Package ‘fmi’

November 17, 2025

Title Hierarchical Permutation Tests for Functional Measurement Invariance

Description Provides a suite of functions to test for Functional Measurement Invariance (FMI) between two groups. Implements hierarchical permutation tests for configural, metric, and scalar invariance, adapting concepts from Multi-Group Confirmatory Factor Analysis (MGCFA) to functional data.
Methods are based on concepts from:
Meredith, W. (1993) <doi:10.1007/BF02294825>,5
Yao, F., Müller, H. G., & Wang, J. L. (2005) <doi:10.1198/016214504000001745>,
and Lee, K. Y., & Li, L. (2022) <doi:10.1111/rssb.12471>.

Version 0.1.7

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports dplyr, ggplot2, knitr, magrittr, methods, purrr, refund, stats, tibble, tidyr, utils

Suggests fda, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Hyeri Kim [aut, cre]

Maintainer Hyeri Kim <gpfl3616@naver.com>

Depends R (>= 4.1.0)

Repository CRAN

Date/Publication 2025-11-17 21:20:07 UTC

Contents

`build_scree_plot` *Build Scree Plot for Configural Invariance*

Description

Creates a ggplot2 scree plot comparing the PVE by FPCs for two groups.

Usage

```
build_scree_plot(fpca_A, fpca_B, groups)
```

Arguments

`fpca_A`	An `fpca.sc` object for Group A.
`fpca_B`	An `fpca.sc` object for Group B.
`groups`	A character vector of the two group names.

Value

A ggplot object.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
g_fac <- factor(sim$group_vec)
idx_A <- which(g_fac == levels(g_fac)[1])
# --- FIX: Added Y= and argvals= ---
fpca_A <- refund::fpca.sc(Y = sim$Y_mat[idx_A, ], argvals = sim$argvals, npc = 3)
fpca_B <- refund::fpca.sc(Y = sim$Y_mat[-idx_A, ], argvals = sim$argvals, npc = 3)
build_scree_plot(fpca_A, fpca_B, levels(g_fac))
```

`compare_latent_means` *Compare Latent Means*

Description

Performs Welch's t-tests on FPC scores between groups. This function should only be run after scalar invariance is established.

Usage

```
compare_latent_means(fmi_results, group_vec)
```

Arguments

`fmi_results`	The list object returned by `run_fmi()`.
`group_vec`	The original group vector used in `run_fmi()`.

Value

A tibble summarizing the t-test results for each FPC.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
fmi_results <- run_fmi(
  Y_mat = sim$Y_mat,
  group_vec = sim$group_vec,
  argvals = sim$argvals,
  n_perms = 9,
  progress = FALSE
)
# This will run only if scalar invariance passed
if (!is.null(fmi_results$scalar) && fmi_results$scalar$passed) {
  mean_results <- compare_latent_means(fmi_results, sim$group_vec)
  print(mean_results)
}
```

`determine_npc` *Determine Number of FPCs Automatically*

Description

Determines the number of functional principal components (FPCs) required to meet a target Proportion of Variance Explained (PVE).

Usage

```
determine_npc(Y_mat, argvals, target_pve = 0.95, max_npc = 6L)
```

Arguments

`Y_mat`	A numeric matrix (N x M) where N is subjects, M is time points.
`argvals`	A numeric vector of length M listing the observation points.
`target_pve`	The target cumulative PVE to reach (default: 0.95).
`max_npc`	The maximum number of components to retain (default: 6L).

