

Package: spantest (via r-universe)

October 11, 2026

Title Mean-Variance Spanning Tests

Version 1.4-4

Date 2026-10-10

Description Provides a comprehensive suite of portfolio spanning tests for asset pricing, such as Huberman and Kandel (1987) [<doi:10.1111/j.1540-6261.1987.tb03917.x>](https://doi.org/10.1111/j.1540-6261.1987.tb03917.x), Gibbons et al. (1989) [<doi:10.2307/1913625>](https://doi.org/10.2307/1913625), Kempf and Memmel (2006) [<doi:10.1007/BF03396737>](https://doi.org/10.1007/BF03396737), Pesaran and Yamagata (2024) [<doi:10.1093/jjfinec/nbad002>](https://doi.org/10.1093/jjfinec/nbad002), Gungor and Luger (2016) [<doi:10.1080/07350015.2015.1019510>](https://doi.org/10.1080/07350015.2015.1019510), and the randomized alpha test of Massacci et al. (2026) [<doi:10.48550/arXiv.2507.17599>](https://doi.org/10.48550/arXiv.2507.17599).

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

URL <https://github.com/ArdiaD/spantest>

BugReports <https://github.com/ArdiaD/spantest/issues>

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports Rcpp, Rdpack, stats

LinkingTo Rcpp, RcppArmadillo

RdMacros Rdpack

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.0

Repository <https://ardiad.r-universe.dev>

Date/Publication 2026-10-11 02:32:14 UTC

RemoteUrl <https://github.com/ardiad/spantest>

RemoteRef HEAD

RemoteSha d2b0baeb98d9240154bf16a50463a4f3f3f39a82

Contents

DGP_NAMES	2
span_as	3
span_bj	5
span_dgp_table	6
span_f1	7
span_f2	8
span_gl_a	9
span_gl_ad	10
span_grs	12
span_hk	13
span_km	14
span_mstv	15
span_py	18
span_simulate	19
Index	23

DGP_NAMES	<i>Names of the built-in simulation processes</i>
-----------	---

Description

A character vector of length twelve giving a name for each preset of [span_simulate\(\)](#). **Position in this vector is the preset number**, so DGP_NAMES[9] is the name of dgp = 9.

Usage

DGP_NAMES

Format

A character vector of length 12.

See Also

[span_dgp_table\(\)](#), [span_simulate\(\)](#)

span_as	<i>Ardia and Sessinou (2025) Subseries-Based Cauchy Combination Test (SCT) for Spanning</i>
---------	---

Description

Computes robust p-values for linear spanning restrictions using a residual-based subseries procedure with Cauchy Combination Test (CCT) aggregation. Supports high-dimensional inference for H_0^δ (variance/spread slopes), H_0^α (intercepts), and the joint null $H_0^{\alpha,\delta}$.

Usage

```
span_as(bench, test, control = list())
```

Arguments

bench	Numeric matrix of benchmark returns, dimension $T \times K$.
test	Numeric matrix of test-asset returns, dimension $T \times N$.
control	Optional list passed to internal computation: - ks Numeric vector of subseries exponents; each sets the NUMBER of blocks to $\text{floor}(T^k)$, each of length about $T / \text{floor}(T^k)$; default $c(1/3)$. - L Numeric vector of the number of multiplier factors in the weights of each statistic (0: no weights); default $c(0, 2)$. See ‘Weights’. - seed Seed of the draw of the weights when $L > 0$; default 123. A call is one draw; a simulation should pass a different seed in each replication; see ‘One draw per call’ and ‘Simulations’.

Details

For each k in ks , the sample is partitioned into $\text{floor}(T^k)$ contiguous, NON-overlapping subseries – an exact partition of $1:T$, not a moving block. Residual perturbations controlled by L generate test statistics robust to serial and cross-sectional dependence and conditional heteroskedasticity. Resulting sub-p-values are aggregated by the Cauchy Combination Test (CCT), which remains valid under dependence and retains power in high dimensions.

Value

A named list of global (combined) p-values. Names encode hypothesis and settings:

- $\text{CCTd}_{L\{L\}_k\{i\}}$ — variance (slope) spanning, $\delta = 0$;
- $\text{CCTa}_{L\{L\}_k\{i\}}$ — alpha spanning, $\alpha = 0$;
- $\text{CCTad}_{L\{L\}_k\{i\}}$ — joint mean–variance spanning, $\alpha = 0$, $\delta = 0$.

Weights

When $L > 0$ the score of each statistic is multiplied by its own weights $\kappa_{i,t} = \prod_{l=1}^L \kappa_{l,i,t}$, $\kappa_{l,i,t} \sim N(1, 1)$, drawn independently of the data and of each other across factors, dates, assets and between the alpha and the delta score of an asset. Since version 1.4-3 this is the definition of the theory; up to version 1.4-2 one weight vector was shared by every asset and by both scores. Sharing it leaves the cross-sectional dependence of the statistics intact, while independent weights shrink it, which matters when the residuals carry a pervasive common factor: at $L = 2$, $T = 250$, $K = 2$ and 10 and $N = 100$ and 1000, the three tests reject a true null in 6.0–8.5\ 2.4–4.8\

One draw per call

At $L > 0$ a call returns the test for one draw of the weights, from seed: the alpha weights first, then the delta weights. The test is valid for any draw — its size is correct — but the p-value it returns on one data set varies from draw to draw, since a Cauchy average of independent terms does not concentrate as N grows; report the seed with the result. R scrambles the seed when initialising the Mersenne-Twister, so draws from consecutive seeds behave as independent draws. Up to version 1.4-2 a control B merged B draws per asset with the Cauchy rule; it was removed in 1.4-3.

