Package ‘bclogit’

March 1, 2026

Title Conditional Logistic Regression

Version 1.1

Author Adam Kapelner [aut, cre] (ORCID: <`https://orcid.org/0000-0001-5985-6792`>), Jacob Tennenbaum [aut] (ORCID: <`https://orcid.org/0009-0006-1408-5035`>)

Maintainer Adam Kapelner <`kapelner@qc.cuny.edu`>

Description Performs inference for Bayesian conditional logistic regression with informative priors built from the concordant pair data. We include many options to build the priors. And we include many options during the inference step for estimation, testing and confidence set creation. For details, see Kapelner and Tennenbaum (2026) ``Improved Conditional Logistic Regression using Information in Concordant Pairs with Software'' <doi:10.48550/arXiv.2602.08212>.

SystemRequirements GNU make

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.0), Rcpp (>= 1.0.14)

Imports checkmate, coda, fastLogisticRegressionWrap, geepack, glmmTMB, methods, RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.6.0)

Suggests testthat (>= 3.0.0), survival, ggdist, data.table

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

URL `https://github.com/Tennenbaum-J/bclogit_package_and_paper_repo`

BugReports `https://github.com/Tennenbaum-J/bclogit_package_and_paper_repo/issues`

RoxygenNote 7.3.3

Biarch true

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-03-01 10:00:08 UTC

Contents

`bclogit-package` *Bayesian Conditional Logistic Regression with Concordant Pairs*

Description

bclogit

Details

Fits a conditional logistic regression model to incidence data. Allows for use of the concordant pairs in the fitting.

Author(s)

`Jacob Tennenbaum \email{Jacob.Tennenbaum51@qmail.cuny.edu}`

`Adam Kapelner \email{kapelner@qc.cuny.edu}`

References

`Jacob Tennenbaum and Adam Kapelner (2026). "Improved Conditional Logistic Regression using Information`

See Also

Useful links:

- https://github.com/Tennenbaum-J/bclogit_package_and_paper_repo
- Report bugs at https://github.com/Tennenbaum-J/bclogit_package_and_paper_repo/issues

`bclogit.formula` *Initialize a new bclogit model*

Description

This function fits a Bayesian conditional logistic regression model, incorporating information from concordant pairs to improve estimation.

Usage

```
## S3 method for class 'formula'
bclogit(
  formula,
  data,
  treatment = NULL,
  strata = NULL,
  subset = NULL,
  na.action = NULL,
  concordant_method = "GLM",
  prior_type = "Naive",
  chains = 4,
  return_raw_stan_output = FALSE,
  prior_variance_treatment = 100,
  stan_refresh = 0,
  ...
)

bclogit(
  formula,
  data,
  treatment = NULL,
  strata = NULL,
  subset = NULL,
  na.action = NULL,
  concordant_method = "GLM",
  prior_type = "Naive",
  chains = 4,
  return_raw_stan_output = FALSE,
  prior_variance_treatment = 100,
  stan_refresh = 0,
  ...
)

## Default S3 method:
bclogit(
  formula = NULL,
  data = NULL,
```

```
  treatment = NULL,
  strata = NULL,
  subset = NULL,
  na.action = NULL,
  concordant_method = "GLM",
  prior_type = "Naive",
  chains = 4,
  return_raw_stan_output = FALSE,
  prior_variance_treatment = 100,
  stan_refresh = 0,
  ...,
  y = NULL,
  X = NULL,
  treatment_name = NULL,
  call = NULL
)
```

Arguments

`formula`	For the formula method, a symbolic description of the model to be fitted.
`data`	A data.frame, data.table, or model.matrix containing the variables (optional for formula method).
`treatment`	Optional vector specifying the treatment variable (required for default method, or can be specified in formula method).
`strata`	Vector specifying the strata (matched pairs).
`subset`	An optional vector specifying a subset of observations.
`na.action`	A function which indicates what should happen when the data contain NAs.
`concordant_method`	The method to use for fitting the concordant pairs and reservoir. Options are "GLM", "GEE", and "GLMM".
`prior_type`	The type of prior to use for the discordant pairs. Options are "Naive", "G prior", "PMP", and "Hybrid".
`chains`	Number of chains for Stan sampling. Default is 4.
`return_raw_stan_output`	Logical; if `TRUE`, the raw Stan posterior samples (iterations x chains x parameters) are stored in the returned object. Default `FALSE`.
`prior_variance_treatment`	Prior variance for the treatment coefficient in the covariance matrix `Sigma_con`. Default is 100.
`stan_refresh`	How often Stan reports sampling progress (in iterations). Default is 0 (silent). Set to a positive integer (e.g., 1 or 100) to see progress.
`...`	Additional arguments passed to `rstan::sampling` (e.g., `iter`, `warmup`, `thin`, `seed`, `control`).
`y`	For the default method, a binary (0,1) vector containing the response of each subject.

