# Package 'wwntests'

November 1, 2022

**Type** Package

**Title** Hypothesis Tests for Functional Time Series

**Version** 1.0.2

**Maintainer** Mihyun Kim <mihyun.kim@mail.wvu.edu>

**Description**

Provides an array of white noise hypothesis tests for functional data and related visualizations. These include tests based on the norms of autocovariance operators that are built under both strong and weak
white noise assumptions. Additionally, tests based on the spectral density operator and on principal component
dimensional reduction are included, which are built under strong white noise assumptions. These methods are
described in Kokoszka et al. (2017) <doi:10.1016/j.jmva.2017.08.004>, Characiejus and Rice (2019)
<doi:10.1016/j.ecosta.2019.01.003>, and Gabrys and Kokoszka (2007) <doi:10.1198/016214507000001111>, respectively.

**License** GPL-3

**Encoding** UTF-8

**Depends** R (>= 3.4.0)

**Imports** sde, stats, ftsa, rainbow, MASS, graphics

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, CompQuadForm, tensorA

**RoxygenNote** 7.2.1.9000

**VignetteBuilder** knitr

**BugReports** https://github.com/veritasmih/wwntests/issues

**NeedsCompilation** no

**Config/testthat/edition** 3

**Author** Mihyun Kim [aut, cre],
Daniel Petoukhov [aut]

**Repository** CRAN

**Date/Publication** 2022-11-01 15:10:02 UTC

# R topics documented:



---

`autocorrelation_coeff_h` *'autocorrelation_coeff_h' Computes the approximate functional autocorrelation coefficient at a given lag.*

---

## Description

'autocorrelation_coeff_h' Computes the approximate functional autocorrelation coefficient at a given lag.

## Usage

```
autocorrelation_coeff_h(f_data, lag)
```

### Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| lag | the lag to use to compute the single lag test statistic |

### Value

numeric value; the approximate functional autocorrelation coefficient at lag h.

---

`autocorrelation_coeff_plot` *Plot Confidence Bounds of Estimated Functional Autocorrelation Coefficients*

---

### Description

'autocorrelation_coeff_plot' Computes the 1-alpha upper confidence bounds for the functional autocorrelation coefficients at lags h = 1:K under both weak white noise (WWN) and strong white noise (SWN) assumptions. It plots the coefficients as well as the bounds for all lags h = 1:K. Note, the SWN bound is constant, while the WWN is dependent on the lag.

### Usage

```
autocorrelation_coeff_plot(
  f_data,
  K = 20,
  alpha = 0.05,
  M = NULL,
  wwn_bound = TRUE
)
```

### Arguments

| | |
|---|---|
| f_data | The functional data matrix with observed functions in the columns. |
| K | A positive Integer value. The maximum lag for which to compute the single-lag test (tests will be computed for lags h in 1:K). |
| alpha | A numeric value between 0 and 1 specifying the significance level to be used in the single-lag test. The default value is 0.05. |
| M | A positive Integer value. Determines the number of Monte-Carlo simulations employed in the Welch-Satterthwaite approximation of the limiting distribution of the test statistics, for each test. |
| wwn_bound | A Boolean value allowing the user to turn off the weak white noise bound. TRUE by default. Speeds up computation when FALSE. |

## Details

This function computes and plots autocorrelation coefficients at lag h, for h in 1:K. It also computes an estimated asymptotic 1 - alpha confidence bound, under the assumption that the series forms a weak white noise. Additionally, it computes a similar (constant) bound under the assumption the series form a strong white noise. Please see the vignette or the references for a more complete treatment.

## Value

Plot of the estimated autocorrelation coefficients for lags h in 1:K with the weak white noise 1-alpha upper confidence bound for each lag, as well as the constant strong white noise 1-alpha confidence bound.

## References

[1] Kokoszka P., & Rice G., & Shang H.L. (2017). Inference for the autocovariance of a functional time series under conditional heteroscedasticity. Journal of Multivariate Analysis, 162, 32-50.

## Examples

```
b <- brown_motion(75, 40)
autocorrelation_coeff_plot(b)
autocorrelation_coeff_plot(b, M = 200)
```

---

`autocov_approx_h` *Compute the approximate autocovariance at specified lag*

---

## Description

'autocov_approx_h' Computes the approximate autocovariance for a given lag h of the functional data

## Usage

```
autocov_approx_h(f_data, lag)
```

## Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `lag` | the lag to use to compute the single lag test statistic |

## Value

A 2-dimensional array encoding the autocovariance matrix for a given lag h.

