Package ‘LMMsolver’

December 5, 2025

Type Package

Title Linear Mixed Models with Sparse Matrix Methods and Smoothing

Description Provides tools for fitting linear mixed models using sparse matrix methods and variance component estimation. Applications include spline-based modeling of spatial and temporal trends using penalized splines (Boer, 2023) <doi:10.1177/1471082X231178591>.

Version 1.0.12

Date 2025-12-03

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.6)

Imports Matrix, methods, Rcpp (>= 0.10.4), spam, splines

LinkingTo Rcpp

RoxygenNote 7.3.3

Suggests rmarkdown, knitr, tinytest, tidyr, ggplot2, maps, sf

VignetteBuilder knitr

URL `https://biometris.github.io/LMMsolver/index.html`, `https://github.com/Biometris/LMMsolver/`

BugReports `https://github.com/Biometris/LMMsolver/issues`

NeedsCompilation yes

Author Martin Boer [aut] (ORCID: <`https://orcid.org/0000-0002-1879-4588`>), Bart-Jan van Rossum [aut, cre] (ORCID: <`https://orcid.org/0000-0002-8673-2514`>)

Maintainer Bart-Jan van Rossum <`bart-jan.vanrossum@wur.nl`>

Repository CRAN

Date/Publication 2025-12-05 08:20:02 UTC

Contents

`APSIMdat` *Simulated Biomass as function of time using APSIM wheat.*

Description

Simulated Biomass as function of time using APSIM wheat.

Usage

```
APSIMdat
```

Format

A data.frame with 121 rows and 4 columns.

env Environment, Emerald in 1993

geno Simulated genotype g001

das Days after sowing

biomass Simulated biomass using APSIM; medium measurement error added

References

Bustos-Korts et al. (2019) Combining Crop Growth Modeling and Statistical Genetic Modeling to Evaluate Phenotyping Strategies doi:10.3389/FPLS.2019.01491

`barley.uniformity.trial`

Uniformity trial of barley

Description

Uniformity trial of barley

Usage

```
barley.uniformity.trial
```

Format

A data.frame with 1076 rows and 3 columns

row row coordinate

col column coordinate

yield yield per plot

Source

H. P. Piepho & E. R. Williams (2010). Linear variance models for plant breeding trials. Plant Breeding, 129, 1-8. doi:10.1111/j.14390523.2009.01654.x

References

Piepho, Hans-Peter, Martin P. Boer, and Emlyn R. Williams. "Two-dimensional P-spline smoothing for spatial analysis of plant breeding trials." Biometrical Journal 64, no. 5 (2022): 835-857.

`coef.LMMsolve` *Coefficients from the mixed model equations of an LMMsolve object.*

Description

Obtain the coefficients from the mixed model equations of an LMMsolve object.

Usage

```
## S3 method for class 'LMMsolve'
coef(object, se = FALSE, ...)
```

Arguments

`object`	an object of class LMMsolve
`se`	calculate standard errors, default FALSE.
`...`	some methods for this generic require additional arguments. None are used in this method.

Value

A list of vectors, containing the estimated effects for each fixed effect and the predictions for each random effect in the defined linear mixed model.

Examples

```
## Fit model on oats data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                data = oats.data)

## Obtain coefficients.
coefs1 <- coef(LMM1)

## Obtain coefficients with standard errors.
coefs2 <- coef(LMM1, se = TRUE)
```

`deviance.LMMsolve` *Deviance of an LMMsolve object*

Description

Obtain the deviance of a model fitted using LMMsolve.

Usage

```
## S3 method for class 'LMMsolve'
deviance(object, relative = TRUE, includeConstant = TRUE, ...)
```

Arguments

`object`	an object of class LMMsolve
`relative`	Deviance relative conditional or absolute unconditional (-2*logLik(object))? Default `relative = TRUE`.
`includeConstant`	Should the constant in the restricted log-likelihood be included. Default is `TRUE`, as for example in `lme4` and SAS. In `asreml` the constant is omitted.
`...`	some methods for this generic require additional arguments. None are used in this method.

Value

The deviance of the fitted model.

