Package ‘mpcmp’

March 4, 2019

Type Package

Title Mean-Parametrized Conway-Maxwell Poisson (COM-Poisson) Regression

Version 0.1.3

Date 2019-02-14

Description A collection of functions for estimation, testing and diagnostic checking for the mean-parametrized Conway-Maxwell Poisson (COM-Poisson) regression model of Huang (2017) <doi:10.1177/1471082X17697749>.

Depends R (>= 2.10)

License GPL (>= 2)

Imports graphics, grDevices, stats

URL `https://github.com/thomas-fung/mpcmp`

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Suggests knitr, rmarkdown

NeedsCompilation no

Author Thomas Fung [aut, cre],
Aya Alwan [aut],
Justin Wishart [aut],
Alan Huang [aut]

Maintainer Thomas Fung <`thomas.fung@mq.edu.au`>

Repository CRAN

Date/Publication 2019-03-04 17:20:03 UTC

R topics documented:

`mpcmp-package` *Mean-parametrized Conway-Maxwell-Poisson Regression*

Description

Mean-parametrized Conway-Maxwell-Poisson Regression

References

#' Fung, T., Alwan, A., Wishart, J. and Huang, A. (2019). The `mpcmp` package for Mean-parametrized Conway-Maxwell-Poisson Regression.

Huang, A. (2017). Mean-parametrized Conway–Maxwell–Poisson regression models for dispersed counts. *Statistical Modelling* **17**, 359–380.

`AIC.cmp` *Akaike's Information Criterion*

Description

A function calculating Akaike's Information Criterion (AIC) based on the log-likelihood value extracted from `logLik.cmp`, according to the formula *-2*log-likelihood + k*npar*, where *npar* represents the number of parameters in the fitted model, and *k=2* for the usual AIC or *k=log(n)* (*n* being the number of observations) for the so-called BIC (Bayesian Information Criterion).

Usage

```
## S3 method for class 'cmp'
AIC(object, ..., k = 2)
```

Arguments

`object`	an object class 'cmp' object, obtained from a call to `glm.cmp`
`...`	other arguments passed to or from other methods (currently unused).
`k`	numeric: the *penalty* per parameter to be used; the default k = 2 is the classical AIC.

Details

When comparing models fitted by maximum likelihood to the same data, the smaller the AIC or BIC, the better the fit.

Value

A numeric value with the corresponding AIC (or BIC, or ..., depends on k).

See Also

`logLik.cmp`, `nobs.cmp`, `glm.cmp`

`attendance` *Attendance data set*

Description

This data set gives the number of days absent from high school and the gender, maths score (standardized score out of 100) and academic programme ('General', 'Academic' and 'Vocational') of 314 students sampled from two urban high schools. The attendance data frame has 314 observations on 5 variables.

Usage

```
data(attendance)
```

Format

A data frame with 314 observations on 5 variables.

id	Identifier
gender	gender
math	standardized math score out of 100
daysabs	number of days absent from high school
prog	academic programme (‘General’, ‘Academic’ and ‘Vocational’)

Source

`http://www.ats.ucla.edu/stat/stata/dae/nb_data.dta`

Examples

```
## For examples see example(glm.cmp)
```

`CBIND` *Combine R Objects by Columns*

Description

Take a sequence of vector, matrix or data-frame arguments and combine them by columns. `CBIND` is used within the package over `cbind` to recycle the shorter arguments so that their number of rows would match.

Usage

```
CBIND(..., deparse.level = 1)
```

Arguments

`...`	(generalized) vectors or matrices. These can be given as named arugments.
`deparse.level`	integer; deparse.level = 0 constructs no labels, deparse.level = 1 (the default) or > 1 constructs labels from the arguments names.

`cmplrtest` *Likelihood Ratio Test for nested COM-Poisson models*

Description

Perform a likelihood ratio chi-sqaured test between nested COM-Poisson models. The test statistics is calculated as *2*(llik- llik_0)*. The test statistics has degrees of freedom *r* where *r* is the difference in the number of parameters between the full and null models.

Usage

```
cmplrtest(object1, object2, digits = 3)
```

Arguments

`object1`	an object class 'cmp', obtained from a call to `glm.cmp`
`object2`	an object class 'cmp', obtained from a call to `glm.cmp`
`digits`	numeric; minimum number of significant digits to be used for most numbers.

Details

Obviously the comparison between two models will only be valid if they are fitted to the same data set.

References

Huang, A. (2017). Mean-parametrized Conway–Maxwell–Poisson regression models for dispersed counts. *Statistical Modelling* **17**, 359–380.

