

Package: gcloglog (via r-universe)

October 27, 2026

Title Methods for Fitting Binomial Models with the Generalized
Complementary Log-Log Function

Version 1.0

Date 2026-10-16

Maintainer Bert van der Veen <bert_van_der_veen@hotmail.com>

Description Functionality for fitting a flexible link function in
binomial regression based on a latent negative-binomial count.
The original link function was proposed by Aranda-Ordaz
(1981)<[doi:10.1093/biomet/68.2.357](https://doi.org/10.1093/biomet/68.2.357)>, and motivation via a
negative-binomial latent count by van der Veen and Hui (in
prep).

License GPL (>= 2)

Imports graphics, lme4, Matrix, methods, nloptr, numDeriv, stats,
utils

Encoding UTF-8

RoxygenNote 7.3.3

URL <https://github.com/BertvanderVeen/gcloglog>

BugReports <https://github.com/BertvanderVeen/gcloglog/issues>

NeedsCompilation no

Author Bert van der Veen [aut, cre], Francis K.C. Hui [aut]

Repository <https://cran.r-universe.dev>

Date/Publication 2026-10-27 15:50:02 UTC

RemoteUrl <https://github.com/cran/gcloglog>

RemoteRef HEAD

RemoteSha 0013dda443905e8aa21d347522a285c45cd605b1

Contents

make.gcloglog	2
profile.gcloglog	3
profile.phi.glm	5

Index**8**

make.gcloglog	<i>Create a Generalized complementary log-log link-function for GLM-type family</i>
---------------	---

Description

This function is used with "family" functions in "glm" type modeling functions.

Usage

```
make.gcloglog(phi)
```

Arguments

phi value for the dispersion parameter the link is evaluated for.

Value

An object of class "link-glm, a list with (amongst others) the components:

linkfun: Link function function(mu) for fixed phi

linkinv: Inverse link function function(eta) for fixed phi

variance: variance function

mu.eta: Derivative function(eta)

valideta: function(eta)TRUE

validmu: function(mu) that ensures $0 < p < 1$

name: the name for the link gcloglog(phi)

Author(s)

Bert van der Veen

References

van der Veen and Hui (2025). In prep.

Examples

```
# Define logit link via gcloglog
library(gcloglog)
gcloglog1 <- make.gcloglog(1)

# Fit logistic regression with gcloglog
x <- rnorm(100)
b1 = 2.3
b0 = 0.1
```

```

eta = b0+b1*x
y = rbinom(length(x), 1, plogis(eta))
data <- data.frame(y = y,x = x)
model <- glm(y~x, family = binomial(link = gcloglog1), data = data)

```

profile.gcloglog

MLE estimation of the shape of a binomial link function

Description

This function takes a model object, estimates a parameter ϕ that controls the shape of the link function, and performs further profiling for retrieving a confidence interval. The implementation heavily borrows from `tmbprofile` in the **TMB** package.

Usage

```

## S3 method for class 'gcloglog'
profile(
  model,
  CI = TRUE,
  alpha = 0.05,
  plot = TRUE,
  h = 0.02,
  ytol = 2,
  ystep = 0.1,
  maxit = 100,
  adaptive = TRUE,
  trace = TRUE,
  warmstart = TRUE,
  ...
)

```

Arguments

<code>model</code>	a model object which accepts a family argument (should correspond to "binomial") with a link function. The model should have corresponding <code>"update"</code> and <code>"logLik"</code> functionality implemented.
<code>CI</code>	logical, defaults to TRUE.
<code>alpha</code>	the confidence level, defaults to 0.05.
<code>plot</code>	logical, defaults to TRUE, so that a plot of the profiling is displayed. Only when CI is set to TRUE.
<code>h</code>	initial adaptive stepsize.
<code>ytol</code>	Adjusts the range of the likelihood values.
<code>ystep</code>	Adjusts the resolution of the likelihood profile.

<code>maxit</code>	Maximum number of iterations for the adaptive algorithm.
<code>adaptive</code>	logical, defaults to TRUE. Implements adaptive step size.
<code>trace</code>	logical, defaults to TRUE. Prints progress.
<code>warmstart</code>	logical, defaults to TRUE. If TRUE, each trial phi is fit starting from the coefficients of the best-fitting model found so far, rather than always restarting from the coefficients of the initial model.
<code>...</code>	other arguments passed to <code>"profile.phi"</code> .

Details

The final model inherits from class `"gcloglog"`, so that its `vcov`, `summary`, and `confint` return standard errors that account for the estimation of phi; `vcov(..., correct = FALSE)` and `summary(..., correct = FALSE)` give those conditional on phi. `confint` defaults to Wald confidence intervals from the corrected covariance; `method = "profile"` (and `"boot"` for glmer fits) are other options conditional on phi.

