

Package ‘hypr’

July 19, 2021

Type Package

Title Hypothesis Matrix Translation

URL <https://maxrabe.com/hypr>

BugReports <https://github.com/mmrabe/hypr/issues>

Version 0.2.2

Description Translation between experimental null hypotheses, hypothesis matrices, and contrast matrices as used in linear regression models. The package is based on the method described in Schad, Vasishth, Hohenstein, and Kliegl (2019) <[doi:10.1016/j.jml.2019.104038](https://doi.org/10.1016/j.jml.2019.104038)> and Rabe, Vasishth, Hohenstein, Kliegl, and Schad (2020) <[doi:10.21105/joss.02134](https://doi.org/10.21105/joss.02134)>.

License GPL-3

Depends R (>= 3.5.0)

Imports MASS, pracma, methods, cli, magrittr,

Suggests knitr, rmarkdown,

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.1.1

Collate 'equations.R' 'hypr.R'

NeedsCompilation no

Author Maximilian M. Rabe [aut, cre] (<<https://orcid.org/0000-0002-2556-5644>>),
Shravan Vasishth [aut] (<<https://orcid.org/0000-0003-2027-1994>>),
Sven Hohenstein [aut] (<<https://orcid.org/0000-0002-9708-1593>>),
Reinhold Kliegl [aut] (<<https://orcid.org/0000-0002-0180-8488>>),
Daniel J. Schad [aut] (<<https://orcid.org/0000-0003-2586-6823>>)

Maintainer Maximilian M. Rabe <maximilian.rabe@uni-potsdam.de>

Repository CRAN

Date/Publication 2021-07-19 10:30:05 UTC

R topics documented:

| | |
|------------------------------|-----------|
| +,hypr,hypr-method | 2 |
| add_intercept | 3 |
| centered_contrasts | 4 |
| cmat | 5 |
| conversions | 7 |
| filler_contrasts | 9 |
| formula<- | 10 |
| ginv2 | 10 |
| hmat | 11 |
| hypr | 12 |
| hypr-class | 14 |
| Index | 18 |

| | |
|--------------------|--|
| +,hypr,hypr-method | <i>Combining hypr objects by addition or interaction</i> |
|--------------------|--|

Description

You can combine one or more hypr objects, i.e. combine their hypothesis to a single hypr object, by adding them with the + or * operators.

Usage

```
## S4 method for signature 'hypr,hypr'
e1 + e2

## S4 method for signature 'hypr,hypr'
e1 * e2

## S4 method for signature 'hypr,hypr'
e1 & e2

## S4 method for signature 'hypr,hypr'
e1 / e2
```

Arguments

e1, e2 hypr objects to concatenate

Value

The combined hypr object

Functions

- *,hypr,hypr-method: Interaction of e1 and e2
- &,hypr,hypr-method: Interaction and main contrasts of e1 and e2
- /,hypr,hypr-method: Nesting levels of e2 within e1

Examples

```
(h1 <- hypr(a~i, b~i)) # a hypr object of two treatments
(h2 <- hypr(i~0)) # an intercept-only hypr object
hc <- h1 + h2
hc
interaction <- h1 & h2
interaction_and_main <- h1 * h2
```

| | |
|---------------|-------------------------|
| add_intercept | <i>Intercept checks</i> |
|---------------|-------------------------|

Description

Non-centered contrasts require an intercept for correct specification of experimental hypotheses. These functions enable the user to check for existence of intercepts and to add or remove intercept columns as needed.

Usage

```
add_intercept(x)
remove_intercept(x)
is_intercept(x)
which_intercept(x)
has_intercept(x)
```

Arguments

x A hypr object

Details

There are functions available to check whether a `hypr` object contains an intercept (`has_intercept`) or which contrast is the intercept (`is_intercept`, `which_intercept`). Moreover, if needed, the user can add (`add_intercept`) or remove (`remove_intercept`) an intercept column to/from a `hypr` object. `add_intercept` and `remove_intercept` do not throw an error if the user attempts to remove a non-existing intercept or add an intercept if there already is one.

