Package 'VGAMdata'

February 14, 2022

Version 1.1-6

Date 2022-02-02

Title Data Supporting the 'VGAM' Package

Author Thomas Yee [aut, cre, cph],
James Gray [dtc]

Maintainer Thomas Yee <t.yee@auckland.ac.nz>

Depends R (>= 3.5.0), methods, stats, VGAM

Description Mainly data sets to accompany the VGAM package and the book ``Vector Generalized Linear and Additive Models: With an Implementation in R'' (Yee, 2015) <DOI:10.1007/978-1-4939-2818-7>. These are used to illustrate vector generalized linear and additive models (VGLMs/VGAMs), and associated models (Reduced-Rank VGLMs, Quadratic RR-VGLMs, Row-Column Interaction Models, and constrained and unconstrained ordination models in ecology). This package now contains some old VGAM family functions which have been replaced by newer ones (often because they are now special cases).

License GPL-2

URL https://www.stat.auckland.ac.nz/~yee/VGAMdata/

Repository CRAN

LazyLoad yes

LazyData yes

NeedsCompilation no

Date/Publication 2022-02-14 13:50:09 UTC

R topics documented:

`VGAMdata-package` *Mainly Data for the VGAM package*

Description

VGAMdata is mainly an assortment of larger data sets which are a useful resource for the **VGAM** package.

Details

This package mainly contains some larger data sets originally distributed with the **VGAM** package. Ideally both packages can be installed and loaded to be fully functional. The main intent was to limit the size of **VGAM** to a bare essential. Many data sets in my monograph will refer to data sets in either package. Recently, some older or less-used **VGAM** family functions have been shifted into **VGAMdata**, and this is likely to continue in the future.

Author(s)

Thomas W. Yee, <t.yee@auckland.ac.nz>.

Maintainer: Thomas Yee <t.yee@auckland.ac.nz>.

References

Yee, T. W. (2015). Vector Generalized Linear and Additive Models: With an Implementation in R. New York, USA: *Springer*.

See Also

`VGAM-package`.

Examples

```
# Example 1; xs.nz
head(xs.nz)
summary(xs.nz)

# Example 2; ugss
head(ugss)
summary(ugss)
```

`airbnb.ac` *Airbnb Accommodation in Two Sardinian Cities*

Description

The `airbnb.ac` data frame has 18159 rows and 9 columns.

Usage

```
data(airbnb.ac)
```

Format

This data frame contains the following columns:

LOS length of stay, in days.

discountWeek Multiplicative factor for the discount for booking one week: $1-$ (published weekly rate) / ($7 \times$ published nightly rate), e.g., 0.2 means a 20 percent savings off the regular price.

NumberofReviews Number of reviews on the website.

PriceAvg Average price per night per person, in Euros.

Bedrooms Number of bedrooms.

Superhost Logical. Superhost?

MinimumStay Minimum stay period, in days.

MaxGuests Maximum number of guests.

City Character, values are `"Alghero"` and `"Cagliari"`.

Details

The data frame comprises Airbnb bookings in two cities located in Sardinia, Italy. The stays were during the whole of 2016. Stays of 30 days or longer and any rows with missing variables were deleted from the original source. Variable `LOS` exhibits heaping at the values 7 and 14 days.

Source

The data was obtained with permission from Luca Frigau, University of Cagliari, from https://www.airbnb.com.

See Also

gaitdzeta, flamingo.

Examples

```
## Not run:
mytab <- with(subset(airbnb.ac, City == "Alghero"), table(LOS))
plot(prop.table(mytab), col = "blue", ylab = "Proportion")

## End(Not run)
```

---

bb.de *Battle of Britain Data (a Luftwaffe subset)*

Description

Luftwaffe losses during a subset of the Battle of Britain.

Usage

```
data(bb.de)
```

Format

The format is a 3-dimensional array. The first dimension is the event (in order: shot down or failed to return, written off, seriously damaged), the second dimension is the day, the third is the aircraft type.

Details

This is a Battle of Britain data set of Luftwaffe losses during operations 26 August–31 August 1940 continued on to 1–7 September 1940. The aircraft types are prefixed `Bf` for Messerschmitt (Bayerische Flugzeugwerke), `Do` for Dornier, `He` for Heinkel, `Ju` for Junkers.

Note that p.151 and p.165 of Bowyer (1990) contain tables (during the first week of September) and almost the same data; however, the former is labelled "shot down" whereas the latter is "shot down or failed to return". The latter is used here. Also, there are some other small discrepancies.

Yet to do: add the data available at other dates, and include the RAF data.

Source

Bowyer, M. J. F. (1990) *The Battle of Britain: 50 years On*. Patrick Stephens Limited, Northamptonshire, U.K.

Examples

```
data(bb.de)
bb.de[,, "Bf109"]

## Not run:
plot(bb.de["sdown",, "Bf109"] ~ as.Date(dimnames(bb.de)[[2]]),
     type = "h", col = "blue", las = 1, lwd = 3,
     ylab = "Frequency", xlab = "1940",
     main = "Numbers shot down (Bf 109)")
abline(h = c(5, 10, 15, 20), lty = "dashed", col = "grey")
points(bb.de["sdown",,"Bf109"] ~ as.Date(dimnames(bb.de)[[2]]), col = "blue")

## End(Not run)
```

---

bd.us *Births and Deaths of 348 Notable Americans*

Description

A 12 x 12 table of the Births and Deaths of 348 Notable Americans. The rows and columns are for each month.

Usage

```
data(bd.us)
```

Format

The format is: chr "bd.us"

Details

Rows denote the month of birth; columns for the month of death. These data appear as Table 1 of Phillips and Feldman (1973), who collected the data from Morris (1965). Not all of the 400 people were used because some had not died by that time and other individuals lacked the information.

Source

Phillips, D. P. and Feldman, K. A. (1973) A dip in deaths before ceremonial occasions: Some new relationships between social integration and mortality, American Sociological Review, **38**, 678–696.

Morris, R. B. (Ed.) (1965) *Four Hundred Notable Americans*. Harper and Row: New York, USA.

See Also

rcim.

Examples

```
print(bd.us)
sum(bd.us)
rowSums(bd.us)
colSums(bd.us)
```

---

belcap *BELCAP Dental Data*

Description

A prospective data set containing the DMFT index of children in Belo Horizonte at the beginning and end of the BELCAP study.

Usage

```
data(belcap)
```

Format

A data frame with 797 observations on the following 5 variables.

`dmftb` a numeric vector, DMFT-Index at the beginning of the study.

`dmfte` a numeric vector, DMFT-Index at the end of the study.

`gender` a factor with levels `0` = female, `1` = male.

`ethnic` a factor with levels `1` = dark, `2` = white, `3` = black.

`school` the kind of prevention measure. A factor with levels `1` = oral health education, `2` = all four methods, `3` = control group, `4` = enrichment of the school diet with ricebran, `5` = mouthrinse with 0.2% NaF-solution, `6` = oral hygiene.

Details

This data set is from the Belo Horizonte Caries Prevention (BELCAP) study. The data is collected from children in Belo Horizonte (Brazil) aged seven years at the start of the study. In in order to determine which method(s) were best for preventing tooth decay, six treatments were randomized to six separate schools. The measure used is the decayed, missing and filled teeth (DMFT) index - a well known and important measure of a persons dental health. Only the eight deciduous molars are considered, so the lowest value is 0, and the highest is 8.

Source

`https://onlinelibrary.wiley.com/` contains the data file (a supplement of the JRSSA article). Downloaded in January 2014 and formatted into R by J. T. Gray, `jamsgry@gmail.com`.

References

Bohning, D., D. Ekkehart, P. Schlattmann, L. Mendonca, and U. Kircher (1999). The Zero-Inflated Poisson Model and the Decayed, Missing and Filled Teeth Index in Dental Epidemiology, *Journal of the Royal Statistical Society, A* **162**(2), 195–209.

Examples

```
data(belcap)
## maybe str(belcap) ; plot(belcap) ...
```

Bell *The Bell Distribution*

Description

Density, and random generation for the Topp-Leone distribution.

Usage

```
dbell(x, shape, log = FALSE)
rbell(n, shape)
```

Arguments

x, n	Same as Uniform.
shape	the (shape) parameter, which is positive.
log	Logical. If log = TRUE then the logarithm of the density is returned.

Details

See bellff, the **VGAM** family function for estimating the parameter s by maximum likelihood estimation.

Value

dbell gives the density, rbell generates random deviates. If shape is large then rbell will become computationally expensive.

See Also

bell.

Examples

```
## Not run: plot(0:15, dbell(0:15, shape = 1.5), type = "h", col = "blue")
```

bellff *Bell Distribution Family Function*

Description

Estimating the shape parameter of the Bell distribution by maximum likelihood estimation.

Usage

```
bellff(lshape = "loglink", zero = NULL, gshape = expm1(1.6 * ppoints(7)))
```

Arguments

`lshape, zero, gshape`
More information is at `CommonVGAMffArguments`.

Details

The Bell distribution has a probability density function that can be written

$$f(y; s) = \frac{s^y \exp(1 - e^s) B_y}{y!}$$

for $y = 0(1)\infty$ and shape parameter $0 < s$. The mean of Y is $\exp(s)s$ (returned as the fitted values). Fisher-scoring is used. This **VGAM** family function handles multiple responses.

The function `bell` returns the first 218 Bell numbers as finite numbers, and returns `Inf` when its argument has a higher value. Hence this **VGAM** family function can only handle low-value counts of less than 219.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, and `vgam`.

Author(s)

T. W. Yee

References

Castellares, F. and Ferrari, S. L. P. and Lemonte, A. J. (2018). On the Bell distribution and its associated regression model for count data. *Applied Mathematical Modelling*, **56**, 172–185.

See Also

`bell`, `dbell`, `poissonff`.

Examples

```
bdata <- data.frame(y = rbell(1000, shape = loglink(0.5, inverse = TRUE)))
bfit <- vglm(y ~ 1, bellff, data = bdata, trace = TRUE, crit = "coef")
coef(bfit, matrix = TRUE)
Coef(bfit)
```

`covid19.nz` *COVID-19 in New Zealand: The First Month or So*

Description

The `covid19.nz` data frame has 69 rows and 3 columns. The number of new cases is tracked daily.

Usage

```
data(covid19.nz)
```

Format

This data frame contains the following columns:

doy output from `as.Date`, is the day of year.

newcases a numeric vector, counts, the number of new cases.

