Package ‘mvmeta’

February 20, 2015

Type Package
Title Multivariate and Univariate Meta-Analysis and Meta-Regression
Version 0.4.5
Date 2014-08-10
Author Antonio Gasparini
Maintainer Antonio Gasparini <antonio.gasparrini@lshtm.ac.uk>
Depends R (>= 2.13.0)
Imports stats, graphics, grDevices, utils
Suggests metafor, meta, rmeta, nlme, MASS, dlnm
Description The package mvmeta consists of a collection of functions to perform fixed and random-effects multivariate and univariate meta-analysis and meta-regression in R.
URL http://www.ag-myresearch.com/package-mvmeta
License GPL (>= 2)
LazyData yes
NeedsCompilation no
Repository CRAN
Date/Publication 2014-08-09 07:01:08

R topics documented:

  mvmeta-package ......................................................... 2
  berkey98 .......................................................... 6
  blup.mvmeta ......................................................... 7
  coef.mvmeta ........................................................ 9
  fibrinogen ......................................................... 11
  hsls ............................................................... 12
  hyp ............................................................... 13
  inputcov .......................................................... 14
  inputna ............................................................ 16
  logLik.mvmeta ..................................................... 18
Description

The package `mvmeta` consists of a collection of functions to perform fixed and random-effects multivariate and univariate meta-analysis and meta-regression in R.

Modelling framework

Multivariate meta-analytical models represent an extension of the standard univariate techniques, where estimates of a single effect size, here defined generally as outcome, are pooled across studies. In multivariate meta-analysis, estimates of multiple outcomes are combined while accounting for their correlation. Multivariate meta-regression also models such multivariate distribution in terms of study-level predictors. These statistical tools were originally proposed to model multiple endpoints in clinical trials. Applications and methodological developments are currently proposed also for network meta-analysis (indirect treatment comparison), for the meta-analysis of multi-parameter associations and for meta-analysis of diagnostic studies, among others. See references below for details.

Similarly to univariate methods, fixed-effects models do not assume heterogeneity among studies, and the estimates are conditional on the set of studies collected in the meta-analysis, based on a multivariate weighted average of study-level estimates. Random-effects meta-analysis, instead, allows a degree of heterogeneity among studies, assuming the (true but unobserved) study-specific outcomes as randomly sampled from a multivariate normal distribution of studies. Inference from these models may therefore be extended to other unmeasured studies assumed to belong to the same (usually hypothetical) population.
Estimation and interpretation exploit here the framework of linear mixed models. The fixed part of the model provides an estimate of fixed effects, which represent the population-averaged outcomes and, in the case of meta-regression, are defined by a set of coefficients associated to study-level predictors. The random part of the model describes the deviation from the population averages, estimating the components of a between-study (co)variance matrix. Assuming \( k \) outcomes \( y_i \) estimated in each of \( i = 1, \ldots, m \) studies, and related to \( p \) study-level predictors \( x_i \), random-effects multivariate meta-regression models can be generally described with:

\[
y_i \sim N_k(X_i \beta, S_i + \Psi)
\]

Here the outcomes \( y_i \) are assumed to be sampled from a multivariate normal distribution of order \( k \). Their distribution is centred on \( X_i \beta \), with \( X_i \) as a \( k \times kp \) design matrix and \( \beta \) the vector of fixed-effects coefficients. The marginal \( k \times k \) (co)variance matrix \( \Sigma_i = S_i + \Psi \) is given by the sum of within (assumed known) and between-study (co)variance matrices \( S_i \) and \( \Psi \), respectively.

Other models are taken as special cases of that above. In multivariate meta-analysis, \( X_i \) becomes an identity matrix with \( p = 1 \), and \( \beta \) reduces to \( k \) intercepts, interpreted as the population-averaged outcomes. For \( k = 1 \), the model reduces to the standard univariate meta-analysis or meta-regression. In fixed-effects meta-analytic models, \( \Psi \) is assumed not to exist, and the variability between studies is due exclusively to the within-study estimation error.

**Estimation methods**

The aim is to estimate the coefficients \( \beta \) and, for random-effects models, the components of the between-study (co)variance matrix \( \Psi \). If the same linear predictor of \( p \) terms \( x_i \) measured in each study is specified for all the outcomes (the only option available in the current version), the dimension of \( \beta \) is \( kp \). The parameters for the random part depend on the chosen structure of the between-study (co)variance matrix \( \Psi \), with \( k(k+1)/2 \) parameters for an unstructured form, and a smaller set of parameters for structured (co)variances.

Different estimator are implemented in the package mvmeta. The estimation options available in the current version are:

- **Fixed-effects**
- **Maximum likelihood** (ML)
- **Restricted maximum likelihood** (REML)
- **Method of moments**
- **Variance components**

The fixed-effects model is fitted through generalized least squares (GLS), assuming the (co)variance structure, composed by the within-study error only, as completely known. Among random-effects models, ML and REML approaches provides fit criteria and inferential test derived from likelihood theory, such as AIC and likelihood ratio test. However, they are based on computationally intensive iterative procedures of optimization, and convergence can be slow for high-dimensional models (with a high number of outcomes). Estimators based on semiparametric alternatives such as the non-iterative method of moments or the iterative variance components approach are generally faster. Further details on estimation methods are given in the related help pages.
Functions and data included in the package

The main function in the package is `mvmeta`, which performs the various models illustrated above. This function resembles standard regression functions in R, and specifies the model through a regression formula. The function returns a list object of class "mvmeta" (see `mvmetaObject`).

The estimation is carried out internally through `mvmeta.fit`, a wrapper which prepares the data and calls specific estimation functions for fitting the models. Specifically, `mvmeta.fixed` is applied for fixed-effects models, while estimators for random-effects models are implemented in the functions `mvmeta.ml` and `mvmeta.reml` for (restricted) maximum likelihood, `mvmeta.mm` for the method of moments, and `mvmeta.vc` for variance components. For likelihood-based methods, iterative optimizations algorithms are used for maximizing the (restricted) likelihood, and specific (co)variance structures for the between-study random effects are available. Fitting parameter options are set by `mvmeta.control`.

Method functions are available for objects of class "mvmeta" (see `mvmetaObject` for a complete list). The method `summary` produces a list of class "summary.mvmeta" for summarizing the fit of the model and providing additional results. The method function `predict` computes predicted values, optionally for a set of new values of the predictors. `blup` gives the (empirical) best linear unbiased predictions for the set of studies used for estimation. Other default or specific method functions for regression can be used on objects of class "mvmeta", such as `fitted` and `residuals`, `logLik`, `AIC` and `BIC`, among others.

Methods for `model.frame` and `model.matrix` are used to extract and construct the model frame and the design matrix of the regression meta-analytical model, respectively. Methods for `na.omit` and `na.exclude` help handle correctly missing values.

Simulations can be produced using the function `mvmetaSim` and the method function `simulate`, which return one or multiple sets of simulated outcomes for a group of studies. The function `inputna` and `inputcov` are used internally to augment the missing data values and to input missing correlations, respectively.

The method function `qtest.mvmeta` (producing an object with class of the same name) performs the (multivariate) Cochran Q test for (residual) heterogeneity, both on the overall multivariate distribution and on each single outcome. The generic method function is `qtest`.

Printing functions for the objects of classes defined above are also provided. Other functions are used internally in the source code, and not exported in the namespace. For users interested in getting into details of the package structure, these functions can be displayed using the triple colon (":::" ) operator. For instance, `mvmeta::glsfit` displays the code of the function `glsfit`. Also, some comments are added in the original source code.

The package includes the datasets `berkeyYX`, `fibrinogen`, `hsls`, `hyp`, `p53` and `smoking` as data frames, which are used in the examples.

Additional information

A list of changes included in the current and previous versions can be found by typing:

```
file.show(system.file("ChangeLog",package="mvmeta"))
```

General information on the development and applications of the `mvmeta` package and on the modelling framework of multivariate meta-analysis, together with an updated version of the R scripts for running the examples in published papers, can be found at [www.ag-myresearch.com](http://www.ag-myresearch.com).
Warnings

This release of the package mvmeta has been tested with different simulated and real datasets. The functions generally perform well under several scenarios, and comparisons with alternative software implementations show good agreement. However, bugs and bad performance under untested conditions may not be excluded. Please report any error or unexpected behaviour to the e-mail address below.

Note

The functions included in the package mvmeta may be applied also to perform standard univariate meta-analysis and meta-regression. However, alternative packages provides a more exhaustive and efficient set of functions. See for example the packages metafor, meta, rmeta and metaLik, among others.

Functions for fitting multivariate models are available in other R packages. The package metafor provides functions to perform fixed or random-effects multivariate models, the latter fitted through maximum or restricted maximum likelihood. A thorough comparison with the implementation offered in mvmeta is not available yet. The package metaSEM (not in CRAN) offers functions to perform fixed-effects and likelihood-based random-effects multivariate meta-analysis through structural equation modelling. The package mvmeta performs fixed-effects and random-effects multivariate meta-analysis using the same method of moments estimator adopted here, although without allowing for missing outcomes or meta-regression. The package mada contains functions for bivariate models used in diagnostic studies.

Use citation("mvmeta") to cite this package.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References


---

**berkey98**

*Five Published Trials on Periodontal Disease*

**Description**

The dataset contains the results of 5 published trials comparing surgical and non-surgical treatments for medium-severity periodontal disease, one year after treatment. The 2 estimated outcomes are average improvements (surgical minus non-surgical, in mm) in probing depth (PD) and attachment level (AL).

**Usage**

berkey98

**Format**

A data frame with 5 observations on the following 7 variables:

- `pubyear` publication year of the trial.
- `npat` number of patients included in the trial.
- `PD` estimated improvement of surgical versus non-surgical treatments in probing depth (mm).
- `AL` estimated improvement of surgical versus non-surgical treatments in attachment level (mm).
- `var_PD` variance of the estimated outcome for PD.
- `cov_PD_AL` variance of the estimated outcome for AL.
- `var_AL` covariance of the estimated outcomes for PD and AL.

Row names specify the author of the paper reporting the results of each trial.
Source


Examples

```r
### REPRODUCE THE RESULTS IN BERKEY ET AL. (1998)

# INSPECT THE DATA
berkey98

# FIXED-EFFECTS
year <- berkey98$pubyear - 1983
model <- mvmeta(cbind(PD, AL) ~ year, S = berkey98[5:7], data = berkey98, method = "fixed")
print(summary(model), digits = 3)

# GLS MODEL (VARIANCE COMPONENTS)
model <- mvmeta(cbind(PD, AL) ~ year, S = berkey98[5:7], data = berkey98, method = "vc",
               control = list(vc.adj = FALSE))
print(summary(model), digits = 3)
round(model$Psi, 3)

# ML MODEL
model <- mvmeta(cbind(PD, AL) ~ year, S = berkey98[5:7], data = berkey98, method = "ml")
print(summary(model), digits = 3)
round(model$Psi, 3)
```

blup.mvmeta

Best Linear Unbiased Predictions from mvmeta Models

Description

This method function computes (empirical) best linear unbiased predictions from fitted univariate or multivariate meta-analytical models represented in objects of class "mvmeta". Such predictions are optionally accompanied by standard errors, prediction intervals or the entire (co)variance matrix of the predicted outcomes.

Usage

```r
## S3 method for class 'mvmeta'
blup(object, se = FALSE, pi = FALSE, vcov = FALSE, pi.level = 0.95,
     format = c("matrix", "list"), aggregate = c("stat", "y"), ...)
```
Arguments

- **object**: an object of class "mvmeta".
- **se**: logical switch indicating if standard errors must be included.
- **pi**: logical switch indicating if prediction intervals must be included.
- **vcov**: logical switch indicating if the (co)variance matrix must be included.
- **pi.level**: a numerical value between 0 and 1, specifying the confidence level for the computation of prediction intervals.
- **format**: the format for the returned results. See Value.
- **aggregate**: when format="matrix" and se or pi are required, the results may be aggregated by statistic or by outcome. See Value.
- ... further arguments passed to or from other methods.

Details

The method function blup produces (empirical) best linear unbiased predictions from mvmeta objects. For random-effects models, predictions are given by the sum of the estimated mean outcomes from the fixed part of the model, plus study-specific deviations predicted as random effects given the between-study distribution.

Predicted outcomes from blup are a shrunk version of study-specific realizations, where study-specific estimates borrow strength from the assumption of an underlying multivariate distribution of outcomes in a (usually hypothetical) population of studies. In practice, the results from blup represent a weighted average between population mean outcomes (estimated by the fixed part of the model) and study-specific estimates. The weights depend from the relative size of the within and between-study covariance matrices reported as components S and Psi in mvmeta objects (see mvmetaObject).