Examples

```
## Fit model on oats.data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Obtain deviance.
deviance(LMM1)
```

`diagnosticsMME` *Give diagnostics for mixed model coefficient matrix C and the cholesky decomposition*

Description

Give diagnostics for mixed model coefficient matrix C and the cholesky decomposition

Usage

```
diagnosticsMME(object)
```

Arguments

`object`	an object of class LMMsolve.

Value

A summary of the mixed model coefficient matrix and its choleski decomposition.

Examples

```
## Fit model on oats data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Obtain deviance.
diagnosticsMME(LMM1)
```

`displayMME` *Display the sparseness of the mixed model coefficient matrix*

Description

Display the sparseness of the mixed model coefficient matrix

Usage

```
displayMME(object, cholesky = FALSE)
```

Arguments

`object`	an object of class LMMsolve.
`cholesky`	Should the cholesky decomposition of the coefficient matrix be plotted?

Value

A plot of the sparseness of the mixed model coefficient matrix.

Examples

```
## Fit model on oats data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Obtain deviance.
displayMME(LMM1)
```

`effDim` *Function to get the Effective Dimensions.*

Description

Function to get the Effective Dimensions.

Usage

```
effDim(object)
```

Arguments

`object`	an object of class LMMsolve

Value

A data.frame with the effective dimensions and penalties.

#' @examples ## Fit model on oats data data(oats.data)

Fit a model with a 1-dimensional spline at the plot level. obj <- LMMsolve(fixed = yield ~ rep + gen, spline = ~spl1D(x = plot, nseg = 20), data = oats.data) effDim(obj)

`fitted.LMMsolve` *Fitted values of an LMMsolve object.*

Description

Obtain the fitted values from a mixed model fitted using LMMSolve.

Usage

```
## S3 method for class 'LMMsolve'
fitted(object, ...)
```

Arguments

`object`	an object of class LMMsolve
`...`	some methods for this generic require additional arguments. None are used in this method.

Value

A vector of fitted values.

Examples

```
## Fit model on oats data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Obtain fitted values.
fitted1 <- fitted(LMM1)
```

`LMMsolve` *Solve Linear Mixed Models*

Description

Solve Linear Mixed Models using REML.

Usage

```
LMMsolve(
  fixed,
  random = NULL,
  spline = NULL,
  group = NULL,
  ginverse = NULL,
  weights = NULL,
  data,
  residual = NULL,
  family = gaussian(),
  offset = 0,
  tolerance = 1e-06,
  trace = FALSE,
  maxit = 250,
  theta = NULL,
  grpTheta = NULL
)
```

Arguments

`fixed`	A formula for the fixed part of the model. Should be of the form "response ~ pred"
`random`	A formula for the random part of the model. Should be of the form "~ pred".
`spline`	A formula for the spline part of the model. Should be of the form "~ spl1D()", ~ spl2D()" or "~spl3D()". Generalized Additive Models (GAMs) can also be used, for example "~ spl1D() + spl2D()"
`group`	A named list where each component is a numeric vector specifying contiguous fields in data that are to be considered as a single term.
`ginverse`	A named list with each component a symmetric matrix, the precision matrix of a corresponding random term in the model. The row and column order of the precision matrices should match the order of the levels of the corresponding factor in the data.
`weights`	A character string identifying the column of data to use as relative weights in the fit. Default value NULL, weights are all equal to one.
`data`	A data.frame containing the modeling data.
`residual`	A formula for the residual part of the model. Should be of the form "~ pred".
`family`	An object of class `family` or `familyLMMsolver` specifying the distribution and link function. See class `family` and and `multinomial` for details.
`offset`	An a priori known component to be included in the linear predictor during fitting. `Offset` be a numeric vector, or a character string identifying the column of data. Default `offset` = `0`.
`tolerance`	A numerical value. The convergence tolerance for the modified Henderson algorithm to estimate the variance components.
`trace`	Should the progress of the algorithm be printed? Default `trace` = `FALSE`.

