

Package ‘dglm’

February 14, 2012

Version 1.6.1

Date 2009-02-05

Title Double generalized linear models

Author Peter K Dunn <pdunn2@usc.edu.au> and Gordon K Smyth

Maintainer Peter K Dunn <pdunn2@usc.edu.au>

Depends R (>= 1.0), statmod(>= 1.0.8)

Description Fitting double generalized linear models

License GPL (>= 2)

Repository CRAN

Date/Publication 2009-02-06 12:30:07

R topics documented:

anova.dglm	2
dglm	3
dglm.control	6
dglm.object	8
summary.dglm	10
Index	13

`anova.dglm`*Analysis of Deviance for Double Generalized Linear Model Fits*

Description

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

Usage

```
## S3 method for class 'dglm'  
anova(object, ..., dispersion = NULL, test = NULL)
```

Arguments

<code>object</code>	objects of class <code>dglm</code> , typically the result of a call to <code>dglm</code> .
<code>dispersion</code>	the dispersion parameter for the fitting family. By default it is obtained from <code>glm.obj</code> .
<code>test</code>	a character string, (partially) matching one of "Chisq", "F" or "Cp". See <code>stat.anova</code> .
<code>...</code>	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="reml"` (stick to `method="ml"`).

Author(s)

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

References

Hastie, T. J. and Pregibon, D. (1992) *Generalized linear models*. Chapter 6 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Smyth, G. K. (1989). Generalized linear models with varying dispersion. *J. R. Statist. Soc. B*, **51**, 47–60.

Smyth, G. K., and Verbyla, A. P. (1999). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics*, **10**, 696-709.

Verbyla, A. P., and Smyth, G. K. (1998). Double generalized linear models: approximate residual maximum likelihood and diagnostics. Research Report, Department of Statistics, University of Adelaide.

See Also

[dglm](#), [anova](#).

Examples

```
# Continuing the example from glm, but this time try
# fitting a Gamma double generalized linear model also.
library(statmod)
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)
```

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

```
dglm(formula=formula(data), dformula = ~ 1, family = gaussian, dlink = "log",
data = sys.parent(), 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 <code>~ predictor</code> , the response being ignored. This specifies the linear predictor for modelling the dispersion. A term of the form <code>offset(expression)</code> is allowed.
family	a description of the error distribution and link function to be used in the model. See glm for more information.
dlink	link function for modelling the dispersion. Any link function accepted by the quasi family is allowed, including <code>power(x)</code> . 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 <code>contrasts.arg</code> 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 <code>betastart</code> 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 <code>dglm</code> itself.
ykeep	logical flag: if TRUE, the vector of responses is returned.
xkeep	logical flag: if TRUE, the <code>model.matrix</code> for the mean model is returned.
zkeep	logical flag: if TRUE, the <code>model.matrix</code> 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 ϕ_i is the dispersion of the i th response (often denoted as the Greek character ‘phi’). We assume the link linear models $g(\mu_i) = \mathbf{x}_i^T \mathbf{b}$ and $h(\phi_i) = \mathbf{z}_i^T \mathbf{a}$, where \mathbf{x}_i and \mathbf{z}_i are vectors of covariates, and \mathbf{b} and \mathbf{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 μ_i , \mathbf{b} and ϕ_i respectively.

The parameters \mathbf{b} are estimated as for an ordinary GLM. The parameters \mathbf{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 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.object](#) 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 (<pdunn2@usc.edu.au>)

References

- Smyth, G. K. (1989). Generalized linear models with varying dispersion. *J. R. Statist. Soc. B*, **51**, 47–60.
- Smyth, G. K., and Verbyla, A. P. (1999). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics*, **10**, 696-709.

Verbyla, A. P., and Smyth, G. K. (1998). Double generalized linear models: approximate residual maximum likelihood and diagnostics. Research Report, Department of Statistics, University of Adelaide.

See Also

[dglm.object](#), [dglm.control](#), Digamma family, Polygamma

Examples

```
# Continuing the example from glm, but this time try
# fitting a Gamma double generalized linear model also.
library(statmod)
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 used ml not reml, so results 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.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

- Smyth, G. K. (1989). Generalized linear models with varying dispersion. *J. R. Statist. Soc. B*, **51**, 47–60.
- Smyth, G. K., and Verbyla, A. P. (1999). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics*, **10**, 696-709.
- Verbyla, A. P., and Smyth, G. K. (1998). Double generalized linear models: approximate residual maximum likelihood and diagnostics. Research Report, Department of Statistics, University of Adelaide.

