Package ‘jmcm’

October 13, 2025

Type Package

Title Joint Mean-Covariance Models using 'Armadillo' and S4

Version 0.2.5

Maintainer Jianxin Pan <`jianxinpan@uic.edu.cn`>

Description Fit joint mean-covariance models for longitudinal data. The models and their components are represented using S4 classes and methods. The core computational algorithms are implemented using the 'Armadillo' C++ library for numerical linear algebra and 'RcppArmadillo' glue.

License GPL (>= 2)

LazyData TRUE

Depends R (>= 3.2.2)

Imports Formula, methods, Rcpp (>= 0.12.14)

LinkingTo Rcpp, RcppArmadillo (>= 0.9.900.1.0), roptim

RoxygenNote 6.0.1

NeedsCompilation yes

Author Jianxin Pan [aut, cre],
Yi Pan [aut]

URL `https://github.com/ypan1988/jmcm/`

BugReports `https://github.com/ypan1988/jmcm/issues`

Suggests testthat, R.rsp

VignetteBuilder R.rsp

Repository CRAN

Date/Publication 2025-10-13 13:30:09 UTC

Contents

`acd_estimation` *Fit Joint Mean-Covariance Models based on ACD*

Description

Fit joint mean-covariance models based on ACD.

Usage

```
acd_estimation(m, Y, X, Z, W, start, mean, trace = FALSE, profile = TRUE,
  errormsg = FALSE, covonly = FALSE, optim_method = "default")
```

Arguments

`m`	an integer vector of numbers of measurements for subject.
`Y`	a vector of responses for all subjects.
`X`	model matrix for the mean structure model.
`Z`	model matrix for the diagonal matrix.
`W`	model matrix for the lower triangular matrix.
`start`	starting values for the parameters in the model.
`mean`	when covonly is true, it is used as the given mean.
`trace`	the values of the objective function and the parameters are printed for all the trace'th iterations.
`profile`	whether parameters should be estimated sequentially using the idea of profile likelihood or not.
`errormsg`	whether or not the error message should be print.
`covonly`	estimate the covariance structure only, and use given mean.
`optim_method`	optimization method, choose "default" or "BFGS"(vmmin in R).

See Also

`mcd_estimation` for joint mean covariance model fitting based on MCD, `hpc_estimation` for joint mean covariance model fitting based on HPC.

`aids` *Aids Data*

Description

The aids dataset comprises a total of 2376 CD4+ cell counts for 369 HIV infected men with a follow up period of approximately eight and half year. The number of measurements for each individual varies from 1 to 12 and the times are not equally spaced. The CD4+ cell data are highly unbalanced.

Usage

```
data(aids)
```

Format

A data frame with 2376 rows and 8 variables

Details

- id: subject id
- time: measurement time
- cd4: CD4+ cell count

`bootcurve` *Plot Fitted Curves and Corresponding Confidence Interval using bootstrapping method*

Description

Plot fitted curves and corresponding 95% confidence interval using bootstrapping method.

Usage

```
bootcurve(object, nboot)
```

Arguments

`object`	a fitted joint mean covariance model of class "jmcmMod", i.e., typically the result of jmcm().
`nboot`	number of the bootstrap replications.

Examples

```
## Not run:
# It may take hours for large bootstrap replications
fit.mcd <- jmcm(I(sqrt(cd4)) | id | time ~ 1 | 1, data=aids,
  triple = c(8, 1, 3), cov.method = 'mcd', control = jmcmControl(trace=T))
bootcurve(fit.mcd, nboot = 1000)

## End(Not run)
```

`cattle` *Cattle Data*

Description

Kenward (1987) reported an experiment in which cattle were assigned randomly to two treatment groups A and B, and their body weights were recorded in kilogram. Thirty animals received treatment A and another 30 received treatment B. The animals were weighted 11 times over a 133-day period; the first 10 measurements for each animal were made at two-week intervals and the last measurement was made one week later. Since no observation was missing, it is considered to be a balanced longitudinal dataset.