Day a numeric vector, 0 for when the first case occurred; incrementally daily after that.

Details

These were collected from https://www.nzherald.co.nz/ during the first month or so after the first case.

Source

The NZ Herald states their source was Johns Hopkins University.

Examples

```
## Not run: plot(newcases ~ doy, covid19.nz, col = "blue", type = "h",
     las = 1, xlab = "Date")

## End(Not run)
```

crashf.au *Fatal Crashes on Australian Roads 2010–2012*

Description

The number of fatal road crashes on Australian roads during 2010–2012. They are cross-classified by time of day (in 6 hour blocks) and day of the week.

Usage

```
data(crashf.au)
```

Format

A data frame with 4 observations on the following 7 variables.

Mon, Tue, Wed, Thu, Fri, Sat, Sun Day of the week.

Details

Each cell is the aggregate number of crashes reported in Australia during each six hour time block throughout the years 2010–2012. The `rownames` are the time period the crashes took place in. Morning is from 3:00am to 8:59am, midday is from 9:00am to 2:59pm, evening is from 3:00pm to 8:59pm and night is from 9:00pm to 2:59am.

Source

`http://www.bitre.gov.au/publications/ongoing/files/RDA_Summary_2012_June.pdf`

References

Road Deaths Australia; 2012 Statistical Summary. Department of Infrastructure and Transport, Australian Government; ISSN: 1323–3688

Downloaded by J. T. Gray, April 2014.

Examples

```
crashf.au
```

crime.us *Estimated Crime in 2009 in USA*

Description

Crime totals and rates, cross-classified by US state, during 2009.

Usage

```
data(crime.us)
```

Format

A data frame with 50 observations on the following 22 variables.

`State` a character vector. White spaces have been replaced by underscores.

`Population` a numeric vector

`ViolentCrimeTotal` a numeric vector

`Murder` a numeric vector

`Rape` a numeric vector

`Robbery` a numeric vector

`Assault` a numeric vector

`PropertyCrimeTotal` a numeric vector

`Burglary` a numeric vector

`LarcenyTheft` a numeric vector

`MotorVehicleTheft` a numeric vector

`ViolentCrimeRate` a numeric vector

`MurderRate` a numeric vector

`RapeRate` a numeric vector

`RobberyRate` a numeric vector

`AssaultRate` a numeric vector

`PropertyCrimeRate` a numeric vector

`BurglaryRate` a numeric vector

`LarcenyTheftRate` a numeric vector

`MotorVehicleTheftRate` a numeric vector

`stateNumber` a numeric vector, running from 1 to 50.

`abbrev` State name as a character vector

Details

Each row is a state of the United States of America. The first half of the columns tend to be totals, and the second half are crime rates per 100,000 population.

The data frame was downloaded as a `.csv` file and edited. The full column names are: State, Population, Violent crime total, Murder and nonnegligent Manslaughter, Forcible rape, Robbery, Aggravated assault, Property crime total, Burglary, Larceny-theft, Motor vehicle theft, Violent Crime rate, Murder and nonnegligent manslaughter rate, Forcible rape rate, Robbery rate, Aggravated assault rate, Property crime rate, Burglary rate, Larceny-theft rate, Motor vehicle theft rate, state Number, abbreviation. Technical details governing the data set are given in the URL.

Source

`http://www.ucrdatatool.gov`, `http://www.ucrdatatool.gov/Search/Crime/State/RunCrimeOneYearofData.cfm`

Examples

```
## Not run:   # Louisiana is the one outlier
plot(MurderRate ~ stateNumber, crime.us,
     axes = FALSE, type = "h", col = 1:6,
     main = "USA murder rates in 2009 (per 100,000 population)")
axis(1, with(crime.us, abbrev), at = with(crime.us, stateNumber),
     col = 1:6, col.tick = 1:6, cex.lab = 0.5)
axis(2)
## End(Not run)
tail(crime.us[ sort.list(with(crime.us, MurderRate)), ])
```

`DeLury` *DeLury's Method for Population Size Estimation*

Description

Computes DeLury's method or Leslie's method for estimating a biological population size.

Usage

```
DeLury(catch, effort, type = c("DeLury","Leslie"), ricker = FALSE)
```

Arguments

`catch`, `effort`	Catch and effort. These should be numeric vectors of equal length.
`type`	Character specifying which of the DeLury or Leslie models is to be fitted. The default is the first value.
`ricker`	Logical. If `TRUE` then the Ricker (1975) modification is computed.

Details

This simple function implements the methods of DeLury (1947). These are called the DeLury and Leslie models. Note that there are many assumptions. These include: (i) Catch and effort records are available for a series of consecutive time intervals. The catch for a given time interval, specified by t, is $c(t)$, and the corresponding effort by $e(t)$. The *catch per unit effort* (CPUE) for the time interval t is $C(t) = c(t)/e(t)$. Let $d(t)$ represent the proportion of the population captured during the time interval t. Then $d(t) = k(t)e(t)$ so that $k(t)$ is the proportion of the population captured during interval t by one unit of effort. Then $k(t)$ is called the *catchability*, and the *intensity* of effort is $e(t)$. Let $E(t)$ and $K(t)$ be the total effort and total catch up to interval t, and $N(t)$ be the number of individuals in the population at time t. It is good idea to plot $\log(C(t))$ against $E(t)$ for `type = "DeLury"` and $C(t)$ versus $K(t)$ for `type = "Leslie"`.

The other assumptions are as follows.

(ii) The population is closed—the population must be closed to sources of animals such as recruitment and immigration and losses of animals due to natural mortality and emigration.

(iii) Catchability is constant over the period of removals.

(iv) The units of effort are independent, i.e., the individual units of the method of capture (i.e., nets, traps, etc) do not compete with each other.

(v) All fish are equally vulnerable to the method of capture—source of error may include gear saturation and trap-happy or trap-shy individuals.

(vi) Enough fish must be removed to substantially reduce the CPUE.

(vii) The catches may remove less than 2% of the population.

Also, the usual assumptions of simple regression such as

(viii) random sampling,

(ix) the independent variable(s) are measured without error—both catches and effort should be known, not estimated,

(x) a line describes the data,

(xi) the errors are independent and normally distributed.

Value

A list with the following components.

`catch, effort`	Catch and effort. Same as the original vectors. These correspond to $c(t)$ and $e(t)$ respectively.
`type, ricker`	Same as input.
`N0`	an estimate of the population size at time 0. Only valid if the assumptions are satisfied.
`CPUE`	Catch Per Unit Effort $= C(t)$.
`K, E`	$K(t)$ and $E(t)$. Only one is computed depending on `type`.
`lmfit`	the `lm` object from the fit of `log(CPUE)` on `K` (when `type = "Leslie"`). Note that the `x` component of the object is the model matrix.

Note

The data in the example below comes from DeLury (1947), and some plots of his are reproduced. Note that he used log to base 10 whereas natural logs are used here. His plots had some observations obscured by the y-axis!

The DeLury method is not applicable to the data frame wffc.nc since the 2008 World Fly Fishing Competition was strictly catch-and-release.

Author(s)

T. W. Yee.

References

DeLury, D. B. (1947). On the estimation of biological populations. *Biometrics*, **3**, 145–167.

Ricker, W. E. (1975). Computation and interpretation of biological statistics of fish populations. *Bull. Fish. Res. Bd. Can.*, **191**, 382–

Yee, T. W. (2010) VGLMs and VGAMs: an overview for applications in fisheries research. *Fisheries Research*, **101**, 116–126.

See Also

wffc.nc.

Examples

```
pounds <- c(  147, 2796, 6888, 7723, 5330, 8839, 6324, 3569, 8120, 8084,
            8252, 8411, 6757, 1152, 1500, 11945, 6995, 5851, 3221, 6345,
            3035, 6271, 5567, 3017, 4559, 4721, 3613,  473,  928, 2784,
            2375, 2640, 3569)
traps  <- c(  200, 3780, 7174, 8850, 5793, 9504, 6655, 3685, 8202, 8585,
            9105, 9069, 7920, 1215, 1471, 11597, 8470, 7770, 3430, 7970,
            4740, 8144, 7965, 5198, 7115, 8585, 6935, 1060, 2070, 5725,
            5235, 5480, 8300)
table1 <- DeLury(pounds/1000, traps/1000)

## Not run:
with(table1, plot(1+log(CPUE) ~ E, las = 1, pch = 19, main = "DeLury method",
     xlab = "E(t)", ylab = "1 + log(C(t))", col = "blue"))

## End(Not run)
omitIndices <- -(1:16)
table1b <- DeLury(pounds[omitIndices]/1000, traps[omitIndices]/1000)
## Not run:
with(table1b, plot(1+log(CPUE) ~ E, las = 1, pch = 19, main = "DeLury method",
     xlab = "E(t)", ylab = "1 + log(C(t))", col = "blue"))
mylmfit <- with(table1b, lmfit)
lines(mylmfit$x[, 2], 1 + predict.lm(mylmfit), col = "red", lty = "dashed")

## End(Not run)
```

```
omitIndices <- -(1:16)
table2 <- DeLury(pounds[omitIndices]/1000, traps[omitIndices]/1000, type = "L")
## Not run:
with(table2, plot(CPUE ~ K, las = 1, pch = 19,
     main = "Leslie method; Fig. III",
     xlab = "K(t)", ylab = "C(t)", col = "blue"))
mylmfit <- with(table2, lmfit)
abline(a = coef(mylmfit)[1], b = coef(mylmfit)[2],
       col = "orange", lty = "dashed")

## End(Not run)
```

`ecb06.it` *Italian Household Data for 2006 and 2014*

Description

Part of the data collected at two time points (2006 and 2014) by the Bank of Italy, as part of the European Central Banks Eurosystem collection of statistics, within the periodic sample surveys on households, businesses and selected intermediaries.

Format

Data frame with the following 20 variables:

`ID` a numeric vector, a unique identification number of the household.

`area` a factor with 5 levels, the Italian geographic area or region in which the household lives: `NW` = North-West, `NE` = North-East, `C` = Center, `S` = South, `I` = Islands. For users wanting a North–South contrast, this variable might be coded as `NC` = North and Center (NW, NE and C), `SI` = South and Islands (S and I).

`sex` a factor with 2 levels, the gender of the head householder: `M` = Male, `F` = Female.