Value

A list containing:

`npc`	The selected number of components.
`pooled_fpca`	The `fpca.sc` object from the `refund` package.
`scree`	A tibble with PVE and cumulative PVE for each component.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
npc_info <- determine_npc(sim$Y_mat, sim$argvals)
print(npc_info$npc)
```

`run_dti_example` *DTI Example Data Wrapper*

Description

A helper function to load and pre-process the DTI dataset from the 'refund' package for use in examples.

Usage

```
run_dti_example(n_perms = 499L)
```

Arguments

`n_perms`	The number of permutations (default: 499L). Passed to `run_fmi`.

Value

Invisibly returns the FMI results list.

Examples

```
# n_perms=9 is for a quick check. Use 499+ for real analysis.
if (requireNamespace("refund", quietly = TRUE)) {
  dti_results <- run_dti_example(n_perms = 9)
}
```

---

run_fmi *Run Hierarchical Functional Measurement Invariance (FMI) Tests*

Description

A wrapper function that performs a sequence of permutation tests for configural, metric, scalar, and (optionally) strict invariance.

Usage

```
run_fmi(
  Y_mat,
  group_vec,
  argvals,
  alpha = 0.05,
  npc = NULL,
  target_pve = 0.95,
  max_npc = 6L,
  n_perms = 499L,
  strict_test = FALSE,
  progress = interactive()
)
```

Arguments

`Y_mat`	A numeric matrix (N x M) where N is subjects, M is time points.
`group_vec`	A vector of length N specifying group membership (2 groups).
`argvals`	A numeric vector of length M listing the observation points.
`alpha`	Significance level (default: 0.05).
`npc`	Optional. The number of FPCs. If NULL (default), it is determined automatically by `determine_npc`.
`target_pve`	The target PVE if `npc` is NULL (default: 0.95).
`max_npc`	The max FPCs if `npc` is NULL (default: 6L).
`n_perms`	The number of permutations (default: 499L).
`strict_test`	Boolean. Whether to perform the strict invariance test (default: FALSE).
`progress`	Boolean. Show progress bars? (default: interactive()).

Value

A list containing test results for each invariance level, settings, and the pooled FPCA object.

Examples

```
 # Use small N and n_perms for a quick example
 sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
 fmi_results <- run_fmi(
   Y_mat = sim$Y_mat,
   group_vec = sim$group_vec,
   argvals = sim$argvals,
   n_perms = 9, # Use 499+ for actual research
   progress = FALSE
 )
print(fmi_results)
```

`run_growth_example` *Berkeley Growth Example Data Wrapper*

Description

A helper function to load and pre-process the Growth dataset from the 'fda' package for use in examples.

Usage

```
run_growth_example(n_perms = 499L)
```

Arguments

`n_perms`	The number of permutations (default: 499L). Passed to `run_fmi`.

Value

Invisibly returns the FMI results list.

Examples

```
 # n_perms=9 is for a quick check. Use 499+ for real analysis.
 if (requireNamespace("fda", quietly = TRUE)) {
   growth_results <- run_growth_example(n_perms = 9)
 }
```

`simulate_fmi_data` *Simulate FMI Data*

Description

Generates functional data for two groups with optional differences in latent means or scalar intercepts.

Usage

```
simulate_fmi_data(
  N_A = 50,
  N_B = 50,
  T_points = 51,
  mean_shift = 0,
  scalar_shift = 0,
  eigenvalues = c(4, 1, 0.5, 0.1),
  noise_sd = 0.5,
  seed = NULL
)
```

Arguments

`N_A`	Sample size for Group A (default: 50).
`N_B`	Sample size for Group B (default: 50).
`T_points`	Number of time points (default: 51).
`mean_shift`	A shift added to FPC 1 scores for Group B (default: 0).
`scalar_shift`	A shift added to the mean function for Group B (default: 0).
`eigenvalues`	A vector of eigenvalues for FPCs (default: c(4, 1, 0.5, 0.1)).
`noise_sd`	Standard deviation of measurement error (default: 0.5).
`seed`	Optional. A random seed for reproducibility.

Value

A list with `Y_mat`, `group_vec`, and `argvals`.

Examples

```
sim_data <- simulate_fmi_data(N_A = 10, N_B = 10, T_points = 20, seed = 123)
str(sim_data)
```

`test_configural` *Test for Configural Invariance*

Description

Permutation test to check if the cumulative PVE is equivalent across groups.

Usage