Value

A single logical value (`has_intercept`), a logical vector (`is_intercept`), an integer index vector (`which_intercept`), or a modified `hypr` object (`add_intercept`, `remove_intercept`)

Functions

- `add_intercept`: Add an intercept column if there is none
- `remove_intercept`: Remove the intercept column if there is one
- `which_intercept`: Return indices, not a logical vector of intercept columns
- `has_intercept`: Check whether any of the contrasts is an intercept

Examples

```
h1 <- hypr(mu1~0, mu2~mu1)
h2 <- hypr(mu2~mu1, mu3~mu1)

stopifnot(has_intercept(h1))
stopifnot(!has_intercept(h2))
stopifnot(which_intercept(h1) == 1)
stopifnot(is_intercept(h1) == c(TRUE,FALSE))
```

centered_contrasts *Contrast centering*

Description

Centeredness of contrasts is critical for the interpretation of interactions and intercepts. There are functions available to check for centered contrasts and to realign contrasts so that they are centered.

Usage

```
centered_contrasts(x)

is_centered(x)

all_centered(x, ignore_intercept = TRUE)

which_centered(x)
```

Arguments

`x` A hypr object
`ignore_intercept`
 If TRUE, the intercept is ignored

Details

The function `centered_contrasts(x)` will return a copy of `x` where all contrasts were centered to a zero mean.

The functions `is_centered(x)` and `which_centered()` indicate which contrasts of `x`, are centered. `all_centered(x)` will return TRUE if all contrasts in `x` are centered or FALSE if at least one contrast is not.

Value

A centered set of hypr contrasts (`centered_contrasts`), a single logical value (`all_centered`), a logical vector (`is_centered`), or an integer index vector (`which_centered`)

Functions

- `is_centered`: Check which contrasts of `x` are centered
- `all_centered`: Check whether all contrasts of `x` are centered
- `which_centered`: Check which contrasts of `x` are centered

| | |
|-------------------|--|
| <code>cmat</code> | <i>Retrieve or set contrast matrix</i> |
|-------------------|--|

Description

Use these functions to retrieve or set a hypr object's contrast matrix. If used for updating, the hypothesis matrix and equations are derived automatically.

Usage

```
cmat(x, add_intercept = FALSE, remove_intercept = FALSE, as_fractions = TRUE)
```

```
cmat(x, add_intercept = FALSE, remove_intercept = FALSE) <- value
```

```
contr.hypothesis(  
  ...,  
  add_intercept = FALSE,  
  remove_intercept = NULL,  
  as_fractions = FALSE  
)
```

```
## S4 replacement method for signature 'ANY,ANY,hypr'
```

```

contrasts(x, how.many) <- value

## S4 replacement method for signature 'ANY,ANY,hypr_cmat'
contrasts(x, how.many) <- value

contr.hypothesis(
  x,
  add_intercept = NULL,
  remove_intercept = FALSE,
  as_fractions = FALSE
) <- value

```

Arguments

| | |
|-------------------------------|---|
| <code>x</code> | A <code>hypr</code> object |
| <code>add_intercept</code> | Add additional intercept column to contrast matrix |
| <code>remove_intercept</code> | If TRUE, tries to find an intercept column (all codes equal) and removes it from the matrix. If NULL, does the same but does not throw an exception if no intercept is found. FALSE explicitly disables this functionality. A numeric argument explicitly identifies the index of the column to be removed. |
| <code>as_fractions</code> | Should the returned matrix be formatted as fractions (using <code>MASS::as.fractions()</code>)? |
| <code>value</code> | contrast matrix |
| <code>...</code> | A list of hypothesis equations for which to retrieve a contrast matrix |
| <code>how.many</code> | see <code>stats::contrasts()</code> |

Details

Basic specification of contrasts in R is accomplished with basic R functions `stats::contrasts()` and `stats::C()` (Chambers & Hastie, 1992). Other relevant packages for this topic are `multcomp` (Bretz et al., 2010), `contrast` (Kuhn et al., 2016), and, including also various vignettes, `emmeans` (Lenth, 2019).

Value

A matrix of contrast codes with contrasts as columns and levels as rows.

Functions

- `cmat<-`: Set contrast matrix
- `contr.hypothesis`: Retrieve contrast matrix with sensible intercept default to override factor contrasts
- `contrasts<- ,ANY,ANY,hypr-method`: Update factor contrasts
- `contrasts<- ,ANY,ANY,hypr_cmat-method`: Update factor contrasts
- `contr.hypothesis<-`: Update contrast matrix with sensible intercept default

References

Chambers, J. M. and Hastie, T. J. (1992) *Statistical models*. Chapter 2 of *Statistical Models* in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Frank Bretz, Torsten Hothorn and Peter Westfall (2010), *Multiple Comparisons Using R*, CRC Press, Boca Raton.

