Package: Westerlund (via r-universe)

May 12, 2026

Type Package

Title Panel Cointegration Tests Based on Westerlund (2007)

Version 0.1.3

Description Implements a functional approximation of the four panel cointegration tests developed by Westerlund (2007) <doi:10.1111/j.1468-0084.2007.00477.x>. The tests are based on structural rather than residual dynamics and allow for heterogeneity in both the long-run cointegrating relationship and the short-run dynamics. The package includes logic for automated lag and lead selection via AIC/BIC, Bartlett kernel long-run variance estimation, and a bootstrap procedure to handle cross-sectional dependence. It also includes a bootstrapping distribution visualization function for diagnostic purposes.

License MIT + file LICENSE

URL https://github.com/bosco-hung/WesterlundTest

BugReports https://github.com/bosco-hung/WesterlundTest/issues

Depends R (>= 4.0.0)

Imports stats, graphics, grDevices, utils, scales, dplyr, ggplot2, tidyr

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

Config/testthat/edition 3

NeedsCompilation no

Author Bosco Hung [aut, cre] (ORCID: <https://orcid.org/0000-0003-3073-303X>)

Maintainer Bosco Hung <bosco.hung@st-annes.ox.ac.uk>

Repository https://cran.r-universe.dev

Date/Publication 2026-04-12 15:00:15 UTC

RemoteUrl https://github.com/cran/Westerlund

RemoteRef HEAD

RemoteSha 2c94a7e0f598c3641312eb3e0b5fa3ce217c3fcc

Contents

`calc_lrvar_bartlett` *Long-Run Variance Estimation with Bartlett Kernel*

Description

Computes a Bartlett-kernel (Newey–West style) long-run variance estimate for a univariate series using Stata-like conventions: missing values are removed (`na.omit`), autocovariances are scaled by $1/n$ (not $1/(n-j)$), and optional centering is controlled by `nodemean`.

Usage

```
calc_lrvar_bartlett(x, maxlag, nodemean = FALSE)
```

Arguments

`x`	A numeric vector. Missing values are removed prior to computation.
`maxlag`	Non-negative integer. Maximum lag order m used in the Bartlett kernel. If `maxlag=0`, the function returns the sample variance estimate γ_0.
`nodemean`	Logical. If `FALSE` (default), the series is centered by subtracting its mean before computing autocovariances. If `TRUE`, the function assumes `x` is already centered and does not de-mean it.

Details

Let x_t be the input series after removing missing values. If `nodemean=FALSE`, the function replaces x_t with $x_t - \bar{x}$.

Define the lag-j autocovariance using the Stata-style scaling:

$$\gamma_j = \frac{1}{n} \sum_{t=j+1}^{n} x_t x_{t-j}, \qquad j = 0, 1, \ldots, m,$$

where n is the length of the cleaned series and $m =$ `maxlag`. Note that the scaling uses $1/n$ for all j (rather than $1/(n-j)$).

The Bartlett kernel weight at lag j is:

$$w_j = 1 - \frac{j}{m+1}.$$

The long-run variance estimate is then:

$$\widehat{\Omega} = \gamma_0 + 2 \sum_{j=1}^{m} w_j \gamma_j.$$

If the cleaned series has zero length, the function returns `NA`.

Value

A single numeric value:

- The Bartlett-kernel long-run variance estimate $\widehat{\Omega}$ (scalar),
- or `NA` if `x` contains no non-missing values.

Technical Notes

This section illustrates how `calc_lrvar_bartlett()` computes a Bartlett-kernel long-run variance (LRV) estimate and how its options map to common time-series preprocessing choices.

Stata-like conventions: This helper function is designed to match Stata-style calculations:

- **Missing values are dropped**: `na.omit(x)` is applied first.
- **Scaling by** $1/n$: autocovariances at all lags divide by n, not by $n - j$.
- **Optional de-meaning**: controlled by `nodemean`.

Centering: By default (`nodemean=FALSE`), the series is centered: $x_t \leftarrow x_t - \bar{x}$. Set `nodemean=TRUE` when you have already centered the series elsewhere.

Choosing maxlag: `maxlag` sets the truncation point m. Larger m captures more serial correlation but increases estimation noise.

See Also

`westerlund_test`

Examples

```
## Example 1: Basic usage
x <- rnorm(200)
calc_lrvar_bartlett(x, maxlag = 4)

## Example 2: maxlag = 0 returns gamma_0
calc_lrvar_bartlett(x, maxlag = 0)

## Example 3: Handle missing values (they are removed)
x_na <- x
x_na[c(5, 10, 50)] <- NA
calc_lrvar_bartlett(x_na, maxlag = 4)

## Example 4: Compare centering choices
## Default: de-mean internally
lr1 <- calc_lrvar_bartlett(x, maxlag = 4, nodemean = FALSE)

## Pre-center and skip de-meaning
x_centered <- x - mean(x)
lr2 <- calc_lrvar_bartlett(x_centered, maxlag = 4, nodemean = TRUE)
```

DisplayWesterlund *Standardize and Display Westerlund ECM Panel Cointegration Test Results*

Description

Formats, standardizes, and prints the Westerlund (2007) ECM-based panel cointegration test results for the null hypothesis of no cointegration. Given raw statistics G_t, G_a, P_t, and P_a, the function computes standardized Z-statistics using tabulated (or hard-coded) asymptotic moments and reports left-tail asymptotic p-values. If a bootstrap distribution is supplied, it also computes bootstrap (robust) p-values for the raw statistics.

Usage

```
DisplayWesterlund(
  stats,
  bootstats = NULL,
  nobs,
  nox,
  constant = FALSE,
  trend = FALSE,
  meanlag = -1,
  meanlead = -1,
  realmeanlag = -1,
  realmeanlead = -1,
  auto = 0,
  westerlund = FALSE,
```

```
  aic = TRUE,
  verbose = FALSE
)
```

Arguments

stats	A numeric vector of length 4 or a 1x4 numeric matrix containing the raw test statistics in the order c(Gt, Ga, Pt, Pa).
bootstats	Optional. A numeric matrix with bootstrap replications of the raw statistics, with 4 columns in the order [Gt, Ga, Pt, Pa]. If provided, robust (bootstrap) p-values are computed and displayed.
nobs	Integer. Number of valid cross-sectional units (series) used in the test.
nox	Integer. Number of covariates in the long-run relationship (length of xvars in upstream calls). Used to select asymptotic moments for standardization in the non-westerlund case.
constant	Logical. Indicates whether a constant was included in the cointegrating relationship. Affects the deterministic-case row used for asymptotic moment lookup.
trend	Logical. Indicates whether a trend was included. If TRUE, constant is assumed to be TRUE and the deterministic-case row for moment lookup changes accordingly.
meanlag	Integer. Mean selected lag length (rounded down to integer) reported in the header when auto is non-zero.
meanlead	Integer. Mean selected lead length (rounded down to integer) reported in the header when auto is non-zero.
realmeanlag	Numeric. Unrounded mean selected lag length (for display when auto is non-zero).
realmeanlead	Numeric. Unrounded mean selected lead length (for display when auto is non-zero).
auto	Logical/integer. If non-zero, the header prints the average AIC-selected lag and lead lengths based on realmeanlag and realmeanlead.
westerlund	Logical. If TRUE, uses the hard-coded Westerlund-specific moment constants for standardization (separate constants for trend vs no-trend). If FALSE, uses lookup tables indexed by deterministic case and number of covariates.
aic	Logical. If TRUE, AIC was used for lag/lead selection; If FALSE, BIC was used.
verbose	Logical. If TRUE, prints additional output.

