Package ‘bkmr’

October 12, 2022

Title Bayesian Kernel Machine Regression

Version 0.2.2


URL https://github.com/jenfb/bkmr

BugReports https://github.com/jenfb/bkmr/issues

Depends R (>= 3.1.2)

License GPL-2

Imports dplyr, magrittr, nlme, fields, truncnorm, tidyr, MASS, tmvtnorm, tibble

RoxygenNote 7.1.2

NeedsCompilation no

Author Jennifer F. Bobb [aut, cre]

Maintainer Jennifer F. Bobb <jenniferfederbobb@gmail.com>

Repository CRAN

Date/Publication 2022-03-28 07:30:01 UTC

R topics documented:

ComputePostmeanHnew ......................................................... 2
ExtractEsts ................................................................. 3
ExtractPIPs ................................................................. 4
ExtractSamps ............................................................... 5
InvestigatePrior ............................................................ 6
kmbayes ..................................................................... 7
OverallRiskSummaries ..................................................... 10
PlotPriorFits ................................................................. 11
PredictorResponseBivar .................................................... 12
PredictorResponseBivarLevels ........................................... 14
ComputePostmeanHnew

**Description**

Compute the posterior mean and variance of \( h \) at a new predictor values

**Usage**

```r
ComputePostmeanHnew(
  fit,
  y = NULL,
  Z = NULL,
  X = NULL,
  Znew = NULL,
  sel = NULL,
  method = "approx"
)
```

**Arguments**

- `fit` An object containing the results returned by a the `kmbayes` function
- `y` a vector of outcome data of length \( n \).
- `Z` an \( n \)-by-\( M \) matrix of predictor variables to be included in the \( h \) function. Each row represents an observation and each column represents a predictor.
- `X` an \( n \)-by-\( K \) matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
- `Znew` matrix of new predictor values at which to predict new \( h \), where each row represents a new observation. If set to NULL then will default to using the observed exposures \( Z \).
- `sel` selects which iterations of the MCMC sampler to use for inference; see details
- `method` method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details
Details

- If `method == "approx"`, the argument `sel` defaults to the second half of the MCMC iterations.
- If `method == "exact"`, the argument `sel` defaults to keeping every 10 iterations after dropping the first 50% of samples, or if this results in fewer than 100 iterations, then 100 iterations are kept.

For guided examples and additional information, go to https://jenfb.github.io/bkmr/overview.html

Value

A list of length two containing the posterior mean vector and posterior variance matrix.

Examples

```r
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)

med_vals <- apply(Z, 2, median)
Znew <- matrix(med_vals, nrow = 1)
h_true <- dat$HFun(Znew)
h_est1 <- ComputePostmeanHnew(fitkm, Znew = Znew, method = "approx")
h_est2 <- ComputePostmeanHnew(fitkm, Znew = Znew, method = "exact")
```

Description

Obtain summary statistics of each parameter from the BKMR fit.

Usage

```r
ExtractEsts(fit, q = c(0.025, 0.25, 0.5, 0.75, 0.975), sel = NULL)
```

Arguments

- `fit`: An object containing the results returned by a `kmbayes` function.
- `q`: Vector of quantiles.
- `sel`: Logical expression indicating samples to keep; defaults to keeping the second half of all samples.
Value

a list where each component is a data frame containing the summary statistics of the posterior distribution of one of the parameters (or vector of parameters) being estimated

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
est <- ExtractEests(fitkm)
names(est)
est$beta
```

---

**ExtractPIPs**

*Extract posterior inclusion probabilities (PIPs) from BKMR model fit*

**Description**

Extract posterior inclusion probabilities (PIPs) from Bayesian Kernel Machine Regression (BKMR) model fit

**Usage**

`ExtractPIPs(fit, sel = NULL, z.names = NULL)`

**Arguments**

- `fit` An object containing the results returned by a the `kmbayes` function
- `sel` logical expression indicating samples to keep; defaults to keeping the second half of all samples
- `z.names` optional argument providing the names of the variables included in the h function.

**Details**

For guided examples, go to [https://jenfb.github.io/bkmr/overview.html](https://jenfb.github.io/bkmr/overview.html)
**ExtractSamps**

**Value**

a data frame with the variable-specific PIPs for BKMR fit with component-wise variable selection, and with the group-specific and conditional (within-group) PIPs for BKMR fit with hierarchical variable selection.

**Examples**

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
ExtractPIPs(fitkm)
```