Max Kuhn, contributions from Steve Weston, Jed Wing, James Forester and Thorn Thaler (2016). *contrast: A Collection of Contrast Methods*. R package version 0.21. <https://CRAN.R-project.org/package=contrast>

Lenth, R. (2019). *emmeans: Estimated Marginal Means, aka Least-Squares Means*. R package version 1.4.1. <https://CRAN.R-project.org/package=emmeans>

See Also

[hypr](#)

Examples

```
h <- hypr(mu1~0, mu2~mu1)
cmat(h) # retrieve the contrast matrix

contr.hypothesis(h) # by default without intercept (removes first column)
contr.hypothesis(mu1~0, mu2~mu1)
```

conversions

Shorthand versions for simple hypothesis translation

Description

These functions can be used to translate between null hypothesis equations, hypothesis matrices, and contrast matrices without defining a `hypr` object. Note that some of these functions do generate a `hypr` object internally but they never return one.

Usage

```
eqs2hmat(
  eqs,
  levels = NULL,
  order_levels = missing(levels),
  as_fractions = TRUE
)

eqs2cmat(eqs, as_fractions = TRUE)

hmat2cmat(hmat, as_fractions = TRUE)
```

```
cmat2hmat(cmat, as_fractions = TRUE)
```

```
hmat2eqs(hmat, as_fractions = TRUE)
```

```
cmat2eqs(cmat, as_fractions = TRUE)
```

Arguments

| | |
|--------------|--|
| eqs | A list of equations |
| levels | (optional) A character vector of variables to be expected (if not provided, automatically generated from all terms occurring in the equations list) |
| order_levels | (optional) Whether to alphabetically order appearance of levels (rows in transposed hypothesis matrix or contrast matrix). Default is TRUE if levels were not explicitly provided. |
| as_fractions | (optional) Whether to output matrix using fractions formatting (via <code>MASS::as.fractions</code>). Defaults to TRUE. |
| hmat | Hypothesis matrix |
| cmat | Contrast matrix |

Value

A list of equations (`hmat2eqs` and `cmat2eqs`), a contrast matrix (`hmat2cmat`, `eqs2cmat`), or a hypothesis matrix (`cmat2hmat`, `eqs2hmat`).

Functions

- `eqs2hmat`: Convert null hypothesis equations to hypothesis matrix
- `eqs2cmat`: Convert null hypothesis equations to contrast matrix
- `hmat2cmat`: Convert hypothesis matrix to contrast matrix
- `cmat2hmat`: Convert contrast matrix to hypothesis matrix
- `hmat2eqs`: Convert hypothesis matrix to null hypothesis equations
- `cmat2eqs`: Convert contrast matrix to null hypothesis equations

Examples

```
# The following examples are based on a 2-level treatment contrast (i.e., baseline and treatment).
hypotheses <- list(baseline = mu1~0, treatment = mu2~mu1)
hypothesis_matrix <- matrix(
  c(c(1, -1), c(0, 1)), ncol = 2, dimnames = list(c("baseline", "treatment"), c("mu1", "mu2")))
contrast_matrix <- matrix(
  c(c(1, 1), c(0, 1)), ncol = 2, dimnames = list(c("mu1", "mu2"), c("baseline", "treatment")))

# Convert a list of null hypothesis equations to ...
# ... a hypothesis matrix:
eqs2hmat(hypotheses)
```



```

# ... a contrast matrix:
eqs2cmat(hypotheses)

# Convert a hypothesis matrix to...
# ... a list of null hypothesis equations:
hmat2eqs(hypothesis_matrix)
# ... a contrast matrix:
hmat2cmat(hypothesis_matrix)

# Convert a contrast matrix to...
# ... a list of null hypothesis equations:
cmat2eqs(contrast_matrix)
# ... a hypothesis matrix:
cmat2hmat(contrast_matrix)

# Are all functions returning the expected results?
stopifnot(all.equal(eqs2hmat(hypotheses, as_fractions = FALSE), hypothesis_matrix))
stopifnot(all.equal(eqs2cmat(hypotheses, as_fractions = FALSE), contrast_matrix))
stopifnot(all.equal(hmat2cmat(hypothesis_matrix, as_fractions = FALSE), contrast_matrix))
stopifnot(all.equal(cmat2hmat(contrast_matrix, as_fractions = FALSE), hypothesis_matrix))

```

| | |
|------------------|----------------------------------|
| filler_contrasts | <i>Generate filler contrasts</i> |
|------------------|----------------------------------|

Description

Fill free degrees of freedom with orthogonal filler contrasts.