See Also

[dglm.object](#), [dglm](#)

Examples

```
### A variation on example(dglm) :
# 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
```

```

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)

```

dglm.object

Double generalized linear model object

Description

Class of objects returned by fitting double generalized linear models.

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 ϕ_i is the dispersion of the i th response (often denoted as the Greek character ‘phi’). We assume the link linear models $g(\mu_i) = \mathbf{x}_i^T \mathbf{b}$ and $h(\phi_i) = \mathbf{z}_i^T \mathbf{z}$, where \mathbf{x}_i and \mathbf{z}_i are vectors of covariates, and \mathbf{b} and \mathbf{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 `mstart`, `betastart` and `phistart` specify starting values for μ_i , \mathbf{b} and ϕ_i respectively.

The parameters \mathbf{b} are estimated as for an ordinary GLM. The parameters \mathbf{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

- Smyth, G. K. (1989). Generalized linear models with varying dispersion. *J. R. Statist. Soc. B*, **51**, 47–60.
- Smyth, G. K., and Verbyla, A. P. (1999). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics*, **10**, 696-709.
- Verbyla, A. P., and Smyth, G. K. (1998). Double generalized linear models: approximate residual maximum likelihood and diagnostics. Research Report, Department of Statistics, University of Adelaide.

See Also

[dglm.object](#), 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))
```

summary.dglm

Summarizing double generalized linear model fits

Description

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

Usage

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

Arguments

<code>object</code>	an object of class "dglm", usually, a result of a call to <code>glm</code> .
<code>dispersion</code>	the dispersion parameter for the fitting family. By default it is obtained from <code>object</code> .

correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
...	further arguments to be passed to <code>summary.glm</code>

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

<code>call</code>	the component from object
<code>terms</code>	the component from object
<code>family</code>	the component from object
<code>deviance</code>	the component from object
<code>aic</code>	NULL here
<code>contrasts</code>	(where relevant) the contrasts used. NOT WORKING??
<code>df.residual</code>	the component from object
<code>null.deviance</code>	the component from object
<code>df.null</code>	the residual degrees of freedom for the null model.
<code>iter</code>	the component from object
<code>deviance.resid</code>	the deviance residuals: see <code>residuals.glm</code>
<code>coefficients</code>	the matrix of coefficients, standard errors, z -values and p -values. Aliased coefficients are omitted.
<code>aliased</code>	named logical vector showing if the original coefficients are aliased.
<code>dispersion</code>	either the supplied argument or the estimated dispersion if the latter is NULL
<code>df</code>	a 3-vector of the rank of the model and the number of residual degrees of freedom, plus number of non-aliased coefficients.
<code>cov.unscaled</code>	the unscaled (<code>dispersion = 1</code>) estimated covariance matrix of the estimated coefficients.
<code>cov.scaled</code>	ditto, scaled by dispersion
<code>correlation</code>	(only if <code>correlation</code> is true.) The estimated correlations of the estimated coefficients.
<code>dispersion.summary</code>	the summary of the fitted dispersion model
<code>outer.iter</code>	the number of outer iteration of the alternating iterations
<code>m2loglik</code>	minus twice the log-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

Smyth, G. K. (1989). Generalized linear models with varying dispersion. *J. R. Statist. Soc. B*, **51**, 47–60.

Smyth, G. K., and Verbyla, A. P. (1999). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics*, **10**, 696-709.

Verbyla, A. P., and Smyth, G. K. (1998). Double generalized linear models: approximate residual maximum likelihood and diagnostics. Research Report, Department of Statistics, University of Adelaide.

See Also

[dglm.object](#), [dglm.control](#), [anova.dglm](#), [summary.glm](#)

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)
```

Index

*Topic **models**

- anova.dglm, 2
- dglm, 3
- dglm.control, 6
- dglm.object, 8
- summary.dglm, 10

*Topic **optimize**

- dglm.control, 6

*Topic **regression**

- anova.dglm, 2
- dglm, 3
- dglm.object, 8

anova, 3
anova.dglm, 2, 12

dglm, 2, 3, 3, 4, 7, 9
dglm.control, 4, 6, 6, 12
dglm.object, 5–7, 8, 10, 12

family, 5, 8

glm, 4
glm.control, 7

model.matrix.default, 4

print.summary.dglm(summary.dglm), 10

stat.anova, 2
summary.dglm, 10
summary.glm, 11, 12