See Also

`glm.cmp`, `update.cmp`

Examples

```
data(takeoverbids)

## Fit full model
M.bids.full <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)

## Fit null model; without whtknght
M.bids.null <- update(M.bids.full, .~.-whtknght)

## Likelihood ratio test for the nested models
cmplrtest(M.bids.full, M.bids.null) # order of objects is not important
```

`coef.cmp` *Extract Model Coefficients from a COM-Poisson Model Fit*

Description

An accessor function used to extract model coefficients from a 'cmp' object. `coefficients` is an alias for `coef`.

Usage

```
## S3 method for class 'cmp'
coef(object, ...)
```

Arguments

`object`	an object class 'cmp' object, obtained from a call to `glm.cmp`
`...`	other arguments passed to or from other methods (currently unused).

Value

Coefficients extracted from the object `object`.

See Also

`fitted.cmp`, `residuals.cmp`, `glm.cmp`.

`COMP_Expected_Values` *Functions to Compute Various Expected Values for the COM-Poisson Distribution*

Description

Functions to approximate the various expected values for the COM-Poisson distribution via truncation. The standard COM-Poisson parametrization is being used here. The lambda and nu values are recycled to match the length of the longer one and that would determine the length of the results. Notice that the sum is hard coded to tuncate at 100 so the approximation will be quite bad if the COM-Poisson has a large rate or mean.

Usage

```
comp_mean_logfactorialy(lambda, nu)

comp_mean_ylogfactorialy(lambda, nu)

comp_means(lambda, nu)

comp_variances(lambda, nu)

comp_variances_logfactorialy(lambda, nu)
```

Arguments

`lambda, nu`	rate and dispersion parameters. Must be positives.

Value

`comp_mean_logfactorialy` gives the mean of *log(Y!)*.

`comp_mean_ylogfactorialy` gives the mean of *ylog(Y!)*.

`comp_means` gives the mean of *Y*.

`comp_variances` gives the variance of *Y*.

`comp_variances_logfactorialy` gives the variance of *log(Y!)*.

`comp_lambdas` *Solve for Lambda for a Particular Mean Parametrized COM-Poisson Distribution*

Description

Given a particular mean parametrized COM-Poisson distribution i.e. mu and nu, this function is used to find a lambda that can satisfy the mean constraint with a combination of bisection and Newton-Raphson updates. The function is also vectorized but will only update those that have not converged.

Usage

```
comp_lambdas(mu, nu, lambdalb = 1e-10, lambdaub = 1900,
  maxlambdaiter = 1000, tol = 1e-06, lambdaint = 1)
```

Arguments

`mu, nu`	mean and dispersion parameters. Must be straightly positive.
`lambdalb, lambdaub`	numeric; the lower and upper end points for the interval to be searched for lambda(s).

`maxlambdaiter`	numeric; the maximum number of iterations allowed to solve for lambda(s).
`tol`	numeric; the convergence threshold. A lambda is said to satisfy the mean constraint if the absolute difference between the calculated mean and the corresponding mu values is less than tol.
`lambdaint`	numeric vector; initial gauss for lambda(s).

Value

The function returns the lambda value(s) that satisfies the mean constraint(s).

`comp_mu_loglik` *Calculate the Log-Likelihood of the COM-Poisson model*

Description

A function to compute the log-likelihood of the COM-Poisson model.

Usage

```
comp_mu_loglik(param, y, xx, offset)
```

Arguments

`param`	numeric vector: the model coefficients & the current value of `nu`. It is assumed that `nu` is in the last position of `param`.
`y`	numeric vector: response variable
`xx`	numeric matrix: the explanatory variables
`offset`	numeric vector: a vector of lenght equal to the number of cases

Value

The log-likelihood value of the COM-Poisson model.

`COM_Poisson_Distribution`

The Conway-Maxwell-Poisson (COM-Poisson) Distribution.

