Package: Meanimiles (via r-universe)

October 19, 2026

Title Estimation of Meanimiles

Version 0.1.0

Description Provides a comprehensive suite of estimation tools for meanimiles, a general class of (risk) functionals. This package includes nonparametric estimators for univariate meanimile evaluation, copula-based estimation for portfolio risk aggregation (full parametric, semiparametric, and nonparametric), and novel estimators for meanimiles in regression settings. Following the articles D. Debrauwer, I. Gijbels, and K. Herrmann (2025) <doi:10.1214/25-EJS2391>, D. Debrauwer and I. Gijbels (2026) <doi:10.1007/s00184-026-01022-9>.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://github.com/DieterDebrauwer/Meanimiles

BugReports https://github.com/DieterDebrauwer/Meanimiles/issues

Depends R (>= 4.5.0)

Imports copula (>= 1.1.7), fitdistrplus (>= 1.2.6), stats

Suggests knitr, rmarkdown

Config/pak/sysreqs libgsl0-dev

Repository https://dieterdebrauwer.r-universe.dev

Date/Publication 2026-04-16 14:18:23 UTC

RemoteUrl https://github.com/dieterdebrauwer/meanimiles

RemoteRef HEAD

RemoteSha 1410186e144d840d39211cf2e0200707de1434dd

Contents

ConditionalCDF 2
MeanimileEstimator 3
MeanimileLinearRegression 4
MeanimileNonparametricRegression 5
PortfolioMeanimile 7

Index **9**

`ConditionalCDF` *Estimate conditional cumulative distribution function*

Description

Computes a doubly smoothed Nadaraya-Watson estimator for the conditional cumulative distribution function.

Usage

```
ConditionalCDF(Y_eval, X_eval, Y_train, X_train, h_y, h_x)
```

Arguments

`Y_eval`	A numeric vector of Y values where the CDF should be evaluated.
`X_eval`	A numeric matrix of X values where the CDF should be evaluated.
`Y_train`	A numeric vector of training Y values.
`X_train`	A numeric matrix of training X values.
`h_y`	A numeric scalar representing the bandwidth for the response variable Y.
`h_x`	A numeric vector representing the bandwidths for the covariates X.

Value

A numeric vector of estimated CDF probabilities.

`MeanimileEstimator` *Estimate a univariate meanimile*

Description

Calculates the empirical meanimile for a given (univariate) data sample, weight function, and loss derivative.

Usage

```
MeanimileEstimator(
  d_tau,
  tau,
  lossDerivative,
  delta = NULL,
  sample,
  grid_size = 1000
)
```

Arguments

`d_tau`	A function representing the density function of the distributional weight function D_tau.
`tau`	A numeric risk level/index (e.g., 0.95). Argument of d_tau function.
`lossDerivative`	Function defining the derivative of the loss function.
`delta`	Optional numeric parameter for the loss function (default:`NULL`).
`sample`	A numeric vector of univariate data.
`grid_size`	An integer defining the resolution of the fallback grid (default: 1000).

Value

A numeric scalar representing the estimated meanimile.

References

D. Debrauwer, I. Gijbels, and K. Herrmann. (2026) On a general class of functionals: Statistical inference and application to risk measures Electronic Journal of Statistics doi:https://doi.org/10.1214/25-EJS2391

Examples

```
set.seed(123)
x <- rnorm(100)
d_tau_expectile <- function(u, tau) {
  1
}
lossDerivativeExpectiles <- function(x, c, delta) {
```

```
  -2 * ifelse(x > c, delta, 1 - delta) * (x - c)
}
MeanimileEstimator(d_tau_expectile, tau = 0, lossDerivativeExpectiles,
    delta = 0.95, sample = x)
```

`MeanimileLinearRegression`

Estimate linear conditional meanimile

Description

Estimates the parameters of a linear conditional meanimile model using a two-fold sample splitting procedure to mitigate bias from the conditional CDF estimation. This function acts as a unified wrapper, allowing the user to seamlessly switch between the square loss formulation and the general loss formulation.

