Package ‘dglm’

December 19, 2022

Version 1.8.5
Date 2022-12-19
Title Double Generalized Linear Models
Author Gordon Smyth, Peter K Dunn <pdunn2@usc.edu.au>, Robert W. Corty
Maintainer Gordon Smyth <smyth@wehi.edu.au>
Depends R (>= 2.8.0)
Imports statmod, stats
Description Model fitting and evaluation tools for double generalized linear models (DGLMs). This class of models uses one generalized linear model (GLM) to fit the specified response and a second GLM to fit the deviance of the first model.
License GPL-2 | GPL-3
NeedsCompilation no
Repository CRAN
Date/Publication 2022-12-19 11:20:02 UTC

R topics documented:

anova.dglm .......................................................... 2
dglm ................................................................. 3
dglm-class .......................................................... 6
dglm.control ......................................................... 9
residuals.dglm ...................................................... 10
summary.dglm ....................................................... 11

Index 14
Description

Compute an analysis of deviance table for one or more double generalized linear model fits.

Usage

```r
## S3 method for class 'dglm'
anova(object, ...)
```

Arguments

- `object`: objects of class `dglm`, typically the result of a call to `dglm`.
- `...`: Not used.

Details

Specifying a single object gives sequential and adjusted likelihood ratio tests for the mean and dispersion model components of the fit. The aim is to test overall significance for the mean and dispersion components of the double generalized linear model fit. The sequential tests (i) set both mean and dispersion models constant, add the mean model and (ii) sequentially add the dispersion model. The adjusted tests determine whether the mean and dispersion models can be set constant separately.

Value

An object of class "anova" inheriting from class "data.frame".

Warning

The anova method is questionable when applied to an "dglm" object with `method="rem1"` (stick to `method="ml"`).

Author(s)

Gordon Smyth, ported to R by Peter Dunn (<pdunn2@usc.edu.au>)

References


**See Also**

`dglm`, `anova`.

**Examples**

```r
# Continuing the example from glm, but this time try
# fitting a Gamma double generalized linear model also.
clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))

# The same example as in glm: the dispersion is modeled as constant
out <- dglm(lot1 ~ log(u), ~1, data=clotting, family=Gamma)
summary(out)

# Try a double glm
out2 <- dglm(lot1 ~ log(u), ~u, data=clotting, family=Gamma)
summary(out2)
anova(out2)
```

---

**dglm**

**Double Generalized Linear Models**

**Description**

Fits a generalized linear model with a link-linear model for the dispersion as well as for the mean.

**Usage**

```r
dglm(formula=formula(data), dformula = ~ 1, family = gaussian, dlink = "log", data = parent.frame(), subset = NULL, weights = NULL, contrasts = NULL, method = "ml", mustart = NULL, betastart = NULL, etastart = NULL, phistart = NULL, control = dglm.control(...), ykeep = TRUE, xkeep = FALSE, zkeep = FALSE, ...) 
dglm.constant(y, family, weights = 1)
```
Arguments

- **formula**: a symbolic description of the model to be fit. The details of model specification are found in `dglm`.
- **dformula**: a formula expression of the form `~ predictor`, the response being ignored. This specifies the linear predictor for modelling the dispersion. A term of the form `offset(expression)` is allowed.
- **family**: a description of the error distribution and link function to be used in the model. See `glm` for more information.
- **link**: link function for modelling the dispersion. Any link function accepted by the quasi family is allowed, including `power(x)`. See details below.
- **data**: an optional data frame containing the variables in the model. See `glm` for more information.
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process.
- **contrasts**: an optional list. See the `contrasts.arg` of `model.matrix.default`.
- **method**: the method used to estimate the dispersion parameters; the default is "reml" for restricted maximum likelihood and the alternative is "ml" for maximum likelihood. Upper case and partial matches are allowed.
- **mustart**: numeric vector giving starting values for the fitted values or expected responses. Must be of the same length as the response, or of length 1 if a constant starting vector is desired. Ignored if `betastart` is supplied.
- **betastart**: numeric vector giving starting values for the regression coefficients in the link-linear model for the mean.
- **etastart**: numeric vector giving starting values for the linear predictor for the mean model.
- **phistart**: numeric vector giving starting values for the dispersion parameters.
- **control**: a list of iteration and algorithmic constants. See `dglm.control` for their names and default values. These can also be set as arguments to `dglm` itself.
- **ykeep**: logical flag: if TRUE, the vector of responses is returned.
- **xkeep**: logical flag: if TRUE, the `model.matrix` for the mean model is returned.
- **zkeep**: logical flag: if TRUE, the `model.matrix` for the dispersion model is returned.
- **...**: further arguments passed to or from other methods.
- **y**: numeric response vector