---

**ExtractSamps**

**Description**

Extract samples of each parameter from the BKMR fit

**Usage**

`ExtractSamps(fit, sel = NULL)`

**Arguments**

- `fit` An object containing the results returned by a the `kmbayes` function
- `sel` logical expression indicating samples to keep; defaults to keeping the second half of all samples

**Value**

a list where each component contains the posterior samples of one of the parameters (or vector of parameters) being estimated
## First generate dataset
```r
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X
```

## Fit model with component-wise variable selection
```r
# Using only 100 iterations to make example run quickly
# Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
samps <- ExtractSamps(fitkm)
```

---

### InvestigatePrior

**Description**
Investigate the impact of the \( r[m]\) parameters on the smoothness of the exposure-response function \( h(z[m])\).

**Usage**
```
InvestigatePrior(
  y,
  Z,
  X,
  ngrid = 50,
  q.seq = c(2, 1, 1/2, 1/4, 1/8, 1/16),
  r.seq = NULL,
  Drange = NULL,
  verbose = FALSE
)
```

**Arguments**
- `y` a vector of outcome data of length \( n \).
- `Z` an \( n \)-by-\( M \) matrix of predictor variables to be included in the \( h \) function. Each row represents an observation and each column represents a predictor.
- `X` an \( n \)-by-\( K \) matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
- `ngrid` Number of grid points over which to plot the exposure-response function.
q.seq Sequence of values corresponding to different degrees of smoothness in the estimated exposure-response function. A value of q corresponds to fractions of the range of the data over which there is a decay in the correlation $\text{cor}(h[i], h[j])$ between two subjects by 50%.

r.seq sequence of values at which to fix r for estimating the exposure-response function

Drange the range of the $z_m$ data over which to apply the values of q.seq. If not specified, will be calculated as the maximum of the ranges of $z_1$ through $z_M$.

verbose TRUE or FALSE: flag indicating whether to print to the screen which exposure variable and q value has been completed

Details
For guided examples, go to https://jenfb.github.io/bkmr/overview.html

Value
a list containing the predicted values, residuals, and estimated predictor-response function for each degree of smoothness being considered

Examples
```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

priorfits <- InvestigatePrior(y = y, Z = Z, X = X, q.seq = c(2, 1/2, 1/4, 1/16))
PlotPriorFits(y = y, Z = Z, X = X, fits = priorfits)
```

**kmbayes**

Fit Bayesian kernel machine regression

**Description**
Fits the Bayesian kernel machine regression (BKMR) model using Markov chain Monte Carlo (MCMC) methods.

**Usage**
```
kmbayes(
  y,
  Z,
  X = NULL,
  iter = 1000,
)```
family = "gaussian",
id = NULL,
verbose = TRUE,
Znew = NULL,
starting.values = NULL,
control.params = NULL,
varsel = FALSE,
groups = NULL,
knots = NULL,
ztest = NULL,
rmethod = "varying",
est.h = FALSE
)

Arguments

y a vector of outcome data of length n.
Z an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents an predictor.
X an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
iter number of iterations to run the sampler
family a description of the error distribution and link function to be used in the model. Currently implemented for gaussian and binomial families.
id optional vector (of length n) of grouping factors for fitting a model with a random intercept. If NULL then no random intercept will be included.
verbose TRUE or FALSE: flag indicating whether to print intermediate diagnostic information during the model fitting.
Znew optional matrix of new predictor values at which to predict h, where each row represents a new observation. This will slow down the model fitting, and can be done as a post-processing step using SamplePred
starting.values list of starting values for each parameter. If not specified default values will be chosen.
control.params list of parameters specifying the prior distributions and tuning parameters for the MCMC algorithm. If not specified default values will be chosen.
varsel TRUE or FALSE: indicator for whether to conduct variable selection on the Z variables in h
groups optional vector (of length M) of group indicators for fitting hierarchical variable selection if varsel=TRUE. If varsel=TRUE without group specification, component-wise variable selections will be performed.
knots optional matrix of knot locations for implementing the Gaussian predictive process of Banerjee et al. (2008). Currently only implemented for models without a random intercept.
ztest optional vector indicating on which variables in Z to conduct variable selection (the remaining variables will be forced into the model).

rmethod for those predictors being forced into the h function, the method for sampling the r[m] values. Takes the value of ‘varying’ to allow separate r[m] for each predictor; ‘equal’ to force the same r[m] for each predictor; or ‘fixed’ to fix the r[m] to their starting values.

est.h TRUE or FALSE: indicator for whether to sample from the posterior distribution of the subject-specific effects h_i within the main sampler. This will slow down the model fitting.

Value

an object of class "bkmrfit" (containing the posterior samples from the model fit), which has the associated methods:

- print (i.e., print.bkmrfit)
- summary (i.e., summary.bkmrfit)

References


See Also

For guided examples, go to https://jenfb.github.io/bkmr/overview.html

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
```
OverallRiskSummaries

