu, Jonathan Benchimol and Yossi Yakhin for their helpful comments and suggestions. All remaining errors are mine.

†Research Department; Bank of Israel; P.O. Box 780, Jerusalem 91007, Israel; itamar.caspi@boi.org.il. The views expressed in this paper are solely those of the author and do not necessarily reflect the views of the Bank of Israel or any of its staff.

1. Introduction

Empirical identification of asset price bubbles in real time, and even in retrospect, is surely not an easy task, and it has been the source of academic and professional debate for several decades.[1] One strand of the empirical literature suggests using time series estimation techniques while exploiting predictions made by finance theory in order to test for the existence of bubbles in the data. The main idea, based on asset pricing theory, suggests that the existence of a bubble component in an observed asset price should be manifested in its dynamics and its stochastic properties. More specifically, theory predicts that if a bubble exists, prices should inherit its explosiveness property. This in turn enables formulating statistical tests aimed at detecting evidence of explosiveness in the data.[2]

One of the attempts to test for rational bubbles in the context of the stock market is found in Diba and Grossman (1988), where the authors suggest using reduced form stationarity tests with regard to stock prices and observable market fundamentals, and to rule out bubbles if the former is found no more explosive than the latter. Evans (1991), however, questions the power of such stationarity based tests in the presence of a periodically collapsing bubble (i.e., one that spontaneously occurs and bursts), an apparent feature of actual stock prices seen in the data. Using simulation methods, Evans (1991) shows that standard unit root and cointegration tests fail to reject the null of no bubble in the presence of periodically collapsing bubbles. Despite his findings, Evans (1991) leaves open the question of a better identification strategy.

More recently, new bubble detection strategies were developed and presented by Phillips et al. (2011, hereafter PWY) and Phillips et al. (2015, hereafter PSY). These strategies are based on recursive and rolling ADF unit root tests that enable us to detect bubbles in the data and to date-stamp their occurrence. These types of tests use a right tail variation of the Augmented Dickey-Fuller unit root test wherein the null hypothesis is of a unit root and the alternative is of a *mildly* explosive process.[3] PWY and PSY show that using recursive and rolling tests

[1] There is a large amount of academic debate with regard to the theoretical plausibility of bubbles (Brunnermeier, 2008; Iraola and Santos, 2008). This paper deals with bubbles of the *rational* type (a.k.a., 'rational bubbles'), i.e., bubbles consistent with the rational expectations hypothesis (Blanchard and Watson, 1982).

[2] For recent surveys on econometric tests for bubbles, see Gürkaynak (2008) and Homm and Breitung (2012).

[3] Phillips and Magdalinos (2007) define a mildly explosive root using the following data generating process

$$y_t = \delta_n y_{t-1} + \varepsilon_t,$$

results in higher power in the detection of bubbles, compared to standard tests on the whole sample. In a Monte Carlo study, Homm and Breitung (2012) compare several time series based tests for the detection of bubbles and find that the PWY strategy performs relatively well in detecting periodically collapsing bubbles and in real time monitoring. Phillips et al. (2015) show through a Monte Carlo study that the PSY strategy outperforms the PWY strategy in the presence of *multiple* bubbles.

This paper introduces Rtadf, an EViews Add-in that allows end users to easily test for the existence of bubbles, by readily applying four variations of the right-tail ADF unit root test, in line with the reduced form approach for bubble detection described above. Four tests include the standard ADF test and a rolling window ADF test, and the more recent PWY supremum ADF (SADF) test and the PSY generalized SADF (GSADF) test. The add-in performs two main tasks. First, it calculates the relevant test statistic, according to the selected test. Second, the add-in derives the corresponding exact finite sample critical values by performing a Monte Carlo simulation, under the assumption of Gaussian innovations, or by the bootstrap, which may be more robust in the presence of non-stationary volatility (Harvey et al., 2016) or facing small sample size (Gutierrez, 2011). The add-in allows the user to choose between performing a sequential and a parallel (multicore) simulation. It is shown that using the latter option results in significant reduction of computational time as the number of recursions needed to complete the simulation rises. (This is most relevant for the GSADF test.)

The rest of the paper is organized as follows. Section 2 presents a basic theoretical model of rational bubbles in a standard asset pricing model. Section 3 introduces the details of the econometric strategy used to detect explosive behavior in asset prices. Section 4 provides general instructions on how to use the Rtadf add-in within the EViews environment. Section 6 presents a hands-on illustration of Rtadf. Finally, Section 7 concludes.

where $\delta_n = 1 + \frac{c}{k_n}$, and where $(k_n)_{n\in\mathbb{N}}$ is a sequence increasing to ∞ such that $k_n = o(n)$ as $n \to \infty$. Limit theory for mildly explosive processes is developed in Phillips and Magdalinos (2007).

2. Asset pricing with rational bubbles

In this section a formal model of asset pricing with a rational bubble is presented. We start by specifying the definition of the single period return on an asset:

$$R_{t+1} \equiv \frac{P_{t+1} + D_{t+1}}{P_t} \tag{1}$$

where $R_{t+1} > 1$ is the gross return on holding the asset from period t to $t+1$, P_t is the price of the asset measured at the end of period t (i.e., the ex-dividend price) and D_{t+1} is the dividend the asset holder is entitled to for holding the stock from period t to $t+1$

Next, following Campbell and Shiller (1988) we write a log-linear approximation of Equation (1)

$$p_t = \kappa + \rho p_{t+1} + (1-\rho)\, d_{t+1} - r_{t+1} \tag{2}$$

where $p_t \equiv \log(P_t)$, $d_t \equiv \log(D_t)$, $r_t \equiv \log(R_t)$, $\rho = 1/\left[1 + e^{\left(\overline{p-d}\right)}\right]$ with $\overline{p-d}$ being the average log price-to-dividend ratio, and

$$\kappa = -\log(\rho) - (1-\rho)\log\left(\frac{1}{\rho} - 1\right).$$

Solving Equation (2) by forward iteration and taking expectations yields the following log-linear approximation of the log price-to-dividend ratio:

$$p_t - d_t = \frac{\kappa}{1-\rho} + \sum_{i=0}^{\infty} \rho^i E_t \left(\Delta d_{t+1+i} - r_{t+1+i}\right) + \lim_{i\to\infty} \rho^i E_t \left(p_{t+i} - d_{t+i}\right). \tag{3}$$

The right hand side of Equation (3) can be decomposed into two components,

$$p_t - d_t = f_t + b_t \tag{4}$$

where

$$f_t = \frac{\kappa}{1-\rho} + \sum_{i=0}^{\infty} \rho^i E_t \left(\Delta d_{t+1+i} - r_{t+1+i}\right), \tag{5}$$

is the fundamental component, stated in terms of the expected dividend growth

rate and expected returns, and where

$$b_t = \lim_{i \to \infty} \rho^i E_t \left(p_{t+i} - d_{t+i} \right), \tag{6}$$

is commonly referred to as the *rational* bubble component. The latter is the focus of the bubble tests described below.

