Package ‘lotri’

September 18, 2024

Title A Simple Way to Specify Symmetric, Block Diagonal Matrices

Version 1.0.0

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. It has expanded to include some matrix manipulation functions that are generally useful for 'rxode2' and 'nlmixr2'.

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 checkmate, 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 7.3.2

Language en-US

LinkingTo cpp11, cpp11armadillo

Author Matthew L. Fidler [aut, cre] (<https://orcid.org/0000-0001-8538-6691>),
Bill Denney [ctb] (<https://orcid.org/0000-0002-5759-428X>)

Repository CRAN

Date/Publication 2024-09-18 04:50:06 UTC

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, rcm = 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.

`rcm` logical; if 'TRUE', the matrix will be reordered to change the matrix to a banded matrix, which is easier to express in 'lotri' than a full matrix. The RCM stands for the reverse Cuthill McKee (RCM) algorithm which is used for this matrix permutation. (see 'rcm()')

`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
```

lotriAsExpression *Change a matrix or lotri matrix to a lotri expression*

Description

Change a matrix or lotri matrix to a lotri expression

Usage

```
lotriAsExpression(
  x,
  useIni = FALSE,
  plusNames = getOption("lotri.plusNames", FALSE),
  nameEst = getOption("lotri.nameEst", 5L)
)
```

Arguments

x	matrix
useIni	use the ini block

`plusNames` logical, when ‘TRUE‘ use the ‘a + b ~ c(1, 0.1, 1)‘ naming convention. Otherwise use the lotri single line convention ‘a ~ 1; b ~ c(0.1, 1)‘

`nameEst` logical or integerish. When logical ‘TRUE‘ will add names to all matrix estimates and ‘TRUE‘ when using the lotri single line convention i.e. ‘a~c(a=1); b~c(a=0.1, b=1)‘. When an integer, the dimension of the matrix being displayed needs to have a dimension above this number before names are displayed.

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)
```

`lotriIsBlockMat` *Determine if the matrix is a block matrix*

Description

Determine if the matrix is a block matrix

Usage

```
lotriIsBlockMat(mat)
```

Arguments

`mat`	matrix to determine if it is a block matrix

Value

logical value, TRUE if it is a block matrix and FALSE otherwise

Author(s)

Matthew L. Fidler

Examples

```
m <- lotri({
  a ~ c(a = 0.4)
  b ~ c(a = 0, b = 0.3)
  c ~ c(a = 0, b = 0, c = 0)
  d ~ c(a = -0.1, b = 0, c = 0, d = 0.2)
  e ~ c(a = 0, b = 0, c = 0, d = 0, e = 0.5)
  f ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 1.3)
  g ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = -0.6, g = 0.8)
  h ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0)
  i ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0,
        i = 0.2)
  j ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0,
        i = 0, j = 0.9)
  k ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0,
        i = 0, j = 0, k = 0.9)
  l ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0,
        i = 0, j = -0.2, k = 0, l = 0.3)
  m ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0,
        i = 0, j = 0, k = 0, l = 0, m = 2.1)
  n ~ c(a = 0.2, b = 0, c = 0, d = 0.2, e = 0, f = 0, g = 0,
        h = 0, i = 0, j = 0, k = 0, l = 0, m = 0, n = 0.4)
  o ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = -1.1, g = 0.9,
        h = 0, i = 0, j = 0, k = 0, l = 0, m = 0, n = 0, o = 4.7)
  p ~ c(a = 0, b = 0, c = 0, d = 0, e = 0, f = 0, g = 0, h = 0,
        i = 0, j = 0.5, k = 0, l = 0.2, m = 0, n = 0, o = 0,
      p = 1.9)
})

lotriIsBlockMat(m)

lotriIsBlockMat(rcm(m))
```

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)
```

---

lotriNearPD *C++ implementation of Matrix's nearPD*

Description

With ‘ensureSymmetry‘ it makes sure it is symmetric by applying 0.5*(t(x) + x) before using lotriNearPD

Usage