`age` a numeric vector, age in years of the head householder.

`marital` a factor with 4 levels, marital status of the head householder: `married` = Married, `single` = Single, `separated` = Separated or divorced, `widowed` = Widowed.

`education` an ordered factor with 8 levels, the education level of the head householder: `none` = No education, `primaryschool` = Primary school, `middleschool` = Middle school, `profschool` = Professional school, `highschool` = High school, `bachelors` = Bachelors degree, `masters` = Masters degree, `higherdegree` = Higher degree.

`job` a factor with 7 levels, the type of job done by the head householder: `worker` = Worker, `employee` = Employee, `manager` = Manager, `business` = Business person, `other` = Other kind of independent job, `retired` = Retired, `unemployed` = Unemployed.

`occupants` a numeric vector, the number of people living in the same house.

`children` a numeric vector, the number of children of the head householder living with him/her.

`other.children` a numeric vector, the number of children of the head householder not living with the household.

`house.owned` a numeric vector, the ownership of the house in which the householder lives; `0` = The house is not owned, `1` = The house is owned.

`houses` a numeric vector, the number of houses owned by the family, including the one in which the family lives.

`earners` a numeric vector, the number of people in the house who have salary or some kind of earnings.

`accounts` a numeric vector, the number of bank accounts collectively owned by the household.

`ccards` a numeric vector, the number of credit cards collectively owned by the household.

`tot.income`, `dep.income`, `pens.income`, `self.income`, `cap.income` numeric vectors, the amount of income (in Euros) collectively earned by the household through different activities. The variables can be negative if the household has installments. In order, they are the total amount of income, the amount of income earned through dependent working, the amount of income earned by the household through pensions, the amount of income earned by the household through self-dependent working, the amount of income earned by the household through capital investments.

Details

The European Central Banks (ECB) Eurosystem requests and helps organize each country within the European Union to routinely collect statistical information via their central banks. These data frames are a subset from data collected by the Bank of Italy. Each year can be considered a cross-sectional study, although there are some people common to each year. Hence the data collectively can be considered partly a longitudinal study too.

Source

Data was downloaded at `https://www.bancaditalia.it` in May 2016 by Lucia Pilleri.

References

Supplements to the Statistical Bulletin, Sample Surveys, Household Income and Wealth in 2006, New series, Volume XVIII, Number 7–28, January 2008. Banca D'Italia, Centro Stampa, Roma, Pubbl. Mensile, `https://www.bancaditalia.it`.

Examples

```
data(ecb06.it); data(ecb14.it)
summary(ecb06.it)
summary(ecb14.it)
## Not run:
with(ecb14.it, table(house.owned))
with(ecb14.it, barplot(table(education), col = "lightblue"))

## End(Not run)
```

exam1 *Examination data*

Description

Exam results of 35 students on 18 questions.

Usage

```
data(exam1)
```

Format

A data frame with 35 observations on the following 18 variables.

q01, q02, q03, q04, q05, q06 binary response

q07, q08, q09, q10, q11, q12 binary response

q13, q14, q15, q16, q17, q18 binary response

Details

For each question, a 1 means correct, a 0 means incorrect. A simple Rasch model may be fitted to this dataframe using `rcim` and `binomialff`.

Source

Taken from William Revelle's *Short Guide to R*, http://www.unt.edu/rss/rasch_models.htm, http://www.personality-project.org/r/. Downloaded in October 2013.

Examples

```
summary(exam1)  # The names of the students are the row names

# Fit a simple Rasch model.
# First, remove all questions and people who were totally correct or wrong
exam1.1 <- exam1  [, colMeans(exam1  ) > 0]
exam1.1 <- exam1.1[, colMeans(exam1.1) < 1]
exam1.1 <- exam1.1[rowMeans(exam1.1) > 0, ]
exam1.1 <- exam1.1[rowMeans(exam1.1) < 1, ]
Y.matrix <- rdata <- exam1.1

## Not run:  # The following needs: library(VGAM)
rfit <- rcim(Y.matrix, family = binomialff(multiple.responses = TRUE),
             trace = TRUE)

coef(rfit)  # Row and column effects
constraints(rfit, matrix = TRUE)  # Constraint matrices side-by-side
dim(model.matrix(rfit, type = "vlm"))  # 'Big' VLM matrix
```

```
## End(Not run)

## Not run:  # This plot shows the (main) row and column effects
par(mfrow = c(1, 2), las = 1, mar = c(4.5, 4.4, 2, 0.9) + 0.1)
saved <- plot(rfit, rcol = "blue", ccol = "orange",
              cylab = "Item effects", rylab = "Person effects",
              rxlab = "", cxlab = "")

names(saved@post)  # Some useful output put here
cbind(saved@post$row.effects)
cbind(saved@post$raw.row.effects)
round(cbind(-saved@post$col.effects), dig = 3)
round(cbind(-saved@post$raw.col.effects), dig = 3)
round(matrix(-saved@post$raw.col.effects, ncol = 1,  # Rename for humans
             dimnames = list(colnames(Y.matrix), NULL)), dig = 3)

## End(Not run)
```

`flamingo` *Flamingo Hotel in Sardinia*

Description

The `flamingo` data frame has 4871 rows and 12 columns.

Usage

```
data(flamingo)
```

Format

This data frame contains the following columns:

LOS length of stay, in days.

year year of the stay.

arrdate date of arrival.

depdate date of departure.

seasonindex seasonality index of the period of the stay, a low value means low season, a high value means very high (peak) season.

booking if the guests booked the room through the hotel's internet website (`internet`), the hotel's telephone (`direct`) or a tour operator (`agency`).

roomtype the 16 hotel's room types.

rmtype3 categorization in 3 groups of `roomtype`.

guests number of guests. Does not include any childrenzz; see the two zz `kids` variables below.

arrangement arrangement type: Bed and Breakfast (`BB`), Half Board (`HB`) and Full Board (`FB`).

kids02 number of kids between 0 and 2 years-old.

kids311 number of kids between 3 and 11 years-old.

Details

The Flamingo Hotel is located on the beach in the southern Sardinia, about 40 kilometers from Cagliari. This data concerns stays from early summer 2019 to late summer 2020 with no stays during the winter period in between. Stays longer than 30 days were deleted from the original source. Variable LOS exhibits heaping at the values 7 and 14 days.

Source

The data was obtained with permission from Luca Frigau, University of Cagliari.

See Also

gaitdzeta, airbnb.ac.

Examples

```
## Not run:  with(flamingo, spikeplot(LOS, col = "blue", ylab = "Proportion"))
```

---

genpoisson *Generalized Poisson Regression*

Description

Estimation of the two-parameter generalized Poisson distribution.

Usage

```
genpoisson(llambda = "rhobitlink", ltheta = "loglink",
           ilambda = NULL, itheta = NULL, imethod = 1,
           ishrinkage = 0.95, zero = "lambda")
```

Arguments

`llambda, ltheta`
: Parameter link functions for λ and θ. See Links for more choices. The λ parameter lies at least within the interval $[-1, 1]$; see below for more details, and an alternative link is rhobitlink. The θ parameter is positive, therefore the default is the log link.

`ilambda, itheta`
: Optional initial values for λ and θ. The default is to choose values internally.

`imethod`
: An integer with value 1 or 2 or 3 which specifies the initialization method for the parameters. If failure to converge occurs try another value and/or else specify a value for ilambda and/or itheta.

`ishrinkage, zero`
: See CommonVGAMffArguments for information.

Details

This family function is *not* recommended for use; instead try genpoisson1 or genpoisson2. For underdispersion with respect to the Poisson try the GTE (generally-truncated expansion) method described by Yee and Ma (2020).

The generalized Poisson distribution has density

$$f(y) = \theta(\theta + \lambda y)^{y-1} \exp(-\theta - \lambda y)/y!$$

for $\theta > 0$ and $y = 0, 1, 2, \ldots$. Now $\max(-1, -\theta/m) \leq \lambda \leq 1$ where $m(\geq 4)$ is the greatest positive integer satisfying $\theta + m\lambda > 0$ when $\lambda < 0$ [and then $P(Y = y) = 0$ for $y > m$]. Note the complicated support for this distribution means, for some data sets, the default link for llambda will not always work, and some tinkering may be required to get it running.

As Consul and Famoye (2006) state on p.165, the lower limits on λ and $m \geq 4$ are imposed to ensure that there are at least 5 classes with nonzero probability when λ is negative.

An ordinary Poisson distribution corresponds to $\lambda = 0$. The mean (returned as the fitted values) is $E(Y) = \theta/(1 - \lambda)$ and the variance is $\theta/(1 - \lambda)^3$.

For more information see Consul and Famoye (2006) for a summary and Consul (1989) for full details.

Value

An object of class "vglmff" (see vglmff-class). The object is used by modelling functions such as vglm, and vgam.

Warning

Monitor convergence! This family function is fragile. Don't get confused because theta (and not lambda) here really matches more closely with lambda of dpois.

Note

This family function handles multiple responses. This distribution is potentially useful for dispersion modelling. Convergence problems may occur when lambda is very close to 0 or 1. If a failure occurs then you might want to try something like llambda = extlogitlink(min = -0.9, max = 1) to handle the LHS complicated constraint, and if that doesn't work, try llambda = extlogitlink(min = -0.8, max = 1), etc.

Author(s)

T. W. Yee. Easton Huch derived the EIM and it has been implemented in the weights slot.

References

Consul, P. C. and Famoye, F. (2006). *Lagrangian Probability Distributions*, Boston, USA: Birkhauser.

Jorgensen, B. (1997). *The Theory of Dispersion Models*. London: Chapman & Hall

Consul, P. C. (1989). *Generalized Poisson Distributions: Properties and Applications*. New York, USA: Marcel Dekker.

Yee, T. W. and Ma, C. (2021). Generally–altered, –inflated and –truncated regression, with application to heaped and seeped counts. *Under review*.

See Also

genpoisson1, genpoisson2, poissonff, dpois. dgenpois0, rhobitlink, extlogitlink.

Examples

```
## Not run:
gdata <- data.frame(x2 = runif(nn <- 500))  # NBD data:
gdata <- transform(gdata, y1 = rnbinom(nn, exp(1), mu = exp(2 - x2)))
fit <- vglm(y1 ~ x2, genpoisson, data = gdata, trace = TRUE)
coef(fit, matrix = TRUE)
summary(fit)
## End(Not run)
```

---

hued *Harvard University Degrees Conferred by Student Ethnicity*

Description

A two-way table of counts; there are 7 ethnic groups by 12 degrees.

