Package 'FPScausal'

August 23, 2026

Type Package

Title Functional Propensity Score for Causal Inference

Version 0.1.1

Description Implements functional propensity score (FPS) weighting for causal inference with functional treatments. The method estimates weights that balance observed confounders by removing their dependence on the functional treatment and uses a dual formulation of the weighting problem for efficient unconstrained optimization. The framework supports scalar, binary, and functional outcomes, as well as functional covariates, and can be used to estimate marginal causal effects in settings with time-varying exposures. The methodology follows Ciardulli, S., Fontana, N., Vantini, S., and Ieva, F. (2026) ``Generalized propensity score weighting for functional causal inference framework'' <doi:10.48550/arXiv.2608.03200>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports fda (>= 6.0.0), ggplot2 (>= 3.4.0), tidyr (>= 1.2.0), MASS (>= 7.3-0), wCorr, patchwork (>= 1.1.0), progress (>= 1.2.0), stats, utils

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Author Nicole Fontana [aut, cre], Simone Ciardulli [aut], Simone Vantini [ths], Francesca Ieva [ths]

Maintainer Nicole Fontana <nicole.fontana@polimi.it>

Repository CRAN

Date/Publication 2026-08-23 10:20:02 UTC

Contents

`fps_effect_estimation` *Estimate causal effect of a functional treatment*

Description

Given the FPS weights produced by `fps_weighting`, estimates the causal effect function $\hat{\mu}(t)$ (scalar/binary outcome) or the causal effect surface $\hat{\mu}(s,t)$ (functional outcome) via weighted least squares. Optional bootstrap inference is available.

Usage

```
fps_effect_estimation(
  outcome,
  fps_object,
  outcome_t_grid = NULL,
  outcome_domain = NULL,
  outcome_domain_name = "t",
  outcome_nbasis = NULL,
  outcome_pve = 0.95,
  treatment_pve = NULL,
  bootstrap = FALSE,
  B = 1000,
  alpha = 0.05,
  true_beta = NULL,
  seed = NULL
)
```

Arguments

`outcome`	Numeric vector (scalar/binary, length n) or n x T matrix (functional outcome).
`fps_object`	Object of class `"fps_weighting"` returned by `fps_weighting`.
`outcome_t_grid`	Numeric vector. Observation grid for functional outcome. Required when `outcome` is a matrix.

`outcome_domain` Numeric `c(a, b)`. Domain for functional outcome. Required when `outcome` is a matrix.

`outcome_domain_name` Character. Name of the outcome domain (default `"s"`).

`outcome_nbasis` Integer or `NULL`. B-spline basis size for the outcome FPCA. Auto-selected if `NULL`.

`outcome_pve` Numeric in (0, 1]. PVE threshold for outcome FPCA (default 0.95).

`treatment_pve` Numeric or `NULL`. If not `NULL`, re-runs FPCA on the treatment with this PVE threshold for the outcome estimation step (allowing L* != L). Default `NULL` (reuses `fps_object$fpca_treatment`).

`bootstrap` Logical. If `TRUE`, compute bootstrap confidence intervals (default `FALSE`).

`B` Integer. Number of bootstrap resamples (default 1000).

`alpha` Numeric. Significance level for bootstrap CIs (default 0.05).

`true_beta` Optional. Numeric vector (scalar/binary) or matrix (functional) containing the true causal effect, used for visual comparison in plots and error metrics in `summary`.

`seed` Integer or `NULL`. Random seed for bootstrap reproducibility.

Details

Scalar and binary outcomes. The treatment FPC scores A_i are regressed on the outcome using `lm` (scalar or binary, linear probability model) with the FPS weights. The estimated effect function is then reconstructed as $\hat{\mu}(t) = \sum_k \hat{\mu}_k \phi_k(t)$.

Functional outcome. For each outcome FPC component j, the regression $c_{ij} \sim A_i$ is solved with the FPS weights. The causal surface is reconstructed as $\hat{\mu}(s,t) = \Phi_X \hat{B} \Phi_Y^\top$ where $\hat{B}$ collects the regression coefficients.

Bootstrap CIs. Scalar/binary: residual bootstrap, B resamples. Functional: pairs bootstrap, B resamples. Pointwise reflected-percentile confidence intervals are returned.