Simulations

A Monte Carlo study should draw new weights in each replication, for instance `seed = base + r` in replication `r`. With the same seed in every replication, every cell of the study is computed on one set of weights: the results are one realisation of the weights, common to all cells, and the Monte Carlo standard errors understate their uncertainty. The weights enter through sums over each subseries, so the effect on size is small but not nil (about one point at $T = 250$, in either direction), and the effect on power can be larger: in the simulation study of Ardia and Sessinou, the weights of seed 123 overstated the power at $L = 2$ by 10 to 25 points at moderate alternatives.

References

Ardia D, Sessinou R (2025). “Robust Inference in Large Panels and Markowitz Portfolios.” doi:10.2139/ssrn.5033399. Working paper.

See Also

Other Alpha Spanning Tests: [span_bj\(\)](#), [span_f1\(\)](#), [span_g1_a\(\)](#), [span_grs\(\)](#), [span_mstv\(\)](#), [span_py\(\)](#)

Other Variance Spanning Tests: [span_f2\(\)](#), [span_km\(\)](#)

Other Joint Mean-Variance Spanning Tests: [span_g1_ad\(\)](#), [span_hk\(\)](#)

Examples

```
set.seed(123)
bench <- matrix(rnorm(300), 100, 3)
test <- matrix(rnorm(200), 100, 2)
out <- span_as(bench, test)
out$CCTa_L0_k1; out$CCTd_L0_k1; out$CCTad_L0_k1
```

span_bj

*Britten–Jones Tangency-Portfolio Spanning Test (1999)***Description**

Tests whether the tangency (maximum Sharpe) portfolio of the augmented universe (benchmarks + test assets) is spanned by the benchmark assets alone. Following Britten–Jones (1999), the statistic arises from a regression of a constant on the raw asset returns (without an intercept) and yields an F test of the tangency-spanning restriction.

Usage

```
span_bj(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
 R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

Following Britten–Jones (1999), the (unnormalized) tangency-portfolio weights are obtained by regressing a vector of ones on the raw asset returns without an intercept. The null that the benchmarks span the tangency portfolio is the restriction that the test assets carry zero weight, tested by comparing the residual sums of squares of the full and benchmark-only regressions. The statistic has an $F_{N, T-K-N}$ reference distribution; finite-sample feasibility requires $T - K - N \geq 1$.

Equivalence with `span_grs()`. Although derived from a regression of ones on raw returns rather than from intercepts, this statistic is numerically identical to the Gibbons–Ross–Shanken statistic returned by `span_grs()`, to machine precision and for every (T, K, N) . The two are the same test of $\alpha = 0$ computed by different routes; agreement between them validates the implementation but is not independent evidence. `span_f1()` is close but genuinely distinct.

Value

A named list with components:

`pval` P-value for the F -statistic under the null.

`stat` Britten–Jones F -statistic.

`H0` Null hypothesis description, "alpha = 0" (the test assets do not expand the tangency portfolio spanned by the benchmarks).

References

Britten-Jones M (1999). "The Sampling Error in Estimates of Mean-Variance Efficient Portfolio Weights." *The Journal of Finance*, **54**(2), 655–671.

See Also

Other Alpha Spanning Tests: [span_as\(\)](#), [span_f1\(\)](#), [span_g1_a\(\)](#), [span_grs\(\)](#), [span_mstv\(\)](#), [span_py\(\)](#)

Examples

```
set.seed(321)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # tests:      T=100, N=2
out <- span_bj(R1, R2)
out$stat; out$pval; out$H0
```

span_dgp_table	<i>The simulation processes, by name and preset number</i>
----------------	--

Description

Returns the correspondence between the names accepted by [span_simulate\(\)](#)'s `dgp` argument, the preset numbers, and the process each one denotes. Useful when reading code or output that refers to a process by number.

Usage

```
span_dgp_table()
```

Value

A data frame with one row per process and columns preset (integer), name (character), innovation, dynamics, df, standardize and scale. Together they are enough to rebuild any preset by hand with [span_simulate\(\)](#)'s explicit arguments; a table that described a process only partly is how a discrepancy between the two once went unnoticed.

The twelve are a 3 x 4 factorial of innovation law by dynamics, so a column varies the dynamics alone: every heavy-tailed preset is standardised, and `scale = "process"` gives each one unit unconditional variance whatever its dynamics.

See Also

[span_simulate\(\)](#), [DGP_NAMES](#)

Examples

```
span_dgp_table()