Usage

```
data(hued)
```

Format

The format is: chr "hued"

Details

The rownames and colnames have been edited. The full names are: Asian/Pacific Islander, Black/Non-Hispanic, Hispanic, International Students, Native American, White/Non-Hispanic, Unknown/Other. The academic year was 2009–2010. GSAS stands for Graduate School of Arts and Sciences. The Other group includes students reported as Two or More Races. See the URL below for more technical details supporting the data.

Source

https://oir.harvard.edu/fact-book

See Also

huie, huse.

Examples

```
print(hued)
```

huie *Harvard University International Enrollments*

Description

A two-way table of counts; there are 12 degrees and 8 areas of the world.

Usage

```
data(huie)
```

Format

The format is: chr "huie"

Details

The rownames and colnames have been edited. The full colnames are: Africa, Asia, Europe, Caribbean and Central and and South America, Middle East, North America, Oceania, Stateless.

The data was for the autumn (Fall) of 2010. GSAS stands for Graduate School of Arts and Sciences. See the URL below for more technical details supporting the data.

Source

https://oir.harvard.edu/fact-book

See Also

hued, huse.

Examples

```
print(huie)
## maybe str(huie) ; plot(huie) ...
```

huse *Harvard University Numbers of Ladder Faculty by School and Ethnicity*

Description

A two-way table of counts; there are 14 schools and 5 race/ethnicities.

Usage

```
data(huse)
```

Format

The format is: chr "huse"

Details

Ladder faculty members of Harvard University are cross-classified by their school and their race/ethnicity. This was for the period 2010–1. Ladder Faculty are defined as Assistant Professors or Convertible Instructors, Associate Professors, and Professors that have been appointed in certain Schools.

Abbreviations: FAS = Faculty of Arts and Sciences = Humanities + Social Sciences + Natural Sciences + SEAS, Natural Sciences = Life Sciences + Physical Sciences, SEAS = School of Engineering and Applied Sciences, HBS = Harvard Business School, HMS = Harvard Medical School, HSPH = Harvard School of Public Health, HLS = Harvard Law School, HKS = Harvard Kennedy School, HGSE = Harvard Graduate School of Education, GSD = Graduate School of Design , HDS = Harvard Divinity School, HSDM = Harvard School of Dental Medicine.

See the URL below for many technical details supporting the data. The table was constructed from pp.31–2 from the source.

Source

https://oir.harvard.edu/fact-book

References

Harvard University Office of the Senior Vice Provost Faculty Development \& Diversity: 2010 Annual Report.

See Also

hued, huie.

Examples

```
print(huse)
## maybe str(huse) ; plot(huse) ...
```

Oalog *One-Altered Logarithmic Distribution*

Description

Density, distribution function, quantile function and random generation for the one-altered logarithmic distribution with parameter pobs1.

Usage

```
doalog(x, shape, pobs1 = 0, log = FALSE)
poalog(q, shape, pobs1 = 0)
qoalog(p, shape, pobs1 = 0)
roalog(n, shape, pobs1 = 0)
```

Arguments

x, q, n, p	Same Unif.
shape, log	Same as Otlog).
pobs1	Probability of (an observed) one, called $pobs1$. The default value of pobs1 = 0 corresponds to the response having a 1-truncated logarithmic distribution.

Details

The probability function of Y is 1 with probability pobs1, else a 1-truncated logarithmic(shape) distribution.

Value

doalog gives the density and poalog gives the distribution function, qoalog gives the quantile function, and roalog generates random deviates.

Note

The argument pobs1 is recycled to the required length, and must have values which lie in the interval $[0, 1]$.

Author(s)

T. W. Yee

See Also

Gaitdlog, oalog, oilog, Otlog.

Examples

```
shape <- 0.75; pobs1 <- 0.10; x <- (-1):7
doalog(x, shape = shape, pobs1 = pobs1)
table(roalog(100, shape = shape, pobs1 = pobs1))

## Not run:  x <- 0:10
barplot(rbind(doalog(x, shape = shape, pobs1 = pobs1),
                dlog(x, shape = shape)),
        beside = TRUE, col = c("blue", "orange"), cex.main = 0.7, las = 1,
        ylab = "Probability", names.arg = as.character(x),
        main = paste("OAL(shape = ", shape, ", pobs1 = ", pobs1,
                   ") [blue] vs",  " Logarithmic(shape = ", shape,
                   ") [orange] densities", sep = ""))
```

```
## End(Not run)
```

---

oalog *One-Altered Logarithmic Distribution*

Description

Fits a one-altered logarithmic distribution based on a conditional model involving a Bernoulli distribution and a 1-truncated logarithmic distribution.

Usage

```
oalog(lpobs1 = "logitlink", lshape = "logitlink",
      type.fitted = c("mean", "shape", "pobs1", "onempobs1"),
      ipobs1 = NULL, gshape = ppoints(8), zero = NULL)
```

Arguments

`lpobs1`	Link function for the parameter p_1 or ϕ, called `pobs1` or `phi` here. See `Links` for more choices.
`lshape`	See `logff` for details.
`gshape, type.fitted`	See `CommonVGAMffArguments` and `fittedvlm` for information.
`ipobs1, zero`	See `CommonVGAMffArguments` for information.

Details

The response Y is one with probability p_1, or Y has a 1-truncated logarithmic distribution with probability $1 - p_1$. Thus $0 < p_1 < 1$, which is modelled as a function of the covariates. The one-altered logarithmic distribution differs from the one-inflated logarithmic distribution in that the former has ones coming from one source, whereas the latter has ones coming from the logarithmic distribution too. The one-inflated logarithmic distribution is implemented in the **VGAM** package. Some people call the one-altered logarithmic a *hurdle* model.

The input can be a matrix (multiple responses). By default, the two linear/additive predictors of `oalog` are $(logit(\phi), logit(s))^T$.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, and `vgam`.

The `fitted.values` slot of the fitted object, which should be extracted by the generic function `fitted`, returns the mean μ (default) which is given by

$$\mu = \phi + (1 - \phi)A$$

where A is the mean of the one-truncated logarithmic distribution. If `type.fitted = "pobs1"` then p_1 is returned.

Note

This family function effectively combines binomialff and otlog into one family function.

Author(s)

T. W. Yee

See Also

Gaitdlog, Oalog, logff, oilog, CommonVGAMffArguments, simulate.vlm.

Examples

```
## Not run: odata <- data.frame(x2 = runif(nn <- 1000))
odata <- transform(odata, pobs1 = logitlink(-1 + 2*x2, inverse = TRUE),
                          shape  = logitlink(-2 + 3*x2, inverse = TRUE))
odata <- transform(odata, y1 = roalog(nn, shape = shape, pobs1 = pobs1),
                          y2 = roalog(nn, shape = shape, pobs1 = pobs1))
with(odata, table(y1))

ofit <- vglm(cbind(y1, y2) ~ x2, oalog, data = odata, trace = TRUE)
coef(ofit, matrix = TRUE)
head(fitted(ofit))
head(predict(ofit))
summary(ofit)

## End(Not run)
```

Oapospois *One-Altered Logarithmic Distribution*

Description

Density, distribution function, quantile function and random generation for the one-altered positive-Poisson distribution with parameter pobs1.

Usage

```
doapospois(x, lambda, pobs1 = 0, log = FALSE)
poapospois(q, lambda, pobs1 = 0)
qoapospois(p, lambda, pobs1 = 0)
roapospois(n, lambda, pobs1 = 0)
```

Arguments

x, q, n, p	Same Unif.
lambda, log	Same as Otpospois).
pobs1	Probability of (an observed) one, called $pobs1$. The default value of pobs1 = 0 corresponds to the response having a 1-truncated positive-Poisson distribution.

Details

The probability function of Y is 1 with probability pobs1, else a 1-truncated positive-Poisson(lambda) distribution.

Value

doapospois gives the density and poapospois gives the distribution function, qoapospois gives the quantile function, and roapospois generates random deviates.

Note

The argument pobs1 is recycled to the required length, and must have values which lie in the interval $[0, 1]$.

Author(s)

T. W. Yee

See Also

oapospoisson, Oipospois, Otpospois.

Examples

```
lambda <- 3; pobs1 <- 0.30; x <- (-1):7
doapospois(x, lambda = lambda, pobs1 = pobs1)
table(roapospois(100, lambda = lambda, pobs1 = pobs1))

## Not run:  x <- 0:10
barplot(rbind(doapospois(x, lambda = lambda, pobs1 = pobs1),
                dpospois(x, lambda = lambda)),
        beside = TRUE, col = c("blue", "orange"), cex.main = 0.7, las = 1,
        ylab = "Probability", names.arg = as.character(x),
        main = paste("OAPP(lambda = ", lambda, ", pobs1 = ", pobs1,
                   ") [blue] vs",  " PosPoisson(lambda = ", lambda,
                   ") [orange] densities", sep = ""))
## End(Not run)
```

---

oapospoisson *One-Altered Positive-Poisson Distribution*

Description

Fits a one-altered positive-Poisson distribution based on a conditional model involving a Bernoulli distribution and a 1-truncated positive-Poisson distribution.

Usage

```
oapospoisson(lpobs1 = "logitlink", llambda = "loglink",
             type.fitted = c("mean", "lambda", "pobs1", "onempobs1"),
             ipobs1 = NULL, zero = NULL)
```

Arguments

lpobs1	Link function for the parameter p_1 or ϕ, called pobs1 or phi here. See Links for more choices.
llambda	See pospoisson for details.
type.fitted	See CommonVGAMffArguments and fittedvlm for information.
ipobs1, zero	See CommonVGAMffArguments for information.

Details

The response Y is one with probability p_1, or Y has a 1-truncated positive-Poisson distribution with probability $1 - p_1$. Thus $0 < p_1 < 1$, which is modelled as a function of the covariates. The one-altered positive-Poisson distribution differs from the one-inflated positive-Poisson distribution in that the former has ones coming from one source, whereas the latter has ones coming from the positive-Poisson distribution too. The one-inflated positive-Poisson distribution is implemented in the **VGAM** package. Some people call the one-altered positive-Poisson a *hurdle* model.