Description

Density, distribution function, quantile function and random generation for the Conway-Maxwell-Poisson distribution with parameter `mu` and `nu`

Usage

```
dcomp(x, mu, nu = 1, lambda, log.p = FALSE, lambdalb = 1e-10,
  lambdaub = 1900, maxlambdaiter = 1000, tol = 1e-06)

pcomp(q, mu, nu = 1, lambda, lower.tail = TRUE, log.p = FALSE,
  lambdalb = 1e-10, lambdaub = 1900, maxlambdaiter = 1000,
  tol = 1e-06)

qcomp(p, mu, nu = 1, lambda, lower.tail = TRUE, log.p = FALSE,
  lambdalb = 1e-10, lambdaub = 1900, maxlambdaiter = 1000,
  tol = 1e-06)

rcomp(n, mu, nu = 1, lambda, lambdalb = 1e-10, lambdaub = 1900,
  maxlambdaiter = 1000, tol = 1e-06)
```

Arguments

`x, q`	vector of quantiles
`mu, nu`	mean and dispersion parameters. Must be strictly positive.
`lambda`	an alternative way than mu to parametrized the distribution. Must be strictly positive
`log.p`	logical; if `TRUE`, probabilities/densities p are returned as $log(p)$.
`lambdalb, lambdaub`	numeric: the lower and upper end points for the interval to be searched for lambda(s).
`maxlambdaiter`	numeric: the maximum number of iterations allowed to solve for lambda(s).
`tol`	numeric: the convergence threshold. A lambda is said to satisfy the mean constraint if the absolute difference between the calculated mean and mu is less than tol.
`lower.tail`	logical; if `TRUE` (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
`p`	vector of probabilites
`n`	number of observations. If `length(n)` > 1, the length is taken to be the number required.

Value

`dcomp` gives the density, `pcomp` gives the distribution function, `qcomp` gives the qunatile function, and `rcomp` generates random deviates.

Invalid arguemnts will result in return value `NaN`, with a warning.

The length of the results is determined by `n` for `rcomp`, and is the maximum of the lengths of the numerical arguments for the other functions.

The numerical arguments other than `n` are recycled to the length of the results. Only the first arugment of the logical arguments are used.

Examples

```
dcomp(0:5, mu = 2, nu = 1.2)
pcomp(5, mu=2, nu =1.2)
p <- (1:9)/10
qcomp(p, mu = 2, nu = 0.8)
rcomp(10, mu = 2, nu = 0.7)
```

`cottonbolls` *Cotton Bolls data set*

Description

This data set gives the observed number of bolls produced by the cotton plants at five growth stages: vegetative, flower-bud, blossom, fig and cotton boll; to examine the effect of five defoliation levels (0

Usage

```
data(cottonbolls)
```

Format

A data frame with 125 observations on 4 variables.

nc number of bolls produced by two cotton plants at harvest

stages growth stage

def artificial defoliation level

def2 square of def

Source

Supplementary Content of Zevini et al. (2014): http://www.leg.ufpr.br/doku.php/publications:papercompanions:zeviani-jas2014

References

Zeviani, W.M., Riberio P.J. Jr., Bonat, W.H., Shimakura S.E. and Muniz J.A. (2014). The Gamma-count distribution in the analysis of experimental underdispersed data. *Journal of Applied Statistics* **41**, 2616–26.

Examples

```
## For examples see example(glm.cmp)
```

`fitted.cmp` *Extract Fitted Values from a COM-Poisson Model Fit*

Description

An accessor function used to extract the fitted values from a 'cmp' object. `fitted.values` is an alias for `fitted`.

Usage

```
## S3 method for class 'cmp'
fitted(object, ...)
```

Arguments

`object`	an object class 'cmp' object, obtained from a call to `glm.cmp`
`...`	other arguments passed to or from other methods (currently unused).

Value

Fitted values `mu` extracted from the object `object`.

See Also

`coef.cmp`, `residuals.cmp`, `glm.cmp`.

`getnu` *Parameter Generator for nu*

Description

A function that use the arguments of a `glm.cmp` call to generate a better initial `nu` estimate.

Usage

```
getnu(param, y, xx, offset, llstart, fsscale = 1, lambdalb = 1e-10,
  lambdaub = 1900, maxlambdaiter = 1000, tol = 1e-06)
```

Arguments

`param`	numeric vector: the model coefficients & the current value of `nu`. It is assumed that `nu` is in the last position of `param`.
`y`	numeric vector: response variable
`xx`	numeric matrix: the explanatory variables
`offset`	numeric vector: a vector of lenght equal to the number of cases
`llstart`	numeric: current log-likelihood value
`fsscale`	numeric: a scaling factor (generally >1) for the relaxed fisher scoring algorithm
`lambdalb, lambdaub`	numeric: the lower and upper end points for the interval to be searched for lambda(s).
`maxlambdaiter`	numeric: the maximum number of iterations allowed to solve for lambda(s).
`tol`	numeric: the convergence threshold. A lambda is said to satisfy the mean constraint if the absolute difference between the calculated mean and a fitted values is less than tol.