# the same process, two ways
set.seed(1); a <- span_simulate(100, 2, 3, dgp = "AR-GARCH-SKST")
set.seed(1); b <- span_simulate(100, 2, 3, dgp = 9)
identical(a, b)
```

span_f1

*F1 Alpha-Spanning Test (Intercepts Only)***Description**

Tests the null $H_0 : \alpha = 0$ that the intercepts of the test assets are jointly zero when regressed on the benchmark assets, i.e., benchmarks span the mean of the test assets. This is the F1 test of Kan & Zhou (2012).

Usage

```
span_f1(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
 R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

Under standard assumptions (i.i.d. returns, full-rank covariances), the reference distribution is $F_{N, T-K-N}$. Finite-sample feasibility requires $T - K - N \geq 1$. The covariance matrices are the maximum-likelihood ones (divisor T), as in Kan and Zhou (2012); with them F1, the first step of their step-down test, is the GRS statistic, so `span_f1()` returns the same value as `span_grs()` and `span_bj()`. Up to version 1.4-2 it used the divisor $T - 1$.

Value

A named list with components:

`pval` P-value for the F -statistic under the null.

`stat` F1 F -statistic.

`H0` Null hypothesis description, "alpha = 0".

References

Kan R, Zhou G (2012). "Tests of Mean-Variance Spanning." *Annals of Economics and Finance*, **13**(1), 145–193.

See Also

Other Alpha Spanning Tests: [span_as\(\)](#), [span_bj\(\)](#), [span_gl_a\(\)](#), [span_grs\(\)](#), [span_mstv\(\)](#), [span_py\(\)](#)

Examples

```
set.seed(123)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # tests:      T=100, N=2
out <- span_f1(R1, R2)
out$stat; out$pval; out$H0
```

span_f2

F2 Variance-Spanning Test (Slopes Only)

Description

Tests the null $H_0 : \delta = 0$ that adding test assets does not improve the minimum-variance frontier spanned by the benchmarks (variance spanning). The statistic compares frontier-defining quantities of the augmented (benchmark + test) universe to those of the benchmark subset.

Usage

```
span_f2(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

Under standard conditions (i.i.d. returns, full-rank covariances), the reference distribution is $F_{N, T-K-N+1}$. Finite-sample feasibility requires $T - K - N + 1 \geq 1$. F2 is the second step of the step-down test of Kan and Zhou (2012): it tests $\delta = 0$ conditional on $\alpha = 0$, so it is a test of $\delta = 0$ alone only when the intercepts are zero; for $\delta = 0$ with free intercepts, use `span_km()`. The covariance matrices are the maximum-likelihood ones (divisor T); up to version 1.4-2 they used $T - 1$.

Value

A named list with components:

- pval P-value for the F -statistic under the null.
- stat F2 F -statistic.
- H0 Null hypothesis description, "delta = 0".

References

Kan R, Zhou G (2012). "Tests of Mean-Variance Spanning." *Annals of Economics and Finance*, **13**(1), 145–193.

See Also

Other Variance Spanning Tests: [span_as\(\)](#), [span_km\(\)](#)

Examples

```
set.seed(123)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # tests:      T=100, N=2
out <- span_f2(R1, R2)
out$stat; out$pval; out$H0
```

span_gl_a

*Gungor-Luger Alpha-Only Spanning Test (2016)***Description**

Tests the null $H_0 : \alpha = 0$ that benchmark assets span the mean (intercepts) of the test assets. Following Gungor & Luger (2016), the procedure uses a Monte Carlo (MC) test based on an $F_{\max}$ statistic with residual sign-flip simulations, yielding local (LMC) and bounds (BMC) Monte Carlo p-values and a three-way decision rule.

Usage

```
span_gl_a(R1, R2, control = list())
```

Arguments

R1	Numeric matrix of benchmark returns, dimension $T \times K$.
R2	Numeric matrix of test-asset returns, dimension $T \times N$.
control	List of options: - totsim Number of MC simulations (default 500). - pval_thresh Significance level for decisions (default 0.05). - do_trace Logical; print progress (default FALSE).

Details

Accept if $pval_LMC > \alpha$; Reject if $pval_BMC \leq \alpha$; otherwise Inconclusive. The subseries sign-flip MC approach is robust to heteroskedasticity, serial dependence, and heavy tails, making it suitable for high-dimensional settings where classical alpha tests (e.g., GRS) may suffer from size distortions.

Value

A list with components:

pval_LMC Local Monte Carlo p-value, the liberal one: an acceptance is read off it.

pval_BMC Bounds Monte Carlo p-value, the conservative one: a rejection is read off it.

stat Observed $F_{\max}$ statistic: the largest asset-level ratio $(SSR_r - SSR_u)/SSR_u$. The usual F form multiplies it by a positive constant, which leaves the ranks, the p-values and the decisions unchanged.

Decisions Decision code: 1 = Accept, 0 = Reject, NA = Inconclusive.

Decisions_string Text label: "Accept", "Reject", or "Inconclusive".

H0 Null hypothesis description, "alpha = 0".

References

Gungor S, Luger R (2016). "Multivariate Tests of Mean-Variance Efficiency and Spanning With a Large Number of Assets and Time-Varying Covariances." *Journal of Business & Economic Statistics*, **34**(2), 161–175.

See Also

Other Alpha Spanning Tests: [span_as\(\)](#), [span_bj\(\)](#), [span_f1\(\)](#), [span_grs\(\)](#), [span_mstv\(\)](#), [span_py\(\)](#)

Examples