Fixed-effects models do not assume study-specific random effects, and the results of blup for these models are identical to predict with interval="confidence".

How to handle predictions for studies removed from estimation due to invalid missing pattern is determined by the na.action argument used in mvmeta to produce object. If na.action=na.omit, studies excluded from estimation will not appear, whereas if na.action=na.exclude they will appear, with values set to NA for all the outcomes. This step is performed by napredict. See Note below.

In the presence of missing values in the study-specific estimated outcome y of the fitted model, correspondent values of point estimates and covariance terms are set to 0, while the variance terms are set to 1e+10. In this case, in practice, the study-specific estimates do not provide any information (their weight is virtually 0), and the prediction tends to the value returned by predict with interval="prediction", when applied to a new but identical set of predictors. See also Note below.

Value

The results may be aggregated in matrices (the default), or returned as lists, depending on the argument format. For multivariate models, the aggregation is ruled by the argument aggregate, and the results may be grouped by statistic or by outcome. If vcov=TRUE, lists are always returned.
**Note**

The definition of missing in model frames used for estimation in `mvmeta` is different than that commonly adopted in other regression models such as `lm` or `glm`. See info on missing values in `mvmeta`.

Differently from `predict`, this method function computes the predicted values in the presence of partially missing outcomes. Interestingly, BLUPs for missing outcomes may be slightly different than predictions returned by `predict` on a new but identical set of predictors, as the BLUP also depends on the random part of the model. Specifically, the function uses information from the between-study covariance to predict missing outcomes given the observed ones.

**Author(s)**

Antonio Gasparini, <antonio.gasparrini@lshtm.ac.uk>

**References**


**See Also**

See `predict` for standard predictions. See `mvmeta-package` for an overview of the package and modelling framework.

**Examples**

```r
# RUN THE MODEL
model <- mvmeta(cbind(PD,AL)~1,S=berkey98[5:7],data=berkey98)

# ONLY BLUP
blup(model)

# BLUP AND SE
blup(model,se=TRUE)

# SAME AS ABOVE, AGGREGATED BY OUTCOME, WITH PREDICTION INTERVALS
blup(model,se=TRUE,pi=TRUE,aggregate="y")

# WITH VCOV, FORCED TO A LIST
blup(model,se=TRUE,pi=TRUE,vcov=TRUE,aggregate="y")
```

---

**coef.mvmeta**

`Extract Coefficients and (Co)Variance Matrix from mvmeta Objects`

**Description**

These method functions return the estimated fixed-effects coefficients and their (co)variance matrix for fitted univariate or multivariate meta-analytical models represented in objects of class "mvmeta".
## coef.mvmeta

### Usage

```r
## S3 method for class 'mvmeta'
coef(object, format=c("vector","matrix"), ...)

## S3 method for class 'mvmeta'
vcov(object, ...)
```

### Arguments

- `object`: an object of class "mvmeta".
- `format`: format of the returned object.
- `...`: further arguments passed to or from other methods.

### Value

For `coef`, a vector (default) or matrix with the estimated fixed-effects coefficients. The matrix-structure is used to store the fixed-effects coefficients in `mvmeta` objects, and is preserved if `format="matrix`.

For `vcov`, the (co)variance matrix of the estimated fixed-effects coefficients.

### Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

### See Also

See `mvmeta-package` for an overview of the package and modelling framework.

### Examples

```r
# RUN THE MODEL
model <- mvmeta(cbind(PD,AL)~pubyear,S=berkey98[5:7],data=berkey98)

# COEFFICIENTS
model$coef
coef(model)
coef(model, format="matrix")
summary(model)$coef

# (CO)VARIANCE MATRIX
vcov(model)
```
Description

The Fibrinogen Studies Collaboration is a meta-analysis of individual data on 154,012 adults from 31 prospective cohort studies with information on plasma fibrinogen and major disease outcomes. The dataset reports a subset of the results of a first-stage analysis consisting of the log-hazard ratio of coronary heart disease for categories of levels of fibrinogen versus a baseline category.

Usage

fibrinogen

Format

A data frame with 31 observations on the following 15 variables:

- cohort: study ID.
- b2, b3, b4, b5: estimated log-hazard ratios for the second to fifth categories versus the baseline category.
- V_2_2, V_3_3, V_4_4, V_5_5: variances of the estimated log-hazard ratios.
- V_2_3, V_2_4, V_2_5, V_3_4, V_3_5, V_4_5: covariances of the estimated log-hazard ratios.

Details

The published analysis adopted a fixed-effects model on 10 categories of fibrinogen (Fibrinogen Studies Collaboration 2004, 2005). Here a subset of the results of the first-stage analysis is reported, namely the log-hazard ratio for 4 categories and associated (co)variance terms, ordered as the lower triangular elements of the (co)variance matrix taken by column. Details on the first-stage model and the second-stage meta-analysis are provided in White (2009) and Jackson and colleagues (2010).

Note

The data provide an example of application of multivariate meta-analysis for multi-parameter association, where a relationship is defined by functions specified by several coefficients. In this case, the coefficients refer to log-hazard ratio for strata of the original variable versus a baseline category. A general overview of the application of multivariate meta-analysis in this setting is provided by Gasparrini and colleagues (2012).

Source


**Examples**

```r

# INSPECT THE DATA
head(fibrinogen)

# REML MODEL
y <- as.matrix(fibrinogen[2:5])
S <- as.matrix(fibrinogen[6:15])
model <- mvmeta(y,S)

# SUMMARIZE THE RESULTS
print(summary(model), digits=3)
round(model$Psi,3)
```

---

**hls**

*High School Longitudinal Study*

**Description**

This is a nationally representative, longitudinal study of more than 21,000 9th graders in 944 schools who will be followed through their secondary and postsecondary years. The data are used for testing whether sex, socioeconomic status and sex by socio-economic status interaction are predictive of the mathematics standardized score in each of the eight race groups.

**Usage**

```
hls
```

**Format**

A data frame with 8 observations on the following 10 variables:

- **race**: race group.
- **b1**, **b2**, **b3**: estimated regression coefficients for sex, socio-economic status and sex by socio-economic status interaction, respectively, on the mathematics standardized score.
- **V11**, **V22**, **V33**: variances of the estimated coefficients.
- **V12**, **V13**, **V23**: covariances of the estimated coefficients.
### Description

The dataset contains the results of ten studies that assess the effectiveness of hypertension treatment for lowering blood pressure. Each study provides complete data on two treatment effects, the difference in systolic blood pressure (SBP) and diastolic blood pressure (DBP) between the treatment and the control groups, where these differences are adjusted for the participants' baseline blood pressures. The within-study correlations of the two outcomes are known. Some trials are conducted on patients with isolated systolic hypertension (ISH).

### Usage

`hyp`

### Format

A data frame with 10 observations on the following 7 variables:

- `study`: study ID.
- `sbp`, `sbp_se`: estimated difference and its standard error in systolic blood pressure.
- `dbp`, `dbp_se`: estimated difference and its standard error in diastolic blood pressure.
- `rho`: within-study correlation between the estimated differences in systolic and diastolic blood pressure.
- `ish`: indicator for studies on patients with isolated systolic hypertension.
Note

The standard errors for the two outcomes are wrongly reported as variances in the original article by Jackson and colleagues (2013).

Source


Examples

```r
### REPRODUCE THE RESULTS IN JACKSON ET AL. (2013)

# INSPECT THE DATA
hyp

# INPUT THE CORRELATION (CAN ALSO BE INPUTTED DIRECTLY, SEE BELOW)
(S <- inputcov(hyp[c("sbp_se","dbp_se"),,],cor=hyp$rho))
# CHECK WITH THE FIRST STUDY
cov2cor(xpndMat(S[1,1]))

# META-ANALYSIS, REML MODEL
model <- mvmeta(cbind(sbp,dbp),S=S,data=hyp)
print(summary(model),digits=2)
round(model$Psi,2)

# META-ANALYSIS, REML MODEL (INPUTTING THE CORRELATION DIRECTLY)
model <- mvmeta(cbind(sbp,dbp),S=cbind(sbp_se,dbp_se)^2,data=hyp,
control=list(Scor=hyp$rho))
print(summary(model),digits=2)

# META-ANALYSIS, MM MODEL
model <- mvmeta(cbind(sbp,dbp),S=S,data=hyp,method="mm")
print(summary(model),digits=2)
round(model$Psi,2)

# META-REGRESSION, REML MODEL
model <- mvmeta(cbind(sbp,dbp)-ish,S=S,data=hyp)
print(summary(model),digits=2)

# META-REGRESSION, MM MODEL
model <- mvmeta(cbind(sbp,dbp)-ish,S=S,data=hyp,method="mm")
print(summary(model),digits=2)
```

inputcov

Input (Co)Variance Matrices
**inputcov**

**Description**

This function inputs (co)variance matrices of a set of outcomes given the corresponding standard deviation and correlation values.

**Usage**

`inputcov(sd, cor=0)`

**Arguments**

- `sd` a \( m \times k \) matrix of standard deviations for \( k \) outcomes in \( m \) matrices, or a vector for \( k \) outcomes in a single matrix.
- `cor` either a vector of length 1, \( m \) or \( k(k - 1)/2 \), or alternatively a \( k \times k \) or \( m \times k \) lower triangular matrix of \( k \) outcomes, the same for all the \( m \) matrices.

**Details**

Depending the number of outcomes \( k \) and matrices \( m \), the argument `cor` is interpreted as:

- if a vector of length 1 (a scalar), the same correlation for all the \( k \) outcomes for all the \( m \) matrices;
- if a vector of length \( m \), the same correlation for all the \( k \) outcomes for each of the \( m \) matrices;
- if a vector of length \( k(k - 1)/2 \), the lower triangular elements (without diagonal, taken by column) of the correlation matrix of the \( k \) outcomes, the same for all the \( m \) matrices;
- if a \( k \times k \) matrix, the correlation matrix for the single matrix (only when \( m=1 \));
- if a \( m \times k(k - 1)/2 \) matrix, each row represents the lower triangular elements (without diagonal, taken by column) of the correlation matrix of the \( k \) outcomes for each of the \( m \) matrices.

**Value**

For a single matrix, the (co)variance matrix itself. For multiple matrices, a \( m \times k(k + 1)/2 \) matrix, where each row represents the vectorized entries of the lower triangle (with diagonal, taken by column) of the related (co)variance matrix (see `vechMat`).

**Note**

This function is called internally by `mvmeta` to input the correlation(s) when only the within-study variances are provided through the argument `S`. In this case, the correlation values are set through the argument `Scor` in the control list (see `mvmeta.control`).

**Author(s)**

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

**See Also**

See `xpndMat`. See `mvmeta.control`. 
Examples

# SOME RANDOM SD FOR A SINGLE MATRIX, WITH CONSTANT CORRELATION
(M <- inputcov(runif(4,0.1,3),0.7))
# CHECK CORRELATION
cov2cor(M)

# NOW WITH A MORE COMPLEX CORRELATION STRUCTURE
(M <- inputcov(runif(3,0.1,3),c(0.7,0.2,0.4)))
cov2cor(M)

# MULTIPLE MATRICES
(V <- matrix(runif(5*3,0.1,3),5,3,dimnames=list(1:5,paste("V",1:3,sep""))))
inputcov(V,0.6)

# WITH REAL DATA WHEN CORRELATIONS AVAILABLE
hyp
(S <- inputcov(hyp[c("sbp_se","dbp_se"),],cor=hyp$rho))
# CHECK FIRST STUDY
cov2cor(xpndMat(S[1,]))

# USED INTERNALLY IN mvmeta
p53
inputcov(sqrt(p53[c("V1","V2"])),0.5)
model <- mvmeta(cbind(y1,y2),S=cbind(V1,V2),data=p53,control=list(Scor=0.5))
model$S

---

inputna  Input Missing Values

Description

This function augment data by replacing missing values. It can be used internally in mvmeta through the control list.

Usage

inputna(y, S, inputvar=10^4)

Arguments

Assuming a meta-analysis or meta-regression based on m studies and k outcomes:

a m-dimensional vector (for univariate models) or m x k matrix (for multivariate models) of outcomes.

§ series of within-study (co)variance matrices of the estimated outcomes for each one of the m studies. Accepted formats by inputna are: a m-dimensional list of k x k matrices; a tri-dimensional k x k x m array; a matrix or data frame with m rows and k(k+1)/2 or k columns, depending on the availability of the within-study correlations.
**inputna**

**inputvar** multiplier for inputting the missing variances in S.

**Details**

The function augments the data by replacing missing values in the outcomes and the associated (co)variance matrices. Specifically, it replaces missing outcomes and missing covariances (if provided) with 0, and missing variances with the largest observed variance multiplied by `inputvar`. This value is expected to be very high, by default $10^4$, so that the corresponding observation contributes only negligibly to the final estimate.