Value

List containing the following:

`param`	the model coefficients & the updated `nu`
`maxl`	the updated log-likelihood
`fsscale`	the final scaling factor used

`glm.cmp` *Fit a Mean Parametrized Conway-Maxwell-Poisson Generalized Linear Model*

Description

The function `glm.cmp` is used to fit a mean parametrized Conway-Maxwell-Poisson generalized linear model with a log-link by using Fisher Scoring iteration.

Usage

```
glm.cmp(formula, data, offset = NULL,
   lambdalb = 1e-10, lambdaub = 1900, maxlambdaiter = 1e3, tol = 1e-6)
```

Arguments

`formula`	an object of class 'formula': a symblic desciption of the model to be fitted.
`data`	an optional data frame containing the variables in the model
`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 numeric vector of length equal to the number of cases.
`lambdalb, lambdaub`	numeric: the lower and upper end points for the interval to be searched for lambda(s). The default value for lambdaub should be sufficient for small to moderate size nu. If nu is large and required a larger `lambdaub`, the algorithm will scale up `lambdaub` accordingly.
`maxlambdaiter`	numeric: the maximum number of iterations allowed to solve for lambda(s).
`tol`	numeric: the convergence threshold. A lambda is said to satisfy the mean constraint if the absolute difference between the calculated mean and a fitted values is less than tol.

Details

Fit a mean parametrizied COM-Poisson regression using maximum likelihood estimation via an iterative Fisher Scoring algorithm.

The COM-Poisson regression model is

Y_i ~ CMP(mu_i, nu),

where

E(Y_i) = mu_i = exp(x_i^T beta),

and *nu > 0* is the dispersion parameter.

The fitted COM-Poisson distribution is over- or under-dispersed if *nu < 1* and *nu > 1* respectively.

Value

A fitted model object of class `cmp` similar to one obtained from `glm` or `glm.nb`.

The function `summary` (i.e., `summary.cmp`) can be used to obtain and print a summary of the results.

The function `plot` (i.e., `plot.cmp`) can be used to produce a range of diagnostic plots.

The generic assessor functions `coef` (i.e., `coef.cmp`), `logLik` (i.e., `logLik.cmp`) `fitted` (i.e., `fitted.cmp`), `nobs` (i.e., `nobs.cmp`), `AIC` (i.e., `AIC.cmp`) and `residuals` (i.e., `residuals.cmp`) can be used to extract various useful features of the value returned by `glm.cmp`.

An object class 'glm.cmp' is a list containing at least the following components:

`coefficients`	a named vector of coefficients
`se_beta`	approximate standard errors (using observed rather than expected information) for coefficients
`residuals`	the *response* residuals (i.e., observed-fitted)
`fitted.values`	the fitted mean values
`rank`	the numeric rank of the fitted linear model
`linear.predictors`	the linear fit on log scale
`df.residuals`	the residuals degrees of freedom
`df.null`	the residual degrees of freedom for the null model
`null.deviance`	The deviance for the null model. The null model will include only the intercept.
`y`	the `y` vector used.
`x`	the model matrix
`model`	the model frame
`call`	the matched call
`formula`	the formula supplied
`terms`	the `terms` object used
`data`	the `data` argument
`offset`	the `offset` vector used
`lambdaub`	the final `lambdaub` used

References

Fung, T., Alwan, A., Wishart, J. and Huang, A. (2018). The `mpcmp` package for Mean-parametrized Conway-Maxwell-Poisson Regression.

Huang, A. (2017). Mean-parametrized Conway–Maxwell–Poisson regression models for dispersed counts. *Statistical Modelling* **17**, 359–380.

See Also

`summary.cmp`, `plot.cmp`, `fitted.cmp` and `residuals.cmp`.

Examples

```
### Huang (2017) Page 368--370: Overdispersed Attendance data

data(attendance)
M.attendance <- glm.cmp(daysabs~ gender+math+prog, data=attendance)
M.attendance
summary(M.attendance)


### Huang (2017) Page 371--372: Underdispersed Takeover Bids data
data(takeoverbids)
M.bids <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)
    M.bids
summary(M.bids)

### Huang (2017) Page 373--375: Underdispersed Cotton bolls data
### Model fitting for predictor V

data(cottonbolls)
M.bolls <- glm.cmp(nc~ 1+stages:def+stages:def2, data= cottonbolls)
M.bolls
summary(M.bolls)
```

`is.wholenumber` *Test for a whole number*

Description

Test for integer/whole number vector

Usage

```
is.wholenumber(x, tol = .Machine$double.eps^0.5)
```

Arguments

`x`	numeric vector to be tested
`tol`	numeric; precision level

`logLik.cmp` *Extract the (Maximized) Log-Likelihood from a COM-Poisson Model Fit*

Description

An accessor function used to extract the (maximized) log-likelihood from a 'cmp' object.