Details

Write $\mu_i = E[y_i]$ for the expectation of the $i$th response. Then $\text{Var}[Y_i] = \phi_i V(\mu_i)$ where $V$ is the variance function and $\phi_i$ is the dispersion of the $i$th response (often denoted as the Greek character ‘phi’). We assume the link linear models $g(\mu_i) = x_i^T b$ and $h(\phi_i) = z_i^T a$, where $x_i$ and $z_i$ are vectors of covariates, and $b$ and $a$ are vectors of regression coefficients affecting the mean and dispersion respectively. The argument `dlink` specifies $h$. See `family` for how to specify $g$. The optional arguments `mustart`, `betastart` and `phistart` specify starting values for $\mu_i$, $b$ and $\phi_i$ respectively.
The parameters $b$ are estimated as for an ordinary GLM. The parameters $a$ are estimated by way of a dual GLM in which the deviance components of the ordinary GLM appear as responses. The estimation procedure alternates between one iteration for the mean submodel and one iteration for the dispersion submodel until overall convergence.

The output from dglm, out say, consists of two glm objects (that for the dispersion submodel is `out$dispersion.fit`) with a few more components for the outer iteration and overall likelihood. The summary and anova functions have special methods for dglm objects. Any generic function that has methods for glms or lms will work on out, giving information about the mean submodel. Information about the dispersion submodel can be obtained by using `out$dispersion.fit` as argument rather than out itself. In particular `drop1(out, scale=1)` gives correct score statistics for removing terms from the mean submodel, while `drop1(out$dispersion.fit, scale=2)` gives correct score statistics for removing terms from the dispersion submodel.

The dispersion submodel is treated as a gamma family unless the original responses are gamma, in which case the dispersion submodel is digamma. This is exact if the original glm family is gaussian, Gamma or inverse.gaussian. In other cases it can be justified by the saddle-point approximation to the density of the responses. The results will therefore be close to exact ML or REML when the dispersions are small compared to the means. In all cases the dispersion submodel has prior weights 1, and has its own dispersion parameter which is 2.

Value

an object of class dglm is returned, which inherits from glm and lm. See `dglm-class` for details.

Note

The anova method is questionable when applied to an dglm object with method="reml" (stick to method="ml").

Author(s)

Gordon Smyth, ported to R by Peter Dunn

References


See Also

`dglm-class`, `dglm.control`, Digamma family, Polygamma
Examples

# Continuing the example from glm, but this time try
# fitting a Gamma double generalized linear model also.
clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))

# The same example as in glm: the dispersion is modelled as constant
# However, dglm uses ml not reml, so the results are slightly different:
out <- dglm(lot1 ~ log(u), ~1, data=clotting, family=Gamma)
summary(out)

# Try a double glm
out2 <- dglm(lot1 ~ log(u), ~u, data=clotting, family=Gamma)
summary(out2)
anova(out2)

# Summarize the mean model as for a glm
summary.glm(out2)

# Summarize the dispersion model as for a glm
summary(out2$dispersion.fit)

# Examine goodness of fit of dispersion model by plotting residuals
plot(fitted(out2$dispersion.fit),residuals(out2$dispersion.fit))

---

dglm-class

Double Generalized Linear Model - class

Description

Class of objects returned by fitting double generalized linear models.

Details

Write \( \mu_i = \mathbb{E}[y_i] \) for the expectation of the \( i \)th response. Then \( \text{Var}[Y_i] = \phi_i V(\mu_i) \) where \( V \) is the variance function and \( \phi_i \) is the dispersion of the \( i \)th response (often denoted as the Greek character ‘phi’). We assume the link linear models \( g(\mu_i) = x_i^T b \) and \( h(\phi_i) = z_i^T a \), where \( x_i \) and \( z_i \) are vectors of covariates, and \( b \) and \( a \) are vectors of regression coefficients affecting the mean and dispersion respectively. The argument dlink specifies \( h \). See family for how to specify \( g \). The optional arguments mustart, betastart and phistart specify starting values for \( \mu_i \), \( b \) and \( \phi_i \) respectively.