Usage

```
data(cattle)
```

Format

A data frame with 660 rows and 4 variables

Details

- id: subject id
- day: measurement time
- group: Treatment A or Treatment B
- weight: cattle weight

`getJMCM` *Extract or Get Generalized Components from a Fitted Joint Mean Covariance Model*

Description

Extract (or "get") "components" - in a generalized sense - from a fitted joint mean covariance model from an object of class "jmcmMod".

Usage

```
getJMCM(object, name, sub.num)

## S3 method for class 'jmcmMod'
getJMCM(object, name = c("m", "Y", "X", "Z", "W", "D", "T",
  "Sigma", "mu", "n2loglik", "grad", "hess", "theta", "beta", "lambda", "gamma",
  "loglik", "BIC", "iter", "triple"), sub.num = 0)
```

Arguments

`object`	a fitted joint mean covariance model of class "jmcmMod", i.e., typically the result of jmcm().
`name`	a character vector specifying the name(s) of the "component". When sub.num is not specified or equal to 0, possible values are: `"m"` a vector of number of measurement for each subject `"Y"` response vector `"X"` model matrix for mean structure `"Z"` model matrix for covariance structure (the diagonal matrix) `"W"` model matrix for covariance structure (the lower triangular matrix) `"theta"` parameter estimates of joint mean covariance model `"beta"` parameter estimates for mean structure model `"lambda"` parameter estimates for covariace structure (the diagonal matrix) `"gamma"` parameter estimates for covariance structure (the lower triangular matrix) `"loglik"` log-likelihood, except for a constant `"BIC"` Bayesian information criterion `"iter"` number of iterations until convergence `"triple"` (p, d, q) When sub.num is specified, possible values are: `"m"` number of measurements for subject i `"Y"` response vector for subject i `"X"` model matrix of subject i for mean structure `"Z"` model matrix of subject i for covariance structure (the diagonal matrix)

	`"W"` model matrix of subject i for covariance structure (the lower triangular matrix)
	`"D"` the estimated diagonal matrix for subject i
	`"T"` the estimated lower triangular matrix for subject i
	`"Sigma"` the estimated covariance matrix for subject i
	`"mu"` the estimated mean for subject i
	`"n2loglik"` the estimated -2l(theta)
	`"grad"` the estimated gradient
	`"hess"` the estimated Hessian matrix
`sub.num`	refer to i's subject

Methods (by class)

- `jmcmMod`: Extract or Get Generalized Components from a Fitted Joint Mean Covariance Model

Examples

```
fit.mcd <- jmcm(I(sqrt(cd4)) | id | time ~ 1 | 1, data = aids,
  triple = c(8, 1, 3), cov.method = 'mcd')

beta <- getJMCM(fit.mcd, "beta")
BIC  <- getJMCM(fit.mcd, "BIC")
Di   <- getJMCM(fit.mcd, "D", 10)
```

`hpc_estimation` *Fit Joint Mean-Covariance Models based on HPC*

Description

Fit joint mean-covariance models based on HPC.

Usage

```
hpc_estimation(m, Y, X, Z, W, start, mean, trace = FALSE, profile = TRUE,
  errormsg = FALSE, covonly = FALSE, optim_method = "default")
```

Arguments

`m`	an integer vector of numbers of measurements for subject.
`Y`	a vector of responses for all subjects.
`X`	model matrix for the mean structure model.
`Z`	model matrix for the diagonal matrix.
`W`	model matrix for the lower triangular matrix.
`start`	starting values for the parameters in the model.

`mean`	when covonly is true, it is used as the given mean.
`trace`	the values of the objective function and the parameters are printed for all the trace'th iterations.
`profile`	whether parameters should be estimated sequentially using the idea of profile likelihood or not.
`errormsg`	whether or not the error message should be print.
`covonly`	estimate the covariance structure only, and use given mean.
`optim_method`	optimization method, choose "default" or "BFGS"(vmmin in R).

See Also

`mcd_estimation` for joint mean covariance model fitting based on MCD, `acd_estimation` for joint mean covariance model fitting based on ACD.

