

Package ‘fbrglm’

July 14, 2026

Title Safe Formula-Based Regularized Generalized Linear Models

Version 0.1.0

Description A formula-based wrapper around 'glmnet' that brings the 'glm()-compatible modeling workflow to regularized generalized linear models. Training-time 'terms', 'xlevels', and 'contrasts' are stored on the fit object and reused at predict time, so the design matrix is reconstructed consistently across sessions. Complete-case bookkeeping is exposed via 'nobs_info', and linearly dependent columns are detected by a QR pivot and reported as 'NA' in 'coef()' and 'summary()' (the 'stats::glm()' convention), distinguishing ``not identifiable" from ``shrunk to zero by the penalty". Novel factor levels at predict time raise the same error 'stats::predict.glm()' does by default, with 'on_new_levels = ``na"' as a production-style opt-in. Accepts character family strings ('gaussian', 'binomial', 'poisson', 'cox', 'multinomial', 'mgaussian') and any 'glm' family object the underlying 'glmnet' itself accepts, including 'Gamma' and fixed-theta negative binomial via 'MASS::negative.binomial'.

URL <https://github.com/dsc-chiba-u/fbrglm>

BugReports <https://github.com/dsc-chiba-u/fbrglm/issues>

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports glmnet, stats, graphics, generics, tibble

Suggests testthat (>= 3.0.0), knitr, rmarkdown, survival, MASS

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Koki Tsuyuzaki [aut, cre]

Maintainer Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

Repository CRAN

Date/Publication 2026-07-14 08:30:02 UTC

Contents

as_cv_glmnet	2
as_glmnet	2
fbrglm	3
Index	5

as_cv_glmnet	<i>Extract the Underlying cv.glmnet Fit</i>
--------------	---

Description

Returns the raw cv.glmnet object stored inside an fbrglm model. This is NULL when the model was fit with lambda = "fix".

Usage

```
as_cv_glmnet(object, ...)
```

Arguments

- | | |
|--------|-------------------|
| object | An fbrglm object. |
| ... | Ignored. |

Value

A cv.glmnet object, or NULL.

as_glmnet	<i>Extract the Underlying glmnet Fit</i>
-----------	--

Description

Returns the raw glmnet object stored inside an fbrglm model. For a lambda = "fix" fit this is the direct `glmnet::glmnet()` return; for a CV fit it is the underlying `glmnet.fit` (`cv_fit$glmnet.fit`).

Usage

```
as_glmnet(object, ...)
```

Arguments

- | | |
|--------|-------------------|
| object | An fbrglm object. |
| ... | Ignored. |

Value

A glmnet object, or NULL if no fit has been attached yet.

fbrglm

*Fit a Formula-Based Regularized GLM***Description**

Fits a regularized generalized linear model with a formula/data interface that mirrors base R's `stats::glm()` while delegating the actual penalized fit to `glmnet::glmnet()` / `glmnet::cv.glmnet()`.

Usage

```
fbrglm(
  formula,
  data,
  family = c("gaussian", "binomial", "poisson"),
  weights = NULL,
  offset = NULL,
  infer = c("none", "split", "selective"),
  selection_frac = 0.2,
  alpha = 1,
  lambda = c("cv_min", "cv_1se", "fix"),
  lambda_value = NULL,
  x = NULL,
  y = NULL,
  ...
)
```

Arguments

formula	A model formula, e.g. $y \sim x_1 + x_2$. For Cox a survival::Surv(time, status) ~ ... LHS is accepted; for mgaussian the LHS is a matrix expression such as cbind(y1, y2) ~
data	A data frame containing the variables in formula.
family	A character string ("gaussian", "binomial", "poisson", "cox", "multinomial", "mgaussian"), a GLM family object (e.g. stats::Gamma(link = "log"), MASS::negative.binomial(= 2)), or a bare family generator (e.g. binomial) – the same surface glmnet itself accepts. Cox, multinomial, and mgaussian are supported but experimental (see Details).
weights	Optional observation weights, passed to glmnet / cv.glmnet.
offset	Optional offset vector, passed to glmnet / cv.glmnet. Reused at predict time when newdata = NULL; for newdata, supply newoffset to predict().
infer	Inference mode: "none", "split", or "selective". Only "none" is implemented; the other two error.
selection_frac	Selection-share for infer = "split" (default 0.2). Stored only; not yet used.
alpha	Elastic-net mixing parameter, passed to glmnet.
lambda	lambda-selection rule: "cv_min", "cv_1se", or "fix".

