Package ‘refitME’

July 23, 2025

Title Measurement Error Modelling using MCEM

Version 1.3.1

Description Fits measurement error models using Monte Carlo Expectation Maximization (MCEM). For specific details on the methodology, see: Greg C. G. Wei & Martin A. Tanner (1990) A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms, Journal of the American Statistical Association, 85:411, 699-704 <doi:10.1080/01621459.1990.10474930> For more examples on measurement error modelling using MCEM, see the 'RMarkdown' vignette: ``'refitME' R-package tutorial''.

Depends R (>= 4.4.0)

Imports MASS, mgcv, VGAM, VGAMdata, caret, expm, mvtnorm, sandwich, stats, dplyr, scales

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Jakub Stoklosa [aut, cre] (ORCID:
<`https://orcid.org/0000-0002-6523-4524`>),
Wenhan Hwang [aut, ctb],
David Warton [aut, ctb]

Maintainer Jakub Stoklosa <`j.stoklosa@unsw.edu.au`>

Repository CRAN

Date/Publication 2025-04-13 05:00:02 UTC

Contents

`anova.refitME` *An ANOVA function for fitted* `refitME` *objects*

Description

An ANOVA function for fitted `refitME` objects.

Usage

```
## S3 method for class 'refitME'
anova(object, ..., dispersion = NULL, test = NULL)
```

Arguments

`object`	: fitted model objects of class `refitME`.
`...`	: further arguments passed through to `lm` or `glm`.
`dispersion`	: the dispersion parameter for the fitting family. By default it is obtained from the object(s).
`test`	: a character string, (partially) matching one of "`Chisq`", "`LRT`", "`Rao`", "`F`" or "`Cp`". See `stat.anova`.

Value

`anova.refitME` produces output identical to `anova.lm`, `anova.glm` or `anova.gam`.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

See Also

`anova`

`anova_MCEMfit_glm` *An ANOVA function for fitted* `MCEMfit_glm` *objects*

Description

An ANOVA function for fitted `MCEMfit_glm` objects.

Usage

```
anova_MCEMfit_glm(object, ..., dispersion = NULL, test = NULL)
```

Arguments

`object`	: fitted model objects of class `MCEMfit_glm`.
`...`	: further arguments passed through to `glm`.
`dispersion`	: the dispersion parameter for the fitting family. By default it is obtained from the object(s).
`test`	: a character string, (partially) matching one of `"Chisq"`, `"LRT"`, `"Rao"`, `"F"` or `"Cp"`.

Value

`anova_MCEMfit_glm` produces output identical to `anova.glm`.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

See Also

`anova.glm`

`Corymbiaeximiadata` *The Corymbia eximia presence-only data set*

Description

Data set consisting of presence-only records for the plant species *Corymbia eximia*, site coordinates 5 covariates for each site.

Usage

```
Corymbiaeximiadata
```

Format

A data set that contains: 8 columns with 86,316 observations (or sites). The columns are defined as follows:

`X` Longitude coordinate.

`Y` Latitude coordinate.

`FC` Recorded number of fire counts for each site.

`MNT` Recorded minimum temperatures for each site.

`MXT` Recorded maximum temperature for each site.

`Rain` Recorded rainfall for each site.

`D.Main` Recorded distance from nearest major road.

`Y.obs` Presences for the plant species *Corymbia eximia* for each site.

Source

See Renner and Warton (2013) for full details on the data and study.

References

Renner, I. W. and Warton, D. I. (2013). Equivalence of MAXENT and Poisson point process models for species distribution modeling in ecology. *Biometrics*, **69**, 274–281.

Examples

```
# Load the data.

data(Corymbiaeximiadata)
```

`Framinghamdata` *The Framingham heart study data set*

Description

Data set consisting of records of male patients with coronary heart disease collected from the Framingham heart study. The `Framinghamdata` data consists of binary responses and four predictor variables collected on ‘n = 1615‘ patients.

Usage

```
Framinghamdata
```

Format

A data set that contains: 5 columns with 1,615 observations. The columns are defined as follows:

`Y` Response indicator (binary variable) of first evidence of CHD status of patient.

`z1` Serum cholesterol level of patient.

`z2` Age of patient.

`z3` Smoking indicator - whether the patient smokes.

`w1` Systolic blood pressure (SBP) of patient - this is the error contaminated variable, calculated from mean scores. The measurement error is 0.00630, see pp. 112 of Carroll *et al.* (2006).

Source

See Carroll *et al.* (2006) for full details on the data and study. Also, see `https://github.com/JakubStats/refitME` for an RMarkdown vignette of an example that uses the data.

