Package ‘MECfda’

March 6, 2025

Type Package

Title Scalar-on-Function Regression with Measurement Error Correction

Version 0.2.0

Maintainer Heyang Ji <jihx1015@outlook.com>

Description Solve scalar-on-function linear models,
including generalized linear mixed effect model and quantile linear regression model,
and bias correction estimation methods due to measurement error.
Details about the measurement error bias correction methods, see
Luan et al. (2023) <doi:10.48550/arXiv.2305.12624>,
Tekwe et al. (2022) <doi:10.1093/biostatistics/kxac017>,
Zhang et al. (2023) <doi:10.5705/ss.202021.0246>,
Tekwe et al. (2019) <doi:10.1002/sim.8179>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Depends R (>= 2.10),

Imports stats, lme4, quantreg, splines, dplyr, MASS, Matrix, gss,
corpcor, fda, magrittr, methods, nlme, glme, mgcv, refund,
pracma

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Heyang Ji [aut, cre, ctb, dtc]
(<https://orcid.org/0009-0001-7494-7227>),
Ufuk Beyaztas [aut, ctb, rev] (<https://orcid.org/0000-0002-5208-4950>),
Nicolas Escobar-Velasquez [com]
(<https://orcid.org/0009-0006-0800-5692>),
Yuanyuan Luan [aut, ctb],
Xiwei Chen [aut, ctb],
Mengli Zhang [aut, ctb],
Roger Zoh [aut, ths],

Lan Xue [aut, ths],
Carmen Tekwe [aut, ths] (<https://orcid.org/0000-0002-1857-2416>)

Repository CRAN

Date/Publication 2025-03-06 22:00:02 UTC

Contents

basis2fun *From the summation series of a functional basis to function value*

Description

Generic function to compute function value from summation series of a functional basis.

Usage

```
basis2fun(object, x)

## S4 method for signature 'bspline_series,numeric'
basis2fun(object, x)

## S4 method for signature 'Fourier_series,numeric'
basis2fun(object, x)

## S4 method for signature 'numericBasis_series,numeric'
basis2fun(object, x)
```

Arguments

object	An object that represents a functional basis.
x	point(s) to take value.

Details

When applied to bspline_series object, equivalent to bsplineSeries2fun.
When applied to Fourier_series object, equivalent to FourierSeries2fun.
When applied to numericBasis_series object, equivalent to numericBasisSeries2fun.

Value

A numeric atomic vector. See bsplineSeries2fun and FourierSeries2fun.

Author(s)

Heyang Ji

bsplineSeries2fun *Compute the value of the b-splines summation series at certain points.*

Description

Compute the function $f(x) = \sum_{i=0}^{k} b_i B_{i,p}(x)$

Usage

```
bsplineSeries2fun(object, x)

## S4 method for signature 'bspline_series,numeric'
bsplineSeries2fun(object, x)
```

Arguments

object	an object of bspline_series class.
x	Value of x.

Value

A numeric atomic vector

Author(s)

Heyang Ji

Examples

```
bsb = bspline_basis(
            Boundary.knots = c(0,24),
            intercept      = TRUE,
            df             = NULL,
            degree         = 3
)
bss = bspline_series(
          coef = c(2,1,1.5,0.5),
          bspline_basis = bsb
)
bsplineSeries2fun(bss,(1:239)/10)
```

`bspline_basis-class` *b-spline basis*

Description

A s4 class that represents a b-spline basis $\{B_{i,p}(x)\}_{i=-p}^{k}$ on the interval $[t_0, t_{k+1}]$, where $B_{i,p}(x)$ is defined as

$$B_{i,0}(x) = \begin{cases} I_{(t_i,t_{i+1}]}(x), & i = 0, 1, \ldots, k \\ 0, & i < 0 \; or \; i > k \end{cases}$$

$$B_{i,r}(x) = \frac{x - t_i}{t_{i+r} - t_i} B_{i,r-1}(x) + \frac{t_{i+r+1} - x}{t_{i+r+1} - t_{i+1}} B_{i+1,r-1}(x)$$

For all the discontinuity points of $B_{i,r}$ $(r > 0)$ in the interval (t_0, t_k), let the value equals its limit, which means

$$B_{i,r}(x) = \lim_{t \to x} B_{i,r}(t)$$

Slots

`Boundary.knots` boundary of the domain of the splines (start and end), which is t_0 and t_{k+1}. Default is $[0, 1]$. See `Boundary.knots` in `bs`.

`knots` knots of the splines, which is $(t_1, \ldots, t_k)$, equally spaced sequence is chosen by the function automatically with equal space ($t_j = t_0 + j \cdot \frac{t_{k+1} - t_0}{k+1}$) when not assigned. See `knots` in `bs`.

`intercept` Whether an intercept is included in the basis, default value is `TRUE`, and must be `TRUE`. See `intercept` `bs`.

`df` degree of freedom of the basis, which is the number of the splines, equal to $p+k+1$. By default $k = 0$, and `df` $= p + 1$. See `df` `bs`.

`degree` degree of the splines, which is the degree of piecewise polynomials p, default value is 3. See `degree` in `bs`.

Author(s)

Heyang Ji

Examples

```
bsb = bspline_basis(
            Boundary.knots = c(0,24),
            intercept      = TRUE,
            df             = NULL,
            degree         = 3
)
```

`bspline_basis_expansion` *B-splines basis expansion for functional variable data*

Description

For a function $f(t), t \in \Omega$, and a basis function sequence $\{\rho_k\}_{k \in \kappa}$, basis expansion is to compute $\int_\Omega f(t)\rho_k(t)dt$. Here we do basis expansion for all $f_i(t), t \in \Omega = [t_0, t_0 + T]$ in functional variable data, $i = 1, \ldots, n$. We compute a matrix $(b_{ik})_{n \times p}$, where $b_{ik} = \int_\Omega f(t)\rho_k(t)dt$. The basis used here is the b-splines basis, $\{B_{i,p}(x)\}_{i=-p}^{k}$, $x \in [t_0, t_{k+1}]$, where $t_{k+1} = t_0 + T$ and $B_{i,p}(x)$ is defined as

$$B_{i,0}(x) = \begin{cases} I_{(t_i, t_{i+1}]}(x), & i = 0, 1, \ldots, k \\ 0, & i < 0 \; or \; i > k \end{cases}$$

$$B_{i,r}(x) = \frac{x - t_i}{t_{i+r} - t_i} B_{i,r-1}(x) + \frac{t_{i+r+1} - x}{t_{i+r+1} - t_{i+1}} B_{i+1,r-1}(x)$$

Usage

```
bspline_basis_expansion(object, n_splines, bs_degree)

## S4 method for signature 'functional_variable,integer'
bspline_basis_expansion(object, n_splines, bs_degree)
```

Arguments

`object`	a `functional_variable` class object.
`n_splines`	the number of splines, equal to $k + p + 1$. See `df` in `bs`.
`bs_degree`	the degree of the piecewise polynomial of the b-splines. See `degree` in `bs`.