Details

What this function does. DisplayWesterlund() takes raw Westerlund ECM test statistics and produces:

- standardized Z-statistics for G_t, G_a, P_t, P_a,
- left-tail asymptotic p-values using pnorm(),
- optionally, bootstrap (robust) p-values when bootstats is provided,

- a console table summarizing the results.

Standardization and deterministic cases. The function selects a deterministic-case index:

- Row 1: no constant, no trend,
- Row 2: constant, no trend,
- Row 3: constant and trend,

implemented as `row_idx = (as.integer(constant) + as.integer(trend)) + 1`. In the non-`westerlund` case, asymptotic means and variances are taken from hard-coded lookup matrices indexed by `row_idx` and `nox`. Z-statistics are computed using:

$$Z = \frac{\sqrt{N}S - \sqrt{N}\mu}{\sqrt{\sigma^2}}$$

for mean-group statistics and analogous formulas for pooled statistics as implemented in the code, where $N =$ `nobs` and S is the raw statistic.

In the `westerlund=TRUE` case, the function uses a separate set of hard-coded mean/variance constants, with different values depending on whether `trend` is included.

Asymptotic p-values. All asymptotic p-values are computed as left-tail probabilities: `pnorm(Z)`.

Bootstrap p-values (robust p-values). If `bootstats` is supplied, bootstrap p-values are computed for each raw statistic using a left-tail empirical rule:

$$\hat{p} = \frac{r+1}{B+1}, \qquad r = \sum_{b=1}^{B} \mathbb{I}(S_b \le S_{\text{obs}}),$$

after dropping non-finite bootstrap draws. This is a common finite-sample correction used in Stata-style bootstrap code.

Return values. In addition to printing, the function returns a list containing the raw statistics, Z-statistics, asymptotic p-values, and (if applicable) bootstrap p-values.

Value

A list containing at minimum:

- `gt`, `ga`, `pt`, `pa`: raw test statistics,
- `gt_z`, `ga_z`, `pt_z`, `pa_z`: standardized Z-statistics,
- `gt_pval`, `ga_pval`, `pt_pval`, `pa_pval`: left-tail asymptotic p-values.

If `bootstats` is provided, the list also includes:

- `gt_pvalboot`, `ga_pvalboot`, `pt_pvalboot`, `pa_pvalboot`: bootstrap (robust) p-values for the raw statistics.

Vignette

This section explains how to use `DisplayWesterlund()` and how its output connects to the broader testing workflow.

Where does `DisplayWesterlund()` fit?: Typically, the workflow is:

1. Compute observed raw statistics via `WesterlundPlain`.
2. Optionally compute a bootstrap distribution via `WesterlundBootstrap`.
3. Call `DisplayWesterlund()` to standardize, print, and return p-values.

The user-facing `westerlund_test` wraps these steps and collects the returned scalars.

Input format for `stats`: `stats` can be either a numeric vector `c(Gt, Ga, Pt, Pa)` or a `1x4` matrix with `[1,]` corresponding to `Gt, Ga, Pt, Pa`.

Asymptotic standardization and left-tail p-values: The function converts raw statistics to Z-statistics using hard-coded asymptotic means and variances. P-values are computed as `pnorm(Z)`, i.e., a left-tail test.

Bootstrap (robust) p-values: If `bootstats` is provided, robust p-values are computed by comparing the observed raw statistic to its bootstrap distribution, using a finite-sample correction: $(r + 1)/(B + 1)$ where r is the number of bootstrap draws less than or equal to the observed statistic.

Examples:

```
## Example 1: Asymptotic-only display (no bootstrap)
stats <- c(Gt = -2.1, Ga = -9.5, Pt = -1.8, Pa = -6.2)

res1 <- DisplayWesterlund(
  stats      = stats,
  bootstats = NULL,
  nobs       = 18,
  nox        = 2,
  constant   = TRUE,
  trend      = FALSE,
  auto       = 0,
  westerlund = FALSE
)

## Example 2: With bootstrap distribution (robust p-values)
set.seed(123)
bootstats <- cbind(
  rnorm(399, mean = -1.8, sd = 0.9),
  rnorm(399, mean = -8.0, sd = 3.0),
  rnorm(399, mean = -1.5, sd = 1.0),
  rnorm(399, mean = -5.0, sd = 3.5)
)

res2 <- DisplayWesterlund(
  stats      = stats,
```

```
  bootstats = bootstats,
  nobs       = 18,
  nox        = 2,
  constant   = TRUE,
  trend      = FALSE,
  auto       = 1,
  realmeanlag  = 1.35,
  realmeanlead = 0.40,
  westerlund = FALSE
)
```

References

Westerlund, J. (2007). Testing for error correction in panel data. *Oxford Bulletin of Economics and Statistics*, 69(6), 709–748.

See Also

`westerlund_test`, `WesterlundPlain`, `WesterlundBootstrap`

`get_diff` *First Difference with Strict Time Indexing*

Description

Computes the first difference of a time-indexed series using strict time-based lagging. The function respects gaps in the time index and returns `NA` when the previous time period does not exist, mirroring Stata's `D.` operator.

Usage

```
get_diff(vec, tvec)
```

Arguments

`vec`	A numeric (or atomic) vector of observations.
`tvec`	A vector of time indices corresponding one-to-one with `vec`. Each value must uniquely identify a time period within the series.

Details

This helper function computes first differences as:

$$\Delta x_t = x_t - x_{t-1},$$

where the lagged value x_{t-1} is obtained using `get_lag`, which performs strict time-based lookup.

Internally, the function calls:

```
val_t_minus_1 <- get_lag(vec, tvec, 1)
```

and then subtracts this lagged vector from `vec`. If the time index contains gaps, or if the previous time period does not exist for a given observation, the lagged value is `NA` and the corresponding difference is also `NA`.

No interpolation or implicit shifting is performed; missing time periods propagate as missing differences.

Value

A vector of the same length as `vec`, containing the first differences aligned by the time index. Elements are `NA` when the previous time period does not exist.

Time Indexing Logic

This section explains how `get_diff()` computes first differences and why strict time indexing matters in the presence of gaps.

Relation to Stata's D. operator: The function replicates the behaviour of Stata's first-difference operator `D.x`. When time periods are missing, Stata returns missing values rather than differencing across gaps. Because `get_diff()` relies on `get_lag`, it follows the same rule.

