The Stata Journal (2021)
21, Number 1, pp. 39–50 DOI: 10.1177/1536867X211000003

Bootstrap unit-root test for random walk with drift: The bsrwalkdrift command

Miguel Dorta
StataCorp
College Station, TX
mdorta@stata.com

Gustavo Sanchez
StataCorp
College Station, TX
gsanchez@stata.com

Abstract. In this article, we introduce the command `bsrwalkdrift`, which is primarily intended to perform a bootstrap unit-root test under the null hypothesis of random walk with drift. The method implemented in this command is considerably more precise than the corresponding case of the conventional augmented Dickey–Fuller test, which can be inaccurate when the true value of the drift term is small relative to the standard deviation of the innovations. The command also has an option to account for deterministic linear trend and another option to perform bootstrap unit-root tests under the null hypothesis of random walk without drift.

Keywords: st0626, bsrwalkdrift, dfuller, drift, bootstrap, unit root, random walk

1 Introduction

One particular case of the augmented Dickey–Fuller (ADF) test is when the null hypothesis is that the data-generation process (DGP) corresponds to a random walk with a nonzero drift. For this case, Hamilton (1994, 495–497) proves that the asymptotic distribution for the test statistic is standard normal. So he suggested that the standard ordinary least-squares (OLS) t and F statistics can be compared with the standard t and F distribution when performing the test on finite samples (see Hamilton [1994, 497]). This method is implemented in the `dfuller` command by specifying the option `drift`.[1]

Although the t distribution can be a good approximation for large enough samples, it is rather inaccurate for small or even medium sample sizes, especially when the true value of the drift term is small relative to the standard deviation of the innovations. For example, our simulations with the `dfuller, drift` command, with a drift term of 0.1, a standard deviation of 1, a sample size of 100 observations, and 2,000 simulation replicates, produced a mean rejection rate of 0.360. That rate looks too far from the nominal rejection probability of 0.05. When we increased the drift term to 0.25 and the sample size to 200, the mean rejection rate was 0.132, which still does not seem near enough to 0.05. This issue was previously investigated by Hylleberg and Mizon (1989).

To overcome the problem above, we wrote the `bsrwalkdrift` command, which uses a bootstrapping technique based on the method proposed by Park (2003). He considers the ADF unit-root tests for autoregressive unit-root models. According to Park (2003),

1. We emphasize that the `drift` option of the `dfuller` command exactly implements the test as defined in Hamilton (1994, 497). The method is asymptotically correct but has poor size performance in finite samples.

testing with bootstrap critical values is expected to correct for the discrepancy between the finite sample and nominal rejection rates. He performed simulations that showed rejection probabilities close to their nominal values. Although his method was originally defined under the null hypothesis of random walk without drift, he also states, "It can be clearly seen that our methodology may also be used to analyze many other unit root tests as well." In this sense, we use a straightforward adaptation for the null hypothesis of a random walk with drift.

Although we primarily wrote `bsrwalkdrift` to correct for the issue described above, we also included an option to account for deterministic linear trend and another option to test the hypothesis of random walk without drift. They allow for testing strategies that require analyzing more than one case, and they are expected to have better finite sampling properties under particular situations that can affect the conventional ADF tests (for example, DGP with nonnormal innovations).

The remainder of the article is organized as follows. Section 2 describes the method implemented in the command. Section 3 presents the syntax for `bsrwalkdrift` and includes an example to illustrate the use of the command. Section 4 presents the simulation results to evaluate the performance of the `bsrwalkdrift` command compared with the `dfuller, drift` command. Section 5 concludes.

2 The method

The method to compute the bootstrap critical value for the unit-root test is based on Park (2003). We test the null hypothesis

$$H_0 : \delta = 0$$

in the model

$$\Delta y_t = \alpha + \delta y_{t-1} + \sum_{i=1}^{p} \beta_i \Delta y_{t-i} + \epsilon_t \tag{1}$$

where ϵ_t is an independent and identically distributed sequence such that $E(\epsilon_t) = 0$ and $E(|\epsilon_t|^r) < \infty$ for $r > 1$.