Under the transversality condition, which implies no-Ponzi game, $\lim_{i \to \infty} \rho^i E_t p_{t+i} = 0$, and the possibility of a bubble is ruled out. Thus, the observed price equals the fundamental price. In contrast, the existence of a strictly positive bubble component, i.e., the situation where actual price exceeds what is implied by fundamentals, requires that investors expect to be compensated for overpayment (over the fundamental price) by the expected appreciation of the bubble component. In other words, investors are willing to pay a premium over the fundamental price only because they expect this premium to appreciate in the next period. Note that this behavior is completely consistent with the rational expectations assumption, hence the name 'rational bubble'.

More importantly, note that Equation (6) implies a submartingale property for b_t since

$$E_t(b_{t+1}) = \frac{1}{\rho} b_t = \left[1 + e^{(\overline{p-d})} \right] b_t, \tag{7}$$

where $\left[1 + e^{(\overline{p-d})} \right] > 0$. Thus, when $b_t \neq 0$, the log bubble component grows at rate g, where $g = e^{(\overline{p-d})} > 0$.

This model reveals important insights regarding the stochastic properties of $p_t - d_t$, according to which, we can formulate an econometric test designed to rule out the presence of a rational bubble component in an observed asset price. To see this, note that the stochastic properties of $p_t - d_t$, implied by (3), are determined by those of f_t and b_t. In turn, the dynamics of f_t are determined by expected Δd_t and r_t. If Δd_t and r_t are *at most* $I(1)$ processes, evidence of explosiveness in $p_t - d_t$ (in this model) can only be the result of the presence of a bubble, i.e., $b_t \neq 0$. Thus, a test for the presence of a bubble can be formulated as a test for explosive behavior in log price-to-dividend ratio, $p_t - d_t$.

3. Testing for bubbles

Following the conventions of PSY, assume the following random walk process with an asymptotically negligible drift:

$$y_t = dT^{-\eta} + \theta y_{t-1} + e_t, \quad e_t \overset{iid}{\sim} N(0, \sigma^2), \quad \theta = 1 \tag{8}$$

where d is a constant, η is a localizing coefficient that controls the magnitude of the drift as the sample size, T, approaches infinity and ε_t is the error term.[4]

Four test strategies implemented by the Rtadf add-in (which includes the ones suggested by PWY and PSY) are all based on some variation of the following reduced form empirical equation:

$$y_t = \mu + \delta y_{t-1} + \sum_{i=1}^{p} \phi_i \Delta y_{t-i} + \varepsilon_t. \tag{9}$$

where y_t is the variable in question (e.g., the price of a stock), μ is an intercept, p is the maximum number of lags, ϕ_i for $i = 1 \ldots p$ are the differenced lags coefficients and ε_t is the error term. Testing for a bubble (explosive behavior) is based on a right-tail variation of the standard ADF unit root test where the null hypothesis is of a unit root and the alternative is of a mildly explosive autoregressive coefficient. Formally, we test for

$$\begin{aligned} H_0: &\quad \delta = 1 \\ H_1: &\quad \delta > 1. \end{aligned}$$

Before proceeding to a description of the tests included in Rtadf, some notation is needed. For simplicity of exposition, we use a sample interval of $[0, 1]$ (i.e., we normalized the original sample by T). Denote by δ_{r_1,r_2} and by ADF_{r_1,r_2} the coefficient estimated by Equation (9) and its corresponding ADF statistic over the (normalized) sample $[r_1, r_2]$. In addition, denote by r_w the (fractional) window size of the regression, defined by $r_w = r_2 - r_1$ and by r_0 the fixed initial window, set by the user. The difference between the tests relates to the manner of setting r_1 and r_2.

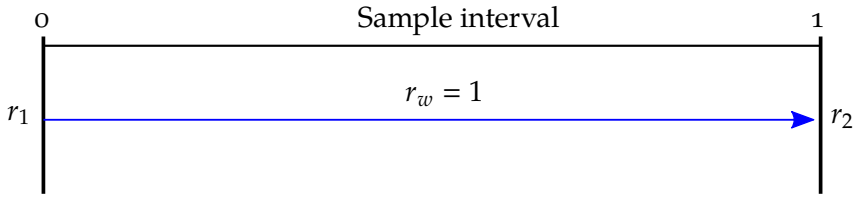
The first test included in Rtadf is a simple right-tailed version of the standard ADF unit root test. In this case, r_1 and r_2 are fixed to the first and last observations, respectively, of the whole sample, where in this case, $r_w = r_0 = 1$ (see Figure 1).[5]

[4] PSY set d, η and θ to unity, while PWY effectively set $\eta \to \infty$ (i.e., random walk without a drift).

[5] The t-statistic from this test matches the one reported by EViews.

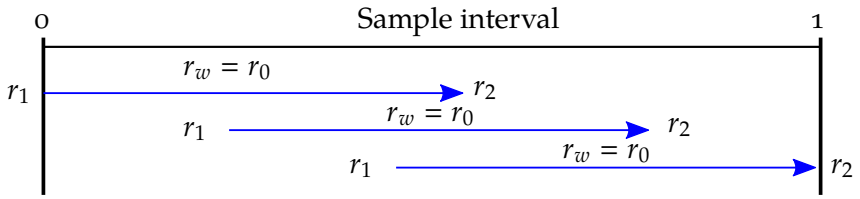
However, the critical values for testing the null hypothesis differ from the ones used in the usual ADF unit root test since we now need the right tail of the statistic's nonstandard distribution.

Figure 1. *Illustration of the ADF procedure.*



The second type of test, the rolling ADF (RADF) test, is a rolling version of the first test in which the ADF statistic is calculated over a rolling window of *fixed* size specified by the user, i.e., $r_w = r_0$ for all estimations. At each step of the RADF procedure, the window's start and end point (r_1 and r_2 respectively) are incremented one observation at a time (see Figure 2). Each estimation yields an ADF statistic, denoted as ADF_{r_1,r_2}. The RADF statistic is defined as the *supremum* ADF_{r_1,r_2} statistic among all possible windows.[6]