Why not use diff()?: The base R function `diff()` computes differences based on vector positions. This implicitly assumes a complete and regularly spaced time index. When time periods are missing, `diff()` can produce misleading results by differencing across gaps. `get_diff()` avoids this by differencing only when the previous time period exists.

See Also

`get_lag`, `get_ts_val`, `westerlund_test`

Examples

```
## Example 1: Regular time series
t <- 1:5
x <- c(10, 20, 30, 40, 50)

get_diff(x, t)
# [1] NA 10 10 10 10

## Example 2: Time series with a gap
t_gap <- c(1, 2, 4, 5)
x_gap <- c(10, 20, 40, 50)

get_diff(x_gap, t_gap)
# [1] NA 10 NA 10

## Explanation:
## At t = 4, the previous period t-1 = 3 does not exist, so the difference is NA.

## Example 3: Comparison with diff()
```

```
diff(x_gap)
# [1] 10 20 10
```

---

get_lag *Time-Indexed Lag Extraction with Explicit Gaps*

Description

Extracts lagged (or led) values from a time-indexed vector using a strict time-based lookup. The function respects gaps in the time index and returns NA when the requested lagged time does not exist, mirroring the behaviour of Stata's time-series lag/lead operators.

Usage

```
get_lag(vec, tvec, k)
```

Arguments

vec	A numeric (or atomic) vector of observations.
tvec	A vector of time indices corresponding one-to-one with vec. Each value must uniquely identify a time period within the series.
k	Integer. The lag order. Positive values correspond to lags (e.g., k = 1 for $t-1$), while negative values correspond to leads (e.g., k = -1 for $t+1$).

Details

This helper function performs strict time-based lagging rather than position-based shifting. Internally, it constructs a mapping from time indices to observed values:

```
val_map <- setNames(vec, tvec)
```

For each observation at time t, the function retrieves the value associated with time $t-k$. If that time value is not present in tvec, the result is NA.

This behaviour is particularly important when working with irregular time series or panel data with gaps. In such cases, simple vector shifting can incorrectly carry values across missing time periods, while get_lag() preserves the correct alignment.

No interpolation, padding, or reordering is performed; missing time periods propagate as NA in the lagged (or led) series.

Value

A vector of the same length as vec, containing the lagged (or led) values aligned by the time index. Elements are NA when the requested time period does not exist.

Technical Background

This section illustrates the logic of `get_lag()` and explains how it differs from standard position-based lagging.

Why not simple shifting?: In many applications, lags are computed by shifting vectors by positions. This implicitly assumes a complete and regular time index. When time periods are missing, such shifting produces incorrect lag values. `get_lag()` avoids this by explicitly matching on time values.

Relation to Stata operators: The function mirrors the behaviour of Stata's time-series operators:

- `L.x`: lagged value at $t - 1$,
- `L2.x`: lagged value at $t - 2$,
- `F.x`: lead value at $t + 1$.

As in Stata, gaps in the time index result in missing values in the lagged series.

See Also

`get_ts_val`, `westerlund_test`, `calc_lrvar_bartlett`

Examples

```
## Example 1: Regular time series
t <- 1:5
x <- c(10, 20, 30, 40, 50)

get_lag(x, t, k = 1)
# [1] NA 10 20 30 40

get_lag(x, t, k = -1)
# [1] 20 30 40 50 NA

## Example 2: Time series with a gap
t_gap <- c(1, 2, 4, 5)
x_gap <- c(10, 20, 40, 50)

get_lag(x_gap, t_gap, k = 1)
# [1] NA 10 NA 40

## Explanation:
## At t = 4, the requested time t-1 = 3 does not exist, so NA is returned.

## Example 3: Higher-order lags
get_lag(x_gap, t_gap, k = 2)
# [1] NA NA NA 20
```

get_ts_val *Strict Time-Series Mapping with Explicit Gaps*

Description

Retrieves lagged (or led) values from a time-indexed vector using a strict time-based mapping.

Usage

```
get_ts_val(vec, tvec, lag)
```

Arguments

vec	A numeric (or atomic) vector of observations.
tvec	A vector of time indices corresponding one-to-one with vec. Values must uniquely identify time periods within the series.
lag	Integer. The lag order. Positive values correspond to lags (e.g., lag = 1 for $t-1$), while negative values correspond to leads (e.g., lag = -1 for $t+1$).

Details

This helper function implements a strict time-based lookup rather than position-based indexing. Internally, it constructs a named mapping from time indices to observed values:

```
val_map <- setNames(vec, tvec)
```

For each observation at time t, the function computes the target time $t - \text{lag}$ and retrieves the corresponding value from the map.

If the target time does not exist in tvec, the function returns NA for that observation. This behaviour exactly mirrors Stata's lag/lead operators in the presence of gaps.

Importantly, the function does **not** interpolate or shift values when time periods are missing. A gap in the time index propagates as NA in the lagged (or led) series.

Value

A vector of the same length as vec, containing the lagged (or led) values aligned by time index. Elements are NA when the requested time period does not exist.

Implementation Details

This section explains why get_ts_val() is useful and how it differs from standard lagging approaches based on vector positions.

Why strict time-based lagging matters: In panel and time-series econometrics, lagged variables should be defined with respect to the time index, not the row position. When data contain gaps in time, simple shifting (e.g., c(NA, x[-length(x)])) can produce incorrect values. get_ts_val() avoids this by explicitly matching on time values.

Relation to Stata operators: This function replicates the behaviour of Stata's time-series operators:

- `L.x`: lagged value at $t-1$,
- `L2.x`: lagged value at $t-2$,
- `F.x`: lead value at $t+1$.

When time periods are missing, Stata returns missing values rather than shifting across gaps.

See Also

`westerlund_test`, `calc_lrvar_bartlett`

Examples

```
## Example 1: Regular time series
t <- 1:5
x <- c(10, 20, 30, 40, 50)

## Lag by one period
get_ts_val(x, t, lag = 1)
# [1] NA 10 20 30 40

## Lead by one period
get_ts_val(x, t, lag = -1)
# [1] 20 30 40 50 NA

## Example 2: Time series with a gap
t_gap <- c(1, 2, 4, 5)
x_gap <- c(10, 20, 40, 50)

## Lag by one period: note the NA at time 4
get_ts_val(x_gap, t_gap, lag = 1)
# [1] NA 10 NA 40

## Explanation:
## - At t = 4, t - 1 = 3 does not exist in t_gap, so NA is returned.

## Example 3: Higher-order lags
get_ts_val(x_gap, t_gap, lag = 2)
# [1] NA NA NA 20
```

---

plot.westerlund_test *Plot Bootstrap Distributions for Westerlund ECM Panel Cointegration Tests*

Description

This function provides an S3 method for visualizing objects of class `westerlund_test`. It creates a faceted `ggplot2` visualization of the bootstrap distributions for the four Westerlund (2007) panel

cointegration statistics: G_t, G_a, P_t, and P_a. The plot displays the kernel density of bootstrap replications, the observed test statistic as a solid line, and the left-tail bootstrap critical value as a dashed line.

