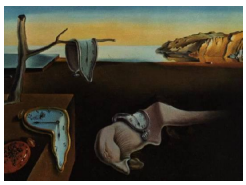

Journal of Statistical Software
October 2009, Volume 32, Issue 2. http://www.jstatsoft.org/

Unit Root CADF Testing with R

Claudio Lupi
University of Molise

Abstract

This paper describes **CADFtest**, an R package for testing for the presence of a unit root in a time series using the covariate-augmented Dickey-Fuller (CADF) test proposed in Hansen (1995b). The procedures presented here are user friendly, allow fully automatic model specification, and allow computation of the asymptotic p values of the test.

Keywords: unit root, stationary covariates, asymptotic p values, R.

1. Introduction and statistical background

Testing for a unit root is a frequent problem in macroeconomic modeling and forecasting. Although many tests have been developed so far, the practitioner's workhorse in this field is still probably the augmented Dickey-Fuller (ADF) test (Dickey and Fuller 1979, 1981; Said and Dickey 1984), which is known to have good size but low power under many conditions.[1] However, the literature on unit root testing has largely proceeded in a univariate framework, with notable exceptions being represented by the panel unit root tests (see Choi 2006b, for a recent survey). In fact, reality is hardly univariate at all and, although univariate representations of multivariate time series exist (see e.g., Zellner and Palm 1974), nevertheless reckoning explicitly the multivariate nature of most economic time series can in principle lead to better testing procedures. In a seminal paper, Hansen (1995b) suggests that, when testing for a unit root, a viable way to exploit the information embodied in related series and increase power is to consider stationary covariates in an otherwise conventional ADF framework. Unless the variable of interest is independent of the stationary covariates considered in the analysis, by the Neyman-Pearson lemma the most powerful test makes use of the information embodied

[1] However, large negative MA components are well known to have adverse effects on the size of the ADF test (see e.g., Schwert 1989). An interesting summary of many Monte Carlo results on different unit root tests can be found in Stock (1994). Haldrup and Jansson (2006) is a recent survey of the methods proposed to improve the size and the power of unit roots tests.

in the stationary covariates themselves (Hansen 1995b, p. 1152). As a consequence, not considering the multivariate dimension of the problem leads to a loss in the power of the test. Using covariates also allows to some extent to couple unit root testing and economic theory, because economic theory can be used as a guideline to chose the appropriate covariates to be included in the analysis (see e.g., Amara and Papell 2006; Elliott and Pesavento 2006), although other approaches can be used as well (Lee and Tsong 2009).

Formally, Hansen (1995b) considers a univariate time series y_t composed of a deterministic and a stochastic component such that

$$\begin{aligned} y_t &= d_t + s_t && (1)\\ a(L)\Delta s_t &= \delta s_{t-1} + v_t && (2)\\ v_t &= b(L)'\left(\Delta x_t - \mu_{\Delta x}\right) + e_t && (3) \end{aligned}$$

where d_t is the deterministic term (usually a constant or a constant and a linear trend), $a(L) := (1 - a_1 L - a_2 L^2 - \ldots - a_p L^p)$ is a polynomial in the lag operator L, Δx_t is a vector of *stationary* covariates, $\mu_{\Delta x} := \mathsf{E}(\Delta x)$, $b(L) := (b_{q_2} L^{-q_2} + \ldots + b_{q_1} L^{q_1})$ is a polynomial where both leads and lags are allowed for. Furthermore, the long-run (zero-frequency) covariance matrix $\Omega := \sum_{k=-\infty}^{\infty} \mathsf{E}\left(\epsilon_t \epsilon_{t-k}'\right)$, with $\epsilon_t := (v_t, e_t)'$ can be defined, from which the long-run squared correlation between v_t and e_t, ρ^2, can be derived. When Δx_t has no explicative power on the long-run movement of v_t, then $\rho^2 \approx 1$. On the contrary, when Δx_t explains nearly all the zero-frequency variability of v_t, then $\rho^2 \approx 0$. The case $\rho^2 = 0$ is ruled out for technical reasons (see Hansen 1995b, p. 1151). It should be noted that this restriction excludes the possibility that the variable y_t be cointegrated with the cumulated stationary covariate(s).

As with the ADF test, Hansen's CADF test is based on different models, according to the different deterministic kernels that the investigator may wish to consider. For example, the model with constant and linear trend is

$$a(L)\Delta y_t = \mu + \theta\, t + \delta y_{t-1} + b(L)'\Delta x_t + e_t\,. \tag{4}$$

Similarly to the conventional ADF test, the CADF test is based on the t-statistic for δ, $\widehat{t(\delta)}$, with the null hypothesis being that a unit root is present, i.e. $H_0 : \delta = 0$, against the one-sided alternative $H_1 : \delta < 0$. Hansen (1995b) refers to the test statistic as the CADF(p, q_1, q_2) statistic.

Hansen (1995b, p. 1154) proves that under the unit-root null, if conventional weak dependence and moment restrictions hold, $\widehat{t(\delta)}$ in the model without deterministic terms is such that

$$\widehat{t(\delta)} \Rightarrow \rho \frac{\int_0^1 W\, dW}{\left(\int_0^1 W^2\right)^{1/2}} + \left(1 - \rho^2\right)^{1/2} N(0,1) \tag{5}$$

where $\Rightarrow$ denotes weak convergence, W is a standard Wiener process, and $N(0,1)$ is a standard normal independent of W. It is interesting to note that (5) is the distribution of a weighted sum of a Dickey-Fuller and a standard normal random variable. If a model with constant or a model with constant and trend are considered, the standard Wiener process W in (5) has to be replaced by a demeaned or a detrended Wiener process, respectively. Note that the asymptotic distribution of the test statistic depends on the nuisance parameter ρ^2 but, provided ρ^2 is given, it can be simulated using standard techniques (see e.g., Hatanaka 1996).

Hansen (1995b, p. 1155) provides the asymptotic critical values of the test, while here we offer a practical way to compute its p values.

Hansen's CADF test is firmly rooted in the ADF tradition and for this reason it can be more familiar and attractive to practitioners than other tests, although Elliott and Jansson (2003) show that Hansen's CADF test is not the point optimal test in general. Feasible point optimal tests in the presence of deterministic components are developed in Elliott and Jansson (2003). A quite different approach is used in this case, the feasible tests being based on VAR models. However, Monte Carlo simulations reported in Elliott and Jansson (2003) suggest that power gains with respect to Hansen's CADF test can be obtained at the cost of slightly worse size performances.

