Package ‘multivator’

May 29, 2017

Type Package
Title A Multivariate Emulator
Version 1.1-9
Depends R(>= 3.0.1)
Imports utils, emulator (>= 1.2-15), mvtnorm, methods
Suggests abind
Date 2017-05-29
Author Robin K. S. Hankin
Maintainer Robin K. S. Hankin <hankin.robin@gmail.com>
Description A multivariate generalization of the emulator package.
License GPL-2
LazyLoad yes
LazyData yes
NeedsCompilation no
Repository CRAN
Date/Publication 2017-05-29 03:39:52 UTC

R topics documented:

  multivator-package ............................................ 2
  apart ............................................................ 3
  as.separate ..................................................... 4
  betahat .......................................................... 5
  compatible ....................................................... 7
  default_LoF ...................................................... 8
  e3mg ............................................................. 8
  experiment ....................................................... 10
  head ............................................................. 11
  ipd ............................................................... 12
  mcneall ........................................................ 12
  mdm ............................................................. 13
multivator-package

mhp .......................................................... 15
mtoys ....................................................... 16
multem ...................................................... 17
obs_maker ................................................... 18
optimal_params ............................................. 19
Print .......................................................... 21
showmap ..................................................... 21
ss .............................................................. 22
toy_mm_maker ................................................. 23

Index 25

multivator-package  A multivariate emulator

Description

A generalization of the emulator as discussed in Hankin 2005

Details

Package: multivator
Type: Package
Version: 1.0
Date: 2009-10-27
License: GPL-2
LazyLoad: yes

Author(s)

Robin K. S. Hankin
Maintainer: <hankin.robin@gmail.com>

References


See Also

multem
**apart**

*Decompose a matrix with multiple columns of dependent variables*

---

**Examples**

```r
data(mtoys)
d <- obs_maker(toy_mm, toy_mhp, toy_LoF, toy_beta)
ex <- experiment(toy_mm, d)

mulatem(toy_mm2, ex, toy_mhp, toy_LoF, give=TRUE)
```

**Description**

Decomposes a matrix with multiple columns of dependent variables into a `mdm` object.

**Usage**

```r
apart(X, dependent, use_rownames = TRUE)
```

**Arguments**

- `X`: A matrix with columns corresponding to either independent variables or dependent variables. The names of the independent variables are taken from the column names of `X`.
- `dependent`: Vector of length `ncol(X)`. If numeric, interpret as the column numbers of the dependent variable. If logical, `TRUE` elements correspond to dependent variables.
- `use_rownames`: Boolean, with default `TRUE` meaning to use the rownames of `X` to create rownames in the returned value.

**Value**

Returns an object of class `experiment`.

**Author(s)**

Robin K. S. Hankin

**See Also**

`as.list`
Examples

data(e3mg)
apart(e3mg, 6:7)

a <- round(emulator::latin.hypercube(6,5),2)
rownames(a) <- c("first","second","third","fourth","fifth","sixth")
colnames(a) <- c(letters[1:3],"length","depth")
jj_expt <- apart(a,4:5)  # use of apart()

x <- get_mdm(jj_expt[c(1,7)])
xold(x) <- 0.5
multem(x,jj_expt, hp=as.mhp(x), give=TRUE)

---

as.separate

Split an object of class experiment into a list of univariate datasets

Description

Split an experiment object into univariate designs; return a list with elements suitable for univariate analysis with the emulator package.

