The \texttt{johansensmall} function package for \textit{gretl}

Sven Schreiber and Andreas Noack Jensen

version 3.2, August 2020

This is a contributed function package to run the Johansen test for cointegration in small samples, supplying Bartlett corrected trace test statistics and bootstrapped p-values. Please send comments or questions to the gretl mailing list at \url{https://gretlml.univpm.it/postorius/lists/gretl-users.gretlml.univpm.it/}.

Andreas did most of the work at first, Sven finished the function interface and resolved some gretl-related details. Later, Sven rewrote the bootstrap, added the UC (unrestricted constant) case, the iid bootstrap, a bootstrap-after-bootstrap (with bias correction) option, and parallel processing.

Distributed under the same license as the gretl program itself (GPL).

1 Usage

The main entry function \texttt{johansensmall()} prints output and returns a bundle with the following members:

- \textit{mresults}: main result matrix, tested ranks $0..n-1$ in rows;
  - 1. 1st col: rank under H0,
  - 2. 2nd col: standard trace stat,
  - 3. 3rd col: standard (asympt.) p-value,
  - 4. 4th col: corrected p-value.
    and only in case of Bartlett:
  - 5. 5th col: corrected trace stat
- \textit{strT1}: obs string, sample start (example: “1980:1”)
- \textit{strT2}: obs string, sample end
- \textit{detspec}: string with deterministic spec (“nc”, “rc”, “uc”, “crt”)
- \textit{nlags}: integer number of lags of the VAR (refers to the levels representation)
- \textit{warnings}: string with misc. messages
  and only in case of bootstrap:
• *breps*: integer number of bootstrap replications

The arguments of this function are (all except the first have default values and can be omitted in principle):

1. list: List of endogenous variables
2. int: Number of lags in the (levels) VAR (default: 2)
4. int: Number of bootstrap replications (zero for Bartlett correction) (default: 0/Bartlett)
5. list: List of unrestricted variables (default: *null*)
6. list: List of restricted exogenous variables (default: *null*)
7. bool: Do wild (yes) or iid (no) bootstrap? (default: yes)
8. bool: Pre-apply a bias correction? (default: no)
9. int: try using parallel processing (MPI); -1: automatic, 0: force MPI off (default), 1: force MPI on

There are also more low-level functions `coint2bartlett` and `coint2boot` (not documented here yet, please see the package source code). For example, in `coint2boot` you can choose to only test a subset of possible rank hypotheses, potentially saving lots of computation time.

### 1.1 Parallel execution (MPI)

Currently parallel processing is possible for bootstrapping but not used by default. If “automatic / -1” is chosen for the last argument (see above) it is used when gretl reports to the package that it detected MPI support\(^1\) and when the chosen number of bootstrap replications is above 1000. The maximum number of used CPU cores (or threads) is set to six in v3.1, and there is currently no way to influence the number of processes used, it is determined by the available hardware. Both limitations may be lifted in the future if no problems arise in practice.

Parallel processing may be explicitly switched off (as in the current default). It may also be switched on manually, for example for testing purposes with less than 1001 replications. Forcing MPI on is discouraged, however, since errors are expected when gretl doesn’t detect MPI support for itself.

In some circumstances the parallel bootstrap running on $N$ cores may be (almost) $N$ times faster than the traditional sequential method. However, actual outcomes are usually less exciting. For example, some CPUs have features like “turbo boost” which means that a single core will run at a much higher clock speed if the other cores are idle – the converse implication is that

\(^1\)For general information about MPI and the way it is supported for gretl/hansl scripting see the manual “gretl + MPI”, available under the help menu of gretl’s main window. In particular, MPI must be installed separately on the machine, and the path to the mpi executable must be correctly specified in gretl’s settings (Tools/Settings/Main/MPI).
in parallel tasks each core will be slower. Another factor is the implicit vectorized parallelization of low-level linear algebra routines in modern CPUs that already takes place in the non-MPI seemingly “sequential” execution. This means that the additional MPI-based parallelization has less value added to offer.

With respect to hyperthreading the package tries to run just as many processes as there are available physical CPU cores.\footnote{2}

The additional overhead of launching MPI and of transferring the necessary data back and forth should also be borne in mind. This (almost) fixed overhead becomes relatively more important for shorter jobs, which is the reason why this package does not attempt parallel execution for up to 1000 replications.