In this paper we present the R (R Development Core Team 2009) package **CADFtest** that allows users to perform Hansen's CADF unit root test easily.[2] The main function `CADFtest()` returns a "`CADFtest`" class object that not only contains the test statistic, but also its asymptotic p value and many other useful details. In fact, the class "`CADFtest`" inherits from the class "`htest`",[3] so that no special `print()` method is needed. However, dedicated `summary()` and `plot()` methods have been developed in order to allow the user to analyze the test results more in detail. A specialized `update()` method is also available that ease testing using different options.

The remainder of the paper is structured as follows: Section 2 discusses the way Hansen's CADF test has been implemented in the function `CADFtest()`, and some applications are illustrated in Section 3. In Section 4, the method to compute the asymptotic p values is illustrated in detail along with the use of the function `CADFpvalues()`. Section 5 offers some comparisons with other existing R packages performing the ADF test. A summary is offered in Section 6.

2. Implementation and use of the function `CADFtest()`

In principle, carrying out Hansen's CADF test is no more complicated than carrying out an ordinary ADF test. What makes things more complicated is the presence of the nuisance parameter ρ^2 in the asymptotic distribution (5). In fact, a consistent estimate of ρ^2 has to be derived in order to choose the correct asymptotic critical value and/or to compute the correct asymptotic p value of the test. The problem is solved into two steps. First, $\hat{e}_t$ and $\hat{v}_t$ are derived; then, their long-run covariance matrix Ω is estimated using a HAC covariance estimator (see e.g., Andrews 1991; Zeileis 2004, 2006; Kleiber and Zeileis 2008).

Once the kind of model (no constant, with constant, with constant and trend) has been chosen, using `CADFtest()` the investigator can either set the polynomial orders p, q_2 and q_1 to fixed values, or can decide the maximum value for each and let the procedure to select and estimate the model according to different information criteria.

In order to estimate ρ^2 it is necessary to estimate e_t and v_t first. For example, if the model with constant and trend (4) is used, then e_t and v_t are estimated as

$$\hat{e}_t = \widehat{a(L)}\Delta y_t - \hat{\mu}^* - \hat{\theta}\, t - \hat{\delta}_\tau y_{t-1} - \widehat{b(L)}' \Delta x_t \tag{6}$$

$$\hat{v}_t = \widehat{b(L)}' \left(\Delta x_t - \overline{\Delta x}\right) + \hat{e}_t \tag{7}$$

[2] The present paper describes version 0.3-0 of the package.

[3] A fairly detailed description of the "`htest`" class can be gathered from within R by typing `?t.test`.

where "$\hat{}$" denote parameters estimated by ordinary least squares and $\overline{\Delta x}$ is the sample average of Δx_t. Once $\hat{e}_t$ and $\hat{v}_t$ have been computed, a kernel-based HAC covariance estimator (Andrews 1991) is used to estimate Ω and hence ρ^2. In order to estimate ρ^2 in a rather flexible way, in `CADFtest()` the `kernHAC()` function included in the **sandwich** package (Zeileis 2004, 2006) is used. This allows the investigator to chose the kernel to be applied, the bandwidth, and if and how prewhitening should be performed. Differently from Hansen (1995b) where a Parzen kernel without prewhitening is used, the default choice in `CADFtest()` is a quadratic spectral kernel with VAR(1) prewhitening. The bandwidth is adaptively chosen using the method proposed in Andrews (1991), but the user is free to change any of these default choices.

The usage of the function is extremely simple:

```
CADFtest(model, X = NULL, type = c("trend", "drift", "none"),
     data = list(), max.lag.y = 1, min.lag.X = 0, max.lag.X = 0,
     dname = NULL, criterion = c("none", "BIC", "AIC", "HQC", "MAIC"), ...)
```

The minimal required input is `CADFtest(y)`, where `y` can be either a vector or a time series. However, if no stationary covariate is specified, an ordinary ADF test is performed. In fact, the ordinary ADF test can be carried out with R using other existing packages such as **fUnitRoots** (Wuertz 2009), **tseries** (Trapletti 2009) and **urca** (Pfaff 2008). In this respect there would be no need to add one further package. However, given that the ADF test can be seen as a particular case of the more general CADF test, it seems logical to leave the user the opportunity to carry out both tests in the same framework, using the same conventions and allowing for the computation of the test p values. Furthermore, as will be shown in Section 3, the interface to `CADFtest()` is very flexible and intuitive, and the results are easy to read: this can make carrying out conventional ADF tests using `CADFtest()` very appealing. As far as the computation of the p values of the ADF test is concerned, `CADFtest()` exploits the facilities offered by `punitroots()` implemented in the package **urca** (Pfaff 2008) that uses the method proposed in MacKinnon (1994, 1996). In principle, it would have been possible (and easy) to use the function `CADFpvalues()` described in detail in Section 4, but given that MacKinnon (1996) describes a method to compute approximate finite sample, rather than asymptotic, p values, it seems fair to refer directly to a function that implements this procedure. However, note that MacKinnon (1996) derives the finite sample p values for Gaussian data: in non-Gaussian settings the finite sample p values are not necessarily more accurate than the asymptotic ones.

All the arguments, with the exception of `min.lag.X` and `max.lag.X` that are relevant only when a CADF test is carried out, work irrespective of the test being ADF or CADF. However, if a proper CADF test has to be performed, at least a stationary covariate must be passed to the procedure. The covariates are passed in a very simple way, using a formula (model) statement. For example, suppose we want to test the variable y using x_1 and x_2 as stationary covariates: if the other default options are accepted, then it is sufficient to specify `CADFtest(y ~ x1 + x2)`. Note that the formula that is passed as argument to the `CADFtest()` function is not the complete model to be used, but it just indicates which variable has to be tested for a unit root (`y`) and which are to be used as stationary covariates in the test (`x1` and `x2`). A formula statement can be used also to specify an ordinary ADF test by typing `CADFtest(y ~ 1)`, where the term "`1`" does not imply that a model with constant will be used, but it simply means that no stationary covariate is passed to the procedure (the deterministic kernel is

always defined by the argument `type` described below).

