Package: MargCond (via r-universe)

March 24, 2025

Type Package

Title Joint Marginal-Conditional Model

Version 1.0.0

Date 2018-04-06

Author James Proudfoot

Maintainer James Proudfoot <jproudfoot@ucsd.edu>

Depends gee, lme4, MASS, Matrix

Description Fits joint marginal conditional models for multivariate longitudinal data, as in Proudfoot, Faig, Natarajan, and Xu (2018) <doi:10.1002/sim.7552>. Development of this package was supported by the UCSD Altman Translational Research Institute, NIH grant UL1TR001442. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

License GPL-2

NeedsCompilation no

Date/Publication 2018-04-09 11:00:52 UTC

Additional_repositories https://cranhaven.r-universe.dev

Config/pak/sysreqs cmake make

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

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/MargCond

RemoteSha 7a1234880bd474034d96d31aa6b809b94f150c44

RemoteSubdir MargCond

Contents

MargCond *Function to fit joint marginal-conditional models for longitudinal multivariate data.*

Description

Produces an object of class "MargCond" which is a marginal-conditional multivariate model.

Usage

```
MargCond(formula, data, ID, tol = 1e-04, max.iter = 50,
       corstr = "independence", silent = F)
```

Arguments

formula	a two-sided linear formula object similar to those in lmer.
data	a data frame in which to interpret the variables occuring in the formula.
ID	a vector which identifies the clusters. The length of ID should be the same as the number of observations. Data are assumed to be sorted so that observations on a cluster are contiguous rows for all entities in the formula.
tol	the tolerance used in the fitting algorithm.
max.iter	the maximum number of iterations for the ES algorithm.
corstr	a character string specifying the correlation structure. The following are permitted: "independence", "fixed", "stat_M_dep", "non_stat_M_dep", "exchangeable", "AR-M" and "unstructured"
silent	a logical variable controlling whether an indication at each iteration is printed.

Details

The joint marginal-conditional model

Care should be taken when specifying the random effects structure (see the singular models section of https://bbolker.github.io/mixedmodels-misc/glmmFAQ.html). As initial estimates for the expectation-substitution algorithm are taken from the univariate mixed model fits, we recommend that these models be fit separately first and examined to ensure that they are not over parameterized.

Value

An object of class "MargCond" representing the fit.

An object of class "MargCond" is a list containing the following components:

coefficients	a named vector of coefficients.
sigma	a named vector of outcome error standard deviations.
SE	a vector of coefficient, random effect, and error standard deviations.

residuals	the residuals, that is response minus fitted values.
working.correlation	the working correlation returned by the GEE step at convergence.
rand.eff	the random effect covariance matrix.
outcomes	vector of outcome names
Call	the matched call.
v.cov	the scaled covariance matrix of theta
obs	the total number of observations
groups	the total number of clusters
converge	logical indicator of whether the expectation-substitution algorithm converged (i.e. the difference between each element of theta from the previous iteration is smaller than tol, and the number of iterations is less than max.iter).

References

Proudfoot J. A., Faig W., Natarajan L., and Xu R. (2018) A joint marginal-conditional model for multivariate longitudinal data. *Statistics in Medicine*. https://doi.org/10.1002/sim.7552

See Also

gee, lmer.

Examples

```
set.seed(2112)
NN = 80
n_times = 1:3

## Simulating some data
simdat <- simDat(n = NN,
                 fixed_effects = list(c(1, 1, 2), c(1.5, 1, 3)),
                 rand_effects = list(1, 1),
                 error_var = c(4, 4),
                 error_structure = 'normal',
                 rho = .35,
                 times = n_times,
                 X = cbind(rep(1, NN * length(n_times)),
                           rnorm(NN * length(n_times), 0, 2),
                           rbinom(NN * length(n_times), 1, .5)),
                 Z = cbind(rep(1, NN * length(n_times))))

## Adding random missing values
aa <- sample(1:nrow(simdat), 10, replace = TRUE)
bb <- sample(1:7, 10, replace = TRUE)
for (i in 1:length(aa)) {
  simdat[aa[i], bb[i]] <- NA
}

## A fit for this simulated multivariate longitudinal data,
```

```
## including a random intercept and exchangeable correlation
## structure.
summary(MargCond(c(outcome1, outcome2) ~ X2 + X3 + (1 | ID),
        data = simdat, ID = simdat$ID, corstr = "exchangeable"))
```

`simDat` *Function to simulate multivariate longitudinal data*

Description

A function that simulates correlated multivariate data based on a set of fixed and random effects.

Usage

```
simDat(n, fixed_effects, rand_effects, error_var = c(2, 2),
error_structure = "normal", rho = 0, times = 1:5, X = NULL, Z = NULL)
```

Arguments

`n`	total sample size (number of clusters)
`fixed_effects`	list of fixed effect vectors for each outcome
`rand_effects`	list of random effect vectors for each outcome
`error_var`	vector of error variances for each outcome
`error_structure`	structure for the random error term, either `"normal"` for multivariate normal or `"50:50 normal"` for a mixture of two normal distributions
`rho`	correlation between outcomes
`times`	times for each repeated measure
`X`	fixed effect design matrix
`Z`	random effect design matrix

Value

A dataframe included simulated outcomes and the design matrices

Examples

```
set.seed(2112)
NN = 80
n_times = 1:3

## Simulating some data
simdat <- simDat(n = NN,
                 fixed_effects = list(c(1, 1, 2), c(1.5, 1, 3)),
                 rand_effects = list(1, 1),
                 error_var = c(4, 4),
```

```
                  error_structure = 'normal',
                  rho = .35,
                  times = n_times,
                  X = cbind(rep(1, NN * length(n_times)),
                            rnorm(NN * length(n_times), 0, 2),
                            rbinom(NN * length(n_times), 1, .5)),
                  Z = cbind(rep(1, NN * length(n_times))))

## Adding random missing values
aa <- sample(1:nrow(simdat), 10, replace = TRUE)
bb <- sample(1:7, 10, replace = TRUE)
for (i in 1:length(aa)) {
  simdat[aa[i], bb[i]] <- NA
}
```

Index