Usage

as.separate(expt)

Arguments

expt  Object of class experiment

Author(s)

Robin K. S. Hankin

Examples

require(emulator)

data(mtoys)
d <- obs_maker(toy_mm, toy_mhp, toy_LoF, toy_beta)

ex <- experiment(toy_mm, d)
jj <- as.separate(ex)  # list of 3: temp,rain,humidity

# now use it in a univariate emulator:
kk <- jj$temp
Various intermediate expressions needed by the multivariate emulator

**Description**

Various intermediate expressions needed by the multivariate emulator

**Usage**

```R
regressor(x, LoF)
beta_hat(expt, hp, LoF, 
betahat_mult(H, Sigmainv, d)
betahat_mult_Sigma(H, Sigma, d)
cstar(x1, x2=x1, expt, hp, LoF = NULL, Sigmainv=NULL, 
eq2.36(H, Sigmainv, d, log=TRUE)
eq2.36_Sigma(H, Sigma, d)
var.matrix(x1, x2=x1, hp, 
```

**Arguments**

- `x, x1, x2` Objects of class `mdm`: multivariate design matrix
- `H` Matrix of regressors (create this with `regressor()`)
- `d` Vector of observations, possibly not all of the same dimensions (eg some elements might be Kelvin, others millimeters of rain per year)
- `expt` Object of class `experiment`
- `Sigma` The variance matrix of `d`
- `log` Boolean, with `TRUE` meaning to return the logarithm of the answer
- `Sigmainv` The inverse of the variance matrix of `d`, with default `NULL` meaning to calculate it directly using `var.matrix()`
- `LoF` A list of functions with default `NULL` meaning to use `default_LoF()`
- `hp` Object of class `mhp`: multivariate hyperparameters
- `...` Extra arguments which are passed (via `var.matrix()`) to `corr.matrix()` of the emulator package
Details

Function \texttt{regressor()} creates a (sort of) direct sum of regressor matrices for an overall regressor matrix. It returns a matrix whose rows are the regressor functions for each row in the \texttt{df} argument. Each type of observation has its own ‘slot’ of columns, the others being filled with zeros.

The emulator package \texttt{should} have used this method (rather than messing about with \texttt{regressor.basis()} and \texttt{regressor.multi()}).

To get the regression coefficients, the user should use function \texttt{beta_hat()}, which is the user-friendly version. It is a wrapper for function \texttt{betahat_mult_sigma()}. The equation for \texttt{var.matrix()} is

\[
c^*(x, x') = c(x, x') - t(x)'A^{-1}t(x') + \{h(x)' - t(x)'A^{-1}H\}(H'^TA^{-1}H)^{-1}\{h(x')' - t(x')'A^{-1}H\}'^T
\]

Author(s)

Robin K. S. Hankin

See Also

\texttt{multem}

Examples

data(mtoys)

\texttt{H <- regressor(toy_mm, toy_LoF)}
\texttt{Sigma <- var.matrix(toy_mm, hp=toy_mhp)}
\texttt{Sigmainv <- solve(Sigma)}

\texttt{jj <- toy_mm_maker(34,35,36)}
\texttt{expt <- experiment(jj,obs_maker(jj,toy_mhp,toy_LoF,toy_beta))}

\texttt{x1 <- jj[c(20,40,100),]}
\texttt{xold(x1) <- 0.2}

\texttt{x2 <- jj[c(11,21:24,40:42),]}
\texttt{xold(x2) <- xold(x2)+0.1}

# primary function of package:
\texttt{multem(x=x1, expt, hp=toy_mhp, LoF=toy_LoF)}

# conditional covariance matrix:
\texttt{cstar(x1,x2, expt, hp=toy_mhp, LoF=toy_LoF)}
compatible

Are two objects compatible?

Description

Function to detect whether two objects are compatible

Usage

compatible(x1, x2)

Arguments

x1, x2 Two objects with names and levels. Typically either objects of class mhp or mdm.

Details

Here, “compatible” means have the same names and levels. If an mdm object and mhp object are compatible, then they may be supplied to (eg) var.matrix().

The function uses identical() to compare the names and levels.

Value

Returns a Boolean.

Note

Cannot yet compare LoF objects.

Author(s)

Robin K. S. Hankin

Examples

data(mtoys)
stopifnot(compatible(toy_mhp, toy_mm))
**default_loF**  
*Default List of functions*

**Description**

Creates a default List of Functions for use with `regressor()`.