Value

Returns a numeric matrix, $(b_{ik})_{n \times p}$, where $b_{ik} = \int_\Omega f(t)\rho_k(t)dt$.

Author(s)

Heyang Ji

bspline_series-class *b-splines summation series.*

Description

A s4 class that represents the summation $\sum_{i=0}^{k} b_i B_{i,p}(x)$ by a bspline_basis object and coefficients b_i $(i = 0, \ldots, k)$.

Slots

`coef` coefficients of the b-splines, b_i $(i = 0, \ldots, k)$.

`bspline_basis` a `bspline_basis` object, represents the b-splines basis used, $\{B_{i,p}(x)\}_{i=-p}^{k}$.

Author(s)

Heyang Ji

Examples

```
bsb = bspline_basis(
            Boundary.knots = c(0,24),
            intercept      = TRUE,
            df             = NULL,
            degree         = 3
)
bss = bspline_series(
          coef = c(2,1,1.5,0.5),
          bspline_basis = bsb
)
```

dim,functional_variable-method *Extract dimensionality of functional data.*

Description

Extract the dimensionality of slot X of functional_variable object.

Usage

```
## S4 method for signature 'functional_variable'
dim(x)
```

Arguments

x	a `functional_variable` object.

Value

Retruns a 2-element numeric vector.

Author(s)

Heyang Ji

Examples

```
fv = functional_variable(X=array(rnorm(12),dim = 4:3),period = 3)
dim(fv)
```

---

extractCoef *Method of class Fourier_series to extract Fourier coefficients*

Description

Method of class Fourier_series to extract Fourier coefficients

Usage

```
extractCoef(object)

## S4 method for signature 'Fourier_series'
extractCoef(object)
```

Arguments

`object`	an object of `Fourier_series` class.

Value

A list that contains the coefficients.

Author(s)

Heyang Ji

Examples

```
fsc = Fourier_series(
           double_constant = 0.5,
           cos = c(0,0.3),
           sin = c(1,0.7),
           k_cos = 1:2,
           )
 extractCoef(fsc)
```

fc.beta — *Extract the value of coefficient parameter function*

Description

Generic function to extract the value of coefficient parameter function of the covariates from linear model with functional covariates at some certain points.

Usage

```
fc.beta(object, ...)

## S4 method for signature 'fcRegression'
fc.beta(object, FC = 1, t_points = NULL)

## S4 method for signature 'fcQR'
fc.beta(object, FC = 1, t_points = NULL)
```

Arguments

`object`	An object that represents a functional covariates linear model.
`...`	More arguments.
`FC`	An integer, represent the ordinal number of the functional covariate. Default is 1, which is take the first functional covariate.
`t_points`	Sequence of the measurement (time) points.

Value

A numeric atomic vector

Author(s)

Heyang Ji

fcQR — *Solve quantile regression models with functional covariate(s).*

Description

Fit a quantile regression models below

$$Q_{Y_i|X_i,Z_i}(\tau) = \sum_{l=1}^{L} \int_\Omega \beta_l(\tau,t) X_{li}(t) dt + (1, Z_i^T)\gamma$$

where $Q_{Y_i}(\tau) = F^{-1}_{Y_i|X_i,Z_i}(\tau)$ is the τ-th quantile of Y_i given $X_i(t)$ and Z_i, $\tau \in (0,1)$.
Model allows one or multiple functional covariate(s) as fixed effect(s), and zero, one, or multiple scalar-valued covariate(s).

Usage

```
fcQR(
  Y,
  FC,
  Z,
  formula.Z,
  tau = 0.5,
  basis.type = c("Fourier", "Bspline"),
  basis.order = 6L,
  bs_degree = 3
)
```

Arguments

`Y`	Response variable, can be an atomic vector, a one-column matrix or data frame, recommended form is a one-column data frame with column name
`FC`	Functional covariate(s), can be a "functional_variable" object or a matrix or a data frame or a list of these object(s)
`Z`	Scalar covariate(s), can be `NULL` or not input (when there's no scalar covariate), an atomic vector (when only one scalar covariate), a matrix or data frame, recommended form is a data frame with column name(s)
`formula.Z`	A formula without the response variable, contains only scalar covariate(s). If not assigned, include all scalar covariates and intercept term.
`tau`	Quantile $\tau \in (0, 1)$, default is 0.5. See `rq`.
`basis.type`	Type of funtion basis. Can only be assigned as one type even if there is more than one functional covariates. Available options: `'Fourier'` or `'Bspline'` or `'FPC'`, represent Fourier basis, b-spline basis, and functional principal component (FPC) basis respectively. For the detailed form for Fourier, b-splines, and FPC basis, see `fourier_basis_expansion`, `bspline_basis_expansion`, and `FPC_basis_expansion`.
`basis.order`	Indicate number of the function basis. When using Fourier basis $\frac{1}{2}, \sin kt, \cos kt, k = 1, \ldots, K$, `basis.order` is the number K. When using b-splines basis $\{B_{i,p}(x)\}_{i=-p}^{k}$, `basis.order` is the number of splines, equal to $k + p + 1$. When using FPC basis, `basis.order` is the number of functional principal components. (same as arguement `df` in `bs`.) May set a individual number for each functional covariate. When the element of this argument is less than the number of functional covariates, it will be used recursively.
`bs_degree`	Degree of the piecewise polynomials if use b-splines basis, default is 3. See `degree` in `bs`.

Value

fcQR returns an object of class "fcQR". It is a list that contains the following elements.

`regression_result`
: Result of the regression.