Usage

```
MeanimileLinearRegression(
  X,
  Y,
  d_tau,
  tau,
  h_x,
  h_y,
  loss_type = c("square", "general"),
  lossDerivative = NULL,
  delta = NULL
)
```

Arguments

`X`	A numeric matrix of covariates (n x d).
`Y`	A numeric vector of the response variable.
`d_tau`	A function representing the density function of the distributional weight function D_tau.
`tau`	A numeric risk level/index. Argument of d_tau function.
`h_x`	A numeric vector of bandwidths for the covariates X.
`h_y`	A numeric scalar bandwidth for the response variable Y.
`loss_type`	A character string specifying the loss formulation. Must be either `"square"` (default) or `"general"`.
`lossDerivative`	A function defining the derivative of the loss function. Required only if `loss_type = "general"`.
`delta`	Optional numeric parameter for the loss function. Used only if `loss_type = "general"`.

Value

A numeric vector of estimated regression coefficients (including the intercept).

Examples

```
set.seed(123)
n <- 400
X <- matrix(rnorm(n * 2), ncol = 2)
Y <- 0.9 * X[, 1] + 0.5 * X[, 2] + rnorm(n)
d_tau_minvar <- function(u, tau) {
  (tau + 1) * u^tau
}
lossDerivativeExpectiles <- function(x, c, delta) {
  -2 * ifelse(x > c, delta, 1 - delta) * (x - c)
}

# 1. Square Loss
MeanimileLinearRegression(X, Y,
  d_tau = d_tau_minvar, tau = 2,
  h_x = c(0.4, 0.4), h_y = 0.4, loss_type = "square"
)

# 2. General Loss
MeanimileLinearRegression(X, Y,
  d_tau = d_tau_minvar, tau = 2,
  h_x = c(0.4, 0.4), h_y = 0.4, loss_type = "general",
  lossDerivative = lossDerivativeExpectiles, delta = 0.9
)
```

MeanimileNonparametricRegression

Estimate nonparametric conditional meanimile

Description

Estimates a conditional meanimile in nonparametric way. This function acts as a unified wrapper, allowing the user to seamlessly switch between the square loss formulation and the general loss formulation.

Usage

```
MeanimileNonparametricRegression(
  X,
  Y,
  X0,
  d_tau,
  tau,
  h_x_loc,
```

```
  h_x_cdf,
  h_y_cdf,
  loss_type = c("square", "general"),
  lossDerivative = NULL,
  delta = NULL
)
```

Arguments

X	A numeric matrix of covariates (n x d).
Y	A numeric vector of the response variable.
X0	A numeric matrix of evaluation points (m x d) where the curve should be estimated.
d_tau	A function representing the density function of the distributional weight function D_tau.
tau	A numeric risk level/index. Argument of d_tau function.
h_x_loc	A numeric vector of local spatial bandwidths for the covariates X.
h_x_cdf	A numeric vector of bandwidths for the covariates X used in the CDF estimation.
h_y_cdf	A numeric scalar bandwidth for the response variable Y used in the CDF estimation.
loss_type	Character string specifying the loss formulation ("square" or "general").
lossDerivative	A function defining the derivative of the loss function. Required for "general".
delta	Optional numeric parameter for the loss function. Only for "general".

Value

A numeric vector of length m, containing the estimated meanimile predictions at each point in X0.