Usage

```
## S3 method for class 'cmp'
logLik(object, ...)

## S3 method for class 'logLik.cmp'
print(x, ...)
```

Arguments

`object`	an object of class 'cmp' object, obtained from a call to `glm.cmp`
`...`	other arguments passed to or from other methods (currently unused).
`x`	an object of class 'logLik.cmp', obtained from a call to `logLik.cmp`.

See Also

`coef.cmp`, `fitted.cmp`, `glm.cmp`

`LRTnu` *Likelihood Ratio Test for nu = 1 of a COM-Poisson model*

Description

Perform a likelihood ratio chi-sqaured test for nu = 1 of a COM-Poisson model. The test statistics is calculated as *2*(llik- llik_0)* where *llik* and *llik_0* are the log-likelihood of a COM-Poisson and Poisson model respectively. The test statistic has 1 degrees of freedom.

Usage

```
LRTnu(object, digits = 3)
```

Arguments

`object`	an object class 'cmp', obtained from a call to `glm.cmp`
`digits`	numeric; minimum number of significant digits to be used for most numbers.

References

Huang, A. (2017). Mean-parametrized Conway–Maxwell–Poisson regression models for dispersed counts. *Statistical Modelling* **17**, 359–380.

Examples

```
data(takeoverbids)
M.bids <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)
LRTnu(M.bids)
```

`model.frame.cmp` *Extract the Model Frame from a COM-Poisson Model Fit*

Description

An accessor function used to extract the model frame from a ’cmp’ object.

Usage

```
## S3 method for class 'cmp'
model.frame(formula, ...)
```

Arguments

`formula`	an object class ’cmp’ object, obtained from a call to `glm.cmp`
`...`	other arguments passed to or from other methods (currently unused).

Value

The method will return the saved `data.frame` used when fitting the cmp model.

See Also

`coef.cmp`, `residuals.cmp`, `glm.cmp`.

`nobs.cmp` *Extract the Number of Observation from a COM-Poisson Model Fit*

Description

An accessor function used to extract the number of observation from a 'cmp' object.

Usage

```
## S3 method for class 'cmp'
nobs(object, ...)
```

Arguments

`object`	an object class 'cmp' object, obtained from a call to `glm.cmp`
`...`	other arguments passed to or from other methods (currently unused).

Value

The number of observations extracted from the object `object`.

See Also

`coef.cmp`, `fitted.cmp`, `glm.cmp`

`nrPIT` *Non-randomized Probability Integral Transform*

Description

Functions to produce the non-randomzied probability integral transform (PIT) to check the adequacy of the distributional assumption of the COM-Poisson model. The majority of the code and descriptions are taken from Dunsmuir and Scott (2015).

Usage

```
compPredProb(object)

compPIT(object, bins = 10)
```

Arguments

`object`	an object class "cmp", obtained from a call to `glm.cmp`.
`bins`	numeric; the number of bins shown in the PIT histogram or the PIT Q-Q plot.

Details

These functions are used to obtain the predictive probabilities and the probability integral transform for a fitted COM-Poisson model. The majority of the code and descriptions are taken from Dunsmuir and Scott (2015).

Value

`compPredprob` returns a list with values:

`upper`	the predictive cumulative probabilities used as the upper bound for computing the non-randomzied PIT.
`lower`	the predictive cumulative probabilities used as the upper bound for computing the non-randomzied PIT.

`compPIT` returns a list with values:

`conditionalPIT`	the conditional probability integral transoformation given the observed counts.
`PIT`	the probability integral transformation.

References

Czado, C., Gneiting, T. and Held, L. (2009). Predictive model assessment for count data. *Biometrics*, **65**, 1254–1261.

Dunsmuir, W.T.M. and Scott, D.J. (2015). The `glarma` Package for Observation-Driven Time Seires Regression of Counts. *Journal of Statistical Software*, **67**, 1–36.

Examples

```
data(takeoverbids)
M.bids <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)
compPredProb(M.bids)
compPIT(M.bids)
```

`PIT_Plot` *PIT Plots for a CMP Object*

Description

Two plots for the non-randomized PIT are currently available for checking the distributional assumption of the fitted CMP model: the PIT histogram, and the uniform Q-Q plot for PIT.