**Usage**

```r
default_loF(x)
```

**Arguments**

- `x` Object with names and levels methods; typically of class `mdm` or `mhp`.

**Value**

Returns a named list with each element giving the regressor functions for that level.

**Author(s)**

Robin K. S. Hankin

**See Also**

`regressor`

**Examples**

```r
data(mtoys)
default_loF(toy_mm)  # note list names == levels(toy_mm)
regressor(toy_mm)    # use default
regressor(toy_mm, toy_loF)  # use a bespoke set
```

---

**e3mg**  
*Output from computer model e3mg*

**Description**

Output from computer model e3mg detailing the depth of the recession and its length as a function of four exogenous parameters

**Usage**

```r
data(e3mg)
```
**Format**

- e3mg is a matrix with 843 rows and 6 columns. Four of the columns are exogenous variables (oil.price, direct.tax, interest.rate, and saving.ratio) and two are model outputs: rec_len, the length (in years) of the recession, and dep_rec, the depth of the recession.
- e3mg_LoF is a list of functions suitable for use with the e3mg dataset

**Details**

The data comprises 843 runs of the e3mg econometric model, used to predict the recession precipitated by the banking crisis.

The depth of the recession is defined as the maximum difference between predicted post-crash GDP and GDP immediately pre-crash.

The length of the recession is defined as the time in years required for GDP to return to pre-crash levels.

**Source**

Data kindly provided by Cambridge Econometrics

**See Also**

apart

**Examples**

```r
data(e3mg)
a <- lm(rec_len~oil.price*direct.tax + direct.tax*saving.ratio + investment, data=data.frame(e3mg))
b <- lm(rec_dep~oil.price*direct.tax + direct.tax*saving.ratio + investment, data=data.frame(e3mg))
plot(residuals(a), residuals(b)) # correlated!

# define an experiment object and find optimal prams
e3mg_expt <- apart(e3mg[1:20,],6:7)
opt <- optimal_params(e3mg_expt, e3mg_LoF, option='c')

# now a point in parameter space:
center <- get_mdm(e3mg_expt)[c(1,40),]
rownames(center) <- c('center_dep','center_len')
xold(center) <- 0

#now predict the behaviour at the center:
multem(center, e3mg_expt, hp=opt, e3mg_LoF, give = TRUE)
```
experiment

Multivariate hyperparameter (mhp) objects

Description

Create and manipulate multivariate hyperparameter (mhp) objects

Usage

experiment(mm, obs)

Arguments

mm Object of class mdm
obs Vector of observations, with elements corresponding to the rows of mm

Details

An “experiment” is an ordered pair of a multivariate design matrix and a vector of observations with entries corresponding to the rows of the design matrix. It functions as a container for the design matrix and observations. It is intended to simplify the calls to many functions in the package which require a design matrix and vector of observations. There are two get methods, get_mdm() and get_obs(), for the design matrix and observations respectively. Note the deliberate absence of set methods.

Value

Returns an object of class experiment, which is used as input to many of the functions in the package.

Author(s)

Robin K. S. Hankin

Examples

data(mtoys)
jj.expt <- experiment(toy_mm, toy_d)

# accessor methods:
get_obs(jj.expt)
get_mdm(jj.expt)

# estimation of coefficients:
beta_hat(jj.expt, toy_mhp, toy_LoF)
head

# use multem():
multem(toy_mm3, jj_expt, toy_mhp, toy_LoF,give=TRUE)

---

**Description**

Print the first few, or last few, lines of a mdm object

**Usage**

```r
## S4 method for signature 'mdm'
head(x, n = 6, ...)
## S4 method for signature 'mdm'
tail(x, n = 6, ...)
```

**Arguments**

- `x` object of class `mdm`
- `n` number of lines to print as per same argument in `head()` and `tail()`
- `...` Further arguments passed to `head()` or `tail()`