The input can be a matrix (multiple responses). By default, the two linear/additive predictors of oapospoisson are $(logit(\phi), log(\lambda))^T$.

Value

An object of class "vglmff" (see vglmff-class). The object is used by modelling functions such as vglm, and vgam.

The fitted.values slot of the fitted object, which should be extracted by the generic function fitted, returns the mean μ (default) which is given by

$$\mu = \phi + (1 - \phi)A$$

where A is the mean of the one-truncated positive-Poisson distribution. If type.fitted = "pobs1" then p_1 is returned.

Note

This family function effectively combines binomialff and otpospoisson into one family function.

Author(s)

T. W. Yee

See Also

Oapospois, pospoisson, oipospoisson, CommonVGAMffArguments, simulate.vlm.

Examples

```
## Not run: odata <- data.frame(x2 = runif(nn <- 1000))
odata <- transform(odata, pobs1  = logitlink(-1 + 2*x2, inverse = TRUE),
                          lambda =  loglink( 1 + 1*x2, inverse = TRUE))
odata <- transform(odata, y1 = roapospois(nn, lambda = lambda, pobs1 = pobs1),
                          y2 = roapospois(nn, lambda = lambda, pobs1 = pobs1))
with(odata, table(y1))

ofit <- vglm(cbind(y1, y2) ~ x2, oapospoisson, data = odata, trace = TRUE)
coef(ofit, matrix = TRUE)
head(fitted(ofit))
head(predict(ofit))
summary(ofit)

## End(Not run)
```

`Oazeta` *One-Altered Logarithmic Distribution*

Description

Density, distribution function, quantile function and random generation for the one-altered zeta distribution with parameter `pobs1`.

Usage

```
doazeta(x, shape, pobs1 = 0, log = FALSE)
poazeta(q, shape, pobs1 = 0)
qoazeta(p, shape, pobs1 = 0)
roazeta(n, shape, pobs1 = 0)
```

Arguments

`x, q, n, p`	Same `Unif`.
`shape, log`	Same as `Otzeta`).
`pobs1`	Probability of (an observed) one, called $pobs1$. The default value of `pobs1 = 0` corresponds to the response having a 1-truncated zeta distribution.

Details

The probability function of Y is 1 with probability `pobs1`, else a 1-truncated zeta distribution.

Value

`doazeta` gives the density and `poazeta` gives the distribution function, `qoazeta` gives the quantile function, and `roazeta` generates random deviates.

Note

The argument pobs1 is recycled to the required length, and must have values which lie in the interval $[0, 1]$.

Author(s)

T. W. Yee

See Also

oazeta, Oizeta, Otzeta, zeta.

Examples

```
shape <- 1.1; pobs1 <- 0.10; x <- (-1):7
doazeta(x, shape = shape, pobs1 = pobs1)
table(roazeta(100, shape = shape, pobs1 = pobs1))

## Not run:  x <- 0:10
barplot(rbind(doazeta(x, shape = shape, pobs1 = pobs1),
                dzeta(x, shape = shape)),
        beside = TRUE, col = c("blue", "orange"), cex.main = 0.7, las = 1,
        ylab = "Probability", names.arg = as.character(x),
        main = paste("OAZ(shape = ", shape, ", pobs1 = ", pobs1,
                   ") [blue] vs",  " zeta(shape = ", shape,
                   ") [orange] densities", sep = ""))
## End(Not run)
```

---

oazeta *One-Altered Zeta Distribution*

Description

Fits a one-altered zeta distribution based on a conditional model involving a Bernoulli distribution and a 1-truncated zeta distribution.

Usage

```
oazeta(lpobs1 = "logitlink", lshape = "loglink",
       type.fitted = c("mean", "shape", "pobs1", "onempobs1"),
       gshape = exp((-4:3)/4), ishape = NULL, ipobs1 = NULL, zero = NULL)
```

Arguments

lpobs1	Link function for the parameter p_1 or ϕ, called pobs1 or phi here. See Links for more choices.
lshape	See zeta for details.
type.fitted	See CommonVGAMffArguments and fittedvlm for information.
gshape, ishape, ipobs1, zero	See CommonVGAMffArguments for information.

Details

The response Y is one with probability p_1, or Y has a 1-truncated zeta distribution with probability $1 - p_1$. Thus $0 < p_1 < 1$, which is modelled as a function of the covariates. The one-altered zeta distribution differs from the one-inflated zeta distribution in that the former has ones coming from one source, whereas the latter has ones coming from the zeta distribution too. The one-inflated zeta distribution is implemented in the **VGAM** package. Some people call the one-altered zeta a *hurdle* model.

The input can be a matrix (multiple responses). By default, the two linear/additive predictors of `oazeta` are $(logit(\phi), log(shape))^T$.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, and `vgam`.

The `fitted.values` slot of the fitted object, which should be extracted by the generic function `fitted`, returns the mean μ (default) which is given by

$$\mu = \phi + (1 - \phi)A$$

where A is the mean of the one-truncated zeta distribution. If `type.fitted = "pobs1"` then p_1 is returned.

Note

This family function effectively combines `binomialff` and `otzeta` into one family function.

Author(s)

T. W. Yee

See Also

`Oazeta`, `zetaff`, `oizeta`, `otzeta`, `CommonVGAMffArguments`, `simulate.vlm`.

Examples

```
## Not run: odata <- data.frame(x2 = runif(nn <- 1000))
odata <- transform(odata, pobs1 = logitlink(-1 + 2*x2, inverse = TRUE),
                          shape =  loglink( 1 + 1*x2, inverse = TRUE))
odata <- transform(odata, y1 = roazeta(nn, shape = shape, pobs1 = pobs1),
                          y2 = roazeta(nn, shape = shape, pobs1 = pobs1))
with(odata, table(y1))

ofit <- vglm(cbind(y1, y2) ~ x2, oazeta, data = odata, trace = TRUE)
coef(ofit, matrix = TRUE)
head(fitted(ofit))
head(predict(ofit))
summary(ofit)

## End(Not run)
```

Oilog *One-Inflated Logarithmic Distribution*

Description

Density, distribution function, quantile function and random generation for the one-inflated logarithmic distribution with parameter `pstr1`.

Usage

```
doilog(x, shape, pstr1 = 0, log = FALSE)
poilog(q, shape, pstr1 = 0)
qoilog(p, shape, pstr1 = 0)
roilog(n, shape, pstr1 = 0)
```

Arguments

`x, q, p, n`	Same as `Uniform`.
`shape`	Vector of parameters that lie in $(0, 1)$.
`pstr1`	Probability of a structural one (i.e., ignoring the logarithmic distribution), called ϕ. The default value of $\phi = 0$ corresponds to the response having an ordinary logarithmic distribution.
`log`	Same as `Uniform`.

Details

The probability function of Y is 1 with probability ϕ, and $Logarithmic(prob)$ with probability $1 - \phi$. Thus

$$P(Y = 1) = \phi + (1 - \phi)P(W = 1)$$

where W is distributed as a $Logarithmic(shape)$ random variable. The **VGAM** family function `oilog` estimates ϕ by MLE.

Value

`doilog` gives the density, `poilog` gives the distribution function, `qoilog` gives the quantile function, and `roilog` generates random deviates.

Note

The argument `pstr1` is recycled to the required length, and usually has values which lie in the interval $[0, 1]$.

These functions actually allow for the *zero-deflated logarithmic* distribution. Here, `pstr1` is also permitted to lie in the interval `[-dlog(1,shape) / (1 -dlog(1,shape)),0]`. The resulting probability of a unit count is *less than* the nominal logarithmic value, and the use of `pstr1` to stand for the probability of a structural 1 loses its meaning.

When `pstr1` equals `-dlog(1,shape) / (1 -dlog(1,shape))` this corresponds to the 1-truncated logarithmic distribution.

Author(s)

T. W. Yee

See Also

Gaitdlog, oilog, rlog, logff, Otlog.

Examples

```
shape <- 0.5; pstr1 <- 0.3; x <- (-1):7
(ii <- doilog(x, shape, pstr1 = pstr1))
max(abs(poilog(1:200, shape) -
        cumsum(shape^(1:200) / (-(1:200) * log1p(-shape))))  # Should be 0

## Not run:  x <- 0:10
par(mfrow = c(2, 1))  # One-Inflated logarithmic
barplot(rbind(doilog(x, shape, pstr1 = pstr1), dlog(x, shape)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("OILogff(", shape, ", pstr1 = ", pstr1, ") (blue) vs",
                     " Logff(", shape, ") (orange)", sep = ""),
        names.arg = as.character(x))

deflat.limit <- -dlog(1, shape) / plog(1, shape, lower.tail = FALSE)
newpstr1 <- round(deflat.limit, 3) + 0.001  # Inside but near the boundary
barplot(rbind(doilog(x, shape, pstr1 = newpstr1),
                dlog(x, shape)),
        beside = TRUE, col = c("blue","orange"),
        main = paste("ODLogff(", shape, ", pstr1 = ", newpstr1, ") (blue) vs",
                     " Logff(", shape, ") (orange)", sep = ""),
        names.arg = as.character(x))
## End(Not run)
```

`oilog` *One-inflated Logarithmic Distribution Family Function*

Description

Fits a 1-inflated logarithmic distribution.

Usage

```
oilog(lpstr1 = "logitlink", lshape = "logitlink",
      type.fitted = c("mean", "shape", "pobs1", "pstr1", "onempstr1"),
      ishape = NULL, gpstr1 = ppoints(8), gshape = ppoints(8), zero = NULL)
```

Arguments

`lpstr1, lshape`	Link functions. For `lpstr1`: the same idea as `zipoisson` except it applies to a structural 1.
`gpstr1, gshape, ishape`	For initial values. See `CommonVGAMffArguments` for information.
`type.fitted, zero`	See `CommonVGAMffArguments` for information.

Details

The 1-inflated logarithmic distribution is a mixture distribution of the logarithmic distribution with some probability of obtaining a (structural) 1. Thus there are two sources for obtaining the value 1. This distribution is written here in a way that retains a similar notation to the one-inflated positive-Poisson, i.e., the probability $P[Y = 1]$ involves another parameter ϕ. See `oipospoisson`.

This family function can handle multiple responses.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, `rrvglm` and `vgam`.

Author(s)

Thomas W. Yee

See Also

`Gaitdlog`, `Oilog`, `logff`, `Oizeta`.