Usage

```
## S3 method for class 'westerlund_test'
plot(
  x,
  title = "Westerlund Test: Bootstrap Distributions",
  conf_level = 0.05,
  colors = list(
    obs = "#D55E00",
    crit = "#0072B2",
    fill = "grey80",
    density = "grey30"
  ),
  lwd = list(obs = 1, crit = 0.8, density = 0.5),
  show_grid = TRUE,
  ...
)
```

Arguments

x	An object of class `westerlund_test`. This list-like object must include `x$bootstrap_distributions` (a numeric matrix with 4 columns for `Gt`, `Ga`, `Pt`, `Pa`) and `x$test_stats` (a named list or vector containing the observed statistics).
title	Character. The main title of the plot.
conf_level	Numeric in (0,1). Left-tail quantile used as the critical value. For example, `conf_level = 0.05` identifies the 5% left-tail critical value.
colors	A named list of colors for plot elements. Expected elements include `obs` for the observed line, `crit` for the critical value line, `fill` for the density area fill, and `density` for the density curve outline.
lwd	A named list of line widths or sizes for the `obs`, `crit`, and `density` elements.
show_grid	Logical. If `TRUE`, displays major panel grids.
...	Additional arguments passed to or from other methods.

Details

The function converts the bootstrap distribution matrix into a long-format data frame to utilize the faceting capabilities of the **ggplot2** package. This implementation uses `ggplot2::geom_density()` for empirical distribution estimation and arranges the four subplots into a 2x2 grid. By using `facet_wrap(~Statistic, scales = "free")`, the plots maintain independent x-axis scales to better visualize the specific range of each statistic. Text annotations are automatically added to each facet to display the exact numerical values of the Observed Stat and the Critical Value for quick reference.

The dashed line represents the bootstrap critical value for a left-tailed test. If the observed statistic, represented by the solid line, is located to the left of the dashed line, it indicates a rejection of the null hypothesis of no cointegration at the specified `conf_level` significance level. This visual approach allows for an immediate assessment of the test results relative to the empirically derived distribution.

The function requires the **ggplot2**, **tidyr**, and **scales** packages to be installed. Because it is an S3 method, it is typically invoked by calling `plot()` directly on the result object of a Westerlund test.

Value

Returns a `ggplot` object. This allows the user to apply further customizations, such as changing themes with `+ theme_bw()` or adding additional layers and labels.

Inference and Visualization

Why use bootstrap distributions?: Westerlund's pooled statistics can be sensitive to nuisance parameters and cross-sectional dependence in finite samples. Visualizing the bootstrap density provides a more robust reference than asymptotic normal curves, offering a clearer picture of the distribution under the null hypothesis.

Customization: Because the function returns a `ggplot` object, users can modify the output easily. For example, the theme can be changed by adding a theme layer, such as `plot(res) + ggplot2::theme_dark()`.

References

Westerlund, J. (2007). Testing for error correction in panel data. *Oxford Bulletin of Economics and Statistics*, 69(6), 709–748.

See Also

`westerlund_test`, `WesterlundBootstrap`, `DisplayWesterlund`

Examples

```
## Example: generating a mock westerlund_test object
set.seed(123)
mock_res <- list(
  bootstrap_distributions = cbind(
    Gt = rnorm(399, mean = -2, sd = 1),
    Ga = rnorm(399, mean = -8, sd = 3),
    Pt = rnorm(399, mean = -2, sd = 1),
    Pa = rnorm(399, mean = -6, sd = 3)
  ),
  test_stats = list(Gt = -3.1, Ga = -10.2, Pt = -2.7, Pa = -7.8)
)
class(mock_res) <- "westerlund_test"

## Use the S3 plot method
p <- plot(mock_res)
```

```
## Display plot
if (interactive()) print(p)
```

`print.westerlund_test` *Print Method for Westerlund ECM Panel Cointegration Tests*

Description

This function provides a clean and concise console output for objects of class `westerlund_test`. Instead of displaying the raw list structure, it summarizes the key results of the Westerlund (2007) panel cointegration test, including the four observed test statistics and information regarding the bootstrap replications if they were performed.

Usage

```
## S3 method for class 'westerlund_test'
print(x, ...)
```

Arguments

`x`	An object of class `westerlund_test`. This is typically the output returned by a call to `westerlund_test` or a manually constructed list assigned that class.
`...`	Further arguments passed to or from other methods. Currently unused but maintained for S3 generic compatibility.

Details

The print method is designed to enhance readability by presenting the test results in a formatted table-like structure. It first identifies the test type and then displays the observed values for the group-mean statistics (G_t and G_a) and the pooled panel statistics (P_t and P_a). This summary provides an immediate overview of the test's findings without requiring the user to navigate the internal list structure of the object.

If the test was conducted with a bootstrap distribution, the method also reports the total number of successful replications. This is particularly useful for verifying that the requested number of bootstrap draws was attained after accounting for any non-finite results. Finally, the output includes a brief instructional note suggesting the use of the `plot()` method for visual inference, which is the recommended way to interpret the bootstrap results relative to the observed statistics.

Value

The function returns the input object `x` invisibly (via `invisible(x)`). Its primary purpose is the side effect of printing a formatted summary to the R console.

References

Westerlund, J. (2007). Testing for error correction in panel data. *Oxford Bulletin of Economics and Statistics*, 69(6), 709–748.

See Also

westerlund_test, plot.westerlund_test

Examples

```
## Example: generating a mock westerlund_test object
set.seed(123)
mock_res <- list(
  test_stats = list(Gt = -3.14, Ga = -10.22, Pt = -2.71, Pa = -7.89),
  bootstrap_distributions = matrix(rnorm(400), ncol = 4)
)
class(mock_res) <- "westerlund_test"

## Calling the object directly invokes print.westerlund_test
mock_res

## Or call it explicitly
print(mock_res)
```

---

shiftNA *NA-Padded Lag and Lead Operator*

Description

Shifts a vector forward or backward by a specified number of positions and fills out-of-range values with NA. This helper is designed for position-based transformations where missing values should be explicitly propagated as NA rather than assumed to be zero.

Usage

```
shiftNA(v, k)
```

Arguments

v	A vector (numeric, character, etc.).
k	Integer. Shift order. Positive values correspond to lags (shifting the series downward, i.e.\ $t - k$), while negative values correspond to leads (shifting the series upward, i.e.\ $t + k$).

Details

This function performs a position-based shift of the input vector v. Unlike strict time-indexed helpers (such as get_lag), shiftNA() does not rely on an explicit time index and does not propagate gaps based on timestamps. Instead, it performs a simple index shift, padding the resulting empty slots with NA.

Let n denote the length of v. The behaviour is:

- k = 0: return v unchanged,

- k > 0: lag by k positions; the first k elements are NA,
- k < 0: lead by $|k|$ positions; the last $|k|$ elements are NA.