`X`	A data.frame, data.table, or model.matrix containing the variables.
`treatment_name`	Optional string name for the treatment variable.
`call`	Optional call object to store in the result.

Value

An object of class `"bclogit"`.

A list of class `bclogit` containing:

`coefficients`	Estimated coefficients (posterior means).
`var`	Variance-covariance matrix of coefficients.
`model`	The fitted Stan model object.
`posterior_samples`	Raw posterior samples as a 3D array (iterations x chains x parameters) from `rstan::extract(model, permuted = FALSE)`. Only populated when `return_raw_stan_output = TRUE`; `NULL` otherwise.
`concordant_model`	The fitted model object for the concordant pairs/reservoir (GLM/GEE/GLMM).
`matched_data`	The processed matched pairs data from the premodeling step.
`prior_info`	Information about the prior derived from concordant pairs.
`call`	The function call.
`terms`	The model terms.
`num_discordant`	Number of discordant pairs used.
`num_concordant`	Number of concordant pairs/reservoir entries used.

A list of class `"bclogit"` containing:

`coefficients`	Estimated coefficients (posterior means).
`var`	Variance-covariance matrix of the coefficients (posterior covariance).
`model`	The fitted Stan model object for the discordant pairs.
`posterior_samples`	Raw posterior samples as a 3D array (iterations x chains x parameters) from `rstan::extract(model, permuted = FALSE)`. Only populated when `return_raw_stan_output = TRUE`; `NULL` otherwise.
`concordant_model`	The fitted model object for the concordant pairs (GLM, GEE, or GLMM).
`matched_data`	The processed matched pairs data from the C++ pre-modeling step.
`prior_info`	A list with elements `mu` (prior mean vector) and `Sigma` (prior covariance matrix) derived from the concordant pairs model.
`call`	The function call.
`terms`	The model terms.
`xlevels`	Factor level information (always `NULL` for this method).
`n`	Total number of observations.

`num_discordant`	Number of discordant pairs used for fitting.
`num_concordant`	Number of concordant pairs used for the prior.
`X_model_matrix_col_names`	Column names of the covariate model matrix.
`treatment_name`	Name of the treatment variable.

Methods (by class)

- `bclogit(formula)`: Formula method
- `bclogit(default)`: Default method for matrix/data input.

Numerical Stability in GEE Pre-modeling

When `concordant_method = "GEE"`, the function applies a layered set of safeguards to ensure the prior covariance `Sigma_con` passed to Stan is valid (positive definite, symmetric, and free of `NA` values).

Naive (model-based) covariance GEE's default covariance is the sandwich (robust) estimator, which can produce degenerate (e.g. negative) variances when the number of independent clusters is small. The code instead uses `geese$vbeta.naiv`, the model-based covariance that assumes the working correlation structure is exactly correct. It is less efficient asymptotically but always well-structured for small samples.

Cholesky positive-definiteness check After extracting the covariance, a Cholesky decomposition is attempted via `tryCatch(chol(Sigma_con), ...)`. This is the canonical test for positive definiteness: it succeeds if and only if the matrix is PD. A failed decomposition sets an `is_pd = FALSE` flag without crashing the call.

GLM fallback for non-PD GEE or GLMM covariance If the GEE naive covariance or the GLMM fixed-effect `vcov` is still not positive definite (GLMM failures are common with boundary variance estimates or convergence issues), a standard `glm(..., family = binomial)` is re-fitted on the same concordant data and its Fisher-information-based `vcov` is used instead. That matrix is always PD for a non-degenerate design matrix, providing a graceful degradation from GEE/GLMM to GLM.

Symmetrization `Sigma_con <- (Sigma_con + t(Sigma_con)) / 2` is applied after the GLM fallback re-extraction and again unconditionally before the matrix is passed to Stan. Name-indexed covariance matching can introduce tiny floating-point asymmetries; averaging with the transpose enforces exact symmetry as required by Stan's Cholesky-based samplers.