**Value**

A matrix with the first $k$ column corresponding to the augmented outcomes, and the remaining $k(k+1)/2$ or $k$ columns (depending on the availability of the within-study covariances) corresponding to vectorized entries of the lower triangle of the related (co)variance matrices.

**Note**

Data augmentation used to be the approach to deal with missing values in the first implementation of multivariate meta-analysis. The current algorithms implemented in `mvmeta` directly account for missing.

Inputting missing values can be useful when two or more outcomes are never observed jointly, and the estimation is entirely based on indirect comparison. This method can be applied in network meta-analysis, also called indirect treatment comparison.

This approach can produce different results than standard methods, especially when the occurrence of missing is substantial. Preliminary analyses indicate that likelihood-based estimation methods do not seem to be affected, while non-iterative estimators such as method of moments and variance components are more sensitive. The user should be careful on the application of missing augmentation.

**Author(s)**

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

**References**


**See Also**

See `inputcov` for inputting (co)variance matrices.
Examples

# INSPECT THE DATA
head(smoking)

# STANDARD APPROACH TO MISSING DATA
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S)
summary(model)

# WITH DATA AUGMENTATION
augdata <- inputna(y,S)
y <- augdata[,1:3]
S <- augdata[,,-c(1:3)]
model <- mvmeta(y,S)
summary(model)
# NB: SAME PARAMETER ESTIMATES, BUT WRONG NUMBER OF OBS

# USED INTERNALLY IN mvmeta
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S,control=list(inputna=TRUE))
summary(model)
# NOW RIGHT NUMBER OF OBS

logLik.mvmeta

Extract Log-Likelihood from mvmeta Objects

Description
This method function returns the log-likelihood for fitted univariate or multivariate meta-analytical models represented in objects of class "mvmeta".

Usage

## S3 method for class 'mvmeta'
logLik(object, ...)

Arguments

object an object of class "mvmeta".
...

Value
A numeric scalar of class "logLik" with attributes, providing the (restricted) log likelihood of the model. Attributes correspond to the component df of mvmeta objects, namely the following scalars: nall (number of observations used for estimation, excluding missing values), nob (equal to nall, minus the number of fixed-effects coefficients for REML models, fixed (number of estimated fixed-effects coefficients), random (number of estimated (co)variance terms).
Note
This function is called by AIC and BIC for computing the Akaike and Bayesian information criteria.

Author(s)
Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

See Also
See the default method logLik. See mmeta-package for an overview of the package and modelling framework.

Examples
# RUN THE MODEL
model <- mvmeta(cbind(PD, AL) - pubyear, S = berkey98[5:7], data = berkey98)

# LOG-LIKELIHOOD
ll <- logLik(model)
ll
attributes(ll)

# AIC and BIC
AIC(model)
BIC(model)

mlprof.fn

Likelihood Functions for mmeta Models

Description
These functions compute the value of the log-likelihood and the related vectors of first partial derivatives for random-effects multivariate and univariate meta-analysis and meta-regression, in terms of model parameters. They are meant to be used internally and not directly run by the users.

Usage
mlprof.fn(par, Xlist, ylist, Slist, nalist, k, m, p, nall, bscov, ctrl)
mlprof.gr(par, Xlist, ylist, Slist, nalist, k, m, p, nall, bscov, ctrl)
remlprof.fn(par, Xlist, ylist, Slist, nalist, k, m, p, nall, bscov, ctrl)
remlprof.gr(par, Xlist, ylist, Slist, nalist, k, m, p, nall, bscov, ctrl)
iter.igls(Psi, Xlist, ylist, Slist, nalist, k, m)
Arguments

Assuming a meta-analysis or meta-regression based on \( m \) studies, \( k \) outcomes and \( p \) predictors:

- A vector representing the random-effects parameters defining the between-study (co)variance matrix.
- A \( k \times k \) matrix representing the current estimate of the between-study (co)variance matrix.
- A \( m \)-dimensional list of study-specific design matrices for the fixed-effects part of the model. Rows corresponding to missing outcomes have been excluded.
- A \( m \)-dimensional list of study-specific vectors of estimated outcomes. Entries corresponding to missing outcomes have been excluded.
- A \( m \)-dimensional list of within-study (co)variance matrices of estimated outcomes. Rows and columns corresponding to missing outcomes have been excluded.
- A \( m \)-dimensional list of \( k \)-dimensional study-specific logical vectors, identifying missing outcomes.
- Numeric scalars: number of outcomes, number of studies included in estimation (equal to the length of lists above), number of predictors (including the intercept), number of observations (excluding missing).
- A string defining the between-study (co)variance structure in likelihood based models. See Details.
- A list of parameters for controlling the fitting process, usually internally set to default values by `mvmeta.control`. The name is chosen to avoid conflicts with the argument control in `optim`.

Details

These functions are called internally by the fitting functions `mvmeta.ml` and `mvmeta.reml` to perform iterative optimization algorithms for estimating random effects meta-analytical models.

The maximization of the (restricted) likelihood starts with few runs of an iterative generalized least square algorithm implemented in `iter.igls`. This can be regarded as a fast and stable way to get starting values close to the maximum for the Quasi-Newton iterative algorithm, implemented in `optim`. Alternatively, starting values can be provided by the user in the control list (see `mvmeta.control`). The function `optim` requires the algorithms to compute the value of the (restricted) likelihood and (optionally) the vector of its first partial derivatives, provided by the related likelihood functions.

These functions actually specify the profiled version of the (restricted) likelihood, expressed only in terms of random-effects parameters, while the estimate of the fixed-effects coefficients is provided at each iteration by the internal function `glsfit`, based on the current value of the between-study (co)variance matrix. At convergence, the value of this profiled version is identical to the full (restricted) likelihood. This approach is computationally efficient, as it reduces the number of parameters in the optimization routine, especially for meta-regression models.

The random-effects parameters in `par` depends on the chosen `structure` for the between-study (co)variance matrix. The parameterization ensures the positive-definiteness of the estimated matrix. A Cholesky decomposition is then performed on the marginal (co)variance matrix in order to
re-express the problem as standard least square equations, an approach which speeds up the com-
putation of matrix inverses and determinants. These equations are finally solved through a QR
decomposition, which guarantees stability. More details are provided in the references below.

Some parameters of the fitting procedures are determined through `mvmeta.control`.

**Value**

`mlprof.fn` and `remlprof.fn` return the value of the (restricted) log-likelihood for a given set of
parameters in `par`. `mlprof.gr` and `remlprof.gr` return instead the related vector of first partial
derivatives. `iter.igls` returns an updated estimate of $\Psi$ given its initial value or the value at the
previous iteration.

**Note**

As stated earlier, these functions are called internally by `mvmeta.ml` and `mvmeta.reml`, and are not
meant to be used directly. In particular, their code does not contain any check on the arguments
provided, which are expected in specific formats. They are however exported in the namespace and
documented for completeness.

**Author(s)**

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

**References**

Gasparrini A, Armstrong B, Kenward MG (2012). Multivariate meta-analysis for non-linear and
here].


1022.

Verlag.

**See Also**

See `mvmeta.fit` and `mvmeta.ml` for additional info on the fitting procedures. See `mvmeta.control`
to determine specific parameters of the fitting procedures. See `mvmetaCovStruct` for (co)variance
structures. See `chol` and `qr` for info on the Cholesky and QR decomposition. See `mvmeta-package`
for an overview of the package and modelling framework.
model.frame.mvmeta  Extract Model Frame and Design Matrix from mvmeta Objects

Description

These method functions return the model frame and design matrix for univariate or multivariate meta-analytical models represented in objects of class "mvmeta".

Usage

```r
## S3 method for class 'mvmeta'
model.frame(formula, ...)
```

```r
## S3 method for class 'mvmeta'
model.matrix(object, ...)
```

Arguments

- `object`, `formula`
  
  an object of class "mvmeta".

- `...`
  
  further arguments passed to or from other methods.

Details

The model frame is produced by `mvmeta` when fitting the meta-analytical model, and stored in the `mvmeta` object if argument `model=TRUE`. Alternatively, the model frame is directly returned by `mvmeta` with argument `method="model.frame"`. The method function `model.frame` simply extracts the saved model frame if available, or otherwise evaluates a call to `mvmeta` when `method="model.frame".

The method function `model.matrix` extracts the design matrix from a fitted meta-analytical model. It first extract the model frame by calling `model.frame`, and then passes the call to the default method.

These methods functions are similar to those provided for regression objects `lm` and `lm`.

Value

For `model.frame`, a data.frame with special attributes (see the default method `model.frame`) and the additional class "data.frame.mvmeta".

For `model.matrix`, the design matrix used to fit the model.

Note

The reason why these specific method functions are made available for class `mvmeta`, and in particular why a new class "data.frame.mvmeta" has been defined for model frames, lies in the special handling of missing values in multivariate meta-analysis models fitted with `mvmeta`. Methods `na.omit` and `na.exclude` for class "data.frame.mvmeta" are useful for properly accounting for missing values when fitting these models.
mvmeta

Author(s)
Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

See Also
See the default methods \texttt{model.frame} and \texttt{model.matrix}. See \texttt{na.omit} and \texttt{na.exclude} on the handling of missing values. See \texttt{mvmeta-package} for an overview of the package and modelling framework.

Examples

\begin{verbatim}
# RUN THE MODEL AND SUMMARIZE THE RESULTS
model <- mvmeta(cbind(PD, AL)~pubyear, S=berkey98[5:7], data=berkey98, method="ml")

# MODEL FRAME
model$model
model.frame(model)
update(model, method="model.frame")
class(model.frame(model))

# MODEL MATRIX
model.matrix(model)
\end{verbatim}

\textbf{Description}

The function \texttt{mvmeta} performs fixed and random-effects multivariate and univariate meta-analysis and meta-regression, with various estimation methods. The function \texttt{mvmeta.fit} is a wrapper for actual fitting functions based on different estimation methods, usually called internally. See \texttt{mvmeta-package} for an overview.

\textbf{Usage}

\begin{verbatim}
mvmeta(formula, S, data, subset, method="reml", bscov="unstr", model=TRUE,
contrasts=NULL, offset, na.action, control=list())
mvmeta.fit(X, y, S, offset=NULL, method="reml", bscov="unstr", control=list())
\end{verbatim}

\textbf{Arguments}

Assuming a meta-analysis or meta-regression based on \textit{m} studies, \textit{k} outcomes and \textit{p} predictors:

an object of class "\texttt{formula}" (or one that can be coerced to that class) offering a symbolic description of the linear predictor of the model to be fitted to each outcome. Alternatively, for meta-analysis with no predictor, a single vector (for
univariate models) or matrix-type object (for multivariate models). Terms in formula must be vector or matrix-type objects, optionally provided in the data argument below. See Details.

**Formula**
a $m \times p$ design matrix containing the study-specific predictors. Usually produced internally by `mvmeta` from formula above.

**y**
a $m$-dimensional vector (for univariate models) or $m \times k$ matrix (for multivariate models) of outcomes. Usually produced internally by `mvmeta` from formula above.

**S**
series of within-study (co)variance matrices of the estimated outcomes for each one of the $m$ studies. Accepted formats by `mvmeta` are: a $m$-dimensional list of $k \times k$ matrices; a tri-dimensional $k \times k \times m$ array; a matrix or data frame with $m$ rows and $k(k+1)/2$ or $k$ columns, depending on the availability of the within-study correlations. Optionally, terms may be provided in the data argument below. `mvmeta` fits accepts only the last option. See Details below.

**data**
an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in formula. If not found in data, the variables are taken from environment(formula), typically the environment from which `mvmeta` is called.

**subset**
an optional vector specifying a subset of observations to be used in the fitting process.

**method**
estimation method: "fixed" for fixed-effects models, "ml" or "reml" for random-effects models fitted through (restricted) maximum likelihood, "mm" for random-effects models fitted through method of moments, and "vc" for random-effects models fitted through variance components. See Details below. If "model.frame", the model frame is returned, as in `lm` or `glm`.

**bscov**
a string defining the between-study (co)variance structure in likelihood based models. Default to "unstr" (unstructured). Among other options, the user can select "diag" (diagonal), "cs" (compound symmetry), "hcs" (heterogeneous compound symmetry), "arf" (autoregressive of first order), "fixed" (fixed). See Details.

**model**
a logical value indicating whether the model frame should be included as a component of the returned value. See the `model.frame` method function.

**contrasts**
an optional list. See the `contrasts.arg` of `model.matrix`.

**offset**
optionally, a $m$-dimensional numeric vector used to specify an a priori known component in the linear predictor. One or more offset terms can be included in the formula instead or as well. See `model.offset`.