References

Carroll, R. J., Ruppert, D., Stefanski, L. A., and Crainiceanu, C. M. (2006). *Measurement Error in Nonlinear Models: A Modern Perspective.* 2nd Ed. London: Chapman & Hall/CRC.

Examples

```
# Load the data.

data(Framinghamdata)
```

`logLik.refitME` *Extract log-Likelihoods for* `refitME` *model objects*

Description

Extract log-Likelihoods for `refitME` model objects. This function subtracts the entropy term from the observed likelihood.

Usage

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

Arguments

`object`	: fitted model objects of class `refitME`.
`...`	: further arguments passed through to `lm` or `glm`.

Value

`logLik.refitME` produces identical output to `logLik` but for `refitME` model objects.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

See Also

`logLik`

`logLik_MCEMfit_lm` *Extract log-Likelihoods for* `MCEMfit_lm` *model objects*

Description

Extract log-Likelihoods for `MCEMfit_lm` model objects. This function subtracts the entropy term from the observed likelihood.

Usage

```
logLik_MCEMfit_lm(object, REML = FALSE, ...)
```

Arguments

`object`	: fitted model objects of class `MCEMfit_lm`.
`REML`	: an optional logical value. If `TRUE` the restricted log-likelihood is returned, else, if `FALSE`, the log-likelihood is returned. Defaults to `FALSE`.
`...`	: further arguments passed through to `lm`.

Value

`logLik_MCEMfit_lm` produces output identical to `logLik`.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

See Also

`logLik`

MCEMfit_CR *Function for fitting VGAM capture-recapture (CR) model using the MCEM algorithm*

Description

Function for fitting VGAM capture-recapture (CR) model using the MCEM algorithm where covariates have measurement error.

Usage

```
MCEMfit_CR(mod, sigma.sq.u, B = 50, epsilon = 1e-05, silent = FALSE)
```

Arguments

mod	: a vglm/vgam object (this is the naive CR model). Make sure the first p input predictor variables in the naive model are the selected error-contaminated variables.
sigma.sq.u	: measurement error (ME) variance. A scalar if there is only one error-contaminated predictor variable, otherwise this must be stored as a vector (of ME variances) or a matrix if the ME covariance matrix is known.
B	: the number of Monte Carlo replication values (default is set to 50).
epsilon	: a set convergence threshold (default is set to 0.00001).
silent	: if TRUE, the convergence message (which tells the user if the model has converged and reports the number of iterations required) is suppressed (default is set to FALSE).

Value

MCEMfit_CR returns model coefficient and population size estimates with standard errors and the effective sample size.

Warning

This function is still under development. Currently the function can only fit the CR model used in the manuscript. IT DOES NOT SUPPORT ALL VGAM families.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

Source

See https://github.com/JakubStats/refitME for an RMarkdown vignette with examples.

References

Stoklosa, J., Hwang, W-H., and Warton, D.I. **refitME**: Measurement Error Modelling using Monte Carlo Expectation Maximization in R.

See Also

MCEMfit_glm

Examples

```
# A VGAM example using the Prinia flaviventris capture-recapture data.

library(refitME)
library(VGAM)

data(Priniadata)

tau <- 17   # No. of capture occasions.
w1 <- Priniadata$w1 # Bird wing length predictor.

CR_naiv <- vglm(cbind(cap, noncap) ~ w1,
   VGAM::posbinomial(omit.constant = TRUE, parallel = TRUE ~ w1),
   data = Priniadata, trace = FALSE)

sigma.sq.u <- 0.37 # ME variance.

CR_MCEM <- refitME(CR_naiv, sigma.sq.u)

detach(package:VGAM)
```

MCEMfit_gam *Function for wrapping the MCEM algorithm on* gam *objects*

Description

Function for wrapping the MCEM algorithm on GAMs where predictors are subject to measurement error/error-in-variables.

Usage

```
MCEMfit_gam(
  mod,
  family,
  sigma.sq.u,
  B = 50,
  epsilon = 1e-05,
  silent = FALSE,
  ...
)
```

Arguments

`mod` : a `gam` object (this is the naive fitted model). Make sure the first p input predictor variables entered in the naive model are the specified error-contaminated variables. These p predictors also need the measurement error variance to be specified in `sigma.sq.u`, see below.

`family` : a specified family/distribution.

`sigma.sq.u` : measurement error (ME) variance. A scalar if there is only one error-contaminated predictor variable, otherwise this must be stored as a vector (of ME variances) or a matrix if the ME covariance matrix is known.