The parameters \( b \) are estimated as for an ordinary GLM. The parameters \( a \) are estimated by way of a dual GLM in which the deviance components of the ordinary GLM appear as responses. The estimation procedure alternates between one iteration for the mean submodel and one iteration for the dispersion submodel until overall convergence.
The output from dglm, out say, consists of two glm objects (that for the dispersion submodel is out$dispersion.fit) with a few more components for the outer iteration and overall likelihood. The summary and anova functions have special methods for dglm objects. Any generic function which has methods for glms or lms will work on out, giving information about the mean submodel. Information about the dispersion submodel can be obtained by using out$dispersion.fit as argument rather than out itself. In particular drop1(out, scale=1) gives correct score statistics for removing terms from the mean submodel, while drop1(out$dispersion.fit, scale=2) gives correct score statistics for removing terms from the dispersion submodel.

The dispersion submodel is treated as a gamma family unless the original responses are gamma, in which case the dispersion submodel is digamma. (Note that the digamma and trigamma functions are required to fit a digamma family.) This is exact if the original GLM family is gaussian, Gamma or inverse.gaussian. In other cases it can be justified by the saddle-point approximation to the density of the responses. The results will therefore be close to exact ML or REML when the dispersions are small compared to the means. In all cases the dispersion submodel as prior weights 1, and has its own dispersion parameter which is 2.

Generation

This class of objects is returned by the dglm function to represent a fitted double generalized linear model. Class "dglm" inherits from class "glm", since it consists of two coupled generalized linear models, one for the mean and one for the dispersion. Like glm, it also inherits from lm. The object returned has all the components of a glm object. The returned component object$dispersion.fit is also a glm object in its own right, representing the result of modelling the dispersion.

Methods

Objects of this class have methods for the functions print, plot, summary, anova, predict, fitted, drop1, add1, and step, amongst others. Specific methods (not shared with glm) exist for summary and anova.

Structure

A dglm object consists of a glm object with the following additional components:

- dispersion.fit: the dispersion submodel: a glm object representing the fitted model for the dispersions. The responses for this model are the deviance components from the original generalized linear model. The prior weights are 1 and the dispersion or scale of this model is 2.
- iter: this component now represents the number of outer iterations used to fit the coupled mean-dispersion models. At each outer iteration, one IRLS is done for each of the mean and dispersion submodels.
- method: fitting method used: "ml" if maximum likelihood was used or "reml" if adjusted profile likelihood was used.
- m2loglik: minus twice the log-likelihood or adjusted profile likelihood of the fitted model.

Note

The anova method is questionable when applied to an dglm object with method="reml" (stick to method="ml").
Author(s)

Gordon Smyth, ported to R by Peter Dunn (<pdunn2@usc.edu.au>)

References


See Also

dglm, Digamma family, Polygamma

Examples

# Continuing the example from glm, but this time try
# fitting a Gamma double generalized linear model also.
clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))

# The same example as in glm: the dispersion is modelled as constant
out <- dglm(lot1 ~ log(u), ~1, data=clotting, family=Gamma)
summary(out)

# Try a double glm
out2 <- dglm(lot1 ~ log(u), ~u, data=clotting, family=Gamma)
summary(out2)
anova(out2)

# Summarize the mean model as for a glm
summary.glm(out2)

# Summarize the dispersion model as for a glm
summary(out2$dispersion.fit)

# Examine goodness of fit of dispersion model by plotting residuals
plot(fitted(out2$dispersion.fit),residuals(out2$dispersion.fit))
dglm.control

Auxiliary for controlling double glm fitting

Description

Auxiliary function as user interface for fitting double generalized linear models. Typically only used when calling dglm.

Usage

dglm.control(epsilon = 1e-007, maxit = 50, trace = FALSE, ...)

Arguments

epsilon
  positive convergence tolerance epsilon; the iterations converge when \((|L_o - L|)/(|L_o| + 1) > \epsilon\), where \(L_o\) is minus twice the values of log-likelihood on the previous iteration, and \(L\) is minus twice the values of log-likelihood on the current.

maxit
  integer giving the maximal number of outer iterations of the alternating iterations.

trace
  logical indicating if (a small amount of) output should be produced for each iteration.

...
  not currently implemented

Details

When 'trace' is true, calls to 'cat' produce the output for each outer iteration. Hence, 'options(digits = *)' can be used to increase the precision; see the example for glm.control.