We begin by fitting the regression

$$\Delta y_t = \alpha + \sum_{i=1}^{p} \beta_i \Delta y_{t-i} + \epsilon_t \tag{2}$$

on a series defined as $(y_{-p}, \ldots, y_0, y_1, \ldots, y_T)$. According to Park (2003), the resampling should be performed from the restricted model (2) instead of the unrestricted model (1). For more details on this, see Basawa et al. (1991).

Then, we compute the estimated residuals $(\widehat{\epsilon_t})$, draw bootstrap samples, and obtain demeaned residuals

$$\left(\widehat{\epsilon_t} - \frac{1}{n} \sum_{i=1}^{n} \widehat{\epsilon_i} \right) \qquad t = 1, \ldots, n$$

denoting them by $(\widetilde{\epsilon_t})$. Notice that without demeaning, the mean of the bootstrap sample would not be zero.

For each bootstrap sample $(\widetilde{\epsilon_t})$, we generate (u_t^*) recursively from $(\widetilde{\epsilon_t})$ using

$$u_t^* = \widehat{\alpha} + \sum_{i=1}^{p} \widehat{\beta}_i u_{t-i}^* + \widetilde{\epsilon_t} \tag{3}$$

starting from $(u_{-p+1}, \ldots, u_0)$. Next, we generate the bootstrap samples (y_t^*) from y_0 through

$$y_t^* = y_0 + \sum_{i=1}^{t} u_{t-i}^* \tag{4}$$

The initializations for (3) and (4) are obtained by fixing $(y_{-p}, \ldots, y_0)$ and using $u_t = \Delta y_t$ to calculate $(u_{-p+1}, \ldots, u_0)$. According to Park (2003), the initializations are important to make his theory applicable; however, they are irrelevant for the models with constant and deterministic trend.

For each bootstrap sample (y_t^*), we fit regressions

$$\Delta y_t^* = \alpha^* + \delta^* y_{t-1}^* + \sum_{i=1}^{p} \beta_i^* \Delta y_{t-i}^* + v_t$$

and collect the t statistics for δ^*. The set of those values constitutes a bootstrap approximation to the null distribution of the t statistic. Therefore, given a significance level of λ, the bootstrap critical value for the test is the corresponding $\lambda\times$100th percentile.

Finally, we fit the regression

$$\Delta y_t = \mu + \delta y_{t-1} + \sum_{i=1}^{p} \beta_i \Delta y_{t-i} + \epsilon_t$$

on the original sample and compute the t statistic associated with δ, which is to be compared with the bootstrap critical value described above. If the t statistic is less than the bootstrap critical value, we reject the hypothesis that there is a unit root. In addition, we compute the p-value as the proportion of the number of values, from the bootstrap approximation to the null distribution, that is less than the t statistic.

2.1 Model with deterministic linear trend

Trending time series could be stationary around a deterministic trend, or the trending behavior could be stochastic. For trending series, the first step of the analysis should be testing for unit root, accounting for a possible deterministic trend. If the series is actually stationary around a deterministic trend, unit-root testing using models without trend is expected to have no power (Campbell and Perron 1991).

Park (2003) dedicated a section of his article to models with deterministic trends. He proposed to first detrend the series and then apply his bootstrapping algorithm to the detrended series. In the case of linear trend, the detrended series may be obtained by fitting the OLS regression

$$y_t = \widehat{\beta}_0 + \widehat{\beta}_1 t + y_t^d$$

and using the residuals y_t^d as the detrended series. In this case, the alternative hypothesis is that the original series is stationary around a deterministic linear trend.

Park (2003) provided analytical proof that testing on a detrended series and the ADF test with trend are asymptotically equivalent not only in the first order but also in the second order.

3 The command bsrwalkdrift

3.1 The command

Syntax

The command syntax is

`bsrwalkdrift` *varname* [*if*] [*in*] [`, lags(#) bsreps(#) siglevel(#) seed(#) nodrift detrend selecic(`*stat*`) maxlag(#) plot nodots`]

The data must be `tsset` before using the command.

Options

`lags(#)` specifies the number of lagged differences. The default is `lags(1)`.

`bsreps(#)` specifies the number of bootstrap replicates. The default is `bsreps(500)`.

`siglevel(#)` specifies the significance level for the test. The default is `siglevel(0.05)`.

