Package: lotri (via r-universe)

June 25, 2024

Title A Simple Way to Specify Symmetric, Block Diagonal Matrices

Version 0.4.4.9000

Maintainer Matthew L. Fidler <matthew.fidler@gmail.com>

Description Provides a simple mechanism to specify a symmetric block diagonal matrices (often used for covariance matrices). This is based on the domain specific language implemented in 'nlmixr2' but expanded to create matrices in R generally instead of specifying parts of matrices to estimate.

License GPL (>=2)

URL https://nlmixr2.github.io/lotri/, https://github.com/nlmixr2/lotri

BugReports https://github.com/nlmixr2/lotri/issues

Depends R (>= 3.4.0)

Imports crayon, methods, stats, utils

Suggests ggplot2, knitr, Matrix, microbenchmark, rmarkdown, testthat

VignetteBuilder knitr

Biarch true

Config/testthat/edition 3

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 7.3.1

Language en-US

Repository https://nlmixr2.r-universe.dev

RemoteUrl https://github.com/nlmixr2/lotri

RemoteRef HEAD

RemoteSha ce4847148725391ed0b00fc5197fa963de212019

Contents

`as.lotri` *As lower triangular matrix*

Description

As lower triangular matrix

Usage

```
as.lotri(x, ..., default = "")

## S3 method for class 'matrix'
as.lotri(x, ..., default = "")

## S3 method for class 'data.frame'
as.lotri(x, ..., default = "")

## Default S3 method:
as.lotri(x, ..., default = "")
```

Arguments

`x`	Matrix or other data frame
`...`	Other factors
`default`	Is the default factor when no conditioning is implemented.

Value

Lower triangular matrix

Author(s)

Matthew Fidler

lotri *Easily Specify block-diagonal matrices with lower triangular info*

Description

Easily Specify block-diagonal matrices with lower triangular info

Usage

```
lotri(x, ..., cov = FALSE, envir = parent.frame(), default = "id")
```

Arguments

x	list, matrix or expression, see details
...	Other arguments treated as a list that will be concatenated then reapplied to this function.
cov	either a boolean or a function accepting a matrix input. When a boolean, 'cov' describes if this matrix definition is actually a rxode2/nlmixr2-style covariance matrix. If so, 'lotri()' will enforce certain regularity conditions: - When diagonal elements are zero, the off-diagonal elements are zero. This means the covariance element is fixed to zero and not truly part of the covariance matrix in general. - For the rest of the matrix, 'lotri' will check that it is non-positive definite (which is required for covariance matrix in general) It is sometimes difficult to adjust covariance matrices to be non-positive definite. For this reason 'cov' may also be a function accepting a matrix input and returning a non-positive definite matrix from this matrix input. When this is a function, it is equivalent to 'cov=TRUE' with the additional ability to correct the matrix to be non-positive definite if needed.
envir	the environment in which expr is to be evaluated. May also be NULL, a list, a data frame, a pairlist or an integer as specified to sys.call.
default	Is the default factor when no conditioning is implemented.

Details

This can take an R matrix, a list including matrices or expressions, or expressions

Expressions can take the form

name ~ estimate

Or the lower triangular matrix when "adding" the names

name1 + name2 ~ c(est1, est2, est3)

The matrices are concatenated into a block diagonal matrix, like bdiag, but allows expressions to specify matrices easier.

Value

named symmetric matrix useful in ‘rxode2()‘ simulations (and perhaps elsewhere)

Author(s)

Matthew L Fidler

Examples

```
## A few ways to specify the same matrix
lotri({et2 + et3 + et4 ~ c(40,
                           0.1, 20,
                           0.1, 0.1, 30)})

## You  do not need to enclose in {}
lotri(et2 + et3 + et4 ~ c(40,
                          0.1, 20,
                          0.1, 0.1, 30),
          et5 ~ 6)
## But if you do enclose in {}, you can use
## multi-line matrix specifications:

lotri({et2 + et3 + et4 ~ c(40,
                           0.1, 20,
                           0.1, 0.1, 30)
          et5 ~ 6
          })

## You can also add lists or actual R matrices as in this example:
lotri(list(et2 + et3 + et4 ~ c(40,
                               0.1, 20,
                               0.1, 0.1, 30),
              matrix(1,dimnames=list("et5","et5"))))

## Overall this is a flexible way to specify symmetric block
## diagonal matrices.

## For rxode2, you may also condition based on different levels of
## nesting with lotri;  Here is an example:

mat <- lotri(lotri(iov.Ka ~ 0.5,
                   iov.Cl ~ 0.6),
             lotri(occ.Ka ~ 0.5,
                   occ.Cl ~ 0.6) | occ(lower=4,nu=3))

mat

## you may access features of the matrix simply by `$` that is

mat$lower # Shows the lower bound for each condition

mat$lower$occ # shows the lower bound for the occasion variable
```

```
## Note that `lower` fills in defaults for parameters.  This is true
## for `upper` true;  In fact when accessing this the defaults
## are put into the list

mat$upper

## However all other values return NULL if they are not present like

mat$lotri

## And values that are specified once are only returned on one list:

mat$nu

mat$nu$occ
mat$nu$id

## You can also change the default condition with `as.lotri`

mat <- as.lotri(mat, default="id")

mat
```

`lotriDataFrameToLotriExpression` *Convert a lotri data frame to a lotri expression*