```
test_configural(Y_mat, group_vec, argvals, npc, n_perms, alpha, progress)
```

Arguments

`Y_mat`	A numeric matrix (N x M) where N is subjects, M is time points.
`group_vec`	A vector of length N specifying group membership (2 groups).
`argvals`	A numeric vector of length M listing the observation points.
`npc`	Optional. The number of FPCs. If NULL (default), it is determined automatically by `determine_npc`.
`n_perms`	The number of permutations (default: 499L).
`alpha`	Significance level (default: 0.05).
`progress`	Boolean. Show progress bars? (default: interactive()).

Value

A list with `passed` (boolean), `p_value`, `statistic`, and `plot`.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
res <- test_configural(sim$Y_mat, sim$group_vec, sim$argvals,
                       npc = 3, n_perms = 9, alpha = 0.05, progress = FALSE)
print(res$p_value)
```

`test_metric` *Test for Metric Invariance*

Description

Permutation test to check if the eigenfunctions are equivalent across groups.

Usage

```
test_metric(Y_mat, group_vec, argvals, npc, n_perms, alpha, progress)
```

Arguments

`Y_mat`	A numeric matrix (N x M) where N is subjects, M is time points.
`group_vec`	A vector of length N specifying group membership (2 groups).
`argvals`	A numeric vector of length M listing the observation points.
`npc`	Optional. The number of FPCs. If NULL (default), it is determined automatically by `determine_npc`.
`n_perms`	The number of permutations (default: 499L).
`alpha`	Significance level (default: 0.05).
`progress`	Boolean. Show progress bars? (default: interactive()).

Value

A list with `passed` (boolean), `p_value`, and `statistic`.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
res <- test_metric(sim$Y_mat, sim$group_vec, sim$argvals,
                   npc = 3, n_perms = 9, alpha = 0.05, progress = FALSE)
print(res$p_value)
```

`test_scalar` *Test for Scalar Invariance*

Description

Permutation test to check if the mean functions (intercepts) are equivalent across groups, after accounting for latent factors.

Usage

```
test_scalar(Y_mat, group_vec, argvals, pooled_fpca, n_perms, alpha, progress)
```

Arguments

`Y_mat`	Numeric matrix (N x M).
`group_vec`	A vector of length N specifying group membership.
`argvals`	Numeric vector of length M.
`pooled_fpca`	The `fpca.sc` object from the `determine_npc` function.
`n_perms`	Integer, number of permutations.
`alpha`	Significance level (default: 0.05).
`progress`	Boolean, show progress bar? (default: interactive()).

Value

A list with passed (boolean), `p_value`, and `statistic`.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
npc_info <- determine_npc(sim$Y_mat, sim$argvals, max_npc = 3)
res <- test_scalar(sim$Y_mat, sim$group_vec, sim$argvals,
                   pooled_fpca = npc_info$pooled_fpca,
                   n_perms = 9, alpha = 0.05, progress = FALSE)
print(res$p_value)
```

`test_strict` *Test for Strict Invariance*

Description

Permutation test to check if the residual error variances are equivalent across groups.

Usage

```
test_strict(Y_mat, group_vec, argvals, npc, n_perms, alpha, progress)
```

Arguments

`Y_mat`	A numeric matrix (N x M) where N is subjects, M is time points.
`group_vec`	A vector of length N specifying group membership (2 groups).
`argvals`	A numeric vector of length M listing the observation points.
`npc`	Optional. The number of FPCs. If NULL (default), it is determined automatically by `determine_npc`.
`n_perms`	The number of permutations (default: 499L).
`alpha`	Significance level (default: 0.05).
`progress`	Boolean. Show progress bars? (default: interactive()).

Value

A list with passed (boolean), `p_value`, and `statistic`.

Examples

```
sim <- simulate_fmi_data(N_A = 20, N_B = 20, T_points = 30)
res <- test_strict(sim$Y_mat, sim$group_vec, sim$argvals,
                   npc = 3, n_perms = 9, alpha = 0.05, progress = FALSE)
print(res$p_value)
```

Index