```
set.seed(1234)
R1 <- matrix(rnorm(300), 100, 3)
R2 <- matrix(rnorm(200), 100, 2)
out <- span_gl_a(R1, R2, control = list(totsim = 100, do_trace = FALSE))
out$Decisions_string; out$pval_LMC; out$pval_BMC
```

span_gl_ad

Gungor–Luger Joint Mean–Variance Spanning Test (2016)

Description

Tests the joint null $H_0 : \alpha = 0, \delta = 0$ that benchmark assets span both intercepts and slopes of the test assets, allowing for heteroskedasticity, serial dependence, and time-varying covariances. Following Gungor & Luger (2016), the procedure uses a Monte Carlo (MC) test based on an $F_{\max}$ statistic with residual sign-flip simulations, yielding local (LMC) and bounds (BMC) Monte Carlo p-values and a three-way decision rule.

Usage

```
span_gl_ad(R1, R2, control = list())
```

Arguments

R1	Numeric matrix of benchmark returns, dimension $T \times K$.
R2	Numeric matrix of test-asset returns, dimension $T \times N$.
control	List of options: totsim Number of MC simulations (default 500). pval_thresh Significance level for decisions (default 0.05). do_trace Logical; print progress (default FALSE).

Details

LMC/BMC follow Gungor & Luger's MC framework with residual sign-flip draws under the null. The rule is: Accept if $pval_LMC > \alpha$; Reject if $pval_BMC \leq \alpha$; otherwise Inconclusive. This approach is robust in high-dimensional and time-varying volatility settings where classical joint spanning tests can be unreliable.

Value

A list with components:

pval_LMC Local Monte Carlo p-value, the liberal one: an acceptance is read off it.

pval_BMC Bounds Monte Carlo p-value, the conservative one: a rejection is read off it.

stat Observed $F_{\max}$ statistic: the largest asset-level ratio $(SSR_r - SSR_u)/SSR_u$. The usual F form multiplies it by a positive constant, which leaves the ranks, the p-values and the decisions unchanged.

Decisions Decision code: 1 = Accept, 0 = Reject, NA = Inconclusive.

Decisions_string Text label: "Accept", "Reject", or "Inconclusive".

H0 Null hypothesis description, "alpha = 0 and delta = 0".

References

Gungor S, Luger R (2016). "Multivariate Tests of Mean-Variance Efficiency and Spanning With a Large Number of Assets and Time-Varying Covariances." *Journal of Business & Economic Statistics*, **34**(2), 161–175.

See Also

Other Joint Mean-Variance Spanning Tests: [span_as\(\)](#), [span_hk\(\)](#)

Examples

```
set.seed(123)
R1 <- matrix(rnorm(300), 100, 3)
R2 <- matrix(rnorm(200), 100, 2)
out <- span_gl_ad(R1, R2, control = list(totsim = 100, do_trace = FALSE))
out$Decisions_string; out$pval_LMC; out$pval_BMC
```

span_grs

*Gibbons–Ross–Shanken (GRS) Alpha Spanning Test (1989)***Description**

Implements the GRS test of the joint null $H_0 : \alpha = 0$ in the multivariate regression of test-asset returns on benchmark portfolios (with an intercept). The statistic assumes homoskedastic, normally distributed errors and is most reliable when T is large relative to K (benchmarks) and N (test assets).

Usage

```
span_grs(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
 R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

Under standard conditions, the reference distribution is $F_{N, T-N-K}$. Finite-sample feasibility requires $T - N - K \geq 1$.

The residual covariance enters in its maximum-likelihood form (divisor T), which is what pairs with the $(T - N - K)/N$ scaling to give an exact F test.

Equivalence with [span_bj\(\)](#). This statistic is numerically identical to the Britten–Jones (1999) statistic returned by [span_bj\(\)](#), to machine precision and for every (T, K, N) : the two are different computational routes to the same test of $\alpha = 0$, not independent procedures. Reporting both is therefore a check on the implementation, not corroborating evidence from two tests. [span_f1\(\)](#) is close but genuinely distinct.

Value

A named list with components:

pval P-value for the F -statistic under the null.

stat GRS F -statistic.

H0 Null hypothesis description, "alpha = 0".

References

Gibbons MR, Ross SA, Shanken J (1989). "A Test of the Efficiency of a Given Portfolio." *Econometrica*, **57**(5), 1121–1152.

See Also

Other Alpha Spanning Tests: [span_as\(\)](#), [span_bj\(\)](#), [span_f1\(\)](#), [span_g1_a\(\)](#), [span_mstv\(\)](#), [span_py\(\)](#)

Examples

```
set.seed(42)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # tests:      T=100, N=2
out <- span_grs(R1, R2)
out$stat; out$pval; out$H0
```

span_hk	<i>Huberman–Kandel Joint Mean–Variance Spanning Test (1987)</i>
---------	---

Description

Tests the joint null $H_0 : \alpha = 0, \delta = 0$ that the benchmark assets span the mean–variance frontier of the augmented (benchmark + test) universe. Following Huberman & Kandel (1987), the statistic compares the frontiers with and without the additional assets.