For a `glm`, the corrected asymptotic covariance is the naive covariance conditional on the estimated link shape, plus a rank-one update, which follows from the joint information of the coefficients and $\log(\phi)$. For a `glmer` fit, it is the fixed-effect block of the inverse of the joint Hessian of the Laplace-approximated log-likelihood in the random-effect parameters (lme4's θ), the fixed effects, and $\log(\phi)$, obtained by finite differences (`hessian`). Random-effect parameters at their boundary (a singular fit) are held fixed.

The correction does not apply when the estimate of phi is at the boundary $\phi = 0$, where the fit is equivalent to cloglog regression (detected as the fit not improving on the log-likelihood of the cloglog link at the same parameter estimates by more than $1e-5$), or when the joint information is not positive definite. In those cases the standard errors are conditional on phi, with a warning from `vcov` and `confint`, and a note in the printed summary.

Value

A list including the optimisation results, CI, and final model fit.

Author(s)

Bert van der Veen

References

van der Veen and Hui (2025). In prep.

Examples

```
# Define logit link via gcloglog
library(gcloglog)
gcloglog1 <- make.gcloglog(1)

# Profile the gcloglog link
x <- rnorm(100)
b1 = 2.3
```

```

b0 = 0.1
eta = b0+b1*x
y = rbinom(length(x), 1, plogis(eta))
data <- data.frame(y = y,x = x)
model <- glm(y~x, family = binomial(link = gcloglog1), data = data)
res <- profile.gcloglog(model)

final.model <- res$final.model
summary(final.model)          # standard errors account for the estimate of phi
confint(final.model)
summary(final.model)          # conditional on the estimate of phi
vcov(final.model, correct = FALSE) # conditional on the estimate of phi

```

profile.phi.glm	<i>MLE estimation of the shape of a binomial link function</i>
-----------------	--

Description

This function takes a model object with a family argument, and estimates a parameter phi that controls the shape of the link function.

Usage

```

## S3 method for class 'glm'
profile.phi(
  model,
  y,
  optimizer = nlminb,
  optControl = list(iter.max = 500, eval.max = 500),
  optArgs = list(lower = log(1.400882e-08)),
  warmstart = TRUE,
  ...
)

## S3 method for class 'merMod'
profile.phi(
  model,
  y,
  optimizer = bobyqa,
  optControl = list(maxit = 500),
  optArgs = NULL,
  warmstart = TRUE,
  ...
)

## Default S3 method:
profile.phi(
  model,

```

```

    y,
    optimizer = bobyqa,
    optControl = list(maxit = 500),
    optArgs = NULL,
    ...
)

profile.phi(model, ...)

```

Arguments

model	a model object which accepts a family argument (should correspond to "binomial") with a link function. The model should have corresponding "update" and "logLik" functionality implemented.
y	response variable.
optimizer	numerical optimisation routine used to estimate phi. Must accept the starting value, objective function, gradient, and control arguments (in that order). Defaults to "optim".
optControl	control arguments passed to optimizer.
optArgs	other arguments passed on to the optimizer as a list.
warmstart	logical, defaults to TRUE. If TRUE, each trial phi is fit starting from the coefficients of the best-fitting model found so far in the search, rather than always restarting from the coefficients of the initial model.
...	other arguments passed to "update".

Value

The optimisation results of the profiling, with log(phi) as the estimated parameter.

Author(s)

Bert van der Veen

References

van der Veen and Hui (2025). In prep.

Examples

```

# Define logit link via gcloglog
library(gcloglog)
gcloglog1 <- make.gcloglog(1)

# Fit logistic regression with gcloglog
x <- rnorm(100)
b1 = 2.3
b0 = 0.1
eta = b0+b1*x
y = rbinom(length(x), 1, plogis(eta))

```

```
data <- data.frame(y = y, x = x)
model <- glm(y~x, family = binomial(link = gcloglog1), data = data)

# Find optimal link shape by profiling phi
res <- profile.phi(model, y = model.response(model.frame(model)))
exp(res$optr$par) # extract phi
```

Index

family, [2](#)

glm, [2](#)

hessian, [4](#)

logLik, [3](#), [6](#)

make.gcloglog, [2](#)

optim, [6](#)

profile.gcloglog, [3](#)

profile.phi, [4](#)

profile.phi (profile.phi.glm), [5](#)

profile.phi.glm, [5](#)

update, [3](#), [6](#)