`jmcm` *Fit Joint Mean-Covariance Models*

Description

Fit a joint mean-covariance model to longitudinal data, via maximum likelihood.

Usage

```
jmcm(formula, data = NULL, triple = c(3, 3, 3), cov.method = c("mcd",
  "acd", "hpc"), optim.method = c("default", "BFGS"),
  control = jmcmControl(), start = NULL)
```

Arguments

`formula`	a two-sided linear formula object describing the covariates for both the mean and covariance matrix part of the model, with the response, the corresponding subject id and measurement time on the left of a operator~, divided by vertical bars ("|").
`data`	a data frame containing the variables named in formula.
`triple`	an integer vector of length three containing the degrees of the three polynomial functions for the mean structure, the log innovation -variances and the autoregressive or moving average coefficients when 'mcd' or 'acd' is specified for cov.method. It refers to the degree for the mean structure, variances and angles when 'hpc' is specified for cov.method.
`cov.method`	covariance structure modelling method, choose 'mcd' (Pourahmadi 1999), 'acd' (Chen and Dunson 2013) or 'hpc' (Zhang et al. 2015).
`optim.method`	optimization method, choose 'default' or 'BFGS' (vmmin in R)
`control`	a list (of correct class, resulting from jmcmControl()) containing control parameters, see the *jmcmControl documentation for details.
`start`	starting values for the parameters in the model.

References

Pan J, Pan Y (2017). "jmcm: An R Package for Joint Mean-Covariance Modeling of Longitudinal Data." *Journal of Statistical Software*, 82(9), 1–29.

Examples

```
cattleA <- cattle[cattle$group=='A', ]
fit.mcd <- jmcm(weight | id | I(ceiling(day/14 + 1)) ~ 1 | 1,
data=cattleA, triple = c(8, 4, 3), cov.method = 'mcd',
control = jmcmControl(trace = TRUE, ignore.const.term = FALSE,
original.poly.order = TRUE))
```

`jmcmControl` *Control of Joint Mean Covariance Model Fitting*

Description

Construct control structures for joint mean covariance model fitting

Usage

```
jmcmControl(trace = FALSE, profile = TRUE, ignore.const.term = TRUE,
  original.poly.order = FALSE, errormsg = FALSE)
```

Arguments

`trace`	whether or not the value of the objective function and the parameters should be print on every trace'th iteration.
`profile`	whether or not parameters should be estimated sequentially using the idea of profile likelihood.
`ignore.const.term`	whether or not the constant term should be considered when calculating log-likelihood and BIC. function
`original.poly.order`	whether or not the original poly order p q d should be used
`errormsg`	whether or not the error message should be print

`jmcmMod-class` *Class "jmcmMod" of Fitted Joint Mean-Covariance Models.*

Description

Class "jmcmMod" of Fitted Joint Mean-Covariance Models.

Slots

- `call` the matched call
- `opt` the optimization result returned by optimizeJmcm
- `args` arguments m, Y, X, Z, W, time
- `triple` an integer vector of length three containing the degrees of the three polynomial functions for the mean structure, the log innovation -variances and the autoregressive or moving average coefficients when 'mcd' or 'acd' is specified for cov.method. It refers to the mean structure, variances and angles when 'hpc' is specified for cov.method.
- `devcomp` the deviance components list

`mcd_estimation` *Fit Joint Mean-Covariance Models based on MCD*

Description

Fit joint mean-covariance models based on MCD.

Usage

```
mcd_estimation(m, Y, X, Z, W, start, mean, trace = FALSE, profile = TRUE,
  errormsg = FALSE, covonly = FALSE, optim_method = "default")
```

Arguments

`m`	an integer vector of numbers of measurements for subject.
`Y`	a vector of responses for all subjects.
`X`	model matrix for the mean structure model.
`Z`	model matrix for the diagonal matrix.
`W`	model matrix for the lower triangular matrix.
`start`	starting values for the parameters in the model.
`mean`	when covonly is true, it is used as the given mean.
`trace`	the values of the objective function and the parameters are printed for all the trace'th iterations.