Usage

```
histcompPIT(object, bins = 10, line = TRUE, colLine = "red",
  colHist = "royal blue", lwdLine = 2, main = NULL, ...)

qqcompPIT(object, bins = 10, col1 = "red", col2 = "black",
  lty1 = 1, lty2 = 2, type = "l", main = NULL, ...)
```

Arguments

`object`	an object class "cmp", obtained from a call to `glm.cmp`.
`bins`	numeric; the number of bins shown in the PIT histogram or the PIT Q-Q plot.
`line`	logical; if `TRUE` (default), the line for displaying the standard uniform distribution will be shown for the purpose of comparison.
`colLine`	numeric or charater: the colour of the line for comparison in PIT histogram.
`colHist`	numeric or character; the colour of the histogram for PIT.
`lwdLine`	numeric; the line widths for the comparison line in PIT histogram.
`main`	character string; a main title for the plot.
`...`	other arguments passed to plot.default and plot.ts.
`col1`	numeric or character; the colour of the sample uniform Q-Q plot in PIT.
`col2`	numeric or character; the colour of the theoretical uniform Q-Q plot in PIT.
`lty1`	integer or character string: the line types for the sample uniform Q-Q plot in PIT, see par(lty = .).
`lty2`	an integer or character string: the line types for the theoretical uniform Q-Q plot in PIT, see par(lty = .).
`type`	1-character string; the type of plot for the sample uniform Q-Q plot in PIT.

Details

The histogram and the Q-Q plot are used to compare the fitted profile with a standard uniform distribution. If they match relatively well, it means the CMP distribution is appropriate for the data.

References

Czado, C., Gneiting, T. and Held, L. (2009). Predictive model assessment for count data. *Biometrics*, **65**, 1254–1261.

Dunsmuir, W.T.M. and Scott, D.J. (2015). The `glarma` Package for Observation-Driven Time Seires Regression of Counts. *Journal of Statistical Software*, **67**, 1–36.

Examples

```
## For examples see example(plot.cmp)
```

`plot.cmp` *Plot Diagnostic for a* `glm.cmp` *Object*

Description

Eight plots (selectable by `which`) are currently available: a plot of deviance residuals against fitted values, a non-randomized PIT histogram, a uniform Q-Q plot for non-randomzied PIT, a histogram of the normal randomized residuals, a Q-Q plot of the normal randomized residuals, a Scale-Location plot of sqrt(| residuals |) against fitted values a plot of Cook's distances versus row lobels a plot of pearson residauls against leverage. By default, four plots (number 1, 2, 6, and 8 from this list of plots) are provided.

Usage

```
## S3 method for class 'cmp'
plot(x, which = c(1L, 2L, 6L, 8L),
  ask = prod(par("mfcol")) < length(which) && dev.interactive(),
  bins = 10, ...)
```

Arguments

`x`	an object class 'cmp' object, obtained from a call to `glm.cmp`
`which`	if a subset of plots is required, specify a subset of the numbers 1:8. See 'Details' below.
`ask`	logical; if `TRUE`, the user is asked before each plot.
`bins`	numeric; the number of bins shown in the PIT histogram or the PIT Q-Q plot.
`...`	other arguments passed to or from other methods (currently unused).

Details

The 'Scale-Location' plot, also called 'Spread-Loation' plot, takes the square root of the absolute standardized deviance residuals ($sqrt|E|$) in order to diminish skewness is much less skewed than than $|E|$ for Gaussian zero-mean E.

The 'Scale-Location' plot uses the standardized deviance residuals while the Residual-Leverage plot uses the standardized pearson residuals. They are given as *R[i]/sqrt(1-h.ii)* where h_ii are the diagonal entries of the hat matrix.

The Residuals-Leverage plot shows contours of equal Cook's distance for values of 0.5 and 1.

There are two plots based on the non-randomized probability integral transformation (PIT) using `compPIT`. These are a histogram and a uniform Q-Q plot. If the model assumption is appropriate, these plots should reflect a sample obtained from a uniform distribution.

There are also two plots based on the normal randomzied residuals calculated using `compnormRandPIT`. These are a histogram and a normal Q-Q plot. If the model assumption is appropriate, these plots should reflect a sample obtained from a normal distribuiton.

See Also

`compPIT`, `compnormRandPIT`, `glm.cmp`

Examples

```
data(takeoverbids)
M.bids <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)

## The default plots are shown
plot(M.bids)

## The plots for the non-randomzied PIT
# plot(M.bids, which = c(2,3))
```

predict.cmp *Model Predicitons for a* `glm.cmp` *Object*

Description

This is a function for obtaining predictions and optionally estimates standard errors of those prediction from a fitted COM-Poisson regression object.

