Package 'mmrm'

December 20, 2022

Type Package

Title Mixed Models for Repeated Measures

Version 0.2.2

Description Mixed models for repeated measures (MMRM) are a popular choice for analyzing longitudinal continuous outcomes in randomized clinical trials and beyond; see Cnaan, Laird and Slasor (1997) <doi:10.1002/(SICI)1097-0258(19971030)16:20%3C2349::AID-SIM667%3E3.0.CO;2-E> for a tutorial and Mallinckrodt, Lane and Schnell (2008) <doi:10.1177/009286150804200402> for a review. This package implements MMRM based on the marginal linear model without random effects using Template Model Builder ('TMB') which enables fast and robust model fitting. Users can specify a variety of covariance matrices, weight observations, fit models with restricted or standard maximum likelihood inference, perform hypothesis testing with Satterthwaite or Kenward-Roger adjustment, and extract least square means estimates by using 'emmeans'.

License Apache License 2.0

URL https://openpharma.github.io/mmrm/

BugReports https://github.com/openpharma/mmrm/issues

Depends R (>= 4.0)

Imports checkmate (>= 2.0), lifecycle, methods, nlme, numDeriv, parallel, Rcpp, Rdpack, stats, stringr, TMB (>= 1.9.1), utils

Suggests emmeans (>= 1.6), estimability, knitr, parallelly (>= 1.32.0), rmarkdown, testthat (>= 3.0.0), xml2

LinkingTo Rcpp, RcppEigen, testthat, TMB (>= 1.9.1)

VignetteBuilder knitr

RdMacros Rdpack

biocViews

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

NeedsCompilation yes

RoxygenNote 7.2.1

Collate 'catch-routine-registration.R' 'component.R' 'data.R' 'emmeans.R' 'fit.R' 'kenwardroger.R' 'mmrm-methods.R' 'mmrm-package.R' 'utils.R' 'satterthwaite.R' 'testing.R' 'tmb-methods.R' 'tmb.R'

Author Daniel Sabanes Bove [aut, cre],
Julia Dedic [aut],
Doug Kelkhoff [aut],
Kevin Kunzmann [aut],
Brian Matthew Lang [aut],
Liming Li [aut],
Ya Wang [aut],
Craig Gower-Page [ctb],
Boehringer Ingelheim Ltd. [cph, fnd],
Gilead Sciences, Inc. [cph, fnd],
F. Hoffmann-La Roche AG [cph, fnd],
Merck Sharp & Dohme, Inc. [cph, fnd]

Maintainer Daniel Sabanes Bove <daniel.sabanes_bove@roche.com>

Repository CRAN

Date/Publication 2022-12-20 09:50:02 UTC

R topics documented:

mmrm-package mmrm *Package*

Description

mmrm implements mixed models with repeated measures (MMRM) in R.

Author(s)

Maintainer: Daniel Sabanes Bove <daniel.sabanes_bove@roche.com>

Authors:

- Julia Dedic <julia.dedic@roche.com>
- Doug Kelkhoff <kelkhoff.douglas@gene.com>
- Kevin Kunzmann <kevin.kunzmann@boehringer-ingelheim.com>
- Brian Matthew Lang <brian.lang@msd.com>
- Liming Li <liming.li@roche.com>
- Ya Wang <ya.wang10@gilead.com>

Other contributors:

- Craig Gower-Page <craig.gower-page@roche.com> [contributor]
- Boehringer Ingelheim Ltd. [copyright holder, funder]
- Gilead Sciences, Inc. [copyright holder, funder]
- F. Hoffmann-La Roche AG [copyright holder, funder]
- Merck Sharp & Dohme, Inc. [copyright holder, funder]

See Also

Useful links:

- https://openpharma.github.io/mmrm/
- Report bugs at https://github.com/openpharma/mmrm/issues

component *Component Access for* mmrm_tmb *Objects*

Description

[Experimental]

Usage

```
component(
  object,
 name = c("cov_type", "n_theta", "n_subjects", "n_timepoints", "n_obs", "beta_vcov",
    "beta_vcov_complete", "varcor", "formula", "dataset", "n_groups", "reml",
   "convergence", "evaluations", "method", "conv_message", "call", "theta_est",
    "beta_est", "beta_est_complete", "beta_aliased", "x_matrix", "y_vector",
    "neg_log_lik", "jac_list", "theta_vcov")
)
```

Arguments

object	(mmrm_tmb) the fitted MMRM.
name	(character) the component(s) to be retrieved.