`B` : the number of Monte Carlo replication values (default is set to 50).

`epsilon` : convergence threshold (default is set to 0.00001).

`silent` : if `TRUE`, the convergence message (which tells the user if the model has converged and reports the number of iterations required) is suppressed (default is set to `FALSE`).

`...` : further arguments passed to `gam`.

Value

`MCEMfit_gam` returns the original naive fitted model object but coefficient estimates and the covariance matrix have been replaced with the final MCEM model fit. Standard errors and the effective sample size (which diagnose how closely the proposal distribution matches the posterior, see equation (2) of Stoklosa, Hwang and Warton) have also been included as outputs.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

Source

See `https://github.com/JakubStats/refitME` for an RMarkdown vignette with examples. With permission from Matt Wand, we have now made these data available in the refitME R-package.

References

Ganguli, B, Staudenmayer, J., and Wand, M. P. (2005). Additive models with predictors subject to measurement error. *Australian & New Zealand Journal of Statistics*, **47**, 193–202.

Stoklosa, J., Hwang, W-H., and Warton, D.I. **refitME**: Measurement Error Modelling using Monte Carlo Expectation Maximization in R.

See Also

`MCEMfit_glm`

Examples

```
# A GAM example using the air pollution data set from the SemiPar package.

library(refitME)
library(mgcv)
library(dplyr)

data(Milanmortdata)

dat.air <- sample_n(Milanmortdata, 100) # Takes a random sample of size 100.

Y <- dat.air[, 6]  # Mortality counts.

n <- length(Y)

z1 <- (dat.air[, 1])
z2 <- (dat.air[, 4])
z3 <- (dat.air[, 5])
w1 <- log(dat.air[, 9])  # The error-contaminated predictor (total suspended particles).

dat <- data.frame(cbind(Y, w1, z1, z2, z3))

gam_naiv <- gam(Y ~ s(w1), family = "poisson", data = dat)

sigma.sq.u <- 0.0915 # Measurement error variance.

B <- 10  # Consider increasing this if you want a more accurate answer.

gam_MCEM <- refitME(gam_naiv, sigma.sq.u, B)

plot(gam_MCEM, select = 1)

detach(package:mgcv)
```

MCEMfit_gen — *Function for fitting any likelihood-based model using the MCEM algorithm*

Description

Function for wrapping the MCEM algorithm on any likelihood-based model where predictors are subject to measurement error/error-in-variables.

Usage

```
MCEMfit_gen(
  mod,
  family,
  sigma.sq.u,
  B = 50,
  epsilon = 1e-05,
  silent = FALSE,
  theta.est = 1,
  shape.est = 1,
  ...
)
```

Arguments

`mod`	: a model object (this is the naive fitted model). Make sure the first p input predictor variables entered in the naive model are the specified error-contaminated variables. These p predictors also need the measurement error variance to be specified in `sigma.sq.u`, see below.
`family`	: a specified family/distribution.
`sigma.sq.u`	: measurement error (ME) variance. A scalar if there is only one error-contaminated predictor variable, otherwise this must be stored as a vector (of ME variances) or a matrix if the ME covariance matrix is known.
`B`	: the number of Monte Carlo replication values (default is set to 50).
`epsilon`	: a set convergence threshold (default is set to 0.00001).
`silent`	: if `TRUE`, the convergence message (which tells the user if the model has converged and reports the number of iterations required) is suppressed (default is set to `FALSE`).
`theta.est`	: an initial value for the dispersion parameter (this is required for fitting negative binomial models).
`shape.est`	: an initial value for the shape parameter (this is required for fitting gamma models).
`...`	: further arguments passed through to the function that was used to fit `mod`, that will be used in refitting. These need only be specified if making changes to the arguments as compared to the original call that produced `mod`.

Value

`MCEMfit_gen` returns the original naive fitted model object but coefficient estimates and residuals have been replaced with the final MCEM model fit. Standard errors are included and returned, if `mod` is a class of object accepted by the **sandwich** package (such as `glm`, `gam`, `survreg` and many more).

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

References

Carroll, R. J., Ruppert, D., Stefanski, L. A., and Crainiceanu, C. M. (2006). *Measurement Error in Nonlinear Models: A Modern Perspective.* 2nd Ed. London: Chapman & Hall/CRC.

Stoklosa, J., Hwang, W-H., and Warton, D.I. **refitME**: Measurement Error Modelling using Monte Carlo Expectation Maximization in R.

