

Package ‘misaem’

September 11, 2025

Type Package

Version 1.1.0

Title Linear Regression and Logistic Regression with Missing Covariates

Date 2025-09-11

Maintainer Julie Josse <julie.josse@inria.fr>

Depends R (>= 3.4.0)

Imports mvtnorm, stats, MASS, norm, methods, abind, glmnet

Suggests knitr, rmarkdown, mice

Description Estimate parameters of linear regression and logistic regression with missing covariates with missing data, perform model selection and prediction, using EM-type algorithms. Jiang W., Josse J., Lavielle M., TraumaBase Group (2020) <[doi:10.1016/j.csda.2019.106907](https://doi.org/10.1016/j.csda.2019.106907)>.

Encoding UTF-8

License GPL-3

URL <https://github.com/julierennes/misaem>

BugReports <https://github.com/julierennes/misaem/issues>

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Author Wei Jiang [aut],
Pavlo Mozharovskiy [ctb],
Imke Mayer [ctb],
Christophe Muller [ctb],
Julie Josse [aut, cre]

Repository CRAN

Date/Publication 2025-09-11 20:30:02 UTC

Contents

combinations	2
imputeEIIIP	3
initialize_df	3
likelihood_saem	4
log_reg	5
louis_lr_saem	6
miss.glm	7
miss.glm.control	8
miss.glm.fit	10
miss.glm.model.select	11
miss.lm	12
miss.lm.control	13
miss.lm.fit	14
miss.lm.model.select	15
predict.miss.glm	16
predict.miss.lm	17
print.miss.glm	18
print.miss.lm	19
print.summary.miss.glm	20
print.summary.miss.lm	20
summary.miss.glm	21
summary.miss.lm	22
Index	23

combinations	<i>combinations</i>
--------------	---------------------

Description

Given all the possible patterns of missingness.

Usage

```
combinations(p)
```

Arguments

p Dimension of covariates.

Value

A matrix containing all the possible missing patterns. Each row indicates a pattern of missingness. "1" means "observed", 0 means "missing".

Examples

```
comb = combinations(5)
```

imputeEllP	<i>Function for imputing single point for linear regression model</i>
------------	---

Description

Function for imputing single point for linear regression model

Usage

```
imputeEllP(point, Sigma.inv)
```

Arguments

point	A single observation containing missing values.
Sigma.inv	Inverse of estimated Σ .

Value

Imputed observation.

initialize_df	<i>Initialize Missing Data in Data Frame</i>
---------------	--

Description

This function initializes missing values in a data frame using mean imputation.

Usage

```
initialize_df(x, method = "mean", seed = NULL)
```

Arguments

x	A data frame or matrix containing missing values to be initialized. If a matrix is provided, it will be converted to a data frame internally.
method	A character string specifying the initialization method. Currently, only "mean" is supported.
seed	An optional integer to set the random seed for reproducibility. Not used in the current implementation.

Details

Missing values are replaced with the column means.

Value

A data frame or matrix with missing values initialized using the mean of each column.

likelihood_saem	<i>likelihood_saem</i>
-----------------	------------------------

Description

Used in main function miss.saem. Calculate the observed log-likelihood for logistic regression model with missing data, using Monte Carlo version of Louis formula.

Usage

```
likelihood_saem(
  beta,
  mu,
  Sigma,
  Y,
  X.obs,
  rindic = as.matrix(is.na(X.obs)),
  whichcolXmissing = (1:ncol(rindic))[apply(rindic, 2, sum) > 0],
  mc.size = 2
)
```

Arguments

beta	Estimated parameter of logistic regression model.
mu	Estimated parameter μ .
Sigma	Estimated parameter Σ .
Y	Response vector $N \times 1$
X.obs	Design matrix with missingness $N \times p$
rindic	Missing pattern of X.obs. If a component in X.obs is missing, the corresponding position in rindic is 1; else 0.
whichcolXmissing	The column index in covariate containing at least one missing observation.
mc.size	Monte Carlo sampling size.