NA sanitization Aliased model terms (collinear columns in the concordant data) cause `coef()` and `vcov()` to return `NA`. These are caught before Stan sees them: `NA` entries in the prior mean vector default to `0` (a neutral prior), and any `NA` in the covariance matrix triggers replacement of the entire matrix with a wide diagonal (`diag(100, p)`), an uninformative but valid prior.

Diffuse prior fallback Multiple failure paths (too few concordant pairs, persistent non-PD matrix, `NA`-contaminated covariance) fall back to `diag(100, p)`, which is positive definite by construction and encodes a broad, independent prior that lets the discordant-pair likelihood dominate.

Treatment prior reset After extracting the concordant-model covariance, the entire first row and column (corresponding to the treatment coefficient) are zeroed and the diagonal entry is set to `prior_variance_treatment` (default 100). This decouples the treatment prior from the nuisance-covariate prior: concordant pairs inform covariate shrinkage but are not allowed to shrink the treatment coefficient, which remains independently diffuse. Zeroing the off-diagonals also eliminates any spurious prior correlation between treatment and covariates induced by the concordant-model fit.

See Also

`summary.bclogit`, `confint.bclogit`, `vcov.bclogit`, `coef.bclogit`

Examples

```
# Example usage
data("fhs")
fit <- bclogit(PREVHYP ~ TOTCHOL + CIGPDAY + BMI + HEARTRTE,
  data = fhs, treatment = PERIOD, strata = RANDID)
summary(fit)
```

`clogit.default` *Frequentist Conditional Logistic Regression*

Description

Fits a conditional logistic regression model for matched pairs using the discordant-pair GLM trick. This is a fast frequentist alternative to `bclogit`.

Usage

```
## Default S3 method:
clogit(
  formula = NULL,
  data = NULL,
  treatment = NULL,
  strata = NULL,
  subset = NULL,
  na.action = NULL,
  do_inference_on_var = "all",
  ...,
  y = NULL,
  X = NULL,
  treatment_name = NULL,
  call = NULL
)

## S3 method for class 'formula'
clogit(
  formula,
```

```
  data,
  treatment = NULL,
  strata = NULL,
  subset = NULL,
  na.action = NULL,
  do_inference_on_var = "all",
  ...
)

clogit(
  formula,
  data,
  treatment = NULL,
  strata = NULL,
  subset = NULL,
  na.action = NULL,
  do_inference_on_var = "all",
  ...
)
```

Arguments

`formula`	For the formula method, a symbolic description of the model.
`data`	A data frame containing the variables (for formula method).
`treatment`	Vector specifying the treatment variable.
`strata`	Vector specifying the strata (matched pairs).
`subset`	An optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	A function which indicates what should happen when the data contain NAs.
`do_inference_on_var`	Which variable(s) to compute standard errors for. `"all"` (default) computes SEs for all coefficients. An integer `j` computes the SE only for the `j`th coefficient (1 = treatment, then covariates in order). `"none"` skips inference entirely.
`...`	Additional arguments passed to `fastLogisticRegressionWrap::fast_logistic_regression` (e.g., convergence tolerances).
`y`	For the default method, a binary (0,1) response vector.
`X`	A data.frame, data.table, or model.matrix containing the variables.
`treatment_name`	Optional string name for the treatment variable.
`call`	Optional call object to store in the result.

Value

A list of class `"clogit_bclogit"` containing:

`coefficients`	Estimated coefficients (posterior means).

`var`	Variance-covariance matrix of the coefficients (diagonal, built from standard errors). `NULL` when `do_inference_on_var` is not `"all"`.
`flr_model`	The fitted fast logistic regression model object returned by `fastLogisticRegressionWrap::fast_logis`
`call`	The function call.
`terms`	The model terms.
`n`	Total number of observations.
`num_discordant`	Number of discordant pairs used for fitting.
`num_concordant`	Number of concordant pairs.
`X_model_matrix_col_names`	Column names of the covariate model matrix.
`treatment_name`	Name of the treatment variable.
`se`	Standard errors of the coefficients.
`z`	Z-statistics for each coefficient.
`pval`	Approximate p-values for each coefficient.
`do_inference_on_var`	The value of the `do_inference_on_var` argument.

An object of class `"clogit_bclogit"`.