`maxit`	A numerical value. The maximum number of iterations for the algorithm. Default `maxit = 250`.
`theta`	initial values for penalty or precision parameters. Default `NULL`, all precision parameters set equal to 1.
`grpTheta`	a vector to give components the same penalty. Default `NULL`, all components have a separate penalty.

Details

A Linear Mixed Model (LMM) has the form

$$y = X\beta + Zu + e, u \sim N(0, G), e \sim N(0, R)$$

where y is a vector of observations, β is a vector with the fixed effects, u is a vector with the random effects, and e a vector of random residuals. X and Z are design matrices.

LMMsolve can fit models where the matrices G^{-1} and R^{-1} are a linear combination of precision matrices $Q_{G,i}$ and $Q_{R,i}$:

$$G^{-1} = \sum_i \psi_i Q_{G,i} \; , R^{-1} = \sum_i \phi_i Q_{R,i}$$

where the precision parameters ψ_i and ϕ_i are estimated using REML. For most standard mixed models $1/\psi_i$ are the variance components and $1/\phi_i$ the residual variances. We use a formulation in terms of precision parameters to allow for non-standard mixed models using tensor product splines.

Value

An object of class `LMMsolve` representing the fitted model. See `LMMsolveObject` for a full description of the components in this object.

See Also

`LMMsolveObject`, `spl1D`, `spl2D`, `spl3D`

Examples

```
## Fit models on oats.data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Fit the same model with genotype as random effect.
LMM1_rand <- LMMsolve(fixed = yield ~ rep,
                      random = ~gen,
                      data = oats.data)

## Fit the model with a 1-dimensional spline at the plot level.
LMM1_spline <- LMMsolve(fixed = yield ~ rep + gen,
                        spline = ~spl1D(x = plot, nseg = 20),
```

```
                       data = oats.data)

## Fit models on multipop data included in the package.
data(multipop)

## The residual variances for the two populations can be different.
## Allow for heterogeneous residual variances using the residual argument.
LMM2 <- LMMsolve(fixed = pheno ~ cross,
                residual = ~cross,
                data = multipop)

## QTL-probabilities are defined by the columns pA, pB, pC.
## They can be included in the random part of the model by specifying the
## group argument and using grp() in the random part.

# Define groups by specifying columns in data corresponding to groups in a list.
# Name used in grp() should match names specified in list.
lGrp <- list(QTL = 3:5)
LMM2_group <- LMMsolve(fixed = pheno ~ cross,
                      group = lGrp,
                      random = ~grp(QTL),
                      residual = ~cross,
                      data = multipop)
```

`LMMsolveObject` *Fitted LMMsolve Object*

Description

An object of class `LMMsolve` returned by the LMMsolve function, representing a fitted linear mixed model. Objects of this class have methods for the generic functions coef, fitted, residuals, loglik and deviance.

Value

An object of class `LMMsolve` contains the following components:

`logL`	The restricted log-likelihood at convergence
`sigma2e`	The residual error
`tau2e`	The estimated variance components
`EDdf`	The effective dimensions
`varPar`	The number of variance parameters for each variance component
`VarDf`	The table with variance components
`theta`	The precision parameters
`coefMME`	A vector with all the estimated effects from mixed model equations

`ndxCoefficients`	The indices of the coefficients with the names
`yhat`	The fitted values
`residuals`	The residuals
`nIter`	The number of iterations for the mixed model to converge
`y`	Response variable
`X`	The design matrix for the fixed part of the mixed model
`Z`	The design matrix for the random part of the mixed model
`lGinv`	List with precision matrices for the random terms
`lRinv`	List with precision matrices for the residual
`C`	The mixed model coefficient matrix after last iteration
`cholC`	The cholesky decomposition of coefficient matrix C
`constantREML`	The REML constant
`dim`	The dimensions for each of the fixed and random terms in the mixed model
`term.labels.f`	The names of the fixed terms in the mixed model
`term.labels.r`	The names of the random terms in the mixed model
`respVar`	The name(s) of the response variable(s).
`splRes`	An object with definition of spline argument
`deviance`	The relative deviance
`family`	An object of class family specifying the distribution and link function
`trace`	A data.frame with the convergence sequence for the log likelihood and effective dimensions

.