Calculate overall risk summaries

Description

Compare estimated h function when all predictors are at a particular quantile to when all are at a second fixed quantile

Usage

```r
OverallRiskSummaries(
  fit,
  y = NULL,
  Z = NULL,
  X = NULL,
  qs = seq(0.25, 0.75, by = 0.05),
  q.fixed = 0.5,
  method = "approx",
  sel = NULL
)
```

Arguments

- `fit`: An object containing the results returned by a the kmbayes function
- `y`: a vector of outcome data of length n.
- `Z`: an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents an predictor.
- `X`: an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
- `qs`: vector of quantiles at which to calculate the overall risk summary
- `q.fixed`: a second quantile at which to compare the estimated h function
- `method`: method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details
- `sel`: selects which iterations of the MCMC sampler to use for inference; see details

Details

- If `method == "approx"`, the argument `sel` defaults to the second half of the MCMC iterations.
- If `method == "exact"`, the argument `sel` defaults to keeping every 10 iterations after dropping the first 50% of samples, or if this results in fewer than 100 iterations, than 100 iterations are kept

For guided examples and additional information, go to [https://jenfb.github.io/bkmr/overview.html](https://jenfb.github.io/bkmr/overview.html)
Value

a data frame containing the (posterior mean) estimate and posterior standard deviation of the overall risk measures

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)

risks.overall <- OverallRiskSummaries(fit = fitkm, qs = seq(0.25, 0.75, by = 0.05),
q.fixed = 0.5, method = "exact")
```

PlotPriorFits

Plot of exposure-response function from univariate KMR fit

Description

Plot the estimated $h(z[m])$ estimated from frequentist KMR for $r[m]$ fixed to specific values

Usage

```r
PlotPriorFits(
  y,
  X,
  Z,
  fits,
  which.z = NULL,
  which.q = NULL,
  plot.resid = TRUE,
  ylim = NULL,
  ...
)
```

Arguments

- `y` a vector of outcome data of length `n`.
- `X` an `n`-by-`K` matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
Z an n-by-M matrix of predictor variables to be included in the \( h \) function. Each row represents an observation and each column represents an predictor.

fits output from \texttt{InvestigatePrior}

which.z which predictors (columns in Z) to plot

which.q which q.values to plot; defaults to all possible

plot.resid whether to plot the data points

ylim plotting limits for the y-axis

other plotting arguments

Value

No return value, generates plot

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

priorfits <- InvestigatePrior(y = y, Z = Z, X = X, q.seq = c(2, 1/2, 1/4, 1/16))
PlotPriorFits(y = y, Z = Z, X = X, fits = priorfits)
```

\texttt{PredictorResponseBivar}

\textit{Predict the exposure-response function at a new grid of points}

Description

Predict the exposure-response function at a new grid of points

Usage

\texttt{PredictorResponseBivar(}

\hspace{1cm} fit,

\hspace{1cm} y = \text{NULL},

\hspace{1cm} Z = \text{NULL},

\hspace{1cm} X = \text{NULL},

\hspace{1cm} z.pairs = \text{NULL},

\hspace{1cm} method = "approx",

\hspace{1cm} ngrid = 50,

\hspace{1cm} q.fixed = 0.5,

\hspace{1cm} sel = \text{NULL},

\hspace{1cm} min.plot.dist = 0.5,
\texttt{)}````
center = TRUE,
z.names = colnames(Z),
verbose = TRUE,
...  
)