Examples

```
set.seed(123)
n <- 300

# 1. Simulate 1D Non-linear Data (Sine Wave)
X <- matrix(runif(n, -2, 2), ncol = 1)
Y <- sin(X[, 1] * 1.5) + rnorm(n, sd = 0.4)

# 2. Define Evaluation Grid
X0 <- matrix(seq(-2, 2, length.out = 50), ncol = 1)

# 3. Define Functions
d_tau_minvar <- function(u, tau) {
  (tau + 1) * u^tau
}
loss_deriv_expectile <- function(x, c, delta) {
  -2 * ifelse(x > c, delta, 1 - delta) * (x - c)
}
```

```
# 4. Fit Square Loss Nonparametric
preds_sq <- MeanimileNonparametricRegression(
  X = X, Y = Y, X0 = X0, d_tau = d_tau_minvar, tau = 2,
  h_x_loc = 0.4, h_x_cdf = 0.4, h_y_cdf = 0.4, loss_type = "square"
)

# 5. Fit General Loss Nonparametric (Expectile)
preds_gen <- MeanimileNonparametricRegression(
  X = X, Y = Y, X0 = X0, d_tau = d_tau_minvar, tau = 2,
  h_x_loc = 0.4, h_x_cdf = 0.4, h_y_cdf = 0.4, loss_type = "general",
  lossDerivative = loss_deriv_expectile, delta = 0.9
)
```

`PortfolioMeanimile` *Estimate portfolio meanimile*

Description

Estimates the risk of an aggregated portfolio using the meanimile framework. Supports parametric and nonparametric modeling of marginal distributions and copula.

Usage

```
PortfolioMeanimile(
  data,
  weights,
  d_tau,
  tau,
  lossDerivative,
  delta = NULL,
  copula_obj = NULL,
  margin_dists = NULL,
  N = 1e+05,
  grid_size = 1000
)
```

Arguments

`data`	A numeric matrix or data.frame (n observations x d assets).
`weights`	A numeric vector of portfolio weights summing to 1.
`d_tau`	A function representing the density function of the distributional weight function D_tau.
`tau`	A numeric risk level/index (e.g., 0.95). Argument of d_tau function.
`lossDerivative`	Function defining the derivative of the loss function.
`delta`	Optional numeric parameter for the loss function (default:NULL).

copula_obj	An unfitted 'copula' object (e.g., copula::gumbelCopula(dim=3)). If NULL, an Empirical Beta Copula is used.
margin_dists	A character vector of base R distribution root names for fitdistrplus::fitdist (e.g., c("norm", "exp", "weibull")). If NULL, empirical margins are used.
N	Integer. Number of Monte Carlo simulations to draw (default: 100 000).
grid_size	An integer defining the resolution of the fallback grid (default: 1000).

Value

A numeric scalar representing the estimated meanimile.

References

D. Debrauwer and I. Gijbels (2026) Copula-based estimation of meanimiles of aggregated risks. Metrika doi:https://doi.org/10.1007/s00184-026-01022-9

Examples

```
gumbel.cop <- copula::gumbelCopula(10 / 9, dim = 3)
myMvdX <- copula::mvdc(
  copula = gumbel.cop, margins = c("norm", "norm", "exp"),
  paramMargins = list(
    list(mean = 0, sd = 1),
    list(mean = 0, sd = 1),
    list(rate = 1)
  )
)
X <- copula::rMvdc(500, myMvdX)

d_tau_expectile <- function(u, tau) {
  1
}
lossDerivativeExpectiles <- function(x, c, delta) {
  -2 * ifelse(x > c, delta, 1 - delta) * (x - c)
}
gumbelCopula <- copula::gumbelCopula(dim = 3)
margin_dists <- c("norm", "norm", "exp")
weights <- c(1 / 3, 1 / 3, 1 / 3)
PortfolioMeanimile(
  data = X, weights, d_tau_expectile, tau = 0, lossDerivativeExpectiles, delta = 0.95,
  copula_obj = gumbelCopula, margin_dists = margin_dists, N = 100000, grid_size = 1000
)
```

Index

ConditionalCDF, 2

MeanimileEstimator, 3
MeanimileLinearRegression, 4
MeanimileNonparametricRegression, 5

PortfolioMeanimile, 7