Value

Observed log-likelihood.

Examples

```
# Generate dataset
N <- 50 # number of subjects
p <- 3 # number of explanatory variables
mu.star <- rep(0,p) # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1, 0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%%chol(Sigma.star) +
  matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# Observed log-likelihood
ll_obs = likelihood_saem(beta.true,mu.star,Sigma.star,y,X.obs)
```

log_reg

*log_reg***Description**

Calculate the likelihood or log-likelihood for one observation of logistic regression model .

Usage

```
log_reg(y, x, beta, iflog = TRUE)
```

Arguments

y	Response value (0 or 1).
x	Covariate vector of dimension $p \times 1$.
beta	Estimated parameter of logistic regression model.
iflog	If TRUE, log_reg calculate the log-likelihood; else likelihood.

Value

Likelihood or log-likelihood.

louis_lr_saem	<i>louis_lr_saem</i>
---------------	----------------------

Description

Used in main function `miss.saem`. Calculate the variance of estimated parameters for logistic regression model with missing data, using Monte Carlo version of Louis formula.

Usage

```
louis_lr_saem(
  beta,
  mu,
  Sigma,
  Y,
  X.obs,
  pos_var = 1:ncol(X.obs),
  rindic = as.matrix(is.na(X.obs)),
  whichcolXmissing = (1:ncol(rindic))[apply(rindic, 2, sum) > 0],
  mc.size = 2
)
```

Arguments

<code>beta</code>	Estimated parameter of logistic regression model.
<code>mu</code>	Estimated parameter μ .
<code>Sigma</code>	Estimated parameter Σ .
<code>Y</code>	Response vector $N \times 1$
<code>X.obs</code>	Design matrix with missingness $N \times p$
<code>pos_var</code>	Index of selected covariates.
<code>rindic</code>	Missing pattern of <code>X.obs</code> . If a component in <code>X.obs</code> is missing, the corresponding position in <code>rindic</code> is 1; else 0.
<code>whichcolXmissing</code>	The column index in covariate containing at least one missing observation.
<code>mc.size</code>	Monte Carlo sampling size.

Value

Variance of estimated β .

Examples

```
# Generate dataset
N <- 50 # number of subjects
p <- 3 # number of explanatory variables
mu.star <- rep(0,p) # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1, 0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%%chol(Sigma.star) +
  matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# Louis formula to obtain variance of estimates
V_obs = louis_lr_saem(beta.true,mu.star,Sigma.star,y,X.obs)
```

miss.glm

Statistical Inference for Logistic Regression Models with Missing Values

Description

This function is used to perform statistical inference for logistic regression model with missing values, by algorithm SAEM.

Usage

```
miss.glm(formula, data, control = list(...), ...)
```

Arguments

formula	an object of class " formula " : a symbolic description of the logistic regression model to be fitted.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which miss.glm is called.
control	a list of parameters for controlling the fitting process. For miss.glm.fit this is passed to miss.glm.control .
...	arguments to be used to form the default control argument if it is not supplied directly.

Value

An object of class "miss.glm": a list with following components:

coefficients	Estimated β .
ll	Observed log-likelihood.
var.covar	Variance-covariance matrix for estimated parameters.
s.err	Standard error for estimated parameters.
mu.X	Estimated μ .
Sig.X	Estimated Σ .
call	the matched call.
formula	the formula supplied.

Examples

```
# Generate dataset
N <- 100 # number of subjects
p <- 3   # number of explanatory variables
mu.star <- rep(0,p) # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1, 0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%%chol(Sigma.star) +
  matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)

# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

df.obs = data.frame(y,X.obs)

# SAEM
miss.list = miss.glm(y~., data=df.obs, print_iter=FALSE,seed=100)
print(miss.list)
print(summary(miss.list))
summary(miss.list)$coef
```

miss.glm.control

Auxiliary for Controlling Fitting

Description

Auxiliary function for `miss.glm` fitting. Typically only used internally by `miss.glm.fit`.

