Package ‘multiCCA’

March 23, 2026

Title Multiple Canonical Correlation Analysis (Kernel and Functional)

Version 0.1.0

Description Implements methods for multiple canonical correlation analysis (CCA) for more than two data blocks,
with a focus on multivariate repeated measures and functional data. The package provides two approaches:
(i) multiple kernel CCA, which embeds each data block into a reproducing kernel Hilbert space to capture
nonlinear dependencies, and (ii) multiple functional CCA, which represents repeated measurements as smooth
functions and performs analysis in a Hilbert space framework. Both approaches are formulated via covariance
operators and solved as generalized eigenvalue problems with regularization to ensure numerical stability.
The methods allow estimation of canonical variables, generalized canonical correlations, and low-dimensional
representations for exploratory analysis and visualization of dependence structures across multiple feature sets.
The implementation follows the framework developed in Górecki, Krzyśko, Gnettner and Kokoszka (2025)
<doi:10.48550/arXiv.2510.04457>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports fda, geigen, ggplot2, rlang

URL https://github.com/Halmaris/multiCCA

BugReports https://github.com/Halmaris/multiCCA/issues

NeedsCompilation no

Author Tomasz Gorecki [aut, cre] (ORCID:
<https://orcid.org/0000-0002-9969-5257>)

Maintainer Tomasz Gorecki <tomasz.gorecki@amu.edu.pl>

Repository CRAN

Date/Publication 2026-03-23 17:50:07 UTC

Contents

hopkins_vs_components *Compute Hopkins statistic for increasing numbers of components*

Description

Computes the Hopkins statistic for representations based on increasing numbers of canonical components obtained from an MCCA model.

Usage

```
hopkins_vs_components(
  fit,
  blocks = NULL,
  max_comp = NULL,
  m = NULL,
  nrep = 50,
  seed = NULL
)
```

Arguments

fit	Fitted object of class mcca_fit.
blocks	Vector of block indices included in the representation. If NULL, all blocks are used.
max_comp	Maximum number of components to evaluate.
m	Number of sampled points used in the Hopkins statistic.
nrep	Number of repetitions used to estimate the statistic.
seed	Optional random seed.

Value

A data frame containing:

- `components` – number of components,
- `hopkins` – mean Hopkins statistic,
- `sd` – standard deviation across repetitions.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)

fit <- mcca_fit(method = "kernel", X = X, ncomp = 3)

hopkins_vs_components(fit, max_comp = 2)
```

`mcca_fit` *Fit multiple canonical correlation analysis*

Description

Fits either kernel MCCA for repeated measures data or functional MCCA.

Usage

```
mcca_fit(
  method = c("kernel", "functional"),
  X,
  ncomp = 2,
  eps = 0.001,
  gamma = NULL,
  nbasis = 5,
  basis_type = "fourier",
  argvals = NULL
)
```

Arguments

`method` Character string specifying the method to use. Either `'kernel'` or `'functional'`.

`X` List of feature blocks. Each element `X[[l]]` is a list of matrices, where `X[[l]][[i]]` is a `T x p_l` matrix representing the observations for subject `i` in block `l`.

`ncomp` Number of leading canonical components.

`eps` Regularization parameter.

`gamma` Gaussian kernel bandwidth for the kernel method.

`nbasis` Number of basis functions for the functional method.

`basis_type` Basis type for the functional method. Either `'fourier'` or `'bspline'`.

`argvals` Optional time grid for the functional method.

Value

An object of class `mcca_fit`. The following S3 methods are available for this class:

- `print()` prints a short summary of the fitted model,
- `summary()` provides a more detailed summary of the model.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2)))

fit <- mcca_fit(method = 'kernel',
  X = X,
  ncomp = 2,
  eps = 1e-2)

print(fit)
summary(fit)
```

`mcca_grid_search` *Grid search for tuning parameters in MCCA*

Description

Performs a grid search over regularization and model parameters for kernel or functional multiple canonical correlation analysis.

Usage

```
mcca_grid_search(
  method = c("kernel", "functional"),
  X,
  eps_grid,
  gamma_grid = NULL,
  nbasis_grid = NULL,
  ncomp_eval = 2,
  criterion = c("first", "sumk"),
  basis_type = "fourier",
  argvals = NULL
)
```

Arguments

`method`	Character string specifying the method. Either ’kernel’ or ’functional’.
`X`	List of feature blocks.
`eps_grid`	Vector of candidate regularization parameters.
`gamma_grid`	Vector of candidate kernel bandwidths (kernel method).
`nbasis_grid`	Vector of candidate numbers of basis funs (fun. method).
`ncomp_eval`	Number of canonical components used in evaluation.
`criterion`	Criterion used to rank models: ’first’ or ’sumk’.
`basis_type`	Basis type used in fun. method: ’fourier’ or ’bspline’.
`argvals`	Optional time grid for functional data.

