Package 'addreg'

October 12, 2022

Title Additive Regression for Discrete Data

Description Methods for fitting identity-link GLMs and GAMs to discrete data, using EM-type algorithms with more stable convergence properties than standard methods.

Version 3.0

Date 2017-12-20

Depends R (>= 3.0.1)

Imports splines, combinat, glm2, turboEM

License GPL (>= 2)

LazyData true

URL https://github.com/mdonoghoe/addreg

NeedsCompilation no

Author Mark W. Donoghoe [aut, cre], Ian C. Marschner [ths]

Maintainer Mark W. Donoghoe <markdonoghoe@gmail.com>

Repository CRAN

Date/Publication 2017-12-23 23:22:06 UTC

R topics documented:

`addreg-package` *Additive Regression for Discrete Data*

Description

Methods for fitting identity-link GLMs and GAMs to discrete data, using EM-type algorithms with more stable convergence properties than standard methods.

Details

Package:	addreg
Type:	Package
Version:	3.0
Date:	2017-12-20
License:	GPL (>= 2)

This package provides methods to fit generalised linear models (GLMs) and generalised additive models (GAMs) with identity link functions to discrete data using binomial, Poisson and negative binomial models. It is planned that future versions will incorporate other types of discrete data models, such as multinomial regression.

The package has two primary functions: `addreg` and `addreg.smooth`, together with various supporting functions. It is useful in two main situations. The first is when a standard GLM routine, such as `glm`, fails to converge with such a model. The second is when a flexible semi-parametric component is desired in these models. One of the main purposes of this package is to provide parametric and semi-parametric adjustment of risk differences and rate differences.

Two approaches based on the EM algorithm can be used. These methods accommodate the required parameter constraints and are more stable than iteratively reweighted least squares. In a combinatorial EM algorithm (Marschner, 2014), a collection of restricted parameter spaces is defined which covers the full parameter space, and the EM algorithm is applied within each restricted parameter space in order to find a collection of restricted maxima of the log-likelihood function, from which can be obtained the global maximum over the full parameter space.

In the expanded EM approach, additional parameters are added to the model, and an EM algorithm finds the MLE of this overparameterised model by imposing constraints on each individual parameter. This requires a single application of the EM algorithm.

In either case, the EM algorithm may be accelerated by using the capabilities of the **turboEM** package.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

Maintainer: Mark W. Donoghoe <markdonoghoe@gmail.com>

References

Donoghoe, M. W. and I. C. Marschner (2014). Stable computational methods for additive binomial models with application to adjusted risk differences. *Computational Statistics and Data Analysis* 80: 184–196.

Donoghoe, M. W. and I. C. Marschner (2015). Flexible regression models for rate differences, risk differences and relative risks. *International Journal of Biostatistics* 11(1): 91–108.

Donoghoe, M. W. and I. C. Marschner (2016). Estimation of adjusted rate differences using additive negative binomial regression. *Statistics in Medicine* 35(18): 3166–3178.

Marschner, I. C. (2010). Stable computation of maximum likelihood estimates in identity link Poisson regression. *Journal of Computational and Graphical Statistics* 19(3): 666–683.

Marschner, I. C. (2014). Combinatorial EM algorithms. *Statistics and Computing* 24(6): 921–940.

See Also

glm

Examples

```
## For examples, see example(addreg) and example(addreg.smooth)
```

---

addreg *Additive Regression for Discrete Data*

Description

addreg fits additive (identity-link) Poisson, negative binomial and binomial regression models using a stable combinatorial EM algorithm.

Usage

```
addreg(formula, mono = NULL, family, data, standard, subset, na.action,
       start = NULL, offset, control = list(...), model = TRUE,
       method = c("cem", "em"),
       accelerate = c("em", "squarem", "pem", "qn"),
       control.method = list(), warn = TRUE, ...)
```

Arguments

formula	an object of class "formula" (or one that can be coerced into that class): a symbolic description of the model to be fitted. The details of model specification are given under "Details". Note that the model must contain an intercept, and 2nd-order terms (such as interactions) or above are currently not supported — see "Note".
mono	a vector indicating which terms in formula should be restricted to have a monotonically non-decreasing relationship with the outcome. May be specified as names or indices of the terms.
family	a description of the error distribution to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function (see family for details of family functions), but here it is restricted to be poisson, negbin1 or binomial family with identity link.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which addreg is called.
standard	a numeric vector of length equal to the number of cases, where each element is a positive constant that (multiplicatively) standardises the fitted value of the corresponding element of the response vector. Ignored for binomial family (two-column specification of response should be used instead).
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set be the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
start	starting values for the parameters in the linear predictor, also with the starting value for the scale as the last element when family = negbin1.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. This should be NULL or a *non-negative* numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset. Ignored for binomial family; not yet implemented for negative binomial models.
control	list of parameters for controlling the fitting process, passed to addreg.control.
model	a logical value indicating whether the *model frame* (and, for binomial models, the equivalent Poisson model) should be included as a component of the returned value.
method	a character string that determines which algorithm to use to find the MLE: "cem" for the combinatorial EM algorithm, which cycles through a sequence of constrained parameter spaces, or "em" for a single EM algorithm based on an over-parameterised model.

accelerate	a character string that determines the acceleration algorithm to be used, (partially) matching one of "em" (no acceleration — the default), "squarem", "pem" or "qn". See turboem for further details. Note that "decme" is not permitted.
control.method	a list of control parameters for the acceleration algorithm, which are passed to the control.method argument of turboem. If any items are not specified, the defaults are used.
warn	a logical indicating whether or not warnings should be provided for non-convergence or boundary values.
...	arguments to be used to form the default control argument if it is not supplied directly.