Usage

```
## S3 method for class 'cmp'
predict(object, newdata = NULL, se.fit = FALSE,
  type = c("link", "response"), ...)
```

Arguments

`object`	an object class 'cmp', obtained from a call to `glm.cmp`.
`newdata`	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
`se.fit`	logical; indicating if standard errors are required.
`type`	the type of prediction required. The default is 'link' which is the scale of the linear predictor i.e., a log scale; the alternative 'response' is on the scale of the response variable. The value of this argument can be abbreviated.
`...`	other arguments passed to or from other methods (currently unused).

Details

If newdata is omitted the predictions are based on the data used for the fit.

Value

If `se.fit = FALSE`, a vector of predictions.

If `se.fit = TRUE`, a list with components

`fit`	Predictions, as for se.fit = FALSE.
`se.fit`	Estimated standard errors.

Examples

```
data(takeoverbids)
M.bids <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)

predict(M.bids)
predict(M.bids, type= "response")
predict(M.bids, se.fit=TRUE, type="response")
```

```
newdataframe <- data.frame(bidprem = 1, finrest = 0, insthold = 0.05,
    leglrest = 0, rearest = 1, regulatn = 0, size = 0.1, whtknght = 1,
    sizesq = .1^2)
predict(M.bids, se.fit=TRUE, newdata = newdataframe, type="response")
```

`print.cmp` *Print Values of COM-Poisson Model*

Description

`print` method for class `cmp`.

Usage

```
## S3 method for class 'cmp'
print(x, ...)
```

Arguments

`x`	an object class 'cmp', obtained from a call to `glm.cmp`.
`...`	other arguments passed to or from other methods (currently unused).

Details

`print.cmp` can be used to print a short summary of object class 'cmp'.

See Also

`summary.cmp`, `coef.cmp`, `fitted.cmp`, `glm.cmp`.

Examples

```
## For examples see example(glm.cmp)
```

`residuals.cmp` *Extract COM-Poisson Model Residuals*

Description

`residuals` is a generic function which extracts model residuals from objects returned by the modelling function `glm.comp`. `resid` is an alias for `residuals` .

Usage

```
## S3 method for class 'cmp'
residuals(object, type = c("deviance", "pearson",
  "response"), ...)
```

Arguments

`object`	an object class 'cmp', obtained from a call to `glm.cmp`.
`type`	the `type` of residuals which should be returned. The alternatives are: 'deviance' (default), 'pearson' and 'response'. Can be abbreviated.
`...`	other arguments passed to or from other methods (currently unused).

Value

Residuals extracted from the object `object`.

See Also

`coef.cmp`, `fitted.cmp`, `glm.cmp`

`rPIT` *Random Normal Probability Integral Transform*

Description

A function to create the normal conditional (randomized) quantile residuals. The majority of the code and descriptions are taken from Dunsmuir and Scott (2015).

Usage

```
compnormRandPIT(object)
```

Arguments

`object`	an object class "cmp", obtained from a call to `glm.cmp`.

Details

The function `compPredProb` produces the non-randomized probability integral transform(PIT). It returns estimates of the cumulative predictive probabilities as upper and lower bounds of a collection of intervals. If the model is correct, a histogram drawn using these estimated probabilities should resemble a histogram obtained from a sample from the uniform distribution.

This function aims to produce observations which instead resemble a sample from a normal distribution. Such a sample can then be examined by the usual tools for checking normality, such as histograms and normal Q-Q plots.

For each of the intervals produced by `compPredProb`, a random uniform observation is generated, which is then converted to a normal observation by applying the inverse standard normal distribution function (using `qnorm`). The vector of these values is returned by the function in the list element `rt`. In addition non-random observations which should appear similar to a sample from a normal distribution are obtained by applying `qnorm` to the mid-points of the predictive distribution intervals. The vector of these values is returned by the function in the list element `rtMid`.

Value

A list consisting of two elements:

`rt`	the normal conditioanl randomized quantile residuals
`rdMid`	the midpoints of the predictive probability intervals

References

Berkowitz, J. (2001). Testing density forecasts, with applications to risk management. *Journal of Business \& Economic Statistics*, **19**, 465–474.

Dunn, P. K. and Smyth, G. K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics*, **5**, 236–244.

Dunsmuir, W.T.M. and Scott, D.J. (2015). The `glarma` Package for Observation-Driven Time Seires Regression of Counts. *Journal of Statistical Software*, **67**, 1–36.