```
lotriNearPD(
  x,
  keepDiag = FALSE,
  do2eigen = TRUE,
  doDykstra = TRUE,
  only.values = FALSE,
  ensureSymmetry = !isSymmetric(x),
  eig.tol = 1e-06,
  conv.tol = 1e-07,
  posd.tol = 1e-08,
  maxit = 100L,
  trace = FALSE
)
```

Arguments

`x` numeric $n \times n$ approximately positive definite matrix, typically an approximation to a correlation or covariance matrix. If `x` is not symmetric (and `ensureSymmetry` is not false), `symmpart(x)` is used.

`keepDiag` logical, generalizing `corr`: if `TRUE`, the resulting matrix should have the same diagonal (`diag(x)`) as the input matrix.

`do2eigen` logical indicating if a ‘posdefify()‘ (like in the package ‘sfsmisc‘) eigen step should be applied to the result of the Higham algorithm

`doDykstra` logical indicating if Dykstra’s correction should be used; true by default. If false, the algorithm is basically the direct fixpoint iteration $Y_k = P_U(P_S(Y_{k-1}))$.

`only.values` logical; if `TRUE`, the result is just the vector of eigenvalues of the approximating matrix.

`ensureSymmetry` logical; by default, `symmpart(x)` is used whenever `isSymmetric(x)` is not true. The user can explicitly set this to `TRUE` or `FALSE`, saving the symmetry test. *Beware* however that setting it `FALSE` for an **a**symmetric input `x`, is typically nonsense!

`eig.tol` defines relative positiveness of eigenvalues compared to largest one, λ_1. Eigenvalues λ_k are treated as if zero when $\lambda_k/\lambda_1 \le eig.tol$.

`conv.tol` convergence tolerance for Higham algorithm.

`posd.tol` tolerance for enforcing positive definiteness (in the final `posdefify` step when `do2eigen` is `TRUE`).

`maxit` maximum number of iterations allowed.

`trace` logical or integer specifying if convergence monitoring should be traced.

Details

This implements the algorithm of Higham (2002), and then (if `do2eigen` is true) forces positive definiteness using code from ‘sfsmisc::posdefify()‘. The algorithm of Knol and ten Berge (1989) (not implemented here) is more general in that it allows constraints to (1) fix some rows (and columns) of the matrix and (2) force the smallest eigenvalue to have a certain value.

Note that setting `corr = TRUE` just sets `diag(.) <- 1` within the algorithm.

Higham (2002) uses Dykstra’s correction, but the version by Jens Oehlschlägel did not use it (accidentally), and still gave reasonable results; this simplification, now only used if `doDykstra = FALSE`, was active in `nearPD()` up to Matrix version 0.999375-40.

Value

unlike the matrix package, this simply returns the nearest positive definite matrix

Author(s)

Jens Oehlschlägel donated a first version to Matrix. Subsequent changes by the Matrix package authors, later modifications to C++ by Matthew Fidler

References

Cheng, Sheung Hun and Higham, Nick (1998) A Modified Cholesky Algorithm Based on a Symmetric Indefinite Factorization; *SIAM J. Matrix Anal.\ Appl.*, **19**, 1097–1110.

Knol DL, ten Berge JMF (1989) Least-squares approximation of an improper correlation matrix by a proper one. *Psychometrika* **54**, 53–61.

Higham, Nick (2002) Computing the nearest correlation matrix - a problem from finance; *IMA Journal of Numerical Analysis* **22**, 329–343.

See Also

A first version of this (with non-optional `corr=TRUE`) has been available as ‘sfsmisc::nearcor()‘ and more simple versions with a similar purpose ‘sfsmisc::posdefify()‘

Examples

```
set.seed(27)
m <- matrix(round(rnorm(25),2), 5, 5)
m <- m + t(m)
diag(m) <- pmax(0, diag(m)) + 1
(m <- round(cov2cor(m), 2))

near.m <- lotriNearPD(m)
round(near.m, 2)
norm(m - near.m) # 1.102 / 1.08

round(lotriNearPD(m, only.values=TRUE), 9)

## A longer example, extended from Jens' original,
## showing the effects of some of the options:

pr <- matrix(c(1,     0.477, 0.644, 0.478, 0.651, 0.826,
               0.477, 1,     0.516, 0.233, 0.682, 0.75,
               0.644, 0.516, 1,     0.599, 0.581, 0.742,
               0.478, 0.233, 0.599, 1,     0.741, 0.8,
               0.651, 0.682, 0.581, 0.741, 1,     0.798,
               0.826, 0.75,  0.742, 0.8,   0.798, 1),
               nrow = 6, ncol = 6)

nc  <- lotriNearPD(pr)
```

`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))
```

`rcm` *Use the RCM algorithm to permute to banded matrix*

Description

The RCM stands for the reverse Cuthill McKee (RCM) algorithm which is used to permute the matrix to a banded matrix.

Usage

```
rcm(x)
```

Arguments

x A symmetric matrix

Value

A permuted matrix that should be banded

Examples

```
m <- lotri({
 a + b + c + d + e + f + g + h + i + j + k + l + m + n + o +
 p ~ c(0.4, 0, 0.3, 0, 0, 0, -0.1, 0, 0, 0.2, 0, 0, 0,
       0, 0.5, 0, 0, 0, 0, 0, 1.3, 0, 0, 0, 0, 0, -0.6, 0.8,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.2,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0.9, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0.9, 0, 0, 0, 0, 0, 0, 0, 0, 0, -0.2, 0, 0.3,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2.1, 0.2, 0, 0, 0.2,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0.4, 0, 0, 0, 0, 0, -1.1,
       0.9, 0, 0, 0, 0, 0, 0, 0, 4.7, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0.5, 0, 0.2, 0, 0, 0, 1.9)
})

rcm(m)
```

Index