Arguments

fit An object containing the results returned by a the kmbayes function
y a vector of outcome data of length n.
Z an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents a predictor.
X an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
z.pairs data frame showing which pairs of predictors to plot
method method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details
ngrid number of grid points in each dimension
q.fixed vector of quantiles at which to fix the remaining predictors in Z
sel logical expression indicating samples to keep; defaults to keeping the second half of all samples
min.plot.dist specifies a minimum distance that a new grid point needs to be from an observed data point in order to compute the prediction; points further than this will not be computed
center flag for whether to scale the exposure-response function to have mean zero
z.names optional vector of names for the columns of Z
verbose TRUE or FALSE: flag of whether to print intermediate output to the screen
... other arguments to pass on to the prediction function

Details

For guided examples, go to https://jenfb.github.io/bkmr/overview.html

Value

a long data frame with the name of the first predictor, the name of the second predictor, the value of the first predictor, the value of the second predictor, the posterior mean estimate, and the posterior standard deviation of the estimated exposure response function

Examples

## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)

## Obtain predicted value on new grid of points for each pair of predictors
## Using only a 10-by-10 point grid to make example run quickly
pred.resp.bivar <- PredictorResponseBivar(fit = fitkm, min.plot.dist = 1, ngrid = 10)

PredictorResponseBivarLevels

*Plot cross-sections of the bivariate predictor-response function*

### Description
Function to plot the $h$ function of a particular variable at different levels (quantiles) of a second variable

### Usage

```r
PredictorResponseBivarLevels(
  pred.resp.df,
  Z = NULL,
  qs = c(0.25, 0.5, 0.75),
  both_pairs = TRUE,
  z.names = NULL
)
```

### Arguments
- **pred.resp.df**: object obtained from running the function `PredictorResponseBivar`
- **Z**: an n-by-M matrix of predictor variables to be included in the $h$ function. Each row represents an observation and each column represents an predictor.
- **qs**: vector of quantiles at which to fix the second variable
- **both_pairs**: flag indicating whether, if $h(z_1)$ is being plotted for $z_2$ fixed at different levels, that they should be plotted in the reverse order as well (for $h(z_2)$ at different levels of $z_1$)
- **z.names**: optional vector of names for the columns of $z$

### Details
For guided examples, go to [https://jenfb.github.io/bkmr/overview.html](https://jenfb.github.io/bkmr/overview.html)
PredictorResponseBivarPair

Value

a long data frame with the name of the first predictor, the name of the second predictor, the value of
the first predictor, the quantile at which the second predictor is fixed, the posterior mean estimate,
and the posterior standard deviation of the estimated exposure response function

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)

## Obtain predicted value on new grid of points for each pair of predictors
## Using only a 10-by-10 point grid to make example run quickly
pred.resp.bivar <- PredictorResponseBivar(fit = fitkm, min.plot.dist = 1, ngrid = 10)
pred.resp.bivar.levels <- PredictorResponseBivarLevels(pred.resp.df = pred.resp.bivar, Z = Z, qs = c(0.1, 0.5, 0.9))
```

PredictorResponseBivarPair

*Plot bivariate predictor-response function on a new grid of points*

Description

Plot bivariate predictor-response function on a new grid of points

Usage

```r
PredictorResponseBivarPair(
  fit,
  y = NULL,
  Z = NULL,
  X = NULL,
  whichz1 = 1,
  whichz2 = 2,
  whichz3 = NULL,
  method = "approx",
  prob = 0.5,
  q.fixed = 0.5,
  sel = NULL,
)```
### Arguments