Other arguments are used to specify the deterministic kernel to be used in the model (`type`), the lead and lag orders (`max.lag.y`, `min.lag.X`, `max.lag.X`), and if the model has to be fixed or selected using a `criterion` such as `"AIC"` (Akaike 1973), `"BIC"` (Schwarz 1978), `"HQC"` (Hannan and Quinn 1979) or `"MAIC"` (Ng and Perron 2001). Indeed, given that a number of competing models with potentially many regressors have to be compared, information criteria offer a handy solution. Furthermore, Hall (1994) shows that when the data are generated by an ARIMA(p_0, 1, 0) process, then the distribution of the ADF test statistic under the null converges asymptotically to the correct distribution when the number of lags in the empirical model is determined by using either the AIC, the BIC, or the HQC. On the other hand, Ng and Perron (2001) argue that standard information criteria should be modified to take into account the fact that the series are $I(1)$ under the null and propose their modified AIC (MAIC) that should be more robust in the presence of negative moving-average errors. Ng and Perron's MAIC is computed by `CADFtest()` in the OLS-detrended version suggested by Perron and Qu (2007). However, notice that although the MAIC should in principle work well also in the CADF framework, its usefulness has been proved only in the simpler ADF context.

When no stationary covariate is passed to the procedure, then lag selection is obviously limited to the lags of Δy_t, but when a proper CADF test is performed, then model selection implies the joint determination of the lags of the differenced dependent variable and the leads and lags of the stationary covariates as well. If `criterion = "none"` (the default choice) is specified, no automatic model selection is performed and the lag orders are fixed to the values passed to the procedure. In particular, `max.lag.y` corresponds to p, the lag order of $a(L)$ in (4), and it is set to 1 by default: it can be set equal to 0 or to a positive integer. `min.lag.X` corresponds to q_2, the maximum lead in $b(L)$ in (4), and it is equal to 0 by default: if modified, it must be set equal to a negative integer (a negative lag is a lead). `max.lag.X` correspond to q_1, the maximum lag in $b(L)$ in (4), and the default choice is 0: if modified, it must be set equal to a positive integer. If `criterion` is different from `"none"`, then all the models with lags polynomials up to the specified orders (of both the y and the covariates) are estimated and the final model to be used is selected on the basis of the chosen `criterion`. The deterministic components to be used in the model are specified using the conventions utilized in the R package **urca** (Pfaff 2008). The default value (`type = "trend"`) implies that the model with constant and trend (4) is used. If `type = "drift"` or `type = "none"` is specified, then the model with constant or the model without deterministic components are utilized. `data` is the data set to be used and `dname` is the name of data: in general there is no need to change `dname`, given that it is automatically computed by the function itself, unless one wants to indicate for example that a specific data set has been used. Further arguments can be passed to the procedure to control the parameters to be used in the HAC covariance estimation. These further arguments can be passed using the conventions valid for the command `kernHAC()` (see the package **sandwich**: Zeileis 2004, 2006). If Hansen's results have to be replicated, then `kernel = "Parzen"` and `prewhite = FALSE` have to be specified, otherwise a quadratic spectral kernel with VAR(1) prewhitening is used by default.

The function `CADFtest()` returns an object of class "`CADFtest`" containing the test statistic (`statistic`), the p value of the test (`p.value`), the lag structure of the selected model (`max.lag.y`, `min.lag.X`, `max.lag.X`), the value of the information criteria (`AIC`, `BIC`, `HQC`, `MAIC`), the estimated value of ρ^2 (`parameter`), and the full estimated model (`est.model`).

Other returned information concern the nature of the test (either CADF or ADF) stored in `method`, the name of data used (`data.name`), the value of δ under the null (`null.value`), the description of the alternative (`alternative`) and the estimated value of δ (`estimate`).

A summary of the test can be obtained just by using a `print()` command. Given that the class "`CADFtest`" inherits from the class "`htest`", the `print()` command produces the standard R output of the "`htest`" class. However, the `summary()` command is also allowed that returns a more detailed account of the test results. For greater flexibility, `print()` can be applied to a `CADFtestsummary` object (produced by `summary()`) to control further printing options. For example, the number of significant digits can be controlled by `digits`, while significance stars can be avoided by setting `signif.stars = FALSE`.

3. Some examples of application

We provide here some simple examples of application of the function `CADFtest()`. Data are taken from the R package **urca** (Pfaff 2008) and refer to the extended Nelson and Plosser (1982) data set used in Schotman and Van Dijk (1991). These are the same data used in Hansen (1995b), so that we will be able to replicate some of the empirical applications proposed there.

First, we load the data and the required package **CADFtest**: all the following examples assume that both have been loaded.

```
R> data("npext", package = "urca")
R> library("CADFtest")
```

A complete description of the data can be retrieved simply by typing `?npext` in R.

We first replicate the analysis carried out in Hansen (1995b, p. 1165) by testing for the presence of a unit root in the log per capita US real GNP using a standard ADF test with constant, trend and three lags:

```
R> ADFt <- CADFtest(npext$gnpperca, max.lag.y = 3)
```

The p value of the test is stored in `ADFt$p.value` and it is easily accessible:

```
R> ADFt$p.value
```

```
[1] 0.08082208
```

As already mentioned, the *finite sample* p value is computed using `punitroots()` implemented in package **urca** (Pfaff 2008). In principle, it would have been possible also to compute the *asymptotic* p value by using the function `CADFpvalues` to be described in detail in the next section by invoking

```
R> CADFpvalues(ADFt$statistic, type = "trend", rho2 = 1)
```

```
[1] 0.07589502
```

When a standard Dickey-Fuller test is performed, `CADFtest()` acts as an interface to existing commands. For example, in the case above equation (4) is estimated using the package **dynlm** (Zeileis 2009) and the test p value is computed using `punitroots()`.

Even if all the results are readily accessible, a summary of the test can be obtained just by typing

```
R> print(ADFt)

        ADF test

data:  npext$gnpperca
ADF(3) = -3.2606, p-value = 0.08082
alternative hypothesis: true delta is less than 0
sample estimates:
     delta
-0.2014652
```

The function correctly warns the user that a conventional ADF test has been performed and reports the main results along with the number of lags used in the test.