`bartlett_kernel` *Bartlett Kernel Function*

## Description

'bartlett_kernel' Computes the Bartlett kernel function at a given point value.

## Usage

```
bartlett_kernel(x)
```

## Arguments

| | |
|---|---|
| `x` | the point value at which the kernel function is evaluated |

## Value

A scalar value; the value of the Bartlett kernel function at the point value x.

---

`block_bootsrap` *'block_bootstrap' Performs a block bootstrap on the functional data f_data with block size b.*

## Description

'block_bootstrap' Performs a block bootstrap on the functional data f_data with block size b.

## Usage

```
block_bootsrap(f_data, b, B = 300, moving = FALSE)
```

## Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `b` | the block size (of each block in each bootstrap sample) |
| `B` | the number of bootstraps samples |
| `moving` | boolean value specifying whether the block bootstrap should be moving or not. A moving black bootstrap samples individual functional observations and adds on the consequent block, rather than sampling blocks of the data. |

## Value

Returns a list of B elements, each element being a block bootstrap sample in the same format as the original functional data f_data.

`brown_motion` *'brown_motion' Creates at J x N matrix, containing N independent Brownian motion sample paths in each of the columns.*

## Description

'brown_motion' Creates at J x N matrix, containing N independent Brownian motion sample paths in each of the columns.

## Usage

```
brown_motion(N, J)
```

## Arguments

| | |
|---|---|
| `N` | the number of independent Brownian motion sample paths to compute. |
| `J` | the number of steps observed for each sample path (the resolution of the data). |

## Value

A J x N matrix containing Brownian motion functional data in the columns.

## Examples

```
b <- brown_motion(250, 50)
```

`B_h_bound` *Compute weak white noise confidence bound for autocorrelation coefficient.*

## Description

'B_h_bound' Computes an approximate asymptotic upper 1-alpha confidence bound for the functional autocorrelation coefficient at lag h under a weak white noise assumption.

## Usage

```
B_h_bound(f_data, lag, alpha = 0.05, M = NULL)
```

## Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `lag` | the lag to use to compute the single lag test statistic |
| `alpha` | the significance level to be used in the hypothesis test |
| `M` | Number of samples to take when applying a Monte-Carlo approximation |

### Value

numeric value; the 1-alpha confidence bound for the functional autocorrelation coefficient at lag h under a weak white noise assumption.

---

`B_iid_bound` *Compute strong white noise confidence bound for autocorrelation coefficient.*

---

### Description

'B_iid_bound' Computes an approximate asymptotic upper 1-alpha confidence bound for the functional autocorrelation coefficient at lag h under the assumption that f_data forms a strong white noise

### Usage

```
B_iid_bound(f_data, alpha = 0.05)
```

### Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `alpha` | the significance level to be used in the hypothesis test |

### Value

Numeric value; the 1-alpha confidence bound for the functional autocorrelation coefficient at lag h under a strong white noise assumption.

---

`center` *Center functional data*

---

### Description

'center' Centers the given functional data

### Usage

```
center(f_data)
```

### Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |

### Value

A matrix of the same form as f_data containing the centered functional data.

`covariance_diag_store` *List storage of diagonal covariances.*

## Description

'covariance_diag_store' Creates a list storage of approximate diagonal covariances computed by the function diagonal_covariance_i

## Usage

```
covariance_diag_store(f_data, K)
```

## Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `K` | the range of lags 1:K to use |

## Value

A list containing K 2-dimensional arrays containing the diagonal covariance matrices of the functional data, for lags h in the range 1:K.

`covariance_i_j` *Compute the approximate covariance tensor for lag windows defined by i,j*

## Description

'covariance_i_j' Computes the approximate covariance tensor of the functional data for lag windows defined by i,j.

## Usage

```
covariance_i_j(f_data, i, j)
```

## Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `i, j` | the indices i,j in 1:T that we are computing the covariance for |

## Value

A 4-dimensional array, encoding the covariance tensor of the functional data for lag windows defined by i,j.

## covariance_i_j_vec *Compute the approximate covariance tensor for lag windows defined by i,j*

### Description

'covariance_i_j_vec' Computes the approximate covariance tensor of the functional data for lag windows defined by i,j; a vectorized version of covariance_i_j.

