Package 'TFORGE'

May 7, 2026

Title Tests for Geophysical Eigenvalues

Version 0.1.17

Description

The eigenvalues of observed symmetric matrices are often of intense scientific interest. This package offers single sample tests for the eigenvalues of the population mean or the eigenvalue-multiplicity of the population mean. For k-samples, this package offers tests for equal eigenvalues between samples. Included is support for matrices with constraints common to geophysical tensors (constant trace, sum of squared eigenvalues, or both) and eigenvectors are usually considered nuisance parameters. Pivotal bootstrap methods enable these tests to have good performance for small samples (n=15 for 3x3 matrices). These methods were developed and studied by Hingee, Scealy and Wood (2026, <doi:10.1080/01621459.2025.2606381>). Also available is a 2-sample test using a Gaussian orthogonal ensemble approximation and an eigenvalue-multiplicity test that assumes orthogonally-invariant covariance.

URL https://github.com/kasselhingee/TFORGE

Imports mvtnorm, purrr, Rdpack

Suggests knitr, rmarkdown, testthat

RdMacros Rdpack

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.5)

LazyData true

VignetteBuilder knitr

NeedsCompilation no

Author Kassel Liam Hingee [aut, cre] (ORCID:
<https://orcid.org/0000-0001-9894-2407>),
Art B. Owen [cph] (./R/scel.R only),
Board of Trustees Leland Stanford Junior University [cph] (./R/scel.R only)

Maintainer Kassel Liam Hingee <kassel.hingee@gmail.com>

Repository CRAN

Date/Publication 2026-03-26 08:20:07 UTC

Contents

`TFORGE-package` *TFORGE: Tests for Geophysical Eigenvalues*

Description

The eigenvalues of observed symmetric matrices are often of intense scientific interest. This package offers single sample tests for the eigenvalues of the population mean or the eigenvalue-multiplicity of the population mean. For k-samples, this package offers tests for equal eigenvalues between samples. Included is support for matrices with constraints common to geophysical tensors (constant trace, sum of squared eigenvalues, or both) and eigenvectors are usually considered nuisance parameters. Pivotal bootstrap methods enable these tests to have good performance for small samples (n=15 for 3x3 matrices). These methods were developed and studied by Hingee, Scealy and Wood (2026, doi:10.1080/01621459.2025.2606381). Also available is a 2-sample test using a Gaussian orthogonal ensemble approximation and an eigenvalue-multiplicity test that assumes orthogonally-invariant covariance.

Details

All tests in this package are for hypotheses about the eigenvalues of the (extrinsic) mean of the sampled population(s). Functions for conducting hypothesis tests start with `test_`. When the matrices are constrained to a submanifold of the space of symmetric matrices we use the extrinsic mean, which projects the usual linear/Euclidean mean so that it satisfies the same constraints as the data. Tests can be calibrated using either a chi-squared distribution or bootstrapping; in simulations bootstrapping was more reliable but slower.

The following functions conduct single sample and k-sample tests of the eigenvalues of the (extrinsic) population mean:

- `test_unconstrained()`
- `test_fixedtrace()` when matrices have fixed trace
- `test_ss1()` when the squared eigenvalues of each matrix sums to 1.
- `test_ss1fixedtrace()` when the squared eigenvalues of each matrix sums to 1 and the trace is zero.

The single sample tests conducted by the above functions test the null hypothesis of a user-provided set of eigenvalues for the extrinsic population mean against the alternative hypothesis that the extrinsic population mean has different eigenvalues. The k-sample tests conducted by the above functions test the null hypothesis that the extrinsic population means have the same eigenvalues.

Additionally `test_unconstrained_aGOE()` can perform 2-sample tests with calibration by a Gaussian Orthogonal Ensemble (GOE) approximation (Schwartzman et al. 2010). A bootstrapped calibration for this test is also available where the GOE approximation is used to stabilise the scale of the statistic.

There are two functions `conf_fixedtrace()` and `conf_ss1fixedtrace()` for estimating confidence regions.

The above tests all require that the eigenvalues of the population mean are distinct (with degraded performance when eigenvalues are very close to each other). Eigenvalues are assumed to be in descending order.

Use `test_multiplicity()` to test the eigenvalue-multiplicity of the population mean of a single sample. A test of the same hypothesis that requires that matrix elements follow a multivariate Gaussian distribution with orthogonally-invariant covariance is also available through `test_multiplicity_OI()` (Schwartzman et al. 2008). `test_multiplicity()` can also be applied to matrices with a constrained trace, but use `test_multiplicity_nonnegative()` for matrices constrained to have non-negative eigenvalues.

In this package, matrices within the same sample are considered independently and identically distributed. Matrices are stored in a flattened form as row-vectors according to `vech()` - see `fsm` for details. Samples may be provided as lists of matrices or in their flattened form so long as the column order matches that of `vech()`.

This package includes a vignette demonstrating an application to anisotropy of magnetic susceptibility data. For example applications of all the hypothesis tests in this package, please see the reproducibility document associated with (Hingee et al. 2026).

Acknowledgements

Colleagues Andrew T. A. Wood and Janice Scealy played crucial roles in developing the statistical concepts and theory. This package was written on Ngunnawal and Ngambri Country.

The package includes `scel.R` for empirical likelihood by Owen (2013), which is used to estimate optimal weights for weighted bootstrapping of samples of constrained matrices. The `scel.R` file was released in 2014-2015 under the under BSD-3-Clause with copyright by Board of Trustees, Leland Stanford Junior University

Author(s)

Maintainer: Kassel Liam Hingee <kassel.hingee@gmail.com> (ORCID)

Other contributors:

- Art B. Owen (./R/scel.R only) [copyright holder]
- Board of Trustees Leland Stanford Junior University (./R/scel.R only) [copyright holder]

References

Hingee KL, Scealy JL, Wood AT (2026). "Nonparametric bootstrap inference for the eigenvalues of geophysical tensors." *Journal of American Statistical Association*, 1-13. doi:10.1080/01621459.2025.2606381.