If we want to obtain a more detailed summary that includes the details of the estimated model, we can just type

```
R> summary(ADFt)

Augmented DF test
                                                ADF test
t-test statistic:                           -3.26058935
p-value:                                     0.08082208
Max lag of the diff. dependent variable:     3.00000000

Call:
dynlm(formula = formula(model), start = obs.1, end = obs.T)

Residuals:
      Min        1Q    Median        3Q       Max
-0.163620 -0.025697  0.007439  0.026647  0.147798

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.201825   0.370695   3.242  0.00182 **
trnd         0.004016   0.001203   3.339  0.00135 **
L(y, 1)     -0.201465   0.061788  -3.261  0.08082 .
L(d(y), 1)   0.391840   0.110751   3.538  0.00072 ***
L(d(y), 2)   0.060429   0.119135   0.507  0.61358
L(d(y), 3)  -0.052543   0.115921  -0.453  0.65176
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05309 on 70 degrees of freedom
Multiple R-squared: 0.2586,        Adjusted R-squared: 0.2057
F-statistic: 5.142 on 3 and 70 DF,  p-value: 0.002855
```

The model output uses the same conventions utilized in the package **dynlm** (Zeileis 2009): `trnd` is the deterministic linear trend, `L(y, 1)` stands for y_{t-1} and `L(d(y), i)` represents Δy_{t-i}. Note that the p value of the lagged dependent variable refers to the unit root null and is therefore consistent with the test p value. Note also that, differently from the conventional usage, the F statistic here pertains to the joint significance of the *stationary* regressors, so that under the null it has the standard F distribution (Sims, Stock, and Watson 1990). If a simple DF test is performed, then the F-statistic is not computed and a `NA` value is returned. If more control on the output summary is desired, then it is possible to store the summary results in an object (of class "`CADFtestsummary`") and print it using `print()` with the desired options (for example, `digits = 3, signif.stars = FALSE`).

Further details about the test can be gathered from the estimated residuals and from residuals plots. The function `residuals()` can be used to extract the estimated residuals as in the following line:

```
R> res.ADFt <- residuals(ADFt)
```

The extracted residuals can then be used in any desired analysis. Fairly informative plots can also be easily produced by a `plot()` command as in

```
R> plot(ADFt)
```

Four plots are produced by default, as in Figure 1. In particular, the standardized residuals, the estimated residuals density along with the test for normality proposed in Jarque and Bera (1980), the estimated residuals autocorrelation function (ACF) and partial autocorrelation function (PACF) are plotted. However, any combination of these plots can be produced as well. For example, if the residuals density is not needed, then it is sufficient to specify

```
R> plot(ADFt, plots = c(1, 3, 4))
```

to produce a visualization as in Figure 2.

In order to show other useful features of the `CADFtest()` command, we carry out now a few data transformations:

```
R> npext$unemrate <- exp(npext$unemploy)
R> L <- ts(npext, start = 1860)
R> D <- diff(L)
R> S <- window(ts.intersect(L, D), start = 1909)
```

Data are now interpreted as annual time series starting in 1860. The sample ends in 1988 (this is easy to verify by invoking the `tsp()` function). Given that `unemploy` is the log of the unemployment rate, while we need the unemployment rate, the series in levels used by

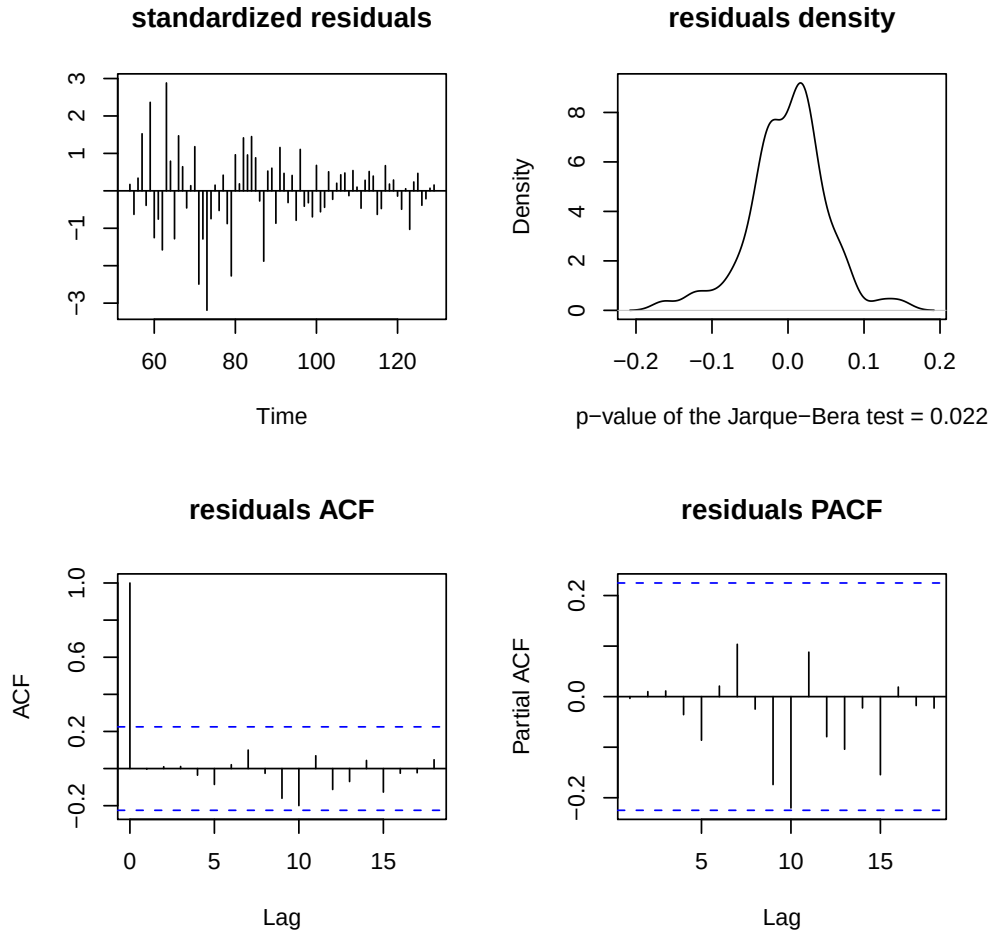

Figure 1: Standardized residuals, residuals density, residuals ACF and residuals PACF.

Hansen (1995b) is computed. The time series in levels are stored in `L`, while `D` stores the first differences of the original variables, that will be used as stationary covariates in the CADF tests. `S` contains all the series over a common sample that starts in 1909, as in Hansen (1995b).

The ADF test could have been also performed by invoking

```
R> ADFt <- CADFtest(L.gnpperca ~ 1, data = S, max.lag.y = 3)
```

Since no stationary covariate is explicitly indicated in the model, the test is performed as an ordinary ADF test (with constant and trend and three lags, as before).