Details

addreg fits a generalised linear model (GLM) with a Poisson or binomial error distribution and identity link function, as well as additive NegBin I models (which are not GLMs). Predictors are assumed to be continuous, unless they are of class factor, or are character or logical (in which case they are converted to factors). Specifying a predictor as monotonic using the mono argument means that for continuous terms, the associated coefficient will be restricted to be non-negative, and for categorical terms, the coefficients will be non-decreasing in the order of the factor levels. This allows semi-parametric monotonic regression functions, in the form of unsmoothed step-functions. For smooth regression functions see addreg.smooth.

As well as allowing monotonicity constraints, the function is useful when a standard GLM routine, such as glm, fails to converge with an identity-link Poisson or binomial model. If glm does achieve successful convergence, and addreg converges to an interior point, then the two results will be identical. However, glm may still experience convergence problems even when addreg converges to an interior point. Note that if addreg converges to a boundary point, then it may differ slightly from glm even if glm successfully converges, because of differences in the definition of the parameter space. addreg produces valid fitted values for covariate values within the Cartesian product of the observed range of covariate values, whereas glm produces valid fitted values just for the observed covariate combinations (assuming it successfully converges). This issue is only relevant when addreg converges to a boundary point.

The computational method is a combinatorial EM algorithm (Marschner, 2014), which accommodates the parameter contraints in the model and is more stable than iteratively reweighted least squares. A collection of restricted parameter spaces is defined which covers the full parameter space, and the EM algorithm is applied within each restricted parameter space in order to find a collection of restricted maxima of the log-likelihood function, from which can be obtained the global maximum over the full parameter space. See Marschner (2010), Donoghoe and Marschner (2014) and Donoghoe and Marschner (2016) for further details.

Acceleration of the EM algorithm can be achieved through the methods of the **turboEM** package, specified through the accelerate argument. However, note that these methods do not have the guaranteed convergence of the standard EM algorithm, particularly when the MLE is on the boundary of its (possibly constrained) parameter space.

Value

addreg returns an object of class "addreg", which inherits from classes "glm" and "lm". The function summary.addreg can be used to obtain or print a summary of the results.

The generic accessor functions coefficients, fitted.values and residuals can be used to extract various useful features of the value returned by addreg. Note that effects will not work.

An object of class "addreg" is a list containing the same components as an object of class "glm" (see the "Value" section of glm), but without contrasts, qr, R or effects components. It also includes:

loglik	the maximised log-likelihood.
aic.c	a small-sample corrected version of Akaike's *An Information Criterion* (Hurvich, Simonoff and Tsai, 1998). This is used by addreg.smooth to choose the optimal number of knots for smooth terms.
xminmax	the minimum and maximum observed values for each of the continuous covariates, to help define the covariate space of the model.

As well as, for Poisson and negative binomial models:

nn.coefficients	estimated coefficients associated with the non-negative parameterisation corresponding to the MLE.
nn.x	non-negative model matrix associated with nn.coefficients.
standard	the standard argument.

Or, for binomial models:

model.addpois	if requested, the addreg object for the associated identity-link Poisson model.

The scale component of the result is fixed at 1 for Poisson and binomial models, and is the constant overdispersion parameter for negative binomial models (that is, scale = $1 + \phi$) where $Var(\mu) = (1 + \phi)\mu$).

Note

Due to the way the covariate space is defined in the CEM algorithm, specifying interactions in the formula is not currently supported by addreg. 2-way interactions between factors can be included by calculating a new factor term that has levels corresponding to all possible combinations of the factor levels. See the Example.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

References

Donoghoe, M. W. and I. C. Marschner (2014). Stable computational methods for additive binomial models with application to adjusted risk differences. *Computational Statistics and Data Analysis* 80: 184–196.

Donoghoe, M. W. and I. C. Marschner (2016). Estimation of adjusted rate differences using additive negative binomial regression. *Statistics in Medicine* 35(18): 3166–3178.

Hurvich, C. M., J. S. Simonoff and C.-L. Tsai (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60(2): 271–293.

Marschner, I. C. (2010). Stable computation of maximum likelihood estimates in identity link Poisson regression. *Journal of Computational and Graphical Statistics* 19(3): 666–683.

Marschner, I. C. (2014). Combinatorial EM algorithms. *Statistics and Computing* 24(6): 921–940.

