Package: fewclust (via r-universe)

November 27, 2026

Title Regression for Clustered Binary Outcomes with Few Clusters

Version 0.4.4

Description Fits regression models for binary outcomes measured repeatedly within a small number of groups (clusters), with tests and intervals that stay calibrated with few groups and short series. The default is the marginalized multilevel model of Heagerty (1999) <doi:10.1111/j.0006-341X.1999.00688.x> with a convergence check and Wald statistics referred to a t distribution; generalized estimating equations with a first-order autoregressive working correlation and the small-sample sandwich correction of Fay and Graubard (2001) <doi:10.1111/j.0006-341X.2001.01198.x> are the fallback. The corrections of Mancl and DeRouen (2001) <doi:10.1111/j.0006-341X.2001.00126.x> and Kauermann and Carroll (2001) <doi:10.1198/016214501753382309> are available, and the sign-flip score test of Hemerik, Goeman and Finos (2020) <doi:10.1111/rssb.12369> gives a p-value for the smallest designs. The defaults follow a simulation study of these methods. Fits take the data first, accept bare column names and have 'broom' methods returning tibbles.

License MIT + file LICENSE

URL https://github.com/lucavd/fewclust

BugReports https://github.com/lucavd/fewclust/issues

Depends R (>= 4.1)

Imports cli (>= 3.6.0), geepack, generics, lpSolve, rlang (>= 1.1.0), stats, tibble, tidyselect (>= 1.2.0), utils

Suggests covr, dplyr, knitr, purrr, rmarkdown, spelling, testthat (>= 3.2.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

LazyData true

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev

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

Date/Publication 2026-11-27 14:06:38 UTC

RemoteUrl https://github.com/lucavd/fewclust

RemoteRef HEAD

RemoteSha a634c8470fd7eb65f1edce1b0560f4daac724f00

Contents

fc_trial 2
fewclust_control 3
fit_fewclust 4
flip_test 7
gee_vcov 9
predict.fewclust 10
simulate_clustered_binary 11
summary.fewclust 12
tidy.fewclust_flip 13

Index **14**

`fc_trial` *Synthetic clustered binary trial*

Description

A small synthetic dataset in the shape the package targets: a binary outcome measured at 4 occasions in each of 24 groups, with a group-level treatment indicator. Generated entirely in-package by `data-raw/fc_trial.R`, which calls `simulate_clustered_binary(n_groups = 24, n_occasions = 4, effect = 0.6, sigma_u = 1, slope_sd = 0, prevalence = 0.3, seed = 2026)`.

Usage

```
fc_trial
```

Format

A data frame with 96 rows and 5 variables:

id group identifier (factor, 24 levels)

time measurement occasion, scaled to $[0, 1]$

arm group-level treatment indicator (0/1, constant within group)

x observation-level continuous covariate

y binary outcome (0/1)

Source

Simulated; see `data-raw/fc_trial.R` in the package sources.

Examples

```
head(fc_trial)
tapply(fc_trial$y, fc_trial$arm, mean)
```

---

fewclust_control *Tuning constants for the fits*

Description

Most users never need this function: the defaults are those of the simulation study behind the package. It is passed to `fit_fewclust()` as `control`.

Usage

```
fewclust_control(
  q = 20L,
  check = TRUE,
  check_coherence = TRUE,
  check_threshold = 0.001,
  fg_bound = 0.75
)
```

Arguments

`q`	Number of Gauss-Hermite quadrature points of the marginalized multilevel model.
`check`	Whether the marginalized model applies its convergence check. When `time` is given the model lets the latent standard deviation follow a linear trend over the occasions; a fit whose optimiser did not converge (`nlm()` code other than 1 or 2), or whose slope of the latent standard deviation is below `check_threshold`, is replaced by the fit with a constant latent standard deviation, but only if `nlm()` reports convergence of that fit (code 1 or 2) and its estimates and standard errors are finite. Otherwise the marginalized model gives no result: `fit_fewclust()` falls back to GEE with `method = "auto"` and stops with an error with `method = "mmm"`. The fit at the boundary is never reported. When `time` is `NULL` the check likewise rejects a constant-SD fit that does not converge or has non-finite estimates or standard errors. The model is recommended only with the check; switching it off is recorded in the notes of the fit.

check_coherence
: Whether the check also replaces a trend fit whose log-likelihood is below that of the fit with a constant latent standard deviation. The constant fit is a sub-model of the trend fit, so a maximum cannot lie below it: the optimiser stopped at a false optimum. This condition is an addition of the package to the procedure of the study; `check_coherence = FALSE` gives that procedure exactly.

check_threshold
: Slope of the latent standard deviation (per occasion) below which the trend fit is considered at the boundary.

fg_bound
: Bound b of the Fay-Graubard correction, strictly inside (0, 1); 0.75 is the value recommended by Fay and Graubard (2001).