Automatic lag selection can be achieved by using the `criterion` argument in the `CADFtest()` command. For example, in the following we fix the maximum lag order p of Δy_{t-p} to $p = 4$ and let the final model to be selected by the BIC, possibly highlighting that the data are from the extended Nelson and Plosser (1982) data set used by Schotman and Van Dijk (1991):

```
R> CADFtest(L.gnpperca ~ 1, data = S, max.lag.y = 4, criterion = "BIC",
+    dname = "Extended Nelson-Plosser data")
```

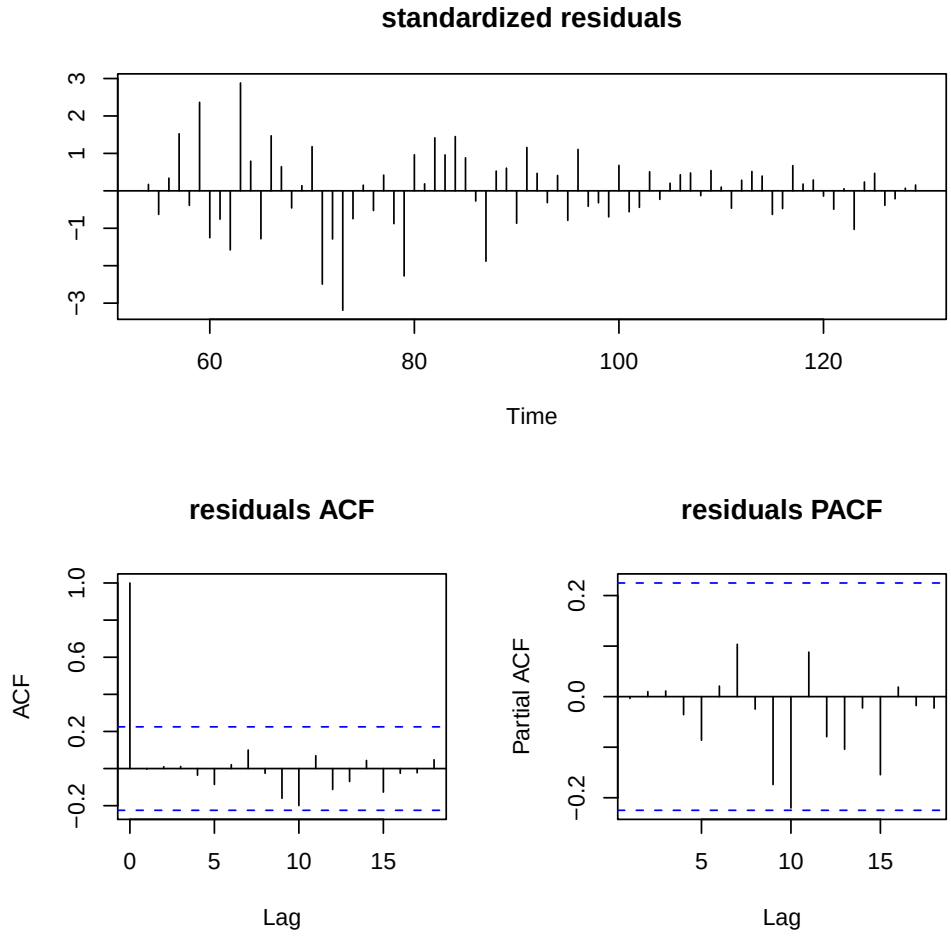

Figure 2: Standardized residuals, residuals ACF and residuals PACF.

```
        ADF test

data:  Extended Nelson-Plosser data
ADF(1) = -3.678, p-value = 0.03002
alternative hypothesis: true delta is less than 0
sample estimates:
     delta
-0.2041227
```

or by updating the object `ADFt` that contains the results of a previous test:

```
R> ADFt2 <- update(ADFt, change = list("max.lag.y = 4", "criterion = 'BIC'",
+    "dname = 'Extended Nelson-Plosser data'"))
```

Of course, when automatic lag selection is enabled, all the models are estimated on the same common sample.

Let's now turn back again to our ADF(3) test performed by using `CADFtest()`:

```
R> ADFt <- CADFtest(L.gnpperca ~ 1, data = S, max.lag.y = 3)
```

Now, suppose that we want to run Hansen's CADF test on the log-real GNP per capita by using the first difference of the unemployment rate as a stationary covariate. The test is carried out with constant and trend and allowing 3 lags for the (differences of the) dependent variable and 0 lags for the covariate. For the results to be fully consistent with Hansen (1995b, Table 8, column 2, p. 1166), `kernel = "Parzen"` and `prewhite = FALSE` have to be specified. Given the last ADF was carried out using 3 lags, we can perform the CADF test just by calling

```
R> CADFt <- update(ADFt, change = list("+ D.unemrate", "kernel = 'Parzen'",
+    "prewhite = FALSE"))
R> print(CADFt)

        CADF test

data:  L.gnpperca ~ D.unemrate
CADF(3,0,0) = -3.413, rho2 = 0.064, p-value = 0.001729
alternative hypothesis: true delta is less than 0
sample estimates:
      delta
-0.08720302
```

Differently from Hansen (1995b), we not only verify that the test is significant at the asymptotic 1% level, but we can also give a precise assessment of the test p value.

Besides the CADF(3,0,0) test, Hansen's original analysis includes some other CADF tests, namely the CADF(3,2,0), CADF(3,2,2), CADF(3,0,2). Instead of using different tests in this way, we rather specify the maximum lag for the dependent and the maximum lead and lag for the stationary covariate, and leave the model to be selected by using the BIC. Of course, the AIC, HQC or MAIC could have been used as well and the orders of all the lags polynomials would have been again selected automatically:

```
R> CADFt <- update(CADFt, change = list("max.lag.X = 3", "min.lag.X = -3",
+    "criterion = 'BIC'"))
R> print(CADFt)

        CADF test

data:  L.gnpperca ~ D.unemrate
CADF(0,2,0) = -4.4072, rho2 = 0.011, p-value = 8.18e-05
alternative hypothesis: true delta is less than 0
sample estimates:
     delta
-0.1086331
```

The selected model is CADF(0,2,0) and the null is rejected for any reasonable confidence level. The last `update()` is equivalent to

```
R> CADFt <- CADFtest(L.gnpperca ~ D.unemrate, data = S, max.lag.y = 3,
+    max.lag.X = 3, min.lag.X = -3, criterion = "BIC", kernel = "Parzen",
+    prewhite = FALSE)
```

Of course, if desired the test can be easily carried out using more than just one covariate. In fact, it is sufficient to specify the model accordingly as in

```
R> CADFt <- CADFtest(L.gnpperca ~ D.unemrate + D.indprod, data = S,
+    max.lag.y = 3, max.lag.X = 3, min.lag.X = -3, criterion = "BIC",
+    kernel = "Parzen", prewhite = FALSE)
```

or using the `update()` command.