`FC.BasisCoefficient`
: A list of Fourier_series or bspline_series object(s), represents the functional linear coefficient(s) of the functional covariates.

`function.basis.type`
: Type of funtion basis used.

`basis.order`
: Same as in the arguemnets.

`data`
: Original data.

`bs_degree`
: Degree of the splines, returned only if b-splines basis is used.

Author(s)

Heyang Ji

Examples

```
data(MECfda.data.sim.0.0)
res = fcQR(FC = MECfda.data.sim.0.0$FC, Y=MECfda.data.sim.0.0$Y, Z=MECfda.data.sim.0.0$Z,
          basis.order = 5, basis.type = c('Bspline'))
```

`fcRegression` *Solve linear models with functional covariate(s)*

Description

Function to fit (generalized) linear model with functional covariate(s). Model allows one or multiple functional covariate(s) as fixed effect(s), and zero, one, or multiple scalar-valued fixed or random effect(s).

Usage

```
fcRegression(
  Y,
  FC,
  Z,
  formula.Z,
  family = gaussian(link = "identity"),
  basis.type = c("Fourier", "Bspline", "FPC"),
  basis.order = 6L,
  bs_degree = 3
)
```

Arguments

`Y` — Response variable, can be an atomic vector, a one-column matrix or data frame, recommended form is a one-column data frame with column name.

`FC` — Functional covariate(s), can be a "functional_variable" object or a matrix or a data frame or a list of these object(s).

`Z` — Scalar covariate(s), can be `NULL` or not input (when there's no scalar covariate), an atomic vector (when only one scalar covariate), a matrix or data frame, recommended form is a data frame with column name(s).

`formula.Z` — A formula without the response variable, contains only scalar covariate(s) (or intercept), use the format of lme4 package if random effects exist. e.g. `~ Z_1 + (1|Z_2)`. (See `lmer` and `glmer`) If not assigned, include all scalar covariates and intercept term as fixed effects.

`family` — A description of the error distribution and link function to be used in the model, see `family`.

`basis.type` — Type of funtion basis. Can only be assigned as one type even if there is more than one functional covariates. Available options: `'Fourier'` or `'Bspline'` or `'FPC'`, represent Fourier basis, b-spline basis, and functional principal component (FPC) basis respectively. For the detailed form for Fourier, b-splines, and FPC basis, see `fourier_basis_expansion`, `bspline_basis_expansion`, and `FPC_basis_expansion`.

`basis.order` — Indicate number of the function basis. When using Fourier basis $\frac{1}{2}, \sin kt, \cos kt, k = 1, \ldots, p_f$, `basis.order` is the number p_f. When using b-splines basis $\{B_{i,p}(x)\}_{i=-p}^{k}$, `basis.order` is the number of splines, equal to $k + p + 1$. (same as arguement `df` in `bs`.) When using FPC basis, `basis.order` is the number of functional principal components. May set a individual number for each functional covariate. When the element of this argument is less than the number of functional covariates, it will be used recursively.

`bs_degree` — Degree of the piecewise polynomials if use b-splines basis, default is 3. See `degree` in `bs`.

Details

Solve linear models with functional covariates below

$$g(E(Y_i|X_i, Z_i)) = \sum_{l=1}^{L} \int_{\Omega_l} \beta_l(t) X_{li}(t) dt + (1, Z_i^T)\gamma$$

where the scalar-valued covariates can be fixed or random effect or doesn't exist (may do not contain scalar-valued covariates).

Value

fcRegression returns an object of class "fcRegression". It is a list that contains the following elements.

`regression_result` — Result of the regression.

FC.BasisCoefficient
: A list of Fourier_series or bspline_series or numeric_basis object(s), represents the functional linear coefficient(s) of the functional covariates.

function.basis.type
: Type of funtion basis used.

basis.order
: Same as in the arguemnets.

data
: Original data.

bs_degree
: Degree of the splines, returned only if b-splines basis is used.

Author(s)

Heyang Ji

Examples

```
data(MECfda.data.sim.0.0)
res = fcRegression(FC = MECfda.data.sim.0.0$FC, Y=MECfda.data.sim.0.0$Y, Z=MECfda.data.sim.0.0$Z,
                   basis.order = 5, basis.type = c('Bspline'),
                   formula.Z = ~ Z_1 + (1|Z_2))
```

FourierSeries2fun — *Compute the value of the Fourier summation series*

Description

Compute the value of the Fourier summation series

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{p_a} a_k \cos\left(\frac{2\pi}{T}k(x-t_0)\right) + \sum_{k=1}^{p_b} b_k \sin\left(\frac{2\pi}{T}k(x-t_0)\right), \qquad x \in [t_0, t_0+T]$$

at some certain point(s).

Usage

```
FourierSeries2fun(object, x)

## S4 method for signature 'Fourier_series,numeric'
FourierSeries2fun(object, x)
```

Arguments

object
: an object of Fourier_series class.

x
: Value of x.

Value

A numeric atomic vector

Author(s)

Heyang Ji

Examples

```
fsc = Fourier_series(
           double_constant = 0.5,
           cos = c(0,0.3),
           sin = c(1,0.7),
           k_cos = 1:2,
           )
          FourierSeries2fun(fsc,1:5)
```

`fourier_basis_expansion` *Fourier basis expansion for functional variable data*

Description

For a function $f(x), x \in \Omega$, and a basis function sequence $\{\rho_k\}_{k \in \kappa}$, basis expansion is to compute $\int_\Omega f(t)\rho_k(t)dt$. Here we do basis expansion for all $f_i(t), t \in \Omega = [t_0, t_0 + T]$ in functional variable data, $i = 1, \ldots, n$. We compute a matrix $(b_{ik})_{n \times p}$, where $b_{ik} = \int_\Omega f(t)\rho_k(t)dt$. The basis used here is the Fourier basis,

$$\frac{1}{2},\ \cos(\frac{2\pi}{T}k[x - t_0]),\ \sin(\frac{2\pi}{T}k[x - t_0])$$

where $x \in [t_0, t_0 + T]$ and $k = 1, \ldots, p_f$.

Usage

```
fourier_basis_expansion(object, order_fourier_basis)

## S4 method for signature 'functional_variable,integer'
fourier_basis_expansion(object, order_fourier_basis)
```

Arguments

`object` a `functional_variable` class object.

`order_fourier_basis` the order of Fourier basis, p_f.