Figure 2. *Illustration of the RADF procedure.*



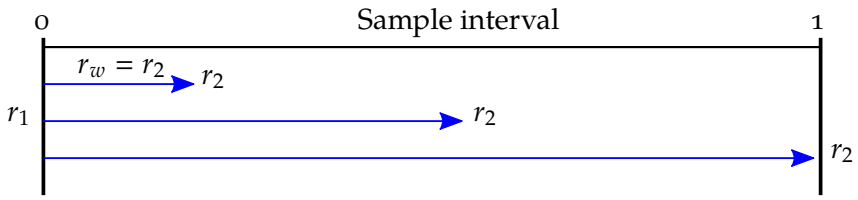
The SADF test, suggested by PWY, is based on recursive calculations of the ADF statistics with a fixed starting point and an expanding window, where the initial size of the window is set by the user. The estimation procedure is as follows (see Figure 3): The first observation in the sample is set as the starting point of the estimation window, r_1, i.e., $r_1 = 0$. Next, the end point of the initial estimation window, r_2, is set according to some choice of minimal window size, r_0, such that the initial window size is $r_w = r_2$ (again, in fraction terms). Finally, the regression is recursively estimated, while incrementing the window size, $r_2 \in [r_0, 1]$, one observation at a time. Each estimation yields an ADF statistic denoted as ADF_{r_2}. Note that in the last step, estimation will be based on the whole sample (i.e., $r_2 = 1$ and the statistic will be ADF_1). The SADF statistic is defined as the *supremum* value

[6]Note that the windows in the RADF procedure are overlapping.

of the ADF_{r_2} sequence for $r_2 \in [r_0, 1]$:

$$SADF(r_0) = \sup_{r_2 \in [r_0, 1]} \{ADF_{r_2}\} \tag{10}$$

Figure 3. *Illustration of the SADF procedure.*



The fourth and last test is the generalized SADF (GSADF), suggested by PSY. This test generalizes the SADF test by allowing more flexible estimation windows, wherein, unlike the SADF procedure, the starting point, r_1, is also allowed to vary within the range $[0, r_2 - r_0]$ (see Figure 4). Formally, the GSADF statistic is defined as

$$GSADF(r_0) = \sup_{\substack{r_2 \in [r_0, 1] \\ r_1 \in [0, r_2 - r_0]}} \{ADF_{r_1}^{r_2}\} \tag{11}$$

Figure 4. *Illustration of the GSADF procedure.*

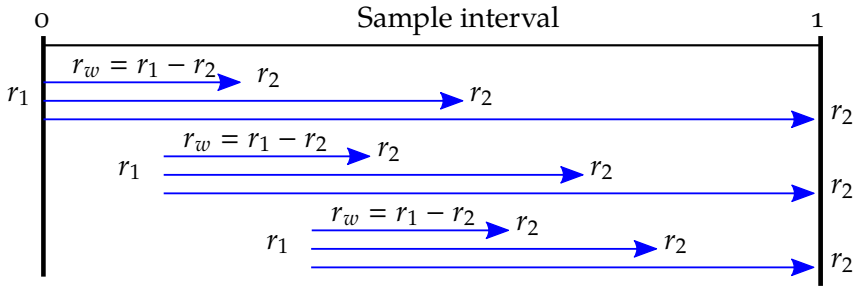


Table 1. *A summary of the tests' null and alternative hypotheses according to Phillips et al. (2015).*

Test	Null hypothesis	Alternative hypothesis
ADF	Unit root	Explosive process
SADF	Unit root	Single periodically collapsing bubble period
GSADF	Unit root	Multiple periodically collapsing bubbles

3.1. *Date-stamping bubble periods*

As PWY and PSY show, the SADF and GSADF procedures can also be used, under general regularity conditions, as a date-stamping strategy that consistently estimates the origination and termination of bubbles. In other words, if the null hypothesis of either of these tests is rejected, one can estimate the start and end points of a specific bubble (or bubbles). The date-stamping procedures will now be presented in brief.[7]

The first date-stamping strategy is based on the SADF test. PWY propose comparing each element of the estimated ADF_{r_2} sequence to the corresponding right-tailed critical values of the standard ADF statistic to identify a bubble initiating at time Tr_2. The estimated origination point of a bubble is the first chronological observation, denoted by T_{r_e}, in which ADF_{r_2} crosses the corresponding critical value from below, while the estimated termination point is the first chronological observation after T_{r_e}, denoted by T_{r_f}, in which ADF_{r_2} crosses the critical value from above. Formally, the estimates of the bubble period (as fractions of the sample) are defined by

$$\hat{r}_e = \inf_{r_2 \in [r_0, 1]} \left\{ r_2 : ADF_{r_2} > cv_{r_2}^{\beta_T} \right\} \tag{12}$$

$$\hat{r}_f = \inf_{r_2 \in [\hat{r}_e, 1]} \left\{ r_2 : ADF_{r_2} < cv_{r_2}^{\beta_T} \right\} \tag{13}$$

where $cv_{r_2}^{\beta_T}$ is the $100(1-\beta_T)\%$ critical value of the standard ADF statistic based on $[Tr_2]$ observations.[8,9]

Similarly, the estimates of the bubble period based on the GSADF are given by

$$\hat{r}_e = \inf_{r_2 \in [r_0, 1]} \left\{ r_2 : BSADF_{r_2}(r_0) > cv_{r_2}^{\beta_{Tr_2}} \right\} \tag{14}$$

$$\hat{r}_f = \inf_{r_2 \in [\hat{r}_e, 1]} \left\{ r_2 : BSADF_{r_2}(r_0) < cv_{r_2}^{\beta_{Tr_2}} \right\} \tag{15}$$

where $cv_{r_2}^{\beta_T}$ is the $100(1-\beta_T)\%$ critical value of the sup ADF statistic based on $[Tr_2]$ observations. $\text{BSADF}(r_0)$ for $r_2 \in [r_0, 1]$, is the backward sup ADF statistic

[7] For a detailed presentation see Phillips et al. (2011), Phillips and Yu (2011) and Phillips et al. (2015).

[8] In order to asymptotically eliminate type I errors, there is a need to let $\beta_T \to 0$ as $T \to 0$. However, in applied work it is convenient to use a constant β_T such as 5%.

[9] Phillips and Yu (2011) argue that the dating rule requires that the duration of the bubble be non-negligible. In Phillips et al. (2015) the authors define $\log(T)/T$ as a minimal lasting time for a bubble period.

that relates to the GSADF statistic by the following relation:

$$GSADF(r_0) = \sup_{r_2 \in [r_0, 1]} \{BSADF_{r_2}(r_0)\}. \tag{16}$$

4. Instructions and details

4.1. Installation

In essence, EViews Add-ins are EViews programs packed in a way that makes them feel and look like built-in EViews procedures.[10] This relatively new feature enables adding procedures and functionalities that have yet to be implemented in official releases of EViews. By using the Add-ins feature and program language, the user is able to augment standard written programs with an interactive user interface, thus making them more general purposed and user friendly. Moreover, unlike regular EViews programs, add-ins have the ability to run directly from EViews objects and/or by commands.