Usage

```
filler_contrasts(x, how.many = nlevels(x), rescale = TRUE)
```

Arguments

| | |
|----------|---|
| x | A hypr object |
| how.many | The total number of contrasts for the new hypr object |
| rescale | If TRUE, the contrast weights will be rescaled |

Examples

```

# A complete Helmert contrast matrix for 4 levels:
h1 <- hypr(~ (mu2-mu1)/2,
           ~ (mu3-(mu1+mu2)/2)/3,
           ~ (mu4-(mu1+mu2+mu3)/3)/4,
           levels = c("mu1", "mu2", "mu3", "mu4")
)

```

```

cmat(h1)

# An incomplete Helmer contrast matrix (2nd contrast dropped)
h2 <- hyper(~ (mu2-mu1)/2,
            ~ (mu4-(mu1+mu2+mu3)/3)/4,
            levels = c("mu1", "mu2", "mu3", "mu4")
           )
cmat(h2)

# Filling the remaining degree of freedom retrieves the contrast
h3 <- filler_contrasts(h2)
cmat(h3)

stopifnot(all.equal(cmat(h3)[,3], cmat(h1)[,2]))

```

`formula<-` *Manipulate the formulas of an S4 object*

Description

This is a generic function for setting an S4 object's formulas.

Usage

```
formula(x, ...) <- value
```

Arguments

| | |
|--------------------|--|
| <code>x</code> | The object to manipulate |
| <code>...</code> | Additional arguments passed on to the method |
| <code>value</code> | The new formula |

`ginv2` *Enhanced generalized inverse function*

Description

This function is a wrapper for `MASS::ginv` and calculates the generalized inverse of `x`.

Usage

```
ginv2(x, as_fractions = TRUE)
```

Arguments

`x` The original matrix
`as_fractions` Whether to format the matrix as fractions (MASS package)

Details

In addition to `MASS::ginv`, this function rounds values, formats the matrix as fractions and copies dimension names from the original matrix.

Value

Generalized inverse of `x`

See Also

[ginv](#)

Examples

```
h <- hypr(mu1~0, mu2~mu1)
hmat(h)

ginv2(hmat(h))
cmat(h)

# cmat is effectively the generalized inverse of hmat
stopifnot(all.equal(
  `class<-`(ginv2(hmat(h)), "matrix"),
  `class<-`(cmat(h), "matrix"), check.attributes = FALSE))
```

hmat

Retrieve and set hypothesis matrix

Description

Use these functions to retrieve or set a `hypr` object's hypothesis matrix. If used for updating, the contrast matrix and equations are derived automatically.

Usage

```
hmat(x, as_fractions = TRUE)

thmat(x, as_fractions = TRUE)

hmat(x) <- value

thmat(x) <- value
```

Arguments

| | |
|--------------|--|
| x | A hypr object |
| as_fractions | Whether to format matrix as fractions (via <code>MASS::as.fractions</code>) |
| value | Hypothesis matrix |

Value

Hypothesis matrix of x

Functions

- `thmat`: Retrieve transposed hypothesis matrix
- `hmat<-`: Set hypothesis matrix
- `thmat<-`: Set transposed hypothesis matrix

Examples

```
h <- hypr(mu1~0, mu2~mu1)

# To retrieve the hypothesis matrix of `h`:
hmat(h)

# To retrieve the transposed hypothesis matrix of `h`:
thmat(h)

# Setting the hypothesis matrix of `h`:
hmat(h) <- matrix(c(1,-1,0,1), ncol=2, dimnames=list(NULL, c("mu1","mu2")))
h

h2 <- hypr() # an empty hypr object
thmat(h2) <- matrix(c(1,0,-1,1), ncol=2, dimnames=list(c("mu1","mu2"), NULL))
h2

# `h` and `h2` should be identical:
stopifnot(all.equal(hmat(h), hmat(h2)))
stopifnot(all.equal(cmat(h), cmat(h2)))
```

hypr

Create a hypr object

Description

Use this function to create hypr objects from null hypothesis equations. Each argument should be one equation. For example, a null hypothesis for the grand mean (GM), often used as the intercept, is usually coded as $\mu \sim 0$.