Formally, for k > 0,

$$\text{out}_t = \begin{cases} \text{NA}, & t \le k, \\ v_{t-k}, & t > k, \end{cases}$$

and for k < 0,

$$\text{out}_t = \begin{cases} v_{t+|k|}, & t \le n - |k|, \\ \text{NA}, & t > n - |k|. \end{cases}$$

Value

A vector of the same length as v, containing the shifted values with NA inserted where the shift would otherwise go out of range.

Usage and Propagation

This section explains the implications of using NA padding in recursive or difference-based calculations.

Why NA-padding?: Using NA is the standard behavior in R for out-of-bounds operations. It ensures that subsequent calculations (like v - shiftNA(v, 1)) correctly result in NA for boundary cases, preventing the accidental use of arbitrary values (like zero) in statistical estimations unless explicitly intended.

Difference from get_lag(): Unlike get_lag, shiftNA() is position-based and ignores time indices. Use shiftNA() when you need a fast, simple shift on vectors that are already correctly ordered.

See Also

WesterlundBootstrap, get_lag, get_diff

Examples

```
v <- c(1, 2, 3, 4, 5)

## Lag by one (k = 1)
shiftNA(v, 1)
# [1] NA 1 2 3 4

## Lead by one (k = -1)
shiftNA(v, -1)
# [1] 2 3 4 5 NA

## Larger shifts
shiftNA(v, 2)
# [1] NA NA 1 2 3

shiftNA(v, -2)
# [1] 3 4 5 NA NA
```

summary.westerlund_test *Summary Method for Westerlund ECM Panel Cointegration Tests*

Description

Provides a detailed summary of the Westerlund (2007) panel cointegration test results. This method extends the basic `print` output by including standardized Z-scores, asymptotic p-values, bootstrap p-values (if applicable), and the Mean Group (MG) parameter estimates.

Usage

```
## S3 method for class 'westerlund_test'
summary(object, ...)

## S3 method for class 'summary.westerlund_test'
print(x, ...)
```

Arguments

`object`	An object of class `westerlund_test`.
`x`	An object of class `summary.westerlund_test`.
`...`	Additional arguments passed to or from other methods.

Details

The `summary` method acts as a bridge between the raw test results and a human-readable report. While the standard `print` method focuses on the raw statistics, `summary` organizes the data into tables that compare the observed statistics against their standardized counterparts and associated p-values.

Specifically, the output includes a summary of the model settings (such as the inclusion of constants or trends), the main test statistics table, and the Mean Group results which highlight the average error correction speed (α) and long-run relationships (β). If a bootstrap was performed, the bootstrap p-values are appended to the main statistics table for direct comparison with asymptotic values.

Value

The `summary` method returns a list of class `summary.westerlund_test` containing:

- `stats_table`: A data frame combining raw stats, Z-scores, and p-values.
- `settings`: A list of the parameters used in the test.
- `mg_results`: A data frame of the Mean Group estimation results.
- `n_units`: The number of cross-sectional units in the panel.

See Also

`westerlund_test`, `plot.westerlund_test`

Examples

```
## Assuming 'res' is the output from westerlund_test()
# res <- westerlund_test(data, ...)
# summary(res)
```

`westerlund_test` *Westerlund ECM-Based Panel Cointegration Test*

Description

Implements the error-correction-based panel cointegration tests proposed by Westerlund (2007). The function computes both mean-group (G_t, G_a) and pooled (P_t, P_a) statistics based on unit-specific ECMs, with Stata-like input validation, time-continuity checks, and optional bootstrap p-values for the raw statistics.

Usage

```
westerlund_test(
  data,
  yvar,
  xvars,
  idvar,
  timevar,
  constant = FALSE,
  trend = FALSE,
  lags,
  leads = NULL,
  westerlund = FALSE,
  aic = TRUE,
  bootstrap = -1,
  indiv.ecm = FALSE,
  lrwindow = 2,
  verbose = FALSE
)
```

Arguments

`data`	A `data.frame` containing the panel data. Panels may be unbalanced, but each unit's time index must be continuous (no holes) after removing missing observations on model variables.
`yvar`	String. Name of the dependent variable.
`xvars`	Character vector. Names of regressors entering the long-run relationship. A maximum of 6 regressors is allowed. If `westerlund=TRUE`, at most one regressor may be included.
`idvar`	String. Column identifying cross-sectional units. Missing values are not allowed.

timevar	String. Column identifying time. Within each unit, the time index must be strictly increasing and continuous (differences of 1).
constant	Logical. Include an intercept in the cointegrating relationship.
trend	Logical. Include a linear trend in the cointegrating relationship. Setting trend=TRUE requires constant=TRUE.
lags	Integer or length-2 integer vector. Fixed lag order or range c(min,max) for the short-run lag length p. This option must be provided.
leads	Integer or length-2 integer vector, or NULL. Fixed lead order or range c(min,max) for the short-run lead length q. If NULL, defaults to 0.
westerlund	Logical. If TRUE, enforces additional restrictions: at least a constant must be included (constant and/or trend) and at most one regressor may be specified.
aic	Logical. If TRUE, uses AIC for lag/lead selection when ranges. If FALSE, uses BIC.
bootstrap	Integer. If bootstrap > 0, runs an internal bootstrap routine with bootstrap replications and returns bootstrap p-values for the raw statistics. If bootstrap <= 0 (default -1), no bootstrap is performed.
indiv.ecm	Logical. If TRUE, gets output of individual ECM regressions.
lrwindow	Integer. Window parameter used for long-run variance estimation in internal routines.
verbose	Logical. If TRUE, prints additional information.

Details

The test is based on estimating unit-specific error-correction models (ECMs) and testing the null hypothesis of no error correction for all cross-sectional units. Four statistics are reported:

- G_t: mean of individual t-ratios of the error-correction coefficient,
- G_a: mean of individually scaled error-correction coefficients,
- P_t: pooled t-type statistic based on a common error-correction coefficient,
- P_a: pooled statistic based on the scaled common coefficient.

Input validation and data checks. The function enforces several Stata-like guardrails:

- A maximum of 6 regressors can be specified in xvars.
- If trend=TRUE, then constant=TRUE is required.
- If westerlund=TRUE, at least a constant must be included and xvars must contain at most one regressor.
- The panel must be declared by specifying idvar and timevar, and idvar may not contain missing values.
- Rows with missing values are excluded, and continuity checks are applied to the resulting usable sample.

Inference. If bootstrap <= 0, the function returns standardized Z-scores and asymptotic (left-tail) p-values. If bootstrap > 0, it additionally produces bootstrap p-values for the raw statistics.

Value

An object of class `westerlund_test`. This is a list containing the following components:

- `test_stats`: A numeric vector containing:
  - `gt`, `ga`, `pt`, `pa`: Raw test statistics.
  - `gt_z`, `ga_z`, `pt_z`, `pa_z`: Standardized Z-scores.
  - `gt_pval`, `ga_pval`, `pt_pval`, `pa_pval`: Asymptotic left-tail p-values.
- `boot_pvals`: A list containing `gt_pvalboot`, `ga_pvalboot`, `pt_pvalboot`, `pa_pvalboot` (only populated if `bootstrap` > `0`).
- `bootstrap_distributions`: A matrix of the bootstrap replicates for each statistic.
- `unit_data`: A `data.frame` with unit-specific ECM results (alpha and beta estimates). `indiv_data`: A list of length equal to the number of cross-sectional units storing unit-specific results.
- `mean_group`: A list containing the Mean Group estimates (`mg_alpha`, `mg_betas`) and their respective standard errors.
- `settings`: A list of the lag/lead specifications and internal parameters used.
- `mg_results`: A `data.frame` summarizing the Mean Group estimation results.