Details

Available component() names are as follows:

- call: low-level function call which generated the model.
- formula: model formula.
- dataset: data set name.
- cov_type: covariance structure type.
- n_theta: number of parameters.
- n_subjects: number of subjects.
- n_timepoints: number of modeled time points.
- n_obs: total number of observations.
- reml: was REML used (ML was used if FALSE).
- neg_log_lik: negative log likelihood.
- convergence: convergence code from optimizer.
- conv_message: message accompanying the convergence code.
- evaluations: number of function evaluations for optimization.

- method: Adjustment method which was used (for mmrm objects), otherwise NULL (for mmrm_tmb objects).
- beta_vcov: estimated variance-covariance matrix of coefficients (excluding aliased coefficients). For Kenward-Roger methods, the adjusted covariance matrix is returned (to still obtain the unadjusted covariance matrix use object$beta_vcov).
- beta_vcov_complete: estimated variance-covariance matrix including aliased coefficients with entries set to NA.
- varcor: estimated covariance matrix for residuals. If there are multiple groups, a named list of estimated covariance matrices for residuals will be returned. The names are the group levels.
- theta_est: estimated variance parameters.
- beta_est: estimated coefficients (excluding aliased coefficients).
- beta_est_complete: estimated coefficients including aliased coefficients set to NA.
- beta_aliased: whether each coefficient was aliased (i.e. cannot be estimated) or not.
- theta_vcov: estimated variance-covariance matrix of variance parameters.
- x_matrix: design matrix used (excluding aliased columns).
- y_vector: response vector used.
- jac_list: Jacobian, see h_jac_list() for details.

Value

The corresponding component of the object, see details.

See Also

In the lme4 package there is a similar function getME().

Examples

```
fit <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data
)
# Get all available components.
component(fit)
# Get convergence code and message.
component(fit, c("convergence", "conv_message"))
# Get modeled formula as a string.
component(fit, c("formula"))
```

`covariance_types` *covariance type*

Description

covariance type

Usage

```
cov_type

cov_type_spatial
```

Format

vector of supported covariance structures. `cov_type` for common time points covariance structures, `cov_type_spatial` for spatial covariance structures.

Details

abbreviation for covariance structures

Common Covariance Structures:

Structure	Description	Parameters	(i, j) element
ad	Ante-dependence	m	$\sigma^2 \prod_{k=i}^{j-1} \rho_k$
adh	Heterogeneous ante-dependence	$2m-1$	$\sigma_i \sigma_j \prod_{k=i}^{j-1} \rho_k$
ar1	First-order auto-regressive	2	$\sigma^2 \rho^{\lvert i-j \rvert}$
ar1h	Heterogeneous first-order auto-regressive	$m+1$	$\sigma_i \sigma_j \rho^{\lvert i-j \rvert}$
cs	Compound symmetry	2	$\sigma^2 [\rho I(i \neq j) + I(i = j)]$
csh	Heterogeneous compound symmetry	$m+1$	$\sigma_i \sigma_j [\rho I(i \neq j) + I(i = j)]$
toep	Toeplitz	m	$\sigma_{\lvert i-j \rvert + 1}$
toeph	Heterogeneous Toeplitz	$2m-1$	$\sigma_i \sigma_j \rho_{\lvert i-j \rvert}$
us	Unstructured	$m(m+1)/2$	σ_{ij}

where i and j denote i-th and j-th time points, respectively, out of total m time points, $1 \leq i, j \leq m$.

Note the **ante-dependence** covariance structure in this package refers to homogeneous ante-dependence, while the ante-dependence covariance structure from SAS `PROC MIXED` refers to heterogeneous ante-dependence and the homogeneous version is not available in SAS.

Spatial Covariance structures:

Structure	Description	Parameters	(i, j) element
sp_exp	spatial exponential	2	$\sigma^2 \rho^{-d_{ij}}$

where d_{ij} denotes the Euclidean distance between time points i and j.

Functions

- `cov_type`: non-spatial covariance structure
- `cov_type_spatial`: spatial covariance structure

`df_1d` *Calculation of Degrees of Freedom for One-Dimensional Contrast*

Description

[Experimental] Calculates the degree of freedom, F statistic and p value for multi-dimensional contrast, depending on the method used in `mmrm()`.

Usage

```
df_1d(object, contrast)
```

Arguments

`object`	(`mmrm`) the MMRM fit.
`contrast`	(`numeric`) contrast vector. Note that this should not include elements for singular coefficient estimates, i.e. only refer to the actually estimated coefficients.

Value

List with `est`, `se`, `df`, `t_stat` and `p_val`.