Author(s)

Gordon Smyth, ported to R\ by Peter Dunn (<pdunn2@usc.edu.au>)

References


See Also
dglm-class, dglm
### Examples

#### A variation on example(dglm):
Continuing the example from glm, but this time try
fitting a Gamma double generalized linear model also.

```r
clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))

# The same example as in glm: the dispersion is modelled as constant
out <- dglm(lot1 ~ log(u), ~1, data=clotting, family=Gamma)
summary(out)

# Try a double glm
oo <- options()
options(digits=12) # See more details in tracing
out2 <- dglm(lot1 ~ log(u), ~u, data=clotting, family=Gamma,
  control=dglm.control(epsilon=0.01, trace=TRUE))
# With this value of epsilon, convergence should be quicker
# and the results less reliable (compare to example(dglm))
summary(out2)
options(oo)
```

---

**residuals.dglm**

*Extract Residuals from Double Generalized Linear Model Fit*

**Description**

This implements the 'residuals' generic for the dglm object.

**Usage**

```r
## S3 method for class 'dglm'
residuals(object, ...)  
```

**Arguments**

- `object`: a "dglm" object
- `...`: any other parameters are passed to `residuals.glm`

**Value**

Numeric vector of residuals from the mean submodel.

**Author(s)**

Robert W. Corty and Gordon Smyth
**summary.dglm**

---

**Summarize Double Generalized Linear Model Fit**

### Description

These functions are all methods for class dglm or summary.glm objects.

### Usage

```r
## S3 method for class 'dglm'
summary(object, dispersion=NULL, correlation = FALSE, ...)
```

### Arguments

- **object**: an object of class "dglm", usually, a result of a call to glm.
- **dispersion**: the dispersion parameter for the fitting family. By default it is obtained from object.
- **correlation**: logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
- **...**: further arguments to be passed to summary.glm

### Details

For more details, see `summary.glm`.

If more than one of etastart, start and mustart is specified, the first in the list will be used.

### Value

`summary.dglm` returns an object of class "summary.dglm", a list with components

- **call**: the component from object
- **terms**: the component from object
- **family**: the component from object
- **deviance**: the component from object
- **aic**: NULL here
- **contrasts** (where relevant) the contrasts used. NOT WORKING??
- **df.residual**: the component from object
- **null.deviance**: the component from object
- **df.null**: the residual degrees of freedom for the null model.
- **iter**: the component from object
- **deviance.resid** the deviance residuals: see residuals.glm
- **coefficients**: the matrix of coefficients, standard errors, z-values and p-values. Aliased coefficients are omitted.
summary.dglm

aliased

dispersion
either the supplied argument or the estimated dispersion if the latter in NULL

df
a 3-vector of the rank of the model and the number of residual degrees of freedom, plus number of non-aliased coefficients.

cov.unscaled
the unscaled \((\text{dispersion} = 1)\) estimated covariance matrix of the estimated coefficients.

cov.scaled
ditto, scaled by \text{dispersion}

correlation
(only if \text{correlation} is true.) The estimated correlations of the estimated coefficients.

dispersion.summary
the summary of the fitted dispersion model

outer.iter
the number of outer iteration of the alternating iterations

m2loglik
minus twice the log-likelihood of the fitted model

Note
The anova method is questionable when applied to an \text{dglm} object with \text{method=\texttt{\"rem1\\"}} (stick to \text{method=\texttt{\"ml\"}}).

Author(s)
Gordon Smyth, ported to \text{R} by Peter Dunn (\texttt{<pdunn2@usc.edu.au>})

References


See Also
\texttt{dglm-class}, \texttt{dglm.control}, \texttt{anova.dglm}, \texttt{summary.glm}

Examples
# Continuing the example from \texttt{glm}, but this time try
# fitting a Gamma double generalized linear model also.
clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))
# The same example as in glm: the dispersion is modelled as constant
out <- dglm(lot1 ~ log(u), ~1, data=clotting, family=Gamma)
summary(out)

# Try a double glm
out2 <- dglm(lot1 ~ log(u), ~u, data=clotting, family=Gamma)
summary(out2)
anova(out2)

# Summarize the mean model as for a glm
summary.glm(out2)

# Summarize the dispersion model as for a glm
summary(out2$dispersion.fit)
Index

* classes
dglm-class, 6

* models
  anova.dglm, 2
dglm, 3
dglm.control, 9
  summary.dglm, 11

* optimize
dglm.control, 9

* regression
  anova.dglm, 2
dglm, 3
dglm-class, 6

anova, 3
anova.dglm, 2, 12
dglm, 2, 3, 3, 4, 7–9
dglm-class, 6
dglm.control, 4, 5, 9, 12

family, 4, 6
glm, 4
glm.control, 9
model.matrix.default, 4
print.summary.dglm (summary.dglm), 11
residuals.dglm, 10
summary.dglm, 11
summary.glm, 11, 12