An object of class `"clogit_bclogit"`.

Methods (by class)

- `clogit(default)`: Default method for matrix/data input.
- `clogit(formula)`: Formula method

See Also

`bclogit`, `summary.clogit_bclogit`, `coef.clogit_bclogit`, `vcov.clogit_bclogit`

Examples

```
data("fhs")
fit <- clogit(PREVHYP ~ TOTCHOL + CIGPDAY + BMI + HEARTRTE,
  data = fhs, treatment = PERIOD, strata = RANDID)
summary(fit)
n <- 200
dat <- data.frame(
  y = rbinom(n, 1, 0.5),
  x1 = rnorm(n),
  treatment = rep(c(0, 1), n / 2),
  strata = rep(1:(n / 2), each = 2)
)
fit <- clogit(y ~ x1, data = dat, treatment = treatment, strata = strata)
# Inference on treatment only (faster):
fit2 <- clogit(y ~ x1, data = dat, treatment = treatment, strata = strata,
               do_inference_on_var = 1)
n <- 200
```

```
dat <- data.frame(
  y = rbinom(n, 1, 0.5),
  x1 = rnorm(n),
  x2 = rnorm(n),
  treatment = rep(c(0, 1), n / 2),
  strata = rep(1:(n / 2), each = 2)
)
fit <- clogit(y ~ x1 + x2, data = dat, treatment = treatment, strata = strata)
summary(fit)
coef(fit)
vcov(fit)
```

`coef.bclogit` *Extract coefficients from a bclogit model*

Description

Extract coefficients from a bclogit model

Usage

```
## S3 method for class 'bclogit'
coef(object, ...)
```

Arguments

`object`	A `bclogit` object.
`...`	Additional arguments.

Value

Named numeric vector of posterior mean coefficients.

See Also

`vcov.bclogit`, `summary.bclogit`

Examples

```
data("fhs")
fit <- bclogit(PREVHYP ~ TOTCHOL + BMI, data = fhs,
               treatment = PERIOD, strata = RANDID)
coef(fit)
```

`coef.clogit_bclogit` *Extract coefficients from a clogit_bclogit model*

Description

Extract coefficients from a clogit_bclogit model

Usage

```
## S3 method for class 'clogit_bclogit'
coef(object, ...)
```

Arguments

`object`	A `clogit_bclogit` object.
`...`	Additional arguments.

Value

Numeric vector of coefficients.

Examples

```
n <- 200
dat <- data.frame(
  y = rbinom(n, 1, 0.5), x1 = rnorm(n),
  treatment = rep(c(0, 1), n / 2),
  strata = rep(1:(n / 2), each = 2)
)
fit <- clogit(y ~ x1, data = dat, treatment = treatment, strata = strata)
coef(fit)
```

`confint.bclogit` *Credible Intervals for bclogit Parameters*

Description

Computes Bayesian credible intervals for the model parameters.

Usage

```
## S3 method for class 'bclogit'
confint(object, parm, level = 0.95, type = c("HPD_one", "CR", "HPD_many"), ...)
```

Arguments

`object`	A `bclogit` object.
`parm`	A specification of which parameters to be given credible intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	The confidence level required (default 0.95).
`type`	Type of interval to compute: "HPD_one" (default unimodal HPD interval via coda), "CR" (equal-tailed credible region), "HPD_many" (multimodal HPD interval via ggdist).
`...`	Additional arguments.

Value

A matrix with columns `lower` and `upper`. For `"HPD_many"`, a parameter may appear on multiple rows when the interval is disjoint. The matrix has a `Probability` attribute.

See Also

`summary.bclogit`, `coef.bclogit`

`fhs` *Framingham Heart Study Dataset*

Description

A subset of the Framingham Heart Study data.

Usage

```
fhs
```

Format

A data frame with 5944 rows and 39 variables:

RANDID Unique identification number for each participant. (This is the strata for the matched pairs).

PERIOD Examination Cycle where 0 = baseline, 1 = endpoint (the treatment variable)

SEX Participant sex (1 = Male, 2 = Female).

TOTCHOL Total serum cholesterol (mg/dL).

AGE Age at exam (years).

SYSBP Systolic blood pressure (mmHg).

DIABP Diastolic blood pressure (mmHg).

CURSMOKE Current smoking status (0 = No, 1 = Yes).

CIGPDAY Number of cigarettes smoked per day.