`seed(#)` specifies the random-number seed.

`nodrift` specifies that the DGP is a random walk without drift for the null hypothesis.[2]

`detrend` detrends the series assuming a linear trend and performs the test on the detrended series. If this option is specified, the command automatically activates the `nodrift` option.

`selecic(`*stat*`)` specifies that the lag order be selected by minimizing information criteria. `maxlag(#)` is required if `selecic(`*stat*`)` is specified. `selecic(aic)` specifies the Akaike information criterion (AIC). `selecic(bic)` specifies the Bayesian infor-

2. When the `nodrift` option is specified, the method is the particular case where α, $\widehat{\alpha}$, and α^* are all set to zero.

mation criterion (BIC). `selecic(`*stat*`)` fits a sequence of regressions using (2) for $p = 0, 1, 2, \ldots, P$ on a common estimation sample, which is the sample for $p = P$. The AIC or BIC is computed for each regression, and the selected lag order corresponds to the minimum value. P is set by `maxlag(#)`.

`maxlag(#)` sets the maximum lag order to #. This option is required if `selecic(`*stat*`)` is specified. # must be greater than 1 and less than a third of the sample size.

`plot` creates a kernel density plot of the bootstrap null distribution.

`nodots` suppresses replication dots.

Stored results

`bsrwalkdrift` stores the following in `r()`:

Scalars

`r(N)`	number of observations
`r(lags)`	number of lags
`r(Zt)`	test statistic
`r(siglevel)`	significance level
`r(critval_bs)`	bootstrap critical value
`r(pval)`	*p*-value
`r(aic)`	AIC statistic, if `selecic(aic)` is specified
`r(bic)`	BIC statistic, if `selecic(bic)` is specified
`r(bsreps)`	number of bootstrap replicates

3.2 Example

We analyze a quarterly series for real gross private domestic investment in the United States for the period from 1980q1 to 2020q1.[3] In figure 1, we plot the natural logarithm of real gross private domestic investment (variable `ln_rgpdinv`). As can be seen, the series has a clear upward trend. It does not seem clear, however, whether the trend is deterministic or stochastic. To clarify the doubt, our testing strategy begins by performing the bootstrap unit-root test on the detrended series by specifying the `detrend` option. If we reject the null hypothesis of random walk, it would be evidence that the series is stationary around a deterministic linear trend. If instead we fail to reject the null hypothesis, the trend would likely be stochastic; more specifically, the series would be a random walk with drift. This is precisely the default case for the `bsrwalkdrift` command, and so we can use it to confirm that.

3. The data were obtained from the Federal Reserve Bank of St. Louis (Federal Reserve Economic Data) using Stata's `import fred` command.