See Also

`MCEMfit_glm` and `MCEMfit_gam`

`MCEMfit_glm` — *Function for wrapping the MCEM algorithm on `lm` or `glm` objects*

Description

Function for wrapping the MCEM algorithm on GLMs where predictors are subject to measurement error/error-in-variables.

Usage

```
MCEMfit_glm(
  mod,
  family,
  sigma.sq.u,
  B = 50,
  epsilon = 1e-05,
  silent = FALSE,
  ...
)
```

Arguments

`mod` : a `lm/glm` object (this is the naive fitted model). Make sure the first p input predictor variables entered in the naive model are the specified error-contaminated variables. These p predictors also need the measurement error variance to be specified in `sigma.sq.u`, see below.

`family` : a specified family/distribution.

`sigma.sq.u` : measurement error (ME) variance. A scalar if there is only one error-contaminated predictor variable, otherwise this must be stored as a vector (of ME variances) or a matrix if the ME covariance matrix is known.

`B` : the number of Monte Carlo replication values (default is set to 50).

`epsilon` : a set convergence threshold (default is set to 0.00001).

`silent` : if `TRUE`, the convergence message (which tells the user if the model has converged and reports the number of iterations required) is suppressed (default is set to `FALSE`).

`...` : further arguments passed to `lm` or `glm`.

Value

`MCEMfit_glm` returns the naive fitted model object where coefficient estimates, the covariance matrix, fitted values, the log-likelihood, and residuals have been replaced with the final MCEM model fit. Standard errors and the effective sample size (which diagnose how closely the proposal distribution matches the posterior, see equation (2) of Stoklosa, Hwang and Warton) have also been included as outputs.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

Source

See `https://github.com/JakubStats/refitME` for an RMarkdown vignette with examples.

References

Carroll, R. J., Ruppert, D., Stefanski, L. A., and Crainiceanu, C. M. (2006). *Measurement Error in Nonlinear Models: A Modern Perspective.* 2nd Ed. London: Chapman & Hall/CRC.

Stoklosa, J., Hwang, W-H., and Warton, D.I. **refitME**: Measurement Error Modelling using Monte Carlo Expectation Maximization in R.

See Also

`MCEMfit_gam`

Examples

```
# A GLM example I - binary response data.

library(refitME)

data(Framinghamdata)

glm_naiv <- glm(Y ~ w1 + z1 + z2 + z3, x = TRUE, family = binomial, data = Framinghamdata)

# The error-contaminated predictor in this example is systolic blood pressure (w1).

sigma.sq.u <- 0.006295  # ME variance, as obtained from Carroll et al. (2006) monograph.

B <- 50 # The number of Monte Carlo replication values.

glm_MCEM <- refitME(glm_naiv, sigma.sq.u, B)
```

`Milanmortdata` *The Milan mortality data set*

Description

The `Milanmortdata` data frame has data on 3652 consecutive days (10 consecutive years: 1st January, 1980 to 30th December, 1989) for the city of Milan, Italy. Note that this data set was originally contained and available from the now discontinued SemiPar R-package. With the permission of Matt Wand we have made these data (now called Milanmortdata) available in the refitME R-package.

Usage

```
Milanmortdata
```

Format

This data frame contains the following columns:

day.num number of days since 31st December, 1979.

day.of.week 1 = Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday, 7 = Sunday.

holiday indicator of public holiday: 1 = public holiday, 0 = otherwise.

mean.temp mean daily temperature in degrees Celcius.

rel.humid relative humidity.

tot.mort total number of deaths.

resp.mort total number of respiratory deaths.

SO2 measure of sulphur dioxide level in ambient air.

TSP total suspended particles in ambient air.

Source

Vigotti, M.A., Rossi, G., Bisanti, L., Zanobetti, A. and Schwartz, J. (1996). Short term effect of urban air pollution on respiratory health in Milan, Italy, 1980-1989. *Journal of Epidemiology and Community Health*, **50**, S71-S75.

References

Ruppert, D., Wand, M.P. and Carroll, R.J. (2003). *Semiparametric Regression* Cambridge University Press.

Examples

```
# Load the data.

data(Milanmortdata)
pairs(Milanmortdata, pch = ".")
```

`Priniadata` *The yellow-bellied Prinia* Prinia flaviventris *capture-recapture data*

Description

Data set consisting of capture-recapture histories 164 uniquely captured birds across 17 weekly capture occasions. Bird wing lengths were also measured in the study.

Usage

```
Priniadata
```

Format

A data set that contains: 3 columns with 164 observations. The columns are defined as follows:

`wl` Bird wing lengths.

`cap` Number of times the individual was captured.