Examples

```
require(glm2)
data(crabs)

#=============================================================================
#  identity-link Poisson model with periodic non-convergence when glm is used
#=============================================================================

crabs.boot <- crabs[crabs$Rep1,-c(5:6)]
crabs.boot$width.shifted <- crabs.boot$Width - min(crabs$Width)

fit.glm <- glm(Satellites ~ width.shifted + factor(Dark) + factor(GoodSpine),
  family = poisson(identity), data = crabs.boot, start = rep(1,4),
  control = glm.control(trace = TRUE))

fit.addreg <- addreg(formula(fit.glm), family = poisson, data = crabs.boot,
  trace = 1)

# Speed up convergence by using single EM algorithm
fit.addreg.em <- update(fit.addreg, method = "em")

# Speed up convergence by using acceleration methods
fit.addreg.acc <- update(fit.addreg, accelerate = "squarem")
fit.addreg.em.acc <- update(fit.addreg.em, accelerate = "squarem")

# Usual S3 methods work on addreg objects
summary(fit.addreg)
vcov(fit.addreg)
confint(fit.addreg)
summary(predict(fit.addreg), type = "response")

fit.addreg2 <- addreg(update(formula(fit.glm), ~ . - factor(GoodSpine)),
  family = poisson, data = crabs.boot, trace = 1)
anova(fit.addreg2, fit.addreg, test = "LRT")

# Account for overdispersion (use start to speed it up a little)
fit.addreg.od <- addreg(Satellites ~ factor(Dark) + factor(GoodSpine),
  family = negbin1, data = crabs.boot, trace = 1,
  start = c(4.3423675,-2.4059273,-0.4531984,5.969648))
summary(fit.addreg.od)
```

`addreg.control` *Auxiliary for Controlling addreg Fitting*

Description

Auxiliary function for addreg fitting. Typically only used internally by nnpois and nnnegbin, but may be used to construct a control argument to these functions.

Usage

```
addreg.control(bound.tol = 1e-06, epsilon = 1e-10, maxit = 10000, trace = 0)
```

Arguments

bound.tol	positive tolerance specifying the interior of the parameter space. If the fitted model is more than bound.tol away from the boundary of the parameter space then it is assumed to be in the interior. This can allow the computational method to terminate early if an interior maximum is found. No early termination is attempted if bound.tol = Inf.
epsilon	positive convergence tolerance ϵ; the estimates are considered to have converged when $\sqrt{\sum(\theta_{old} - \theta_{new})^2}/\sqrt{\sum \theta_{old}^2} < \epsilon$, where θ is the vector of parameter estimates. See conv.test.
maxit	integer giving the maximum number of EM algorithm iterations for a given parameterisation.
trace	number indicating level of output that should be produced. >= 1 gives output for each parameterisation, >= 2 gives output at each iteration.

Details

This is used similarly to glm.control. The control argument of addreg is by default passed to the control argument of nnpois or nnnegbin.

When trace is greater than zero, calls to cat produce the output. Hence, options(digits = *) can be used to increase the precision.

Value

A list with components named as the arguments.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

glm.control, the equivalent function for glm fitting.

nnpois and nnnegbin, the functions used to fit addreg models.

Examples

```
## Variation on example(glm.control) :

counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
oo <- options(digits = 12)
addreg.D93X <- addreg(counts ~ outcome + treatment, family = poisson,
  trace = 2, epsilon = 1e-2)
options(oo)
coef(addreg.D93X)
```

---

addreg.smooth *Smooth Additive Regression for Discrete Data*

Description

addreg.smooth fits additive (identity-link) Poisson, negative binomial and binomial regression models using a stable EM algorithm. It provides additional flexibility over addreg by allowing for semi-parametric terms.

Usage

```
addreg.smooth(formula, mono = NULL, family, data, standard, subset,
              na.action, offset, control = list(...), model = TRUE,
              model.addreg = FALSE, method = c("cem", "em"),
              accelerate = c("em", "squarem", "pem", "qn"),
              control.method = list(), ...)
```

Arguments

formula	an object of class "formula" (or one that can be coerced into that class): a symbolic description of the model to be fitted. The details of model specification are given under "Details". The model must contain an intercept and at least one semi-parametric term, included by using the B or Iso functions. Note that 2nd-order terms (such as interactions) or above are not currently supported (see addreg).
mono	a vector indicating which terms in formula should be restricted to have a monotonically non-decreasing relationship with the outcome. May be specified as names or indices of the terms. Iso() terms are always monotonic.
family	a description of the error distribution to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function (see family for details of family functions), but here it is restricted to be poisson, negbin1 or binomial family with identity link.

data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which addreg.smooth is called.
standard	a numeric vector of length equal to the number of cases, where each element is a positive constant that (multiplicatively) standardises the fitted value of the corresponding element of the response vector. Ignored for binomial family (the two-column specification of response should be used instead).
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set be the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. This should be NULL or a *non-negative* numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset. Ignored for binomial family; not yet implemented for negative binomial models.
control	list of parameters for controlling the fitting process, passed to addreg.control.
model	a logical value indicating whether the *model frame* (and, for binomial models, the equivalent Poisson model) should be included as a component of the returned value.
model.addreg	a logical value indicating whether the fitted addreg object should be included as a component of the returned value.
method	a character string that determines which EM-type algorithm to use to find the MLE: "cem" for the combinatorial EM algorithm, which cycles through a sequence of constrained parameter spaces, or "em" for a single EM algorithm based on an overparameterised model.
accelerate	a character string that determines the acceleration algorithm to be used, (partially) matching one of "em" (no acceleration — the default), "squarem", "pem" or "qn". See turboem for further details. Note that "decme" is not permitted.
control.method	a list of control parameters for the acceleration algorithm, which are passed to the control.method argument of turboem. If any items are not specified, the defaults are used.
...	arguments to be used to form the default control argument if it is not supplied directly.

Details

addreg.smooth performs the same fitting process as addreg, providing a stable maximum likelihood estimation procedure for identity-link Poisson, negative binomial or binomial models, with the added flexibility of allowing semi-parametric B and Iso terms (note that addreg.smooth will

stop with an error if no semi-parametric terms are specified in the right-hand side of the formula; addreg should be used instead).