Vignette

This section demonstrates how to use `westerlund_test()` and interprets common outputs.

What the test does: The test evaluates the null hypothesis of **no error correction for all cross-sectional units**. Rejection suggests that at least some units exhibit error-correcting behaviour.

Data requirements: unbalanced panels and time continuity: Unbalanced panels are permitted as long as each unit's time index is continuous after removing rows with missing values. If any unit has gaps (e.g., time jumps from 2001 to 2003), the function stops and reports the unit identifier.

Examples:

```
## Plain (asymptotic) test
res_plain <- westerlund_test(
  data       = df,
  yvar       = "y",
  xvars      = c("x1","x2"),
  idvar      = "id",
  timevar    = "t",
  constant   = TRUE,
  trend      = FALSE,
  lags       = 1,
  leads      = 0,
  westerlund = FALSE,
  bootstrap  = -1,
  indiv.ecm  = FALSE,
  lrwindow   = 2
)
```

Interpreting common errors:

- **Continuous time-series are required**: at least one unit has holes in the time index. Fix by ensuring the usable sample is continuous.
- **At least ... observations are required**: units are too short for the requested lag/lead orders.
- **If a trend is included, a constant must be included**: set `constant=TRUE` when `trend=TRUE`.

References

Westerlund, J. (2007). Testing for error correction in panel data. *Oxford Bulletin of Economics and Statistics*, 69(6), 709–748.

See Also

`WesterlundPlain`, `WesterlundBootstrap`, `DisplayWesterlund`

`westerlund_test_mg` *Print Mean-Group ECM Output and Long-Run Relationship Tables*

Description

Internal reporting helper that prints two Stata-style coefficient tables: (1) the mean-group error-correction model (short-run ECM output) and (2) the estimated long-run relationship with short-run adjustment. This function delegates all table computation and formatting to `westerlund_test_reg`.

Usage

```
westerlund_test_mg(b, V, b2, V2, auto, verbose = FALSE)
```

Arguments

`b`	A numeric vector of coefficients for the long-run relationship and short-run adjustment terms (as reported in the second table).
`V`	A numeric variance-covariance matrix corresponding to `b`.
`b2`	A numeric vector of coefficients for the mean-group error-correction model (as reported in the first table).
`V2`	A numeric variance-covariance matrix corresponding to `b2`.
`auto`	Logical/integer. If non-zero, prints a note indicating that short-run coefficients (apart from the error-correction term) are omitted because selected lag/lead lengths may differ across units.
`verbose`	Logical. If `TRUE`, prints additional output.

Details

What it prints. The function prints two sections:

1. **Mean-group error-correction model:** This corresponds to reporting the mean-group ECM coefficients (often denoted `b2` with variance `V2`). If `auto` is non-zero, a note is printed to remind that short-run coefficients may be omitted due to unit-specific lag/lead selection.
2. **Estimated long-run relationship and short run adjustment:** This corresponds to reporting the long-run relationship and the adjustment dynamics (`b` with variance `V`).

In each section, `westerlund_test_reg` is called to compute standard errors, z-statistics, two-sided p-values, and 95% confidence intervals, and to print the formatted coefficient table using `stats::printCoefmat()`.

Intended use. This is an internal display helper used in Stata-aligned reporting flows. It is not required for computing the Westerlund test statistics themselves.

Value

A named `list` containing two matrices:

`mg_model`	A numeric matrix containing the Mean-group ECM results (coefficients, SE, z, p-values, and CI).
`long_run`	A numeric matrix containing the long-run relationship and adjustment results.

Vignette

This section shows the expected calling pattern for `westerlund_test_mg()` and explains the role of the `auto` flag.

Two-table reporting: In mean-group implementations, it is common to show:

- a short-run ECM table (possibly with omitted coefficients when lag/lead lengths differ by unit),
- a long-run relationship table with the short-run adjustment term.

`westerlund_test_mg()` prints these tables sequentially.

Examples:

```
## Long-run relationship table inputs
b <- c(alpha = -0.20, beta = 1.05)
V <- diag(c(0.03^2, 0.10^2))
rownames(V) <- colnames(V) <- names(b)

## Mean-group ECM table inputs
b2 <- c(ec = -0.18)
V2 <- matrix(0.04^2, nrow = 1, ncol = 1)
rownames(V2) <- colnames(V2) <- names(b2)

## Print both sections (auto = TRUE prints the omission note)
westerlund_test_mg(b, V, b2, V2, auto = TRUE, verbose = FALSE)
```

See Also

westerlund_test_reg, westerlund_test

`westerlund_test_reg` *Formatted Coefficient Table for Westerlund Test Reporting*

Description

Internal helper that calculates standard errors, z-statistics, p-values, and confidence intervals for a given coefficient vector and covariance matrix, then prints a Stata-style regression table.

Usage

```
westerlund_test_reg(b, V, verbose = FALSE)
```

Arguments

`b`	A named numeric vector of coefficients.
`V`	A numeric variance-covariance matrix corresponding to `b`.
`verbose`	Logical. If `TRUE`, prints additional output.

Details

Calculation Logic. The function replicates the behavior of Stata's `ereturn post` when no degrees of freedom are specified, using the standard normal distribution (Z) for all inference:

- **Standard Errors:** Derived as the square root of the diagonal of `V`.
- **z-statistics:** Calculated as $z = \hat{\beta}/SE(\hat{\beta})$.
- **P-values:** Two-tailed probabilities from the standard normal distribution.
- **Confidence Intervals:** Calculated as $\hat{\beta} \pm 1.96 \times SE(\hat{\beta})$.

Formatting. The table is printed using `stats::printCoefmat()` to ensure clean alignment and decimal consistency. It includes columns for the Coefficient, Standard Error, z-statistic, P-value, and the 95% Confidence Interval.

Intended use. This is an internal utility called by `westerlund_test_mg`. It provides a standardized way to report results across different parts of the Westerlund cointegration test output.

Value

This function returns a numeric matrix with rows corresponding to the coefficients in `b` and columns for "Coef.", "Std. Err.", "z", "P>|z|", and the 95% confidence interval bounds.

Reporting Style

This section describes the alignment with econometric software reporting standards.

Stata Compatibility: The output format (column headers and statistical assumptions) is designed to match the output seen in Stata's `xtcpmg` or `xtwest` routines. This ensures that users transitioning from or comparing results with Stata find the output familiar.