Value

Returns a numeric matrix, $(b_{ik})_{n \times p}$, where $b_{ik} = \int_\Omega f(t)\rho_k(t)dt$.

Author(s)

Heyang Ji

`Fourier_series-class` *s4 class of Fourier summation series*

Description

A s4 class that represents the linear combination of Fourier basis functions below:

$$\frac{a_0}{2} + \sum_{k=1}^{p_a} a_k \cos\left(\frac{2\pi}{T}k(x - t_0)\right) + \sum_{k=1}^{p_b} b_k \sin\left(\frac{2\pi}{T}k(x - t_0)\right), \qquad x \in [t_0, t_0 + T]$$

Details

If not assigned, $t_0 = 0, T = 2\pi$. If not assigned, k_cos and k_sin equals 1, 2, 3, ...

Slots

`double_constant` value of a_0.

`cos` values of coefficients of $\cos$ waves, a_k.

`sin` values of coefficients of $\sin$ waves, b_k.

`k_cos` values of k corresponding to the coefficients of $\cos$ waves

`k_sin` values of k corresponding to the coefficients of $\sin$ waves

`t_0` left end of the domain interval, t_0

`period` length of the domain interval, T.

Author(s)

Heyang Ji

Examples

```
fsc = Fourier_series(
          double_constant = 0.5,
          cos = c(0,0.3),
          sin = c(1,0.7),
          k_cos = 1:2,
          )
```

`FPC_basis_expansion` *Functional principal component basis expansion for functional variable data*

Description

For a function $f(t), t \in \Omega$, and a basis function sequence $\{\rho_k\}_{k\in\kappa}$, basis expansion is to compute $\int_\Omega f(t)\rho_k(t)dt$. Here we do basis expansion for all $f_i(t), t \in \Omega = [t_0, t_0+T]$ in functional variable data, $i = 1, \dots, n$. We compute a matrix $(b_{ik})_{n\times p}$, where $b_{ik} = \int_\Omega f(t)\rho_k(t)dt$. The basis we use here is the functional principal component (FPC) basis induced by the covariance function of the functional variable. Suppose $K(s,t) \in L^2(\Omega \times \Omega)$, $f(t) \in L^2(\Omega)$. Then K induces an linear operator $\mathcal{K}$ by

$$(\mathcal{K}f)(x) = \int_\Omega K(t,x)f(t)dt$$

If $\xi(\cdot) \in L^2(\Omega)$ such that

$$\mathcal{K}\xi = \lambda\xi$$

where $\lambda \in C$, we call ξ an eigenfunction/eigenvector of $\mathcal{K}$, and λ an eigenvalue associated with ξ. For a stochastic process $\{X(t), t \in \Omega\}$ we call the orthogonal basis $\{\xi_k\}_{k=1}^\infty$ corresponding to eigenvalues $\{\lambda_k\}_{k=1}^\infty$ ($\lambda_1 \geq \lambda_2 \geq \dots$), induced by

$$K(s,t) = \text{Cov}(X(t), X(s))$$

a functional principal component (FPC) basis for $L^2(\Omega)$.

Usage

```
FPC_basis_expansion(object, npc)

## S4 method for signature 'functional_variable,integer'
FPC_basis_expansion(object, npc)
```

Arguments

`object`	a `functional_variable` class object. The minimum and maximum of the slot `t_points` should be respectively equal to the slot `t_0` and slot `t_0` plus slot `period`.
`npc`	The number of functional principal components. See `npc` in `fpca.sc`.

Value

Returns a numeric matrix, $(b_{ik})_{n\times p}$, with an extra attribute `numeric_basis`, which represents the FPC basis. The attribute `numeric_basis` is a `numeric_basis` object. See `numeric_basis`. The slot `basis_function` is also a numeric matrix, denoted as $(\zeta_{jk})_{m\times p}$

$$b_{ik} = \int_\Omega f(t)\xi_k(t)dt$$

$$\zeta_{jk} = \xi_k(t_j)$$

Author(s)

Heyang Ji

Examples

```
n<-50; ti<-seq(0,1,length.out=101)
X<-t(sin(2*pi*ti)%*%t(rnorm(n,0,1)))
object = functional_variable(X = X, t_0 = 0, period = 1, t_points = ti)
a = FPC_basis_expansion(object,3L)
dim(a)
```

`functional_variable-class`
Function-valued variable data.

Description

A s4 class that represents data of a function-valued variable. The format is $f_i(t),\ t \in \Omega = [t_0, t_0 + T]$ where i is the observation (subject) index, t represents the measurement (time) points.

Slots

`X` a matrix $(x_{ij})_{n\times m}$, where $x_{ij} = f_i(t_j)$, represents the value of $f_i(t_j)$, each row represent an observation (subject), each column is corresponding to a measurement (time) point.

`t_0` start of the domain (time period), t_0. Default is 0.

`period` length of the domain (time period), T. Default is 1.

`t_points` sequence of the measurement points, $(t_1, \ldots, t_m)$. Default is $t_k = t_0 + \frac{(2k-1)T}{2(m+1)}$.

Author(s)

Heyang Ji

Examples

```
X = array(rnorm(12),dim = 4:3)
functional_variable(X=X,period = 3)
```

ME.fcLR_IV *Bias correction method of applying linear regression to one functional covariate with measurement error using instrumental variable.*

Description

See detailed model in reference

Usage

```
ME.fcLR_IV(
  data.Y,
  data.W,
  data.M,
  t_interval = c(0, 1),
  t_points = NULL,
  CI.bootstrap = FALSE
)
```

Arguments

data.Y	Response variable, can be an atomic vector, a one-column matrix or data frame, recommended form is a one-column data frame with column name.
data.W	A dataframe or matrix, represents W, the measurement of X. Each row represents a subject. Each column represent a measurement (time) point.
data.M	A dataframe or matrix, represents M, the instrumental variable. Each row represents a subject. Each column represent a measurement (time) point.
t_interval	A 2-element vector, represents an interval, means the domain of the functional covariate. Default is `c(0,1)`, represent interval $[0, 1]$.
t_points	Sequence of the measurement (time) points, default is `NULL`.
CI.bootstrap	Whether to return the confidence using bootstrap method. Default is `FALSE`.