Owen AB (2013). "Self-concordance for empirical likelihood." *Canadian Journal of Statistics*, **41**(3), 387–397.

Schwartzman A, Dougherty RF, Taylor JE (2010). "Group Comparison of Eigenvalues and Eigenvectors of Diffusion Tensors." *Journal of the American Statistical Association*, **105**(490), 588–599. doi:10.1198/jasa.2010.ap07291.

Schwartzman A, Mascarenhas WF, Taylor JE (2008). "Inference for Eigenvalues and Eigenvectors of Gaussian Symmetric Matrices." *The Annals of Statistics*, **36**(6), 2886–2919. https://www.jstor.org/stable/25464736.

See Also

Useful links:

- https://github.com/kasselhingee/TFORGE

`boot_calib` *Bootstrap calibration for single-sample and k-sample tests*

Description

Performs a bootstrap hypothesis test using the supplied statistic. The `stdx` parameter is used to define an empirical distribution that satisfies the null hypothesis.

Usage

```
boot_calib(x, stdx, stat, B, ...)
```

Arguments

x	Symmetric matrix observations. Either a fsm or a kfsm.
stdx	Either a fsm or kfsm of matrices transformed to satisfy the null hypothesis OR sampling weights for each matrix in x for weighted bootstrap calibration (sampling weights should be optimised and also generate an empirical distribution that satisfies the null hypothesis).
stat	Function to compute the statistic.
B	The number of bootstrap samples to use.
...	Passed to stat

Details

The function stat is applied to x and all resamples, with the result returned in the t0 and nullt elements of the returned object, respectively.

Errors in evaluating stat on resamples are recorded in the nullt_messages and lead to NA values for the statistic in the nullt element of the returned object.

The p-value is the fraction of non-NA resample statistic values that are greater than stat applied to x.

Value

A list of

- pval the p-value from the test
- t0 the statistic for the observations x
- nullt The statistic evaluated on the resamples
- stdx The stdx passed into boot_calib()
- B The number of resamples requested
- nullt_messages Any error messages for the corresponding resample

The returned object has a bespoke class TFORGE for easy use of print().

chisq_calib *Chi Squared Calibration for Testing*

Description

Similar to `boot_calib()`, but uses chi-squared calibration instead of bootstrapping.

Usage

```
chisq_calib(x, stat, df, ...)
```

Arguments

`x`	Symmetric matrix observations. Either a `fsm` or a `kfsm`.
`stat`	Function to compute the statistic.
`df`	Degrees of freedom of the chi-squared distribution
`...`	Passed to `stat`

Value

A list of

- `pval` the p-value from the test
- `t0` the statistic for the observations `x`
- `df` The degrees of freedom of the chi-squared distribution

The returned object has class `TFORGE` (same as `boot_calib()`) for easy use of `print()`.

conf_fixedtrace *Eigenvalue confidence region under fixed trace constraint*

Description

When a 3x3 symmetric matrix has a fixed-trace constraint, the vector of its eigenvalues lies on a 2D plane. This function calculates the boundary of an approximate confidence region in this 2D plane using the same statistic as `test_fixedtrace()`. The returned boundary can be used to plot the confidence region. The function `conf_fixedtrace_inregion()` returns where given points are in the estimated confidence region.

Usage

```
conf_fixedtrace(x, alpha = 0.05, B = 1000, npts = 1000, check = TRUE)

conf_fixedtrace_inregion(evals, cr)
```

Arguments

x	A single sample of 3x3 symmetric matrices. x must be either an fsm object or something that as_fsm() can parse.
alpha	Desired significance level of the approximate confidence region.
B	Number of bootstrap resamples.
npts	Number of points on the boundary of the region to compute.
check	If TRUE, then the extrinsic means of 100 new resamples will be used to check the coverage of the region.
evals	A set of eigenvalues with the same trace as matrices in x.
cr	A confidence region returned by conf_fixedtrace().

Details

Uses the same statistic as test_fixedtrace() and bootstrap resampling to obtain approximate bounds on the eigenvalues of a population mean. The statistic has a quadratic form so that the boundary of the confidence region is an ellipse, but for plotting simplicity the ellipse is returned as a dense set of npts points. A warning will be generated if the confidence region leaves the space of distinct descending-order eigenvalues and a check of coverage of bootstrap resamples is available.

Value

A list:

- est: the eigenvalues of the mean matrix.
- boundary: A matrix with 3 columns and npts rows giving the boundary of the region. Each row corresponds to a point on the boundary and the columns are the first, second and final eigenvalue.
- Omega: The estimated covariance of the (projected) eigenvalues
- threshold: The threshold (estimated via resampling) on the statistic.

---

conf_ss1fixedtrace *Eigenvalue confidence interval under trace=0 and sum of square constraint*

Description

When a 3x3 symmetric matrix has a trace of zero and the sum of squared eigenvalues is one, then the eigenvalues of the matrix lie on a circle in 3D space. Under these situations, this function calculates a confidence region (i.e. an interval) for the eigenvalues of the population's extrinsic mean. The function conf_ss1fixedtrace_inregion() returns whether a set of eigenvalues is inside a confidence region returned by conf_fixedtrace().

Usage

```
conf_ss1fixedtrace(x, alpha = 0.05, B = 1000, check = TRUE)

conf_ss1fixedtrace_inregion(evals, cr)
```

Arguments

x	A single sample of 3x3 symmetric matrices. x must be either an fsm object or something that as_fsm() can parse.
alpha	Desired significance level of the approximate confidence region.
B	Number of bootstrap resamples.
check	If TRUE, then the extrinsic means of 100 new resamples will be used to check the coverage of the region.
evals	A set of eigenvalues with trace of zero and sum of squares of one.
cr	A confidence region returned by conf_ss1fixedtrace().