BMI Body Mass Index (kg/m^2).

DIABETES Diabetes status (0 = No, 1 = Yes).

BPMEDS Use of Anti-hypertensive medication (0 = No, 1 = Yes).

HEARTRTE Heart rate (beats/minute).

GLUCOSE Fast blood glucose (mg/dL).

educ Education level.

PREVCHD Prevalence of Coronary Heart Disease.

PREVAP Prevalence of Angina Pectoris.

PREVMI Prevalence of Myocardial Infarction.

PREVSTRK Prevalence of Stroke.

PREVHYP Prevalence of Hypertension.

TIME Number of days since baseline exam.

HDLC High Density Lipoprotein Cholesterol.

LDLC Low Density Lipoprotein Cholesterol.

DEATH Death status.

ANGINA Angina Pectoris status.

HOSPMI Hospitalized Myocardial Infarction.

MI_FCHD Myocardial Infarction or Fatal Coronary Heart Disease.

ANYCHD Any Coronary Heart Disease event.

STROKE Stroke status.

CVD Cardiovascular Disease status.

HYPERTEN Hypertension status.

TIMEAP Time to Angina Pectoris.

TIMEMI Time to Myocardial Infarction.

TIMEMIFC Time to Myocardial Infarction or Fatal CHD.

TIMECHD Time to Any CHD.

TIMESTRK Time to Stroke.

TIMECVD Time to Cardiovascular Disease.

TIMEDTH Time to Death.

TIMEHYP Time to Hypertension.

Details

This dataset was constructed by running the following code:

```
pacman::p_load(riskCommunicator, data.table)
data(”framingham”)
D = data.table(framingham)
D = D[!is.na(CIGPDAY)] #we drop missing data in covariates
D = D[!is.na(BMI)]
```

```
D = D[!is.na(HEARTRTE)]
D = D[!is.na(TOTCHOL)]
Dba = D[PERIOD %in% c(1,3)] #we drop intermediate periods so we have matched pairs
Dba[PERIOD == 1, PERIOD := 0]
Dba[PERIOD == 3, PERIOD := 1]
Dba[, num_periods_per_id := .N, by = RANDID]
Dba = Dba[num_periods_per_id == 2] #we drop intermediate periods so we have matched pairs
Dba[, num_periods_per_id := NULL]
```

Source

`https://biolincc.nhlbi.nih.gov/teaching/`

`formula.bclogit` *Extract model formula*

Description

Extract model formula

Usage

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

Arguments

`x`	A `bclogit` object.
`...`	Additional arguments.

Value

The `formula` used in the model, or `NULL` when the model was fitted via the default (matrix) interface.

See Also

`bclogit`

Examples

```
data("fhs")
fit <- bclogit(PREVHYP ~ TOTCHOL + BMI, data = fhs,
               treatment = PERIOD, strata = RANDID)
formula(fit)
```

`print.summary.bclogit` *Print summary of a bclogit model*

Description

Print summary of a bclogit model

Usage

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

Arguments

`x`	A `summary.bclogit` object.
`digits`	Number of significant digits to print.
`...`	Additional arguments.

Value

Invisibly returns `x`.

See Also

`summary.bclogit`, `confint.bclogit`

Examples

```
data("fhs")
fit <- bclogit(PREVHYP ~ TOTCHOL + BMI, data = fhs,
               treatment = PERIOD, strata = RANDID)
print(summary(fit))
```

`print.summary.clogit_bclogit` *Print summary of a clogit_bclogit model*

Description

Print summary of a clogit_bclogit model

Usage

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

Arguments

`x`	A `summary.clogit_bclogit` object.
`digits`	Number of significant digits to print.
`...`	Additional arguments.

Value

Invisibly returns `x`.