Value

Returns a ME.fcLR_IV class object. It is a list that contains the following elements.

beta_tW	Parameter estimates.
CI	Confidence interval, returnd only when CI.bootstrap is TRUE.

References

Tekwe, Carmen D., et al. "Instrumental variable approach to estimating the scalar-on-function regression model w ith measurement error with application to energy expenditure assessment in childhood obesity." Statistics in medicine 38.20 (2019): 3764-3781.

Examples

```
data(MECfda.data.sim.0.3)
res = ME.fcLR_IV(data.Y = MECfda.data.sim.0.3$Y,
                 data.W = MECfda.data.sim.0.3$W,
                 data.M = MECfda.data.sim.0.3$M)
```

ME.fcQR_CLS — *Bias correction method of applying quantile linear regression to dataset with one functional covariate with measurement error using corrected loss score method.*

Description

Zhang et al. proposed a new corrected loss function for a partially functional linear quantile model with functional measurement error in this manuscript. They established a corrected quantile objective function of the observed measurement that is an unbiased estimator of the quantile objective function that would be obtained if the true measurements were available. The estimators of the regression parameters are obtained by optimizing the resulting corrected loss function. The resulting estimator of the regression parameters is shown to be consistent.

Usage

```
ME.fcQR_CLS(
  data.Y,
  data.W,
  data.Z,
  tau = 0.5,
  t_interval = c(0, 1),
  t_points = NULL,
  grid_k,
  grid_h,
  degree = 45,
  observed_X = NULL
)
```

Arguments

data.Y	Response variable, can be an atomic vector, a one-column matrix or data frame, recommended form is a one-column data frame with column name.
data.W	A 3-dimensional array, represents W, the measurement of X. Each row represents a subject. Each column represent a measurement (time) point. Each layer represents an observation.
data.Z	Scalar covariate(s), can be not input or NULL (when there's no scalar covariate), an atomic vector (when only one scalar covariate), a matrix or data frame, recommended form is a data frame with column name(s).
tau	Quantile $\tau \in (0, 1)$, default is 0.5.

`t_interval` A 2-element vector, represents an interval, means the domain of the functional covariate. Default is c(0,1), represent interval $[0, 1]$.

`t_points` Sequence of the measurement (time) points, default is `NULL`

`grid_k` An atomic vector, of which each element is candidate number of basis.

`grid_h` A non-zero-value atomic vector, of which each element is candidate value of tunning parameter.

`degree` Used in computation for derivative and integral, defult is 45, large enough for most scenario.

`observed_X` For estimating parametric variance. Default is `NULL`.

Value

Returns a ME.fcQR_CLS class object. It is a list that contains the following elements.

`estimated_beta_hat` Estimated coefficients from corrected loss function (including functional part)

`estimated_beta_t` Estimated functional curve

`SE_est` Estimated parametric variance. Returned only if observed_X is not `NULL`.

`estimated_Xbasis` The basis matrix we used

`res_naive` results of naive method

References

Zhang, Mengli, et al. "PARTIALLY FUNCTIONAL LINEAR QUANTILE REGRESSION WITH MEASUREMENT ERRORS." Statistica Sinica 33 (2023): 2257-2280.

Examples

```
data(MECfda.data.sim.0.1)

res = ME.fcQR_CLS(data.Y = MECfda.data.sim.0.1$Y,
                 data.W = MECfda.data.sim.0.1$W,
                data.Z = MECfda.data.sim.0.1$Z,
                tau = 0.5,
                grid_k = 4:7,
                grid_h = 1:2)
```

ME.fcQR_IV.SIMEX *Bias correction method of applying quantile linear regression to dataset with one functional covariate with measurement error using instrumental variable.*

Description

Perform a two-stage strategy to correct the measurement error of a function-valued covariate and then fit a linear quantile regression model. In the first stage, an instrumental variable is used to estimate the covariance matrix associated with the measurement error. In the second stage, simulation extrapolation (SIMEX) is used to correct for measurement error in the function-valued covariate. See detailed model in the reference.

Usage

```
ME.fcQR_IV.SIMEX(
  data.Y,
  data.W,
  data.Z,
  data.M,
  tau = 0.5,
  t_interval = c(0, 1),
  t_points = NULL,
  formula.Z,
  basis.type = c("Fourier", "Bspline"),
  basis.order = NULL,
  bs_degree = 3
)
```

Arguments

data.Y	Response variable, can be an atomic vector, a one-column matrix or data frame, recommended form is a one-column data frame with column name.
data.W	A dataframe or matrix, represents W, the measurement of X. Each row represents a subject. Each column represent a measurement (time) point.
data.Z	Scalar covariate(s), can be not input or NULL (when there's no scalar covariate), an atomic vector (when only one scalar covariate), a matrix or data frame, recommended form is a data frame with column name(s).
data.M	A dataframe or matrix, represents M, the instrumental variable. Each row represents a subject. Each column represent a measurement (time) point.
tau	Quantile $\tau \in (0, 1)$, default is 0.5.
t_interval	A 2-element vector, represents an interval, means the domain of the functional covariate. Default is c(0,1), represent interval $[0, 1]$.
t_points	Sequence of the measurement (time) points, default is NULL.

formula.Z A formula without the response variable, contains only scalar covariate(s), no random effects. If not assigned, include all scalar covariates and intercept term.

basis.type Type of funtion basis. Can only be assigned as one type even if there is more than one functional covariates. Available options: 'Fourier' or 'Bspline', represent Fourier basis and b-spline basis respectively. For the detailed form for Fourier and b-splines basis, see `fourier_basis_expansion` and `bspline_basis_expansion`.

basis.order Indicate number of the function basis. When using Fourier basis $\frac{1}{2}, \sin kt, \cos kt, k = 1, \ldots, K$, basis.order is the number K. When using b-splines basis $\{B_{i,p}(x)\}_{i=-p}^{k}$, basis.order is the number of splines, equal to $k+p+1$. (same as arguement `df` in `bs`.) May set a individual number for each functional covariate. When the element of this argument is less than the number of functional covariates, it will be used recursively.

bs_degree Degree of the piecewise polynomials if use b-splines basis, default is 3. See `degree` in `bs`.