4. p values computation and the function `CADFpvalues()`

The possibility of computing the p values of a test greatly increases the chances that the test is effectively used by practitioners. This is *a fortiori* true when the test procedure requires the use of non-standard tables available only in few specialized papers. There are even instances where computation of the p values is necessary for further investigations, as is the case for some panel unit root tests (see e.g., Maddala and Wu 1999; Choi 2001, 2006a; Costantini, Lupi, and Popp 2007). The R function `CADFpvalues()` presented here allows the computation of asymptotic p values of the CADF test proposed in Hansen (1995b). `CADFpvalues()` is used within the `CADFtest()` function to compute the p values of the test along with the other test results already discussed. However, `CADFpvalues()` can also be used separately from the main testing procedure.

The method used to compute the p values has been originally proposed in Costantini *et al.* (2007) and is based on a response surface approach similar to that proposed in MacKinnon (1994, 1996) for the p values of the ADF test (classical references on the estimation and use of response surfaces are, among others Hendry 1984; Ericsson 1986). Differently from what happens with reference to the Dickey-Fuller distribution which is free from nuisance parameters, the asymptotic distribution of the CADF test statistic depends on the nuisance parameter $0 < \rho^2 \leq 1$, so that the asymptotic distribution (5) has to be simulated over a grid of values for ρ^2. When $\rho^2 = 1$ the distribution coincides with the ordinary Dickey-Fuller distribution.

In order to obtain fairly good approximations, here a grid of 40 values $\rho^2 \in \{0.025, 0.050, 0.0725, \ldots, 1\}$ is considered. For each of the three models (without deterministic components, with constant, with constant and linear trend), 100,000 replications[4] have been used for each value of ρ^2. The Wiener functionals have been simulated using a standard approach (see e.g., Hatanaka 1996, p. 67) with $T = 5,000$ (for the "no constant", "constant" and "constant plus trend" case, standard, demeaned and detrended Wiener processes have been used, respectively). On the basis of the simulated values, for each value of ρ^2 1,005 asymptotic quantiles q_ρ have been derived corresponding to the probabilities $p = (0.00025, 0.00050,$

[4]Simulations have been carried out using R.

0.00075, ..., 0.001, 0.002, ..., 0.998, 0.999, 0.99925, 0.99950, 0.99975). As a result, we obtained a $1,005 \times 40$ matrix of estimated quantiles. Along the rows of the matrix it is possible to read how a given quantile varies with ρ^2. Indeed, the estimated quantiles vary very smoothly with ρ^2 (see Costantini *et al.* 2007).

For each row of the quantile matrix the model

$$q_\rho(p) = \beta_0 + \beta_1\rho^2 + \beta_2\left(\rho^2\right)^2 + \beta_3\left(\rho^2\right)^3 + \epsilon \tag{8}$$

is estimated and the $\hat{\beta}$'s are saved in a $1,005 \times 4$ table. The tables of estimated coefficients for the "no constant", "constant" and "constant plus trend" case, respectively are used by the function `CADFpvalues()` in order to compute the asymptotic p values for any value of $0 < \rho^2 \leq 1$ for the relevant model.[5]

The way the computation of the p values proceeds in `CADFpvalues()` is essentially the following:

1. The relevant table of parameters is read, depending on the specific model used ("no constant", "constant" or "constant plus trend").

2. For any desired value ρ_0^2 of ρ^2, the estimated parameters are used to compute for all the 1,005 probability values p the fitted quantiles $\widehat{q_{\rho_0}}(p)$ as

$$\widehat{q_{\rho_0}}(p) = \widehat{\beta_0} + \widehat{\beta_1}\rho_0^2 + \widehat{\beta_2}\left(\rho_0^2\right)^2 + \widehat{\beta_3}\left(\rho_0^2\right)^3 . \tag{9}$$

3. The approach suggested in MacKinnon (1994, 1996) can now be used on $\widehat{q_{\rho_0}}$ to derive the p value. First, given the value $\widehat{t(\delta)}$ of the test statistic, it is necessary to find the fitted quantile $\widehat{q_{\rho_0}}$ that is closest to $\widehat{t(\delta)}$ and the corresponding probability $\tilde{p}$.

4. The regression

$$\Phi^{-1}(p) = \gamma_0 + \gamma_1\widehat{q_{\rho_0}}(p) + \gamma_2\widehat{q_{\rho_0}}^2(p) + \gamma_3\widehat{q_{\rho_0}}^3(p) + \nu_p \tag{10}$$

   where $\Phi^{-1}(p)$ is the inverse of the cumulative standard normal distribution is estimated locally on an interval of p centered on $\tilde{p}$. In `CADFpvalues()` local interpolation takes place using 11 values centered on $\tilde{p}$.

5. The p value associated with the estimated test statistic $\widehat{t(\delta)}$ is finally obtained from

$$\Phi\left(\hat{\gamma}_0 + \hat{\gamma}_1\widehat{t(\delta)} + \hat{\gamma}_2\widehat{t(\delta)}^2 + \hat{\gamma}_3\widehat{t(\delta)}^3\right) . \tag{11}$$

While computation is rather involved, from the user's viewpoint the usage of the function is extremely simple:

```
CADFpvalues(t0, rho2 = 0.5, type = "trend")
```

[5]The estimated tables of coefficients are available from within the package.

where `t0` is the value of the test statistic $\widehat{t(\delta)}$, `rho2` is the estimated value of ρ^2, and `type` assumes the values `"trend"` (the default), `"drift"` or `"none"` as discussed above when a model with constant plus trend, with constant or without constant is considered.

For example, suppose that we want to know the p values of the tests reported in Hansen (1995b, Table 10). These tests are carried out using models with constant and trend. Specifically, consider the CADF(3,0,0) and CADF(3,2,0) whose test statistics are -2.2 and -1.7, with $\hat{\rho}^2$ equal to 0.53 and 0.20, respectively. The computation of the p values of these tests is immediate:

```
R> CADFpvalues(t0 = -2.2, rho2 = 0.53)

[1] 0.2447352

R> CADFpvalues(t0 = -1.7, rho2 = 0.2)

[1] 0.2189253
```

It is now clear that both tests do not reject the null.

If desired, `CADFpvalues()` can be used also to compute the asymptotic p values of the ordinary ADF test, as shown above in Section 3. In fact, it is sufficient to set `rho2 = 1` to obtain the asymptotic p values of the Dickey-Fuller distribution. For example

```
R> CADFpvalues(-0.44, type = "drift", rho2 = 1)

[1] 0.9018844
```

computes a p value that can be compared directly with the values reported for example in Banerjee, Dolado, Galbraith, and Hendry (1993, Table 4.2).