Description

Convert a lotri data frame to a lotri expression

Usage

```
lotriDataFrameToLotriExpression(data, useIni = FALSE)
```

Arguments

`data`	lotri data frame
`useIni`	Use ‘ini‘ instead of ‘lotri‘ in the expression

Value

expression of the lotri syntax equivalent to the data.frame provided

Author(s)

Matthew L. Fidler

Examples

```
 x <- lotri({
  tka <- 0.45; label("Log Ka")
  tcl <- 1; label("Log Cl")
  tv <- 3.45; label("Log V")
  eta.ka ~ 0.6
  eta.cl ~ 0.3
  eta.v ~ 0.1
  add.err <- 0.7
})

df <- as.data.frame(x)

lotriDataFrameToLotriExpression(df)

# You may also call as.expression directly from the lotri object

as.expression(x)
```

`lotriEst` *Extract or remove lotri estimate data frame from lotri object*

Description

Extract or remove lotri estimate data frame from lotri object

Usage

```
lotriEst(x, drop = FALSE)
```

Arguments

x	lotri object
drop	boolean indicating if the lotri estimate should be dropped

Value

data frame with estimates or NULL if there is not a data.frame attached

Examples

```
fix1 <- lotri({
   a <- c(0, 1); backTransform("exp"); label("a label")
   b <- c(0, 1, 2)
   c <- fix(1)
   d <- fix(0, 1, 2)
   e <- c(0, 1, 2, fixed)
   f+g ~ c(1,
```

```
          0.5, 1)
 })

# Extract the attached lotri estimate data frame
lotriEst(fix1)

# Remove the attached lotri estimate data frame
lotriEst(fix1, drop=TRUE)
```

`lotriMat` *Create a matrix from a list of matrices*

Description

This creates a named banded symmetric matrix from a list of named symmetric matrices.

Usage

```
lotriMat(matList, format = NULL, start = 1L)
```

Arguments

`matList`	list of symmetric named matrices
`format`	The format of dimension names when a sub-matrix is repeated. The format will be called with the dimension number, so "ETA[%d]" would represent "ETA[1]", "ETA[2]", etc
`start`	The number the counter of each repeated dimension should start.

Value

Named symmetric block diagonal matrix based on concatenating the list of matrices together

Author(s)

Matthew Fidler

Examples

```
testList <- list(lotri({et2 + et3 + et4 ~ c(40,
                           0.1, 20,
                           0.1, 0.1, 30)}),
                 lotri(et5 ~ 6))

testList

lotriMat(testList)
```

```
# Another option is to repeat a matrix a number of times.  This
# can be done with list(matrix, # times to repeat).

# In the example below, the first matrix is repeated 3 times
testList <- list(list(lotri({et2 + et3 + et4 ~ c(40,
                           0.1, 20,
                           0.1, 0.1, 30)}), 3),
                 lotri(et5 ~ 6))

lotriMat(testList)

# Notice that the dimension names `et2`, `et3` and `et4` are
# repeated.

# Another option is to name the dimensions.  For example it could
# be `ETA[1]`, `ETA[2]`, etc by using the 'format' option:

lotriMat(testList, "ETA[%d]")

# Or could start with ETA[2]:

lotriMat(testList, "ETA[%d]", 2)
```

`lotriMatInv` *Converts a matrix into a list of block matrices*

Description

Converts a matrix into a list of block matrices

Usage

```
lotriMatInv(mat)
```

Arguments

`mat`	Matrix to convert to a list of block matrices

Details

This is the inverse of ‘lotriMat()‘

Value

A list of block matrixes

Author(s)

Matthew Fidler

Examples

```
# Create a block matrix using `lotri()`
mat <- lotri({
   a+b ~ c(1,
           0.5, 1)
   c ~ 1
   d +e ~ c(1,
            0.5, 1)
})

print(mat)

# now convert t a list of matrices

mat2 <- lotriMatInv(mat)
print(mat2)

# Of course you can convert it back to a full matrix:

mat3 <- lotriMat(mat2)

print(mat3)
```

`lotriSep` *Separate a lotri matrix into above and below lotri matrices*

Description

This is used for creating nesting simulations in ‘rxode2()‘ and may not be useful for external function calls.

Usage

```
lotriSep(x, above, below, aboveStart = 1L, belowStart = 1L)
```

Arguments

`x`	lotri matrix
`above`	Named integer vector listing variability above the id level. Each element lists the number of population differences in the whole data-set (as integer)
`below`	Named integer vector listing variability below the id level. Each element lists the number of items below the individual level. For example with 3 occasions per individual you could use ’c(occ=3L)’
`aboveStart`	Add the attribute of where THETA[#] will be added
`belowStart`	Add the attribute of where ETA[#] will be added

Value

List of two lotri matrices

Author(s)

Matthew Fidler

Examples

```
omega <- lotri(lotri(eta.Cl ~ 0.1,
                       eta.Ka ~ 0.1) | id(nu=100),
                 lotri(eye.Cl ~ 0.05,
                       eye.Ka ~ 0.05) | eye(nu=50),
                 lotri(iov.Cl ~ 0.01,
                       iov.Ka ~ 0.01) | occ(nu=200),
                 lotri(inv.Cl ~ 0.02,
                       inv.Ka ~ 0.02) | inv(nu=10))

lotriSep(omega, above=c(inv=10L), below=c(eye=2L, occ=4L))
```

Index