**na.action**
a function which indicates what should happen when the data contain NAs. Default to na.action setting of `options`, usually `na.omit`. `na.exclude` can be useful. See details on missing values in `mvmeta`.

**control**
list of parameters for controlling the fitting process. These are passed to `mvmeta.control` by `mvmeta.fit` to replace otherwise selected default values.

**Details**

The function `mvmeta` resembles standard regression functions in R (see `lm` or `glm`). This function defines the design matrix and the vector (for univariate models) or matrix (for multivariate models)
of outcome responses, and calls the wrapper mvmeta.fit to perform the actual fitting. The latter prepares the data and calls specific fitting functions, depending on the chosen method. Functions other than mvmeta are not expected to be called directly for model fitting.

The model is specified through a regression formula. Simple meta-analysis is specified with the formula $y \sim 1$, where the left-hand side is a vector (in univariate models) or a matrix (in multivariate models), optionally of form cbind(y1,...,yk), with terms stored in data. Alternatively, matrix or vector objects are allowed, and the formula is retrieved internally adding $\sim 1$. In meta-regression, other terms are added in the right-hand side of the formula, defining the linear predictor common to all outcomes. Factors, variable transformations and interactions are allowed, following the standard formula specification. Labels are automatically retrieved from the objects in formula. See formula for further details. See lm or glm for info on the other arguments.

The within-study (co)variances are provided through the argument S, usually as a matrix. If the correlations are available, each of the $m$ row represents the $k(k+1)/2$ vectorized entries of the lower triangle of the related (co)variance matrix, taken by column (see xpndMat). If correlations are not available, each row represents the $k$ variances, and the correlations are inputted internally through the argument Scor of the control list. See inputcov.

Different estimator are available in the package mvmeta and chosen through the argument method. In the current version, the options are:

- method="fixed": Fixed-effects estimator
- method="ml": Maximum likelihood (ML) estimator
- method="reml": Restricted maximum likelihood (REML) estimator
- method="mm": Method of moments estimator
- method="vc": Variance components estimator

Specific fitting functions are called internally. Likelihood-based methods allow alternative (co)variance structures for the between-study random effects through the argument bscov. See their help pages for further details on the estimation methods, following the links above.

Missing values are allowed in both sides of formula. In the case of missing predictors (right-hand side of formula), the related study is entirely excluded from estimation. In contrast, a study still contributes to estimation if at least outcome is non-missing. This behaviour is different than in standard regression functions such as lm or glm. Before the call to mvmeta.fit, studies matching such missing definition are removed from the the model frame. The missing pattern in S must be consistent with that in y. See further details on handling missing values in mvmeta.

The fitting procedure can be controlled through the additional terms specified in control, which are passed to the function mvmeta.control.

Value

The mvmeta function typically returns a list object of class "mvmeta" representing the meta-analytical model fit, as described in mvmetaObject. When method="data.frame", the model is not fitted and the model frame is returned, namely a data frame with special attributes (see the default method model.frame) and, in this case, the additional class "data.frame.mvmeta".

The wrapper function mvmeta.fit is usually called internally in mvmeta, and returns an intermediate list object with some of the components expected in the "mvmeta" class.

Several method functions for regression objects are available, either default or specifically-written for the "mvmeta" class. See mvmetaObject for a complete list.
Note

In the current version, the same linear predictor specified in formula is set for all the outcomes.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References


See Also

See additional info on the estimation procedures at the related page of the fitting functions. See alternative (co)variance structures for likelihood-based estimation methods. See handling of missing values in mvmeta. See lm or glm for standard regression functions. See mvmeta-package for an overview of this modelling framework.

Examples

```r
### BIVARIATE META-ANALYSIS, ESTIMATED THROUGH REML

# RUN THE MODEL
model <- mvmeta(cbind(PD, AL), S=berkey98[5:7], data=berkey98)

# SUMMARIZE THE RESULTS
summary(model)

# RESIDUALS AND FITTED VALUES
residuals(model)
fitted(model)

# LOG-LIKELIHOOD AND AIC VALUE
logLik(model)
AIC(model)

### BIVARIATE META-REGRESSION, ESTIMATED THROUGH METHOD OF MOMENTS

# RUN THE MODEL AND SUMMARIZE THE RESULTS
```
model <- mvmeta(cbind(PD,AL)-pubyear,S=berkey98[5:7],data=berkey98,method="mm")
summary(model)

# BLUP ESTIMATES AND 90% PREDICTION INTERVALS, AGGREGATED BY OUTCOME
blup(model,pi=TRUE,aggregate="y",pi.level=0.90)

# COCHRAN Q TEST FOR RESIDUAL HETEROGENEITY
qtest(model)

# PREDICTED AVERAGED OUTCOMES AND STANDARD ERRORS FROM YEAR 1985 TO 1989
newdata <- data.frame(pubyear=1985:1989)
predict(model,newdata,se=TRUE)

# MODEL FRAME AND MODEL MATRIX
model.frame(model)
model.matrix(model)

### UNIVARIATE META-REGRESSION, FIXED-EFFECTS MODEL

# RUN THE MODEL
model <- mvmeta(sbp=ish,S=sbp_se^2,data=hyp,method="fixed")
summary(model)

# RESIDUALS AND FITTED VALUES
residuals(model)
fitted(model)

# COCHRAN Q TEST FOR RESIDUAL HETEROGENEITY
qtest(model)

### MULTIVARIATE META-ANALYSIS WITH MORE THAN 2 OUTCOMES

# RUN THE MODEL
y <- as.matrix(fibrinogen[2:5])
S <- as.matrix(fibrinogen[6:15])
model <- mvmeta(y,S)
summary(model)

### IN THE PRESENCE OF MISSING VALUES

# RUN THE MODEL
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S)
summary(model)

# MODEL FRAME: SEE help(na.omit.data.frame.mvmeta) FOR MORE EXAMPLES
model.frame(model)
### WHEN WITHIN-STUDY COVARIANCES ARE NOT AVAILABLE AND/OR NEED TO BE INPUTTED

```r
# GENERATE S
(S <- inputcov(hyp[c("sbp_se","dbp_se"),,cor=hyp$rho])

# RUN THE MODEL
model <- mvmeta(cbind(sbp,dbp),S=S,data=hyp)

# INPUTTING THE CORRELATION DIRECTLY IN THE MODEL
model <- mvmeta(cbind(y1,y2),cbind(V1,V2),data=p53,control=list(Scor=0.95))
summary(model)

# SEE help(hyp) AND help(p53) FOR MORE EXAMPLES
```

### STRUCTURING THE BETWEEN-STUDY (CO)VARIANCE

```r
# DIAGONAL
S <- as.matrix(hsls[5:10])
model <- mvmeta(cbind(b1,b2,b3),data=hsls,bcov="diag")
summary(model)
model$Psi

# COMPOUND SYMMETRY
model <- mvmeta(cbind(b1,b2,b3),data=hsls,bcov="cs")
summary(model)
model$Psi

# SEE help(mvmetaCovStruct) FOR DETAILS AND ADDITIONAL EXAMPLES
```

### USE OF THE CONTROL LIST

```r
# PRINT THE ITERATIONS AND CHANGE THE DEFAULT FOR STARTING VALUES
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S,control=list(showiter=TRUE,igls.iter=20))

# SEE help(mvmeta.control) FOR FURTHER DETAILS
```

---

**mvmeta.control**

Ancillary Parameters for Controlling the Fit in mvmeta Models

### Description

This internal function sets the parameter options used for fitting meta-analytical models, commonly to pre-specified default values. It is usually internally called by `mvmeta.fit`.  

mvmeta.control

Usage

mvmeta.control(optim=list(), showiter=FALSE, maxiter=100, initPsi=NULL, Psifix=NULL, Psicor=0, Scor=0, inputna=FALSE, inputvar=10^4, igls.iter=10, vc.adj=TRUE, reltol=sqrt(.Machine$double.eps), set.negeigen=sqrt(.Machine$double.eps))

Arguments

optim list of parameters passed to the control argument of the function optim, which performs the quasi-Newton optimization in likelihood-based random-effects models. See optim.

showiter logical. If TRUE, the progress of iterative optimization is shown.

maxiter positive interger value. Maximum number of iterations in methods involving optimization procedures.

initPsi either a matrix or a vector of its lower triangular elements (with diagonal, taken by column) from which starting values of the parameters of the between-study (co)variance matrix are derived, used in the optimization procedure for likelihood-based random-effects models. If NULL (the default, and recommended), the starting value is created internally through an iterative generalized least square algorithm.

Psifix either a matrix or a vector of its lower triangular elements (with diagonal, taken by column) equal or proportional to the between-study (co)variance. Only used when bscov="fixed" or bscov="prop" in mvmeta, and, if not provided, it set internally to a 0 or identity matrix, respectively.

Psicor either a scalar, vector or matrix representing the between-study correlation(s) (see inputcov). Only used when bscov="cor" in mvmeta.

Scor either a scalar, vector or matrix representing the within-study correlation(s) to be inputted when the covariances are not provided, and ignored if they are (see inputcov).

inputna logical. If missing values must be internally inputted. To be used with caution, see inputa.

inputvar multiplier for inputting the missing variances, to be passed as an argument to inputna.

igls.iter number of iteration of the iterative generalized least square algorithm to be run in the hybrid optimization procedure of linkelihood-based models to provide the starting value. See iter.igls.

vc.adj logical. If TRUE, an adjustment to the way the marginal variance part is computed in the variance components estimator is applied. See mvmeta.vc.

reltol relative convergence tolerance in methods involving optimization procedures. The algorithm stops if it is unable to reduce the value by a factor of reltol * (abs(val) + reltol) at a step.

set.negeigen positive value. Value to which negative eigenvalues are to be set in estimators where such method is used to force positive semi-definiteness of the estimated between-study (co)variance matrix.
mvmeta.fixed

Details

The control argument of mvmeta is by default passed to mvmeta.fit, which uses its elements as arguments of mvmeta.control.

Many arguments refer to specific fitting procedures. Refer to the help page of the related estimator for details.

The function is expected to be extended and/or modified at every release of the package mvmeta.

Value

A list with components named as the arguments.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References


See Also

See mvmeta. See also glm.control. See the help pages of the related fitting functions for details on each parameter. See mvmeta-package for an overview of this modelling framework.

Examples

# PRINT THE ITERATIONS (SEE ?optim) AND CHANGE THE DEFAULT FOR STARTING VALUES
model <- mvmeta(cbind(PD, AL)~pubyear, S=berkey98[5:7], data=berkey98,
               control=list(showiter=TRUE, igls.iter=20))

# INPUT THE CORRELATION
model <- mvmeta(cbind(y1,y2), S=cbind(V1,V2), data=p53, control=list(Scor=0.5))

mvmeta.fixed

Fixed-Effects Estimator for mvmeta Models

Description

This function implements a generalized least square estimator for fixed-effects multivariate and univariate meta-analysis and meta-regression. It is meant to be used internally and not directly run by the users.

Usage

mvmeta.fixed(Xlist, ylist, Slist, nalist, k, m, p, nall, control, ...)
Arguments

Assuming a meta-analysis or meta-regression based on \( m \) studies, \( k \) outcomes and \( p \) predictors:

- A \( m \)-dimensional list of study-specific design matrices for the fixed-effects part of the model. Rows corresponding to missing outcomes have been excluded.

\$\textbf{Xlist}\$

- A \( m \)-dimensional list of study-specific vectors of estimated outcomes. Entries corresponding to missing outcomes have been excluded.

\$\textbf{Slist}\$

- A \( m \)-dimensional list of within-study (co)variance matrices of estimated outcomes. Rows and columns corresponding to missing outcomes have been excluded.

\$\textbf{Nalist}\$

- A \( m \)-dimensional list of \( k \)-dimensional study-specific logical vectors, identifying missing outcomes.

\( \textbf{k} \), \( \textbf{m} \), \( \textbf{p} \), \( \textbf{nall} \) numeric scalars: number of outcomes, number of studies included in estimation (equal to the length of lists above), number of predictors (including the intercept), number of observations (excluding missing).

\$\textbf{control}\$

- List of parameters for controlling the fitting process, usually internally set to default values by \texttt{mvmeta.control}.

\( \ldots \)

- Further arguments passed to or from other methods. Currently not used.

Details

The estimation involves only the \( kp \) fixed-effects coefficients.