**Value**

Returns a truncated `mdm` object. The levels of the types are unchanged.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
data("mtoys")
head(toy_mm)
tail(toy_mm, 3)
```
### Positive definite matrices

**Description**

Is a matrix symmetric positive-definite?

**Usage**

`ipd(mat)`

**Arguments**

- `mat` A matrix

**Value**

Returns either `TRUE` if symmetric positive-definite; or `FALSE`, printing a diagnostic message.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
data(mtoys)
stopifnot(ipd(crossprod(matrix(rnorm(30),10))))
stopifnot(ipd(M(toy_mhp)))
```

### Dataset due to McNeall

**Description**

Data, due to McNeall, from 92 runs of a climate model

**Usage**

`data(mcneall)`
Details

McNeall used a numerical climate model and ran it 92 times, on a design matrix specified on 16 independent variables as detailed in McNeall 2008.

The model output is a temperature distribution over the surface of the Earth. The model gives 2048 temperatures, corresponding to 2048 grid squares distributed over the Earth. A vector of 2048 temperatures may be displayed on a global map using the `showmap()` function.

The 92 model runs are presented in the form of a 2048 by 92 matrix `mcneall_temps`, each column of which corresponds to a run. A row of 92 temperatures corresponds to the temperature at a particular place on the earth as predicted by each of the 92 model runs.

Following McNeall, a principal component analysis on the maps was performed. The first four were used. Matrix `eigenmaps` is a 2048 by 4 matrix, with columns corresponding to the four principal components.

Matrix `mcneall_pc` is a 92-by-20 matrix. The first 16 columns correspond to the independent variables (i.e., the design matrix); columns 17-20 correspond to the first four principal components of the model output. The 92 rows correspond to the 92 model runs.

The package can be used on the `mcneall_temps` matrix; use `apart()` to generate a `mdm` object. A reasonably optimized hyperparameters object of class `mhp` is given as `opt_mcneall`.

References


See Also

`showmap`

Examples

```r
data(mcneall)

showmap(mcneall_temps[,1], pc=FALSE, landmask=landmask)
```

---

**mdm**

*Multivariate design matrices*

Description

Multivariate design matrices are represented using objects of class `mdm`.
Usage

```r
mdm(xold, types)
as.mdm(x, ...)
is.mdm(x)
as.list(x, ...)
as.matrix(x, ...)
```  
```r
## S4 method for signature 'mdm,missing,missing'
as.data.frame(x, row.names=NULL,optional=TRUE, ...)
## S4 method for signature 'mdm'
rbind(x, ..., deparse.level=1)
types(x)
xold(x)
```

Arguments

- `xold`: Matrix of design points, each row being a point in parameter space
- `types`: A factor holding the types of each observation
- `x`: An object of class `mdm`
- `row.names,optional`: Currently ignored
- `...`: Further arguments passed to `NextMethod()`
- `deparse.level`: As for `rbind()`

Details

Various functionality for creating and manipulating objects of class `mdm` (Multivariate Design Matrix).

Note

The internal representation has two slots, one for the design matrix proper (a matrix), and one for the types of observation (a factor).

Author(s)

Robin K. S. Hankin

See Also

`mhp, apart`

Examples

```r
mm <- toy_mm_maker(7,8,9)
is.mdm(mm)
xold(mm) <- matrix(rnorm(108),27,4)
mm[1,1] <- 0.3
```
**mhp**

Multivariate hyperparameter (mhp) objects

**Description**

Create and manipulate multivariate hyperparameter (mhp) objects

**Usage**

```r
mhp(M, B, levels = NULL, names = NULL)
is.mhp(x)
M(x)
M(x) <- value
B(x)
B(x) <- value
levels(x)
summary(object,...)
```

**Arguments**

- `M` : Variance matrix (must be positive definite)
- `B` : Array of roughness parameters. Each slice (ie `B[,]`, i) must be positive-definite
- `levels` : Character vector holding the levels. Default `NULL` means to use `rownames(M)` or `dimnames(B[[3]])`
- `names` : Character vector holding the names of the dimensions. Default of `NULL` means to use `dimnames(B[[1]])`
- `x,object` : Object of class `mhp`
- `value` : Replacement object
- `...` : Further arguments passed to the `summary` method

**Details**

An mhp object must have names and levels, so either provide them explicitly with the eponymous arguments, or give named arrays to `M` and `B`.