Value

A list:

- est: the eigenvalues of the mean matrix
- lower and upper: the two ends of the confidence interval
- Omega: The estimated covariance of the (projected) eigenvalues
- threshold: The threshold (estimated via resampling) on the statistic

---

cov_evals *Compute the Covariance of Eigenvalues*

Description

For a random symmetric matrix Y, calculates the covariance of the eigenvalues of Y using the covariance of the elements of Y and the eigenvectors of the mean of Y.

Usage

```
cov_evals(evecs, mcov)
```

Arguments

evecs	Matrix with columns that are eigenvectors of the mean of Y.
mcov	Covariance of vech(Y), where Y is the random matrix.

Details

For any two columns a and b of `evecs`, computes the covariance

$$\mathrm{Cov}(a^\top Y a, b^\top Y b) = (a \otimes a)^\top \mathbb{D} C_0 \mathbb{D}^\top (b \otimes b),$$

where a and b are the columns of `evecs` and C_0=`mcov` is the covariance of $\mathrm{vech}(Y)$. $\mathbb{D}$ and $\otimes$ is the duplication matrix and Kronecker product respectively.

The returned matrix has rows and columns that are in the same order as the columns of `evecs`.

When the eigenvalues are distinct, then passing estimated eigenvectors to `cov_evals()` yields an estimate of the asymptotic covariance of the eigenvalues.

See Supplement B.2 for more information and derivation.

Value

A symmetric matrix with same number of columns as `evecs`.

`estimate_OIcov` *Estimate parameters of orthogonally invariant covariance*

Description

Orthogonally-invariant covariance is a restrictive structure, but if it holds then a suite of tools is available (Schwartzman et al. 2008). Any orthogonally-invariant covariance can be specified by just two parameters τ and σ^2. For Gaussian-distributed elements, this function estimates the parameters τ and σ^2 by maximum-likelihood from the data and using a maximum-likelihood estimate of the population mean (Lemma 3.3, Schwartzman et al. 2008).

Usage

```
estimate_OIcov(x, Mhat, tau = NULL)
```

Arguments

`x`	A single sample of symmetric matrices. `x` must be either an `fsm` object or something that `as_fsm()` can parse.
`Mhat`	A maximum-likelihood estimate of the population mean
`tau`	The parameter τ. If supplied only σ^2 will be estimated.

Value

A named list of σ^2 and τ

Orthogonally-Invariant Covariance

A symmetric random matrix Y with a Gaussian distribution has orthogonally-invariant covariance if and only if QYQ^T has the same distribution as Y for any orthogonal matrix Q.

Using the parameterisation of τ and σ^2 by Schwartzman et al. (2008):

- the covariance of the off-diagonal elements of Y is $I\sigma^2/2$ where I is the identity matrix of the correct size.
- the covariance of the diagonal elements of Y is $\sigma^2(I + 11^T\tau/(1-\tau p))$ where p is the number of columns of Y and 1 is the vector of ones.
- the covariance between diagonal elements and non-diagonal elements is zero (i.e. they are independent).

References

Schwartzman A, Mascarenhas WF, Taylor JE (2008). "Inference for Eigenvalues and Eigenvectors of Gaussian Symmetric Matrices." *The Annals of Statistics*, **36**(6), 2886–2919. https://www.jstor.org/stable/25464736.

---

fsm *Flat storage of symmetric matrices*

Description

The `TFORGE_fsm` class, short for *Flat Symmetric Matrices*, is for storing a collection of symmetric matrices with each matrix stored as a row vector according to `vech()`. The `TFORGE_fsm` class is itself a thin wrapper of the array class. So, for example, `x[1, ]` will return the vectorised-form of the first matrix in the collection, and `inv_vech(x[1,])` will be the first matrix in non-flat form. The `TFORGE_kfsm` class is for a collection of multiple `TFORGE_fsm`. The function `as_flat()` automatically converts data to either `TFORGE_kfsm` or `TFORGE_fsm`.

Usage

```
as_flat(x, ...)

as_kfsm(x, ...)

as_fsm(x, ...)
```

Arguments

`x`	For `as_fsm()` a list of symmetric matrices, or a 2D array of flattened matrices. For `as_kfsm()` a list of objects that can be passed to `as_fsm()` (i.e. a list of a lists of matrices, or a list of 2D arrays). For `as_flat()`, `x` can be suitable for either `as_fsm()` or `as_kfsm()`.
`...`	Passed to `isSymmetric()` for testing whether matrices are symmetric.

Details

The matrices inside x must all have the same dimension.

The function as_flat() automatically chooses between a TFORGE_kfsm or a TFORGE_fsm:

- If x is a list of symmetric matrices then it will return a TFORGE_fsm.
- If x is a list of lists of equal-sized matrices then it returns a TFORGE_kfsm, with each element of the larger list a TFORGE_fsm.
- If x is a list of 2D arrays, each satisfying as_fsm(), then as_flat() will return a TFORGE_kfsm.
- In the rare case that x is a list of 2D arrays of flattened matrices, but the 2D arrays happen to be perfectly symmetric (requires size of collections to perfectly relate to the dimension of the matrix observations) then as_flat() will mistakenly treat each element of x as a symmetric matrix and return a TFORGE_fsm.

Value

An object with class TFORGE_kfsm or TFORGE_fsm.

Functions

- as_flat(): Automatically convert to either a TFORGE_kfsm or a TFORGE_fsm.
- as_kfsm(): Convert multiple collections of matrices into a kfsm. x must be a list, with each entry of x a separate collection of matrices.
- as_fsm(): For x a list of symmetric matrices of the same size, flattens x into a 2D array where the ith row is a vectorised version vech(x[[i]]) of the ith matrix of x. If x is already flattened then as_fsm() will check that the number of columns are consistent with a flattened symmetric matrix.