Usage

```
miss.glm.control(
  maxruns = 500,
  tol_em = 1e-07,
  nmcmc = 2,
  tau = 1,
  k1 = 50,
  subsets = NA,
  seed = NA,
  print_iter = TRUE,
  var_cal = TRUE,
  ll_obs_cal = TRUE,
  alpha = 0,
  lambda = 0,
  save_trace = FALSE,
  init_method = "mean",
  ignore_errors = FALSE
)
```

Arguments

maxruns	maximum number of iterations. The default is maxruns = 500.
tol_em	the tolerance to stop SAEM. The default is tol_em = 1e-7.
nmcmc	the MCMC length. The default is nmcmc = 2.
tau	rate τ in the step size $(k - k_1)^{-\tau}$. The default is tau = 1.
k1	number of first iterations k_1 in the step size $(k - k_1)^{-\tau}$. The default is k1=50.
subsets	Index of selected covariates if any. The default is all the covariates.
seed	an integer as a seed set for the random generator.
print_iter	logical indicating if output should be produced for each iteration.
var_cal	logical indicating if the variance of estimated parameters should be calculated.
ll_obs_cal	logical indicating if the observed log-likelihood should be calculated.
alpha	elastic-net mixing parameter, with $0 \leq \alpha \leq 1$. The default is alpha = 0 (ridge).
lambda	non-negative regularization parameter. The default is lambda = 0 (no regularization).
save_trace	logical indicating if the trace of beta values over iterations should be saved.
init_method	method to initialize the missing data. "mean", its default value, is the only option currently implemented.
ignore_errors	logical indicating if errors during the fitting process should be ignored (TRUE, raising a warning instead) or stop the execution (FALSE). Default is FALSE.

Value

A list with components named as the arguments.

Examples

```
## For examples see example(miss.glm)
```

miss.glm.fit

Fitting Logistic Regression Models with Missing Values

Description

This function is used inside `miss.glm` to fit logistic regression model with missing values, by algorithm SAEM.

Usage

```
miss.glm.fit(x, y, control = list())
```

Arguments

<code>x</code>	design matrix with missingness $N \times p$.
<code>y</code>	response vector $N \times 1$.
<code>control</code>	a list of parameters for controlling the fitting process. For <code>miss.glm.fit</code> this is passed to <code>miss.glm.control</code> .

Value

a list with following components:

<code>coefficients</code>	Estimated β .
<code>ll</code>	Observed log-likelihood.
<code>var.covar</code>	Variance-covariance matrix for estimated parameters.
<code>s.err</code>	Standard error for estimated parameters.
<code>mu.X</code>	Estimated μ .
<code>Sig.X</code>	Estimated Σ .

Examples

```
## For examples see example(miss.glm)
```

miss.glm.model.select *miss.glm.model.select*

Description

Model selection for the logistic regression model with missing data.

Usage

```
miss.glm.model.select(Y, X, seed = NA)
```

Arguments

Y	Binary response vector $N \times 1$
X	Design matrix with missingness $N \times p$
seed	An integer as a seed set for the random generator. The default value is 200.

Value

An object of class "miss.glm".

Examples

```
# Generate dataset
N <- 40 # number of subjects
p <- 3  # number of explanatory variables
mu.star <- rep(0,p) # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1, 0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%%chol(Sigma.star) +
  matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
Y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X <- X.complete
X[patterns] <- NA
# model selection for SAEM
miss.model = miss.glm.model.select(Y,X,seed=100)
print(miss.model)
```

miss.lm

*Statistical Inference for Linear Regression Models with Missing Values***Description**

This function is used to perform statistical inference for linear regression model with missing values, by algorithm EM.

Usage

```
miss.lm(formula, data, control = list(...), ...)
```

Arguments

formula	an object of class " formula " : a symbolic description of the linear regression model to be fitted.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>miss.lm</code> is called.
control	a list of parameters for controlling the fitting process. For <code>miss.lm.fit</code> this is passed to miss.lm.control .
...	arguments to be used to form the default control argument if it is not supplied directly.