Value

Returns a ME.fcQR_IV.SIMEX class object. It is a list that contains the following elements.

coef.X A Fourier_series or bspline_series object, represents the functional coefficient parameter of the functional covariate.

coef.Z The estimate of the linear coefficients of the scalar covariates.

coef.all Original estimate of linear coefficients.

function.basis.type Type of funtion basis used.

basis.order Same as in the input arguements.

t_interval A 2-element vector, represents an interval, means the domain of the functional covariate.

t_points Sequence of the measurement (time) points.

formula Regression model.

formula.Z formula object contains only the scalar covariate(s).

zlevels levels of the non-continuous scalar covariate(s).

References

Tekwe, Carmen D., et al. "Estimation of sparse functional quantile regression with measurement error: a SIMEX approach." Biostatistics 23.4 (2022): 1218-1241.

Examples

```
data(MECfda.data.sim.0.2)

res = ME.fcQR_IV.SIMEX(data.Y = MECfda.data.sim.0.2$Y,
                       data.W = MECfda.data.sim.0.2$W,
                       data.Z = MECfda.data.sim.0.2$Z,
                       data.M = MECfda.data.sim.0.2$M,
```

```
                        tau = 0.5,
                        basis.type = 'Bspline')
```

`ME.fcRegression_MEM` *Use UP_MEM or MP_MEM substitution to apply (generalized) linear regression with one functional covariate with measurement error.*

Description

The Mixed-effect model (MEM) approach is a two-stage-based method that employs functional mixed-effects models. It allows us to delve into the nonlinear measurement error model, where the relationship between the true and observed measurements is not constrained to be linear, and the distribution assumption on the observed measurement is relaxed to encompass the exponential family rather than being limited to the Gaussian distribution. The MEM approach employs point-wise (UP_MEM) and multi-point-wise (MP_MEM) estimation procedures to avoid potential computational complexities caused by analyses of multi-level functional data and computations of potentially intractable and complex integrals.

Usage

```
ME.fcRegression_MEM(
  data.Y,
  data.W,
  data.Z,
  method = c("UP_MEM", "MP_MEM", "average"),
  t_interval = c(0, 1),
  t_points = NULL,
  d = 3,
  family.W = c("gaussian", "poisson"),
  family.Y = "gaussian",
  formula.Z,
  basis.type = c("Fourier", "Bspline"),
  basis.order = NULL,
  bs_degree = 3,
  smooth = FALSE,
  silent = TRUE
)
```

Arguments

`data.Y`	Response variable, can be an atomic vector, a one-column matrix or data frame, recommended form is a one-column data frame with column name.
`data.W`	A 3-dimensional array, represents W, the measurement of X. Each row represents a subject. Each column represent a measurement (time) point. Each layer represents an observation.

- `data.Z` — Scalar covariate(s), can be not input or `NULL` (when there's no scalar covariate), an atomic vector (when only one scalar covariate), a matrix or data frame, recommended form is a data frame with column name(s).
- `method` — The method to construct the substitution X. Available options: 'UP_MEM', 'MP_MEM', 'average'.
- `t_interval` — A 2-element vector, represents an interval, means the domain of the functional covariate. Default is c(0,1), represent interval $[0, 1]$.
- `t_points` — Sequence of the measurement (time) points, default is `NULL`.
- `d` — The number of time points involved for MP_MEM (default and miniumn is 3).
- `family.W` — Distribution of W given X, Available options: "gaussian","poisson".
- `family.Y` — A description of the error distribution and link function to be used in the model, see `family`.
- `formula.Z` — A formula without the response variable, contains only scalar covariate(s), use the format of lme4 package if random effects exist. e.g. ~ Z_1 + (1|Z_2). If not assigned, include all scalar covariates and intercept term as fixed effects.
- `basis.type` — Type of function basis. Can only be assigned as one type even if there is more than one functional covariates. Available options: 'Fourier' or 'Bspline', represent Fourier basis and b-spline basis respectively. For the detailed form for Fourier and b-splines basis, see `fourier_basis_expansion` and `bspline_basis_expansion`.
- `basis.order` — Indicate number of the function basis. When using Fourier basis $\frac{1}{2}, \sin kt, \cos kt, k = 1, \ldots, K$, basis.order is the number K. When using b-splines basis $\{B_{i,p}(x)\}_{i=-p}^{k}$, basis.order is the number of splines, equal to $k + p + 1$. (same as arguement `df` in `bs`.) May set a individual number for each functional covariate. When the element of this argument is less than the number of functional covariates, it will be used recursively.
- `bs_degree` — Degree of the piecewise polynomials if use b-splines basis, default is 3. See `degree` in `bs`.
- `smooth` — Whether to smooth the substitution of X. Default is `FALSE`.
- `silent` — Whether not to show the state of the running of the function. Default is `TRUE`.

Value

Returns a `fcRegression` object. See `fcRegression`.

References

Luan, Yuanyuan, et al. "Scalable regression calibration approaches to correcting measurement error in multi-level generalized functional linear regression models with heteroscedastic measurement errors." arXiv preprint arXiv:2305.12624 (2023).

Examples

```
data(MECfda.data.sim.0.1)
res = ME.fcRegression_MEM(data.Y = MECfda.data.sim.0.1$Y,
                          data.W = MECfda.data.sim.0.1$W,
                          data.Z = MECfda.data.sim.0.1$Z,
```

```
                    method = 'UP_MEM',
                    family.W = "gaussian",
                    basis.type = 'Bspline')
```

`MECfda.data.sim.0.0` *Simulated data*

Description

Simulated data

`MECfda.data.sim.0.1` *Simulated data*

Description

Simulated data

`MECfda.data.sim.0.2` *Simulated data*

Description

Simulated data

`MECfda.data.sim.0.3` *Simulated data*

Description

Simulated data

`MECfda.data.sim.1.0` *Simulated data*

Description

Simulated data

MECfda.data.sim.1.1 *Simulated data*

Description

Simulated data

MECfda.data.sim.1.2 *Simulated data*

Description

Simulated data

MECfda.data.sim.1.3 *Simulated data*

Description

Simulated data

MECfda_simDataGen_fcReg *Simulation Data Generation: Scalar-on-function Regression*

Description

Generate data set for scalar-on-function regression

Usage

```
MECfda_simDataGen_fcReg(
  N = 100,
  distribution = c("Gaussian", "Bernoulli"),
  t_interval,
  t_points,
  n_t = 100,
  seed = 0
)
```

Arguments

`N`	Sample size.
`distribution`	Conditional distribution of response varaible given the covariate $(Y_i \| X_i(t), Z_i)$. There are two options: `'Gaussian'` and `'Bernoulli'`.
`t_interval`	A 2-element vector, represents an interval, means the domain of the functional covariate. Default is `c(0,1)`, represent interval $[0, 1]$.
`t_points`	the measurement points of functional variables, should be numeric vector.
`n_t`	Number of measurement time points. Overwritten if argument `t_points` is assigned.
`seed`	Pseudo-random number generation seed.