The routine is based on a standard generalized least square (GLS) algorithm implemented in the internal function \texttt{glsfit}. The between-study (co)variance matrix is set to zero, so the marginal (co)variance matrix, composed only by elements of the within-study component, is assumed as completely known. Similarly to the likelihood-based estimators implemented in \texttt{mvmeta.ml} and \texttt{mvmeta.reml}, the computation involves Cholesky and and QR decompositions for computational stability and efficiency. The method is described in details in Gasparrini and collaborators (2012) (see references below).

Value

This function returns an intermediate list object, whose components are then processed by \texttt{mvmeta.fit}. Other components are added later through \texttt{mvmeta} to finalize an object of class "mvmeta".

Note

As stated earlier, this function is called internally by \texttt{mvmeta.fit}, and is not meant to be used directly. In particular, its code does not contain any check on the arguments provided, which are expected in specific formats. The function is however exported in the namespace and documented for completeness.

The arguments above are prepared by \texttt{mvmeta.fit} from its arguments \( X \), \( y \) and \( S \). The list structure, although requiring more elaborate coding, is computationally more efficient, as it avoids the specification of sparse block-diagonal matrices, especially for meta-analysis involving a large number of studies.
Some parameters of the fitting procedures are determined by the control argument, with default set by `mvmeta.control`. No missing values are accepted in the fitting functions. See details on missing values.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References


See Also

See `mvmeta` for the general usage of the functions. See `mvmeta.control` to determine specific parameters of the fitting procedures. Use the triple colon operator (`:::`) to access the code of the internal functions, such as `glsfit`. See `mvmeta-package` for an overview of the package and modelling framework.

Examples

```r
# UNIVARIATE FIXED-EFFECTS MODEL
model <- mvmeta(yC=S=SCC, data=smoking, method="fixed")
summary(model)

# MULTIVARIATE FIXED-EFFECTS MODEL
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S,method="fixed")
summary(model)

# MULTIVARIATE FIXED-EFFECTS MODEL: REPLICATE THE RESULTS IN BERKEY ET AL. 1998
model <- mvmeta(cbind(PD,AL)-I(pubyear-1983),S=berkey98[1:3],
data=berkey98,method="fixed")
summary(model)
```

---

`mvmeta.ml` 

*ML and REML Estimators for mvmeta Models*
**Description**

These functions implement maximum likelihood (ML) and restricted maximum likelihood (REML) estimators for random-effects multivariate and univariate meta-analysis and meta-regression. They are meant to be used internally and not directly run by the users.

**Usage**

```r
mvmeta.ml(xlist, ylist, Slist, nalist, k, m, p, nall, bscov, control, ...)

mvmeta.reml(xlist, ylist, Slist, nalist, k, m, p, nall, bscov, control, ...)
```

**Arguments**

Assuming a meta-analysis or meta-regression based on \( m \) studies, \( k \) outcomes and \( p \) predictors:

- `xlist` a \( m \)-dimensional list of study-specific design matrices for the fixed-effects part of the model. Rows corresponding to missing outcomes have been excluded.
- `ylist` a \( m \)-dimensional list of study-specific vectors of estimated outcomes. Entries corresponding to missing outcomes have been excluded.
- `Slist` a \( m \)-dimensional list of within-study (co)variance matrices of estimated outcomes. Rows and columns corresponding to missing outcomes have been excluded.
- `nalist` a \( m \)-dimensional list of \( k \)-dimensional study-specific logical vectors, identifying missing outcomes.
- `k`, `m`, `p`, `nall` numeric scalars: number of outcomes, number of studies included in estimation (equal to the length of lists above), number of predictors (including the intercept), number of observations (excluding missing).
- `bscov` a string defining the between-study (co)variance structure in likelihood based models. See Details.
- `control` list of parameters for controlling the fitting process, usually internally set to default values by `mvmeta.control`.
- `...` further arguments passed to or from other methods. Currently not used.

**Details**

The estimation involves \( kp \) fixed-effects coefficients and a number of random-effects parameters defining the between-study (co)variance matrix, depending on the chosen (co)variance structure. Up to \( k(k + 1)/2 \) parameters are needed for an unstructured form, while a smaller set are required for structured matrices.

The hybrid estimation procedure is based first on few runs of iterative generalized least square algorithm and then quasi-Newton iterations, using specific likelihood functions, until convergence. The estimation algorithm adopts a profiled (or concentrated) approach, that is expressed only in terms of the random-effects parameters. Cholesky and QR decompositions are used for computational stability and efficiency, and for assuring the positive-definiteness of the estimated between-study (co)variance matrix. See the help page for the likelihood functions for further details. The method is described in details in Gasparrini and collaborators (2012) (see references below).
Value

These functions return an intermediate list object, whose components are then processed by `mvmeta.fit`. Other components are added later through `mvmeta` to finalize an object of class “mvmeta”.

Note

As stated earlier, these functions are called internally by `mvmeta.fit`, and are not meant to be used directly. In particular, their code does not contain any check on the arguments provided, which are expected in specific formats. The functions are not exported in the namespace, and only documented for completeness.

The arguments above are prepared by `mvmeta.fit` from its arguments `X`, `y` and `S`. The list structure, although requiring more elaborate coding, is computationally more efficient, as it avoids the specification of sparse block-diagonal matrices, especially for meta-analysis involving a large number of studies.

Some parameters of the fitting procedures are determined by the `control` argument, with default set by `mvmeta.control`. No missing values are accepted in the fitting functions. See details on missing values.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References


See Also

See `mvmeta` for the general usage of the functions. See `mvmeta.control` to determine specific parameters of the fitting procedures. Use the triple colon operator (‘:::’) to access the code of the internal functions, such as `glsfit`. See `mvmeta-package` for an overview of the package and modelling framework.
mvmeta.mm

Method of Moments Estimator for mvmeta Models

Description
This function implements a method of moments estimator for multivariate and univariate random-effects meta-analysis and meta-regression. It is meant to be used internally and not directly run by the users.

Usage
mvmeta.mm(Xlist, ylist, Slist, nalist, k, m, p, nall, control, ...)

Arguments
Assuming a meta-analysis or meta-regression based on \( m \) studies, \( k \) outcomes and \( p \) predictors:

a \( m \)-dimensional list of study-specific design matrices for the fixed-effects part of the model. Rows corresponding to missing outcomes have been excluded.
xlist a \( m \)-dimensional list of study-specific vectors of estimated outcomes. Entries corresponding to missing outcomes have been excluded.

slist a \( m \)-dimensional list of within-study (co)variance matrices of estimated outcomes. Rows and columns corresponding to missing outcomes have been excluded.

nalist a \( m \)-dimensional list of \( k \)-dimensional study-specific logical vectors, identifying missing outcomes.

\( k, m, p, nall \) numeric scalars: number of outcomes, number of studies included in estimation (equal to the length of lists above), number of predictors (including the intercept), number of observations (excluding missing).

control list of parameters for controlling the fitting process, usually internally set to default values by \texttt{mvmeta.control}.

... further arguments passed to or from other methods. Currently not used.

Details

The estimation involves \( kp \) fixed-effects coefficients and \( k(k + 1)/2 \) random-effects parameters, corresponding to the lower triangular entries of the between-study (co)variance matrix.

The approach implemented here represents the multivariate extension of the traditional estimator proposed by DerSimonian and Laird (1986), and simplifies to the standard method in the univariate case. This non-iterative routine forces the positive semi-definiteness of the estimated between-study (co)variance matrix by setting its negative eigenvalues to zero.

The specific method of moment estimator used here is described in Jackson and collaborators (2013), and represents a generalization of that developed by Chen and collaborators (2012). However, this general version is computationally more intensive, and may turn out to be slow when applied to meta-analysis of a relatively high number of studies. An alternative and computationally faster method of moment estimator was previously proposed by Jackson and collaborators (2010), although it is not invariant to reparameterization. This latter estimator is not implemented yet in \texttt{mvmeta}. See references below.

Value

This function returns an intermediate list object, whose components are then processed by \texttt{mvmeta.fit}. Other components are added later through \texttt{mvmeta} to finalize an object of class “mvmeta”.

Note

As stated earlier, this function is called internally by \texttt{mvmeta.fit}, and is not meant to be used directly. In particular, its code does not contain any check on the arguments provided, which are expected in specific formats. The function is however exported in the namespace and documented for completeness.

The arguments above are prepared by \texttt{mvmeta.fit} from its arguments \( X, y \) and \( S \). The list structure, although requiring more elaborate coding, is computationally more efficient, as it avoids the specification of sparse block-diagonal matrices, especially for meta-analysis involving a large number of studies.
Some parameters of the fitting procedures are determined by the control argument, with default set by `mvmeta.control`. No missing values are accepted in the fitting functions. See details on missing values.

**Author(s)**

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

**References**


**See Also**

See `mvmeta` for the general usage of the functions. See `mvmeta.control` to determine specific parameters of the fitting procedures. Use the triple colon operator (`:::`) to access the code of the internal functions, such as `fbtr`. See `mvmeta-package` for an overview of the package and modelling framework.

**Examples**

```r
# MM ESTIMATOR: UNIVARIATE MODEL
model <- mvmeta(PD~pubyear,S=berkey98[,5],data=berkey98,method="mm")
summary(model)

# MM ESTIMATOR: REPRODUCE THE RESULTS IN CHEN ET AL. (2012)
S <- as.matrix(hsls[5:10])
model <- mvmeta(cbind(b1,b2,b3),S=data=hsls,method="mm")
summary(model)

# MM ESTIMATOR: REPRODUCE THE RESULTS IN JACKSON ET AL. (2013)
S <- inputcov(hyp["sbp_se","dbp_se"],cor=hyp$rho)
model <- mvmeta(cbind(sbp,dbp),S=S,data=hyp,method="mm")
summary(model)
```
Description

This function implements a variance components estimator for multivariate and univariate random-effects meta-analysis and meta-regression. It is meant to be used internally and not directly run by the users.

Usage

mvmeta vc(Xlist, ylist, Slist, nalist, k, m, p, nall, control, ...)

Arguments

Assuming a meta-analysis or meta-regression based on \( m \) studies, \( k \) outcomes and \( p \) predictors:

- \( xlist \) a \( m \)-dimensional list of study-specific design matrices for the fixed-effects part of the model. Rows corresponding to missing outcomes have been excluded.
- \( ylist \) a \( m \)-dimensional list of study-specific vectors of estimated outcomes. Entries corresponding to missing outcomes have been excluded.
- \( Slist \) a \( m \)-dimensional list of within-study (co)variance matrices of estimated outcomes. Rows and columns corresponding to missing outcomes have been excluded.
- \( nalist \) a \( m \)-dimensional list of \( k \)-dimensional study-specific logical vectors, identifying missing outcomes.
- \( k, m, p, nall \) numeric scalars: number of outcomes, number of studies included in estimation (equal to the length of lists above), number of predictors (including the intercept), number of observations (excluding missing).
- \( control \) list of parameters for controlling the fitting process, usually internally set to default values by \texttt{mvmeta.control}.
- ... further arguments passed to or from other methods. Currently not used.

Details

The estimation involves \( kp \) fixed-effects coefficients and \( k(k + 1)/2 \) random-effects parameters, corresponding to the lower triangular entries of the between-study (co)variance matrix.

The procedure is based on the estimate of the between-study (co)variance as the difference between the marginal (co)variance and the average within-study (co)variance. This in turn requires the estimate of the marginal (co)variance, obtained by the residuals of the fitted model. The procedure is iterative, with the current estimate of the between-study (co)variance plugged into a generalized least square (GLS) routine. Starting values are provided by a fixed-effects estimator (see \texttt{mvmeta.fixed}). The algorithm is fast and generally converges with few iterations.

Similar versions of this estimator has been previously proposed. Berkey and collaborators (1998) simply called it GLS method, and a non-iterative approach was proposed by Ritz and collaborators.
(2008), referred to as MVEE3. A non-iterative version for univariate models is discussed in Sidik and Jonkman (2007). The results from Berkey and collaborators (1998) are reproduced in the example below.

In the original approach, the estimate of the marginal (co)variance is obtained from the sum of the residual components using a denominator equal to \( m(M-p) \). Following the development proposed by Kauermann and Carroll (2001) and Fay and Graubard (2001) in the context of sandwich (co)variance estimators, then discussed by Lu and collaborators (2007), an adjusted denominator can be computed as a quantity derived from the hat matrix. This method is expected to perform better in the presence of missing values and small data sets. This alternative adjustment is chosen by default by setting \( vc = \text{adj} = \text{TRUE} \) in the control argument.

The variance component estimator is not bounded to provide a positive semi-definite between-study (co)variance matrix, as shown in the simulation study by Liu and colleagues (2009). Here positive semi-definiteness is forced by setting the negative eigenvalues of the estimated matrix to zero at each iteration. Little is known about the impact of such constraint.