Value

An object of class "miss.lm": a list with following components:

coefficients	Estimated β .
ll	Observed log-likelihood.
s.resid	Estimated standard error for residuals.
s.err	Standard error for estimated parameters.
mu.X	Estimated μ .
Sig.X	Estimated Σ .
call	the matched call.
formula	the formula supplied.

Examples

```
# Generate complete data
set.seed(1)
mu.X <- c(1, 1)
Sigma.X <- matrix(c(1, 1, 1, 4), nrow = 2)
n <- 50
p <- 2
```

```

X.complete <- matrix(rnorm(n*p), nrow=n)%*%chol(Sigma.X) +
  matrix(rep(mu.X,n), nrow=n, byrow = TRUE)
b <- c(2, 3, -1)
sigma.eps <- 0.25
y <- cbind(rep(1, n), X.complete) %*% b + rnorm(n, 0, sigma.eps)

# Add missing values
p.miss <- 0.10
patterns <- runif(n*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# Estimate regression using EM
df.obs = data.frame(y,X.obs)
miss.list = miss.lm(y~., data=df.obs)
print(miss.list)
print(summary(miss.list))
summary(miss.list)$coef

```

miss.lm.control

Auxiliary for Controlling Fitting

Description

Auxiliary function for [miss.lm](#) fitting. Typically only used internally by [miss.lm.fit](#).

Usage

```
miss.lm.control(maxruns = 500, tol_em = 1e-07, print_iter = TRUE)
```

Arguments

maxruns	maximum number of iterations. The default is maxruns = 500.
tol_em	the tolerance to stop EM. The default is tol_em = 1e-4.
print_iter	logical indicating if output should be produced for each iteration.

Value

A list with components named as the arguments.

Examples

```
## For examples see example(miss.lm)
```

miss.lm.fit

*Fitting Linear Regression Model with Missing Values***Description**

This function is used inside `miss.lm` to fit linear regression model with missing values, by EM algorithm.

Usage

```
miss.lm.fit(x, y, control = list())
```

Arguments

<code>x</code>	design matrix with missingness $N \times p$.
<code>y</code>	response vector $N \times 1$.
<code>control</code>	a list of parameters for controlling the fitting process. For <code>miss.lm.fit</code> this is passed to miss.lm.control .

Value

a list with following components:

<code>coefficients</code>	Estimated β .
<code>ll</code>	Observed log-likelihood.
<code>s.resid</code>	Estimated standard error for residuals.
<code>s.err</code>	Standard error for estimated parameters.
<code>mu.X</code>	Estimated μ .
<code>Sig.X</code>	Estimated Σ .

Examples

```
## For examples see example(miss.lm)
```

miss.lm.model.select *miss.lm.model.select*

Description

Model selection for the linear regression model with missing data.

Usage

```
miss.lm.model.select(Y, X)
```

Arguments

Y	Response vector $N \times 1$
X	Design matrix with missingness $N \times p$

Value

An object of class "miss.lm".

Examples

```
# Generate complete data
set.seed(1)
mu.X <- c(1, 1)
Sigma.X <- matrix(c(1, 1, 1, 4), nrow = 2)
n <- 50
p <- 2
X.complete <- matrix(rnorm(n*p), nrow=n)%*%chol(Sigma.X) +
  matrix(rep(mu.X,n), nrow=n, byrow = TRUE)
b <- c(2, 0, -1)
sigma.eps <- 0.25
y <- cbind(rep(1, n), X.complete) %*% b + rnorm(n, 0, sigma.eps)

# Add missing values
p.miss <- 0.10
patterns <- runif(n*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# model selection
miss.model = miss.lm.model.select(y, X.obs)
print(miss.model)
```

predict.miss.glm	<i>Prediction on test with missing values for the logistic regression model.</i>
------------------	--

Description

Prediction on test with missing values for the logistic regression model.