Examples

```
x <- list(list(matrix(c(1,2,3,2,4,5,3,5,6), 3),
               matrix(c(2,3,4,3,5,6,4,6,7), 3)),
          list(matrix(c(0.1,0.2,0.3,0.2,0.4,0.5,0.3,0.5,0.6), 3),
               matrix(c(0.2,0.3,0.4,0.3,0.5,0.6,0.4,0.6,0.7), 3)))
as_kfsm(x)
summary(as_flat(x))
```

Gonjo *AMS data of the Gonjo Basin*

Description

This is the anisotropy of magnetic susceptibility (AMS) data from a 3km thick section of redbeds in the Gonjo Basin in eastern Tibet that was analysed by Li et al. (2020).

Usage

```
Gonjo
```

Format

A list with entry `datatable` containing one row per specimen and entry `matrices` containing the AMS tensor (i.e. symmetric matrix) for each specimen. The `datatable` entry has 542 rows and 25 variables:

- `Name`: character — Specimen name
- `really depth`: double — Depth of specimen (in meters)
- `Field`: double — Unsure — Li et al. (2020) say they applied a 300 A/m magnetic field
- `Freq.`: double — Frequency of oscillation of the applied magnetic field
- `Km`: double — Mean magnetic susceptibility
- `L`: double — Lineation (λ_1/λ_2)
- `F`: double — Foliation (λ_2/λ_3)
- `P`: double — Uncorrected degree of anisotropy (λ_1/λ_3)
- `Pj`: double — Corrected degree of anisotropy
- `T`: double — Shape factor
- `U`: double — *Unsure*
- `Q`: double — *Unsure*
- `E`: double — *Unsure*
- `K1decI` and `K1incI`: doubles — In-situ direction of first eigenvector
- `K2decI` and `K2incI`: doubles — In-situ direction of second eigenvector
- `K3decI` and `K3incI`: doubles — In-situ direction of third eigenvector
- `K1decT` and `K1incT`: doubles — Tilt-corrected direction of first eigenvector
- `K2decT` and `K2incT`: doubles — Tilt-corrected of second eigenvector
- `K3decT` and `K3incT`: doubles — Tilt-corrected of third eigenvector

Details

The AMS matrices were calculated using the in-situ directions by Dr. Janice Scealy.

Source

doi:10.5281/zenodo.3666760

References

Li S, van Hinsbergen DJJ, Shen Z, Najman Y, Deng C, Zhu R (2020). "Anisotropy of Magnetic Susceptibility (AMS) Analysis of the Gonjo Basin as an Independent Constraint to Date Tibetan Shortening Pulses." *Geophysical Research Letters*, **47**(8), e2020GL087531. doi:10.1029/2020GL087531.

has_fixedtrace *Check if the supplied sample(s) have fixed trace*

Description

Compares the trace of all the supplied matrices to check that they are equal.

Usage

```
has_fixedtrace(x, tolerance = sqrt(.Machine$double.eps))
```

Arguments

x	A sample or multiple samples of matrices suitable for as_flat().
tolerance	Tolerance on the relative difference, passed to all.equal()

Value

TRUE or FALSE

has_ss1 *Check whether the supplied sample(s) have equal sum of squared eigenvalues*

Description

Compares whether the sum of the squared eigenvalues of the supplied symmetric matrices match each other using the property that the sum of the squared eigenvalues of Y equals the trace of Y %*% Y.

Usage

```
has_ss1(x, tolerance = sqrt(.Machine$double.eps))
```

Arguments

x	A sample or multiple samples of matrices suitable for as_flat().
tolerance	Tolerance on the relative difference, passed to all.equal()

Value

TRUE or FALSE

normalise_ss1 *Normalise so that Sum of Squared Eigenvalues is One*

Description

Scales symmetric tensors so that the square of the eigenvalues sum to one.

Usage

```
normalise_ss1(x)
```

Arguments

x	A sample of matrices suitable for `as_fsm()`.

Value

A `TFORGE_fsm` object.

normalise_trace *Scale symmetric matrices to have trace of one*

Description

Scales symmetric matrices by their trace, so that resulting matrices have a trace of one.

Usage

```
normalise_trace(x)

normalize_trace(x)
```

Arguments

x	A sample of matrices suitable for `as_fsm()`.

Details

The method will create `Inf` values for tensors that have a trace of zero.

Value

A set of flattened symmetric matrices (i.e. `TFORGE_fsm` class).

project_trace *Project diagonal elements to have trace of zero*

Description

Projects the diagonal elements of symmetric matrices onto the plane through the origin and orthogonal to the vector $(1, 1, 1, ..., 1)^\top$. The trace of the resulting symmetric matrices is zero.

Usage

```
project_trace(x)
```

Arguments

x	A sample of matrices suitable for as_fsm().

Value

A set of flattened symmetric matrices (i.e. TFORGE_fsm class)

rsymm_norm *Simulate Symmetric Matrices with Multivariate Normal Elements*

Description

Simulate symmetric matrices with elements from a multivariate Normal distribution.

Usage

```
rsymm_norm(n, mean, sigma = diag(length(vech(mean))))

rsymm(n, mean, sigma = diag(length(vech(mean))))
```

Arguments

n	Number of matrices to generate
mean	A symmetric matrix specifying the mean of the distribution.
sigma	A covariance matrix for the vectorised lower triangular elements (arranged by vech()) of the symmetric matrix. It is passed to mvtnorm::rmvnorm() without any transformation.

Details

The mean matrix is vectorised using the vech() function and then used as the mean vector in the mvtnorm::rmvnorm() function. The covariance matrix sigma is passed unchanged to mvtnorm::rmvnorm(). Symmetric matrices can be obtained by applying inv_vech() to each simulated vector.