Value

A list of class `"fewclust_control"`.

Examples

```
fewclust_control()

# the convergence check exactly as in the simulation study
fit_fewclust(fc_trial, y ~ time * arm,
  id = id, time = time,
  control = fewclust_control(check_coherence = FALSE)
)
```

---

fit_fewclust *Fit a model for a clustered binary outcome*

Description

Fits a regression model for a binary outcome measured repeatedly within a small number of groups (clusters), with tests and intervals that remain calibrated with few groups. The default is the method recommended by the simulation study behind the package: the marginalized multilevel model of Heagerty (1999) with a convergence check, its Wald statistics referred to a t distribution on N - p degrees of freedom (N groups, p coefficients). When that model gives no usable estimate, GEE with the Fay-Graubard small-sample correction and the same t reference is fitted instead (using an AR(1) working correlation with `time` and an exchangeable working correlation without `time`, falling back to independence if the default working correlation degenerates), and a message says so.

Usage

```
fit_fewclust(
  data,
  formula,
  id,
  time = NULL,
  method = c("auto", "mmm", "gee"),
```

```
  corstr = NULL,
  correction = c("fg", "md", "kc", "none"),
  conf_level = 0.95,
  control = fewclust_control()
)
```

Arguments

data	A data frame or tibble with one row per observation.
formula	A two-sided formula for the mean model, for example `y ~ time * arm + x`. The outcome must be binary: 0/1, logical, or a factor with two levels (the second level is the event).
id	The column that identifies the groups (clusters). A bare column name or a string; inside your own functions use `{{ }}`.
time	Optional numeric column with the measurement occasion. It orders the observations within a group and gives the marginalized model its serial dependence; recommended for repeated measurements.
method	`"auto"` (default), `"mmm"` or `"gee"`.
corstr	Working correlation of GEE: `"ar1"`, `"exchangeable"` or `"independence"`. When `NULL` (default), uses `"ar1"` if `time` is given and `"exchangeable"` if `time` is `NULL`, falling back to `"independence"` if that working correlation degenerates.
correction	Small-sample correction of the GEE sandwich variance: `"fg"` (Fay-Graubard, default), `"md"` (Mancl-DeRouen), `"kc"` (Kauermann-Carroll) or `"none"`.
conf_level	Confidence level of the intervals.
control	Tuning constants from `fewclust_control()`.

Details

The coefficients are marginal (population-averaged) log odds ratios.

Value

An object of class `"fewclust"`. Use `print()`, `summary()`, `coef()`, `vcov()`, `confint()`, `predict()`, `nobs()`, and the broom verbs `tidy()`, `glance()` and `augment()`, which return tibbles. The list also holds `method` (`"mmm"` or `"gee"`, the model reported), `fallback` (`NULL`, or why the marginalized model was not used), `df`, `notes`, `convergence` (the record of the convergence check of the marginalized model, also when it gave no result and GEE was reported instead) and `engine` (the underlying fit).

Methods

- `"mmm"`: the marginalized multilevel model (Heagerty 1999). With `time` the latent standard deviation follows a linear trend over the occasions and a first-order transition term links consecutive observations; the convergence check (see `fewclust_control()`) replaces a trend fit at the boundary with the fit with a constant latent standard deviation when that fit converges and has finite estimates and standard errors; otherwise the model gives no result (`"auto"` then

fits GEE, "mmm" stops with an error). The fit at the boundary is never reported. Without time the latent standard deviation is constant and there is no transition term. Standard errors are model-based on both paths (with and without time), so they rely on the dependence structure being adequately captured; when the model includes within-cluster covariates whose effect may vary across groups (random slopes), compare with method = "gee".

- "gee": geepack::geeglm() with the scale parameter fixed at 1, on rows ordered by group and time, with the small-sample sandwich variance chosen by correction (see gee_vcov()). When corstr is not supplied and the default working correlation fails to converge or gives a non-positive-definite working covariance, GEE is refitted with corstr = "independence" and a message says so. With time-varying covariates and a non-independence working correlation, consistency of GEE requires the full-covariate conditional mean assumption of Pepe and Anderson (1994); where covariate feedback is suspected, use corstr = "independence".
- "auto" (default): "mmm", and "gee" (with corstr and correction) when "mmm" gives no usable estimate.