Examples

```
object <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data
)
contrast <- numeric(length(object$beta_est))
contrast[3] <- 1
df_1d(object, contrast)
```

df_md *Calculation of Degrees of Freedom for Multi-Dimensional Contrast*

Description

[Experimental] Calculates the estimate, standard error, degree of freedom, t statistic and p-value for one-dimensional contrast, depending on the method used in `mmrm()`.

Usage

```
df_md(object, contrast)
```

Arguments

object	(mmrm) the MMRM fit.
contrast	(matrix) numeric contrast matrix, if given a numeric then this is coerced to a row vector. Note that this should not include elements for singular coefficient estimates, i.e. only refer to the actually estimated coefficients.

Value

List with num_df, denom_df, f_stat and p_val (2-sided p-value).

Examples

```
object <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data
)
contrast <- matrix(data = 0, nrow = 2, ncol = length(object$beta_est))
contrast[1, 2] <- contrast[2, 3] <- 1
df_md(object, contrast)
```

emmeans_support *Support for* emmeans

Description

[Experimental]

This package includes methods that allow mmrm objects to be used with the emmeans package. emmeans computes estimated marginal means (also called least-square means) for the coefficients of the MMRM.

Examples

```
fit <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data
)
if (require(emmeans)) {
  emmeans(fit, ~ ARMCD | AVISIT)
}
```

---

fev_data *Example Data on FEV1*

Description

[Experimental]

Usage

```
fev_data
```

Format

A `tibble` with 800 rows and 7 variables:

- `USUBJID`: subject ID.
- `AVISIT`: visit number.
- `ARMCD`: treatment, `TRT` or `PBO`.
- `RACE`: 3-category race.
- `SEX`: sex.
- `FEV1_BL`: FEV1 at baseline (%).
- `FEV1`: FEV1 at study visits.
- `WEIGHT`: weighting variable.

Note

Measurements of FEV1 (forced expired volume in one second) is a measure of how quickly the lungs can be emptied. Low levels of FEV1 may indicate chronic obstructive pulmonary disease (COPD).

Source

This is an artificial dataset.

`fit_mmrm` *Low-Level Fitting Function for MMRM*

Description

[Experimental]

This is the low-level function to fit an MMRM. Note that this does not try different optimizers or adds Jacobian information etc. in contrast to `mmrm()`.

Usage

```
fit_mmrm(formula, data, weights, reml = TRUE, control = mmrm_control())
```

Arguments

`formula`	(`formula`) model formula with exactly one special term specifying the visits within subjects, see details.
`data`	(`data.frame`) input data containing the variables used in `formula`.
`weights`	(`vector`) input vector containing the weights.
`reml`	(`flag`) whether restricted maximum likelihood (REML) estimation is used, otherwise maximum likelihood (ML) is used.
`control`	(`mmrm_control`) list of control options produced by `mmrm_control()`.

Details

The `formula` typically looks like: `FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID)` so specifies response and covariates as usual, and exactly one special term defines which covariance structure is used and what are the visit and subject variables. Always use only the first optimizer if multiple optimizers are provided.

Value

List of class `mmrm_tmb`, see `h_mmrm_tmb_fit()` for details.

Examples

```
formula <- FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID)
data <- fev_data
system.time(result <- fit_mmrm(formula, data, rep(1, nrow(fev_data))))
```

`fit_single_optimizer` *Fitting an MMRM with Single Optimizer*

Description

[Experimental]

This function helps to fit an MMRM using `TMB` with a single optimizer, while capturing messages and warnings.

Usage

```
fit_single_optimizer(
  formula,
  data,
  weights,
  reml = TRUE,
  ...,
  control = mmrm_control(...)
)
```

Arguments

`formula`	(`formula`) the model formula, see details.
`data`	(`data`) the data to be used for the model.
`weights`	(`vector`) an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
`reml`	(`flag`) whether restricted maximum likelihood (REML) estimation is used, otherwise maximum likelihood (ML) is used.
`...`	Additional arguments to pass to `mmrm_control()`.
`control`	(`mmrm_control`) object.

Details

`fit_single_optimizer` will fit the `mmrm` model using the `control` provided. If there are multiple optimizers provided in `control`, only the first optimizer will be used.

Value

The `mmrm_fit` object, with additional attributes containing warnings, messages, optimizer used and convergence status in addition to the `mmrm_tmb` contents.