**Value**

This function returns an intermediate list object, whose components are then processed by \texttt{mvmeta.fit}. Other components are added later through \texttt{mvmeta} to finalize an object of class "mvmeta".

**Note**

As stated earlier, this function is called internally by \texttt{mvmeta.fit}, and is not meant to be used directly. In particular, its code does not contain any check on the arguments provided, which are expected in specific formats. The function is however exported in the namespace and documented for completeness.

The arguments above are prepared by \texttt{mvmeta.fit} from its arguments \( X, y \) and \( S \). The list structure, although requiring more elaborate coding, is computationally more efficient, as it avoids the specification of sparse block-diagonal matrices, especially for meta-analysis involving a large number of studies.

Some parameters of the fitting procedures are determined by the control argument, with default set by \texttt{mvmeta.control}. No missing values are accepted in the fitting functions. See details on missing values.

**Author(s)**

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

**References**


See Also
See mvmeta for the general usage of the functions. See mvmeta.control to determine specific parameters of the fitting procedures. Use the triple colon operator ('::::') to access the code of the internal functions, such as sumlist. See mvmeta-package for an overview of the package and modelling framework.

Examples

# VC ESTIMATOR: UNIVARIATE MODEL
model <- mvmeta(PD~pubyear,S=berkey98[,5],data=berkey98,method="vc")
summary(model)

# VC ESTIMATOR: MULTIVARIATE MODEL
model <- mvmeta(cbind(PD,AL)~pubyear,S=berkey98[5:7],data=berkey98,method="vc")
summary(model)

# VC ESTIMATOR: NON-ITERATIVE VERSION
model <- mvmeta(cbind(PD,AL)~pubyear,S=berkey98[5:7],data=berkey98,method="vc",
               control=list(maxiter=1))
summary(model)

# VARIANCE COMPONENTS ESTIMATOR: REPLICATE THE RESULTS IN BERKEY ET AL. (1998)
model <- mvmeta(cbind(PD,AL)~I(pubyear-1983),S=berkey98[5:7],
data=berkey98,method="vc",control=list(vc.adj=FALSE))
summary(model)

mvmetaCovStruct  Covariance Structures for mvmeta Models

Description
Alternative covariance structures for the between-study (co)variance matrix of random effects in multivariate meta-analysis or meta-regression, usually defined through the argument bscov of the function mvmeta.

Options
Assuming a meta-analysis or meta-regression based on k outcomes:

- unstr: an unstructured form for a general positive-definite matrix. The matrix is represented by k(k + 1)/2 unrestricted parameters defined as the upper triangular entries of its Cholesky decomposition.
• **diag**: a diagonal positive-definite matrix. The matrix is represented by \( k \) unrestricted parameters defined as the logarithm of the diagonal values.

• **id**: a multiple of the identity positive-definite matrix. The matrix is represented by a single unrestricted parameter defined as the logarithm of the diagonal value.

• **cs**: a positive-definite matrix with compound symmetry structure. The matrix is represented by 2 unrestricted parameters defined as the logarithm of the diagonal value and the transformed correlation. The latter is parameterized so to obtain a correlation value between \(-1/(k-1)\) and 1, in order to ensure positive-definiteness.

• **hcs**: a positive-definite matrix with heterogeneous compound symmetry structure. The matrix is represented by \( k + 1 \) unrestricted parameters defined as the logarithm of the diagonal values and the transformed correlation. The latter is parameterized so to obtain a correlation value between \(-1/(k-1)\) and 1, in order to ensure positive-definiteness.

• **ar1**: a positive-definite matrix with autoregressive structure of first order. The matrix is represented by \( k + 1 \) unrestricted parameters defined as the logarithm of the diagonal value and the logistic transformed correlation. The latter is parameterized so to obtain a correlation value between -1 and 1.

• **prop**: a positive-definite matrix proportional to that provided by the user through the argument `psifix` in the control list (see `mvmeta.control`). The matrix is represented by 1 unrestricted parameter defined as the logarithm of the multiplier.

• **cor**: a positive-definite matrix with correlation structure provided by the user through the argument `psicor` in the control list (see `mvmeta.control`). The matrix is represented by \( k \) unrestricted parameters defined as the logarithm of the diagonal values.

• **fixed**: a known matrix provided by the user through the argument `psifix` in the control list (see `mvmeta.control`). The matrix is known and no parameters are needed to represent it.

### Details

The structuring of the (co)variance matrix of random effects is only available for models estimated through (restricted) maximum likelihood.

The unrestricted parameters defining the between-study matrix are estimated in the iterative optimization algorithm. The starting values are usually obtained by an iterative generalized least square algorithm (see the specific `likelihood functions`). The algorithm computes an unstructured matrix. For structured forms, the starting values for variances and/or correlations are taken as the average of the related elements. Although rarely needed and not recommended, the user can provide a starting value of the (co)variance matrix, from which the parameters are derived (see `mvmeta.control`).

### Details

The structuring of the between-study (co)variance matrix of random effects is only available for models estimated through (restricted) maximum likelihood.

The unrestricted parameters defining the between-study matrix are estimated in the iterative optimization algorithm. The starting values are usually obtained by an iterative generalized least square algorithm (see the specific `likelihood functions`). The algorithm computes an unstructured matrix. For structured forms, the starting values for variances and/or correlations are taken as the average of the related elements. Although rarely needed and not recommended, the user can
provided a starting value of the (co)variance matrix, from which the parameters are derived (see `mvmeta.control`).

**Note**

The choice of structures can affect the performance of the optimization procedure, determining forms of likelihood surfaces which induce convergence to local maxima. In particular, structures such as multiple of identity of proportional to a fixed matrix are based on strong assumptions and should be used with caution.

**Author(s)**

Antonio Gasparini, <antonio.gasparrini@lshtm.ac.uk>

**References**


**See Also**

See `mvmeta`. See `lm` or `glm` for standard regression functions. See `mvmeta-package` for an overview of this modelling framework.

**Examples**

```r
# UNSTRUCTURED AND STRUCTURED BETWEEN-STUDY (CO)VARIANCE
y <- as.matrix(fibrinogen[2:5])
S <- as.matrix(fibrinogen[6:15])
model <- mvmeta(y,S)
summary(model)
model$Psi

# DIAGONAL
model <- mvmeta(y,S,bcov="diag")
summary(model)
model$Psi

# HETEROGENEOUS COMPOUND SYMMETRY
model <- mvmeta(y,S,bcov="hcs")
summary(model)
model$Psi

# PROPORTIONAL
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S,bcov="prop",control=list(Psifix=diag(3)+1))
summary(model)
model$Psi

# CORRELATION
model <- mvmeta(y,S,bcov="cor",control=list(Psicor=0.2))
```
**mvmetaObject**

summary(model)
model$Psi

---

### Description

An object returned by the `mvmeta` function, inheriting from class "mvmeta", and representing a fitted univariate or multivariate meta-analytical model.

### Value

Objects of class "mvmeta" are lists with defined components. Dimensions of such components may refer to \(k\) outcome parameters, \(p\) predictors and \(m\) studies used for fitting the model (the latter can be different from those originally selected due to missing). The following components needs to be included in a legitimate `mvmeta` object:

- **coefficients**: a \(p\)-dimensional vector (for univariate models) or a \(p \times k\) matrix (for multivariate models) of the fixed-effects coefficients.
- **vcov**: estimated \(kp \times kp\) (co)variance matrix of the fixed-effects coefficients.
- **Psi**: for random-effects models, the estimated \(k \times k\) between-study (co)variance matrix.
- **residuals**: a \(m\)-dimensional vector (for univariate models) or \(m \times k\) matrix (for multivariate models) of residuals, that is observed minus fitted values.
- **fitted.values**: a \(m\)-dimensional vector (for univariate models) or \(m \times k\) matrix (for multivariate models) of fitted mean values.
- **df.residual**: the residual degrees of freedom.
- **rank**: the numeric rank of the fitted model.
- **logLik**: the (restricted) log-likelihood of the fitted model. Set to `NA` for non-likelihood models.
- **converged, niter**: for models with iterative estimation methods, logical scalar indicating if the algorithm eventually converged, and number of iterations.
- **negeigen**: for models fitted through method of moments, the number of negative eigenvalues in the estimated between-study (co)variance matrix, then set to 0.
- **control**: a list with the values of the control arguments used, as returned by `mvmeta.control`.
- **method**: the estimation method.
- **bscov**: a string defining the between-study (co)variance structure in likelihood based models.
- **S**: a \(m \times k(k + 1)/2\) matrix, where each row represents the vectorized entries of the lower triangle of the related within-study (co)variance matrix, taken by column. See `mvmeta`.
**dim**

list with the following scalar components: \( m \) (number of studies included in estimation, which could be lower than the total number in the presence of missing values), \( k \) (number of outcome parameters), \( p \) (number of coefficients for each outcome parameter).

**df**

list with the following scalar components: \( nall \) (number of observations used for estimation, excluding missing values), \( nobs \) (equal to \( nall \), minus the number of fixed-effects coefficients in REML models), \( fixed \) (number of estimated fixed-effects coefficients), \( random \) (number of estimated (co)variance terms).

**lab**

list with the following label vectors: \( m \) for the \( m \) studies, \( k \) for the \( k \) outcome parameters, \( p \) for the \( p \) predictors (including intercept). The first two are derived from the vector/matrix of outcome parameters in `formula`, the third from the design matrix derived from `model.matrix`.

**model**

the model frame used for fitting. Reported if `model=TRUE` in `mvmeta`. See `model.frame`.

**call**

the function call.

**na.action**

(where relevant) information returned by `model.frame` on the special handling of NAs. See info on missing values.

**formula**

the model supplied.

**terms**

the `terms` object representing the fitted model.

**contrasts**

(where relevant) the contrasts used.

**xlevels**

(where relevant) a record of the levels of the factors used in fitting.

**Methods**

A number of methods functions are available for `mvmeta` objects, most of them common to other regression functions.

Specifically-written method functions are defined for `predict` (standard predictions) and `blup` (best linear unbiased predictions). The method function `simulate` produces simulated outcomes from a fitted model, while `qtest` performs the Cochran Q test for heterogeneity. Other methods have been produced for `summary`, `logLik`, `coef`, and `vcov`.

Specific methods are also available for `model.frame` and `model.matrix`. In particular, the former produces the model frame (a data frame with special attributes storing the variables used for fitting) with the additional class "data.frame.mvmeta". Methods `na.omit` and `na.exclude` for this class are useful for the handling of missing values in `mvmeta` objects.

Printing functions for the objects of classes defined above are also provided. `anova` methods for performing tests in `mvmeta` objects are in development.

All the methods above are visible (exported from the namespace) and documented. In additions, several default method functions for regression are also applicable to objects of class "mvmeta", such as `fitted`, `residuals`, `AIC`, `BIC` and `update`, among others.

**Author(s)**

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>
References


See Also

See `mvmeta`. See `lm` or `glm` for standard regression functions. See `mvmeta-package` for an overview of this modelling framework.

Examples

```r
# RUN THE MODEL
g <- mvmeta(cbind(PD, AL) ~ pubyear, S = berkey98[5:7], data = berkey98)

# INSPECT THE OBJECT
g

# LABELS
g$lab

# FORMULA
g$formula

# CONVERGED?
g$converged
```

---

### mvmetaSim

*Simulating Responses for mvmeta Models*

**Description**

These functions simulate sets of multivariate or univariate responses for a group of studies, in terms of their mean (expected) values and within and between-study (co)variances. These sets of outcomes can be used in meta-analytical models for simulation purposes.

**Usage**

```r
mvmetaSim(y, S, Psi, sd, cor, nsim = 1, seed = NULL, posdef_tol)
```

```r
## S3 method for class 'mvmeta'
simulate(object, nsim = 1, seed = NULL, ...)
```

**Arguments**

In order to simulate $k$ outcomes for $m$ studies:

- a $m$-dimensional vector (for simulating univariate responses) or $m \times k$ matrix (for simulating multivariate responses) of mean (expected) outcomes.
series of within-study (co)variance matrices of the outcomes for each one of the
\( m \) studies. Accepted formats are a \( m \)-dimensional list of \( k \times k \) matrices; a tri-
dimensional \( k \times k \times m \) array; or a \( m \times k(k+1)/2 \) matrix or data frame where
each row represents the vectorized entries of the lower triangle of the related
(co)variance matrix, taken by column (see \texttt{xpndMat}).

\( \Psi \) the between-study (co)variance matrices of the outcomes. Accepted formats are
a \( k \times k \) matrix or a \( k(k+1)/2 \)-dimensional vector, representing the vectorized
entries of the lower triangle of the related (co)variance matrix, taken by column
(see \texttt{xpndMat}).