The method partitions the parameter space associated with the semi-parametric part of the model into a sequence of constrained parameter spaces, and defines a fully parametric addreg model for each. The model with the highest log-likelihood is the MLE for the semi-parametric model (see Donoghoe and Marschner, 2015).

Acceleration of the EM algorithm can be achieved through the methods of the **turboEM** package, specified through the accelerate argument. However, note that these methods do not have the guaranteed convergence of the standard EM algorithm, particularly when the MLE is on the boundary of its (possibly constrained) parameter space.

Value

An object of class "addreg.smooth", which contains the same objects as class "addreg" (the same as "glm" objects, without contrasts, qr, R or effects components), as well as:

model.addreg	if model.addreg is TRUE; the addreg object for the fully parametric model corresponding to the fitted model.
xminmax.smooth	the minimum and maximum observed values for each of the smooth terms in the model, to help define the covariate space.
full.formula	the component from interpret.addreg.smooth(formula) that contains the formula term with any additional arguments to the B function removed.
knots	a named list containing the knot vectors for each of the smooth terms in the model.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

References

Donoghoe, M. W. and I. C. Marschner (2015). Flexible regression models for rate differences, risk differences and relative risks. *International Journal of Biostatistics* 11(1): 91–108.

Marschner, I. C. (2014). Combinatorial EM algorithms. *Statistics and Computing* 24(6): 921–940.

See Also

addreg

Examples

```
## Simple example
dat <- data.frame(x1 = c(3.2,3.3,3.4,7.9,3.8,0.7,2.0,5.4,8.4,3.0,1.8,5.6,5.5,9.0,8.2),
  x2 = c(1,0,0,1,0,1,0,0,0,0,1,0,1,1,0),
  n = c(6,7,5,9,10,7,9,6,6,7,7,8,6,8,10),
  y = c(2,1,2,6,3,1,2,2,4,4,1,2,5,7,7))
m1 <- addreg.smooth(cbind(y, n-y) ~ B(x1, knot.range = 1:3) + factor(x2), mono = 1,
  data = dat, family = binomial, trace = 1)
```

```
plot(m1, at = data.frame(x2 = 0:1))
points(dat$x1, dat$y / dat$n, col = rainbow(2)[dat$x2 + 1], pch = 20)
```

---

anova.addreg *Analysis of Deviance for addreg Fits*

Description

Compute an analysis of deviance table for more than one GLM fitted using addreg.

Usage

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

Arguments

object, ...	objects of class "addreg", typically the result of a call to addreg, or a list of objects for the "addreglist" method.
test	a character string, (partially) matching one of "Chisq", "LRT", "Rao", "F" or "Cp". See stat.anova.

Details

Unlike anova.glm, specifying a single object is not allowed.

The table has a row for the residual degrees of freedom and deviance for each model. For all but the first model, the change in degrees of freedom and deviance is also given. (This only makes statistical sense if the models are nested.) It is conventional to list the models from smallest to largest, but this is up to the user.

Models where the MLE lies on the boundary of the parameter space will be automatically removed from the list (with a warning), because asymptotic results to not apply to such models.

The table will optionally contain test statistics (and p-values) comparing the reduction in deviance for the row to the residuals. Mallows' *Cp* statistic is the residual deviance plus twice the estimate of σ^2 times the residual degrees of freedom, which is closely related to AIC. You can also choose "LRT" and "Rao" for likelihood ratio tests and Rao's efficient score test. The former is synonymous with "Chisq" (although both have an asymptotic chi-square distribution).

Value

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

Warning

The comparison between two or more models will only be valid if they are fitted to the same dataset. This may be a problem if there are missing values and R's default of na.action = na.omit is used, and anova will detect this with an error.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

addreg, anova.glm, anova

Examples

```
## For an example, see example(addreg)
```

B.Iso *Defining Smooths in addreg.smooth Formulae*

Description

Function used in the definition of smooth terms within addreg.smooth model formulae. The function does not evaluate a smooth — it exists purely to help set up a model using smooths.

Usage

```
B(..., knots = NULL, knot.range = 0:5)

Iso(...)
```

Arguments

...	variable that this smooth is a function of. Note that unlike gam, smooths that are functions of more than one variable are not supported.
knots	*unique* positions of *interior* knots of a B-spline basis. Boundary knots are created automatically.
knot.range	if knots is not specified, a vector containing a series of non-negative integers denoting the number of *interior* knots for which the model will be fit. These are placed at evenly-spaced quantiles of the observed covariate values. At least one of knots or knot.range must be non-missing.

Details

The function does not evaluate the variable arguments; the output from this function is used when producing the model matrix, at which point the actual basis functions are constructed.

B is used to specify an order-3 B-spline basis (which can be restricted to be monotonically non-decreasing via the mono argument in addreg.smooth). If length(knot.range) > 1, models with each of the specified number of interior knots will be fit, and the model with the best (smallest) aic.c will be returned.

Iso is used to specify an isotonic basis, designed such that the resulting function has non-negative increments at each observed covariate value. When Iso is used, the resulting function will always be monotonically non-decreasing, regardless of the value of mono.

Value

An object of class "B.smooth" (for B) or "Iso.smooth" (for Iso), which is a list with the following elements:

term	name of the term provided in the ... argument.
termlabel	label for the term in the model; e.g. for term "x" it will be "B(x)" or "Iso(x)".
knots	vector of interior knots (if specified). NA for Iso.
knot.range	vector of number of interior knots. NA for Iso.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

addreg.smooth

s performs a similar function in the mgcv package.