5. Other R implementations of the ADF test

While **CADFtest** implements Hansen's covariate augmented Dickey-Fuller test and includes the ADF test as a special case, other R packages can perform the ADF test. However, we believe that the function `CADFtest()` has a more flexible and convenient interface than other existing functions have.

urca (Pfaff 2008) is a leading R package for the analysis of integrated and cointegrated time series that includes the `ur.df()` function for the ADF test. However, this command cannot deal with missing values and does not have a `data` argument. Therefore, in order to carry out the same ADF test with three lags we did before, we need to specify all the data details manually, leaving out the first 49 observations for which we have missing values:

```
R> library("urca")
R> adf.urca <- ur.df(npext$gnpperca[-(1:49)], type = "trend", lags = 3)
```

Apart from the call being less flexible, the results are not as easy to read as are those that can be obtained from `CADFtest()`. Only the critical values, taken from Dickey and Fuller (1981) and Hamilton (1994), are available to judge the significance of the test, while the test p value is not reported. In fact, if a `summary()` is performed, the p value associated to the coefficient of the lagged dependent variable is computed using the t distribution that is obviously incorrect under the null. Automatic lag selection is possible on the basis of the AIC and BIC criteria (not the HQ or the MAIC) and a `plot()` method is available that, when applied to `adf.urca`, produces a result similar to Figure 2.

As with `ur.df()`, `adf.test()` in package **tseries** (Trapletti 2009) does not allow for missing values. The typical call would be

```
R> library("tseries")
R> adf.tseries <- adf.test(npext$gnpperca[-(1:49)], k = 3)
```

The output is easily interpretable, but no `summary()` or `plot()` methods are offered. Furthermore, the model is restricted to the case with constant and trend only, and the p value is computed using a simplified procedure based on the interpolation of the values reported in Banerjee *et al.* (1993, Table 4.2, p. 103). Finally, `adf.test()` does not offer automatic lag selection options.

fUnitRoots (Wuertz 2009) is another important R package performing unit root tests. In particular, the function `unitrootTest()` performs the ADF test and computes the relevant finite sample p values using the approach developed in MacKinnon (1996). However, no automatic lag selection is performed. The object passed to the procedure must be a vector or a time series, but no data argument is used, so that the usual call would be something like

```
R> library("fUnitRoots")
R> adf.fUnitRoots <- unitrootTest(npext$gnpperca, lags = 3, type = "ct")
```

The function can cope with leading missing values, in the sense that the model is correctly estimated. However, in our example the initial 49 missing values are considered in the length of the series, so that the finite sample p value is incorrect. In order to obtain correct p values, no missing values should be present so that in our case the series should again be adjusted manually.

6. Summary

This paper presents the R package **CADFtest** that allows unit root testing using the covariate-augmented Dickey-Fuller (CADF) test originally proposed in Hansen (1995b).

Differently from the already available routines written in GAUSS and in MATLAB (Hansen 1995a), the present functions are easy to use, do not require the user to modify the programs, and allow the computation of the asymptotic p values of the tests. Beside being extremely useful in general, p values computation opens to the possibility of using the CADF tests in unit root combination tests, for example in the context of macro panels (see e.g., Maddala and Wu 1999; Choi 2001, 2006a; Costantini *et al.* 2007). When used to perform conventional ADF tests, **CADFtest** also encompasses the main features of other existing R packages, with a more flexible and intuitive interface.

CADFtest can be downloaded from the Comprehensive R Archive Network (CRAN) at http://CRAN.R-project.org/package=CADFtest.

Computational details

The functions illustrated in this paper have been implemented in R version 2.9.2 (R Development Core Team 2009) using the following packages, listed in alphabetical order: **dynlm** version 0.2-3 (Zeileis 2009), **sandwich** version 2.2-1 (Zeileis 2004, 2006), **tseries** version 0.10-19 (Trapletti 2009), **urca** version 1.2-2 (Pfaff 2008).

Acknowledgments

I would like to thank Achim Zeileis for his many comments and suggestions that helped me in improving on previous versions of the package. Both the software and the paper greatly benefited from the detailed comments I received from two anonymous referees and an associate editor. I am grateful to Mauro Costantini and Stephan Popp for comments and discussion. Of course, none of them is responsible for any remaining error. I owe a special thank you to the authors of the R packages used in the development of **CADFtest**.

This text was typeset in LaTeX using R (R Development Core Team 2009) and `Sweave()` (Leisch 2002, 2003).

References

Akaike H (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In BN Petrov, F Csaki (eds.), *Second International Symposium on Information Theory*, pp. 267–281. Akademiai Kaido, Budapest.

Amara J, Papell DH (2006). "Testing for Purchasing Power Parity Using Stationary Covariates." *Applied Financial Economics*, **16**(1-2), 29–39.

Andrews DWK (1991). "Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation." *Econometrica*, **59**(3), 817–858.

Banerjee A, Dolado J, Galbraith JW, Hendry DF (1993). *Co-Integration, Error-Correction, and the Econometric Analysis of Non-Stationary Data*. Advanced Texts in Econometrics. Oxford University Press, Oxford.

Choi I (2001). "Unit Root Tests for Panel Data." *Journal of International Money and Finance*, **20**(2), 249–272.

Choi I (2006a). "Combination Unit Root Tests for Cross-Sectionally Correlated Panels." In D Corbae, SN Durlauf, BE Hansen (eds.), *Econometrics Theory and Practice: Frontiers of Analysis and Applied Research*, chapter 12, pp. 311–333. Cambridge University Press, Cambridge.

Choi I (2006b). "Nonstationary Panels." In TC Mills, K Patterson (eds.), *Econometric Theory*, volume 1 of *Palgrave Handbook of Econometrics*, chapter 13, pp. 511–539. Palgrave MacMillan, New York.

Costantini M, Lupi C, Popp S (2007). "A Panel-CADF Test for Unit Roots." *Economics and Statistics Discussion Paper 39/07*, University of Molise. URL http://econpapers.repec.org/paper/molecsdps/esdp07039.htm.

Dickey DA, Fuller WA (1979). "Distributions of the Estimators for Autoregressive Time Series with a Unit Root." *Journal of the American Statistical Association*, **74**(366), 427–431.

Dickey DA, Fuller WA (1981). "Likelihood Ratio Statistics for Autoregressive Time Series with a Unit Root." *Econometrica*, **49**(4), 1057–1072.

Elliott G, Jansson M (2003). "Testing for Unit Roots with Stationary Covariates." *Journal of Econometrics*, **115**(1), 75–89.