Value

A set of flattened symmetric matrices as a TFORGE_fsm object. See as_fsm().

Examples

```
rsymm_norm(100, diag(c(3,2,1)))
```

---

rsymm_t *Simulate Symmetric Matrices with Multivariate t Elements*

Description

Simulate symmetric matrices with elements from a multivariate t distribution.

Usage

```
rsymm_t(n, mean, df = 1, sigma = diag(length(vech(mean))))
```

Arguments

n	Number of matrices to generate.
mean	A symmetric matrix specifying the mean of the distribution.
df	Degrees of freedom for the t distribution.
sigma	The scale parameter matrix for the elements arranged by vech(). sigma is passed to mvtnorm::rmvt() without any transformation.

Details

The function uses Representation A in (Lin 1972) to simulate multivariate-t vectors. The mean matrix is vectorised using the vech() function and then used as the mean vector in the mvtnorm::rmvt() function. The scale parameter matrix sigma is passed unchanged to mvtnorm::rmvt(). The covariance of the resulting vectors is sigma * df / (df - 2). Symmetric matrices can be obtained by applying inv_vech() to each simulated vector.

Value

A set of flattened symmetric matrices as a TFORGE_fsm object. See as_fsm().

References

Lin P (1972). "Some characterizations of the multivariate t distribution." *Journal of Multivariate Analysis*, **2**(3), 339–344. doi:10.1016/0047259X(72)900218.

Examples

```
rsymm_t(100, mean = matrix(1, nrow = 3, ncol = 3), df = 10, sigma = diag(c(3,2,1,1,1,1)))
```

`test_fixedtrace` *Test for eigenvalues when trace is fixed*

Description

For a single sample of symmetric matrices with fixed trace, test eigenvalues of the population mean. For multiple samples of symmetric matrices with fixed trace, test for equality of the eigenvalues of the population means. The test statistic is calculated by `stat_fixedtrace()`.

Usage

```
test_fixedtrace(x, evals = NULL, B, maxit = 25)

stat_fixedtrace(x, evals = NULL)
```

Arguments

`x`	A single sample of symmetric matrices or multiple samples of symmetric matrices. See `as_flat()`.
`evals`	When `x` is a single sample, the null hypothesis is that the (extrinsic) mean of the population has eigenvalues equal to `evals`. For multiple samples `evals` must be omitted.
`B`	Number of bootstrap samples. If `B = 'chisq'` then a chi-squared calibration is used instead.
`maxit`	The maximum number of Newton steps allowed in empirical likelihood optimisation (Owen 2013).

Details

Test hypotheses described below. The fixed trace constraint forces the vector of eigenvalues to lie in a plane. The test statistic accounts for this constraint by using an orthonormal basis in the plane. Weighted bootstrap calibration is used (see 'Weighted Bootstrapping' below).

Eigenvalues must be distinct.

The test statistic is calculated by `stat_fixedtrace()`.

Value

A `TFORGE` object (see `boot_calib()` or `chisq_calib()`) with the eigenvalues of the null hypothesis in the `null_evals` attribute for `t0`.

Weighted Bootstrapping

This function uses a form of weighted bootstrapping called b-boostrapping (Hall and Presnell 1999). An empirical distribution is defined by sampling weights for each observation in the original sample. The sampling weights must be such that the (extrinsic) mean of the empirical distribution is

$$c\hat{Q}\Lambda\hat{Q}^\top,$$

where $\hat{Q}$ are the eigenvectors of the sample mean, Λ is a diagonal matrix of eigenvalues specified by either the null hypothesis (for single sample tests) or estimated as the common eigenvalues of multiple populations (for k-sample tests). In some situations c is a free scalar to enable projection of the Euclidean mean to the extrinsic mean, otherwise $c = 1$. If no such sampling weights exist (i.e. the convex hull of the data does not contain $c\hat{Q}\Lambda\hat{Q}^\top$), then the test rejects with `pval=0` and a warning.

The sampling weights are also optimised to maximise empirical likelihood (Owen 2013).

Hypotheses

For a single sample the null hypothesis is that the population (extrinsic) mean has eigenvalues of `evals`; the alternative hypothesis is that the eigenvalues are not equal to `evals`. For multiple samples, `evals` must be omitted and the null hypothesis is that the population (extrinsic) means have the same eigenvalues.

References

Hall P, Presnell B (1999). "Intentionally Biased Bootstrap Methods." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **61**(1), 143–158. ISSN 1369-7412, 2680742, https://www.jstor.org/stable/2680742.

Owen AB (2013). "Self-concordance for empirical likelihood." *Canadian Journal of Statistics*, **41**(3), 387–397.

`test_multiplicity` *Test eigenvalue multiplicity*

Description

Tests the multiplicity of the eigenvalues a population's mean. The test statistic is computed by `stat_multiplicity()`. The null hypothesis is that the population mean has the specified the multiplicity of eigenvalues. For unconstrained symmetric matrices or symmetric matrices with fixed trace use `test_multiplicity()`. For matrices constrained to have non-negative eigenvalues use `test_multiplicity_nonnegative()`.

Usage

```
test_multiplicity(x, mult, B = 1000, refbasis = "sample")

test_multiplicity_nonnegative(
  x,
  mult,
  B = 1000,
  maxit = 25,
  refbasis = "sample"
)
```

```
stat_multiplicity(x, mult, evecs = NULL, refbasis = "sample")

translate2multiplicity(x, mult)
```

Arguments

x	A sample of matrices suitable for as_fsm().
mult	A vector specifying the eigenvalue multiplicity under the null hypothesis in descending order of eigenvalue size.
B	Number of bootstrap samples. If B = 'chisq' then a chi-squared calibration is used instead.
refbasis	Select the basis of the eigenspaces. See details.
maxit	The maximum number of Newton steps allowed in empirical likelihood optimisation (Owen 2013).
evecs	For debugging only. Supply eigenvectors of population mean.