See Also

`westerlund_test_mg`, `westerlund_test`

Examples

```
## Define simple coefficient vector and VCV
b <- c(alpha = -0.20, beta = 1.05)
V <- diag(c(0.03^2, 0.10^2))
names(b) <- rownames(V) <- colnames(V) <- c("alpha", "beta")

## Print the formatted table
westerlund_test_reg(b, V, verbose = TRUE)
```

WesterlundBootstrap *Bootstrap Routine for Westerlund ECM Panel Cointegration Tests*

Description

Internal bootstrap routine for generating a bootstrap distribution of the Westerlund (2007) ECM-based panel cointegration statistics under the null hypothesis of no error correction. The routine estimates short-run dynamics for each unit, constructs residual-based bootstrap samples in a Stata-aligned manner, and re-computes G_t, G_a, P_t, and P_a by calling `WesterlundPlain` on each bootstrap sample.

Usage

```
WesterlundBootstrap(
  data,
  touse,
  idvar,
  timevar,
  yvar,
  xvars,
  constant = FALSE,
  trend = FALSE,
  lags,
  leads = NULL,
  westerlund = FALSE,
  aic = TRUE,
  bootstrap = 100,
```

```
  indiv.ecm = FALSE,
  lrwindow = 2,
  verbose = FALSE
)
```

Arguments

data	A data.frame containing the original panel data.
touse	Logical vector of length nrow(data) indicating rows eligible for estimation. The routine further removes rows with missing yvar or xvars.
idvar	String. Column identifying cross-sectional units.
timevar	String. Column identifying time. Within-unit time gaps are handled by Stata-style differencing in the bootstrap setup.
yvar	String. Name of the dependent variable (levels) in the original data.
xvars	Character vector. Names of regressors (levels) in the original data.
constant	Logical. Include a constant term in the short-run regression used to obtain bootstrap residuals and coefficients.
trend	Logical. Include a trend term in the short-run regression used to obtain bootstrap residuals and coefficients.
lags	Integer or length-2 integer vector. Fixed lag order or range c(min,max) used for selecting short-run dynamics in the bootstrap setup.
leads	Integer or length-2 integer vector, or NULL. Fixed lead order or range c(min,max) used for selecting short-run dynamics. If NULL, defaults to 0.
westerlund	Logical. If TRUE, uses a Westerlund-style information criterion and trimming logic in the bootstrap setup, matching the westerlund mode used elsewhere.
aic	Logical. If TRUE, uses AIC for lag/lead selection when ranges. If FALSE, uses BIC.
bootstrap	Integer. Number of bootstrap replications.
indiv.ecm	Logical. If TRUE, gets output of individual ECM regressions..
lrwindow	Integer. Bartlett kernel window (maximum lag) forwarded to WesterlundPlain during bootstrap re-estimation.
verbose	Logical. If TRUE, prints additional output.

Details

Purpose and status. WesterlundBootstrap() is an internal engine typically called by westerlund_test when bootstrap inference is requested. It returns a matrix of bootstrap test statistics that can be used to compute bootstrap p-values for the *raw* statistics.

High-level algorithm. For each bootstrap replication, the routine:

1. Filters the original data using touse and removes missing yvar/xvars.
2. For each unit, estimates a short-run model for Δy_t using Stata-style differencing that respects time gaps: if t is not consecutive, the corresponding difference is set to missing.

3. Stores the estimated coefficients for lagged Δy_t and leads/lags/current Δx_t, and constructs residuals e.
4. Demeans residuals e within each unit and constructs within-unit centered differenced regressors on the residual-support rows.
5. Resamples residual-support clusters to form bootstrap innovations, recursively generates bootstrap Δy_t with an AR recursion implied by the estimated Δy coefficients, integrates to levels to obtain bootstrap `booty` and `bootx` series, and enforces Stata-style truncation rules.
6. Calls `WesterlundPlain` on the constructed bootstrap panel to compute G_t, G_a, P_t, and P_a.

The resulting statistics are stored row-by-row in `BOOTSTATS`.

Time indexing in bootstrap setup. Unlike the main estimation routines (which use strict time-indexed helpers), this bootstrap routine uses a local differencing helper `diff_ts()` that mimics Stata's `D.` operator under gaps: differences are marked `NA` when `diff(timevec) != 1`.

Lag/lead selection in the bootstrap setup. If `lags` and/or `leads` are supplied as ranges, the routine performs an information-criterion search over candidate short-run specifications for each unit. In `westerlund=TRUE` mode, a Westerlund-style criterion is used; otherwise, a standard AIC-based criterion is applied.

Random number generation. The routine sets the RNG kind to Mersenne-Twister and uses a deterministic sequence given the current RNG state. Users should set `set.seed()` upstream if reproducibility is desired.

Value

A list containing:

- `BOOTSTATS`: a numeric matrix with `bootstrap` rows and 4 columns, containing bootstrap replications of G_t, G_a, P_t, and P_a (in that order).

Vignette

This section describes how `WesterlundBootstrap()` constructs bootstrap samples and how it is typically used in the overall testing workflow.

How does this relate to `westerlund_test()`?: When the user-facing `westerlund_test` is called with `bootstrap > 0`, it typically:

1. obtains a bootstrap distribution via `WesterlundBootstrap()`,
2. computes the observed statistics via `WesterlundPlain`,
3. derives bootstrap p-values by comparing observed raw statistics to the bootstrap distribution.

Residual-based bootstrap under the null: The routine first estimates a short-run model for Δy_t within each unit and extracts residuals. It then resamples these residuals (clustered on the residual-support rows) and uses the stored short-run coefficients to propagate bootstrap dynamics.

Handling time gaps: To mirror Stata semantics in differencing, the bootstrap setup uses a local difference rule: if two adjacent observations are not exactly one time unit apart, the difference is set to missing for that location.

Example:

```
## Upstream: set a seed for reproducibility
set.seed(123)

boot_res <- WesterlundBootstrap(
  data       = df,
  touse      = touse,
  idvar      = "id",
  timevar    = "t",
  yvar       = "y",
  xvars      = c("x1","x2"),
  constant   = TRUE,
  trend      = FALSE,
  lags       = 1,
  leads      = 0,
  westerlund = FALSE,
  bootstrap  = 399,
  indiv.ecm  = TRUE,
  lrwindow   = 2,
  verbose    = FALSE
)

## Example left-tail p-value for Gt (assuming 'obs' exists)
# mean(boot_res$BOOTSTATS[,1] <= obs$Gt, na.rm = TRUE)
```

References

Westerlund, J. (2007). Testing for error correction in panel data. *Oxford Bulletin of Economics and Statistics*, 69(6), 709–748.

See Also

westerlund_test, WesterlundPlain

WesterlundPlain *Compute Raw Westerlund ECM Panel Cointegration Statistics (Plain Routine)*

Description

Internal plain (non-bootstrap) routine for computing the four Westerlund (2007) ECM-based panel cointegration test statistics G_t, G_a, P_t, and P_a. The function estimates unit-specific ECM regressions to form the mean-group statistics and then constructs pooled (panel) statistics using cross-unit aggregation and partialling-out steps. Time indexing is handled strictly via gap-aware lag/difference helpers.