Value

return a list with following elements.

`Y`	An atomic vector of response variable
`Z`	A dataframe with a binary and a continuous scalar-valued covariate.
`FC`	A list of two 'functional_variable' class object.
`t_interval`	Same as in the input argument.
`t_points`	Sequence of the measurement (time) points.

Examples

```
dat_sim = MECfda_simDataGen_fcReg(100,"Bernoulli")
res = fcRegression(FC = dat_sim$FC, Y=dat_sim$Y, Z=dat_sim$Z,
                   basis.order = 3, basis.type = c('Fourier'),
                   family = binomial(link = "logit"))
```

`MECfda_simDataGen_ME` *Simulation Data Generation: Measurement Error Bias Correction of Scalar-on-function Regression*

Description

Generate data set for measurement error bias correction methods for scalar-on-function regression in package MECfda

Usage

```
MECfda_simDataGen_ME(
  N = 100,
  J_W = 7,
  forwhich = c("MEM", "IV", "CLS", "IV.SIMEX"),
  t_interval = c(0, 1),
  n_t = 24,
  seed = 0
)
```

Arguments

N	Sample size.
J_W	Number of repeated measurement (period), if applicable.
forwhich	For which method of measurement error bias correction method the data set is generated. There are two options: 'MEM','IV','CLS','IV.SIMEX'.
t_interval	A 2-element vector, represents an interval, means the domain of the functional covariate. Default is `c(0,1)`, represent interval $[0, 1]$.
n_t	Number of measurement time points.
seed	Pseudo-random number generation seed.

Value

return a list that possibly contains following elements.

Y	An atomic vector of response variable
Z	A dataframe with a binary and a continuous scalar-valued covariate.
W	Observed values of function-valued covariate.
M	Instrumental vairable.
t_interval	Same as in the input argument.
t_points	Sequence of the measurement (time) points.

Examples

```
for (i in 1:4) {MECfda_simDataGen_ME(forwhich = c('MEM','IV','CLS','IV.SIMEX')[i])}
```

MEM_X_hat

Get MEM substitution for (generalized) linear regression with one functional covariate with measurement error.

Description

The function to get the data of $\hat{X}_i(t)$ using the mixed model based measurement error bias correction method proposed by Luan et al. See ME.fcRegression_MEM

Usage

```
MEM_X_hat(
  data.W,
  method = c("UP_MEM", "MP_MEM", "average"),
  d = 3,
  family.W = c("gaussian", "poisson"),
  smooth = FALSE
)
```

Arguments

`data.W`	A 3-dimensional array, represents W, the measurement of X. Each row represents a subject. Each column represent a measurement (time) point. Each layer represents an observation.
`method`	The method to construct the substitution X. Available options: `'UP_MEM'`, `'MP_MEM'`, 'average'.
`d`	The number of time points involved for MP_MEM (default and miniumn is 3).
`family.W`	Distribution of W given X, Available options: `"gaussian"`, `"poisson"`.
`smooth`	Whether to smooth the substitution of X. Default is `FALSE`.

Value

A numeric value matrix of $\hat{X}_i(t)$.

References

Luan, Yuanyuan, et al. "Scalable regression calibration approaches to correcting measurement error in multi-level generalized functional linear regression models with heteroscedastic measurement errors." arXiv preprint arXiv:2305.12624 (2023).

Examples

```
data(MECfda.data.sim.0.1)
X_hat = MEM_X_hat(data.W = MECfda.data.sim.0.1$W,
                  method = 'UP_MEM',
                  family.W = "gaussian")
```

---

numericBasisSeries2fun

Compute the value of the basis function summation series at certain points.

Description

Compute the function $f(x) = \sum_{k=0}^{p} c_k \rho_k(x)$, $x \in \Omega$

Usage

```
numericBasisSeries2fun(object, x)

## S4 method for signature 'numericBasis_series,numeric'
numericBasisSeries2fun(object, x)
```

Arguments

object	an object of numericBasis_series class.
x	Value of x.

Value

A numeric atomic vector

Author(s)

Heyang Ji

Examples

```
t_0 = 0
period = 1
t_points = seq(0.05,0.95,length.out = 19)
nb = numeric_basis(
  basis_function = cbind(1/2,cos(2*pi*t_points),sin(2*pi*t_points)),
  t_points       = t_points,
  t_0            = t_0,
  period         = period
)
ns = numericBasis_series(coef = c(0.8,1.2,1.6),numeric_basis = nb)
numericBasisSeries2fun(ns,seq(0,1,length.out = 51))
```

---

numericBasis_series-class

Linear combination of a sequence of basis functions represented numerically

---

Description

A linear combination of basis function $\{\rho_k\}_{k=1}^p$,

$$\sum_{k=1}^{p} c_k \rho_k(t).$$

Slots

coef linear coefficient $\{c_k\}_{k=1}^p$.

numeric_basis $\{\rho_k\}_{k=1}^p$ represented by a numeric_basis object. See numeric_basis.

Author(s)

Heyang Ji

Examples

```
t_0 = 0
period = 1
t_points = seq(0.05,0.95,length.out = 19)
nb = numeric_basis(
  basis_function = cbind(1,cos(t_points),sin(t_points)),
  t_points       = t_points,
  t_0            = t_0,
  period         = period
)
ns = numericBasis_series(coef = c(0.8,1.2,1.6),numeric_basis = nb)
```

`numeric_basis-class` *Numeric representation of a function basis*

Description

A s4 class that numerically represents a basis of linear space of function.
$\{\rho_k\}_{k=1}^{\infty}$ denotes a basis of function linear space. Some times the basis cannot be expressed analytically. But we can numerically store the space by the value of a finite subset of the basis functions at some certain points in the domain, $\rho_k(t_j), k = 1, \ldots, p, j = 1, \ldots, m$. The s4 class is to represent a finite sequence of functions by their values at a finite sequence of points within their domain, in which all the functions have the same domain and the domain is an interval.