Examples

```
## Not run:  odata <- data.frame(x2 = runif(nn <- 1000))  # Artificial data
odata <- transform(odata, pstr1 = logitlink(-1 + x2, inverse = TRUE), shape = 0.5)
odata <- transform(odata, y1 = roilog(nn, shape, pstr1 = pstr1))
with(odata, table(y1))
fit1 <- vglm(y1 ~ x2, oilog(zero = "shape"), data = odata, trace = TRUE)
coef(fit1, matrix = TRUE)

## End(Not run)
```

`Oiposbinom` *One-Inflated Positive Binomial Distribution*

Description

Density, distribution function, quantile function and random generation for the one-inflated positive binomial distribution with parameter `pstr1`.

Usage

```
doiposbinom(x, size, prob, pstr1 = 0, log = FALSE)
poiposbinom(q, size, prob, pstr1 = 0)
qoiposbinom(p, size, prob, pstr1 = 0)
roiposbinom(n, size, prob, pstr1 = 0)
```

Arguments

`x, p, q, n`	Same as `Posbinom`.
`size, prob`	Same as `Posbinom`.
`pstr1`	Probability of a structural one (i.e., ignoring the positive binomial distribution), called ϕ. The default value of $\phi = 0$ corresponds to the response having a positive binomial distribution. However, `pstr1` can also be negative, in which case it ceases its interpretation as a probability, and this is known as *one-deflation*.
`log`	Logical. Return the logarithm of the answer?

Details

The probability function of Y is 1 with probability ϕ, and $PosBinomial(size, prob)$ with probability $1 - \phi$. Thus

$$P(Y = 1) = \phi + (1 - \phi)P(W = 1)$$

where W is distributed as a positive $binomial(size, prob)$ random variable.

Value

`doiposbinom` gives the density, `poiposbinom` gives the distribution function, `qoiposbinom` gives the quantile function, and `roiposbinom` generates random deviates.

Note

The argument `pstr1` is recycled to the required length, and usually has values which lie in the interval $[0, 1]$.

These functions actually allow for the *zero-deflated binomial* distribution. Here, `pstr1` is also permitted to lie in the interval $[-A, 0]$ for some positive quantity A. The resulting probability of a unit value is *less than* the nominal positive binomial value, and the use of `pstr1` to stand for the probability of a structural 1 loses its meaning.

If `pstr1` equals A then this corresponds to the 0- and 1-truncated binomial distribution.

Author(s)

T. W. Yee

See Also

posbinomial, dbinom, binomialff.

Examples

```
size <- 10; prob <- 0.2; pstr1 <- 0.4; x <- (-1):size
(ii <- doiposbinom(x, size, prob, pstr1 = pstr1))
table(roiposbinom(100, size, prob, pstr1 = pstr1))
round(doiposbinom(x  , size, prob, pstr1 = pstr1) * 100)  # Should be similar

## Not run:  x <- 0:size
par(mfrow = c(2, 1))  # One-Inflated Positive Binomial
barplot(rbind(doiposbinom(x, size, prob, pstr1 = pstr1),
              dposbinom(x, size, prob)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("OIPB(", size, ",", prob, ", pstr1 = ", pstr1, ") (blue) vs",
                     " PosBinomial(", size, ",", prob, ") (orange)", sep = ""),
        names.arg = as.character(x))

# Zero-deflated Pos Binomial
deflat.limit <- -dposbinom(1, size, prob) / (1 - dposbinom(1, size, prob))
deflat.limit <- size * prob / (1 + (size-1) * prob - 1 / (1-prob)^(size-1))
newpstr1 <- round(deflat.limit, 3) + 0.001  # A little from the boundary
barplot(rbind(doiposbinom(x, size, prob, pstr1 = newpstr1),
               dposbinom(x, size, prob)),
        beside = TRUE, col = c("blue","orange"),
        main = paste("ODPB(", size, ",", prob, ", pstr1 = ", newpstr1, ") (blue) vs",
                     " PosBinomial(", size, ",", prob, ") (orange)", sep = ""),
        names.arg = as.character(x))
## End(Not run)
```

---

oiposbinomial *One-Inflated Positive Binomial Distribution Family Function*

Description

Fits a one-inflated positive binomial distribution by maximum likelihood estimation.

Usage

```
oiposbinomial(lpstr1 = "logitlink", lprob = "logitlink",
              type.fitted = c("mean", "prob", "pobs1", "pstr1", "onempstr1"),
              iprob = NULL, gpstr1 = ppoints(9), gprob  = ppoints(9),
              multiple.responses = FALSE, zero = NULL)
```

Arguments

`lpstr1, lprob`	Link functions for the parameter ϕ and the positive binomial probability μ parameter. See `Links` for more choices. See `CommonVGAMffArguments` also. For the one-*deflated* model see below.
`type.fitted`	See `CommonVGAMffArguments` and `fittedvlm`.
`iprob, gpstr1, gprob`	For initial values; see `CommonVGAMffArguments`.
`multiple.responses`	Logical. See `binomialff` and `posbinomial`.
`zero`	See `CommonVGAMffArguments` for information.

Details

These functions are based on

$$P(Y = y) = \phi + (1 - \phi)N\mu(1 - \mu)^N/(1 - (1 - \mu)^N),$$

for $y = 1/N$, and

$$P(Y = y) = (1 - \phi)\binom{N}{Ny}\mu^{Ny}(1 - \mu)^{N(1-y)}/(1 - (1 - \mu)^N).$$

for $y = 2/N, \ldots, 1$. That is, the response is a sample proportion out of N trials, and the argument `size` in `roiposbinom` is N here. Ideally $N > 2$ is needed. The parameter ϕ is the probability of a structural one, and it satisfies $0 < \phi < 1$ (usually). The mean of Y is $E(Y) = \phi + (1 - \phi)\mu/(1 - (1 - \mu)^N)$ and these are returned as the default fitted values. By default, the two linear/additive predictors for `oiposbinomial()` are $(logit(\phi), logit(\mu))^T$.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm` and `vgam`.

Note

The response variable should have one of the formats described by `binomialff`, e.g., a factor or two column matrix or a vector of sample proportions with the `weights` argument specifying the values of N.

To work well, one ideally needs large values of N and μ much greater than 0, i.e., the larger N and μ are, the better. If $N = 1$ then the model is unidentifiable since the number of parameters is excessive.

Estimated probabilities of a structural one and an observed one are returned, as in `zipoisson`.

The one-*deflated* positive binomial distribution might be fitted by setting `lpstr1 = "identitylink"`, albeit, not entirely reliably. See `zipoisson` for information that can be applied here.

Author(s)

T. W. Yee

See Also

roiposbinom, posbinomial, binomialff, rbinom.

Examples

```
size <- 10  # Number of trials; N in the notation above
nn <- 200
odata <- data.frame(pstr1  = logitlink( 0, inverse = TRUE),  # 0.50
                    mubin1 = logitlink(-1, inverse = TRUE),  # Mean of usual binomial
                    svec   = rep(size, length = nn),
                    x2     = runif(nn))
odata <- transform(odata,
                   mubin2 = logitlink(-1 + x2, inverse = TRUE))
odata <- transform(odata,
                   y1 = roiposbinom(nn, svec, pr = mubin1, pstr1 = pstr1),
                   y2 = roiposbinom(nn, svec, pr = mubin2, pstr1 = pstr1))
with(odata, table(y1))
fit1 <- vglm(y1 / svec ~  1, oiposbinomial, data = odata,
             weights = svec, trace = TRUE, crit = "coef")
fit2 <- vglm(y2 / svec ~ x2, oiposbinomial, data = odata,
             weights = svec, trace = TRUE)

coef(fit1, matrix = TRUE)
Coef(fit1)  # Useful for intercept-only models
head(fitted(fit1, type = "pobs1"))  # Estimate of P(Y = 1)
head(fitted(fit1))
with(odata, mean(y1))  # Compare this with fitted(fit1)
summary(fit1)
```

Oipospois *One-Inflated Positive Poisson Distribution*

Description

Density, distribution function, quantile function and random generation for the one-inflated positive Poisson distribution with parameter pstr1.

Usage

```
doipospois(x, lambda, pstr1 = 0, log = FALSE)
poipospois(q, lambda, pstr1 = 0)
qoipospois(p, lambda, pstr1 = 0)
roipospois(n, lambda, pstr1 = 0)
```

Arguments

x, p, q, n	Same as Pospois.
lambda	Vector of positive means.

pstr1	Probability of a structural one (i.e., ignoring the positive Poisson distribution), called ϕ. The default value of $\phi = 0$ corresponds to the response having a positive Poisson distribution.
log	Logical. Return the logarithm of the answer?

Details

The probability function of Y is 1 with probability ϕ, and $PosPoisson(\lambda)$ with probability $1 - \phi$. Thus

$$P(Y = 1) = \phi + (1 - \phi)P(W = 1)$$

where W is distributed as a positive $Poisson(\lambda)$ random variate.

Value

doipospois gives the density, poipospois gives the distribution function, qoipospois gives the quantile function, and roipospois generates random deviates.

Note

The argument pstr1 is recycled to the required length, and usually has values which lie in the interval $[0, 1]$.

These functions actually allow for the *zero-deflated Poisson* distribution. Here, pstr1 is also permitted to lie in the interval [-lambda / (expm1(lambda) -lambda),0]. The resulting probability of a unit count is *less than* the nominal positive Poisson value, and the use of pstr1 to stand for the probability of a structural 1 loses its meaning.

When pstr1 equals -lambda / (expm1(lambda) -lambda) this corresponds to the 0- and 1-truncated Poisson distribution.

Author(s)

T. W. Yee

See Also

Pospois, oapospoisson, oipospoisson, otpospoisson, pospoisson, dpois, poissonff.