EViews add-ins are available for EViews users with versions 7.1 and above. Installing the Rtadf add-in (or any other add-in for that matter) on an existing copy of EViews can be done manually by downloading the self extracting installation file from the download section at the EViews website at `http://www.eviews.com/Addins/addins.shtml` where it is listed under Rtadf*.[11] Alternatively, EViews users with version 8 can download the add-in while inside EViews by clicking **Add-ins** → **Manage Add-ins**, selecting the Rtadf add-in from the list presented under the **Available** tab and clicking the **Install** button. In general, note that all other add-ins available on the list are written either by the EViews staff or by outside users.

Each add-in published on the EViews website (including Rtadf) has a corresponding support thread in the EViews Add-in Support forum, which can be found at `http://forums.eviews.com/viewforum.php?f=2`.

4.2. Using the add-in

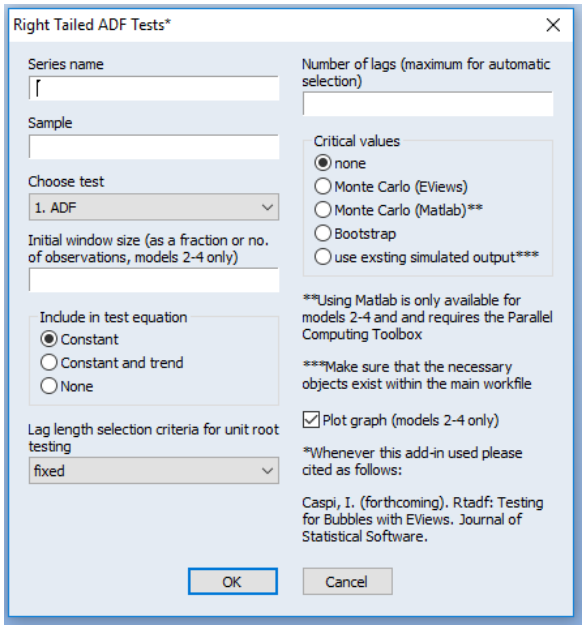
The Rtadf add-in can only be run from a series object. Initiating the add-in's dialog box is done by opening a series object and than clicking **Proc** → **Add-ins** → **Right**

[10] To R users, the concept is similar to R packages as with MATLAB users and tool-boxes.

[11] The asterisk next to the add-in's name indicates the add-in was developed by an EViews user rather than by IHS EViews.

Tail ADF tests. The test dialog box, presented in Figure 5, enables the user to set the sample period, type of test, initial window size (as a fraction or number of observations), deterministic terms in the test equation and the number of lags in the ADF equation (p in Equation 9), where it can be either fixed by the user, or automatically selected according to some information criterion. In addition, it allows the user to choose the option of simulating critical values for the test (thus prompting the simulation dialog box described below) and whether to view a graph that includes the sequence of ADF statistics, the corresponding critical value sequence and the actual series.

Figure 5. *Dialog box.*



As for the critical values, the add-in currently provides five options. Choosing **none** will skip the simulation part altogether and just show the estimated test statistic. Choosing either **Monte Carlo (with EViews)** or **Monte Carlo (with MATLAB** will prompt the simulation dialog box (see Figure 6). Note that simulation via MATLAB is only available for the RADF, SADF and GSADF and requires having the MATLAB as well as the Parallel Computing Toolbox. Alternativly, we can choose to base our inference on the bootstrap method by choosong **Bootstrap**. Finally, choosing the "use existing simulation output" will base the statistical inference and a previously run (Monte Carlo) simulation. This can save time in cases where existing critical values are appropriate for the new series at hand (in

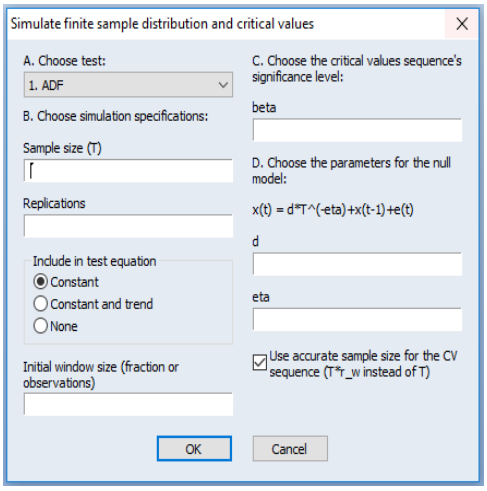
the sense of sample size, minimum window, etc.).[12]

5. Finite sample inference

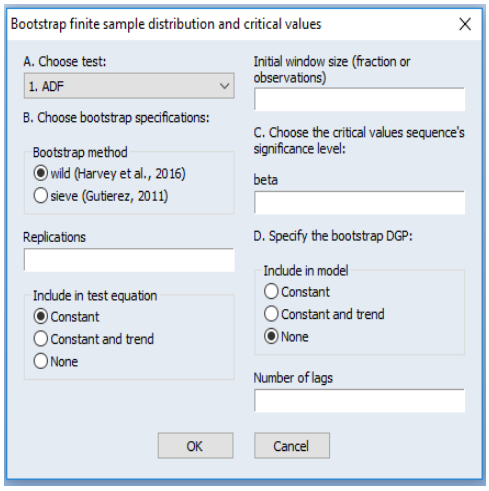
Whenever one of the 'simulate critical values' options are checked (Monte Carlo or Bootstrap), a new dialog box will prompted (see Figure 6), where all the necessary specifications for the simulation experiment can be edited. The output of the simulation experiment includes the 90, 95 and 99 percent quantiles of the approximate finite sample distribution of the statistic in question, the corresponding p-value of the test statistic and the critical values sequence for the date-stamping process.

Figure 6. *Critical values simulation dialog box.*

(a) *Monte Carlo.*



(b) *Bootstrap.*



The add-in provides two main methods for conducting finite sample inference. The first method, which is based on Monte Carlo simulation, is used in Phillips et al. (2011); Phillips and Yu (2011); Phillips et al. (2015). The second method, which is based on the bootstrap method, is used in Gutierrez (2011), Harvey et al. (2016) and Milunovich et al. (2016).

In what follows we provide details on how exactly the add-in generates the data needed for statistical inference, i.e., critical values and p-values.

[12]Before using the "use existing simulation output" option we need to make sure that the necessary objects exist in our main workfile. For example, is we run the GSADF test with "use existing simulation output" checked, the objects 'gsadf', 'gsadf_cv', 'gsadf_dist' and 'cv' must exist in the workfile.

5.1. *Monte Carlo based inference*

When the **Monte Carlo** simulation option is chosen, whether it is with EViews or MATLAB, critical values for all four tests are performed according to the following algorithm:

Step 1: Draw one realization at length T based on the null model (given by Equation (8)).

Step 2: Estimate Equation 9 by OLS (with or without recursion, depending on the test).

Step 3: Store the relevant test statistic (ADF/RADF/SADF/GSADF).

Repeat steps 1–3 N times (where N is a large number.)