A coefficient is usable when its estimate and standard error are finite and smaller than 10 in absolute value on the logit scale; otherwise its statistic, p-value and interval are NA (with a warning). A model with no usable coefficient gives no result.

Errors, warnings, messages and notes

The function stops with an error (class "fewclust_error") when no estimate can be computed: an outcome that is not binary or is constant, an outcome that varies within one group only, separation (checked with the exact linear programme of Konis 2007), as many coefficients as groups or more, a single observation per group, repeated values of time within a group, random-effects terms in the formula, or no usable estimate from the requested method(s). It warns (class "fewclust_warning") when the result is less reliable: fewer than 10 groups; GEE with 2 occasions and fewer than 15 groups; a GEE correction other than Fay-Graubard; AR(1) working correlation without time; coefficients without inference; a marginalized model that did not converge; series with gaps or unequally spaced occasions. Messages (class "fewclust_message") report a fallback to GEE or to an independence working correlation, a missing time, and within-cluster covariates under the marginalized model. Notes stored in the fit and shown by print() describe small designs and the convergence check.

References

Heagerty PJ (1999). Marginally specified logistic-normal models for longitudinal binary data. *Biometrics* 55(3), 688-698. doi:10.1111/j.0006341X.1999.00688.x

Fay MP, Graubard BI (2001). Small-sample adjustments for Wald-type tests using sandwich estimators. *Biometrics* 57(4), 1198-1206. doi:10.1111/j.0006341X.2001.01198.x

Konis K (2007). Linear programming algorithms for detecting separated data in binary logistic regression models. DPhil thesis, University of Oxford.

Pepe MS, Anderson GL (1994). A cautionary note on inference for marginal regression models with longitudinal data and general correlated response data. *Communications in Statistics - Simulation and Computation* 23(4), 939-951. doi:10.1080/03610919408813210

See Also

flip_test() for a p-value with very small designs, gee_vcov() for the sandwich estimators.

Examples

```
fit <- fit_fewclust(fc_trial, y ~ time * arm, id = id, time = time)
fit
tidy(fit, conf.int = TRUE)

# GEE with an exchangeable working correlation
fit_fewclust(fc_trial, y ~ time * arm,
  id = id, time = time,
  method = "gee", corstr = "exchangeable"
)

# in a pipe
fc_trial |>
  fit_fewclust(y ~ time + arm + x, id = id, time = time) |>
  glance()
```

---

flip_test *Sign-flip score test for one coefficient*

Description

Tests that one coefficient of the mean model is zero with the sign-flip score test of Hemerik, Goeman and Finos (2020), treating the groups as independent units. It gives a p-value only: no estimate and no interval. It is useful with 2 or 3 occasions and few groups, where the models of `fit_fewclust()` often give no usable estimate and have little power; there the test keeps its level, but it has little power too.

Usage

```
flip_test(
  data,
  formula,
  id,
  time = NULL,
  term = NULL,
  reps = 1999,
  enum_max = 12,
  seed = NULL
)
```

Arguments

`data`	A data frame (or tibble).
`formula`	Two-sided formula for the mean model, e.g. `y ~ time * arm`.
`id`	The column that identifies the groups: their contributions change sign together. A bare column name or a string.

time	Optional numeric column of the measurement occasions. It only orders the rows within a group, which makes the result identical whatever the order of the rows of data.
term	The coefficient to test: a column name of the model matrix (for example "time:arm"). By default the last one, which for y ~ time * arm is the interaction.
reps	Number of random sign assignments when there are more than enum_max groups.
enum_max	Largest number of groups for which all 2^N sign assignments are enumerated.
seed	Optional seed. When given, the random signs are drawn after set.seed(seed) and the random number stream of the session is left untouched; when NULL, the current stream is used.

Details

The model without the tested term is fitted as a logistic regression with independent observations (stats::glm.fit(), tolerance 1e-12, at most 100 iterations). The effective score contribution of each group to the tested term is its score contribution minus the projection on the other coefficients. The p-value compares the absolute sum of the contributions with its distribution under changes of sign: all 2^N sign assignments when there are at most enum_max groups, otherwise reps random assignments plus the identity. With other coefficients estimated the test is asymptotically exact and robust to differences in variance between groups.

The test is not computable, and returns NA with a warning that gives the reason, when the outcome is constant; when the model without the tested term fails, does not converge or diverges (a standardised coefficient of 1,000 or more in absolute value, as with separation); when its information matrix is singular; or when the score contributions are all zero or numerically zero relative to the scale of the tested covariate (the tested term is not identified, for instance when one arm has only zero outcomes).