Examples

```
data(takeoverbids)
M.bids <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)
compnormRandPIT(M.bids)
```

`summary.cmp` *Summary of COM-Poisson Model Fit*

Description

`summary` method for class `cmp`.

Usage

```
## S3 method for class 'cmp'
summary(object, digits = max(3L, getOption("digits") - 3L),
  ...)
```

Arguments

`object`	an object class 'cmp', obtained from a call to `glm.cmp`.
`digits`	numeric; minimum number of significant digits to be used for most numbers.
`...`	other arguments passed to or from other methods (currently unused).

Details

`summary.glm` tries to be smart about formatting the coefficients, standard errors and gives 'significance starts'. The `coefficients` component of the result gives the estimated coefficients and their estimated standard errors, together with their ratio. This third column is labelled as `Z value` as the dispersion is fixed for this family. A forth column gives the two-tailed p-value corresponding to `Z value` based on the asymptotic Normal reference distribuiton.

See Also

`coef.cmp`, `fitted.cmp`, `glm.cmp`.

Examples

```
## For examples see example(glm.cmp)
```

`takeoverbids` *Takeover Bids data set*

Description

This data set gives the number of bids received by 126 US firms that were successful targets of tender offers during the period 1978–1985, along with some explanatory variables on the defensive actions taken by management of target firm, firm-specific characteristics and intervention taken by federal regulators. The `takeoverbids` data frame has 126 observations on 14 variables. The descriptions below are taken from Sáez-Castillo and Conde-Sánchez (2013).

Usage

```
data(takeoverbids)
```

Format

A data frame with 126 observations on 14 variables.

bidprem bid price divided by price 14 working days before bid

docno doc no

finrest indicator variable for proposed change in ownership structure

insthold percentage of stock held by institutions

leglrest indicator variable for legal defence by lawsuit

numbids number of bids recevied after the initial bid

obs Identifier

rearest indicator variable for proposed changes in asset structure

regulatn indicator variable for Department of Justice intervention

size total book value of assets in billions of dollars

takeover Indicator. 1 if the company was being taken over

weeks time in weeks between the initial and final offers

whtknght indicator varible for management invitation for friendly third-party bid

sizesq book value squared

Source

Journal of Applied Econometrics data archive: `http://qed.econ.queensu.ca/jae/`.

References

Cameron, A.C. and Johansson, P. (1997). Count Data Regression Models using Series Expansions: with Applications. *Journal of Applied Econometrics* **12** 203–223.

Cameron, A.C. and Trivedi P.K. (1998). Regression analysis of count data, Cambridge University Press, `http://cameron.econ.ucdavis.edu/racd/racddata.html` chapter 5.

Croissant Y (2011) Ecdat: Datasets for econometrics, R Package, version 0.1–6.1.

Jaggia, S. and Thosar, S. (1993). Multiple Bids as a Consequence of Target Management Resistance *Review of Quantitative Finance and Accounting* **3**, 447–457.

Examples

```
## For examples see example(glm.cmp)
```

`update.cmp` *Update and Re-fit a COM-Poisson Model*

Description

`update` (i.e., `update.cmp`) will upate and (by-default) re-fit a model. It is identical to `update` in the `stats` package.

Usage

```
## S3 method for class 'cmp'
update(object, formula., ..., evaluate = TRUE)
```

Arguments

`object`	an object class ’cmp’, obtained from a call to `glm.cmp`.
`formula.`	changes to the existing formula in `object` – see `update.formula` for details
`...`	other arguments passed to or from other methods (currently unused).
`evaluate`	logical; if `TRUE` evaluate the new call otherwise simply ruturn the call

See Also

`glm.cmp`, `update.formula`, `cmplrtest`.

Examples

```
data(takeoverbids)

## Fit full model
M.bids.full <- glm.cmp(numbids ~ leglrest + rearest + finrest + whtknght
    + bidprem + insthold + size + sizesq + regulatn, data=takeoverbids)
M.bids.full
```

```
## Dropping whtknght
M.bids.null <- update(M.bids.full, .~.-whtknght)
M.bids.null
```

`Z` *Calculate the Normalizing Constant for COM-Poisson distribution*

Description

A function to approximate the normalizing constant for COM-Poisson distributions via truncation. The standard COM-Poisson parametrization is being used here. Notice that the sum is hard coded to tuncate at 100 so the approximation will be quite bad if the COM-Poisson has a large rate or mean.

Usage

```
Z(lambda, nu)
```

Arguments

`lambda`	rate parameter, straightly positive
`nu`	diepsersoin parameter, straightly positive

Index