### Usage

```
covariance_i_j_vec(f_data, i, j)
```

### Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| i, j | the indices i,j in 1:T that we are computing the covariance for |

### Value

A 4-dimensional array, encoding the covariance tensor of the functional data for lag windows defined by i,j.

## daniell_kernel *Daniell Kernel Function*

### Description

'daniell_kernel' Computes the Daniell kernel function at a given point value.

### Usage

```
daniell_kernel(x)
```

### Arguments

| | |
|---|---|
| x | the point value at which the kernel function is evaluated |

### Value

A scalar value; the value of the Daniell kernel function at the point value x.

## diagonal_autocov_approx_0 *Compute the diagonal covariance*

### Description

'diagonal_autocov_approx_0' Computes the diagonal covariance of the given functional data.

### Usage

```
diagonal_autocov_approx_0(f_data)
```

### Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |

### Value

A numeric value; integral approximation of the diagonal covariance of the functional data.

## diagonal_covariance_i *Compute the approximate diagonal covariance matrix for lag windows defined by i*

### Description

'diagonal_covariance_i' Computes the approximate diagonal covariance matrix of the functional data for lag windows defined by i.

### Usage

```
diagonal_covariance_i(f_data, i)
```

### Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| i | the index in 1:T that we are computing the covariance for |

### Value

A 2-dimensional array, encoding the covariance matrix of the functional data for lag windows defined by i.

far_1_S *'far_1_S' Simulates an FAR(1,S)-fGARCH(1,1) process with N independent observations, each observed discretely at J points on the interval [0,1].*

## Description

'far_1_S' Simulates an FAR(1,S)-fGARCH(1,1) process with N independent observations, each observed discretely at J points on the interval [0,1].

## Usage

```
far_1_S(N, J, S, type = "IID", burn_in = 50)
```

## Arguments

| | |
|---|---|
| N | the number of fGARCH(1,1) curves to sample. |
| J | the number of points at which each curve is sampled (the resolution of the data). |
| S | the autoregressive operator of the model, between 0 and 1, indicating the level of conditional heteroscedasticity. |
| type | the assumed model of the error term. The default argument is 'IID', under which the errors are assumed to be independent and identically distributed. The alternative argument is 'fGARCH', which will assume that the errors follow an fGARCH(1,1) process. |
| burn_in | the number of initial samples to burn (discard). |

## Value

A J x N matrix containing FAR(1,S) functional data in the columns.

## Examples

```
f <- far_1_S(100, 50, 0.75)
```

fgarch_1_1 *'fgarch_1_1' Simulates an fGARCH(1,1) process with N independent observations, each observed*

## Description

'fgarch_1_1' Simulates an fGARCH(1,1) process with N independent observations, each observed

## Usage

```
fgarch_1_1(N, J, delta = 0.01, burn_in = 50)
```

### Arguments

| | |
|---|---|
| N | the number of fGARCH(1,1) curves to sample. |
| J | the number of points at which each curve is sampled (the resolution of the data). |
| delta | a parameter used in the variance recursion of the model. |
| burn_in | the number of initial samples to burn (discard). |

### Value

A list containing two J x N matrices, the former containing the sample of fGARCH(1,1) curves and the latter containing the respective variance values.

### Examples

```
f <- fgarch_1_1(100, 50)
```

---

## fport_test *Compute Functional Hypothesis Tests*

---

### Description

'fport_test' Computes a variety of functional portmanteau hypothesis tests. All hypothesis tests in this package are accessible through this function.

### Usage

```
fport_test(
  f_data,
  test = "multi-lag",
  lag = NULL,
  iid = FALSE,
  M = NULL,
  kernel = "Bartlett",
  bandwidth = "adaptive",
  components = 3,
  bootstrap = FALSE,
  block_size = "adaptive",
  moving = FALSE,
  straps = 300,
  alpha = 0.05,
  complete_test = FALSE,
  suppress_raw_output = FALSE,
  suppress_print_output = FALSE
)
```