Value

An object of class `"fps_effect_estimation"`, a named list with:

outcome_type Character: `"scalar"`, `"binary"`, or `"functional"`.

beta Estimated causal effect, evaluated on `t_grid` (numeric vector for scalar/binary) or on the `t_grid` x `outcome_t_grid` grid (matrix for functional).

beta_unweighted Same as `beta` but from unweighted regression.

fpca_treatment FPCA of the treatment used in estimation.

fpca_outcome `NULL` for scalar/binary; FPCA list for functional outcome.

ci_lower, ci_upper `NULL` if `bootstrap = FALSE`; otherwise lower and upper bootstrap CI bounds (same shape as `beta`).

alpha Significance level used.

t_grid Treatment domain grid.

outcome_t_grid `NULL` for scalar/binary; outcome grid for functional.

domain_name Treatment domain name.

outcome_domain_name Outcome domain name.

true_beta Passed through unchanged.

fps_object The input `fps_weighting` object.

call The matched call.

See Also

`fps_weighting`, `simulate_fps_data`

Examples

```
dat <- simulate_fps_data(n = 2000, setting = "LL", seed = 1)

w <- fps_weighting(dat$X, dat$t_grid, c(0, 1), covariates = dat$C)

# Scalar outcome, no bootstrap
eff <- fps_effect_estimation(dat$Y, w, true_beta = dat$true_beta)
plot(eff, type = "effect")
plot(eff, type = "comparison")

# With bootstrap (small B for illustration)
eff_boot <- fps_effect_estimation(dat$Y, w, bootstrap = TRUE, B = 100,
                                  true_beta = dat$true_beta, seed = 42)
plot(eff_boot, type = "significance")
```

`fps_weighting` *Estimate functional propensity score weights*

Description

Computes covariate-balancing weights for a functional treatment using the empirical-likelihood balancing framework of Ciardulli, S. and Fontana, N. (2026). Treatment is represented via FPCA (Karhunen–Loeve expansion); The treatment is first represented via Functional Principal Component Analysis (FPCA) through its Karhunen–Loeve expansion truncated at rank L; the resulting FPC scores and observed confounders are balanced by solving the dual of the empirical-likelihood problem via the BFGS quasi-Newton algorithm. Functional covariates enter the balancing step through their own FPC scores.

Usage

```
fps_weighting(
  treatment,
  treat_grid = NULL,
  treat_domain = NULL,
```

```
  domain_name = "s",
  nbasis = NULL,
  pve = 0.95,
  covariates,
  cov_grids = NULL,
  cov_domains = NULL,
  cov_nbasis = NULL,
  cov_pve = 0.95,
  normalize = TRUE,
  tol = 1e-08,
  maxit = 1000
)
```

Arguments

`treatment`	n x T numeric matrix of observed treatment trajectories, or an `fd` object from the **fda** package.
`treat_grid`	Numeric vector of length T giving the observation grid of the treatment. Required when `treatment` is a matrix; inferred automatically when `treatment` is an `fd` object.
`treat_domain`	Numeric vector `c(a, b)` specifying the domain of the treatment. If `NULL` (default) and `treatment` is a matrix, the domain is inferred as `c(min(treat_grid), max(treat_grid))`. Inferred automatically when `treatment` is an `fd` object.
`domain_name`	Character string naming the domain variable (default `"s"`). Used in axis labels and domain-overlap checks.
`nbasis`	Integer. Number of B-spline basis functions used for the treatment FPCA. If `NULL` (default), chosen automatically as `max(10, round(0.6 * length(treat_grid)))`.
`pve`	Numeric in (0, 1]. Proportion of variance explained threshold for the treatment FPCA (default 0.95).
`covariates`	Either (a) an n x p numeric matrix of scalar covariates, or (b) a named list with elements `scalar` (n x p matrix, may be `NULL`) and `functional` (a list of matrices or `fd` objects representing functional covariates).
`cov_grids`	A list of numeric vectors (one per functional covariate) giving the observation grids. Required if `covariates$functional` contains matrices; inferred from the domain when `NULL`.
`cov_domains`	A list of numeric vectors `c(a, b)` (one per functional covariate). If `NULL`, inferred from `cov_grids` extremes.
`cov_nbasis`	A list of integers (or `NULL`) for B-spline basis sizes of functional covariates. Defaults to auto-selection.
`cov_pve`	Numeric in (0, 1]. PVE threshold for functional covariate FPCA (default 0.95).
`normalize`	Logical. If `TRUE` (default), standardise FPC scores and confounders before the dual optimisation.
`tol`	Relative convergence tolerance for the BFGS optimiser (default 1e-8).
`maxit`	Maximum number of BFGS iterations (default 1000).