Usage

```
span_hk(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

The test evaluates whether adding the test assets changes the efficient frontier implied by the benchmarks. Under standard regularity conditions, the statistic has an F reference with $(2N, 2(T - K - N))$ degrees of freedom. Finite-sample feasibility requires $T - K - N \geq 1$. The covariance matrices are the maximum-likelihood ones (divisor T), which makes the statistic the likelihood-ratio form; up to version 1.4-2 they used $T - 1$.

Value

A named list with components:

pval P-value for the F -statistic under the null.

stat F -statistic value.

H0 Null hypothesis description, "alpha = 0 and delta = 0".

References

Huberman G, Kandel S (1987). “Mean-Variance Spanning.” *The Journal of Finance*, **42**(4), 873–888.

See Also

Other Joint Mean-Variance Spanning Tests: [span_as\(\)](#), [span_gl_ad\(\)](#)

Examples

```
set.seed(123)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # tests:      T=100, N=2
out <- span_hk(R1, R2)
out$stat; out$pval; out$H0
```

span_km

Kempf–Mommel GMVP Spanning Test

Description

Tests whether the Global Minimum Variance Portfolio (GMVP) of the combined (benchmark + test) universe equals the GMVP of the benchmark assets alone. Following Kempf & Mommel (2006), the null assesses whether adding new assets improves the minimum-variance frontier.

Usage

```
span_km(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

The null hypothesis H_0 is that augmenting the benchmark set with the test assets does not change the GMVP weights ($\Delta = 0$), i.e., the GMVP of the full universe coincides with that of the benchmark subset. The test uses the Kempf–Mommel regression

$$r_{1t} = \alpha + \sum_{j>1} \beta_j (r_{1t} - r_{jt}) + \varepsilon_t,$$

whose slopes are the global-minimum-variance portfolio weights ($w_j = \beta_j$ for $j > 1$ and $w_1 = 1 - \sum_{j>1} \beta_j$). The test assets leave the GMVP unimproved exactly when their β block is zero,

so H_0 is an N -dimensional linear restriction, tested by an F statistic with reference distribution $F_{N, T-K-N}$. Finite-sample feasibility requires $T - K - N \geq 1$.

Note that regressing a vector of ones on the returns instead—the Britten–Jones tangency construction used by `span_bj()`—produces a test of $\alpha = 0$, not of $\Delta = 0$. The two are easily confused: they share the difference-of-returns design matrix and differ only in the dependent variable and the intercept.

Value

A named list with components:

`pval` P-value for the F-statistic under the null.

`stat` F-statistic value.

`H0` Null hypothesis description, "delta = 0" (equivalently, GMVP of the benchmark set equals GMVP of the full universe).

References

Kempf A, Memmel C (2006). "Estimating the Global Minimum Variance Portfolio." *Schmalenbach Business Review*, **58**(4), 332–348.

See Also

Other Variance Spanning Tests: `span_as()`, `span_f2()`

Examples

```
set.seed(123)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # test assets: T=100, N=2
ans <- span_km(R1, R2)
ans$pval; ans$stat; ans$H0
```

span_mstv

Randomized Alpha Test of Massacci, Sarno, Trapani and Vallarino

Description

Implements the randomized test of Massacci et al. (2026) for the joint null that every intercept is zero in a linear factor pricing model. The test is built from equation-by-equation estimation, needs no covariance matrix, and allows N to grow faster than T . Both the one-shot test of their Theorem 3.1 and the derandomized decision rule of their Section 3.2 are returned.

Usage

```
span_mstv(R1, R2, control = list())
```

Arguments

R1	Numeric matrix of benchmark returns, dimension $T \times K$.
R2	Numeric matrix of test-asset returns, dimension $T \times N$.
control	Optional list: - nu Number of finite moments the data are assumed to admit, $\nu \geq 4$; default 5, the value of the authors' simulations. See 'Choosing nu'. - tau Nominal level at which the derandomized rule decides; default 0.05. It does not affect pval. - B Number of perturbation draws of the derandomized rule; default $\text{floor}(\log(N)^2)$, their guideline. - seed Seed of the perturbation draws: first the one behind stat and pval, then the B draws of the derandomized rule; default 123. The caller's RNG state is restored. A simulation must pass a different seed in each replication; see 'Simulations'.

Details

Write $\hat{\alpha}_i$ for the OLS intercept of test asset i on the benchmarks and $\hat{s}_{NT}$ for the pooled root mean squared residual over the whole panel. The test rescales the intercepts into

$$\psi_i = \left(T^{1/\nu} |\hat{\alpha}_i| / \hat{s}_{NT} \right)^{\nu/2} = \sqrt{T} (|\hat{\alpha}_i| / \hat{s}_{NT})^{\nu/2},$$

which drifts to zero under the null and diverges under the alternative, perturbs them with independent standard normals, and takes the largest, $Z_{N,T} = \max_i(\psi_i + \omega_i)$. Under the null $Z_{N,T}$ is the maximum of N standard normals, so $a_N^{-1}(Z_{N,T} - b_N)$ is asymptotically Gumbel with the usual extreme-value constants, giving both `crit` and `pval`. Only rates of convergence are used, which is what frees the procedure from estimating a covariance matrix and lets N outgrow T .