## Arguments

| | |
|---|---|
| f_data | The functional data matrix with observed functions in the columns. |
| test | A String specifying the hypothesis test. Currently available tests are referred to by their string handles: "single-lag", "multi-lag", "spectral", "independence", and "imhof". Please see the Details section of the documentation, or the vignette, for a short overview of the available tests. For a more complete treatment of these hypothesis tests, please consult the references. |
| lag | A positive integer value. Only used for the "single-lag", "multi-lag", "independence", and "imhof" tests. This parameter specifies the single lag, or maximum lag, to be used by the specified test. |
| iid | Only used for the "single-lag" and "multi-lag" tests. A Boolean value, FALSE by default. If given TRUE, the hypothesis test will use a strong-white noise assumption (instead of a weak-white noise assumption). |
| M | Only used for the "single-lag" and "multi-lag" tests. A positive Integer. Determines the number of Monte-Carlo simulations employed in the Welch-Satterthwaite approximation of the limiting distribution of the test statistic. |
| kernel | Only used for the "spectral" test. A String, 'Bartlett' by default. Specifies the kernel to be used in the "spectral" test. Currently supported kernels are the 'Bartlett' and 'Parzen' kernels. |
| bandwidth | Only used for the "spectral" test. Either a String or a positive Integer value, 'adaptive' by default. Determines the bandwidth (or lag-window) to be used for the test. Given the string handle 'adaptive', the bandwidth is computed via a bandwidth selection method which aims to minimize the integrated normed error of the spectral density operator. If the given string handle is 'static', the bandwidth is computed to be n^(1/(2q + 1)), where n is the sample size and q is the kernel order. If a positive integer is given, that will be the bandwidth that is used. |
| components | Only used for the "independence" test. A positive Integer value. Determines the number of functional principal components to use (ranked by their importance). |
| bootstrap | Only used for the "single-lag" test. A Boolean value, FALSE by default. If given TRUE, the hypothesis test is evaluated by approximating the limiting distribution of the test statistic via a block bootstrapping process. |
| block_size | Only used for the "single-lag" test in the case when 'bootstrap' = TRUE. A positive Integer value, with the default value being computed via the adaptive bandwidth selection method in the "spectral" test. Determines the block size (of each block in each bootstrap sample) if the test is being bootstrapped. |
| moving | Only used for the "single-lag" test in the case when 'bootstrap' = TRUE. A Boolean value, FALSE by default If given TRUE, the performed block bootstrap will be moving rather than stationary. |
| straps | Only used for the "single-lag" test in the case when 'bootstrap' = TRUE. A positive Integer with a default value of 300. Determines the number of bootstrap samples to take if the test is being bootstrapped. |
| alpha | Numeric value between 0 and 1 specifying the significance level to be used in the specified hypothesis test. The default value is 0.05. Note, the significance value |

is only ever used to compute the 1-alpha quantile of the limiting distribution of the specified test's test statistic.

`complete_test` A Boolean value, FALSE by default. If TRUE, the function requires no other parameters other than f_data, and will return a table with a single column containing p-values from an array of tests contained in the rows.

`suppress_raw_output`
A Boolean value, FALSE by default. If given TRUE, the function will not return a list containing the p-value, quantile and statistic, and instead only prints output to the console.

`suppress_print_output`
A Boolean value, FALSE by default. If TRUE, the function will not print any output to the console.

## Details

The "single-lag" portmanteau test is based on the sample autocovariance function computed from the functional data. This test assesses the significance of lagged autocovariance operators at a single, user-specified lag h. More specifically, it tests the null hypothesis that the lag-h autocovariance operator is equal to 0. This test is designed for stationary functional time-series, and is valid under conditional heteroscedasticity conditions. The required parameter for this test are 'lag', which determines the lag at which the test is evaluated. If this parameter is left blank, it will take a default of 1. The optional parameters for this test are 'iid', 'M', 'bootstrap', 'block_size', 'straps', 'moving', and 'alpha'.

The "multi-lag" portmanteau test is also based on the sample autocovariance function computed from the functional data. This test assesses the cumulative significance of lagged autocovariance operators, up to a user-selected maximum lag K. More specifically, it tests the null hypothesis that the first K lag-h autocovariance operators (h going from 1 to K) is equal to 0. This test is designed for stationary functional time-series, and is valid under conditional heteroscedasticity conditions. The required parameter for this test is 'lag', which determines the maximum lag at which the test is evaluated. If this parameter is left blank, it will take a default of 20. The optional parameters for this test are 'iid', 'M', 'bootstrap', 'block_size', 'straps', 'moving', and 'alpha'.