Step 4: Calculate the 90%, 95% and 99% quantiles of the distribution of the relevant statistic.

As can be seen in Figure 6(a), simulation setup in Rtadf is very flexible, allowing the user to specify the type of test, appropriate sample size, number of replications, deterministic terms in the test equation, initial window size, significance level for the critical value sequence (i.e., β), specification for the parameters of the data generating process for the null hypothesis (i.e., set values for d, η and θ in Equation (8)). In addition the user can choose whether to use T or Tr_w in the null model when calculating the simulated critical value sequence (the latter is more accurate but can be very time consuming for large samples while the former is less accurate but runs faster).[13]

5.1.1. *Run-time comparisons*

Because of their recursive nature, applying the available tests available in the Rtadf add-in tend to be time consuming as the number of observations increases. Nonetheless, running time can be significantly decreased by using the multi-core capabilities of your machine.

The results of the running time simulations are presented in Table 2. All tests were run using EViews 9 and MATLAB 2014a (with the Parallel Computing Toolbox installed). The hardware used includes a core i7-4702MQ CPU with 16 GB RAM.

[13]Note that by default, sample size and the initial window size used in the previous step are shown in the simulation dialog box.

The table includes the running time (in seconds) for all four tests and for two sample sizes - 100 and 200 observations. As we can see, for 100 observations, in the RADF and the SADF tests, there is not much of a difference in running times between EViews and MATLAB. The reason for this similarity is that for a small number of replications, the communication time between EViews and MATLAB and the time it takes to set up a multi-core session, outweighs the benefits of the parallelized simulation. When the sample size is doubled to 200 observations, the efficiency of multi core calculations kicks in, and enables MATLAB to outperform EViews by roughly 50% faster than the single core simulation in EViews. As for the GSADF test, which involves many more estimations per replication, using the parallel computing option improves running time by more than 90% (!).

Table 2. *Comparison of tests' running time (in seconds).*

Test type	$T = 100$ r_0=0.19		$T = 200$ r_0=0.14	
	EViews	MATLAB	EViews	MATLAB
ADF	0.538	-	0.53	
RADF	16.75	17.52	34.60	17.52
SADF	16.56	15.19	34.71	17.90
GSADF	715.45	62.29	3197.87	232.75

5.2. Bootstrap based inference

When the **Bootstrap** option is chosen within the dialog box, the critical values are calculated according to the following algorithm. First, note that under the null, the model can be written as

$$\Delta y_t = \mu + \sum_{j=1}^{p} \phi_j \Delta y_{t-j} + \varepsilon_t \tag{17}$$

Step 1: Define the sample of innovations as $\varepsilon_t = \Delta y_t$ if the coefficients for the autoregressive lags and deterministic (constant and trend) are constrained to zero, and $\varepsilon_t = \Delta y_t - \hat{\mu} - \sum_{j=1}^{p} \hat{\phi}_j \Delta y_{t-j}$ otherwise, where μ and ,ϕ_j for $j = 1, \ldots, p$ are estimated by OLS.

Step 2: Generate T observations of bootstrapped innovations. For the wild bootstrap innovations are defined as $\varepsilon_t^* = w_t \varepsilon_t$, where $w_t \sim i.i.d.N(0,1)$, independent of ϵ_t. For the sieve method, the innovations are re-sampled (with replacement) from the centered residuals $\varepsilon_t - \bar{\varepsilon}_t$, where $\bar{\varepsilon}_t = (T - p - 1)^{-1} \sum_{t=2+p}^{T} \varepsilon_t$.

Step 3: If the coefficients for the autoregressive lags and deterministic (constant and trend) are constrained to zero, construct the bootstrapped sample y_t^* as $y_t* = \sum_{j=1}^{T} \varepsilon_t^*$, where $y_1^* = 0$. Otherwise, generate y_t^* using the OLS estimates of μ and ϕ_j for $j = 1, \ldots, p$,

$$y_t^* = \hat{\mu} + \sum_{j=1}^{k} \hat{\phi}_j \Delta y_{t-j}^* + \varepsilon_t^* \tag{18}$$

where $\Delta y_t^* = \Delta y_t$ for $t = 2, \ldots, p+1$ and $y_t^* = y_1$

Step 4: Compute the test statistic (ADF/RADF/SADF/GSADF) for y_t^* and repeat steps 1-4 B times in order to derive the bootstrap distribution of the test statistic.

Step 5: Calculate the 90%, 95% and 99% quantiles of the distribution of the relevant statistic.

As can be seen in Figure 6(b), the bootstrap setup in Rtadf is very flexible, allowing the user to specify the method of bootstrap used (wild or sieve), type of test, appropriate sample size, number of replications, deterministic terms in the test equation, initial window size, significance level for the critical value sequence (i.e., β), specification for the parameters of the data generating process for the bootstrap model (i.e., deterministic terms and lag length).[14]

5.3. Approximate p-values

Quantiles calculated in step 4 (Monte Carlo) and step 5 (bootstrap) are used for testing the null of unit root against the alternative of an explosive process. In addition, the simulation output includes the p-value of the test statistic, defined as the probability of observing a statistic as extreme as under the null, that is calculated as

$$p(\hat{\tau}) = \frac{1}{N} \sum_{j=1}^{N} \mathbb{I}(\tau_j > \hat{\tau}), \tag{19}$$

where $\hat{\tau}$ is the estimated test statistic (ADF/RADF/SADF/GSADF), N is the number of replications, $\mathbb{I}(\cdot)$ denotes the indicator function which is equal to 1 if the argument is true and 0 otherwise and τ_j are the simulated test statistics

[14]Note that by default, the initial window size used in the previous step are shown in the simulation dialog box.

($j = 1, \cdots, N$). The sequences of critical values, which are necessary for the date-stamping procedure, are also derived by the simulation.

5.4. Usage via a command line

The Rtadf add-in can also be called upon via a command line. This feature enables using the add-in's capabilities as an integrated part of other EViews programs. The syntax is given by:

series_name.**rtadf**(*options*)