Examples

```
mod_fit <- fit_single_optimizer(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data,
  weights = rep(1, nrow(fev_data)),
  optimizer = "nlminb"
)
attr(mod_fit, "converged")
```

`free_cores` *Get an approximate number of free cores.*

Description

[Deprecated] use `parallelly::availableCores(omit = 1)` instead

Usage

```
free_cores()
```

Details

- This uses the maximum load average at 1, 5 and 15 minutes on Linux and Mac machines to approximate the number of busy cores. For Windows, the load percentage is multiplied with the total number of cores.
- We then subtract this from the number of all detected cores. One additional core is not used for extra safety.

Value

The approximate number of free cores, which is an integer between 1 and one less than the total cores.

Note

If executed during a unit test and on CRAN then always returns 1 to avoid any parallelization.

mmrm *Fit an MMRM*

Description

[Experimental]

This is the main function fitting the MMRM.

Usage

```
mmrm(
  formula,
  data,
  weights = NULL,
  reml = TRUE,
  control = mmrm_control(...),
  ...
)
```

Arguments

formula	(formula) the model formula, see details.
data	(data) the data to be used for the model.
weights	(vector) an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
reml	(flag) whether restricted maximum likelihood (REML) estimation is used, otherwise maximum likelihood (ML) is used.
control	(mmrm_control) fine-grained fitting specifications list created with mmrm_control().
...	arguments passed to mmrm_control().

Details

The formula typically looks like: FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID) so specifies response and covariates as usual, and exactly one special term defines which covariance structure is used and what are the time point and subject variables. The covariance structures in the formula can be found in covariance_types.

The time points have to be unique for each subject. That is, there cannot be time points with multiple observations for any subject. The rationale is that these observations would need to be correlated, but it is not possible within the currently implemented covariance structure framework to do that correctly.

When optimizer is not set, first the default optimizer (L-BFGS-B) is used to fit the model. If that converges, this is returned. If not, the other available optimizers from h_get_optimizers(), including BFGS, CG and nlminb are tried (in parallel if n_cores is set and not on Windows). If none of the optimizers converge, then the function fails. Otherwise the best fit is returned.

Note that fine-grained control specifications can either be passed directly to the mmrm function, or via the control argument for bundling together with the mmrm_control() function. Both cannot be used together, since this would delete the arguments passed via mmrm.

Value

An mmrm object.

Note

The mmrm object is also an mmrm_fit and an mmrm_tmb object, therefore corresponding methods also work (see mmrm_tmb_methods).

Additional contents depend on the choice of the adjustment method:

- If Satterthwaite adjustment is used, the Jacobian information jac_list is included.
- If Kenward-Roger adjustment is used, kr_comp contains necessary components and beta_vcov_adj includes the adjusted coefficients covariance matrix.

Use of the package emmeans is supported, see emmeans_support.

Examples

```
fit <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data
)

# Direct specification of control details:
fit <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data,
  weights = fev_data$WEIGHTS,
  method = "Kenward-Roger"
)

# Alternative specification via control argument (but you cannot mix the
# two approaches):
fit <- mmrm(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data,
  control = mmrm_control(method = "Kenward-Roger")
)
```

mmrm_control *Control Parameters for Fitting an MMRM*

Description

[Experimental] Fine-grained specification of the MMRM fit details is possible using this control function.

Usage

```
mmrm_control(
  n_cores = 1L,
  method = c("Satterthwaite", "Kenward-Roger", "Kenward-Roger-Linear"),
  start = NULL,
  accept_singular = TRUE,
  drop_visit_levels = TRUE,
  ...,
  optimizers = h_get_optimizers(...)
)
```

Arguments

n_cores	(int) number of cores to be used.
method	(string) adjustment method for degrees of freedom and coefficients covariance matrix.
start	(numeric or NULL) optional start values for variance parameters.
accept_singular	(flag) whether singular design matrices are reduced to full rank automatically and additional coefficient estimates will be missing.
drop_visit_levels	(flag) whether to drop levels for visit variable, if visit variable is a factor, see details.
...	additional arguments passed to h_get_optimizers().
optimizers	(list) optimizer specification, created with h_get_optimizers().

Details

The drop_visit_levels flag will decide whether unobserved visits will be kept for analysis. For example, if the data only has observations at visits VIS1, VIS3 and VIS4, by default they are treated to be equally spaced, the distance from VIS1 to VIS3, and from VIS3 to VIS4, are identical. However, you can manually convert this visit into a factor, with levels = c("VIS1", "VIS2", "VIS3",

"VIS4"), and also use drop_visits_levels = FALSE, then the distance from VIS1 to VIS3 will be double, as VIS2 is a valid visit. However, please be cautious because this can lead to convergence failure when using an unstructured covariance matrix and there are no observations at the missing visits.

Value

List of class mmrm_control with the control parameters.