The "spectral" portmanteau test is based on the spectral density operator. It essentially measures the proximity of a functional time series to a white noise - the constant spectral density operator of an uncorrelated series. Unlike the "single-lag" and "multi-lag" tests, this test is not for general white noise series, and may not hold under functional conditionally heteroscedastic assumptions. The optional parameters for this test are 'kernel', 'bandwidth', and 'alpha'.

The "independence" portmanteau test is a test of independence and identical distribution based on a dimensionality reduction by projecting the data onto the most important functional principal components. It is based on the resulting lagged cross-variances. This test is not for general white noise series, and may not hold under functional conditionally heteroscedastic assumptions. The required parameters for this test are 'lag' and 'components'. The 'lag' parameter determines the maximum lag at which the test is evaluated. The 'components' parameter determines the number of the most important principal components to use (importance is determined by the proportion of the variance that is explained by the individual principal component.)

The "imhof" portmanteau test is an analogue of the "single-lag" test. While the "single-lag" test computes the limiting distribution of the test statistic via a Welch-Satterthwaite approximation, the "imhof" test directly computes the coefficients of the quadratic form in Normal variables which

the test statistic converges too as the sample size goes to infinity. We warn the user that this test is extremely computationally expensive, and is only recommended for small datasets as a means of cross-verification against the single-lag test. The required parameter for this test is 'lag', which determines the lag at which the test is evaluated. The "imhof" test requires the "tensorA" and "CompQuadForm" packages. Note also that the imhof test does not return a statistic, and thus returns a list with only 2 elements if suppress_raw_output = FALSE.

## Value

If suppress_raw_output = FALSE, a list containing the test statistic, the 1-alpha quantile of the limiting distribution, and the p-value computed from the specified hypothesis test. Also prints output containing a short description of the test, the p-value, and additional information about the test if suppress_print_output = FALSE. If 'complete-test' = TRUE, will return a 1-column table instead containing the p-values for a variety of tests, which are given short descriptions in the index of the table.

## References

[1] Kokoszka P., & Rice G., & Shang H.L. (2017). Inference for the autocovariance of a functional time series under conditional heteroscedasticity. Journal of Multivariate Analysis, 162, 32-50.

[2] Characiejus V., & Rice G. (2019). A general white noise test based on kernel lag-window estimates of the spectral density operator. Econometrics and Statistics, submitted.

[3] Gabrys R., & Kokoszka P. (2007). Portmanteau Test of Independence for Functional Observations. Journal of the American Statistical Association, 102:480, 1338-1348, DOI: 10.1198/016214507000001111.

[4] Zhang X. (2016). White noise testing and model diagnostic checking for functional time series. Journal of Econometrics, 194, 76-95.

[5] Chen W.W. & Deo R.S. (2004). Power transformations to induce normality and their applications. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 66, 117–130.

## Examples

```
b <- brown_motion(250, 50)
fport_test(b, test = 'single-lag', lag = 10)
fport_test(b, test = 'multi-lag', lag = 10, alpha = 0.01)
fport_test(b, test = 'single-lag', lag = 1, M = 250)
fport_test(b, test = 'spectral', kernel = 'Bartlett', bandwidth = 'static', alpha = 0.05)
fport_test(b, test = 'spectral', alpha = 0.1, kernel = 'Parzen', bandwidth = 'adaptive')
fport_test(b, test = 'independence', components = 3, lag = 3)
```

---

`iid_covariance` *Compute part of the covariance under a strong white noise assumption*

---

## Description

'iid_covariance' A helper function used to compute one of the two independent sum terms in the computation of the approximate covariance of the functional data under a strong white noise assumption.

### Usage

```
iid_covariance(f_data)
```

### Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |

### Value

A 2-dimensional matrix containing one of the two independent sums in the computation of the covariance.

---

## iid_covariance_vec *Compute part of the covariance under a strong white noise assumption*

---

### Description

'iid_covariance_vec' A helper function used to compute one of the two independent sum terms in the computation of the approximate covariance of the functional data under a strong white noise assumption; a vectorized version of iid_covariance.

### Usage

```
iid_covariance_vec(f_data)
```

### Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |

### Value

A 2-dimensional matrix containing one of the two independent sums in the computation of the covariance.

---

## independence_test *Independence Test*

---

### Description

'independence_test' Computes the independence test with a user-specified number of principal components and range of lags.