1.2 GUI

The main function of the package is also perfectly usable through the graphical interface. See above for explanations of the arguments/choices, and Figure 1. The returned bundle is offered for inspection via gretl’s mechanism in the result window’s icon bar.

2 Some econometric background

These functions are implementations of: "A Small Sample Correction for the Test of Cointegrating Rank in the Vector Autoregressive Model" by Soren Johansen, Econometrica 2002; and: "Bootstrap Determination of the Co-Integration Rank in Vector Autoregressive Models", by Giuseppe Cavaliere, Anders Rahbek and Robert Taylor, Econometrica 2012; and since v2.0 also of the extension to the unrestricted constant case by Cavaliere/Rahbek/Taylor 2015, J of Time Series Analysis.

If further unrestricted exogenous terms are included, a justification of this implementation is implicitly given by: "A bootstrap algorithm for testing cointegration rank in VAR models in the presence of stationary variables", by Anders Rygh Swensen, J of Econometrics 2011. In particular, what this package does seems to be Swensen’s Algorithm 1 in combination with his Remarks 3 and 6; that is, we always estimate the system under the rank of the null hypothesis (what Swensen calls "restricted", deviating from his step ii), and we only use the actual values of the exogenous terms, not estimating an extra autoregressive model for them (avoiding step i in his Algorithm 1).


The p-values values of the trace test statistic are gamma distribution approximations from "Approximations to the Asymptotic Distributions of Cointegration Tests", by Jurgen Doornik, Journal of Economic Surveys 1998.

\footnote{2On gretl versions prior to 2019c this information is not available, so a different heuristic is used. This workaround may lead to the use of less processes than physical cores if the CPU does not have hyperthreading and has more than two cores.}
Figure 1: Location of the package’s attachment in the gretl main menus, and its GUI dialog window. (Here gretl running on Windows with the AWM18.gdt example dataset in quarterly frequency loaded.)
The wild bootstrap is based on Gaussian random draws for the multiplicative factors.

3 Changelog

- 3.2 (August 2020):
  - hopefully fix the non-working case of running a model with exogenous terms (apart from the standard deterministic cases) in parallel with MPI – it could always be run OK in the standard non-parallel way, however
  - add to coint2boot a “secret” option to do a parametric (Gaussian) bootstrap instead of resampling from the estimated residuals
  - add some screenshots to this documentation
  - some more internal refactoring
- 3.1 (December 2019): MPI: choose number of processes based on $sysinfo.ncores instead of $sysinfo.nproc.
- 3.0 (September 2019):
  - require gretl 2018c (because of new internal syntax,MPI, and lists in bundle)
  - add parallel bootstrap through MPI, sometimes giving a considerable speedup for high numbers of replications
  - internal refactoring because of that
  - switch to this PDF documentation
- 2.7 (Sep 2017): go on when a certain random matrix is only rarely not pos.-def.
- 2.6 (July 2017):
  - remove the option to check roots in every iteration (thanks to Giuseppe Cavaliere for the feedback) -- this changes the position of function arguments (is a backwards-incompatible change)
  - now always use actual initial values (should have an effect mainly in case of no constant)
- 2.5 (June 2017):
  - do the residual recentering earlier, strictly following the papers,
  - fix bug with wrong initial values (mainly for ranks > 0)
  - fix trivial but again fatal data handling bug for the case w/o lagged differences (just one lag) in combination with exogenous terms,
  - forbid 0 lags in coint2boot(),

5
do not erroneously overwrite the signal for explosive roots when bootstrapping the CI test, handle the case of 1 lag and rank 0

- 2.4 (May 2017): fix trivial but fatal syntax bugs in the weird case when original data are explosive

- 2.3 (April 2017): add bias correction option

- 2.2 (April 2017): fix treatment of exogenous terms in the bootstrap

- 2.1 (March 2017, not released): marginal changes to the string labels and to this help text

- 2.0 (February 2017):
  - change the name from coint2finite to johansensmall,
  - completely rewrite the bootstrap function,
  - add UC case from Cavaliere/Rahbek/Taylor 2015 JTSA,
  - change the deterministic spec interface because of that --> incompatible change!
  - add choice of iid bootstrap and add recentering of residuals for bootstrapping,
  - add option to check and discard draws with explosive roots,
  - inter alia fixed a probable bug that the deterministics under the alternative were wrong, (which probably led to a conservative test in many cases).

- 1.1 (June 2014): fix help-text typo and menu location