- **fit**: An object containing the results returned by a `kmbayes` function
- **y**: a vector of outcome data of length \(n\).
- **Z**: an \(n\)-by-\(M\) matrix of predictor variables to be included in the \(h\) function. Each row represents an observation and each column represents a predictor.
- **X**: an \(n\)-by-\(K\) matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
- **whichz1**: vector identifying the first predictor that (column of \(Z\)) should be plotted
- **whichz2**: vector identifying the second predictor that (column of \(Z\)) should be plotted
- **whichz3**: vector identifying the third predictor that will be set to a pre-specified fixed quantile (determined by `prob`)
- **method**: method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details
- **prob**: pre-specified quantile to set the third predictor (determined by `whichz3`); defaults to 0.5 (50th percentile)
- **q.fixed**: vector of quantiles at which to fix the remaining predictors in \(Z\)
- **sel**: logical expression indicating samples to keep; defaults to keeping the second half of all samples
- **ngrid**: number of grid points to cover the range of each predictor (column in \(Z\))
- **min.plot.dist**: specifies a minimum distance that a new grid point needs to be from an observed data point in order to compute the prediction; points further than this will not be computed
- **center**: flag for whether to scale the exposure-response function to have mean zero
- **...**: other arguments to pass on to the prediction function

### Value

A data frame with value of the first predictor, the value of the second predictor, the posterior mean estimate, and the posterior standard deviation

### Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
```
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)

## Obtain predicted value on new grid of points
## Using only a 10-by-10 point grid to make example run quickly
pred.resp.bivar12 <- PredictorResponseBivarPair(fit = fitkm, min.plot.dist = 1, ngrid = 10)

---

**PredictorResponseUnivar**

*Plot univariate predictor-response function on a new grid of points*

**Description**

Plot univariate predictor-response function on a new grid of points

**Usage**

```r
PredictorResponseUnivar(
  fit,
  y = NULL,
  Z = NULL,
  X = NULL,
  which.z = 1:ncol(Z),
  method = "approx",
  ngrid = 50,
  q.fixed = 0.5,
  sel = NULL,
  min.plot.dist = Inf,
  center = TRUE,
  z.names = colnames(Z),
  ...
)
```

**Arguments**

- `fit` An object containing the results returned by a the kmbayes function
- `y` a vector of outcome data of length n.
- `Z` an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents an predictor.
- `X` an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
- `which.z` vector identifying which predictors (columns of Z) should be plotted
method method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details

ngrid number of grid points to cover the range of each predictor (column in Z)

q.fixed vector of quantiles at which to fix the remaining predictors in Z

sel logical expression indicating samples to keep; defaults to keeping the second half of all samples

min.plot.dist specifies a minimum distance that a new grid point needs to be from an observed data point in order to compute the prediction; points further than this will not be computed

center flag for whether to scale the exposure-response function to have mean zero

z.names optional vector of names for the columns of z

... other arguments to pass on to the prediction function

Details

For guided examples, go to https://jenfb.github.io/bkmr/overview.html

Value

a long data frame with the predictor name, predictor value, posterior mean estimate, and posterior standard deviation

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
pred.resp.univar <- PredictorResponseUnivar(fit = fitkm)
```

---

### print.bkmrfit

Print basic summary of BKMR model fit

Description

print method for class "bkmrfit"
### Usage

```r
## S3 method for class 'bkmrfit'
print(x, digits = 5, ...)
```

### Arguments

- `x`: an object of class "bkmrfit"
- `digits`: the number of digits to show when printing
- `...`: further arguments passed to or from other methods.

### Value

No return value, prints basic summary of fit to console

### Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
fitkm
```

---

### SamplePred

**Obtain posterior samples of predictions at new points**

### Description

Obtains posterior samples of \( \hat{E}(Y) = h(Z_{new}) + \beta X_{new} \) or of \( g^{-1}[E(y)] \)

### Usage

```r
SamplePred(
  fit,
  Znew = NULL,
  Xnew = NULL,
  Z = NULL,
  X = NULL,
  y = NULL,
  sel = NULL,
```

```r
type = c("link", "response"),
...
)

Arguments

fit An object containing the results returned by a the kmbayes function
Znew optional matrix of new predictor values at which to predict new h, where each row represents a new observation. If not specified, defaults to using observed Z values
Xnew optional matrix of new covariate values at which to obtain predictions. If not specified, defaults to using observed X values
Z an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents an predictor.
X an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
y a vector of outcome data of length n.
 sel A vector selecting which iterations of the BKMR fit should be retained for inference. If not specified, will default to keeping every 10 iterations after dropping the first 50% of samples, or if this results in fewer than 100 iterations, than 100 iterations are kept
type whether to make predictions on the scale of the link or of the response; only relevant for the binomial outcome family
... other arguments; not currently used