**Value**

Returns an object of class `mhp`

**Author(s)**

Robin K. S. Hankin
mtoys

Toy datasets

Description

Toy datasets that illustrate the package

Usage

toy_lof
toy_mm
toy_mm2
toy_mm3
toy_mhp

Format

- toy_lof is a list of three functions that work with regressor() and toy_df
- toy_M is an example M matrix for use with mhp()
- toy_B is an example of a B array of roughness coefficients for use with mhp()
- toy_mm and toy_mm2 are examples of a mdm object, generated with function toy_mm_maker(). These objects are marginals from the same multivariate observation.
- toy_mm3 and toy_mm4 are small examples of mdm objects
- toy_mhp is an example of a mhp object
- toy_beta is a numeric vector that works with the above objects

Details

These objects are intended as simple working ‘toy’ examples of the various things needed to use the emulator.

Note that toy_d and toy_d2 are the marginals of the same observation (see the vignette).
**multem**

**Author(s)**
Robin K. S. Hankin

**References**


**See Also**

- `toy_mm_maker`

**Examples**

```r
data(mtoys)
obs-maker(toy_mm, toy_mhp, toy_lof, toy_beta)
multem(toy_mm2, toy_expt, toy_mhp, toy_lof, give=TRUE)
```

---

**multem**  
*The multivariate emulator*

**Description**

A multivariate generalization of the `interpolant()` function of the `emulator` package

**Usage**

```r
multem(x, expt, hp, LoF = NULL, give=FALSE, Sigmainv=NULL, ...)
```

**Arguments**

- **x**: Points at which the function is to be estimated in the form of an object of class `mdm`
- **expt**: Points at which the code has been evaluated (`x_known`), in the form of an object of class `experiment`
- **hp**: Hyperparameter object, of class `mhp`
- **give**: Boolean, with `TRUE` meaning to return extra information and default `FALSE` meaning to return just the mean
- **Sigmainv**: The inverse of the variance matrix of the observations with default `NULL` meaning to calculate using `var.matrix()`
- **LoF**: List of regressor functions
- **...**: Further arguments passed to `var.matrix()`
Details

This is the central function of the package. It is the analogue of `interpolant()` of the emulator package.

Author(s)

Robin K. S. Hankin

See Also

`betahat_mult`

Examples

data(mtoys)
d <- obs_maker(toy_mm, toy_mhp, toy_LoF, toy_beta)
ex <- experiment(toy_mm, d)

Sigmainv <- solve(var.matrix(toy_mm, hp=toy_mhp))
multem(x=toy_mm2, expt=ex, hp=toy_mhp, LoF=toy_LoF, give=TRUE)

obs_maker

Create observations

Description

A function to create observations using known parameters and hyperparameters

Usage

`obs_maker(x, hp, LoF, beta, Sigma=NULL, ...)`

Arguments

- `x` Object of class `mdm`: each row is a point in parameter space
- `hp` Object of class `mhp`
- `LoF` List of functions
- `beta` Vector of regression coefficients
- `Sigma` Variance matrix, with default `NULL` meaning to use `var.matrix(x,hp)`
- `...` Further arguments passed to `var.matrix()`

Details

Uses the `mvtnorm` package to generate observations directly from the parameters and hyperparameters as a Gaussian process.
optimal_params

Value

Returns a (named) vector of observations. Note that the observations may have different units (e.g., temperature in Kelvin, rainfall in millimeters per year).