Elliott G, Pesavento E (2006). "On the Failure of Purchasing Power Parity for Bilateral Exchange Rates after 1973." *Journal of Money, Credit, and Banking*, **38**(6), 1405–1430.

Ericsson NR (1986). "Post-Simulation Analysis of Monte Carlo Experiments: Interpreting Pesaran's (1974) Study of Non-Nested Hypothesis Test Statistics." *Review of Economic Studies*, **53**(4), 691–707.

Haldrup N, Jansson M (2006). "Improving Size and Power in Unit Root Testing." In TC Mills, K Patterson (eds.), *Econometric Theory*, volume 1 of *Palgrave Handbook of Econometrics*, chapter 7, pp. 252–277. Palgrave MacMillan, Basingstoke.

Hall A (1994). "Testing for a Unit Root in Time Series with Pretest Data-Based Model Selection." *Journal of Business and Economic Statistics*, **12**(4), 461–470.

Hamilton JD (1994). *Time Series Analysis*. Princeton University Press, Princeton, NJ.

Hannan EJ, Quinn BG (1979). "The Determination of the Order of an Autoregression." *Journal of the Royal Statistical Society B*, **41**(2), 190–195.

Hansen BE (1995a). ***UR_REG**: GAUSS and MATLAB Procedures to Compute the Covariate Augmented Dickey-Fuller Test*. URL http://www.ssc.wisc.edu/~bhansen/progs/et_95.html.

Hansen BE (1995b). "Rethinking the Univariate Approach to Unit Root Testing: Using Covariates to Increase Power." *Econometric Theory*, **11**(5), 1148–1171.

Hatanaka M (1996). *Time-Series-Based Econometrics: Unit Roots and Cointegration*. Advanced Texts in Econometrics. Oxford University Press, Oxford.

Hendry DF (1984). "Monte Carlo Experimentation in Econometrics." In Z Griliches, M Intriligator (eds.), *Handbook of Econometrics*, volume 2, chapter 16, pp. 937–976. Elsevier Science Publishers, Amsterdam.

Jarque CM, Bera AK (1980). "Efficient Tests for Normality, Homoscedasticity and Serial Independence of Regression Residuals." *Economics Letters*, **6**(3), 255–259.

Kleiber C, Zeileis A (2008). *Applied Econometrics with R*. Springer-Verlag, New York.

Lee CF, Tsong CC (2009). "Covariate Selection for Testing Purchasing Power Parity." *Applied Economics*. Forthcoming.

Leisch F (2002). "Dynamic Generation of Statistical Reports Using Literate Data Analysis." In W Härdle, B Rönz (eds.), *COMPSTAT 2002 – Proceedings in Computational Statistics*, pp. 575–580. Physica-Verlag, Heidelberg.

Leisch F (2003). "Sweave and Beyond: Computations on Text Documents." In K Hornik, F Leisch, A Zeileis (eds.), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing, Vienna, Austria*. ISSN 1609-395X, URL http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/.

MacKinnon JG (1994). "Approximate Asymptotic Distribution Functions for Unit-Root and Cointegration Tests." *Journal of Business and Economic Statistics*, **12**(2), 167–176.

MacKinnon JG (1996). "Numerical Distribution Functions for Unit Root and Cointegration Tests." *Journal of Applied Econometrics*, **11**(6), 601–618.

Maddala GS, Wu S (1999). "A Comparative Study of Unit Root Tests with Panel Data and a New Simple Test." *Oxford Bulletin of Economics and Statistics*, **61**(Supplement 1), 631–652.

Nelson CR, Plosser CR (1982). "Trends and Random Walks in Macroeconomic Time Series: Some Evidence and Implications." *Journal of Monetary Economics*, **10**(2), 139–162.

Ng S, Perron P (2001). "Lag Length Selection and the Construction of Unit Root Tests with Good Size and Power." *Econometrica*, **69**(6), 1519–1554.

Perron P, Qu Z (2007). "A Simple Modification to Improve the Finite Sample Properties of Ng and Perron's Unit Root Tests." *Economics Letters*, **94**(1), 12–19.

Pfaff B (2008). *Analysis of Integrated and Cointegrated Time Series with R*. 2nd edition. Springer-Verlag, New York.

R Development Core Team (2009). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Said SE, Dickey DA (1984). "Test for Unit Roots in Autoregressive-Moving Average Models of Unknown Order." *Biometrika*, **71**(3), 599–607.

Schotman PC, Van Dijk HK (1991). "On Bayesian Routes to Unit Roots." *Journal of Applied Econometrics*, **6**(4), 387–401.

Schwarz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**(2), 461–464.

Schwert GW (1989). "Tests for Unit Roots: A Monte Carlo Investigation." *Journal of Business and Economic Statistics*, **7**(2), 147–159.

Sims CA, Stock JH, Watson MW (1990). "Inference in Linear Time Series Models with Some Unit Roots." *Econometrica*, **58**(1), 113–144.

Stock JH (1994). "Unit Roots, Structural Breaks and Trends." In RF Engle, DL McFadden (eds.), *Handbook of Econometrics*, volume 4, chapter 46, pp. 2739–2841. Elsevier Science Publishers, Amsterdam.

Trapletti A (2009). ***tseries**: Time Series Analysis and Computational Finance*. R package version 0.10-19, URL http://CRAN.R-project.org/package=tseries.

Wuertz D (2009). ***fUnitRoots**: Trends and Unit Roots*. R package version 260.75, URL http://CRAN.R-project.org/package=fUnitRoots.

Zeileis A (2004). "Econometric Computing with HC and HAC Covariance Matrix Estimators." *Journal of Statistical Software*, **11**(10), 1–17. URL http://www.jstatsoft.org/v11/i10/.

Zeileis A (2006). "Object-Oriented Computation of Sandwich Estimators." *Journal of Statistical Software*, **16**(9), 1–16. URL http://www.jstatsoft.org/v16/i09/.

Zeileis A (2009). ***dynlm**: Dynamic Linear Regression*. R package version 0.2-3, URL http://CRAN.R-project.org/package=dynlm.

Zellner A, Palm FC (1974). "Time Series Analysis and Simultaneous Equation Econometric Models." *Journal of Econometrics*, **2**(1), 17–54.

Affiliation:

Claudio Lupi
Department of Economics, Management and Social Sciences (SEGeS)
University of Molise
Via De Sanctis I-86100 Campobasso, Italy
E-mail: lupi@unimol.it

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 32, Issue 2 *Submitted:* 2008-12-29
October 2009 *Accepted:* 2009-08-10