\( \text{sd} \) a \( k \)-dimensional vector of between-study standard deviations.

\( \text{cor} \) between-study correlations. Either a scalar, a vector or a matrix. See \texttt{inputcov}.

\( \text{nsim} \) number of simulation sets.

\( \text{seed} \) an object specifying if and how the random number generator should be initial-
ized.

\( \text{posdeftol} \) tolerance (relative to largest variance) for numerical lack of positive-definiteness.
Default to the square root of the machine precision.

\( \text{object} \) an object of class "mvmeta".

... further arguments passed to or from other methods.

Details

The set(s) of responses can be simulated either from a fitted model, using the method function
\texttt{simulate} for objects of class "mvmeta", or directly through the function \texttt{mvmetaSim}. In the former
case, the fitted values from the model are used as mean (expected) outcomes, together with the
within and estimated between-study (co)variance. In the latter option, this information need to be
provided by the user in the correct dimensions and forms.

In \texttt{mvmetaSim}, the between-study (co)variance matrix can be inputted directly through \( \Psi \), or given
in the form \( D R D \), with a diagonal matrix \( D \) of standard deviations and correlation matrix \( R \).
These values are provided through \texttt{sd} and \texttt{cor}. See \texttt{inputcov} for details.

The functions simulate the responses for each study separately from a marginal multivariate normal
distribution with mean equal to the expected values and (co)variance equal to the sum of the within
and between-study components. The computation is identical to that implemented in the func-
tion \texttt{mvrnorm} of the package \texttt{MASS}, involving a eigen decomposition of the marginal (co)variance
matrix. Numerical negative definiteness is allowed with a tolerance specified by \texttt{posdeftol}, and
positive semi-definiteness is then forced by truncating the eigenvalues at zero.

Value

If \( \text{nsim}=1 \), a matrix or vector of simulated \( k \) outcomes for the \( m \) studies. If more simulation sets
are required (\( \text{nsim} \) higher than 1), a list of matrices or vectors.

Note

Studies with missing values in the fitted values or in the components of the within (co)variances are
excluded by \texttt{simulate}. Missing values are instead not accepted in \texttt{metaSim}.
Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

See Also

See simulate for the general method function. See inputcov for inputting correlations. See mvmeta-package for an overview of the package and modelling framework.

Examples

# RUN A MODEL
model <- mvmeta(cbind(PD, AL) ~ pubyear, S = berkey98[5:7], data = berkey98)

# SIMULATE A NEW SET OF OUTCOMES
simulate(model)

# SIMULATE FROM SCRATCH: 3 OUTCOMES, 8 STUDIES
(y <- matrix(0, 8, 3))
(S <- inputcov(matrix(runif(8*3, 0, 1), 8, 3), dimnames = list(NULL,
   c("V1", "V2", "V3")), cor = c(0, 0.5, 0.7)))
(Psi <- inputcov(1:3, cor = 0.3))
mvmetaSim(y, S, Psi)

# ALTERNATIVELY, DEFINE Psi THROUGH STANDARD DEVIATIONS AND CORRELATION 0.2
mvmetaSim(y, S, sd = 1:3, cor = 0.3)

# 2 SIMULATION SETS
mvmetaSim(y, S, Psi, nsim = 2)
Arguments

object  an object of class "data.frame.mvmeta".
...  further arguments passed to or from other methods.

Details

A model frame of class "data.frame.mvmeta" is produced by \texttt{mvmeta}. A call to \texttt{na.omit} or \texttt{na.exclude} removes from the model frame the rows corresponding to studies with invalid missing pattern. In addition, a \texttt{na.action} attribute is added to the model frame, namely a numeric vector corresponding to the removed rows and class "omit" or "exclude", respectively. This information is used by \texttt{naresid} and \texttt{napredict} to deal with missing values in functions such as \texttt{fitted}, \texttt{residuals}, \texttt{predict} and \texttt{blup}, among others.

The definition of missing, identifying an invalid missing pattern, is different in multivariate meta-analytical models performed through \texttt{mvmeta} if compared to other regression functions such as \texttt{lm} or \texttt{glm}. Specifically, while a study is removed if at least an observation for one predictor is missing, partially missing outcomes do not prevent the study to contribute to estimation (see \texttt{mvmeta}). Specific methods \texttt{na.omit} and \texttt{na.exclude} for class "data.frame.mvmeta" allow this different definition.

Value

These functions returns the model frame object with rows corresponding to studies with invalid missing pattern being removed. They also add the related \texttt{na.action} attribute as explained above.

Author(s)

Antonio Gasparrini, \texttt{antonio.gasparrini@lshtm.ac.uk}

See Also

See \texttt{na.action}, \texttt{naresid} and \texttt{napredict}. See \texttt{model.frame}. See \texttt{mvmeta-package} for an overview of the package and modelling framework.

Examples

\begin{verbatim}
# INPUT MISSING VALUES IN PREDICTOR AND ONE RESPONSE
data <- berkey98

# RUN THE MODEL
model <- mvmeta(cbind(PD,AL)-pubyear,S=data[5:7],data=data,method="ml")

# SUMMARIZE: NOTE THE NUMBER OF STUDIES AND OBSERVATIONS
summary(model)
df.residual(model)

# EXTRACT THE MODEL FRAME WITH na.pass
model.frame(model,na.action="na.pass")
# EXTRACT THE MODEL FRAME WITH na.omit (DEFAULT)
\end{verbatim}
p53

Mutant p53 Gene and Squamous Cell Carcinoma

Description
The dataset includes studies providing evidence about whether the presence of mutant p53 tumour suppressor gene is a prognostic factor for patients presenting with squamous cell carcinoma arising from the oropharynx cavity. Unadjusted estimates of log hazard ratios of mutant p53 to normal p53 for disease-free and overall survival, together with the associated variances, are collected from 6 observational studies.

Usage
p53

Format
A data frame with 6 observations on the following 5 variables:

- study: study ID.
- y1, V1: estimate and associated variance of the log hazard ratio for disease-free survival.
- y2, V2: estimate and associated variance of the log hazard ratio for overall survival.

Details
Only 3 studies provide estimates for disease-free survival. The within-study correlations are not reported in the original studies but are expected to be highly positively correlated. The original data are described in Tandon and colleagues (2010) and used as an example in Jackson and colleagues (2011).

Note
The data provide an example of application of multivariate meta-analysis when the within-study correlations are not known. These correlations can be inputted directly in the `mvmeta` function through the control argument. See `mvmeta.control` for details.
Source


Examples

```r
### REPRODUCE THE RESULTS OF EXAMPLE 3 IN JACKSON ET AL. (2011)

# INSPECT THE DATA
p53

# REML MODEL WITH INPUTTED CORRELATION EQUAL TO 0.95
model <- mvmeta(cbind(y1,y2),cbind(V1,V2),data=p53,control=list(Scor=0.95))
print(summary(model),digits=2)
```

**predict.mvmeta**  
Predicted Values from mvmeta Models

Description

This method function computes predictions from fitted univariate or multivariate meta-analytical models represented in objects of class "mvmeta", optionally for a new set of predictor values in meta-regression models. Predictions are optionally accompanied by standard errors, confidence intervals or the entire (co)variance matrix of the predicted outcomes.

Usage

```
## S3 method for class 'mvmeta'
predict(object, newdata, se=FALSE, ci=FALSE, vcov=FALSE, 
       interval=c("confidence","prediction"), ci.level=0.95, 
       format=c("matrix","list"), aggregate=c("stat","y"), na.action=na.pass, ...)
```

Arguments

- `object`: an object of class "mvmeta".
- `newdata`: An optional data frame in which to look for variables values with which to predict from meta-regression models.
- `se`: logical switch indicating if standard errors must be included.
- `ci`: logical switch indicating if confidence intervals must be included.
- `vcov`: logical switch indicating if the (co)variance matrix must be included.
- `interval`: type of prediction. See Details.
- `ci.level`: a numerical value between 0 and 1, specifying the confidence level for the computation of confidence intervals.
predict.mvmeta

format the format for the returned results. See Value.
aggregate when format="matrix" and se or ci are required, the results may be aggregated by statistic or by outcome. See Value
na.action a function which indicates what should happen when the data contain NAs. The default to the value saved in object. See Note.

Details

The method function predict produces predicted values from mvmeta objects, obtained by evaluating the original call to mvmeta in the frame newdata. For both fixed and random-effects models, estimated predictions are only based on the fixed part of the model, differently from blup.

For random-effects models, if interval="confidence" (the default), standard errors, confidence intervals and (co)variance matrix of the predicted values are computed only using the estimated (co)variance matrix of the fixed-effects coefficients. If interval="prediction", the estimated between-study (co)variance matrix, stored in the Psi component of mvmeta objects, is also added. In this case, interpretation of the uncertainty reflects that of a new single study sampled from the same population.

If newdata is omitted, the predictions are based on the data used for the fit. In that case how to handle predictions for studies removed from estimation due to invalid missing pattern is determined by the na.action argument used in mvmeta to produce object. If na.action=na.omit, studies excluded from estimation will not appear, whereas if na.action=na.exclude they will appear, with values set to NA for all the outcomes. This step is performed by napredict. See Notes.

Value

The results may be aggregated in matrices (the default), or returned as lists, depending on the argument format. For multivariate models, the aggregation is ruled by the argument aggregate, and the results may be grouped by statistic or by outcome. If vcov=TRUE, lists are always returned.

Note

The definition of missing in model frames used for estimation in mvmeta is different than that commonly adopted in other regression models such as lm or glm. See info on missing values in mvmeta.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References

See Also

See blup for best linear unbiased predictions. See the default method predict. See mvmeta-package for an overview of the package and modelling framework.

Examples

```
# RUN THE MODEL
model <- mvmeta(cbind(PD, AL) ~ pubyear, S = berkey98[5:7], data = berkey98)

# PREDICTED FROM YEAR 1985 TO 1987, WITH LABELS

# AVERAGED OUTCOMES AND SE
predict(model, newdata, se = TRUE)

# SAME AS ABOVE, AGGREGATED BY OUTCOME
predict(model, newdata, se = TRUE, aggregate = "y")

# SAME AS ABOVE, WITH PREDICTION INTERVALS
predict(model, newdata, se = TRUE, aggregate = "y", interval = "prediction")

# WITH VCOV, FORCED TO A LIST
predict(model, newdata, se = TRUE, vcov = TRUE, aggregate = "y")
```
Author(s)
Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

See Also
See qtest.mvmeta for the specific method function applied to objects of class "mvmeta".

qtest.mvmeta  Cochran Q Test of Heterogeneity for mvmeta Models

Description
This method function performs a Cochran Q test of (residual) heterogeneity on fitted univariate or multivariate meta-analytical models represented in objects of class "mvmeta".

Usage
```r
## S3 method for class 'mvmeta'
qtest(object, ...)

## S3 method for class 'qtest.mvmeta'
print(x, digits=3, ...)
```

Arguments
- `object, x` objects of classes "mvmeta" and "qtest.mvmeta", respectively.
- `digits` an integer specifying the number of digits to which printed results must be rounded.
- `...` further arguments passed to or from other methods.

Details
In multivariate models, the test assesses the null hypothesis that the variability in the multivariate distribution of the outcomes is explained only in terms of estimation error in each study, measured by the within-study (co)variance matrices stored in the component `S` of `mvmeta` objects. This is equal to test the hypothesis that the between-study (co)variance matrix is a zero matrix, and there is no random deviation in study-specific estimates. Tests for single outcome parameters, comparable to estimates from multiple univariate meta-analysis, are also reported. This test reduces to the standard Q test in univariate models.

The function compute the statistics by actually fitting the related fixed-effects model, re-evaluating the call of the model with method changed to "fixed".
Value

A list object of class "qtest.mvmeta" with the following components:

- **Q**
  
  the vector of test statistics for overall and outcome-specific tests, distributed under the null hypothesis as a Chi-square with degrees of freedom **df**.

- **df**
  
  the vector of degrees of freedom of the null distribution for overall and outcome-specific tests. For the overall test, equal to the number of observations used for estimation minus the number of coefficients in the fixed part of the model. For outcome-specific test, equal to number of observed values minus the number of coefficients.

- **pvalue**
  
  the vector of p-values for overall and outcome-specific tests.

- **residual**
  
  logical switch indicating if a meta-regression model is assessed, meaning that the tested heterogeneity is residual.

- **k**
  
  dimensionality of the overall test, that is the number of outcome parameters in the model.

As usual, the print method function for class "qtest.mvmeta" does not return any value.

Note

Tests on single outcome parameters are performed by extracting the related estimates and variances, but they do not account for the correlation between them, which nevertheless has been considered in estimation. These tests are not therefore comparable with those performed by running a univariate model on each outcome parameter.