Examples

```
## See example(addreg.smooth) for an example of specifying smooths in model
##  formulae.
```

---

confint.addreg *Confidence Intervals for addreg Model Parameters*

Description

Computes confidence intervals for one or more parameters in a fitted addreg model.

Usage

```
## S3 method for class 'addreg'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	a fitted model object, resulting from a call to addreg.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.
...	additional argument(s) passed to confint.default.

Details

Calculates confidence intervals for model parameters assuming asymptotic normality, using `vcov.addreg(object)`. As such, if the MLE is on the boundary of the parameter space, (i.e. `object$boundary == TRUE`) the normality assumption is invalid and `NA` is returned.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1-(1-level)/2 in % (by default 2.5% and 97.5%).

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

`confint.default`, `vcov.addreg`

Examples

```
## For an example, see example(addreg)
```

---

contr.isotonic *Contrast Matrix for Isotonic Covariate*

Description

Return something similar to a contrast matrix for a categorical covariate that we wish to be monotonically non-decreasing in a specified order.

Usage

```
contr.isotonic(n, perm, contrasts = TRUE, sparse = FALSE)

contr.opisotonic(n, perm, contrasts = TRUE, sparse = FALSE)
```

Arguments

`n`	a vector of levels for a factor, or the number of levels.
`perm`	a permutation of the levels of `n` (or of the numbers `1:n`), which define the order in which the coefficients must be monotonically non-decreasing.
`contrasts`	a logical indicating whether constrasts should be computed.
`sparse`	included for compatibility reasons. Has no effect.

Details

`contr.isotonic` is used in creating the design matrix for categorical covariates with a specified order under a particular parameterisation. For Poisson and negative binomial models, this occurs if a categorical covariate is defined as monotonic; for binomial models, each parameterisation defines a permutation of the levels that must be monotonically increasing.

For overparameterised binomial models, the design matrix for categorical covariates must include isotonic-style dummy covariates for every possible permutation of the levels. This is the function of `contr.opisotonic`.

In the order specified by `perm`, the coefficient associated with each level is the sum of increments between the preceding levels. That is, the first level is defined as 0, the second as $0 + d_2$, the third as $0 + d_2 + d_3$, and so on. In fitting the model, these increments are constrained to be non-negative.

Note that these are not 'contrasts' as defined in the theory for linear models; rather this is used to define the `contrasts` attribute of each variable so that `model.matrix` produces the desired design matrix.

Value

A matrix with `n` rows and `k` columns, with `k=n-1` if `contrasts` is `TRUE` and `k=n` if `contrasts` is `FALSE`.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

`model.matrix`, which uses `contr.isotonic` to create the design matrix.

`contr.treatment`, `contrasts` for their usual use in regression models.

Examples

```
contr.isotonic(4,1:4)
contr.isotonic(4,c(1,3,2,4))

# Show how contr.isotonic applies within model.matrix
x <- factor(round(runif(20,0,2)))
mf <- model.frame(~x)
contrasts(x) <- contr.isotonic(levels(x), levels(x))
model.matrix(mf)
```

conv.test *Convergence Test Based on L2 Norm*

Description

Performs a test of convergence based on the L2 norm of the change in the parameter estimates.

Usage

```
conv.test(theta1, theta2, epsilon)
```

Arguments

theta1	vector of parameter estimates at previous step.
theta2	vector of parameter estimates at current step.
epsilon	positive convergence tolerance.

Details

This is used as the convergence test in the addreg fitting functions, because the EM algorithm may converge slowly such that the test based on the deviance used in glm.fit (see glm.control) may report convergence at a point away from the actual optimum.

Value

A logical; TRUE if `sqrt(sum((theta1-theta2)**2))/sqrt(sum(theta1**2))` < epsilon, FALSE otherwise.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

Examples

```
theta.old <- c(4,5,6)
theta.new <- c(4.05,5,6)

conv.test(theta.old, theta.new, 0.01)
conv.test(theta.old, theta.new, 0.005)
```

`interpret.addreg.smooth` *Interpret an addreg.smooth Formula*

Description

This is an internal function of package `addreg`. It is a service routine for `addreg.smooth` which interprets the smooth parts of the model formula and returns modified formulas to be used in the fitting functions.

Not normally called directly.

Usage

```
interpret.addreg.smooth(formula)
```

Arguments

`formula`	A formula as supplied to `addreg.smooth`, which includes at least one `B` or `Iso` term.

Value

A list with components:

`full.formula`	a `formula` object which is the same as the `formula` supplied, but with additional arguments removed from the smooth terms. E.g. `B(x, knot.range = 0:2)` would appear as `B(x)` in this formula.
`fake.formula`	a `formula` object which is the same as the `formula` supplied, but with smooth terms replaced by their covariates alone. E.g. `B(x, knot.range = 0:2)` would appear as `x` in this formula. Used to construct the model matrix.
`smooth.spec`	a named list containing the results of `eval`uating the smooth terms. See `B` and `Iso` for details.
`smooth.ind`	a vector containing the indices of the smooth components in the `formula`.
`terms`	the result of running `terms.formula(formula, specials = c("B", "Iso"))`.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

`addreg.smooth`

Examples