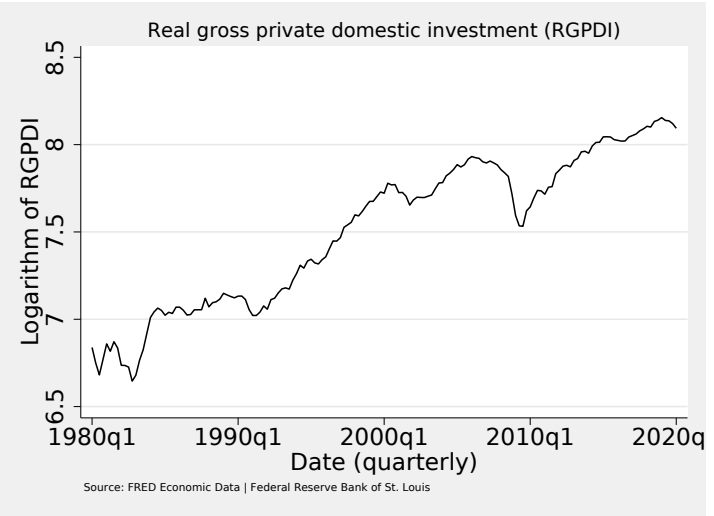

Figure 1. Natural logarithm of real gross private domestic investment

We begin by using `bsrwalkdrift` with the `detrend` option to test that `ln_rgpdinv` is a random-walk process against the alternative that the series is stationary around a linear trend. We tell the command to select the lag order using the AIC. To do so, we specify the options `selecic(aic)` and `maxlag(8)`. We also specify 5,000 bootstrap replicates, keep the default 0.05 significance level, and specify some other options as well.

```
. use rgpdinv
. tsset qdate
        time variable:  qdate, 1980q1 to 2020q1
                delta:  1 quarter
. bsrwalkdrift ln_rgpdinv, selecic(aic) maxlag(8) detrend plot nodots
> bsreps(5000) seed(1413)
Test performed on detrended series assuming a linear trend, which
is asymptotically equivalent to the ADF test with linear trend.
Lag-order selection using common
estimations samples
```

Lag	N	AIC
0	152	-616.0392
1	152	-637.903
2	152	-640.0416
3	152	-639.2999
4	152	-640.1445
5	152	-638.5531
6	152	-638.5844
7	152	-636.6319
8	152	-634.8304

```
 Selected lag order = 4
OLS auxiliary regression on detrended series
```

D.ln_rgpdinv	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
ln_rgpdinv						
L1.	-.0512055	.0217138	-2.36	0.020	-.09411	-.008301
LD.	.2827119	.0771007	3.67	0.000	.1303681	.4350556
L2D.	.097219	.0785152	1.24	0.218	-.0579195	.2523576
L3D.	.1962017	.0772655	2.54	0.012	.0435324	.348871
L4D.	-.1681958	.0742893	-2.26	0.025	-.3149844	-.0214072
_cons	-.0008574	.0024417	-0.35	0.726	-.0056818	.0039671

```
Number of observations = 156
Performing bootstrap (5000 replicates) ...
Results of the bootstrap unit root test
=======================================
H_0: Random Walk without drift
H_1: Stationary series around a linear trend
Number of lags = 4
Number of bootstrap replicates = 5000
Significance level = .05
Test statistic =   -2.3582   --   Bootstrap Critical value =   -3.0084
P-value =    0.2228
```

`bsrwalkdrift` issues an explanatory note associated with the `detrend` option (see the output). After that, the command reports a table with the AIC up to lag 8 followed by the selected lag order, which is lag 4 in this case. Then, it follows the OLS auxiliary regression for the detrended series. After that, the command reports the results of the bootstrap testing method. The most important results are the test statistic, which is taken from the OLS auxiliary regression; the bootstrap critical value; and the

p-value based on the bootstrap null distribution. Because the value of the test statistic (-2.3582) is larger than the bootstrap critical value (-3.0084), the null hypothesis cannot be rejected. The p-value (0.2228) leads to the same conclusion, which will always be the case if compared with the specified significance level. Hence, we can rule out that the series is stationary around a linear trend. Because the `plot` option was specified, the command creates a kernel density plot of the bootstrap null distribution, with vertical dotted lines representing the bootstrap critical value and the test statistic (see figure 2).

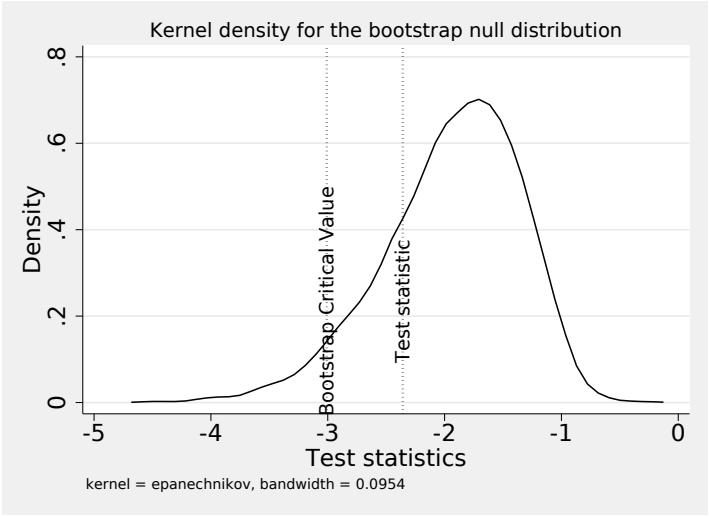

Figure 2. Kernel density plot of the bootstrap null distribution for the detrended series