`profile`	whether parameters should be estimated sequentially using the idea of profile likelihood or not.
`errormsg`	whether or not the error message should be print.
`covonly`	estimate the covariance structure only, and use given mean.
`optim_method`	optimization method, choose "default" or "BFGS"(vmmin in R).

See Also

`acd_estimation` for joint mean covariance model fitting based on ACD, `hpc_estimation` for joint mean covariance model fitting based on HPC.

`meanplot` *Plot Fitted Mean Curves*

Description

plot fitted mean curves

Usage

```
meanplot(object)
```

Arguments

`object`	a fitted joint mean covariance model of class "jmcmMod", i.e., typically the result of jmcm().

Examples

```
cattleA <- cattle[cattle$group=='A', ]
fit.mcd <- jmcm(weight | id | I(ceiling(day/14 + 1)) ~ 1 | 1, data=cattleA,
  triple = c(8, 3, 4), cov.method = 'mcd')
meanplot(fit.mcd)
```

`modular` *Modular Functions for Joint Mean Covariance Model Fits*

Description

Modular Functions for joint mean covariance model fits

Usage

```
ldFormula(formula, data = NULL, triple = c(3, 3, 3), cov.method = c("mcd",
  "acd", "hpc"), optim.method = c("default", "BFGS"),
  control = jmcmControl(), start = NULL)

optimizeJmcm(m, Y, X, Z, W, time, cov.method, optim.method, control, start)

mkJmcmMod(opt, args, triple, cov.method, optim.method, mc)
```

Arguments

`formula`	a two-sided linear formula object describing the covariates for both the mean and covariance matrix part of the model, with the response, the corresponding subject id and measurement time on the left of a operator~, divided by vertical bars ("|").
`data`	a data frame containing the variables named in formula.
`triple`	an integer vector of length three containing the degrees of the three polynomial functions for the mean structure, the log innovation -variances and the autoregressive or moving average coefficients when 'mcd' or 'acd' is specified for cov.method. It refers to the degree for the mean structure, variances and angles when 'hpc' is specified for cov.method.
`cov.method`	covariance structure modelling method, choose 'mcd' (Pourahmadi 1999), 'acd' (Chen and Dunson 2013) or 'hpc' (Zhang et al. 2015).
`optim.method`	optimization method, choose 'default' or 'BFGS' (vmmin in R)
`control`	a list (of correct class, resulting from jmcmControl()) containing control parameters, see the *jmcmControl documentation for details.
`start`	starting values for the parameters in the model.
`m`	an integer vector of number of measurements for each subject.
`Y`	a vector of responses for all subjects.
`X`	model matrix for mean structure model.
`Z`	model matrix for the diagonal matrix.
`W`	model matrix for the lower triangular matrix.
`time`	a vector of time from the data.
`opt`	optimized results returned by optimizeJmcm.
`args`	arguments returned by ldFormula.
`mc`	matched call from the calling function.

`regressogram` *Plot Sample Regressograms and Fitted Curves*

Description

Plot the sample regressograms based on the sample covariance matrix and superimpose the corresponding fitted curves to check the model fitting when the longitudinal dataset is balanced.

Usage

```
regressogram(object, time)
```

Arguments

`object`	a fitted joint mean covariance model of class "jmcmMod", i.e., typically the result of jmcm().
`time`	a vector of obeservation time points

Examples

```
cattleA <- cattle[cattle$group=='A', ]
fit.mcd <- jmcm(weight | id | I(ceiling(day/14 + 1)) ~ 1 | 1, data=cattleA,
  triple = c(8, 3, 4), cov.method = 'mcd')
regressogram(fit.mcd, time = 1:11)
```

`show,jmcmMod-method` *Print information for jmcmMod-class*

Description

Print information for jmcmMod-class

Usage

```
## S4 method for signature 'jmcmMod'
show(object)
```

Arguments

`object`	a fitted joint mean covariance model of class "jmcmMod", i.e., typically the result of jmcm().

Index