`noncap` Number of times the individual was not captured.

Source

See Hwang, Huang and Wang (2007) for full details on the data and study.

References

Hwang, W. H., Huang, S. Y. H., and Wang, C. (2007). Effects of measurement error and conditional score estimation in capture–recapture models. *Statistica Sinica*, **17**, 301-316.

Examples

```
# Load the data.

data(Priniadata)
```

`refitME` *A wrapper function for correcting measurement error in predictor variables via the MCEM algorithm*

Description

Function that extracts the fitted (naive) model object and wraps the MCEM algorithm to correct for measurement error/error-in-variables in predictors.

Usage

```
refitME(mod, sigma.sq.u, B = 50, epsilon = 1e-05, silent = FALSE, ...)
```

Arguments

`mod` : any (S3 class) fitted object that responds to the generic functions `family`, `model.frame`, `update` and `predict`, and accepts weighted observations via `weights`. The `mod` argument specifies the naive fitted model. Make sure the first p input predictor variables in the naive model are the selected error-contaminated predictors variables. Also, the `mod` argument allows `vlgm/vgam` (S4 class) model objects when using the `posbinomial` family – this is a specific function developed for fitting closed population capture–recapture models, see `MCEMfit_CR`.

`sigma.sq.u` : measurement error (ME) variance. A scalar if there is only one error-contaminated predictor variable, otherwise this must be stored as a vector (of known ME variances) or a matrix if the ME covariance matrix is known.

`B` : the number of Monte Carlo replication values (default is set 50).

`epsilon` : convergence threshold (default is set to 0.00001).

`silent` : if `TRUE`, the convergence message (which tells the user if the model has converged and reports the number of iterations required) is suppressed (default is set to `FALSE`).

`...` : further arguments passed through to the function that was used to fit `mod`, that will be used in refitting. These need only be specified if making changes to the arguments as compared to the original call that produced `mod`.

Value

`refitME` returns the naive fitted model object where coefficient estimates, the covariance matrix, fitted values, the log-likelihood, and residuals have been replaced with the final MCEM model fit. Standard errors are included and returned, if `mod` is a class of object accepted by the **sandwich** package (such as `glm`, `gam`, `survreg` and many more). The effective sample size (which diagnose how closely the proposal distribution matches the posterior, see equation (2) of Stoklosa, Hwang and Warton) have also been included as outputs.

Author(s)

Jakub Stoklosa, Wen-Han Hwang and David I. Warton.

Source

See `https://github.com/JakubStats/refitME` for an RMarkdown vignette with examples.

References

Carroll, R. J., Ruppert, D., Stefanski, L. A., and Crainiceanu, C. M. (2006). *Measurement Error in Nonlinear Models: A Modern Perspective.* 2nd Ed. London: Chapman & Hall/CRC.

Stoklosa, J., Hwang, W-H., and Warton, D.I. **refitME**: Measurement Error Modelling using Monte Carlo Expectation Maximization in R.

See Also

`MCEMfit_glm`, `MCEMfit_gam` and `MCEMfit_gen`

Examples

```
# A GLM example I - binary response data.

library(refitME)

data(Framinghamdata)

glm_naiv <- glm(Y ~ w1 + z1 + z2 + z3, x = TRUE, family = binomial, data = Framinghamdata)

# The error-contaminated predictor variable in this example is systolic blood pressure (w1).

sigma.sq.u <- 0.01259/2 # ME variance, as obtained from Carroll et al. (2006) monograph.

B <- 50  # The number of Monte Carlo replication values.

glm_MCEM <- refitME(glm_naiv, sigma.sq.u, B)
```

sqrt.na *Function that replaces NA with zero for a matrix*

Description

This function replaces NA with zero for a matrix.

Usage

```
## S3 method for class 'na'
sqrt(x)
```

Arguments

x : a matrix

Value

sqrt.na returns a matrix.

Author(s)

Jakub Stoklosa

wt.var *Function that calculates a weighted variance*

Description

This function that calculates a weighted variance for a given vector.

Usage

```
wt.var(x, w)
```

Arguments

x	: a vector of numerical data.
w	: a vector of equal length to x representing the weights.

Value

wt.var returns a single value from analysis requested.

Source

The developer of this function is Jeremy VanDerWal. See https://rdrr.io/cran/SDMTools/src/R/wt.mean.R

Examples

```
# Define simple data
x = 1:25 # Set of numbers.
wt = runif(25) # Some arbitrary weights.

# Display variances (unweighted and then weighted).
var(x)
wt.var(x, wt)
```

Index