So far, the evidence is against stationarity around a linear trend and in favor of stochastic trend. Thus, we now use the default case of `bsrwalkdrift` (random walk with drift); and so, we just remove the `detrend` option and maintain the other options previously specified.

```
. bsrwalkdrift ln_rgpdinv, selecic(aic) maxlag(8) plot nodots bsreps(5000)
> seed(1413)
Lag-order selection using common
estimations samples
```

Lag	N	AIC
0	152	-616.0295
1	152	-634.9757
2	152	-635.4621
3	152	-633.7591
4	152	-636.5777
5	152	-635.7435
6	152	-634.7038
7	152	-633.196
8	152	-631.9053

```
Selected lag order = 4
```

```
OLS auxiliary regression
```

D.ln_rgpdinv	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
ln_rgpdinv						
L1.	-.0046374	.0060268	-0.77	0.443	-.0165458	.0072711
LD.	.2730164	.0782828	3.49	0.001	.118337	.4276959
L2D.	.0714819	.0790616	0.90	0.367	-.0847364	.2277001
L3D.	.171264	.0777387	2.20	0.029	.0176597	.3248682
L4D.	-.204087	.0737032	-2.77	0.006	-.3497176	-.0584564
_cons	.0399913	.0454015	0.88	0.380	-.0497177	.1297004

```
Number of observations = 156
Performing bootstrap (5000 replicates) ...
Results of the bootstrap unit root test
========================================
H_0: Random Walk with drift
H_1: Stationary series
Number of lags = 4
Number of bootstrap replicates = 5000
Significance level = .05
Test statistic =   -0.7695   --   Bootstrap Critical value =   -2.3527
P-value =    0.4816
```

The selected lag order was again lag 4. The test statistic (-0.7695) is larger than the bootstrap critical value (-2.3527), which must be consistent with the p-value (0.4816); therefore, the null hypothesis of random walk with drift cannot be rejected. This result with the detrended case indicates that the series is actually a random walk with drift. Figure 3 is the plot of the corresponding kernel density for the bootstrap null distribution.

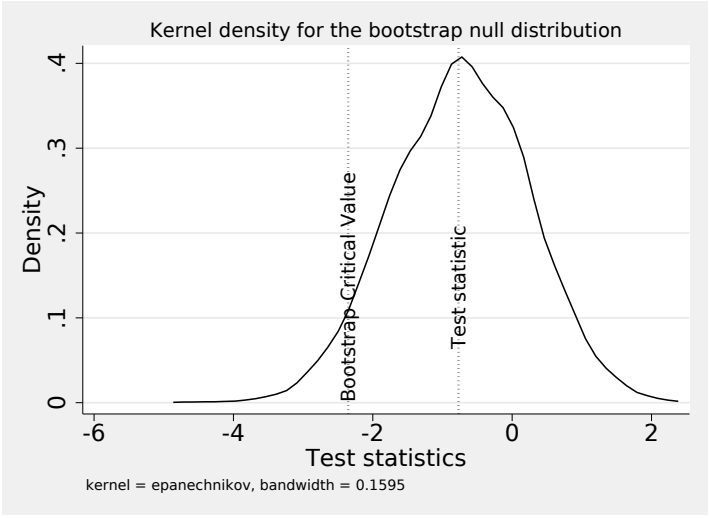

Figure 3. Kernel density plot of the bootstrap null distribution

4 Simulations

We evaluated some finite-sample properties of `bsrwalkdrift` by performing Monte Carlo simulations.

The model is

$$y_t = \alpha + y_{t-1} + \beta_1 \Delta y_{t-1} + \beta_2 \Delta y_{t-2} + e_t$$

where

$e_t \sim N(0, 1)$

$\beta_1 = 0.5$

$\beta_2 = -0.2$

$\alpha = 0.05$, 0.1, 0.25, 0.5, and 0.75

nobs = 25, 50, 100, 200, and 500

For each combination of α and nobs (sample size), we generated the data for y with the model above, and we used

```
. bsrwalkdrift y, lag(2) bsreps(200) siglevel(0.05)
> dfuller y, drift lag(2)
```

for 2,000 simulation replicates, where `bsrwalkdrift` uses 200 bootstrap replicates for each simulation replicate. The rejection results were stored for all the replicates, and then we estimated the mean rejection rates for a 0.05 significance level.