Author(s)

Robin K. S. Hankin

See Also

toy_mm_maker

Examples

data(mtoys)
d <- obs_maker(toy_mm, toy_mhp, toy_LoF, toy_beta)d <- obs_maker(toy_mm_maker(6, 7, 8), toy_mhp, toy_LoF, toy_beta)

optimal_params

Optimization of the hyperparameters

Description

Optimization of the hyperparameters using a sequence of subfunctions.

Usage

optimal_params (expt, LoF, start_hp, option = "a", ...)
optimal_B (expt, LoF, start_hp, option = "a", verbose=FALSE, ...)
optimal_identical_B(expt, LoF, start_hp, verbose=FALSE, ...)
optimal_diag_M (expt, LoF, start_hp)
optimal_M (expt, LoF, start_hp, ...)

Arguments

expt Object of class experiment
LoF List of functions
start_hp Start value for the hyperparameters, an object of class mhp. The various optimization routines use the different parts of start_hp as start points, and incrementally update it
option In function optimal_B() and consequently optimal_params(), a character indicating whether to allow the scales to differ or not.

• Default option “a” is the simplest: each univariate B matrix is a multiple of the identity matrix.
• Option “b” allows the B matrices to be any (positive definite) diagonal matrix.
• Option “c” specifies that $B[i,j]$ is diagonal for each $j$ and furthermore that $B[i,i,1] = B[i,i,2] = \ldots = B[i,i,r]$. This option calls `optimal_identical_B()`.

**verbose**

In function `optimal_B()`, Boolean with TRUE meaning to print debugging information and default FALSE meaning not to print anything.

... Further arguments passed to the optimization routine

**Details**

The user-friendly wrapper function is `optimal_params()`. This calls function `optimal_B()` first, as most of the analysis is conditional on $B$. Then `optimal_diag_M()` is called; this places the maximum likelihood estimate for $\sigma^2$ on the diagonal of $M$. Finally, `optimal_M()` is called, which assigns the off-diagonal elements of $M$.

Each of the subfunctions returns an object appropriate for insertion into a `mhp` object.

The “meat” of `optimal_params()` is

```r
B(out) <- optimal_B (mm, d, LoF, start_hp=out, option=option, ...)
diag(M(out))) <- optimal_diag_M(mm, d, LoF, start_hp=out, ...)
M(out) <- optimal_M (mm, d, LoF, start_hp=out, ...)
return(out)
```

See how object `out` is modified sequentially, it being used as a start point for the next function.

**Value**

Returns a `mhp` object.

**Note**

Function `optimal_diag_M()` uses MLEs for the diagonals, but using each type of observation separately. It is conceivable that there is information that is not being used here.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
data(mtoys)

optimal_params(toy_expt, toy_LoF, toy_mhp, option='c', control=list(maxit=1))
```
Methods for printing mhp and mdm objects

Description

Methods for printing nicely

Usage

```r
## S3 method for class 'mdm'
print(x, ...)
## S3 method for class 'mhp'
print(x, ...)
```

Arguments

- `x` An object of class `mdm` or `mhp`
- `...` Further arguments (currently ignored)

Author(s)

Robin K. S. Hankin

Examples

```r
data(n.toys)
a <- as.mhp(toy_mm)
a
```

Function to plot the McNeall dataset

Description

A small wrapper function to plot a global map of temperature, which is useful when analyzing the McNeall dataset

Usage

```r
showmap(a, p.c, landmask, ...)
```
Arguments

- **z**: A vector of length 2048 corresponding to temperatures on the Earth’s surface.
- **pc**: Boolean, with TRUE meaning to interpret z as a principal component and FALSE meaning to interpret z as a temperature map.
- **landmask**: A matrix of zeros and ones corresponding to the Earth’s surface with zero indicating sea and one indicating land; use `data(mcneall)`.
- **...**: Further arguments passed to `filled.contour()`.

Author(s)

Robin K. S. Hankin

See Also

- `mcneall`

Examples