Table 3. *Options for the command line.*

Basic options	
const (*default*)	Include a constant in the test equation.
trend	Include a constant and a linear time trend in the test equation.
none	Do not include a constant or time trend.
info = *arg* (*default* = 'fixed')	Either fixed or information criterion to use when computing automatic lag length selection: 'aic' (Akaike), 'sic' (Schwarz), 'hqc' (Hannan-Quinn), 'maic' (Modified Akaike), 'msic' (Modified Schwarz), 'mhqc' (Modified Hannan-Quinn).
lag = *integer* (*default*=0)	Either a fixed number of lags (if 'fixed' is chosen for the 'info' option) or maximum lag length to consider when performing automatic lag length selection.
Test options	
model = *integer* (*default*=1)	Type of test: '1' (ADF), '2' (RADF), '3' (SADF), '4'(GSADF)
win = *number* (*default*=$\lfloor T(0.01 + 1.8/\sqrt{T})\rfloor$)	Initial window size (in fraction terms or in number of observations).
Simulation options	
usecurrent	Use previously simulated critical values that are stored in the workfile.
sim	Simulate critical values (Monte Carlo with EViews as default).
matlab	Parallel Monte Carlo simulation via MATLAB.
d = *number* (*default*=1)	See Equation (8). (Only relevant for Monte Carlo methods)
eta = *number* (*default*=1)	See Equation (8). (Only relevant for Monte Carlo methods)
theta = *number* (*default*=1)	See Equation (8). (Only relevant for Monte Carlo methods)
boot	Bootstrap critical values. (Wild bootstrap as default)
sieve	Bootstrap critical values using the sieve method.
bnone (*default*)	Do not include a constant or time trend in the bootstrap model.
bconst	Include a constant in the bootstrap model.
trend	Include a constant and a linear time trend in the bootstrap model.
blag	number of lags to include in the bootstrap model.
rep = *integer* (*default*=1000)	Simulation's number of replications.
beta = [0,1] (default=0.95)	Significance level for the critical values sequence. (see section 3.1.)
Trw	Use Tr_w instead of T (*default*) for calculating of the sequence of critical values.
Other options	
graph	Create a graph of the results.
print	Print output from the test.

where the available options are detailed in Table 3. Next, we show a couple of command line examples. First, the command:

```
snp.rtadf(const,model = 3, print)
```

performs an SADF test on the series SNP with the test equation including a constant term and prints the results. Second, the command:

```
nasdaq.rtadf(trend, model = 4, info = aic, lag = 4, win = 0.02, sim,
rep = 2000, graph, print)
```

performs a GSADF test on the series NASDAQ where the test equation includes a constant and a deterministic linear time trend and the initial window size is set to 2% of the sample, simulates critical values using 2000 replications, generates a graph and prints the results.

Third, the command:

```
nasdaq.rtadf(trend, model = 4, info = aic, lag = 4, win = 0.02, sim, matlab,
rep = 2000, graph, print)
```

is similar to the above command, only now the simulation is run in parallel mode using MATLAB.

Fourth, the command:

```
nasdaq.rtadf(trend, model = 4, info = aic, lag = 4, win = 0.02, usecurrent
graph, print)
```

is similar to the above command, only now the statistical inference is based on a previously run simulation.[15]

The following examples show how to use the command line options in the context of the bootstrap. First, the command:

```
brent.rtadf(model = 3, sim, boot, rep=999, graph, print)
```

runs the SADF test on the series BRENT and simulates critical values based on (wild) bootstrap using 999 replications, generates a graph and prints the results.

The command:

[15]Before using the 'usecurrent' option we need to make sure that the necessary objects exist in our main workfile. For example, is we run the GSADF test with 'usecurrent' the objects 'gsadf', 'gsadf_cv', 'gsadf_dist' and 'cv' must exist in the workfile.

```
brent.rtadf(model = 3, sim, boot, sieve, rep=999, graph, print)
```

is similar to the above command, only now the bootstrap method is set to 'sieve'.

Lastly, the command:

```
brent.rtadf(model = 3, sim, lag=2, boot, rep=999, bconst, blag=2, graph,

print)
```

runs the SADF test on the series BRENT where the lag length in the test regression is set to 2, and simulates critical values based on (wild) bootstrap using 999 replications, where the bootstrap model includes an intercept and two lagged first differences, generates a graph and prints the results.

6. Illustration

In this section we demonstrate in detail how to use Rtadf using two examples. In the first example we replicate the results of the SADF test on the S&P 500 price=to-dividend ratio. In the second example we demonstrate the use of the bootstrap version of the SADF test by replicating some of the results that appear in Harvey et al. (2016).

6.1. S&P 500 price-to-dividend ratio

We now turn to replicating the results reported in Phillips et al. (2015), Table 8 (pp. 33) and Figure 8 (pp. 35). The analysis is based on monthly data of the S&P 500 price-dividend ratio (the series object name in this example is SNP) over the period of 1871:M1 to 2010:M12 that includes 1680 observations, see Figure 7.[16]

In order to start the bubble detection process with the price-dividend ratio (SNP) series, first open the SNP series object and then click **Proc** → **Add-ins** → **Right tail ADF tests** (see Figure 8). Next, specify the test parameters as in PSY–see Figure 9(a) below–and then click the **OK** button. Note that just like in PSY, the initial window size is set to 36 observations (which constitutes approximately 2% of the whole sample), the lag length of the ADF test (p in Equation 9) is set to zero. Make sure that the "simulating critical values" option is checked so that the output includes the necessary critical values for testing the null hypothesis.

Simulation parameters' specifications are also set in accordance with PSY by adjusting the parameters in the simulation dialog box, which prompts right after clicking the **OK**

[16]The data used by in PSY can be downloaded from https://sites.google.com/site/shupingshi/PrgGSADF.zip?attredirects=0.

Figure 7. *S&P 500 price-dividend ratio, 1871:M1–2010:M12.*

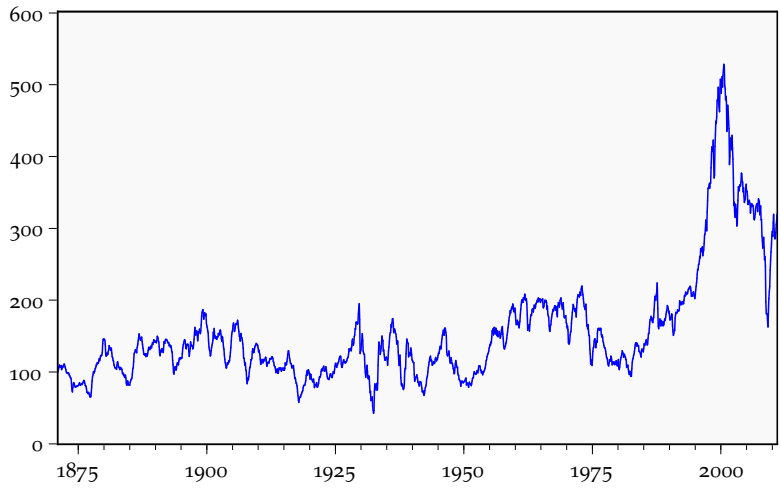
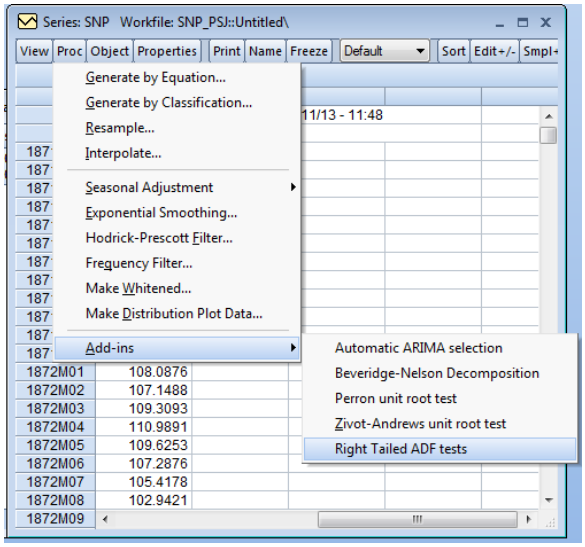