`logLik.LMMsolve` *Log-likelihood of an LMMsolve object*

Description

Obtain the Restricted Maximum Log-Likelihood of a model fitted using LMMsolve.

Usage

```
## S3 method for class 'LMMsolve'
logLik(object, includeConstant = TRUE, ...)
```

Arguments

`object`	an object of class LMMsolve
`includeConstant`	Should the constant in the restricted log-likelihood be included. Default is `TRUE`, as for example in `lme4` and SAS. In `asreml` the constant is omitted.
`...`	some methods for this generic require additional arguments. None are used in this method.

Value

The restricted maximum log-likelihood of the fitted model.

Examples

```
## Fit model on oats data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Obtain log-likelihood.
logLik(LMM1)

## Obtain log-likelihood without constant.
logLik(LMM1, includeConstant = FALSE)
```

`mLogLik` *Function to obtain restricted log-likelihood and the first derivatives of the log-likelihood, given values for the penalty parameters*

Description

Function to obtain restricted log-likelihood and the first derivatives of the log-likelihood, given values for the penalty parameters

Usage

```
mLogLik(object, theta)
```

Arguments

`object`	an object of class LMMsolve
`theta`	a matrix with values of precision parameters theta.

Value

A data.frame with logL and the first derivatives of log-likelihood

---

`multinomial` *Family Object for Multinomial Model*

Description

The Multinomial model is not part of the standard family. The implementation is based on Chapter 6 in Fahrmeir et al. (2013).

Usage

```
multinomial()
```

Value

An object of class `familyLMMsolver` with the following components:

`family`	character string with the family name.
`linkfun`	the link function.
`linkinv`	the inverse of the link function.
`dev.resids`	function giving the deviance for each observation as a function of (y, mu, wt)

References

Fahrmeir, Ludwig, Thomas Kneib, Stefan Lang, Brian Marx, Regression models. Springer Berlin Heidelberg, 2013.

`multipop` *Simulated QTL mapping data set*

Description

Simulated QTL mapping data set

Usage

```
multipop
```

Format

A data.frame with 180 rows and 6 columns.

cross Cross ID, two populations, AxB and AxC

ind Genotype ID

pA Probability that individual has alleles from parent A

pB Probability that individual has alleles from parent B

pC Probability that individual has alleles from parent C

pheno Simulated phenotypic value

`oats.data` *Alpha lattice design of spring oats*

Description

Alpha lattice design of spring oats

Usage

```
oats.data
```

Format

A data.frame with 72 rows and 7 columns

plot plot number

rep replicate

block incomplete block

gen genotype

yield dry matter yield

row row

col column

Details

The response is grain yield in kg per hectare. The design was an alpha design with 24 varieties, three replicates and six incomplete blocks of size four per replicate. The 72 plots were arranged in a single linear array.

Source

J. A. John & E. R. Williams (1995). Cyclic and computer generated designs. Chapman and Hall, London. Page 146.

References

Boer, Martin P., Hans-Peter Piepho, and Emlyn R. Williams. "Linear variance, P-splines and neighbour differences for spatial adjustment in field trials: how are they related?." JABES 25, no. 4 (2020): 676-698.

`obtainSmoothTrend` *Obtain Smooth Trend.*

Description

Obtain the smooth trend for models fitted with a spline component.

Usage

```
obtainSmoothTrend(
  object,
  grid = NULL,
  newdata = NULL,
  deriv = 0,
  includeIntercept = FALSE,
  which = 1
)
```

Arguments

`object`	An object of class LMMsolve.
`grid`	A numeric vector having the length of the dimension of the fitted spline component. This represents the number of grid points at which a surface will be computed.
`newdata`	A data.frame containing new points for which the smooth trend should be computed. Column names should include the names used when fitting the spline model.
`deriv`	Derivative of B-splines, default 0. At the moment only implemented for spl1D.
`includeIntercept`	Should the value of the intercept be included in the computed smooth trend? Ignored if deriv > 0.
`which`	An integer, for if there are multiple splxD terms in the model. Default value is 1.