## Usage

```
independence_test(
  f_data,
  components,
  lag,
  alpha = 0.05,
  suppress_raw_output = FALSE,
  suppress_print_output = FALSE
)
```

## Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| components | A positive Integer specifying the number of principal components to project the data on; ranked in order of importance (importance is determined by the proportion of the variance that is explained by the individual principal component.) |
| lag | A positive Integer value, specifying the maximum lag to include - this can be seen as the bandwidth or lag-window. |
| alpha | Numeric value between 0 and 1 specifying the significance level to be used in the specified hypothesis test. The default value is 0.05. Note, the significance value is only ever used to compute the 1-alpha quantile of the limiting distribution of the specified test's test statistic. |
| suppress_raw_output | Boolean value, FALSE by default. If TRUE, the function will not return the list containing the p-value, quantile, and statistic. |
| suppress_print_output | Boolean value, FALSE by default. If TRUE, the function will not print any output to the console. |

## Details

The "independence" portmanteau test is a test of independence and identical distribution based on a dimensionality reduction by projecting the data onto the most important functional principal components. It is based on the resulting lagged cross-variances. This test is not for general white noise series, and may not hold under functional conditionally heteroscedastic assumptions. Please consult the vignette for a deeper exposition, and consult the reference for a complete treatment.

## Value

If suppress_raw_output = FALSE, a list containing the test statistic, the 1-alpha quantile of the limiting distribution, and the p-value computed from the specified hypothesis test. Also prints output containing a short description of the test, the p-value, and additional information about the test if suppress_print_output = FALSE.

## References

[1] Gabrys R., & Kokoszka P. (2007). Portmanteau Test of Independence for Functional Observations. Journal of the American Statistical Association, 102:480, 1338-1348, DOI: 10.1198/016214507000001111.

## Examples

```
b <- brown_motion(250, 100)
independence_test(b, components = 3, lag = 5)
```

---

`multi_lag_test` *Multi-Lag Hypothesis Test*

---

## Description

'multi_lag_test' Computes the multi-lag hypothesis test over a range of user-specified lags.

## Usage

```
multi_lag_test(
  f_data,
  lag = 20,
  M = NULL,
  iid = FALSE,
  alpha = 0.05,
  suppress_raw_output = FALSE,
  suppress_print_output = FALSE
)
```

## Arguments

| | |
|---|---|
| `f_data` | the functional data matrix with observed functions in the columns |
| `lag` | Positive integer value. The lag to use to compute the single lag test statistic |
| `M` | Positive integer value. Number of Monte-Carlo simulation for Welch-Satterthwaite approximation. |
| `iid` | A Boolean value, FALSE by default. If given TRUE, the hypothesis test will use a strong-white noise assumption (instead of a weak-white noise assumption). |
| `alpha` | Numeric value between 0 and 1 specifying the significance level to be used in the specified hypothesis test. The default value is 0.05. Note, the significance value is only ever used to compute the 1-alpha quantile of the limiting distribution of the specified test's test statistic. |
| `suppress_raw_output` | Boolean value, FALSE by default. If TRUE, the function will not return the list containing the p-value, quantile, and statistic. |
| `suppress_print_output` | Boolean value, FALSE by default. If TRUE, the function will not print any output to the console. |

## Details

The "multi-lag" portmanteau test is also based on the sample autocovariance function computed from the functional data. This test assesses the cumulative significance of lagged autocovariance operators, up to a user-selected maximum lag K. More specifically, it tests the null hypothesis that the first K lag-h autocovariance operators (h going from 1 to K) is equal to 0. This test is designed for stationary functional time-series, and is valid under conditional heteroscedasticity conditions.

## Value

If suppress_raw_output = FALSE, a list containing the test statistic, the 1-alpha quantile of the limiting distribution, and the p-value computed from the specified hypothesis test. Also prints output containing a short description of the test, the p-value, and additional information about the test if suppress_print_output = FALSE.

## References

[1] Kokoszka P., & Rice G., & Shang H.L. (2017). Inference for the autocovariance of a functional time series under conditional heteroscedasticity. Journal of Multivariate Analysis, 162, 32-50.

## Examples

```
b <- brown_motion(150, 50)
multi_lag_test(b, lag = 5)
multi_lag_test(b, lag = 10, M = 50)
```

---

`parzen_kernel` *Parzen Kernel Function*

---

## Description

'parzen_kernel' Computes the Parzen kernel function at a given point value.