```r
data(mcneall)
showmap(mcneall_temps[,1],pc=FALSE,landmask=landmask)
```

---

**ss**

*Overall variance matrix*

Description

Calculates the maximum correlations possible consistent with the roughness parameters.

Usage

- `ss(A, B, Ainv, Binv)`
- `ss_matrix(hp,useM=TRUE)`
- `ss_matrix_simple(hp,useM=TRUE)`

Arguments

- **A,B**: Positive-definite matrices (roughness parameters).
- **Ainv,Binv**: The inverses of A and B; if missing, compute explicitly.
- **hp**: An object of class mhp.
- **useM**: Boolean, with default TRUE meaning to multiply (pointwise) by M and FALSE meaning not to (so giving the maximum correlation consistent with the roughness matrices B).
Details

Function \( \text{ss}() \) calculates the maximum possible correlation between observations of two Gaussian processes at the same point (equation 24 of the vignette):

\[
\left| \left( \frac{1}{2} B_r + \frac{1}{2} B_s \right) \left( \frac{1}{2} B_r^{-1} + \frac{1}{2} B_s^{-1} \right) \right|^{-1/4}
\]

Functions \( \text{ss\_matrix}() \) and \( \text{ss\_matrix\_simple}() \) calculate the maximum covariances among the types of object specified in the \( \text{hp} \) argument, an object of class \( \text{mhp} \). Function \( \text{ss\_matrix}() \) is the preferred form; function \( \text{ss\_matrix\_simple}() \) is a less efficient, but more transparent, version. The two functions should return identical output.

Value

Function \( \text{ss}() \) returns a scalar, \( \text{ss\_matrix}() \) a matrix of covariances.

Note

Thanks to Stephen Stretton for a crucial insight here

Author(s)

Robin K. S. Hankin

Examples

```r
data(ntoys)
ss_matrix(toy_mhp)
```

---

**toy\_mm\_maker**  
*Make a toy mm object*

Description

Create a toy mhp object with three levels: temperature, rainfall, and humidity.

Usage