Figure 8. *S&P 500 object and the Rtadf proc menu.*



button in the main dialog box, see Figure 9(b). Clicking **OK** results in finite sample critical values for the conduction the SADF test.[17]

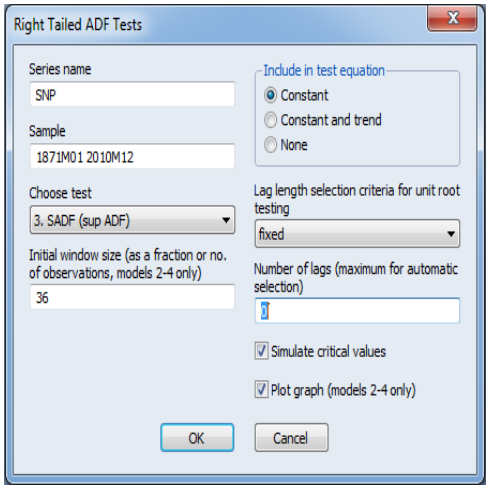
Summary output of the SADF test is displayed in Figure 10(a). The output is presented within the series object and it includes two panels.[18] The top panel is a table that holds the estimated SADF t-statistic followed by the corresponding (right tail) 90%, 95% and 99% critical values derived from the simulated statistic's distribution. Note that the sup value of this sequence is 3.443, whereas in Phillips et al. (2015) it equals 3.30. However,

[17] The simulation in this example may take a while since it involves running $(1680-36)\times 2000 = 3,288,000$ regressions. (On an Intel Core i5 with 4GB of memory it took over an hour.)

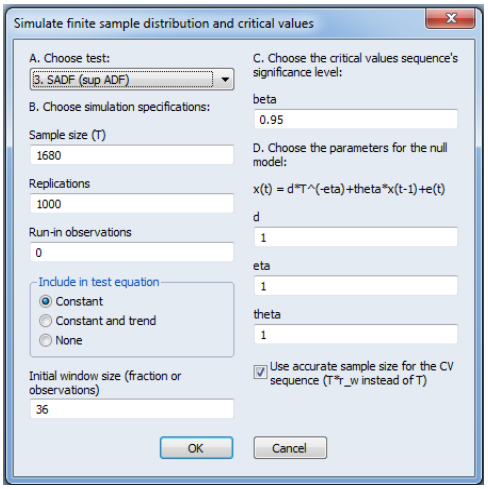
[18] In this example, the output was extracted to an EViews spool object by clicking on the 'Freeze' button.

Figure 9. *Parameter settings for the S&P 500 SADF test and simulation.*

(a) *Test dialog box.*



(b) *Simulation dialog box.*



using the MATLAB code published by PSY, gives an SADF statistic that equals 3.443.[19] The simulated critical values, which appear below the SADF statistic, match exactly those in PSY, Table 8.

Table 4. *Comparison of date-stamping procedures results for the S&P 500.*

Period No.	Phillips et al. (2015)	Rtadf
1	1879:M10–1880:M4	1879:M5–1880:M4
2	1997:M07–2001:M8	1997:M7–2002:M5

The bottom panel of the spool presents the date-stamping procedure for the SADF test. The graph includes the SNP series (in green), the ADF_{r_2} statistic sequence (in blue) and the corresponding 95% critical values sequence (in red). The data used to plot the graph are now available as series objects within the workfile under the names 'sadf' (the blue line) and 'sadf_cv' (the red line), see Figure 10(b).[20] The add-in successfully identifies two bubble periods, just like in PSY, though with minor differences in the start and end point (see Table 4). In addition, the add-in identifies one bubble period lasting four months in the beginning of the sample and a couple of 'blips' of bubbles lasting for one observation (i.e., one month). The source of discrepancy might be differences in the random number generator used by each software. However, if we ignore "too-short-lasting" bubble periods

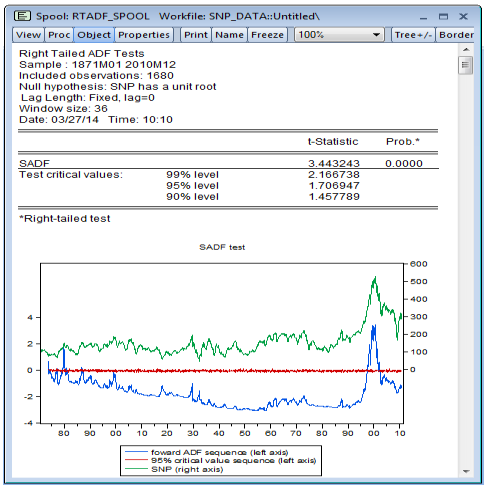
[19]The add-in's estimate of the GSADF statistic for the SNP series (not shown) is the same as the one in PSY (2013).

[20]The names of these series will change according to the test we use. For instance, if we apply the GSADF test, the output will include two new series objects named 'gsadf' (the sequence of the BSADF statistics values) and 'gsadf_cv' (the sequence of the corresponding critical values). Note that each run of the test runs over the previous output. Hence, if there is a need to save these series, copy them under different names.

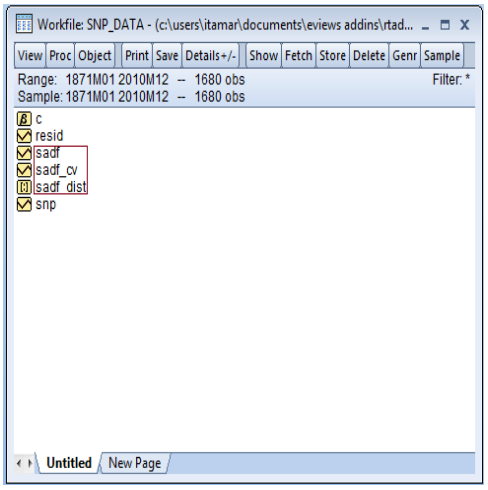
(PSY recommend to restrict identification to ones lasting more than $\log(T)$ units of time measures, which in this case equals $\log(1680) \approx 7$ months), the results are very similar. Note that the whole procedure described in this section can be accomplished via the

Figure 10. *S&P 500 SADF output.*

(a) *Output spool.*



(b) *Workfile with the added objects. (marked with the red ractangle.)*



execution of a single command line:

```
snp.rtadf(model = 3, win = 36, sim, rep = 2000, Trw, graph, print)
```

Concluding the illustration, the SADF test results point to the presence of *at least* one bubble in the S&P 500 price index at the 1% significance level (since 3.443 > 2.141). However, since there seems to be evidence for at least two bubble periods (1879-80 and 1997-2002), as a second step (which is not pursued here), there is a strong support to using the GSADF test in the next step.