## Usage

```
parzen_kernel(x)
```

## Arguments

| | |
|---|---|
| `x` | the point value at which the kernel function is evaluated |

## Value

A scalar value; the value of the Parzen kernel function at the point value x.

Q_WS_hyp_test *Compute size alpha single-lag hypothesis test under weak or strong white noise assumption*

## Description

'Q_WS_hyp_test' Computes the size alpha test of a single lag hypothesis under a weak white noise or strong white noise assumption using a Welch-Satterthwaite Approximation.

## Usage

```
Q_WS_hyp_test(
  f_data,
  lag,
  alpha = 0.05,
  iid = FALSE,
  M = NULL,
  bootstrap = FALSE,
  block_size = "adaptive",
  straps = 300,
  moving = FALSE
)
```

## Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| lag | the lag to use to compute the single lag test statistic |
| alpha | the significance level to be used in the hypothesis test |
| iid | boolean value, if given TRUE, the hypothesis test will use a strong-white noise assumption. By default is FALSE, in which the hypothesis test will use a weak-white noise assumption. |
| M | Number of samples to take when applying a Monte-Carlo approximation |
| bootstrap | boolean value, if given TRUE, the hypothesis test is done by approximating the limiting distribution of the test statistic via a block bootstrap algorithm. FALSE by default |
| block_size | the block size to be used in the block bootstrap method (in each bootstrap sample). 10 by default. |
| straps | the number of bootstrap samples to take; 300 by default |
| moving | boolean value; determines whether or not the block bootstrap should be moving |

## Value

A list containing the p-value, the quantile, and a boolean value indicating whether or not the hypothesis is rejected.

scalar_covariance_i_j *Compute the approximate covariance at a point for lag windows defined by i,j*

## Description

'scalar_covariance_i_j' Computes the approximate covariance at a point of the functional data for lag windows defined by i,j; a scalarized version of covariance_i_j that takes point estimates.

## Usage

```
scalar_covariance_i_j(f_data, i, j, times)
```

## Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| i, j | the indices i,j in 1:T that we are computing the covariance for |
| times | A vector with 4 columns containing indices specifying which subset of f_data to consider |

## Value

A numeric value; the covariance of the functional data at a point for lag windows defined by i,j.

scalar_covariance_i_j_vec *Compute the approximate covariance at a point for lag windows defined by i,j*

## Description

'scalar_covariance_i_j_vec' Computes the approximate covariance at a point of the functional data for lag windows defined by i,j; a vectorized version of scalar_covariance_i_j.

## Usage

```
scalar_covariance_i_j_vec(f_data, i, j, times)
```

## Arguments

| | |
|---|---|
| f_data | the functional data matrix with observed functions in the columns |
| i, j | the indices i,j in 1:T that we are computing the covariance for |
| times | A vector with 4 columns containing indices specifying which subset of f_data to consider |

## Value

A numeric value; the covariance of the functional data at a point for lag windows defined by i,j.

---

`single_lag_test` *Single-Lag Hypothesis Test*

---

## Description

'single_lag_test' Computes the single-lag hypothesis test at a single user-specified lag.

## Usage

```
single_lag_test(
  f_data,
  lag = 1,
  alpha = 0.05,
  iid = FALSE,
  M = NULL,
  bootstrap = FALSE,
  block_size = "adaptive",
  straps = 300,
  moving = FALSE,
  suppress_raw_output = FALSE,
  suppress_print_output = FALSE
)
```

## Arguments

| | |
|---|---|
| `f_data` | The functional data matrix with observed functions in the columns |
| `lag` | Positive integer value. The lag to use to compute the single lag test statistic. |
| `alpha` | Numeric value between 0 and 1 specifying the significance level to be used in the specified hypothesis test. The default value is 0.05. Note, the significance value is only ever used to compute the 1-alpha quantile of the limiting distribution of the specified test's test statistic. |
| `iid` | A Boolean value, FALSE by default. If given TRUE, the hypothesis test will use a strong-white noise assumption (instead of a weak-white noise assumption). |
| `M` | Positive integer value. Number of Monte-Carlo simulations for the Welch-Satterthwaite approximation. |
| `bootstrap` | A Boolean value, FALSE by default If given TRUE, the hypothesis test is done by approximating the limiting distribution of the test statistic via a block bootstrap process. |
| `block_size` | A positive Integer value, with the default value being computed via the adaptive bandwidth selection method in the "spectral" test. Determines the block size (of each block in each bootstrap sample) if the test is being bootstrapped. |

straps A positive Integer, with a default value of 300. Determines the number of bootstrap samples to take if the test is being bootstrapped. Only used if 'bootstrap' == TRUE.