Value

A data.frame with predictions for the smooth trend on the specified grid. The standard errors are saved if ‘deriv‘ has default value 0.

Examples

```
## Fit model on oats data
data(oats.data)

## Fit a model with a 1-dimensional spline at the plot level.
LMM1_spline <- LMMsolve(fixed = yield ~ rep + gen,
```

```
                         spline = ~spl1D(x = plot, nseg = 20),
                         data = oats.data)

## Obtain the smooth trend for the fitted model on a dense grid.
smooth1 <- obtainSmoothTrend(LMM1_spline,
                             grid = 100)

## Obtain the smooth trend on a new data set - plots 10 to 40.
newdat <- data.frame(plot = 10:40)
smooth2 <- obtainSmoothTrend(LMM1_spline,
                             newdata = newdat)

## The first derivative of the smooth trend can be obtained by setting deriv = 1.
smooth3 <- obtainSmoothTrend(LMM1_spline,
                             grid = 100,
                             deriv = 1)

## For examples of higher order splines see the vignette.
```

`predict.LMMsolve` *Predict function*

Description

Predict function

Usage

```
## S3 method for class 'LMMsolve'
predict(
  object,
  newdata,
  type = c("response", "link"),
  se.fit = FALSE,
  deriv = NULL,
  ...
)
```

Arguments

`object`	an object of class LMMsolve.
`newdata`	A data.frame containing new points for which the smooth trend should be computed. Column names should include the names used when fitting the spline model.
`type`	When this has the value "link" the linear predictor fitted values or predictions (possibly with associated standard errors) are returned. When type = "response" (default) fitted values or predictions on the scale of the response are returned (possibly with associated standard errors).

`se.fit`	calculate standard errors, default `FALSE`.
`deriv`	Character string of variable for which to calculate the first derivative; default `NULL`.
`...`	other arguments. Not yet implemented.

Value

A data.frame with predictions for the smooth trend on the specified grid. The standard errors are saved if ‘se.fit=TRUE‘.

Examples

```
## simulate some data
f <- function(x) { 0.3 + 0.4*x + 0.2*sin(20*x) }
set.seed(12)
n <- 150
x <- seq(0, 1, length = n)
sigma2e <- 0.04
y <- f(x) + rnorm(n, sd = sqrt(sigma2e))
dat <- data.frame(x, y)

## fit the model
obj <- LMMsolve(fixed = y ~ 1,
         spline = ~spl1D(x, nseg = 50), data = dat)

## make predictions
newdat <- data.frame(x = seq(0, 1, length = 5))
pred <- predict(obj, newdata = newdat, se.fit = TRUE)
pred

## make predictions for derivative of x:
pred2 <- predict(obj, newdata = newdat, se.fit = TRUE, deriv = "x")
pred2
```

`residuals.LMMsolve` *Residuals of an LMMsolve object.*

Description

Obtain the residuals from a mixed model fitted using LMMSolve.

Usage

```
## S3 method for class 'LMMsolve'
residuals(object, ...)
```

Arguments

`object`	an object of class LMMsolve
`...`	some methods for this generic require additional arguments. None are used in this method.

Value

A vector of residuals.

Examples

```
## Fit model on oats.data
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                 data = oats.data)

## Obtain fitted values.
residuals1 <- residuals(LMM1)
```

`SeaSurfaceTemp` *Sea Surface Temperature*

Description

Sea Surface Temperature

Usage

```
SeaSurfaceTemp
```

Format

A data.frame with 15607 rows and 4 columns.

lon longitude

lat latitude

sst sea surface temperature in Kelvin

type defines training and test set

References

Cressie et al. (2022) Basis-function models in spatial statistics. Annual Review of Statistics and Its Application. doi:10.1146/annurevstatistics040120020733

`spl1D` *Fit P-splines*

Description

Fit multi dimensional P-splines using sparse implementation.