Usage

```
hypr(
  ...,
  levels = NULL,
  add_intercept = FALSE,
  remove_intercept = FALSE,
  order_levels = missing(levels)
)
```

Arguments

| | |
|------------------|---|
| ... | A list of null hypothesis equations |
| levels | (Optional) A list of terms/levels to use. If supplied, matrix rows/columns will be in this order. An error will be thrown if an equation contains a level that is not in this vector. |
| add_intercept | If TRUE, an intercept will be added |
| remove_intercept | If TRUE, an intercept will be dropped |
| order_levels | (Optional) Whether to order the rows/columns of the hypothesis/contrast matrices alphabetically. Default is TRUE if levels were not explicitly provided. |

Details

You may call the function without any arguments. In that case, an empty `hypr` object is returned. This is useful if you want to derive equations from a known hypothesis matrix or contrast matrix.

Basic specification of contrasts in R is accomplished with basic R functions `stats::contrasts()` and `stats::C()` (Chambers & Hastie, 1992). Other relevant packages for this topic are `multcomp` (Bretz et al., 2010), `contrast` (Kuhn et al., 2016), and, including also various vignettes, `emmeans` (Lenth, 2019).

Value

A `hypr` object

References

Chambers, J. M. and Hastie, T. J. (1992) *Statistical models*. Chapter 2 of *Statistical Models* in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Frank Bretz, Torsten Hothorn and Peter Westfall (2010), *Multiple Comparisons Using R*, CRC Press, Boca Raton.

Max Kuhn, contributions from Steve Weston, Jed Wing, James Forester and Thorn Thaler (2016). *contrast: A Collection of Contrast Methods*. R package version 0.21. <https://CRAN.R-project.org/package=contrast>

Lenth, R. (2019). *emmeans: Estimated Marginal Means, aka Least-Squares Means*. R package version 1.4.1. <https://CRAN.R-project.org/package=emmeans>

See Also

[contrasts](#) and [C](#) for basic specification of contrasts in R, S4 class [hypr](#), [cmat](#), [contr.hypothesis](#) for retrieval of contrast matrices from [hypr](#) objects

Examples

```
# Create an empty hypr object (no hypotheses):
h <- hypr()

# Treatment contrast:
h <- hypr(mu1~0, mu2~mu1, mu3~mu1, mu4~mu1)

# Identical version:
h <- hypr(~mu1, ~mu2-mu1, ~mu3-mu1, ~mu4-mu1)

contr.hypothesis(h)

# Generate a dataset
set.seed(123)
M <- c(mu1 = 10, mu2 = 20, mu3 = 10, mu4 = 40) # condition means
N <- 5 # number of observations per condition
SD <- 10 # residual SD
simdat <- do.call(rbind, lapply(names(M), function(x) {
  data.frame(X = x, DV = as.numeric(MASS::mvrnorm(N, unname(M[x]), SD^2, empirical = TRUE)))
}))
simdat$X <- factor(simdat$X, levels=levels(h))
simdat

# Check agreement of hypothesis levels and factor levels
stopifnot(levels(h) == levels(simdat$X))

# Linear regression
contrasts(simdat$X) <- contr.hypothesis(h)

round(coef(summary(lm(DV ~ X, data=simdat))),3)
```

hypr-class

S4 class “hypr” and its methods

Description

A [hypr](#) object contains equations, a hypothesis matrix and a contrast matrix, all of which are related to each other. See below for methods.

Usage

```
## S4 method for signature 'hypr'
show(object)

## S4 method for signature 'hypr'
levels(x)

## S4 method for signature 'hypr'
nlevels(x)

## S4 method for signature 'hypr'
names(x)

## S4 replacement method for signature 'hypr'
names(x) <- value

## S4 replacement method for signature 'hypr'
levels(x) <- value

## S4 method for signature 'hypr'
formula(x, ...)

## S4 replacement method for signature 'hypr'
formula(x, ...) <- value
```

Arguments

| | |
|-----------|--|
| object, x | a hypr object |
| value | New value (list of equations for formula, character vector for levels and names) |
| ... | (ignored) |

Details

To generate a hypr object, use the [hypr](#) function.

Value

A character vector of level names
 An integer denoting the number of levels
 A character vector of contrast names
 A list of null hypothesis equations

Methods (by generic)

- `show`: Show summary of hypr object, including contrast equations, the (transposed) hypothesis matrix and the derived contrast matrix.