Value

An object of class "fps_weighting", which is a named list with the following components:

weights Numeric vector of length n. Positive weights summing to 1.

fpca_treatment List returned by the internal FPCA routine, containing FPC scores (scr), eigenfunctions (efn), mean function (mean), eigenvalues (eval), variance proportions (varprop), cumulative PVE (perc), raw pca.fd object (pca_fd), number of components retained (L), and the t_grid and domain used.

fpca_covariates List of FPCA results for functional covariates, or NULL if none were supplied.

scalar_covariates The n x p scalar covariate matrix used.

conf_matrix Full augmented confounder matrix fed to the optimiser (scalar covariates column-bound with FPC scores of functional covariates).

convergence Convergence code from optim (0 = success).

domain_name The domain name passed via domain_name.

call The matched call.

See Also

fps_effect_estimation, simulate_fps_data

Examples

```
dat <- simulate_fps_data(n = 2000, setting = "LL", seed = 1)

# Scalar covariates only (treat_domain inferred from treat_grid)
w <- fps_weighting(
  treatment   = dat$X,
  treat_grid  = dat$t_grid,
  covariates  = dat$C
)
print(w)
plot(w, type = "balance")

# Include one functional covariate
w2 <- fps_weighting(
  treatment   = dat$X,
  treat_grid  = dat$t_grid,
  treat_domain = c(0, 1),
  covariates  = list(scalar = dat$C, functional = list(dat$D)),
  cov_grids   = list(dat$t_grid)
)
plot(w2, type = "balance")
```

`plot.fps_effect_estimation` *Plot diagnostics for fps_effect_estimation objects*

Description

Plot diagnostics for fps_effect_estimation objects

Usage

```
## S3 method for class 'fps_effect_estimation'
plot(
  x,
  type = "effect",
  point = NULL,
  which_domain = "treatment",
  alpha = NULL,
  max_efn = 4,
  ...
)
```

Arguments

`x`	An object of class `"fps_effect_estimation"`.
`type`	Character. One of:
	`"effect"` Weighted estimate of $\mu(t)$ (or $\mu(s,t)$) with CI ribbon and optional true-mu overlay. Uses bootstrap CI if available, otherwise analytical SE-based CI.
	`"comparison"` Weighted vs unweighted side by side, both with CI ribbons.
	`"fpca_treatment"` Scree + eigenfunctions of the treatment FPCA used in estimation.
	`"fpca_outcome"` Scree + eigenfunctions of the outcome FPCA (functional outcome only).
	`"bootstrap_slice"` 1-D slice(s) of the causal effect surface with bootstrap CI band. Pass a scalar or vector to `point`; multiple points yield a patchwork panel. Functional outcome only.
	`"significance"` Regions / points where the CI excludes 0.
`point`	Numeric scalar or vector. Time point(s) at which to slice the effect surface (for `type = "bootstrap_slice"`).
`which_domain`	Character. Either `"treatment"` or `"outcome"` (for `type = "bootstrap_slice"`).
`alpha`	Numeric. Significance level; defaults to `x$alpha`.
`max_efn`	Integer. Maximum eigenfunctions shown in FPCA panels.
`...`	Ignored.

Value

A **ggplot2** object.

`plot.fps_weighting` *Plot diagnostics for fps_weighting objects*

Description

Plot diagnostics for fps_weighting objects

Usage

```
## S3 method for class 'fps_weighting'
plot(x, type = "balance", max_efn = 4, ...)
```

Arguments

`x`	An object of class `"fps_weighting"`.
`type`	Character. One of: `"balance"` Absolute Pearson correlations between each treatment FPC score and each confounder, before (red) and after (blue) weighting, displayed as connected point-line chart. `"fpca_treatment"` Scree plot (eigenvalues + cumulative PVE) and panel of leading eigenfunctions for the treatment FPCA. `"fpca_covariates"` Same as `"fpca_treatment"` for each functional covariate (returns a list of **ggplot2** objects). `"weights"` Boxplot of the estimated weights with a horizontal reference line at the uniform weight 1/n.
`max_efn`	Integer. Maximum number of eigenfunctions to show in FPCA panels (default 4).
`...`	Ignored.