Details

For test_multiplicity(), bootstrap resampling is conducted from the null hypothesis by first translating the original sample to satisfy the null hypothesis with translate2multiplicity(). For test_multiplicity_nonnegative(), weighted bootstrapping is used (see 'Weighted Bootstrapping' below).

On refbasis: An estimate of each eigenspace specified by mult can be obtained from the eigenvectors of the sample mean. The eigenvectors create an orthonormal basis of the (estimated) eigenspace, however the choice of orthonormal basis for the estimated eigenspace effects the performance. This choice is specified by the parameter refbasis. Setting refbasis = "sample" uses the eigenvectors of the sample mean as the basis, however the resulting statistic does not appear to be pivotal. Choosing the orthonormal basis independently of the data does result in a pivotal asymptotically chi-squared statistic. Setting refbasis = "random" will do exactly this, by applying a uniformly random rotation of the relevant eigenvectors of the sample mean. We recommend using refbasis = "sample" (which requires bootstrap calibration) because test power is much higher than refbasis = "random". We recommend that the number bootstrap resamples is at least 1000 if refbasis = "sample".

For 3x3 matrices, the weighted-bootstrapping method used by test_multiplicity_nonnegative() has poor test size for samples smaller than 20; larger matrices will likely need larger samples.

Due to the random rotation of the eigenvectors when refbasis = "random", use set.seed() if you want the answer to be repeatable.

Value

A TFORGE object (see boot_calib() or chisq_calib()) including p-value of the test (slot pval) and the statistic for x (slot t0).

Weighted Bootstrapping

This function uses a form of weighted bootstrapping called b-boostrapping (Hall and Presnell 1999). An empirical distribution is defined by sampling weights for each observation in the original sample.

The sampling weights must be such that the (extrinsic) mean of the empirical distribution is

$$c\hat{Q}\Lambda\hat{Q}^\top,$$

where $\hat{Q}$ are the eigenvectors of the sample mean, Λ is a diagonal matrix of eigenvalues specified by either the null hypothesis (for single sample tests) or estimated as the common eigenvalues of multiple populations (for k-sample tests). In some situations c is a free scalar to enable projection of the Euclidean mean to the extrinsic mean, otherwise $c = 1$. If no such sampling weights exist (i.e. the convex hull of the data does not contain $c\hat{Q}\Lambda\hat{Q}^\top$), then the test rejects with `pval=0` and a warning.

The sampling weights are also optimised to maximise empirical likelihood (Owen 2013).

Examples

```
x <- rsymm_norm(15, mean = diag(c(2, 1, 1, 0)))
test_multiplicity(x, mult = c(1, 2, 1))
```

---

test_multiplicity_OI *Test of eigenvalue multiplicity assuming orthogonally invariant covariance*

Description

Given a sample from a population of symmetric matrices with Gaussian-distributed elements and orthogonally-invariant covariance, corollary 4.3 by Schwartzman et al. (2008) provides a method to test the eigenvalue multiplicity of the mean matrix. Orthogonally-invariant covariance is a strong assumption and may not be valid; consider using `test_multiplicity()` if you are unsure.

Usage

```
test_multiplicity_OI(x, mult, B = "chisq", refbasis = NULL)
```

Arguments

`x`	A sample of matrices suitable for `as_fsm()`.
`mult`	A vector specifying the eigenvalue multiplicity under the null hypothesis in descending order of eigenvalue size.
`B`	Number of bootstrap samples. If `B = 'chisq'` then a chi-squared calibration is used instead.
`refbasis`	Ignored (for compatibility with `test_multiplicity()`).

Details

The orthogonally invariant covariance matrix is estimated by `estimate_OIcov()`. The maximum-likelihood estimate of the population mean under the null hypothesis is computed according to (Theorem 4.2, Schwartzman et al. 2008).

Value

A TFORGE object (see boot_calib() or chisq_calib()) including p-value of the test (slot pval) and the statistic for x (slot t0).

---

test_ss1 *Test for eigenvalues when sum of squared eigenvalues is 1*

Description

For a single sample of symmetric matrices where sum of squared eigenvalues = 1, test eigenvalues of the population mean. For multiple samples of symmetric matrices where sum of squared eigenvalues = 1, test for equality of the eigenvalues of the population means. The test statistic is calculated by stat_ss1().

Usage

```
test_ss1(x, evals = NULL, B = 1000, maxit = 25)

stat_ss1(x, evals = NULL)
```

Arguments

x	A single sample of symmetric matrices or multiple samples of symmetric matrices. See as_flat().
evals	When x is a single sample, the null hypothesis is that the (extrinsic) mean of the population has eigenvalues equal to evals. For multiple samples evals must be omitted.
B	Number of bootstrap samples. If B = 'chisq' then a chi-squared calibration is used instead.
maxit	The maximum number of Newton steps allowed in empirical likelihood optimisation (Owen 2013).

Details

Test hypotheses described below. The sum of squared eigenvalues constraint forces the set of eigenvalues to lie on a sphere (or circle). The test statistic accounts for this constraint by projecting eigenvalues onto a plane perpendicular to the direction of the sample average's eigenvalues.

Weighted bootstrap calibration is used (see 'Weighted Bootstrapping' below).

Eigenvalues must be distinct.

Value

A TFORGE object (see boot_calib() or chisq_calib()) with the eigenvalues of the null hypothesis in the null_evals attribute for t0.

Hypotheses

For a single sample the null hypothesis is that the population (extrinsic) mean has eigenvalues of `evals`; the alternative hypothesis is that the eigenvalues are not equal to `evals`. For multiple samples, `evals` must be omitted and the null hypothesis is that the population (extrinsic) means have the same eigenvalues.