Value

A list containing:

- `results` – data frame with grid search scores,
- `best_fit` – fitted MCCA model for the best parameter set,
- `best_row` – row of the grid corresponding to the best parameters.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)

res <- mcca_grid_search(
  method = "kernel",
  X = X,
  eps_grid = c(1e-3, 1e-2),
  gamma_grid = c(0.1, 1),
  ncomp_eval = 2
)

res$best_row
```

mcca_pipeline *Run the full MCCA analysis pipeline*

Description

This function performs a complete analysis workflow for multiple canonical correlation analysis (MCCA). It fits either the kernel or functional MCCA model, computes the Hopkins statistic for increasing numbers of components, and produces diagnostic plots.

Usage

```
mcca_pipeline(
  method = c("kernel", "functional"),
  X,
  groups = NULL,
  labels = NULL,
  ncomp = 5,
  eps = 0.001,
  gamma = NULL,
  nbasis = 5,
  basis_type = "fourier",
  argvals = NULL,
  hopkins_blocks = NULL,
  hopkins_max_comp = NULL,
  hopkins_m = NULL,
  hopkins_nrep = 100,
  block_x = 1,
  block_y = 2,
  pair_comp = 1
)
```

Arguments

method	Character string specifying the method to use. Either 'kernel' or 'functional'.
X	List of feature blocks. Each element `X[[l]]` is a list of matrices, where `X[[l]][[i]]` is a `T x p_l` matrix representing the observations for subject `i` in block `l`.
groups	Optional grouping variable used for coloring points in plots.
labels	Optional vector of labels used to annotate points in plots.
ncomp	Number of canonical components to estimate.

eps Regularization parameter used in the generalized eigenproblem.

gamma Gaussian kernel bandwidth used in the kernel MCCA method.

nbasis Number of basis functions used in the functional MCCA method.

basis_type Basis type used in the functional method. Either 'fourier' or 'bspline'.

argvals Optional time grid used for functional data smoothing.

hopkins_blocks Vector of block indices used when computing the Hopkins statistic. If NULL, all blocks are used.

hopkins_max_comp
Maximum number of components used when computing the Hopkins statistic. If NULL, all available components are used.

hopkins_m Number of sampled points used in the Hopkins statistic.

hopkins_nrep Number of repetitions used when estimating the Hopkins statistic.

block_x Index of the first block used in the pairwise component plot.

block_y Index of the second block used in the pairwise component plot.

pair_comp Index of the canonical component used in the pairwise plot.

Value

A list containing:

- fit – fitted MCCA model object.
- hopkins – data frame with Hopkins statistic values.
- pair_plot – ggplot object with pairwise component scatter plot.
- hopkins_plot – ggplot object showing Hopkins statistic versus number of components.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)

groups <- sample(1:2, n, replace = TRUE)

res <- mcca_pipeline(
  method = 'kernel',
  X = X,
  groups = groups,
  ncomp = 3,
  eps = 1e-2
)
```

```
res$fit
head(res$hopkins)
```

plot_hopkins_curve *Plot Hopkins statistic curve*

Description

Plots the Hopkins statistic as a function of the number of canonical components.

Usage

```
plot_hopkins_curve(df, title = NULL)
```

Arguments

`df`	Data frame returned by `hopkins_vs_components()`.
`title`	Optional plot title.

Value

A `ggplot2` object.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)

fit <- mcca_fit(method = "kernel", X = X, ncomp = 3)
H <- hopkins_vs_components(fit, max_comp = 2)

plot_hopkins_curve(H)
```

plot_mcca_pair *Plot canonical components for two blocks*

Description

Creates a scatter plot comparing canonical components from two blocks.

Usage

```
plot_mcca_pair(
  fit_x,
  fit_y = NULL,
  block_x = 1,
  block_y = 2,
  comp = 1,
  groups = NULL,
  labels = NULL,
  title = NULL
)
```

Arguments

fit_x	Fitted object of class mcca_fit.
fit_y	Optional second MCCA object. If NULL, fit_x is used.
block_x	Block index for x-axis.
block_y	Block index for y-axis.
comp	Component index.
groups	Optional grouping variable.
labels	Optional point labels.
title	Optional plot title.

Value

A ggplot2 object.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)
```

```
fit <- mcca_fit(method = "kernel", X = X, ncomp = 2)

plot_mcca_pair(fit)
```

plot_mcca_scatter *Plot canonical components for a single block*

Description

Creates a scatter plot of two canonical components obtained from an MCCA model for a selected block.

Usage

```
plot_mcca_scatter(
  fit,
  block = 1,
  comp_x = 1,
  comp_y = 2,
  groups = NULL,
  labels = NULL,
  add_ellipse = TRUE,
  point_size = 2.5,
  title = NULL
)
```

Arguments

`fit`	Object of class `mcca_fit`.
`block`	Block index.
`comp_x`	Component used on the x-axis.
`comp_y`	Component used on the y-axis.
`groups`	Optional grouping variable.
`labels`	Optional point labels.
`add_ellipse`	Logical; whether to draw group ellipses.
`point_size`	Size of plotted points.
`title`	Optional plot title.

Value

A `ggplot2` object.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)

fit <- mcca_fit(method = "kernel", X = X, ncomp = 2)

plot_mcca_scatter(fit)
```

---

predict.mcca_fit *Predict canonical component scores for new data*

Description

Projects new observations onto canonical components obtained from a fitted MCCA model (kernel or functional).

Usage

```
## S3 method for class 'mcca_fit'
predict(object, newdata, ...)
```

Arguments

object	Object of class mcca_fit.
newdata	List of feature blocks with the same structure as used during model fitting.
...	Additional arguments (unused).

Value

List of matrices containing projected component scores for each block.

Examples

```
set.seed(1)

n <- 20
T_len <- 10

X <- list(
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 3), T_len, 3)),
  lapply(seq_len(n), function(i) matrix(rnorm(T_len * 2), T_len, 2))
)

fit <- mcca_fit(method = 'kernel', X = X, ncomp = 2)

predict(fit, X)
```

---

summary.mcca_fit *Summarize an MCCA model*

Description

Provides a summary of a fitted MCCA model, including the number of blocks, sample size, regularization parameter, and generalized canonical correlations.

Usage

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

Arguments

`object`	An object of class `mcca_fit`.
`...`	Additional arguments (currently unused).

Value

An object of class `summary.mcca_fit`.

Index