Examples

```
lambda <- 3; pstr1 <- 0.2; x <- (-1):7
(ii <- doipospois(x, lambda, pstr1 = pstr1))
table(roipospois(100, lambda, pstr1 = pstr1))
round(doipospois(1:10, lambda, pstr1 = pstr1) * 100)  # Should be similar

## Not run:  x <- 0:10
par(mfrow = c(2, 1))  # One-Inflated Positive Poisson
barplot(rbind(doipospois(x, lambda, pstr1 = pstr1), dpospois(x, lambda)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("OIPP(", lambda, ", pstr1 = ", pstr1, ") (blue) vs",
                     " PosPoisson(", lambda, ") (orange)", sep = ""),
```

```
        names.arg = as.character(x))

deflat.limit <- -lambda / (expm1(lambda) - lambda)  # 0-deflated Pos Poisson
newpstr1 <- round(deflat.limit, 3) + 0.001  # Inside and near the boundary
barplot(rbind(doipospois(x, lambda, pstr1 = newpstr1),
              dpospois(x, lambda)),
        beside = TRUE, col = c("blue","orange"),
        main = paste("ODPP(", lambda, ", pstr1 = ", newpstr1, ") (blue) vs",
                     " PosPoisson(", lambda, ") (orange)", sep = ""),
        names.arg = as.character(x))
## End(Not run)
```

`oipospoisson` *One-inflated Positive Poisson Distribution Family Function*

Description

Fits a 1-inflated positive Poisson distribution.

Usage

```
oipospoisson(lpstr1 = "logitlink", llambda = "loglink",
   type.fitted = c("mean", "lambda", "pobs1", "pstr1", "onempstr1"),
   ilambda = NULL, gpstr1 = (1:19)/20, gprobs.y = (1:19)/20,
   imethod = 1, zero = NULL)
```

Arguments

`lpstr1`, `llambda`
: For `lpstr1`: the same idea as `zipoisson` except it applies to a structural 1.

`ilambda`, `gpstr1`, `gprobs.y`, `imethod`
: For initial values. See `CommonVGAMffArguments` for information.

`type.fitted`, `zero`
: See `CommonVGAMffArguments` for information.

Details

The 1-inflated positive Poisson distribution is a mixture distribution of the positive (0-truncated) Poisson distribution with some probability of obtaining a (structural) 1. Thus there are two sources for obtaining the value 1. It is similar to a zero-inflated Poisson model, except the Poisson is replaced by a positive Poisson and the 0 is replaced by 1. This distribution is written here in a way that retains a similar notation to the zero-inflated Poisson, i.e., the probability $P[Y = 1]$ involves another parameter ϕ. See `zipoisson`.

This family function can handle multiple responses.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, `rrvglm` and `vgam`.

Warning

Under- or over-flow may occur if the data is ill-conditioned.

Author(s)

Thomas W. Yee

See Also

Oipospois, pospoisson, oapospoisson, otpospoisson, zipoisson, poissonff, simulate.vlm.

Examples

```
## Not run: set.seed(1)
pdata <- data.frame(x2 = runif(nn <- 1000))  # Artificial data
pdata <- transform(pdata, pstr1 = 0.5, lambda = exp(3 - x2))
pdata <- transform(pdata, y1 = roipospois(nn, lambda, pstr1 = pstr1))
with(pdata, table(y1))
fit1 <- vglm(y1 ~ x2, oipospoisson, data = pdata, trace = TRUE)
coef(fit1, matrix = TRUE)

## End(Not run)
```

Oizeta *One-Inflated Zeta Distribution*

Description

Density, distribution function, quantile function and random generation for the one-inflated zeta distribution with parameter pstr1.

Usage

```
doizeta(x, shape, pstr1 = 0, log = FALSE)
poizeta(q, shape, pstr1 = 0)
qoizeta(p, shape, pstr1 = 0)
roizeta(n, shape, pstr1 = 0)
```

Arguments

x, q, p, n	Same as Uniform.
shape	Vector of positive shape parameters.
pstr1	Probability of a structural one (i.e., ignoring the zeta distribution), called ϕ. The default value of $\phi = 0$ corresponds to the response having an ordinary zeta distribution.
log	Same as Uniform.

Details

The probability function of Y is 1 with probability ϕ, and $Zeta(shape)$ with probability $1 - \phi$. Thus

$$P(Y = 1) = \phi + (1 - \phi)P(W = 1)$$

where W is distributed as a $zeta(shape)$ random variable.

Value

`doizeta` gives the density, `poizeta` gives the distribution function, `qoizeta` gives the quantile function, and `roizeta` generates random deviates.

Note

The argument `pstr1` is recycled to the required length, and usually has values which lie in the interval $[0, 1]$.

These functions actually allow for the *zero-deflated zeta* distribution. Here, `pstr1` is also permitted to lie in the interval `[-dzeta(1,shape) / (1 -dzeta(1,shape)),0]`. The resulting probability of a unit count is *less than* the nominal zeta value, and the use of `pstr1` to stand for the probability of a structural 1 loses its meaning.

When `pstr1` equals `-dzeta(1,shape) / (1 -dzeta(1,shape))` this corresponds to the 1-truncated zeta distribution.

Author(s)

T. W. Yee

See Also

`Zeta`, `zetaff`. `Otzeta`,

Examples

```
shape <- 1.5; pstr1 <- 0.3; x <- (-1):7
(ii <- doizeta(x, shape, pstr1 = pstr1))
max(abs(poizeta(1:200, shape) -
        cumsum(1/(1:200)^(1+shape)) / zeta(shape+1)))  # Should be 0

## Not run:  x <- 0:10
par(mfrow = c(2, 1))  # One-Inflated zeta
barplot(rbind(doizeta(x, shape, pstr1 = pstr1), dzeta(x, shape)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("OIZeta(", shape, ", pstr1 = ", pstr1, ") (blue) vs",
                     " Zeta(", shape, ") (orange)", sep = ""),
        names.arg = as.character(x))

deflat.limit <- -dzeta(1, shape) / pzeta(1, shape, lower.tail = FALSE)
newpstr1 <- round(deflat.limit, 3) + 0.001  # Inside but near the boundary
barplot(rbind(doizeta(x, shape, pstr1 = newpstr1),
              dzeta(x, shape)),
```

```
        beside = TRUE, col = c("blue","orange"),
        main = paste("ODZeta(", shape, ", pstr1 = ", newpstr1, ") (blue) vs",
                     " Zeta(", shape, ") (orange)", sep = ""),
        names.arg = as.character(x))
## End(Not run)
```

`oizeta` *One-inflated Zeta Distribution Family Function*

Description

Fits a 1-inflated zeta distribution.

Usage

```
oizeta(lpstr1 = "logitlink", lshape = "loglink",
  type.fitted = c("mean", "shape", "pobs1", "pstr1", "onempstr1"),
  ishape = NULL, gpstr1 = ppoints(8), gshape = exp((-3:3) / 4), zero = NULL)
```

Arguments

`lpstr1`, `lshape` For `lpstr1`: the same idea as `zipoisson` except it applies to a structural 1.

`gpstr1`, `gshape`, `ishape`
For initial values. See `CommonVGAMffArguments` for information.

`type.fitted`, `zero`
See `CommonVGAMffArguments` for information.

Details

The 1-inflated zeta distribution is a mixture distribution of the zeta distribution with some probability of obtaining a (structural) 1. Thus there are two sources for obtaining the value 1. This distribution is written here in a way that retains a similar notation to the zero-inflated Poisson, i.e., the probability $P[Y = 1]$ involves another parameter ϕ. See `zipoisson`.

This family function can handle multiple responses.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, `rrvglm` and `vgam`.

Warning

Under- or over-flow may occur if the data is ill-conditioned. Lots of data is needed to estimate the parameters accurately. Usually, probably the shape parameter is best modelled as intercept-only.

Author(s)

Thomas W. Yee

See Also

Oizeta, zetaff, oazeta, otzeta, diffzeta, zeta, Oizipf.

Examples

```
## Not run:  odata <- data.frame(x2 = runif(nn <- 1000))  # Artificial data
odata <- transform(odata, pstr1 = logitlink(-1 + x2, inverse = TRUE), shape = exp(-0.5))
odata <- transform(odata, y1 = roizeta(nn, shape, pstr1 = pstr1))
with(odata, table(y1))
fit1 <- vglm(y1 ~ x2, oizeta(zero = "shape"), data = odata, trace = TRUE)
coef(fit1, matrix = TRUE)

## End(Not run)
```

Oizipf *One-Inflated Zipf Distribution*

Description

Density, distribution function, quantile function and random generation for the one-inflated Zipf distribution with parameter pstr1.

Usage

```
doizipf(x, N, shape, pstr1 = 0, log = FALSE)
poizipf(q, N, shape, pstr1 = 0)
qoizipf(p, N, shape, pstr1 = 0)
roizipf(n, N, shape, pstr1 = 0)
```

Arguments

x, q, p, n	Same as Uniform.
N, shape	See Zipf.
pstr1	Probability of a structural one (i.e., ignoring the Zipf distribution), called ϕ. The default value of $\phi = 0$ corresponds to the response having an ordinary Zipf distribution.
log	Same as Uniform.

Details

The probability function of Y is 1 with probability ϕ, and $Zipf(N, s)$ with probability $1 - \phi$. Thus

$$P(Y = 1) = \phi + (1 - \phi)P(W = 1)$$

where W is distributed as a $Zipf(N, s)$ random variable. The **VGAM** family function oizeta estimates the two parameters of this model by Fisher scoring.

Value

doizipf gives the density, poizipf gives the distribution function, qoizipf gives the quantile function, and roizipf generates random deviates.

Note

The argument pstr1 is recycled to the required length, and usually has values which lie in the interval $[0, 1]$.

These functions actually allow for the *zero-deflated Zipf* distribution. Here, pstr1 is also permitted to lie in the interval [-dzipf(1,N,s) / (1 -dzipf(1,N,s)),0]. The resulting probability of a unit count is *less than* the nominal zipf value, and the use of pstr1 to stand for the probability of a structural 1 loses its meaning.

When pstr1 equals -dzipf(1,N,s) / (1 -dzipf(1,N,s)) this corresponds to the 1-truncated zipf distribution.

Author(s)

T. W. Yee

See Also

oizeta. Zipf, zipf, Oizeta.

Examples

```
N <- 10; shape <- 1.5; pstr1 <- 0.3; x <- (-1):N
(ii <- doizipf(x, N, shape, pstr1 = pstr1))

## Not run:  x <- 0:10
par(mfrow = c(2, 1))  # One-Inflated zipf
barplot(rbind(doizipf(x, N, shape, pstr1 = pstr1), dzipf(x, N, shape)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("OIZipf(", N, ", ", shape, ", pstr1 = ", pstr1, ") (blue) vs",
                     " Zipf(", N, ", ", shape, ") (orange)", sep = ""),
        names.arg = as.character(x))

deflat.limit <- -dzipf(1, N, shape) / (1 - dzipf(1, N, shape))
newpstr1 <- round(deflat.limit, 3) + 0.001  # Inside but near the boundary
barplot(rbind(doizipf(x, N, shape, pstr1 = newpstr1),
                dzipf(x, N, shape)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("ODZipf(", N, ", ", shape, ", pstr1 = ", newpstr1, ") (blue) vs",
                     " Zipf(", N, ", ", shape, ") (orange)", sep = ""),
        names.arg = as.character(x))
## End(Not run)
```

oizipf *One-inflated Zipf Distribution Family Function*

Description

Fits a 1-inflated Zipf distribution.