Weighted Bootstrapping

This function uses a form of weighted bootstrapping called b-boostrapping (Hall and Presnell 1999). An empirical distribution is defined by sampling weights for each observation in the original sample. The sampling weights must be such that the (extrinsic) mean of the empirical distribution is

$$c\hat{Q}\Lambda\hat{Q}^\top,$$

where $\hat{Q}$ are the eigenvectors of the sample mean, Λ is a diagonal matrix of eigenvalues specified by either the null hypothesis (for single sample tests) or estimated as the common eigenvalues of multiple populations (for k-sample tests). In some situations c is a free scalar to enable projection of the Euclidean mean to the extrinsic mean, otherwise $c = 1$. If no such sampling weights exist (i.e. the convex hull of the data does not contain $c\hat{Q}\Lambda\hat{Q}^\top$), then the test rejects with `pval=0` and a warning.

The sampling weights are also optimised to maximise empirical likelihood (Owen 2013).

References

Hall P, Presnell B (1999). "Intentionally Biased Bootstrap Methods." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **61**(1), 143–158. ISSN 1369-7412, 2680742, https://www.jstor.org/stable/2680742.

Owen AB (2013). "Self-concordance for empirical likelihood." *Canadian Journal of Statistics*, **41**(3), 387–397.

`test_ss1fixedtrace` *Test eigenvalues when trace=0 and sum of square eigenvalues = 1*

Description

For a single sample, test eigenvalues of the population mean. For multiple samples, test for equality of the eigenvalues of the population means. This function is for 3x3 symmetric matrices with trace of zero and sum of squared eigenvalues of one. These constraints combine so that the space of possible sets of (ordered) eigenvalues is a circle. The test statistic is calculated by `stat_ss1fixedtrace()`.

Usage

```
test_ss1fixedtrace(x, evals = NULL, B = 1000, maxit = 25)

stat_ss1fixedtrace(x, evals = NULL)
```

Arguments

x	A single sample of symmetric matrices or multiple samples of symmetric matrices. See as_flat().
evals	When x is a single sample, the null hypothesis is that the (extrinsic) mean of the population has eigenvalues equal to evals. For multiple samples evals must be omitted.
B	Number of bootstrap samples. If B = 'chisq' then a chi-squared calibration is used instead.
maxit	The maximum number of Newton steps allowed in empirical likelihood optimisation (Owen 2013).

Details

Test hypotheses described below.

The sum of squared eigenvalues constraint forces the set of eigenvalues to lie on a sphere and the trace constraint forces eigenvalues onto a plane. Combined the constraints force eigenvalues onto a circle in 3D Euclidean space. The test statistic accounts for these constraints by projecting eigenvalues onto a line tangential to this circle and orthogonal to the null-hypothesis eigenvalues.

Weighted bootstrap calibration is used (see 'Weighted Bootstrapping' below).

Eigenvalues must be distinct.

Value

A TFORGE object (see boot_calib() or chisq_calib()) with the eigenvalues of the null hypothesis in the null_evals attribute for t0.

Hypotheses

For a single sample the null hypothesis is that the population (extrinsic) mean has eigenvalues of evals; the alternative hypothesis is that the eigenvalues are not equal to evals. For multiple samples, evals must be omitted and the null hypothesis is that the population (extrinsic) means have the same eigenvalues.

Weighted Bootstrapping

This function uses a form of weighted bootstrapping called b-boostrapping (Hall and Presnell 1999). An empirical distribution is defined by sampling weights for each observation in the original sample. The sampling weights must be such that the (extrinsic) mean of the empirical distribution is

$$c\hat{Q}\Lambda\hat{Q}^\top,$$

where $\hat{Q}$ are the eigenvectors of the sample mean, Λ is a diagonal matrix of eigenvalues specified by either the null hypothesis (for single sample tests) or estimated as the common eigenvalues of multiple populations (for k-sample tests). In some situations c is a free scalar to enable projection of the Euclidean mean to the extrinsic mean, otherwise $c = 1$. If no such sampling weights exist (i.e. the convex hull of the data does not contain $c\hat{Q}\Lambda\hat{Q}^\top$), then the test rejects with pval=0 and a warning.

The sampling weights are also optimised to maximise empirical likelihood (Owen 2013).

References

Hall P, Presnell B (1999). "Intentionally Biased Bootstrap Methods." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **61**(1), 143–158. ISSN 1369-7412, 2680742, https://www.jstor.org/stable/2680742.

Owen AB (2013). "Self-concordance for empirical likelihood." *Canadian Journal of Statistics*, **41**(3), 387–397.

`test_unconstrained` *Pivotal bootstrap test of mean eigenvalues*

Description

For a single sample of symmetric matrices, test eigenvalues of the population mean. For multiple samples of symmetric matrices, test for equality of the eigenvalues of the population means. Eigenvalues must be distinct.

Usage

```
test_unconstrained(x, evals = NULL, evecs = NULL, B = 1000)

stat_unconstrained(x, evals = NULL, evecs = NULL)

translate_evalsofav(x, evals)
```

Arguments

`x`	A single sample of symmetric matrices or multiple samples of symmetric matrices. See `as_flat()`.
`evals`	When `x` is a single sample, the null hypothesis is that the (extrinsic) mean of the population has eigenvalues equal to `evals`. For multiple samples `evals` must be omitted.
`evecs`	For a single sample, specify eigenvectors to test under the assumption that the population mean's eigenvectors are the columns of `evecs`. The order of these eigenvectors matters and should be such that eigenvalues are in descending order.
`B`	Number of bootstrap samples. If `B = 'chisq'` then a chi-squared calibration is used instead.

Details

Test hypotheses described below. For a single sample, the eigenvectors of the population mean in the null and alternative hypotheses may be prespecified by `evecs`.

Bootstrap resampling is conducted from a population that satisfies the null hypothesis by translating each sample in `x` with `translate_evalsofav()` to so that the sample average has the null eigenvalues. The test statistic is calculated by `stat_unconstrained()`.