- `levels`: Retrieve the levels (variable names) used in a `hypr` object
- `nlevels`: Retrieve the number of levels (variable names) used in a `hypr` object
- `names`: Retrieve the contrast names used in a `hypr` object
- `names<-`: Set the contrast names used in a `hypr` object
- `levels<-`: Set the levels used in a `hypr` object
- `formula`: Retrieve a `hypr` object's null hypothesis equations.
- `formula<-`: Modify a `hypr` object's null hypothesis equations

Slots

`eqs` List of null hypotheses
`hmat` Hypothesis matrix
`cmat` Contrast matrix

See Also

[hypr](#), [cmat](#), [hmat](#)

Examples

```
# Equations and matrices in a hypr object are always congruent
# Therefore creating a hypr object h and then copying ...
h <- hypr(mu1~0, mu2~mu1)

# ... its equations, ...
h2 <- hypr()
formula(h2) <- formula(h)

# ... its hypothesis matrix, ...
h3 <- hypr()
hmat(h3) <- hmat(h)

# ... or its contrast matrix ...
h4 <- hypr()
cmat(h4) <- cmat(h)

# ... over to another hypr object is the same as copying the object:
h5 <- h

# check that hypr objects are equal by comparing hmat() and cmat()
stopifnot(all.equal(hmat(h), hmat(h2)))
stopifnot(all.equal(cmat(h), cmat(h2)))
stopifnot(all.equal(hmat(h), hmat(h3)))
stopifnot(all.equal(cmat(h), cmat(h3)))
stopifnot(all.equal(hmat(h), hmat(h4)))
stopifnot(all.equal(cmat(h), cmat(h4)))
stopifnot(all.equal(hmat(h), hmat(h5)))
stopifnot(all.equal(cmat(h), cmat(h5)))
```



```
h <- hypr(mu1~0, mu2~mu1)
formula(h)

h2 <- hypr()
formula(h2) <- formula(h)
h2
formula(h2)

# After updating, matrices should be equal
stopifnot(all.equal(hmat(h), hmat(h2)))
stopifnot(all.equal(cmat(h), cmat(h2)))
```

Index

`*`, `hypr`, `hypr-method`
 (`+`, `hypr`, `hypr-method`), 2
`+`, `hypr`, `hypr-method`, 2
`/`, `hypr`, `hypr-method`
 (`+`, `hypr`, `hypr-method`), 2
`&`, `hypr`, `hypr-method`
 (`+`, `hypr`, `hypr-method`), 2

`add_intercept`, 3
`all_centered` (`centered_contrasts`), 4

C, 14
`centered_contrasts`, 4
`cmat`, 5, 14, 16
`cmat2eqs` (`conversions`), 7
`cmat2hmat` (`conversions`), 7
`cmat`<- (`cmat`), 5
`contr.hypothesis`, 14
`contr.hypothesis` (`cmat`), 5
`contr.hypothesis`<- (`cmat`), 5
`contrasts`, 14
`contrasts`<- ,ANY,ANY, `hypr-method` (`cmat`),
 5
`contrasts`<- ,ANY,ANY, `hypr_cmat-method`
 (`cmat`), 5
`conversions`, 7

`eqs2cmat` (`conversions`), 7
`eqs2hmat` (`conversions`), 7

`filler_contrasts`, 9
`formula`, `hypr-method` (`hypr-class`), 14
`formula`<- , 10
`formula`<- , `hypr-method` (`hypr-class`), 14

`ginv`, 11
`ginv2`, 10

`has_intercept` (`add_intercept`), 3
`hmat`, 11, 16
`hmat2cmat` (`conversions`), 7

`hmat2eqs` (`conversions`), 7
`hmat`<- (`hmat`), 11
`hypr`, 7, 12, 14–16
`hypr-class`, 14

`is_centered` (`centered_contrasts`), 4
`is_intercept` (`add_intercept`), 3

`levels`, `hypr-method` (`hypr-class`), 14
`levels`<- , `hypr-method` (`hypr-class`), 14

`MASS::as.fractions`, 8, 12
`MASS::as.fractions`() , 6

`names`, `hypr-method` (`hypr-class`), 14
`names`<- , `hypr-method` (`hypr-class`), 14
`nlevels`, `hypr-method` (`hypr-class`), 14

`remove_intercept` (`add_intercept`), 3

`show`, `hypr-method` (`hypr-class`), 14
`stats::C`() , 6, 13
`stats::contrasts`() , 6, 13

`thmat` (`hmat`), 11
`thmat`<- (`hmat`), 11

`which_centered` (`centered_contrasts`), 4
`which_intercept` (`add_intercept`), 3