Usage

```
## S3 method for class 'miss.glm'
predict(object, newdata = NULL, seed = NA, method = "map", mc.size = 100, ...)
```

Arguments

object	a fitted object of class inheriting from "miss.glm".
newdata	a data frame in which to look for variables with which to predict. It can contain missing values.
seed	An integer as a seed set for the random generator.
method	The name of method to deal with missing values in test set. It can be 'map' (maximum a posteriori) or 'impute' (imputation by conditional expectation). Default is 'map'.
mc.size	The number of Monte Carlo samples to use when method is 'map'. Default is 100.
...	Further arguments passed to or from other methods.

Value

pr.saem	The prediction result for logistic regression: the probability of response $y=1$.
---------	--

Examples

```
# Generate dataset
N <- 100 # number of subjects
p <- 3   # number of explanatory variables
mu.star <- rep(0,p) # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1, 0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%*%chol(Sigma.star) +
  matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)

# Generate missingness
p.miss <- 0.10
```

```

patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

df.obs = data.frame(y,X.obs)

# SAEM
miss.list = miss.glm(y~., data=df.obs, print_iter=FALSE,seed=100)

# Generate new dataset for prediction
Nt <- 20
Xt <- matrix(rnorm(Nt*p), nrow=Nt)%*%chol(Sigma.star)+
  matrix(rep(mu.star,Nt), nrow=Nt, byrow = TRUE)
# Generate missingness in new dataset
patterns <- runif(Nt*p)<p.miss
Xt.obs <- Xt
Xt.obs[patterns] <- NA

# Prediction with missing values
miss.prob = predict(miss.list, data.frame(Xt.obs), method='map')
print(miss.prob)

```

predict.miss.lm

Prediction on test with missing values for the linear regression model.

Description

Prediction on test with missing values for the linear regression model.

Usage

```

## S3 method for class 'miss.lm'
predict(object, newdata = NULL, seed = NA, ...)

```

Arguments

object	a fitted object of class inheriting from "miss.lm".
newdata	a data frame in which to look for variables with which to predict. It can contain missing values.
seed	An integer as a seed set for the random generator.
...	Further arguments passed to or from other methods.

Value

pr.y	The prediction result for linear regression.
------	--

Examples

```

# Generate complete data
set.seed(1)
mu.X <- c(1, 1)
Sigma.X <- matrix(c(1, 1, 1, 4), nrow = 2)
n <- 50 # train set size
p <- 2 # number of covariates
X.complete <- matrix(rnorm(n*p), nrow=n)%*%chol(Sigma.X) +
  matrix(rep(mu.X,n), nrow=n, byrow = TRUE)
b <- c(2, 3, -1)
sigma.eps <- 0.25
y <- cbind(rep(1, n), X.complete) %*% b +
  rnorm(n, 0, sigma.eps)

# Add missing values
p.miss <- 0.10
patterns <- runif(n*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA
# Estimate regression using EM
df.obs = data.frame(y ,X.obs)
miss.list = miss.lm(y~., data=df.obs)

# Generate new dataset for prediction
nt <- 20
Xt <- matrix(rnorm(nt*p), nrow=nt)%*%chol(Sigma.X)+
  matrix(rep(mu.X,nt), nrow=nt, byrow = TRUE)
# Generate missingness in new dataset
patterns <- runif(nt*p)<p.miss
Xt.obs <- Xt
Xt.obs[patterns] <- NA

# Prediction with missing values
miss.pred = predict(miss.list, data.frame(Xt.obs))
print(miss.pred)

```

```
print.miss.glm
```

```
Print miss.glm
```

Description

Print results for class `miss.glm`.

Usage

```

## S3 method for class 'miss.glm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

```

Arguments

`x` an object of class "miss.glm", usually, a result of a call to [miss.glm](#).
`digits` minimal number of significant digits.
`...` further arguments passed to or from other methods.