```
# Specify a smooth model with knot.range
res <- interpret.addreg.smooth(y ~ B(x, knot.range = 0:2) + x2)
# The knot.range is removed from the full.formula...
print(res$full.formula)
# ...but is stored in the $smooth.spec component of the result:
print(res$smooth.spec$x$knot.range)
```

---

negbin1 *Family Functions for Negative Binomial 1 Models*

Description

Specifies the information required to fit a negative binomial 1 (NB1) model.

Usage

```
negbin1(link, phi = stop("'phi' must be given"))
```

Arguments

link	included for compatibility with family. For addreg models, this will always be "identity".
phi	the value of the scale parameter of the NB1 distribution (see "Details"). This can be set to NA for initialisation, but during estimation the family should be updated with the current estimate, and must be strictly positive.

Details

The NB1 distribution can be parameterised in terms of a mean μ and scale parameter ϕ (the phi argument of this function), such that if $Y \sim NB1(\mu, \phi)$, then $E(Y) = \mu$ and $Var(Y) = (1 + \phi)\mu$.

These can be related to the size and prob arguments of the NegBinomial functions by size = μ/ϕ and prob = $1/(1 + \phi)$.

Value

An object of class "family": see family for full details. Note that when the estimate of phi is updated in a model, this family object must be reloaded using the new estimate.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

nnnegbin *ECME Algorithm for Additive Negative Binomial 1 Model*

Description

Finds the maximum likelihood estimate of an additive negative binomial (NB1) model using an ECME algorithm, where each of the mean coefficients is restricted to be non-negative.

Usage

```
nnnegbin(y, x, standard, offset, start, control = addreg.control(),
         accelerate = c("em", "squarem", "pem", "qn"),
         control.method = list())
```

Arguments

y	non-negative integer response vector.
x	non-negative covariate matrix.
standard	standardising vector, where each element is a positive constant that (multiplicatively) standardises the fitted value of the corresponding element of the response vector. The default is a vector of ones.
offset	non-negative additive offset vector. The default is a vector of zeros.
start	vector of starting values for the parameter estimates. The last element is the starting value of the scale, and must be > 1. The remaining elements are for the additive mean parameters, and must be greater than control$bound.tol.
control	an addreg.control object, which controls the fitting process.
accelerate	a character string that determines the acceleration algorithm to be used, (partially) matching one of "em" (no acceleration – the default), "squarem", "pem" or "qn". See turboem for further details. Note that "decme" is not permitted.
control.method	a list of control parameters for the acceleration algorithm. See turboem for details of the parameters that apply to each algorithm. If not specified, the defaults are used.

Details

This is a workhorse function for addreg, and runs the ECME algorithm to find the constrained non-negative MLE associated with an additive NB1 model.

Value

A list containing the following components

coefficients	the constrained non-negative maximum likelihood estimate of the mean parameters.
scale	the maximum likelihood estimate of the scale parameter.

residuals	the residuals at the MLE, that is y - fitted.values
fitted.values	the fitted mean values.
rank	the number of parameters in the model (named "rank" for compatibility — we assume that models have full rank)
family	included for compatibility — will always be negbin1(identity).
linear.predictors	included for compatibility — same as fitted.values (as this is an identity-link model).
deviance	up to a constant, minus twice the maximised log-likelihood (with respect to a saturated NB1 model with the same scale).
aic	a version of Akaike's *An Information Criterion*, minus twice the maximised log-likelihood plus twice the number of parameters.
aic.c	a small-sample corrected version of Akaike's *An Information Criterion* (Hurvich, Simonoff and Tsai, 1998).
null.deviance	the deviance for the null model, comparable with deviance. The null model will include the offset and an intercept.
iter	the number of iterations of the EM algorithm used.
weights	included for compatibility — a vector of ones.
prior.weights	included for compatibility — a vector of ones.
standard	the standard vector passed to this function.
df.residual	the residual degrees of freedom.
df.null	the residual degrees of freedom for the null model.
y	the y vector used.
converged	logical. Did the ECME algorithm converge (according to conv.test)?
boundary	logical. Is the MLE on the boundary of the parameter space — i.e. are any of the coefficients < control$bound.tol?
loglik	the maximised log-likelihood.
nn.design	the non-negative x matrix used.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>.

References

Donoghoe, M. W. and I. C. Marschner (2016). Estimation of adjusted rate differences using additive negative binomial regression. *Statistics in Medicine* 35(18): 3166–3178.

Hurvich, C. M., J. S. Simonoff and C.-L. Tsai (1998). Smoothing parameter selection in non-parametric regression using an improved Akaike information criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60(2): 271–293.

nnpois *EM Algorithm for Identity-link Poisson GLM*

Description

Finds the maximum likelihood estimate of an identity-link Poisson GLM using an EM algorithm, where each of the coefficients is restricted to be non-negative.

Usage