Value

A **ggplot2** object (or a list of them for `type = "fpca_covariates"`).

print.fps_effect_estimation *Print method for fps_effect_estimation objects*

Description

Print method for fps_effect_estimation objects

Usage

```
## S3 method for class 'fps_effect_estimation'
print(x, ...)
```

Arguments

x	An object of class "fps_effect_estimation".
...	Ignored.

Value

Invisibly returns x.

print.fps_weighting *Print method for fps_weighting objects*

Description

Print method for fps_weighting objects

Usage

```
## S3 method for class 'fps_weighting'
print(x, ...)
```

Arguments

x	An object of class "fps_weighting".
...	Ignored.

Value

Invisibly returns x.

`simulate_fps_data` *Simulate functional propensity score data*

Description

Generates a synthetic dataset for testing and illustrating the FPScausal workflow.

Usage

```
simulate_fps_data(
  n = 200,
  setting = c("LL", "LN", "NL", "NN"),
  outcome_type = c("scalar", "functional"),
  p_scalar = 3,
  include_functional_cov = TRUE,
  domain = c(0, 1),
  seed = NULL
)
```

Arguments

`n`	Integer. Number of subjects (default 200).
`setting`	Character. One of '"LL"', '"LN"', '"NL"', '"NN"', where the first letter controls the treatment-confounder relationship and the second controls the confounder-outcome relationship. Default '"LL"'.
`outcome_type`	Character. Either '"scalar"' or '"functional"'. Default '"scalar"'.
`p_scalar`	Integer. Number of scalar confounders. Default 3.
`include_functional_cov`	Logical. If 'TRUE' (default), include one functional covariate D(t) in the returned list.
`domain`	Numeric vector `c(a, b)` giving the time domain of the functional objects. Default `c(0, 1)`, matching the paper's simulation study. Change this to use a different time range (e.g. `c(50, 70)` for age in years). The 51 evaluation points are always equally spaced within `domain`.
`seed`	Integer or NULL. Random seed for reproducibility.

Details

Treatment X_i(t) is built from 6 Fourier eigenfunctions with eigenvalues (16, 12, 8, 4, 1, 0.5). **Scalar confounders** C_i are 3-dimensional vectors whose relationship to X's FPC scores is either linear or quadratic. An optional **functional covariate** D_i(t) is generated from 4 Fourier components. The **scalar outcome** is Y_i = 1 + integral(beta(t) * X_i(t)) + g(C_i) + N(0,25), and the **functional outcome** is Y_i(t) = mu0(t) + integral(mu(s,t) * X_i(s) ds) + h(D_i) + GP_error.

The four settings ("LL", "LN", "NL", "NN") vary whether the treatment-confounder ("L"inear / "N"onlinear) and confounder-outcome ("L"inear / "N"onlinear) relationships are linear or quadratic.

Value

A named list with:

X n x 51 matrix. Observed treatment trajectories on [0,1].

Y If ‘outcome_type = "scalar"‘: numeric vector of length n. If ‘outcome_type = "functional"‘: n x 51 matrix.

C n x p_scalar matrix. Scalar confounders.

D n x 51 matrix. Functional covariate (if ‘include_functional_cov = TRUE‘, else ‘NULL‘).

t_grid Numeric vector of 51 equally-spaced points on [0,1].

true_beta True causal effect. For scalar outcome: numeric vector of length 51. For functional outcome: 51 x 51 matrix mu(s,t).

setting The ‘setting‘ argument used.

outcome_type The ‘outcome_type‘ argument used.

Examples

```
dat <- simulate_fps_data(n = 100, setting = "LL", outcome_type = "scalar",
                         seed = 42)
str(dat)
```

---

summary.fps_effect_estimation

Summary method for fps_effect_estimation objects

Description

Summary method for fps_effect_estimation objects

Usage

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

Arguments

`object`	An object of class `"fps_effect_estimation"`.
`...`	Ignored.

Value

Invisibly returns `object`.

summary.fps_weighting *Summary method for fps_weighting objects*

Description

Summary method for fps_weighting objects

Usage

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

Arguments

object	An object of class "fps_weighting".
...	Ignored.

Value

Invisibly returns object.

Index