Details

The units of a basis of a linear space should be linearly independent. But the program doesn't check the linear dependency of the basis function when a `numeric_basis` object is initialized.

Slots

`basis_function` matrix of the value of the functions, $(\zeta_{jk})_{m\times p}$, where $\zeta_{ik} = \rho_k(t_j), j = 1, \ldots, m, k = 1, \ldots, p$. Each row of the matrix is corresponding to a point of t. Each column of the matrix is corresponding to a basis function.

`t_points` a numeric atomic vector, represents the points in the domains of the function where the function values are taken. The jth element is corresponding to jth row of slot `basis_function`.

`t_0` left end of the domain interval.

`period` length of the domain interval.

Author(s)

Heyang Ji

Examples

```
t_0 = 0
period = 1
t_points = seq(0.05,0.95,length.out = 19)
numeric_basis(
  basis_function = cbind(1/2,cos(t_points),sin(t_points)),
  t_points       = t_points,
  t_0            = t_0,
  period         = period
)
```

---

numeric_basis_expansion

Numeric basis expansion for functional variable data

Description

For a function $f(t), t \in \Omega$, and a basis function sequence $\{\rho_k\}_{k \in \kappa}$, basis expansion is to compute $\int_\Omega f(t)\rho_k(t)dt$. Here we do basis expansion for all $f_i(t), t \in \Omega = [t_0, t_0 + T]$ in functional variable data, $i = 1, \ldots, n$. We compute a matrix $(b_{ik})_{n\times p}$, where $b_{ik} = \int_\Omega f(t)\rho_k(t)dt$. The basis we use here is numerically represented by the value of basis functions at some points in the domain.

Usage

```
numeric_basis_expansion(object, num_basis)

## S4 method for signature 'functional_variable,numeric_basis'
numeric_basis_expansion(object, num_basis)
```

Arguments

`object`	a `functional_variable` class object. The minimum and maximum of the slot `t_points` should be respectively equal to the slot `t_0` and slot `t_0` plus slot `period`.
`num_basis`	a `numeric_basis` class object, representing the function basis. See `numeric_basis`.

Value

Returns a numeric matrix, $(b_{ik})_{n\times p}$, with an extra attribute `numeric_basis`, which is the `numeric_basis` object input by the argument `num_basis`.

Author(s)

Heyang Ji

`plot,bspline_series-method`
Plot b-splines basis summation series.

Description

Plot b-splines basis summation series.

Usage

```
## S4 method for signature 'bspline_series'
plot(x)
```

Arguments

x	A `bspline_series` object.

Value

No return value. Generate a scatter plot.

Author(s)

Heyang Ji

Examples

```
bsb = bspline_basis(
Boundary.knots = c(0,24),
intercept      = TRUE,
df             = NULL,
degree         = 3
)
bss = bspline_series(
coef = c(2,1,1.5,3),
bspline_basis = bsb
)
plot(bss)
```

plot,Fourier_series-method

Plot Fourier basis summation series.

Description

Plot Fourier basis summation series.

Usage

```
## S4 method for signature 'Fourier_series'
plot(x)
```

Arguments

x A Fourier_series object.

Value

No return value. Generate a scatter plot.

Author(s)

Heyang Ji

Examples

```
fsc = Fourier_series(
double_constant = 0.5,
cos = c(0,0.3),
sin = c(1,0.7),
k_cos = 1:2,
)
plot(fsc)
```

plot,numericBasis_series-method

Plot numeric basis function summation series.

Description

Plot numeric basis function summation series.

Usage

```
## S4 method for signature 'numericBasis_series'
plot(x)
```

Arguments

x	A numericBasis_series object.

Author(s)

Heyang Ji

Examples

```
t_0 = 0
period = 1
t_points = seq(0.05,0.95,length.out = 19)
nb = numeric_basis(
  basis_function = cbind(1/2,cos(2*pi*t_points),sin(2*pi*t_points)),
  t_points       = t_points,
  t_0            = t_0,
  period         = period
)
ns = numericBasis_series(coef = c(0.8,1.2,1.6),numeric_basis = nb)
plot(ns)
```

---

predict.fcQR *Predicted values based on fcQR object*

Description

Predicted values based on the Quantile linear model with functional covariates represented by a "fcQR" class object.

Usage

```
## S3 method for class 'fcQR'
predict(object, newData.FC, newData.Z = NULL, ...)
```

Arguments

object	A fcQR class object produced by fcQR.
newData.FC	A atomic vector or a matrix or a dataframe or a functional_variable class object or a list of objects above. See argument FC in fcRegression.
newData.Z	A dataframe or a matrix or a atomic vector. See argument Z in fcRegression.
...	Further arguments passed to or from other methods predict.rq.

Details

If no new data is input, will return the fitted value.

Value

See predict.rq.

Author(s)

Heyang Ji

---

predict.fcRegression *Predicted values based on fcRegression object*

Description

Predicted values based on the linear model with functional covariates represented by a "fcRegression" class object.

Usage

```
## S3 method for class 'fcRegression'
predict(object, newData.FC, newData.Z = NULL, ...)
```

Arguments

object	A fcRegression class object produced by fcRegression.
newData.FC	A atomic vector or a matrix or a dataframe or a functional_variable class object or a list of objects above. See argument FC in fcRegression.
newData.Z	A dataframe or a matrix or a atomic vector. See arguement Z in fcRegression.
...	Further arguments passed to or from other methods, including predict.lm, predict.glm, predict.merMod.

Details

If no new data is input, will return the fitted value.

Value

See predict.lm, predict.glm, predict.merMod.

Author(s)

Heyang Ji

Examples

```
data(MECfda.data.sim.0.0)
res = fcRegression(FC = MECfda.data.sim.0.0$FC, Y=MECfda.data.sim.0.0$Y, Z=MECfda.data.sim.0.0$Z,
                   basis.order = 5, basis.type = c('Bspline'),
                   formula.Z = ~ Z_1 + (1|Z_2))
data(MECfda.data.sim.1.0)
predict(object = res, newData.FC = MECfda.data.sim.1.0$FC,newData.Z = MECfda.data.sim.1.0$Z)
```

Index