Because the perturbation does not vanish asymptotically, two researchers applying the one-shot test to the same data can reach different conclusions. Their remedy is to repeat the perturbation B times and record the share $Q_{N,T,B}(\tau)$ of replications that do not reject, then decide against the null when that share falls below $(1 - \tau) - f(B)$ with $f(B) = B^{-1/4}$. As in the authors' replication code, the share is simulated with $B = \lfloor (\log N)^2 \rfloor$ draws by default, so it is a multiple of $1/B$ and the decision depends on the draws. Up to version 1.4-2 the share was evaluated in closed form, as its $B \rightarrow \infty$ limit $\prod_i \Phi(c_\tau - \psi_i)$; that limit is a different rule in practice, since with B between 15 and 50 the simulated share is coarse.

The Gumbel limit is an approximation in N , and a poor one when N is small: at $N = 2$ the one-shot test rejects about a quarter of the time at a nominal 5%. The function therefore returns NA below $N = 10$, and N in the hundreds is where the procedure is meant to operate.

Value

A named list with components:

- pval P-value of the one-shot test, from the Gumbel limit.
- stat The one-shot statistic $Z_{N,T}$.
- H0 Null hypothesis description, "alpha = 0".

`crit` Critical value c_τ of the one-shot test.

`Q` Derandomized quantity $Q_{N,T,B}(\tau)$: the share of the B perturbation draws whose statistic does not exceed `crit`.

`reject` Decision of the derandomized rule: TRUE when $Q < (1 - \tau) - B^{-1/4}$.

`nu, B` The settings used.

All components except `H0` are NA when the test does not apply.

Choosing nu

ν is the number of finite moments assumed, entering only through the exponent $\nu/2$; the theory needs $\nu \geq 4$ and the admissible growth of N relative to T widens with it. It can be estimated with a tail-index estimator, or bounded below by testing $E|y_{i,t}|^{\nu_0}$ for a candidate ν_0 . The default here is $\nu = 5$, the value of the authors' simulations; their empirical application reports $\nu = 4$ and $\nu = 5$. A smaller ν makes the test more liberal. Up to version 1.4-2 the default was 4.

Simulations

The size of the one-shot test is a probability over the data and the perturbation together, so a Monte Carlo study must draw a new perturbation in each replication, for instance `seed = base + r` in replication `r`. With the same seed in every replication, all replications share one vector ω and the rejection rate measures the test conditionally on that draw. Under the null at small K the ψ_i are close to zero and the statistic is essentially $\max_i \omega_i$, so that rate is near 0 or near 1 whatever the data: with the default seed and $N = 1000$, $\max_i \omega_i = 3.24$ against a critical value of 3.98, and the rejection rate is 0 instead of about 4%. The derandomized rule draws its B perturbations from the same seed, so the same holds for it.

References

Massacci D, Sarno L, Trapani L, Vallarino P (2026). "A General Randomized Test for Alpha." *Journal of the American Statistical Association*. doi:10.48550/arXiv.2507.17599. Forthcoming.

See Also

Other Alpha Spanning Tests: [span_as\(\)](#), [span_bj\(\)](#), [span_f1\(\)](#), [span_gl_a\(\)](#), [span_grs\(\)](#), [span_py\(\)](#)

Examples

```
set.seed(123)
R1 <- matrix(rnorm(3 * 250), 250, 3)      # benchmarks: T=250, K=3
R2 <- matrix(rnorm(100 * 250), 250, 100) # tests:      T=250, N=100
out <- span_mstv(R1, R2)
out$pval; out$Q; out$reject
```

span_py

*Pesaran–Yamagata Alpha Spanning Test (2024)***Description**

Implements the Pesaran–Yamagata test for the joint null that all intercepts are zero in a multi-factor spanning regression with possible cross-sectional dependence across test assets.

Usage

```
span_py(R1, R2)
```

Arguments

R1 Numeric matrix of benchmark returns, dimension $T \times K$.
 R2 Numeric matrix of test-asset returns, dimension $T \times N$.

Details

The null hypothesis is that all intercepts are zero ($\alpha = 0$), meaning the benchmark assets span the expected returns of the test assets. The statistic adjusts for cross-sectional dependence via the residual covariance and has an asymptotic $\mathcal{N}(0, 1)$ reference under large T, N . It uses the asset-level t-statistics and the pairwise residual correlations only, never an inverse of the $N \times N$ residual covariance, so it is defined when N exceeds T . Finite-sample safeguards require $T - K - 1 > 4$ and at least two test assets ($N \geq 2$); otherwise pval and stat are returned as NA. Up to version 1.4-2 it also required $N \leq T - K - 1$.

Value

A named list with components:

pval P-value under the standard normal reference distribution.

stat Standardized test statistic.