Usage

```
spl1D(
  x,
  nseg,
  pord = 2,
  degree = 3,
  cyclic = FALSE,
  scaleX = TRUE,
  xlim = range(x),
  cond = NULL,
  level = NULL
)

spl2D(
  x1,
  x2,
  nseg,
  pord = 2,
  degree = 3,
  cyclic = c(FALSE, FALSE),
  scaleX = TRUE,
  x1lim = range(x1),
  x2lim = range(x2),
  cond = NULL,
  level = NULL
)

spl3D(
  x1,
  x2,
  x3,
  nseg,
  pord = 2,
  degree = 3,
  scaleX = TRUE,
  x1lim = range(x1),
  x2lim = range(x2),
  x3lim = range(x3)
)
```

Arguments

`x`, `x1`, `x2`, `x3`	The variables in the data containing the values of the `x` covariates.
`nseg`	The number of segments
`pord`	The order of penalty, default `pord = 2`
`degree`	The degree of B-spline basis, default `degree = 3`
`cyclic`	Cyclic or linear B-splines; default `cyclic=FALSE`
`scaleX`	Should the fixed effects be scaled.
`xlim`, `x1lim`, `x2lim`, `x3lim`	A numerical vector of length 2 containing the domain of the corresponding x covariate where the knots should be placed. Default set to `NULL`, when the covariate range will be used.
`cond`	Conditional factor: splines are defined conditional on the level. Default `NULL`.
`level`	The level of the conditional factor. Default `NULL`.

Value

A list with the following elements:

- `X` - design matrix for fixed effect. The intercept is not included.
- `Z` - design matrix for random effect.
- `lGinv` - a list of precision matrices
- `knots` - a list of vectors with knot positions
- `dim.f` - the dimensions of the fixed effect.
- `dim.r` - the dimensions of the random effect.
- `term.labels.f` - the labels for the fixed effect terms.
- `term.labels.r` - the labels for the random effect terms.
- `x` - a list of vectors for the spline variables.
- `pord` - the order of the penalty.
- `degree` - the degree of the B-spline basis.
- `scaleX` - logical indicating if the fixed effects are scaled.
- `EDnom` - the nominal effective dimensions.

Functions

- `spl2D()`: 2-dimensional splines
- `spl3D()`: 3-dimensional splines

See Also

`LMMsolve`

Examples

```
## Fit model on oats data
data(oats.data)

## Fit a model with a 1-dimensional spline at the plot level.
LMM1_spline <- LMMsolve(fixed = yield ~ rep + gen,
                        spline = ~spl1D(x = plot, nseg = 20),
                        data = oats.data)

summary(LMM1_spline)

## Fit model on US precipitation data from spam package.
data(USprecip, package = "spam")

## Only use observed data
USprecip <- as.data.frame(USprecip)
USprecip <- USprecip[USprecip$infill == 1, ]

## Fit a model with a 2-dimensional P-spline.
LMM2_spline <- LMMsolve(fixed = anomaly ~ 1,
                        spline = ~spl2D(x1 = lon, x2 = lat, nseg = c(41, 41)),
                        data = USprecip)

summary(LMM2_spline)
```

`summary.LMMsolve` *Summarize Linear Mixed Model fits*

Description

Summary method for class "LMMsolve". Creates either a table of effective dimensions (which = "dimensions") or a table of variances (which = "variances").

Usage

```
## S3 method for class 'LMMsolve'
summary(object, which = c("dimensions", "variances"), ...)

## S3 method for class 'summary.LMMsolve'
print(x, ...)
```

Arguments

`object`	An object of class LMMsolve
`which`	A character string indicating which summary table should be created.
`...`	Some methods for this generic require additional arguments. None are used in this method.
`x`	An object of class summary.LMMsolve, the result of a call to summary.LMM

Value

A data.frame with either effective dimensions or variances depending on which.

Methods (by generic)

- `print(summary.LMMsolve)`: print summary

Examples

```
## Fit model on oats data.
data(oats.data)

## Fit simple model with only fixed effects.
LMM1 <- LMMsolve(fixed = yield ~ rep + gen,
                data = oats.data)

## Obtain table of effective dimensions.
summ1 <- summary(LMM1)
print(summ1)

## Obtain table of variances.
summ2 <- summary(LMM1,
                which = "variances")
print(summ2)
```

Index