Usage

```
WesterlundPlain(
  data,
  touse,
  idvar,
  timevar,
  yvar,
  xvars,
  constant = FALSE,
  trend = FALSE,
  lags,
  leads = NULL,
  lrwindow = 2,
  westerlund = FALSE,
  aic = TRUE,
  bootno = FALSE,
  indiv.ecm = FALSE,
  verbose = FALSE
)
```

Arguments

data	A data.frame containing panel data.
touse	Logical vector of length nrow(data) indicating rows eligible for estimation. Rows are further filtered to remove missing yvar and xvars.
idvar	String. Column identifying cross-sectional units.
timevar	String. Column identifying time.
yvar	String. Name of the dependent variable (levels).
xvars	Character vector. Names of regressors in the long-run relationship (levels).
constant	Logical. If TRUE, includes a constant term in the ECM design matrix.
trend	Logical. If TRUE, includes a linear time trend in the ECM design matrix.
lags	Integer or length-2 integer vector. Fixed lag order or range c(min,max) for short-run dynamics. If a range is supplied, the routine performs an information-criterion search over candidate lag/lead combinations.
leads	Integer or length-2 integer vector, or NULL. Fixed lead order or range c(min,max). If NULL, defaults to 0.
lrwindow	Integer. Bartlett kernel window (maximum lag) used in long-run variance calculations via calc_lrvar_bartlett.
westerlund	Logical. If TRUE, uses a Westerlund-specific information criterion and trimming logic for variance estimation.
aic	Logical. If TRUE, uses AIC for lag/lead selection when ranges. If FALSE, uses BIC.
bootno	Logical. If TRUE, prints a short header and progress dots (intended for higher-level routines).
indiv.ecm	Logical. If TRUE, gets output of individual ECM regressions.
verbose	Logical. If TRUE, prints additional output.

Details

Purpose and status. `WesterlundPlain()` is typically called internally by `westerlund_test`. It returns the four *raw* test statistics and lag/lead diagnostics needed for printing and standardization.

Workflow overview. The routine proceeds in two main stages:

1. **Unit-specific ECM regressions (Loop 1):** For each cross-sectional unit, it constructs an ECM with Δy_t as the dependent variable and includes deterministic terms (optional), y_{t-1}, x_{t-1}, lagged Δy_t, and leads/lags of Δx_t. Lags and leads are computed using strict time-indexed helpers (`get_lag`, `get_diff`), which respect gaps in the time index. If `lags` and/or `leads` are provided as ranges, an information-criterion search selects the lag/lead orders for each unit. The routine stores the unit-level error-correction estimate $\hat{\alpha}_i$ and its standard error.
2. **Pooled (panel) aggregation (Loop 2):** Using the mean of selected lag/lead orders across units, the routine constructs pooled quantities needed for P_t and P_a via partialling-out regressions and cross-unit aggregation of residual products.

Long-run variance calculations. Long-run variances are computed using `calc_lrvar_bartlett` with `maxlag` = `lrwindow`. In `westerlund=TRUE` mode, the routine applies Stata-like trimming at the start/end of the differenced series based on selected lags/leads prior to long-run variance estimation.

Returned statistics. Let $\hat{\alpha}_i$ denote the unit-specific error-correction coefficient on y_{t-1} (as constructed in the ECM), with standard error $\widehat{\mathrm{se}}(\hat{\alpha}_i)$. The routine computes:

- G_t: the mean of the individual t-ratios $\hat{\alpha}_i / \widehat{\mathrm{se}}(\hat{\alpha}_i)$,
- G_a: a scaled mean-group statistic using a unit-specific normalization factor derived from long-run variances,
- P_t: a pooled t-type statistic based on a pooled $\hat{\alpha}$ and its pooled standard error,
- P_a: a pooled scaled statistic using an average effective time dimension.

Value

A nested list containing:

- `stats`: A list of the four raw Westerlund test statistics:
  - `Gt`: Mean-group tau statistic.
  - `Ga`: Mean-group alpha statistic.
  - `Pt`: Pooled tau statistic.
  - `Pa`: Pooled alpha statistic.
- `indiv_data`: A named list where each element corresponds to a cross-sectional unit (ID), containing:
  - `ai`: The estimated speed of adjustment (alpha).
  - `seai`: The standard error of alpha (adjusted for degrees of freedom).
  - `betai`: Vector of long-run coefficients ($\beta = -\lambda/\alpha$).
  - `blag`, `blead`: The lags and leads selected for that specific unit.
  - `ti`: Raw observation count for the unit.
  - `tnorm`: Degrees of freedom used for normalization.
  - `reg_coef`: If `indiv.ecm` = `TRUE`, the full coefficient matrix from `westerlund_test_reg`.

- `results_df`: A summary `data.frame` containing all unit-level results in vectorized format.
- `settings`: A list of routine metadata:
  - `constant`: Logical indicating if a constant was included.
  - `trend`: Logical indicating if a trend was included.
  - `meanlag`, `meanlead`: Integer averages of the selected unit lags/leads.
  - `realmeanlag`, `realmeanlead`: Numeric averages of the selected unit lags/leads.
  - `auto`: Logical; `TRUE` if automatic selection (ranges) was used.

Internal Logic

Two-stage structure: Loop 1 (mean-group) estimates unit-specific ECMs. Each unit produces an estimated error-correction coefficient on y_{t-1} and an associated standard error. These are aggregated into G_t and G_a.

Loop 2 (pooled) fixes a common short-run structure based on the average selected lag/lead orders and constructs pooled residual products to obtain P_t and P_a.

Strict time indexing and gaps: All lags and differences are computed using strict time-based helpers (`get_lag`, `get_diff`). This ensures that gaps in the time index propagate as missing values rather than shifting across gaps.

References

Westerlund, J. (2007). Testing for error correction in panel data. *Oxford Bulletin of Economics and Statistics*, 69(6), 709–748.

See Also

`westerlund_test`, `WesterlundBootstrap`, `get_lag`, `get_diff`, `calc_lrvar_bartlett`

Examples

```
set.seed(123)
N <- 5
T <- 20
df <- data.frame(
  id = rep(1:N, each = T),
  t  = rep(1:T, N),
  y  = rnorm(N * T),
  x1 = rnorm(N * T),
  x2 = rnorm(N * T)
)

touse <- rep(TRUE, nrow(df))

plain_res <- WesterlundPlain(
  data       = df,
  touse      = touse,
  idvar      = "id",
  timevar    = "t",
  yvar       = "y",
```

```
  xvars      = c("x1","x2"),
  lags       = 1,
  leads      = 0
)

# Accessing results from the nested structure:
stats <- plain_res$stats
print(c(Gt = stats$Gt, Ga = stats$Ga, Pt = stats$Pt, Pa = stats$Pa))

# Checking unit-specific coefficients for ID '101'
unit_101 <- plain_res$indiv_data[["101"]]
print(unit_101$ai)
```

Index