Value

No return value, called for coefficient and standard error estimates print.

Examples

```
## For examples see example(miss.glm)
```

print.miss.lm	<i>Print miss.lm</i>
---------------	----------------------

Description

Print results for class `miss.lm`.

Usage

```
## S3 method for class 'miss.lm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x` an object of class "miss.lm", usually, a result of a call to [miss.lm](#).
`digits` minimal number of significant digits.
`...` further arguments passed to or from other methods.

Value

No return value, called for coefficient and standard error estimates print.

Examples

```
## For examples see example(miss.lm)
```

```
print.summary.miss.glm
```

Print Summary of miss.glm

Description

Print results for class `summary.miss.glm`.

Usage

```
## S3 method for class 'summary.miss.glm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

<code>x</code>	an object of class <code>"summary.miss.glm"</code> , usually, a result of a call to summary.miss.glm .
<code>digits</code>	minimal number of significant digits.
<code>...</code>	further arguments passed to or from other methods.

Value

No return value, called for summary print.

Examples

```
## For examples see example(miss.glm)
```

```
print.summary.miss.lm
```

Print Summary of miss.lm

Description

Print results for class `summary.miss.lm`.

Usage

```
## S3 method for class 'summary.miss.lm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

<code>x</code>	an object of class <code>"summary.miss.lm"</code> , usually, a result of a call to summary.miss.lm .
<code>digits</code>	minimal number of significant digits.
<code>...</code>	further arguments passed to or from other methods.

Value

No return value, called for summary print.

Examples

```
## For examples see example(miss.lm)
```

summary.miss.glm	<i>Summarizing Fits for miss.glm</i>
------------------	--------------------------------------

Description

Summary for class `miss.glm`.

Usage

```
## S3 method for class 'miss.glm'
summary(object, ...)
```

Arguments

<code>object</code>	an object of class "miss.glm", usually, a result of a call to miss.glm .
<code>...</code>	Further arguments passed to or from other methods.

Value

An object of class "summary.miss.glm", a list with following components:

<code>coefficients</code>	The matrix of coefficients and standard errors
<code>loglikelihood</code>	Observed log-likelihood.
<code>call</code>	the component from object.
<code>formula</code>	the component from object.

Examples

```
## For examples see example(miss.glm)
```

summary.miss.lm	<i>Summarizing Fits for miss.lm</i>
-----------------	-------------------------------------

Description

Summary for class `miss.lm`.

Usage

```
## S3 method for class 'miss.lm'  
summary(object, ...)
```

Arguments

<code>object</code>	an object of class "miss.lm", usually, a result of a call to miss.lm .
<code>...</code>	Further arguments passed to or from other methods.

Value

An object of class "summary.miss.lm", a list with following components:

<code>coefficients</code>	The matrix of coefficients and standard errors.
<code>loglikelihood</code>	Observed log-likelihood.
<code>call</code>	the component from object.
<code>formula</code>	the component from object.

Examples

```
## For examples see example(miss.lm)
```

Index

combinations, [2](#)

formula, [7](#), [12](#)

imputeEllP, [3](#)

initialize_df, [3](#)

likelihood_saem, [4](#)

log_reg, [5](#)

louis_lr_saem, [6](#)

miss.glm, [7](#), [8](#), [19](#), [21](#)

miss.glm.control, [7](#), [8](#), [10](#)

miss.glm.fit, [8](#), [10](#)

miss.glm.model.select, [11](#)

miss.lm, [12](#), [13](#), [19](#), [22](#)

miss.lm.control, [12](#), [13](#), [14](#)

miss.lm.fit, [13](#), [14](#)

miss.lm.model.select, [15](#)

predict.miss.glm, [16](#)

predict.miss.lm, [17](#)

print.miss.glm, [18](#)

print.miss.lm, [19](#)

print.summary.miss.glm, [20](#)

print.summary.miss.lm, [20](#)

summary.miss.glm, [20](#), [21](#)

summary.miss.lm, [20](#), [22](#)