```
nnpois(y, x, standard, offset, start, control = addreg.control(),
       accelerate = c("em", "squarem", "pem", "qn"),
       control.method = list())
```

Arguments

y	non-negative integer response vector.
x	non-negative covariate matrix.
standard	standardising vector, where each element is a positive constant that (multiplicatively) standardises the fitted value of the corresponding element of the response vector. The default is a vector of ones.
offset	non-negative additive offset vector. The default is a vector of zeros.
start	starting values for the parameter estimates. Each element must be greater than control$bound.tol.
control	an addreg.control object, which controls the fitting process.
accelerate	a character string that determines the acceleration algorithm to be used, (partially) matching one of "em" (no acceleration – the default), "squarem", "pem" or "qn". See turboem for further details. Note that "decme" is not permitted.
control.method	a list of control parameters for the acceleration algorithm. See turboem for details of the parameters that apply to each algorithm. If not specified, the defaults are used.

Details

This is a workhorse function for addreg, and runs the EM algorithm to find the constrained non-negative MLE associated with an identity-link Poisson GLM. See Marschner (2010) for full details.

Value

A list containing the following components

coefficients	the constrained non-negative maximum likelihood estimate of the parameters.
residuals	the residuals at the MLE, that is y - fitted.values
fitted.values	the fitted mean values.

`rank`	the number of parameters in the model (named "`rank`" for compatibility — we assume that models have full rank)
`family`	included for compatibility — will always be `poisson(identity)`.
`linear.predictors`	included for compatibility — same as `fitted.values` (as this is an identity-link model).
`deviance`	up to a constant, minus twice the maximised log-likelihood.
`aic`	a version of Akaike's *An Information Criterion*, minus twice the maximised log-likelihood plus twice the number of parameters.
`aic.c`	a small-sample corrected version of Akaike's *An Information Criterion* (Hurvich, Simonoff and Tsai, 1998).
`null.deviance`	the deviance for the null model, comparable with `deviance`. The null model will include the offset and an intercept.
`iter`	the number of iterations of the EM algorithm used.
`weights`	included for compatibility — a vector of ones.
`prior.weights`	included for compatibility — a vector of ones.
`standard`	the `standard` vector passed to this function.
`df.residual`	the residual degrees of freedom.
`df.null`	the residual degrees of freedom for the null model.
`y`	the `y` vector used.
`converged`	logical. Did the EM algorithm converge (according to `conv.test`)?
`boundary`	logical. Is the MLE on the boundary of the parameter space — i.e. are any of the `coefficients` < `control$bound.tol`?
`loglik`	the maximised log-likelihood.
`nn.design`	the non-negative `x` matrix used.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>.

This function is based on code from Marschner, Gillett and O'Connell (2012) written by Alexandra Gillett.

References

Hurvich, C. M., J. S. Simonoff and C.-L. Tsai (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60(2): 271–293.

Marschner, I. C. (2010). Stable computation of maximum likelihood estimates in identity link Poisson regression. *Journal of Computational and Graphical Statistics* 19(3): 666–683.

Marschner, I. C., A. C. Gillett and R. L. O'Connell (2012). Stratified additive Poisson models: Computational methods and applications in clinical epidemiology. *Computational Statistics and Data Analysis* 56(5): 1115–1130.

`plot.addreg.smooth` *Default addreg.smooth Plotting*

Description

Takes a fitted `addreg.smooth` object produced by `addreg.smooth` and plots the component smooth functions that make it up, on the scale of the linear predictor, for specified values of the other covariates.

Usage

```
## S3 method for class 'addreg.smooth'
plot(x, type = c("response", "link"), at = data.frame(),
     knotlines = TRUE, nobs = 1000, ...)
```

Arguments

`x`	a fitted `addreg.smooth` object as produced by `addreg.smooth`.
`type`	the type of prediction required. Note that, unlike `predict.addreg.smooth`, `"terms"` is not a valid option. Also, because `addreg.smooth` only applies identity-link models, `"response"` and `"link"` will have the same results — they are included for consistency.
`at`	a data frame containing the values at which the prediction should be evaluated. The columns must contain the covariates in the model, and several rows may be provided (in which case, multiple lines are drawn on the same plot). Cannot be missing or `NULL`.
`knotlines`	logical; if vertical lines should be drawn on the plot to indicate the locations of the knots for B-spline terms.
`nobs`	the number of points which should be used to create the curve. These are placed evenly along the range of the observed covariate values from the original model.
`...`	other graphics parameters to pass on to plotting commands (note: some will not work).

Details

For each smooth covariate in the model of `x`, `predict.addreg.smooth` is used to obtain predicted values for the range of that covariate, with the other covariates remaining fixed at their values given in `at`. Several rows may be provided in `at`, in which case, one curve is drawn for each, and they are coloured using `rainbow(nrow(at))`. If the model contains a single smooth covariate and no other covariates, `at` may be provided as an empty data frame, `data.frame()`.

Value

The function simply generates plots.

Note

If this function is too restrictive, it may be easier to use `predict.addreg.smooth` to get predictions for the dataset of your choice, and do the plotting manually.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

`addreg.smooth`, `predict.addreg.smooth`

Examples

```
## For an example, see example(addreg.smooth)
```

---

predict.addreg *Predict Method for addreg Fits*

Description

Obtains predictions from a fitted `addreg` object.

Usage