H0 Null hypothesis description, "alpha = 0".

References

Pesaran MH, Yamagata T (2024). “Testing for alpha in linear factor pricing models with a large number of securities.” *Journal of Financial Econometrics*, **22**(2), 407–460.

See Also

Other Alpha Spanning Tests: [span_as\(\)](#), [span_bj\(\)](#), [span_f1\(\)](#), [span_gl_a\(\)](#), [span_grs\(\)](#), [span_mstv\(\)](#)

Examples

```

set.seed(123)
R1 <- matrix(rnorm(300), 100, 3) # benchmarks: T=100, K=3
R2 <- matrix(rnorm(200), 100, 2) # tests:      T=100, N=2
out <- span_py(R1, R2)
out$pval; out$stat; out$H0

```

span_simulate

*Simulate Benchmark and Test-Asset Returns for Spanning Studies***Description**

Generates a panel of benchmark returns R_1 ($n \times K$) and test-asset returns R_2 ($n \times N$) from a factor model with a controllable mean-variance spanning violation, matching the data-generating processes used in the size/power study of Ardia and Sessinou (2025). It is a fast, self-contained replacement for ad-hoc `fGarch::garchSim()` loops: the innovations are drawn in one vectorised call and the AR / GARCH dynamics are applied with a lightweight recursion.

Usage

```

span_simulate(
  n,
  K,
  N,
  ncp = 0,
  rho_factor = 0.8,
  rho_error = 0.5,
  innovation = c("normal", "t", "skew-t"),
  dynamics = c("iid", "ar", "garch", "ar-garch"),
  sparse = FALSE,
  df = 5,
  xi = 0.9,
  ar = 0.2,
  garch = c(omega = 0.1, alpha = 0.1, beta = 0.8),
  standardize = TRUE,
  scale = c("innovation", "process"),
  dgp = NULL,
  gamma = NULL,
  burnin = 500L
)

```

Arguments

<code>n</code>	Integer, number of time periods (sample size).
<code>K</code>	Integer, number of benchmark assets.
<code>N</code>	Integer, number of test assets.

ncp	Non-centrality controlling the spanning violation (0 = null).
rho_factor, rho_error	Toeplitz correlation decays for the factors and the idiosyncratic terms, in $[0, 1)$.
innovation	Marginal law of the innovations: "normal", "t", or "skew-t". Ignored if dgp is supplied.
dynamics	Serial structure: "iid", "ar", "garch", or "ar-garch". Ignored if dgp is supplied.
sparse	Logical; if TRUE, zero the first $\lfloor N/2 \rfloor$ intercepts (sparse alternative).
df	Degrees of freedom for the Student-/skew- t innovations.
xi	Skewness parameter for the skew- t innovations.
ar	AR(1) coefficient (used by "ar" and "ar-garch").
garch	Named numeric vector $c(\omega, \alpha, \beta)$ for the GARCH(1,1) variance recursion.
standardize	Logical; standardise t innovations to unit variance.
scale	Where unit variance is imposed. "innovation" (the default, and the textbook parameterisation) gives the innovations unit variance, so an AR(1) process carries $1/(1 - \phi^2)$. "process" divides each series by its own unconditional standard deviation, so the process itself has unit variance whatever its dynamics. The twelve presets use "process", so that they differ in the shape of the dependence and the thickness of the tails and not in how much variance they carry. Rescaling does not change the autocorrelation function, so ar keeps its usual meaning; and it does not change any test result under the spanning null, where it rescales the whole system and the tests are scale-invariant.
dgp	Optional preset process, given either as a name — one of "iid-N", "iid-ST", "iid-SKST", "GARCH-N", "GARCH-ST", "GARCH-SKST", "AR-N", "AR-ST", "AR-SKST", "AR-GARCH-N", "AR-GARCH-ST", "AR-GARCH-SKST" (case-insensitive, surrounding whitespace ignored) — or as a whole number in 1:12, which may itself be given as a string ("9" and 9 select the same preset). Name matching is exact rather than partial, since "AR" is a prefix of six of the twelve names. Names are preferred: they say which cell of the innovation-by-dynamics grid is meant, whereas the number must be looked up. Numbers remain accepted so existing scripts and stored results keep working; see span_dgp_table() for the correspondence. The preset numbers are not the DGP numbers of Ardia and Sessinou: presets 7–9 are AR-GARCH and 10–12 are AR, where the paper's DGP7–9 are AR and DGP10–12 AR-GARCH. Pass names to reproduce the paper. Setting dgp overrides innovation/dynamics/df/xi/standardize.
gamma	Optional pervasive common component in the idiosyncratic terms: a single number, or a vector of N loadings, applied as $\varepsilon_{it} + \gamma_i g_t$ with g_t drawn from the same process as everything else. NULL (the default) draws nothing and leaves the simulation exactly as it was before the argument existed. Unlike the Toeplitz decay of rho_error, this dependence does not die out across the cross-section, so it leaves the approximate-sparsity regime the spanning tests assume. Pass the loadings explicitly — for instance <code>gamma = runif(N, 0.7, 0.9)</code> — so that a replication does not depend on the state of the generator inside the function.
burnin	Integer, number of initial observations discarded to remove the AR/GARCH transient.