Value

An object of class "fewclust_flip", a list with p.value (NA when not computable), statistic (the sum of the effective score contributions), term, n_groups, n_signs (sign assignments in the reference distribution, the identity included), enumerated, computable, reason (NA when computable), contributions (one per group) and call. tidy() turns it into a one-row tibble.

References

Hemerik J, Goeman JJ, Finos L (2020). Robust testing in generalized linear models by sign flipping score contributions. *Journal of the Royal Statistical Society: Series B* 82(3), 841-864. doi:10.1111/rssb.12369

See Also

fit_fewclust()

Examples

```
flip_test(fc_trial, y ~ time * arm, id = id, time = time, seed = 1)
```

```
# with 3 occasions and 12 groups
d <- simulate_clustered_binary(n_groups = 12, n_occasions = 3, seed = 4)
d |>
  flip_test(y ~ time * arm, id = id, time = time, seed = 1) |>
  tidy()
```

---

gee_vcov *Small-sample corrected sandwich variance for a GEE fit*

Description

Computes the Liang-Zeger sandwich variance of a `geepack::geeglm()` fit and its three small-sample corrections, implemented directly in this package: Fay-Graubard (`"fg"`), Mancl-DeRouen (`"md"`) and Kauermann-Carroll (`"kc"`).

Usage

```
gee_vcov(fit, type = c("unc", "fg", "md", "kc"), fg_bound = 0.75)
```

Arguments

`fit`	A fitted `geepack::geeglm()` object.
`type`	`"unc"` (Liang-Zeger, uncorrected), `"fg"`, `"md"` or `"kc"`.
`fg_bound`	Bound b for the Fay-Graubard multiplier $(1 - \min(b, Q_{jj}))^{-1/2}$; default 0.75, must lie strictly inside (0, 1).

Details

The Kauermann-Carroll factor $(I - H_i)^{-1/2}$ is computed through a similarity transformation that makes the matrix symmetric before the eigendecomposition, so eigenvalues are real by construction. When a cluster's leverage reaches 1 (for instance when a covariate identifies a single cluster), or when the estimated working covariance is not positive definite, the variance is mathematically undefined: the function then signals a classed error (condition class `"fewclust_correction_error"`) naming the cluster, rather than returning an unstable number. A final gate rejects any non-finite or non-positive-diagonal variance for every estimator.

Prior weights and non-contiguous cluster ids are handled exactly as `geepack::geeglm()` handles them (each maximal run of equal ids is one cluster); fits made with the `waves` argument of `geepack::geeglm()` are rejected with a classed error, since the working correlation cannot be reconstructed from row positions.

Value

A $p \times p$ variance-covariance matrix for the regression coefficients.

References

Fay MP, Graubard BI (2001). Small-sample adjustments for Wald-type tests using sandwich estimators. *Biometrics* 57(4), 1198-1206. Mancl LA, DeRouen TA (2001). A covariance estimator for GEE with improved small-sample properties. *Biometrics* 57(1), 126-134. Kauermann G, Carroll RJ (2001). A note on the efficiency of sandwich covariance matrix estimation. *JASA* 96(456), 1387-1396.

Examples

```
d <- simulate_clustered_binary(n_groups = 20, seed = 1)
m <- geepack::geeglm(y ~ time + arm,
  id = id, data = d,
  family = binomial("logit")
)
sqrt(diag(gee_vcov(m, "unc")))
sqrt(diag(gee_vcov(m, "fg")))
```

---

predict.fewclust *Predictions from a fewclust fit*

Description

Marginal (population-averaged) predictions: the linear predictor on the log-odds scale or the probability of the event.

Usage

```
## S3 method for class 'fewclust'
predict(object, newdata = NULL, type = c("link", "response"), ...)
```

Arguments

`object`	A `"fewclust"` fit.
`newdata`	Optional data frame with the covariates of the mean model; by default the observations used in the fit.
`type`	`"link"` (log odds, default) or `"response"` (probability).
`...`	Unused.

Value

A numeric vector.

Examples

```
fit <- fit_fewclust(fc_trial, y ~ time * arm, id = id, time = time)
nd <- data.frame(time = c(0, 1), arm = 1)
predict(fit, newdata = nd, type = "response")
```

`simulate_clustered_binary` *Simulate clustered binary outcome data*

Description

Generates data from a random-intercept (optionally random-slope) logistic model: observations nested in groups, a group-level treatment indicator `arm` (balanced by design), a within-group measurement occasion `time` scaled to $[0, 1]$, and an observation-level covariate `x`. The package's examples, tests and vignettes all draw their data from this generator, so everything runs without external data.