Details

For guided examples, go to https://jenfb.github.io/bkmr/overview.html

Value

a matrix with the posterior samples at the new points

Examples

set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
med_vals <- apply(Z, 2, median)
```
SimData <- matrix(med_vals, nrow = 1)
h_true <- dat$HFun(Znew)
set.seed(111)
samps3 <- SamplePred(fitkm, Znew = Znew, Xnew = cbind(0))
head(samps3)

SimData

Simulate dataset

Description

Simulate predictor, covariate, and continuous outcome data

Usage

SimData(
  n = 100,
  M = 5,
  sigsq.true = 0.5,
  beta.true = 2,
  hfun = 3,
  Zgen = "norm",
  ind = 1:2,
  family = "gaussian"
)

Arguments

n Number of observations
M Number of predictor variables to generate
sigsq.true Variance of normally distributed residual error
beta.true Coefficient on the covariate
hfun An integer from 1 to 3 identifying which predictor-response function to generate
Zgen Method for generating the matrix Z of exposure variables, taking one of the values c("unif", "norm", "corr", "realistic")
ind select which predictor(s) will be included in the h function; how many predictors that can be included will depend on which h function is being used.
family a description of the error distribution and link function to be used in the model. Currently implemented for gaussian and binomial families.

Details

- hfun = 1: A nonlinear function of the first predictor
- hfun = 2: A linear function of the first two predictors and their product term
- hfun = 3: A nonlinear and nonadditive function of the first two predictor variables
SingVarIntSummaries

Value
a list containing the parameter values and generated variables of the simulated datasets

Examples

```r
c <- set.seed(5)
dat <- SimData()
```

---

SingVarIntSummaries  Single Variable Interaction Summaries

Description
Compare the single-predictor health risks when all of the other predictors in Z are fixed to their a specific quantile to when all of the other predictors in Z are fixed to their a second specific quantile.

Usage

```r
SingVarIntSummaries(
  fit,
  y = NULL,
  Z = NULL,
  X = NULL,
  which.z = 1:ncol(Z),
  qs.diff = c(0.25, 0.75),
  qs.fixed = c(0.25, 0.75),
  method = "approx",
  sel = NULL,
  z.names = colnames(Z),
  ...
)
```

Arguments

- **fit** An object containing the results returned by a the kmbayes function
- **y** a vector of outcome data of length n.
- **Z** an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents an predictor.
- **X** an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
- **which.z** vector indicating which variables (columns of Z) for which the summary should be computed
- **qs.diff** vector indicating the two quantiles at which to compute the single-predictor risk summary
SingVarIntSummaries

qs.fixed vector indicating the two quantiles at which to fix all of the remaining exposures in Z

method method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details

sel logical expression indicating samples to keep; defaults to keeping the second half of all samples

z.names optional vector of names for the columns of z

Details

- If method == "approx", the argument sel defaults to the second half of the MCMC iterations.
- If method == "exact", the argument sel defaults to keeping every 10 iterations after dropping the first 50% of samples, or if this results in fewer than 100 iterations, than 100 iterations are kept

For guided examples and additional information, go to https://jenfb.github.io/bkmr/overview.html

Value

a data frame containing the (posterior mean) estimate and posterior standard deviation of the single-predictor risk measures

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)

risks.int <- SingVarIntSummaries(fit = fitkm, method = "exact")
```
SingVarRiskSummaries  Single Variable Risk Summaries

Description

Compute summaries of the risks associated with a change in a single variable in Z from a single level (quantile) to a second level (quantile), for the other variables in Z fixed to a specific level (quantile)

Usage

SingVarRiskSummaries(
  fit,
  y = NULL,
  Z = NULL,
  X = NULL,
  which.z = 1:ncol(Z),
  qs.diff = c(0.25, 0.75),
  q.fixed = c(0.25, 0.5, 0.75),
  method = "approx",
  sel = NULL,
  z.names = colnames(Z),
  ...
)