Examples

```
n <- 200
dat <- data.frame(
  y = rbinom(n, 1, 0.5), x1 = rnorm(n),
  treatment = rep(c(0, 1), n / 2),
  strata = rep(1:(n / 2), each = 2)
)
fit <- clogit(y ~ x1, data = dat, treatment = treatment, strata = strata)
print(summary(fit))
```

`summary.bclogit` *Summarize a bclogit model*

Description

Summarize a bclogit model

Usage

```
## S3 method for class 'bclogit'
summary(object, level = 0.95, inference_method = "HPD_one", ...)
```

Arguments

`object`	A `bclogit` object.
`level`	Confidence level for credible intervals (default 0.95).
`inference_method`	Method used for both the displayed confidence set bounds and the p-value (computed via bisection over alpha). Options are: `"HPD_one"` (default) uses unimodal HPD intervals (C++) with 20 bisection iterations, `"HPD_many"` uses `ggdist::hdi` which supports disjoint (multimodal) HPD regions, with 50 bisection iterations (requires the **ggdist** package). Confidence set bounds are shown when the HPD is a single interval; if disjoint, they are set to `NA` (use `confint.bclogit` with `type = "HPD_many"` to see all intervals), `"CR"` uses equal-tailed credible intervals (quantile-based, C++) with 20 bisection iterations.
`...`	Additional arguments (not used).

Value

A list of class `"summary.bclogit"` containing:

`call`	The original function call.
`coefficients`	A matrix with one row per parameter and columns for the posterior mean estimate, posterior median estimate, standard error, lower and upper credible interval bounds, optionally `Rhat` and `n_eff` convergence diagnostics (when available from Stan), and `Pr(tx!=0)` (the Bayesian p-value).
`num_discordant`	Number of discordant pairs used for fitting.
`num_concordant`	Number of concordant pairs used for the prior.
`level`	The credible interval level used.
`inference_method`	The inference method used for interval and p-value computation.
`prior_info`	A list with elements `mu` and `Sigma` describing the prior derived from the concordant pairs model.
`treatment_name`	Name of the treatment variable.

See Also

`print.summary.bclogit`, `confint.bclogit`, `coef.bclogit`, `vcov.bclogit`

`summary.clogit_bclogit` *Summarize a clogit_bclogit model*

Description

Summarize a clogit_bclogit model

Usage

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

Arguments

`object`	A `clogit_bclogit` object.
`...`	Additional arguments (not used).

Value

A list of class `"summary.clogit_bclogit"` containing:

`call`	The original function call.
`coefficients`	A matrix with one row per parameter and columns `Estimate`, `Std. Error`, `z value`, and `Pr(>|z|)`.
`num_discordant`	Number of discordant pairs used for fitting.
`num_concordant`	Number of concordant pairs.
`n`	Total number of observations.
`treatment_name`	Name of the treatment variable.
`do_inference_on_var`	The value of the `do_inference_on_var` argument.

See Also

`clogit`, `print.summary.clogit_bclogit`, `coef.clogit_bclogit`, `vcov.clogit_bclogit`

Examples

```
n <- 200
dat <- data.frame(
  y = rbinom(n, 1, 0.5),
  x1 = rnorm(n),
  treatment = rep(c(0, 1), n / 2),
  strata = rep(seq_len(n / 2), each = 2)
)
fit <- clogit(y ~ x1, data = dat, treatment = treatment, strata = strata)
summary(fit)
```

`vcov.bclogit` *Extract variance-covariance matrix from a bclogit model*

Description

Extract variance-covariance matrix from a bclogit model

Usage

```
## S3 method for class 'bclogit'
vcov(object, ...)
```

Arguments

`object`	A `bclogit` object.
`...`	Additional arguments.

Value

A square matrix of the posterior covariance of the coefficients, derived from the MCMC samples.

See Also

`coef.bclogit`, `summary.bclogit`

Examples

```
data("fhs")
fit <- bclogit(PREVHYP ~ TOTCHOL + BMI, data = fhs,
               treatment = PERIOD, strata = RANDID)
vcov(fit)
```

`vcov.clogit_bclogit` *Extract variance-covariance matrix from a clogit_bclogit model*

Description

Extract variance-covariance matrix from a clogit_bclogit model

Usage

```
## S3 method for class 'clogit_bclogit'
vcov(object, ...)
```

Arguments

`object`	A `clogit_bclogit` object.
`...`	Additional arguments.

Value

A diagonal matrix of the estimated coefficient variances (squared standard errors), or `NULL` when `do_inference_on_var` is not `"all"`.

See Also

`coef.clogit_bclogit`, `summary.clogit_bclogit`

Examples

```
n <- 200
dat <- data.frame(
  y = rbinom(n, 1, 0.5),
  x1 = rnorm(n),
  treatment = rep(c(0, 1), n / 2),
  strata = rep(seq_len(n / 2), each = 2)
)
fit <- clogit(y ~ x1, data = dat, treatment = treatment, strata = strata)
vcov(fit)
```

Index