6.2. *WTI and Brent oil prices*

Next, we turn to replicating the results reported in Harvey et al. (2016), Table 4 (pp. 566). In their paper, Harvey et al. (2016) apply the standard SADF test along with a wild bootstrap version of the test to several real commodity prices. Our illustration focuses on monthly data of real WTI and Brent oil prices (the series object names in this example are WIT_R and BRENT_R) over the period of 2000:M1 to 2013:M12 (168 observations), see Figure 7.[21] For comparison purposes, the number of lags and minimal window size are set as in Harvey et al. (2016) – BIC and $r_0 = 0.15$.

[21]The data used in this example was kindly provided by David I. Harvey and A.M. Taylor.

The commands

```
wti_r.rtadf(model=3,win=0.15,info=sic,lag=6,sim,rep=9999,print,graph)
```

and

```
brent_r.rtadf(model=3,win=0.15,info=sic,lag=6,sim,rep=9999,print,graph)
```

calculate the SADF statistic (which is denoted as PWY statistic in Harvey et al.) and provide critical values p-values by Monte Carlo methods with 9,999 replications (which is denoted as PWY in Harvey et al.).

The commands

```
wti_r.rtadf(model=3,win=0.15,info=sic,lag=6,sim,boot,rep=9999,print,graph)
```

and

```
brent_r.rtadf(model=3,win=0.15,info=sic,lag=6,sim,boot,rep=9999,print,graph)
```

calculate the SADF statistic (denoted as PWY statistic by Harvey et al.) and provide critical values p-values based on the wild bootstrap with 9'999 replications (denoted as PWY* by Harvey et al.).

Table 5 presents the output resulting from the above commands, along with the corresponding results that appear in Harvey et al. As can bee seen in the third column of the table, the estimated SADF statsitics, for both WTI_r and BRENT_r exactly match those shown in Harvey et al. (2016). The fourth and fifth column of the table hold the p-values that are derived from both Monte Carlo and the bootstrap methods. Notably the Rtadf p-values differ from the ones reported in Harvey et al. Nonetheless, the qualitative conclusions remain roughly the same.[22]

Table 5. *Comparison of SADF test result for the real 4WTI and Brent oil prices.*

Series	Source	SADF statistic	p-value	
			Monte Carlo	Bootstrap
BRENT_R	Rtadf	2.073	0.019	0.1033
	Harvey et al. (2016)	2.073	0.026	0.097
WTI_R	Rtadf	2.230	0.014	0.1056
	Harvey et al. (2016)	2.230	0.021	0.105

[22]The discrepancies in the Monte Carlo simulations based p-values are mostly due to the fact that within the simulation, the Rtadf add-in runs the ADF regressions without any differenced lags while Harvey et al. (2016) run ADF regressions with lag length chosen according to the BIC.

7. Concluding remarks

This paper presents a new EViews add-in, Rtadf, that implements newly developed asset price bubble detection strategies, all based on right tail versions of the standard reduced form ADF unit root test, where the null of unit root is tested against the alternative of a mildly explosive process. In this case, rejection of the null for a specific time series may serve as evidence of an asset price bubble.

This paper began with a short background on the methodological developments of reduced form econometric approaches for bubble detection alongside a theoretical asset pricing model which helps to clarify the rationale behind the reduced form approach. Next, we gave a brief technical discussion on the bubble detection tests included in Rtadf, and finally, two illustration of using the add-in in the context of the S&P 500 stocks index and oil prices were presented.

References

Blanchard, O. J. and Watson, M. W. (1982). Bubbles, Rational Expectations and Financial Markets. *NBER Working Paper*, (w0945).

Brunnermeier, M. K. (2008). Bubbles. In Durlauf, S. N. and Blume, L. E., editors, *The New Palgrave Dictionary of Economics*. Palgrave Macmillan, Basingstoke.

Campbell, J. Y. and Shiller, R. J. (1988). The Dividend-Price Ratio and Expectations of Future Dividends and Discount Factors. *Review of financial studies*, 1(3):195–228.

Diba, B. T. and Grossman, H. I. (1988). Explosive Rational Bubbles in Stock Prices? *The American Economic Review*, 78(3):520–530.

Evans, G. W. (1991). Pitfalls in Testing for Explosive Bubbles in Asset Prices. *The American Economic Review*, 81(4):922–930.

Gürkaynak, R. (2008). Econometric tests of asset price bubbles: Taking stock. *Journal of Economic Surveys*, 22(1):166–186.

Gutierrez, L. (2011). Bootstrapping asset price bubbles. *Economic Modelling*, 28(6):2488–2493.

Harvey, D. I., Leybourne, S. J., Sollis, R., and Taylor, A. R. (2016). Tests for explosive financial bubbles in the presence of non-stationary volatility. *Journal of Empirical Finance*, 38:548–574.

Homm, U. and Breitung, J. (2012). Testing for Speculative Bubbles in Stock Markets: A Comparison of Alternative Methods. *Journal of Financial Econometrics*, 10(1):198–231.

Iraola, M. A. and Santos, M. S. (2008). Speculative Bubbles. In Durlauf, S. N. and Blume, L. E., editors, *The New Palgrave Dictionary of Economics*. Palgrave Macmillan, Basingstoke.

MATLAB (2010). *version 7.10.0 (R2014a)*. The MathWorks Inc., Natick, Massachusetts.

Milunovich, G., Shi, S.-P., and Tan, D. (2016). Bubble detection and sector trading in real time. *Available at SSRN 2827051*.

Phillips, P. C. B. and Magdalinos, T. (2007). Limit Theory for Moderate Deviations from a Unit Root. *Journal of Econometrics*, 136(1):115–130.

Phillips, P. C. B., Shi, S., and Yu, J. (2015). Testing for multiple bubbles: Historical episodes of exuberance and collapse in the S&P 500. *International Economic Review*, 56(4):1043–1078.

Phillips, P. C. B., Wu, Y., and Yu, J. (2011). Explosive behavior in the 1990s NASDAQ: When did exuberance escalate asset values? *International Economic Review*, 52(1):201–226.

Phillips, P. C. B. and Yu, J. (2011). Dating the Timeline of Financial Bubbles During the Subprime Crisis. *Quantitative Economics*, 2(3):455–491.

Quantitative Micro Software (2016). *EViews, Version 9.5*. Irvine CA, USA.