Usage

```
simulate_clustered_binary(
  n_groups = 24,
  n_occasions = 4,
  effect = 0.6,
  sigma_u = 1,
  slope_sd = 0,
  prevalence = 0.3,
  seed = NULL
)
```

Arguments

`n_groups`	Number of groups (clusters). At least 2.
`n_occasions`	Number of measurement occasions per group. At least 1.
`effect`	Log-odds coefficient of the group-level `arm` indicator.
`sigma_u`	Standard deviation of the random intercept.
`slope_sd`	Standard deviation of the group-varying slope on `time`; `0` (the default) generates random intercepts only.
`prevalence`	Target marginal prevalence of the outcome, in (0, 1).
`seed`	Optional integer seed. When supplied, the caller's RNG state is left untouched.

Details

The intercept is solved from `prevalence` using the logistic-normal attenuation approximation, so the realised marginal prevalence is close to the requested one for moderate `sigma_u`.

Value

A tibble with `n_groups * n_occasions` rows and columns `id` (factor), `time` (numeric in $[0, 1]$), `arm` (0/1, constant within group), `x` (numeric) and `y` (0/1).

Examples

```
d <- simulate_clustered_binary(n_groups = 12, n_occasions = 4, seed = 1)
d
fit_fewclust(d, y ~ time + arm, id = id, time = time)
```

`summary.fewclust` *Summarise a fewclust fit*

Description

Summarise a fewclust fit

Usage

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

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

Arguments

`object, x`	A `"fewclust"` fit, or its summary.
`...`	Unused.
`digits`	Number of significant digits to print.

Value

`summary()` returns an object of class `"summary.fewclust"`, printed with the coefficient table, the confidence intervals, details of the model (convergence check of the marginalized model, working correlation of GEE) and the notes.

Examples

```
fit <- fit_fewclust(fc_trial, y ~ time * arm, id = id, time = time)
summary(fit)
```

`tidy.fewclust_flip` *Tidy, glance and augment a fewclust fit*

Description

The broom verbs, returning tibbles.

Usage

```
## S3 method for class 'fewclust_flip'
tidy(x, ...)

## S3 method for class 'fewclust'
tidy(x, conf.int = FALSE, conf.level = x$conf_level, ...)

## S3 method for class 'fewclust'
glance(x, ...)

## S3 method for class 'fewclust'
augment(x, newdata = NULL, type.predict = c("response", "link"), ...)
```

Arguments

`x`	A `"fewclust"` fit, or a `"fewclust_flip"` test for `tidy()`.
`...`	Unused.
`conf.int`	Whether to add the confidence interval.
`conf.level`	Confidence level; by default that of the fit.
`newdata`	Optional data frame for `augment()`; by default the observations used in the fit.
`type.predict`	Scale of `.fitted` in `augment()`: `"response"` (probability, default) or `"link"` (log odds).

Value

`tidy()`: one row per coefficient, with `term`, `estimate`, `std.error`, `statistic`, `df`, `p.value` (and `conf.low`, `conf.high`). For a sign-flip test: one row with `term`, `statistic`, `p.value`, `n_groups`, `n_signs` and `method`. `glance()`: one row describing the fit. `augment()`: the data with `.fitted` (and `.resid`, the response residual, for the data of the fit).

Examples

```
fit <- fit_fewclust(fc_trial, y ~ time * arm, id = id, time = time)
tidy(fit, conf.int = TRUE)
glance(fit)
augment(fit)
```

Index

∗ **datasets**
fc_trial, 2

augment(), *5*
augment.fewclust *(*tidy.fewclust_flip*)*, 13

fc_trial, 2
fewclust_control, 3
fewclust_control(), *5*
fit_fewclust, 4
fit_fewclust(), *3*, *7*, *8*
flip_test, 7
flip_test(), *6*

gee_vcov, 9
gee_vcov(), *6*
geepack::geeglm(), *6*, *9*
glance(), *5*
glance.fewclust *(*tidy.fewclust_flip*)*, 13

predict.fewclust, 10
print.summary.fewclust *(*summary.fewclust*)*, 12

simulate_clustered_binary, 11
stats::glm.fit(), *8*
summary.fewclust, 12

tidy(), *5*, *8*
tidy.fewclust *(*tidy.fewclust_flip*)*, 13
tidy.fewclust_flip, 13