moving A Boolean value, FALSE by default If given TRUE, the performed block bootstrap will be moving rather than stationary.

suppress_raw_output
Boolean value, FALSE by default. If TRUE, the function will not return the list containing the p-value, quantile, and statistic.

suppress_print_output
Boolean value, FALSE by default. If TRUE, the function will not print any output to the console.

## Details

The "single-lag" portmanteau test is based on the sample autocovariance function computed from the functional data. This test assesses the significance of lagged autocovariance operators at a single, user-specified lag h. More specifically, it tests the null hypothesis that the lag-h autocovariance operator is equal to 0. This test is designed for stationary functional time-series, and is valid under conditional heteroscedasticity conditions.

## Value

If suppress_raw_output = FALSE, a list containing the test statistic, the 1-alpha quantile of the limiting distribution, and the p-value computed from the specified hypothesis test. Also prints output containing a short description of the test, the p-value, and additional information about the test if suppress_print_output = FALSE.

## References

[1] Kokoszka P., & Rice G., & Shang H.L. (2017). Inference for the autocovariance of a functional time series under conditional heteroscedasticity. Journal of Multivariate Analysis, 162, 32-50.

## Examples

```
f <- far_1_S(150, 50, S = 0.75)
single_lag_test(f, lag = 1)
single_lag_test(f, lag = 2, M=100)
```

---

# spectral_test *Spectral Density Test*

---

## Description

The "spectral" portmanteau test is based on the spectral density operator. It essentially measures the proximity of a functional time series to a white noise - the constant spectral density operator of an uncorrelated series. Unlike the "single-lag" and "multi-lag" tests, this test is not for general white noise series, and may not hold under functional conditionally heteroscedastic assumptions.

## Usage

```
spectral_test(
  f_data,
  kernel = "Bartlett",
  bandwidth = "adaptive",
  alpha = 0.05,
  suppress_raw_output = FALSE,
  suppress_print_output = FALSE
)
```

## Arguments

| | |
|---|---|
| f_data | The functional data matrix with observed functions in the columns |
| kernel | A String specifying the kernel function to use. The currently supported kernels are the 'Bartlett' and 'Parzen' kernels. The default kernel is 'Bartlett'. |
| bandwidth | A String or positive Integer value which specifies the bandwidth to use. Currently admitted string handles are 'static' which computes the bandwidth p via p = n^(1/(2q+1)) where n is the sample size and q is the kernel order, or 'adaptive' which uses a bandwidth selection method that is based on the functional data. |
| alpha | Numeric value between 0 and 1 specifying the significance level to be used for the test. The significance level is 0.05 by default. Note, the significance value is only ever used to compute the 1-alpha quantile of the limiting distribution of the specified test's test statistic. |
| suppress_raw_output | Boolean value, FALSE by default. If TRUE, the function will not return the list containing the p-value, quantile, and statistic. |
| suppress_print_output | Boolean value, FALSE by default. If TRUE, the function will not print any output to the console. |

## Details

'spectral_test' Computes the spectral hypothesis test under a user-specified kernel function and bandwidth; automatic bandwidth selection methods are provided.

## Value

If suppress_raw_output = FALSE, a list containing the test statistic, the 1-alpha quantile of the limiting distribution, and the p-value computed from the specified hypothesis test. Also prints output containing a short description of the test, the p-value, and additional information about the test if suppress_print_output = FALSE.

## References

[1] Characiejus V., & Rice G. (2019). A general white noise test based on kernel lag-window estimates of the spectral density operator. Econometrics and Statistics, submitted.

[2] Chen W.W. & Deo R.S. (2004). Power transformations to induce normality and their applications. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 66, 117–130.

## Examples

```
b <- brown_motion(100, 50)
spectral_test(b)
spectral_test(b, kernel = 'Parzen', bandwidth = 'adaptive')
spectral_test(b, kernel = 'Bartlett', bandwidth = 2)
```

# Index