```
## S3 method for class 'addreg'
predict(object, newdata = NULL, type = c("link", "response", "terms"), terms = NULL,
        na.action = na.pass, checkminmax = TRUE, ...)
```

Arguments

`object`	a fitted object of class inheriting from `"addreg"`.
`newdata`	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
`type`	the type of prediction required. The default is on the scale of the linear predictors; the alternative `"response"` is on the scale of the response variable. The `"terms"` option returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale. The value of this argument can be abbreviated.
`terms`	with `type = "terms"` by default all terms are returned. A character vector specifies which terms are to be returned.
`na.action`	function determining what should be done with missing values in `newdata`. The default is to predict `NA`.
`checkminmax`	logical indicating whether or not values of continuous covariates in `newdata` should be checked to ensure they lie within the covariate space associated with the fitted model. Otherwise predicted values could lie outside the parameter space.
`...`	further arguments passed to or from other methods.

Details

If newdata is omitted the predictions are based on the data used for the fit. In that case how cases with missing values in the original fit are treated is determined by the na.action argument of that fit. If na.action = na.omit, omitted cases will not appear in the residuals; if na.action = na.exclude they will appear, with residual value NA. See also napredict.

Value

A vector or matrix of predictions. For type = "terms", this is a matrix with a column per term, and may have an attribute "constant".

Note

Variables are first looked for in newdata and then searched for in the usual way (which will include the environment of the formula used in the fit). A warning will be given if the variables found are not of the same length as those in newdata if it was supplied.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

addreg

predict.glm for the equivalent method for models fit using glm.

Examples

```
## For an example, see example(addreg)
```

---

predict.addreg.smooth *Predict Method for addreg.smooth Fits*

Description

Obtains predictions from a fitted addreg.smooth object.

Usage

```
## S3 method for class 'addreg.smooth'
predict(object, newdata = NULL, type = c("link", "response", "terms"),
        terms = NULL, na.action = na.pass, ...)
```

Arguments

object	a fitted object of class inheriting from "addreg.smooth".
newdata	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
type	the type of prediction required. The default is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable. The "terms" option returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale. The value of this argument can be abbreviated.
terms	with type = "terms" by default all terms are returned. A character vector specifies which terms are to be returned.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	further arguments passed to or from other methods.

Details

predict.addreg.smooth constructs the underlying basis functions for smooth variables in newdata and runs predict.addreg to obtain predictions. Note that if values of smooth covariates in newdata are outside the covariate space of object, an error will be returned.

If newdata is omitted, the predictions are based on the data used for the fit. In that case how cases with missing values in the original fit are treated is determined by the na.action argument of that fit. If na.action = na.omit, omitted cases will not appear in the residuals; if na.action = na.exclude they will appear, with residual value NA. See also napredict.

Value

A vector or matrix of predictions. For type = "terms", this is a matrix with a column per term, and may have an attribute "constant".

Note

Variables are first looked for in newdata and then searched for in the usual way (which will include the environment of the formula used in the fit). A warning will be given if the variables found are not of the same length as those in newdata if it was supplied.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

addreg.smooth, predict.addreg

predict.glm for the equivalent method for models fit using glm.

Examples

```
## For an example, see example(addreg.smooth)
```

summary.addreg *Summarizing addreg Model Fits*

Description

These functions are all methods for class addreg or summary.addreg objects.

Usage

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

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

Arguments

object	an object of class "addreg", usually from a call to addreg or addreg.smooth.
x	an object of class "summary.addreg", usually from a call to summary.addreg.
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
digits	the number of significant digits to use when printing.
signif.stars	logical; if TRUE, 'significance stars' are printed for each coefficient.
...	further arguments passed to or from other methods.

Details

These perform the same function as summary.glm and print.summary.glm, producing similar results for addreg models. print.summary.addreg additionally prints the small-sample corrected AIC (aic.c), the number of EM iterations for the parameterisation corresponding to the MLE, and for negative binomial models, the estimate of ϕ (scale-1) and its standard error.

The dispersion used in calculating standard errors is fixed as 1 for binomial and Poisson models, and is estimated via maximum likelihood for negative binomial models.

Value

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

call	the component from object.
family	the component from object.
deviance	the component from object.
aic	the component from object.
aic.c	the component from object.

df.residual	the component from object.
null.deviance	the component from object.
df.null	the component from object.
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	included for compatibility — always FALSE.
dispersion	the inferred/estimated dispersion.
df	included for compatibility — a 3-vector of the number of coefficients, the number of residual degrees of freedom, and the number of coefficients (again).
cov.unscaled	the unscaled (dispersion = 1) estimated covariance matrix of the estimated coefficients. NaN if object$boundary == TRUE.
cov.scaled	ditto, scaled by dispersion.
correlation	if correlation is TRUE, the estimated correlations of the estimated coefficients. NaN if object$boundary == TRUE.

For negative binomial models, the object also contains

phi	the estimate of ϕ (scale-1).
var.phi	the estimated variance of phi.

Note

If object$boundary == TRUE, the standard errors of the coefficients are not valid, and a matrix of NaNs is returned by vcov.addreg.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

addreg, summary.glm

Examples

```
## For an example, see example(addreg)
```

vcov.addreg *Calculate Variance-Covariance Matrix for a Fitted addreg Model Object*

Description

Returns the variance-covariance matrix of the main parameters of a fitted addreg model object.

Usage

```
## S3 method for class 'addreg'
vcov(object, ...)
```

Arguments

object	an object of class "addreg", usually from a call to addreg or addreg.smooth.
...	additional arguments for method functions.

Details

An equivalent method to vcov, to use with addreg models.

Value

A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model. This should have row and column names corresponding to the parameter names given by the coef method.

Note

If object$boundary == TRUE, the standard errors of the coefficients are not valid, and a matrix of NaNs is returned.

Author(s)

Mark W. Donoghoe <markdonoghoe@gmail.com>

See Also

summary.addreg, vcov.glm

Examples

```
## For an example, see example(addreg)
```

Index