```r
toy\_mm\_maker(na, nb, nc, include\_first = TRUE)
```

Arguments

- `na`, `nb`, `nc`  
  Numbers of observations for each level
- `include\_first`  
  Boolean, with default \( \text{TRUE} \) meaning to include an extra observation of each level at the midpoint of the domain
Value

Returns an object of class mhp.

Author(s)

Robin K. S. Hankin

Examples

toy_mm_maker(4, 5, 6, FALSE)
toy_mm_maker(1, 1, 2, TRUE)
# Index

## *Topic* datasets
- e3mg, 8
- mcneall, 12
- mtoys, 16

## *Topic* package
- multivariate-package, 2
- [,experiment-method (experiment), 10
- [,mdm-method (mdm), 13
- [<-,mdm-method (mdm), 13

- apart, 3, 9, 14
- as.data.frame (mdm), 13
- as.data.frame,experiment,ANY,ANY-method (experiment), 10
- as.data.frame,experiment-method (experiment), 10
- as.data.frame,mdm,missing,missing-method (mdm), 13
- as.data.frame,mhp-method (mdm), 13
- as.list, 3
- as.list (mdm), 13
- as.list,mdm-method (mdm), 13
- as.matrix (mdm), 13
- as.matrix,mdm-method (mdm), 13
- as.mdm (mdm), 13
- as.mdm,mdm-method (mdm), 13
- as.mhp (mhp), 15
- as.mhp,experiment-method (mhp), 15
- as.mhp,mdm-method (mhp), 15
- as.separate, 4

- B (mhp), 15
- B<- (mhp), 15
- beta_hat (betahat), 5
- betahat, 5
- betahat_mult, 18
- betahat_mult (betahat), 5
- betahat_mult_Sigma (betahat), 5
- compatible, 7
- cstar (betahat), 5
- default_LoF, 8
- dim,mdm-method (mdm), 13
- e3mg, 8
- e3mg_LoF (e3mg), 8
- eigenmaps (mcneall), 12
- eq2.36 (betahat), 5
- eq2.36_Sigma (betahat), 5
- experiment, 10
- experiment-class (experiment), 10
- get_mdm (experiment), 10
- get_obs (experiment), 10
- head, 11
- head,experiment-method (experiment), 10
- head,mdm-method (head), 11
- ipd, 12
- is.mdm (mdm), 13
- is.mhp (mhp), 15
- landmask (mcneall), 12
- levels (mhp), 15
- levels,experiment-method (mdm), 13
- levels,mdm-method (mdm), 13
- levels,mhp-method (mhp), 15
- levels<-,mdm-method (mdm), 13
- levels<-,mhp-method (mhp), 15
- M (mhp), 15
- M<- (mhp), 15
- mcneall, 12, 22
- mcneall_pc (mcneall), 12
- mcneall_temps (mcneall), 12
- mdm, 13, 16
- mdm-class (mdm), 13
- mean_temp (mcneall), 12
- mhp, 14, 15
mhp-class (mhp), 15
toys, 16
multem, 2, 6, 17
multivator (multivator-package), 2
multivator-package, 2

names (mhp), 15
names, mdm-method (mdm), 13
names, mhp-method (mhp), 15
names<- (mdm), 13
names<-., mdm-method (mdm), 13
names<-, mhp-method (mhp), 15
ncol, mdm-method (mdm), 13
nrow, mdm-method (mdm), 13

obs_maker, 18
opt_mcneall (mcneall), 12
optimal_B (optimal_params), 19
optimal_diag_M (optimal_params), 19
optimal_identical_B (optimal_params), 19
optimal_M (optimal_params), 19
optimal_params, 19

Print, 21
print.experiment (experiment), 10
print.mdm (Print), 21
print.mhp (Print), 21
print.mhpSummary (mhp), 15

rbind (mdm), 13
rbind, mdm-method (mdm), 13
regressor, 8
regressor (betahat), 5
rownames, mdm-method (mdm), 13
rownames<-, mdm-method (mdm), 13

show, mdm-method (Print), 21
show, mhp-method (Print), 21
showmap, 13, 21
ss, 22
ss_matrix (ss), 22
ss_matrix_simple (ss), 22
summary (mhp), 15
summary, mhp-method (mhp), 15

tail (head), 11
tail, experiment-method (experiment), 10
tail, mdm-method (head), 11
toy_B (mtoys), 16
toy_beta (mtoys), 16
toy_d (mtoys), 16
toy_d2 (mtoys), 16
toy_expt (mtoys), 16
toy_LoF (mtoys), 16
toy_M (mtoys), 16
toy_mhp (mtoys), 16
toy_mm (mtoys), 16
toy_mm2 (mtoys), 16
toy_mm3 (mtoys), 16
toy_mm4 (mtoys), 16
toy_mm_maker, 17, 19, 23
toy_point (mtoys), 16
types (mdm), 13
types, mdm-method (mdm), 13
types, mhp-method (mhp), 15
types<- (mdm), 13
types<-, mdm-method (mdm), 13
types<-, mhp-method (mhp), 15

var_matrix (betahat), 5
xold (mdm), 13
xold, mdm-method (mdm), 13
xold<- (mdm), 13
xold<-., mdm-method (mdm), 13

mcneall, 3
mcneall-package, 3
mdm, 13
mdm-package, 13
mhp, 15
mhp-package, 15
mtoys, 16
multivator-package, 2
experiment, 10
experiment-package, 10
h2o, 24
h2o-package, 24
showmap, 13, 21
showmap-package, 13, 21
betahat, 5
var_matrix, 5