Below, we used the `tabdisp` command to create a table with the simulation results. `rejrate` is the mean rejection rate for `dfuller, drift`, and `bsrejrate` is the mean rejection rate of `bsrwalkdrift`. `llbsrrate` and `ulbsrrate` are the lower and upper limits, respectively, of the 95% confidence intervals for `bsrejrate`.

The output shows that `rejrate` tends to be farther from the nominal rate (0.05) because both the value of `alpha` (the drift term) and the number of observations are smaller. On the other hand, `bsrejrate` is always nicely near the nominal rate. Also, notice that in all the cases, the confidence intervals cover the nominal rate.

```
. tabdisp nobs, by(alpha) cellvar(rejrate bsrejrate llbsrrate ulbsrrate)
```

alpha and nobs	rejrate	bsrejrate	llbsrrate	ulbsrrate
0.05				
25	0.373	0.054	0.044	0.064
50	0.414	0.047	0.037	0.056
100	0.430	0.054	0.044	0.063
200	0.402	0.051	0.041	0.060
500	0.357	0.058	0.048	0.069
0.10				
25	0.380	0.045	0.036	0.054
50	0.357	0.049	0.039	0.058
100	0.360	0.058	0.048	0.069
200	0.325	0.052	0.043	0.062
500	0.226	0.058	0.048	0.069
0.25				
25	0.283	0.047	0.037	0.056
50	0.250	0.052	0.043	0.062
100	0.192	0.056	0.046	0.067
200	0.132	0.054	0.045	0.064
500	0.090	0.047	0.038	0.057
0.50				
25	0.179	0.049	0.040	0.058
50	0.120	0.047	0.038	0.057
100	0.102	0.053	0.043	0.063
200	0.082	0.052	0.043	0.062
500	0.064	0.050	0.040	0.060
0.75				
25	0.120	0.052	0.042	0.061
50	0.090	0.044	0.035	0.053
100	0.078	0.049	0.039	0.058
200	0.062	0.046	0.037	0.055
500	0.068	0.054	0.044	0.064

5 Conclusion

The mean rejection rates from our simulations showed that `bsrwalkdrift` provides a correction for the ADF test that is particularly relevant with small or medium sample sizes and a small or medium drift term. Therefore, this command would be a recommendable alternative to perform unit-root testing for the null hypothesis that the process is a random walk with a drift. Finally, the command has an option to test for random walk appropriate for trending series, and there is another option to test for random walk without drift, which would be appropriate for series with no apparent trend.

6 Acknowledgment

We would like to thank W. Robert Reed of the University of Canterbury in Christchurch, New Zealand, for bringing to our attention via simulations the poor size performance of the conventional ADF test for random walk with drift. His findings motivated us to look for an alternative method and implement it in a Stata command.

7 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 21-1
. net install st0626     (to install program files, if available)
. net get st0626         (to install ancillary files, if available)
```

8 References

Basawa, I. V., A. K. Mallik, W. P. McCormick, J. H. Reeves, and R. L. Taylor. 1991. Bootstrapping unstable first-order autoregressive processes. *Annals of Statistics* 19: 1098–1101. https://doi.org/10.1214/aos/1176348142.

Campbell, J. Y., and P. Perron. 1991. Pitfalls and opportunities: What macroeconomists should know about unit roots. *NBER Macroeconomics Annual 1991* 6: 141–201. https://doi.org/10.1086/654163.

Hamilton, J. D. 1994. *Time Series Analysis*. Princeton, NJ: Princeton University Press.

Hylleberg, S., and G. E. Mizon. 1989. A note on the distribution of the least squares estimator of a random walk with drift. *Economic Letters* 29: 225–230. https://doi.org/10.1016/0165-1765(89)90065-7.

Park, J. Y. 2003. Bootstrap unit root tests. *Econometrica* 71: 1845–1895. https://doi.org/10.1111/1468-0262.00471.

About the authors

Miguel Dorta is a Senior Statistician at StataCorp.

Gustavo Sanchez is the Director of Technical Services at StataCorp.