Author(s)

Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

References


See Also

See qtest for the generic method function. See mvmeta-package for an overview of the package and modelling framework.
Examples

```r
# RUN THE MODEL
model <- mvmeta(cbind(PD, AL)^-1, S=berkey98[5:7], data=berkey98)

# MULTIVARIATE COCHRAN Q TEST FOR HETERGENEITY
test <- qtest(model)
print(test, digits=2)
unclass(test)
```

Description

The dataset contains the results of 24 trials comparing four alternative interventions to promote smoking cessation. The trials have different designs, comparing two or three different interventions. The data consist of the number of successes out of the total participants, and the estimated log-odds ratio for arms B, C, and D relative to arm A, as well as the (co)variance matrix of these three estimates.

Usage

`smoking`

Format

A data frame with 24 observations on the following 19 variables:

- `study`: study ID.
- `design`: design of the trial, reporting the interventions being compared.
- `da, dB, dC, dD`: number of successes for each intervention.
- `na, nB, nC, nD`: number of participants for each intervention.
- `yB, yC, yD`: estimated log-odds ratios for interventions B, C and D versus intervention A.
- `SBB, SBC, SBD, SCC, SCD, SDD`: variances and co-variances of the estimated log-odds ratios for interventions B, C and D versus intervention A. The order corresponds to the lower triangular elements of the (co)variance matrix taken by column.

Details

Intervention A is chosen as the reference category. Trials without an arm A (trials 2 and 21-24) are augmented with an arm A with 0.01 individuals and 0.001 successes. Trials containing zero cells (trials 9 and 20) have 1 individual with 0.5 successes added to each intervention. Details on the data augmentation and estimation of (co)variances of the log-odds ratios are provided by White (2011).
Note

The data provide an example of application of network meta-analysis, also referred to as indirect mixed-treatment comparison. Additional information using examples based on these data are provided by Lu and Ades (2006), White (2011) and Higgins and colleagues (2012).

Source


Examples

```r
### REPRODUCE THE RESULTS IN WHITE (2011)

# INSPECT THE DATA
head(smoking)
names(smoking)

# UNSTRUCTURED BETWEEN-STUDY (CO)VARIANCE
y <- as.matrix(smoking[11:13])
S <- as.matrix(smoking[14:19])
model <- mvmeta(y,S)
summary(model)

# STRUCTURED BETWEEN-STUDY (CO)VARIANCE (PROPORTIONAL)
model <- mvmeta(y,S,bcov="prop",control=list(Psifix=diag(3)+1))
summary(model)

# SEE help(mvmetaCovStruct) for additional info and examples
```

---

**summary.mvmeta**

*Summarizing mvmeta Models*

Description

Print and summary method functions for fitted univariate or multivariate meta-analytical models represented in objects of class "mvmeta".

Usage

```r
## S3 method for class 'mvmeta'
summary(object, ci.level=0.95, ...)

## S3 method for class 'summary.mvmeta'
print(x, digits=4, ...)
```
## S3 method for class 'mvmeta'

`print(x, digits=4, ...)`

### Arguments

- **object**: an object of class "mvmeta" produced by a call to `mvmeta`.
- **x**: an object of class "mvmeta" or "summary.mvmeta", produced by calls to `mvmeta` or `summary.mvmeta`, respectively.
- **ci.level**: a numerical value between 0 and 1, specifying the confidence level for the computation of confidence intervals.
- **digits**: an integer specifying the number of digits to which printed results must be rounded.
- **...**: further arguments passed to or from other methods.

### Details

The `print` method function for class "mvmeta" only returns basic information on the fitted model, namely the call, estimated fixed-effects coefficients, dimensions and fit statistics (log-likelihood, AIC, BIC).

The `summary` method function computes additional statistics and tests, and produces a list object of class "summary.mvmeta". The `print` method function for this class shows additional information, such as tables reporting the estimates for the fixed and random-effects parts of the model, Cochran Q test for heterogeneity and $I^2$.

### Value

The `summary` method function for `mvmeta` objects produces a list of class "summary.mvmeta". The components of the lists are some of those stored in the related `mvmeta` object, plus the following:

- **coefficients**: a matrix reporting point estimates, standard errors, z statistics and related p-values of the test, and confidence intervals for the $k_p$ fixed-effects coefficients. Note this is different than the component with the same name stored in `mvmeta` objects, simply reporting the point estimates (see `mvmetaObject`).
- **AIC**: the value of the Akaike information criterion for the fitted mvmeta model, obtained through a call to `AIC`.
- **BIC**: the value of the Bayesian information criterion for the fitted mvmeta model, obtained through a call to `BIC`.
- **corFixed**: the $k_p \times k_p$ correlation matrix of the fixed-effects coefficients, obtained from the (co)variance matrix `vcov` (see `mvmetaObject` and `vcov`).
- **corRandom**: the $k \times k$ correlation matrix of the random effects, obtained from the between-study (co)variance matrix `Psi` (see `mvmetaObject`).
- **qstat**: results from the Cochran Q test for heterogeneity, namely a list corresponding to a `qtest.mvmeta` object without its class, obtained through `qtest`.
- **ci.level**: the confidence level used for defining the confidence intervals for the estimates of the fixed-effects coefficients.
As usual, the print method functions for classes "mvmeta" and "summary.mvmeta" do not return any value.

Author(s)

Antonio Gasparini, <antonio.gasparrini@lshtm.ac.uk>

References


See Also

See mvmeta and mvmetaObject.

Examples

# RUN THE MODEL
model <- mvmeta(cbind(PD, AL)~pubyear, S=berkey98[5:7], data=berkey98)

# SIMPLE PRINT
model
# DEFINE DIGITS
print(model,digit=2)
# SUMMARY WITH 80TH CONFIDENCE INTERVALS
summary(model,ci.level=0.80)

vechMat

Vectorization and expansion of symmetric matrices

Description

The function vechMat transforms a symmetric matrix in a vector containing its lower triangular elements, taken by column. The function xpndMat reverses this transformation.

Usage

vechMat(mat, diag=TRUE)

xpndMat(vech)

Arguments

mat a square matrix.
vech a vector.
diag a logical switch indicating if the diagonal entries must be included.
vechMat

Value
A vector for vechMat, a symmetric matrix for xpndMat.

Author(s)
Antonio Gasparrini, <antonio.gasparrini@lshtm.ac.uk>

See Also
See functions vech and xpnd in package MCMCpack.

Examples

# GENERATE A POSITIVE-DEFINITE MATRIX, VECTORIZE IT AND THEN RE-EXPAND
(M <- crossprod(matrix(rnorm(9),3)))
(v <- vechMat(M))
xpndMat(v)

# EXTRACT VECTORIZED S, EXPAND TO A LIST, AND RE-VECTORIZE
(S <- as.matrix(berkey98[5:7]))
(Slist <- lapply(seq(nrow(S)),function(i) xpndMat(S[i,i])))
t(sapply(Slist,vechMat))
Index

*Topic datasets
  berkey98, 6
  fibrinogen, 11
  hsls, 12
  hyp, 13
  p53, 49
  smoking, 55

*Topic htest
  qtest, 52
  qtest.mvmeta, 53

*Topic manip
  inputcov, 14
  inputna, 16
  na.omit.data.frame.mvmeta, 47
  vechMat, 58

*Topic methods
  blup.mvmeta, 7
  coef.mvmeta, 9
  loglik.mvmeta, 18
  model.frame.mvmeta, 22
  mvmetaSim, 45
  na.omit.data.frame.mvmeta, 47
  predict.mvmeta, 50
  qtest, 52
  qtest.mvmeta, 53
  summary.mvmeta, 56

*Topic models
  blup.mvmeta, 7
  coef.mvmeta, 9
  logLik.mvmeta, 18
  mlprof.fn, 19
  model.frame.mvmeta, 22
  mvmeta, 23
  mvmeta.control, 28
  mvmeta.fixed, 30
  mvmeta.ml, 32
  mvmeta.mm, 35
  mvmeta.vc, 38
  mvmetaCovStruct, 40
  mvmetaObject, 43
  mvmetaSim, 45
  na.omit.data.frame.mvmeta, 47
  predict.mvmeta, 50
  qtest.mvmeta, 53
  summary.mvmeta, 56

*Topic multivariate
  blup.mvmeta, 7
  coef.mvmeta, 9
  loglik.mvmeta, 18
  mlprof.fn, 19
  model.frame.mvmeta, 22
  mvmeta, 23
  mvmeta.control, 28
  mvmeta.fixed, 30
  mvmeta.ml, 32
  mvmeta.mm, 35
  mvmeta.vc, 38
  mvmetaCovStruct, 40
  mvmetaObject, 43
  mvmetaSim, 45
  na.omit.data.frame.mvmeta, 47
  predict.mvmeta, 50
  qtest.mvmeta, 53
  summary.mvmeta, 56

*Topic package
  mvmeta-package, 2

*Topic regression
  blup.mvmeta, 7
  coef.mvmeta, 9
  logLik.mvmeta, 18
  mlprof.fn, 19
  model.frame.mvmeta, 22
  mvmeta, 23
  mvmeta.control, 28
  mvmeta.fixed, 30
  mvmeta.ml, 32
  mvmeta.mm, 35
  mvmeta.vc, 38

60
INDEX

mvmetaCovStruct, 40
mvmetaObject, 43
mvmetaSim, 45
na.omit.data.frame.mvmeta, 47
predict.mvmeta, 50
qtest.mvmeta, 53
summary.mvmeta, 56

AIC, 4, 19, 44, 57
as.data.frame, 24

berkey98, 4, 6
BIC, 4, 19, 44, 57
blup, 4, 44, 48, 51, 52
blup(blup.mvmeta), 7
blup.mvmeta, 7

chol, 21
class, 52
coeff, 44
coeff.mvmeta, 9
control, 16

fibrinogen, 4, 11
fitted, 4, 44, 48
formula, 23, 25, 44

glm, 9, 24–26, 42, 45, 48, 51
glm.control, 30

hsls, 4, 12
hyp, 4, 13

inputcov, 4, 14, 17, 25, 29, 46, 47
inputna, 4, 16, 29
iter.igls, 29
iter.igls(mlprof.fn), 19

lm, 9, 22, 24–26, 42, 45, 48, 51
logLik, 4, 19, 44
logLik.mvmeta, 18

methods, 52
mlprof.fn, 19
mlprof.gr(mlprof.fn), 19
model.frame, 4, 22–25, 44, 48
model.frame.mvmeta, 22
model.matrix, 4, 23, 24, 44
model.matrix.mvmeta
  (model.frame.mvmeta), 22

model.offset, 24
mvmeta, 4, 8, 9, 15, 16, 22, 23, 29–32, 34, 36, 37, 39, 40, 42–45, 48, 49, 51, 57, 58
mvmeta-package, 2
mvmeta.control, 4, 15, 20, 21, 24, 25, 28,
  31–34, 36–43, 49
mvmeta.fit, 4, 21, 28, 30, 31, 34, 36, 39
mvmeta.fixed, 4, 30, 38
mvmeta.ml, 4, 20, 21, 31, 32
mvmeta.mm, 4, 35
mvmeta.reml, 4, 20, 21, 31
mvmeta.reml(mvmeta.ml), 32
mvmeta vc, 4, 29, 38
mvmetaCovStruct, 21, 40
mvmetaCovStruct, 21, 40
mvmetaObject, 4, 8, 25, 43, 57, 58
mvmetaSim, 4, 45

na.action, 48
na.exclude, 4, 22–24, 44
na.exclude.data.frame.mvmeta
  (na.omit.data.frame.mvmeta), 47
na.omit, 4, 22–24, 44
na.omit.data.frame.mvmeta, 47
napredict, 8, 48, 51
naresid, 48

offset, 24
optim, 20, 29
options, 24

p53, 4, 49
predict, 4, 8, 9, 44, 48, 52
predict.mvmeta, 50
print.mvmeta(summary.mvmeta), 56
print.qtest.mvmeta(qtest.mvmeta), 53
print.summary.mvmeta(summary.mvmeta),
  56

qr, 21
qtest, 4, 44, 52, 54, 57
qtest.mvmeta, 4, 52, 53, 53

remlprof.fn(mlprof.fn), 19
remlprof.gr(mlprof.fn), 19
residuals, 4, 44, 48

simulate, 4, 44, 47
simulate.mvmeta(mvmetaSim), 45
smoking, 4, 55
structure, 20
summary, 4, 44  
summary.mvmeta, 56  

terms, 44  

update, 44  

cov, 44, 57  
cov.mvmeta (coef.mvmeta), 9  
vechMat, 15, 58  

xpndMat, 15, 25, 46  
xpndMat (vechMat), 58