Arguments

fit  An object containing the results returned by a the kmbayes function
y   a vector of outcome data of length n.
Z   an n-by-M matrix of predictor variables to be included in the h function. Each row represents an observation and each column represents a predictor.
X   an n-by-K matrix of covariate data where each row represents an observation and each column represents a covariate. Should not contain an intercept column.
which.z vector indicating which variables (columns of Z) for which the summary should be computed
qs.diff vector indicating the two quantiles q_1 and q_2 at which to compute h(z_{q2}) - h(z_{q1})
q.fixed vector of quantiles at which to fix the remaining predictors in Z
method method for obtaining posterior summaries at a vector of new points. Options are "approx" and "exact"; defaults to "approx", which is faster particularly for large datasets; see details
sel   logical expression indicating samples to keep; defaults to keeping the second half of all samples
z.names optional vector of names for the columns of z
...   other arguments to pass on to the prediction function
Details

- If method == "approx", the argument sel defaults to the second half of the MCMC iterations.
- If method == "exact", the argument sel defaults to keeping every 10 iterations after dropping the first 50% of samples, or if this results in fewer than 100 iterations, than 100 iterations are kept.

For guided examples and additional information, go to https://jenfb.github.io/bkmr/overview.html.

Value

a data frame containing the (posterior mean) estimate and posterior standard deviation of the single-predictor risk measures.

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
risks.singvar <- SingVarRiskSummaries(fit = fitkm, method = "exact")
```

summary.bkmrfit

Summarizing BKMR model fits

Description

summary method for class "bkmrfit"

Usage

```r
## S3 method for class 'bkmrfit'
summary(
  object,
  q = c(0.025, 0.975),
  digits = 5,
  show_ests = TRUE,
  show_MH = TRUE,
  ...
)
```
Arguments

object: an object of class "bkmrfit"
q: quantiles of posterior distribution to show
digits: the number of digits to show when printing
show_est: logical; if TRUE, prints summary statistics of posterior distribution
show_MH: logical; if TRUE, prints acceptance rates from the Metropolis-Hastings algorithm
...

Value

No return value, prints more detailed summary of fit to console

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
summary(fitkm)
```

---

**TracePlot**

Trace plot

Description

Trace plot

Usage

```r
TracePlot(
  fit,
  par,
  comp = 1,
  sel = NULL,
  main = "",
  xlab = "iteration",
  ylab = "parameter value",
  ...
)
```
Arguments

- **fit**: An object containing the results returned by a the `kmbayes` function.
- **par**: which parameter to plot
- **comp**: which component of the parameter vector to plot
- **sel**: logical expression indicating samples to keep; defaults to keeping the second half of all samples
- **main**: title
- **xlab**: x axis label
- **ylab**: y axis label
- **...**: other arguments to pass onto the plotting function

Details

For guided examples, go to https://jenfb.github.io/bkmr/overview.html

Value

No return value, generates plot

Examples

```r
## First generate dataset
set.seed(111)
dat <- SimData(n = 50, M = 4)
y <- dat$y
Z <- dat$Z
X <- dat$X

## Fit model with component-wise variable selection
## Using only 100 iterations to make example run quickly
## Typically should use a large number of iterations for inference
set.seed(111)
fitkm <- kmbayes(y = y, Z = Z, X = X, iter = 100, verbose = FALSE, varsel = TRUE)
TracePlot(fit = fitkm, par = "beta")
TracePlot(fit = fitkm, par = "sigsq.eps")
TracePlot(fit = fitkm, par = "r", comp = 1)
```
Index

ComputePostmeanHnew, 2
ExtractEsts, 3
ExtractPIPs, 4
ExtractSamps, 5
InvestigatePrior, 6, 12
kmbayes, 7
OverallRiskSummaries, 10
PlotPriorFits, 11
PredictorResponseBivar, 12, 14
PredictorResponseBivarLevels, 14
PredictorResponseBivarPair, 15
PredictorResponseUnivar, 17
print, 9
print.bkmrfit, 9, 18
SamplePred, 8, 19
SimData, 21
SingVarIntSummaries, 22
SingVarRiskSummaries, 24
summary, 9
summary.bkmrfit, 9, 25
TracePlot, 26