Value

A TFORGE object (see boot_calib() or chisq_calib()) with the eigenvalues of the null hypothesis in the null_evals attribute for t0.

Hypotheses

For a single sample the null hypothesis is that the population (extrinsic) mean has eigenvalues of evals; the alternative hypothesis is that the eigenvalues are not equal to evals. For multiple samples, evals must be omitted and the null hypothesis is that the population (extrinsic) means have the same eigenvalues.

Examples

```
test_unconstrained(rsymm_norm(15, diag(c(3,2,1))), evals = c(3, 2, 1), B = 100)
test_unconstrained(list(rsymm_norm(15, diag(c(3,2,1))),
                        rsymm_norm(15, diag(c(3,2,1)))), B = 100)
```

---

test_unconstrained_aGOE

Two Sample Test of Equal Eigenvalues Using GOE Approximation

Description

Applies the equal-eigenvalue hypothesis test between two samples by Schwartzman et al. (2010). The null hypothesis is that the population means of each sample have the same eigenvalues, regardless of eigenvectors. The test uses a statistic from the situation that both populations are Gaussian Orthogonal Ensembles (Gaussian-distributed independent elements with the variance on the off diagonal elements half that of the diagonal elements). The distribution of this statistic for more general populations is approximated using a tangent space and the Welch-Satterthwaite approximation.

Usage

```
test_unconstrained_aGOE(
  x,
  x2 = NULL,
  B = "chisq",
  nullevals = "av",
  scalestat = TRUE
)
```

Arguments

x	A single sample of matrices (passed to as_fsm()) or a list of two samples of matrices (passed to as_kfsm()).
x2	If x is a single sample then x2 must be the second sample. Otherwise x2 should be NULL.

B	Number of bootstrap samples. If B = 'chisq' then a chi-squared calibration is used instead.
nullevals	For internal testing of bootstrap calibration. "av" assumes the eigenvalues under the null hypothesis are the average of the eigenvalues of the two sample means. "1" and "2" assume the null eigenvalues are equal to the eigenvalue of the first and second sample respectively.
scalestat	If TRUE then the statistic is divided by the estimated a. This modified statistic has approximately the same scale regardless of the data, although the thickness of the distribution tails (related to v) will vary. Simulations and bootstrapping behaviour more generally suggests that scalestat=TRUE leads to better test size and power when using bootstrap calibration.

Details

The test statistic is equation 11 of (Schwartzman et al. 2010). For chi-squared calibration, the p value of the test is computed using the scaled chi-squared distribution reached at the end of (Section 2.4, Schwartzman et al. 2010). This distribution approximates the distribution of the test statistic and the scale a and degrees of freedom v are estimated from the data.

Value

A TFORGE object (see boot_calib() or chisq_calib()) including p-value of the test (slot pval) and the statistic for x (slot t0). The returned object contains further slots specific to this test:

- a Plug-in estimate of the a in the final equation of (Section 2.4, Schwartzman et al. 2010).
- v Plug-in estimate of the v in the final equation of (Section 2.4, Schwartzman et al. 2010).
- var_Lambda_evals The variance of the eigenvalues of Schwartzman et al Λ matrix, which may relate to the quality of the Welch-Satterthwaite approximation.

References

Schwartzman A, Dougherty RF, Taylor JE (2010). "Group Comparison of Eigenvalues and Eigenvectors of Diffusion Tensors." *Journal of the American Statistical Association*, **105**(490), 588–599. doi:10.1198/jasa.2010.ap07291.

---

vecd *Flatten a symmetric matrix into a vector preserving Frobenius norm.*

Description

The vecd operator as used by Schwartzman et al. (2008) flattens a symmetric matrix into a vector of unique elements such that the Frobenius norm of the matrix equals the Euclidean norm of the vector. This means that the off-diagonal elements are scaled by $\sqrt{2}$. In the returned vector the diagonal elements are first then the (scaled) off-diagonal elements; this ordering is different to vech().

Usage

```
vecd(m)
```

Arguments

m	A symmetric matrix

Details

The vecd() function has a single line of code:

```
c(diag(m), sqrt(2) * m[lower.tri(m, diag = FALSE)])
```

The matrix m is not checked for symmetry.

Value

A vector.

References

Schwartzman A, Mascarenhas WF, Taylor JE (2008). "Inference for Eigenvalues and Eigenvectors of Gaussian Symmetric Matrices." *The Annals of Statistics*, **36**(6), 2886–2919. https://www.jstor.org/stable/25464736.

Examples

```
m <- inv_vech(1:6)
vecd(m)
```

---

vech *Flatten a symmetric matrix into a vector.*

Description

The vech operator as defined by (Section 3.8 Magnus and Neudecker 2019) flattens symmetric matrices into vectors. Columns are extracted from left to right with entries above the diagonal ignored. The function inv_vech() is the inverse of vech(). The dimension of the matrix can be obtained from its flattened form by dim_vech().

Usage

```
vech(m, name = FALSE)

inv_vech(x)

dim_vech(x)
```

Arguments

m	A symmetric matrix
name	If TRUE vector elements are named eij where i is the row and j is the column.
x	A flattened symmetric matrix (as a vector).

Details

The extraction is conveniently performed by m[lower.tri(m), diag = TRUE]. The matrix m is not checked for symmetry.

Value

vech() returns a vector. inv_vech() returns a matrix. dim_vech() returns an integer.

References

Magnus JR, Neudecker H (2019). *Matrix Differential Calculus with Applications in Statistics and Econometrics, 3rd Edition*, 3 edition. John Wiley and Sons Ltd. ISBN 978-1-119-54120-2, https://learning.oreilly.com/library/view/matrix-differential-calculus/9781119541202/.

Examples

```
m <- inv_vech(1:6)
dim_vech(1:6)
vech(m)
```

Index