<code>lambda_value</code>	Numeric lambda used when <code>lambda = "fix"</code> .
<code>x, y</code>	Optional pre-built design matrix and response. Not yet supported; supply formula + data instead.
<code>...</code>	Additional arguments forwarded to <code>glmnet::glmnet()</code> / <code>glmnet::cv.glmnet()</code> (<code>nlambda</code> , <code>nfolds</code> , <code>standardize</code> , ...).

Details

Current scope: `infer = "none"` only, with the same family argument surface as `glmnet` itself. The character strings "gaussian", "binomial", "poisson", "cox", "multinomial", and "mgaussian" are accepted; so are GLM family objects (e.g. `stats::Gamma(link = "log")`, `MASS::negative.binomial(theta = 2)`). Native Cox, multinomial, and mgaussian paths are exercised by the tests but marked **experimental**: more unusual usage (Cox strata, tie handling, time-varying covariates) is not yet validated. Joint theta estimation in the spirit of `MASS::glm.nb()` is out of scope; pass the desired theta to `MASS::negative.binomial()` directly. `lambda` rules are `cv_min / cv_1se / fix`. Rank-deficient designs are handled in the spirit of `stats::glm()`: linearly dependent columns are dropped via a QR pivot, the underlying `glmnet` fit only sees the independent subset, and the dropped columns surface as NA in `coef()` / `summary()`. Novel factor levels in `newdata` at predict time also follow `stats::predict.glm()` by default – an unseen level raises an error. Production scoring pipelines can opt into `predict(fit, newdata, on_new_levels = "na")` to set affected rows to NA (with a warning) instead. Heavier features (split / selective inference) are tracked in `TODO.md`.

Value

An object of class `c("fbrglm", "regularized_glm")` with fields including `family` (the value passed to `glmnet` – a string or a family object), `family_name` (a short display string), `weights`, `offset`, `alpha`, `lambda_rule`, `lambda_value`, `infer`, `selection_frac`, `fit` (the underlying `glmnet` object), `cv_fit` (`cv.glmnet`, or NULL for `lambda = "fix"`), `coefficients`, `nonzero`, `terms`, `xlevels`, `contrasts`, `x_colnames`, `x_train`, `nobs_info` (`n_total` / `n_dropped_missing` / `n_used`), and `rank_info` (`rank` / `ncol` / `rank_deficient` / `pivot` / `kept_cols` / `dropped_cols`). When the design is rank-deficient, linearly dependent columns are dropped before fitting (in the spirit of `stats::glm()`); their entries in `coefficients` are reported as NA to distinguish "not identifiable" from "shrunk to zero by penalty".

Index

`as_cv_glmnet`, [2](#)

`as_glmnet`, [2](#)

`coef.fbrglm(fbrglm)`, [3](#)

`fbrglm`, [3](#)

`glmnet::cv.glmnet()`, [3](#), [4](#)

`glmnet::glmnet()`, [2–4](#)

`nobs.fbrglm(fbrglm)`, [3](#)

`plot.fbrglm(fbrglm)`, [3](#)

`predict.fbrglm(fbrglm)`, [3](#)

`print.fbrglm(fbrglm)`, [3](#)

`print.summary.fbrglm(fbrglm)`, [3](#)

`stats::glm()`, [3](#), [4](#)

`stats::predict.glm()`, [4](#)

`summary.fbrglm(fbrglm)`, [3](#)