Examples

```
mmrm_control(
  optimizer_fun = stats::optim,
  optimizer_args = list(method = "L-BFGS-B")
)
```

---

mmrm_tmb_methods *Methods for* mmrm_tmb *Objects*

Description

[Experimental]

Usage

```
## S3 method for class 'mmrm_tmb'
coef(object, complete = TRUE, ...)

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

## S3 method for class 'mmrm_tmb'
model.frame(formula, full = FALSE, ...)

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

## S3 method for class 'mmrm_tmb'
formula(x, ...)

## S3 method for class 'mmrm_tmb'
vcov(object, complete = TRUE, ...)

## S3 method for class 'mmrm_tmb'
VarCorr(x, sigma = NA, ...)

## S3 method for class 'mmrm_tmb'
deviance(object, ...)
```

```
## S3 method for class 'mmrm_tmb'
AIC(object, corrected = FALSE, ..., k = 2)

## S3 method for class 'mmrm_tmb'
BIC(object, ...)

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

Arguments

object	(mmrm_tmb) the fitted MMRM object.
complete	(flag) whether to include potential non-estimable coefficients.
...	not used.
formula	(mmrm_tmb) same as object.
full	(flag) whether to include subject, visit and weight variables.
x	(mmrm_tmb) same as object.
sigma	cannot be used (this parameter does not exist in MMRM).
corrected	(flag) whether corrected AIC should be calculated.
k	(number) the penalty per parameter to be used; default k = 2 is the classical AIC.

Value

Depends on the method, see Functions.

Functions

- coef(mmrm_tmb): obtains the estimated coefficients.
- fitted(mmrm_tmb): obtains the fitted values.
- model.frame(mmrm_tmb): obtains the model frame.
- logLik(mmrm_tmb): obtains the attained log likelihood value.
- formula(mmrm_tmb): obtains the used formula.
- vcov(mmrm_tmb): obtains the variance-covariance matrix estimate for the coefficients.
- VarCorr(mmrm_tmb): obtains the variance-covariance matrix estimate for the residuals.
- deviance(mmrm_tmb): obtains the deviance, which is defined here as twice the negative log likelihood, which can either be integrated over the coefficients for REML fits or the usual one for ML fits.

- `AIC(mmrm_tmb)`: obtains the Akaike Information Criterion, where the degrees of freedom are the number of variance parameters (`n_theta`). If `corrected`, then this is multiplied with `m / (m - n_theta - 1)` where `m` is the number of observations minus the number of coefficients, or `n_theta + 2` if it is smaller than that (Hurvich and Tsai 1989; Burnham and Anderson 1998).
- `BIC(mmrm_tmb)`: obtains the Bayesian Information Criterion, which is using the natural logarithm of the number of subjects for the penalty parameter `k`.
- `print(mmrm_tmb)`: prints the object.

References

- Hurvich CM, Tsai C (1989). "Regression and time series model selection in small samples." *Biometrika*, **76**(2), 297–307. doi:10.2307/2336663.
- Burnham KP, Anderson DR (1998). "Practical use of the information-theoretic approach." In *Model selection and inference*, 75–117. Springer. doi:10.1007/9781475729177_3.

Examples

```
formula <- FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID)
object <- fit_mmrm(formula, fev_data, weights = rep(1, nrow(fev_data)))
# Estimated coefficients:
coef(object)
# Fitted values:
fitted(object)
# Model frame:
model.frame(object)
model.frame(object, full = TRUE)
# Log likelihood given the estimated parameters:
logLik(object)
# Formula which was used:
formula(object)
# Variance-covariance matrix estimate for coefficients:
vcov(object)
# Variance-covariance matrix estimate for residuals:
VarCorr(object)
# REML criterion (twice the negative log likelihood):
deviance(object)
# AIC:
AIC(object)
AIC(object, corrected = TRUE)
# BIC:
BIC(object)
```

`refit_multiple_optimizers`

Refitting MMRM with Multiple Optimizers

Description

[Experimental]

Usage

```
refit_multiple_optimizers(fit, ..., control = mmrm_control(...))
```

Arguments

fit	(mmrm_fit) original model fit from fit_single_optimizer().
...	Additional arguments passed to mmrm_control().
control	(mmrm_control) object.

Value

The best (in terms of log likelihood) fit which converged.

Note

For Windows, no parallel computations are currently implemented.

Examples

```
fit <- fit_single_optimizer(
  formula = FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
  data = fev_data,
  weights = rep(1, nrow(fev_data)),
  optimizer = "nlminb"
)
best_fit <- refit_multiple_optimizers(fit)
```

Index