Details

The latent factors z are K innovation series, cross-sectionally correlated through the Cholesky factor of a Toeplitz matrix with decay `rho_factor`; the idiosyncratic terms are N innovation series with Toeplitz decay `rho_error`. The benchmarks are these series themselves, $R_1 = z$, and the test assets are

$$R_2 = \alpha + R_1 B + \varepsilon, \quad B_{1,j} = (1 + \text{ncp})(2 - K), \quad B_{k,j} = 1 + \text{ncp} \ (k \geq 2), \quad \alpha_j = \text{ncp},$$

so that $\delta_j = 1 - \sum_k B_{k,j} = -\text{ncp}$. Under `ncp = 0` the benchmarks span the test assets ($\alpha = 0$, $\delta = 0$); larger `|ncp|` moves both restrictions away from the null, α_j to `ncp` and δ_j to `-ncp`. Up to version 1.4-2 the benchmarks were $[z_1, z_{-1} + z_1]$ with loadings $1 + \text{ncp}$ on z : the same loadings on the benchmarks, but benchmarks with correlations near 0.9 and unequal variances rather than the Toeplitz structure of the process. With `sparse = TRUE` the first $\lfloor N/2 \rfloor$ intercepts are set to zero (a sparse alternative); α is added column-wise, i.e. asset j receives α_j .

Each innovation series is defined by innovation (marginal law) and dynamics (serial structure): "iid", an AR(1) with coefficient `ar`, a GARCH(1,1) with parameters `garch`, or their "ar-garch" combination. Student- t innovations are standardised to unit variance when `standardize = TRUE`; skew- t innovations (standardised Fernandez-Steel skew- t) are always unit-variance, so `standardize` has no effect on them.

`dgp` is a convenience argument selecting one of the twelve processes of Ardia and Sessinou (2025); when supplied it overrides `innovation`, `dynamics`, `df`, `xi` and `standardize`:

1	normal, iid	7	normal, AR-GARCH
2	Student-t(5), iid	8	Student-t(5), AR-GARCH
3	skew-t(4, 0.9), iid	9	skew-t(4, 0.9), AR-GARCH
4	normal, GARCH	10	normal, AR
5	Student-t(5), GARCH	11	Student-t(5), AR
6	skew-t(4, 0.9), GARCH	12	skew-t(4, 0.9), AR

The twelve are a 3×4 factorial: one innovation law per family — normal, standardised t_5 , standardised skew- t with 4 df and $\xi = 0.9$ — repeated across the four dynamics, so a column varies the dynamics alone.

Value

A list with components `R1` ($n \times K$ benchmark returns) and `R2` ($n \times N$ test-asset returns), suitable as the `R1/R2` arguments of the spanning tests in this package.

Note

Set the RNG state with `set.seed` beforehand for reproducibility.

References

- Ardia D, Sessinou R (2025). "Robust Inference in Large Panels and Markowitz Portfolios." doi:10.2139/ssrn.5033399. Working paper.
- Gungor S, Luger R (2016). "Multivariate Tests of Mean-Variance Efficiency and Spanning With a Large Number of Assets and Time-Varying Covariances." *Journal of Business & Economic Statistics*, 34(2), 161–175.

See Also

[span_grs\(\)](#), [span_gl_ad\(\)](#), [span_as\(\)](#)

Examples

```
set.seed(1)
sim <- span_simulate(n = 250, K = 3, N = 10, ncp = 0)      # spanning null
span_grs(sim$R1, sim$R2)$pval

sim2 <- span_simulate(n = 250, K = 3, N = 10, ncp = 0.2)  # alternative
span_grs(sim2$R1, sim2$R2)$pval
```

Index

* Alpha Spanning Tests

- span_as, 3
- span_bj, 5
- span_f1, 7
- span_gl_a, 9
- span_grs, 12
- span_mstv, 15
- span_py, 18

* Joint Mean-Variance Spanning Tests

- span_as, 3
- span_gl_ad, 10
- span_hk, 13

* Variance Spanning Tests

- span_as, 3
- span_f2, 8
- span_km, 14

DGP_NAMES, 2, 6

set.seed, 21

span_as, 3

span_as(), 6, 7, 9–11, 13–15, 17, 18, 22

span_bj, 5

span_bj(), 4, 7, 10, 12, 13, 15, 17, 18

span_dgp_table, 6, 20

span_dgp_table(), 2

span_f1, 7

span_f1(), 4–6, 10, 12, 13, 17, 18

span_f2, 8

span_f2(), 4, 15

span_gl_a, 9

span_gl_a(), 4, 6, 7, 13, 17, 18

span_gl_ad, 10

span_gl_ad(), 4, 14, 22

span_grs, 12

span_grs(), 4–7, 10, 17, 18, 22

span_hk, 13

span_hk(), 4, 11

span_km, 14

span_km(), 4, 9

span_mstv, 15

span_mstv(), 4, 6, 7, 10, 13, 18

span_py, 18

span_py(), 4, 6, 7, 10, 13, 17

span_simulate, 19

span_simulate(), 2, 6