Usage

```
oizipf(N = NULL, lpstr1 = "logitlink", lshape = "loglink",
  type.fitted = c("mean", "shape", "pobs1", "pstr1", "onempstr1"),
  ishape = NULL, gpstr1 = ppoints(8), gshape = exp((-3:3) / 4), zero = NULL)
```

Arguments

N	Same as zipf.
lpstr1, lshape	For lpstr1: the same idea as zipoisson except it applies to a structural 1.
gpstr1, gshape, ishape	For initial values. See CommonVGAMffArguments for information.
type.fitted, zero	See CommonVGAMffArguments for information.

Details

The 1-inflated Zipf distribution is a mixture distribution of the Zipf distribution with some probability of obtaining a (structural) 1. Thus there are two sources for obtaining the value 1. This distribution is written here in a way that retains a similar notation to the zero-inflated Poisson, i.e., the probability $P[Y = 1]$ involves another parameter ϕ. See zipoisson.

This family function can handle multiple responses.

Value

An object of class "vglmff" (see vglmff-class). The object is used by modelling functions such as vglm, rrvglm and vgam.

Warning

Under- or over-flow may occur if the data is ill-conditioned. Lots of data is needed to estimate the parameters accurately. Usually, probably the shape parameter is best modelled as intercept-only.

Author(s)

Thomas W. Yee

See Also

Oizipf. zipf, Oizeta.

Examples

```
## Not run:  odata <- data.frame(x2 = runif(nn <- 1000))  # Artificial data
odata <- transform(odata, pstr1 = logitlink(-1 + x2, inverse = TRUE),
                          myN   = 10,
                          shape = exp(-0.5))
odata <- transform(odata, y1 = roizipf(nn, N = myN, s = shape, pstr1 = pstr1))
with(odata, table(y1))
fit1 <- vglm(y1 ~ x2, oizipf(zero = "shape"), data = odata, trace = TRUE)
coef(fit1, matrix = TRUE)

## End(Not run)
```

---

oly12 *2012 Summer Olympics: Individuals Data*

Description

Individual data for the Summer 2012 Olympic Games.

Usage

```
data(oly12)
```

Format

A data frame with 10384 observations on the following 14 variables.

`Name` The individual competitor's name.

`Country` Country.

`Age` A numeric vector, age in years.

`Height` A numeric vector, height in m.

`Weight` A numeric vector, weight in kg.

`Sex` A factor with levels `F` and `M`.

`DOB` A Date, date of birth.

`PlaceOB` Place of birth.

`Gold` Numeric vector, number of such medals won.

`Silver` Similar to `Gold`.

`Bronze` Similar to `Gold`.

`Total` A numeric vector, total number of medals.

`Sport` A factor with levels `Archery`, `Athletics`, `Athletics`, `Triathlon`, `Badminton`, etc.

`Event` The sporting event.

Details

This data set represents a very small modification of a `.csv` spreadsheet from the source below. Height has been converted to meters, and date of birth is of a `"Date"` class (see `as.Date`). A few non-ASCII characters have been replaced by some ASCII sequence (yet to be fixed up properly).

Some competitors share the same name. Some errors in the data are likely to exist.

Source

Downloaded from `http://www.guardian.co.uk/sport/series/london-2012-olympics-data` in 2013-03; more recently it has changed to https://www.theguardian.com/sport/series/london-2012-olympics-data.

Examples

```
data(oly12)
mtab <- with(oly12, table(Country, Gold))
(mtab <- head(sort(mtab[, "1"] + 2 * mtab[, "2"], decreasing = TRUE), 10))

## Not run:
barplot(mtab, col = "gold", cex.names = 0.8, names = abbreviate(names(mtab)),
        beside = TRUE, main = "2012 Summer Olympic Final Gold Medal Count",
        ylab = "Gold medal count", las = 1, sub = "Top 10 countries")

## End(Not run)
```

Otlog *One-truncated Logarithmic Distribution*

Description

Density, distribution function, quantile function, and random generation for the one-truncated logarithmic distribution.

Usage

```
dotlog(x, shape, log = FALSE)
potlog(q, shape, log.p = FALSE)
qotlog(p, shape)
rotlog(n, shape)
```

Arguments

`x, q`	Vector of quantiles. For the density, it should be a vector with integer values > 1 in order for the probabilities to be positive.
`p`	vector of probabilities.
`n`	number of observations. Same as in `runif`.

shape	The parameter value c described in in logff. Here it is called shape because $0 < c < 1$ is the range.
log, log.p	Logical. If log.p = TRUE then all probabilities p are given as log(p).

Details

The one-truncated logarithmic distribution is a logarithmic distribution but with the probability of a one being zero. The other probabilities are scaled to add to unity. Some more details are given in logff.

Value

dotlog gives the density, potlog gives the distribution function, qotlog gives the quantile function, and rotlog generates random deviates.

Note

Given some response data, the **VGAM** family function otlog estimates the parameter shape. Function potlog() suffers from the problems that plog sometimes has.

Author(s)

T. W. Yee

See Also

Gaitdlog, otlog, rlog, Oilog.

Examples

```
dotlog(1:20, 0.5)
rotlog(20, 0.5)

## Not run:  shape <- 0.8; x <- 1:10
plot(x, dotlog(x, shape = shape), type = "h", ylim = 0:1,
     sub = "shape=0.8", las = 1, col = "blue", ylab = "Probability",
     main = "1-truncated logarithmic distribution: blue=PMF; orange=CDF")
lines(x + 0.1, potlog(x, shape = shape), col = "orange", lty = 3, type = "h")
## End(Not run)
```

---

otlog *One-truncated Logarithmic Distribution*

Description

Estimating the (single) parameter of the 1-truncated logarithmic distribution.

Usage

```
otlog(lshape = "logitlink", gshape = ppoints(8), zero = NULL)
```

Arguments

`lshape, gshape, zero`
: Same as `logff`.

Details

The 1-truncated logarithmic distribution is a logarithmic distribution but with the probability of a one being zero. The other probabilities are scaled to add to unity. Some more details can be found at `logff`. Multiple responses are permitted.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, and `vgam`.

Author(s)

T. W. Yee

See Also

`Gaitdlog`, `Otlog`, `logff`, `oalog`, `oilog`, `simulate.vlm`.

Examples

```
## Not run:
odata <- data.frame(y1 = rotlog(n = 1000, shape = logitlink(1/3, inverse = TRUE)))
ofit <- vglm(y1 ~ 1, otlog, data = odata, trace = TRUE, crit = "c")
coef(ofit, matrix = TRUE)
Coef(ofit)
with(odata,
    hist(y1, shape = TRUE, breaks = seq(0.5, max(y1) + 0.5, by = 1),
         border = "blue"))
x <- seq(1, with(odata, max(y1)), by = 1)
with(odata, lines(x, dotlog(x, Coef(ofit)[1]), col = "orange", type = "h", lwd = 2))
## End(Not run)
```

Otpospois *One-truncated Positive-Poisson Distribution*

Description

Density, distribution function, quantile function, and random generation for the one-truncated positive-Poisson distribution.

Usage

```
dotpospois(x, lambda, log = FALSE)
potpospois(q, lambda, log.p = FALSE)
qotpospois(p, lambda)
rotpospois(n, lambda)
```

Arguments

`x, q, p, n`	Same as `Pospois`.
`lambda, log, log.p`	Same as `Pospois`.

Details

The one-truncated positive-Poisson is a Poisson distribution but with the probability of a one and a zero being 0. That is, its support is 2, 3, ... The other probabilities are scaled to add to unity. Some more details are given in `pospoisson`.

Value

`dotpospois` gives the density, `potpospois` gives the distribution function, `qotpospois` gives the quantile function, and `rotpospois` generates random deviates.

Note

Given some response data, the **VGAM** family function `otpospoisson` estimates the parameter `lambda`.

Author(s)

T. W. Yee

See Also

`otpospoisson`, `Pospois`, `Oipospois`.

Examples

```
dotpospois(1:20, 0.5)
rotpospois(20, 0.5)

## Not run:  lambda <- 4; x <- 1:10
plot(x, dotpospois(x, lambda = lambda), type = "h", ylim = 0:1,
     sub = "lambda=4", las = 1, col = "blue", ylab = "Probability",
     main = "1-truncated positive-Poisson distribution: blue=PMF; orange=CDF")
lines(x + 0.1, potpospois(x, lambda = lambda), col = "orange", lty = 3, type = "h")
## End(Not run)
```

---

otpospoisson *One-truncated Poisson Distribution*

Description

Estimating the (single) parameter of the 1-truncated positive Poisson distribution.

Usage

```
otpospoisson(llambda = "loglink",
   type.fitted = c("mean", "lambda", "prob0", "prob1"),
   ilambda = NULL, imethod = 1, zero = NULL)
```

Arguments

`llambda`, `type.fitted`, `ilambda`
: Same as `pospoisson`.

`imethod`, `zero`
: Same as `pospoisson`.

Details

The 1-truncated positive Poisson distribution has support on 2, 3, ... It is a Poisson distribution but with the probability of a one or zero being 0. The other probabilities are scaled to add to unity. Some more details can be found at `pospoisson`. Multiple responses are permitted.

Value

An object of class `"vglmff"` (see `vglmff-class`). The object is used by modelling functions such as `vglm`, and `vgam`.

Author(s)

T. W. Yee

See Also

`Otpospois`, `oipospoisson`, `simulate.vlm`.

Examples

```
## Not run:
odata <- data.frame(y1 = rotpospois(n = 1000, lambda = loglink(1, inverse = TRUE)))
ofit <- vglm(y1 ~ 1, otpospoisson, data = odata, trace = TRUE, crit = "c")
coef(ofit, matrix = TRUE)
Coef(ofit)
with(odata,
    hist(y1, prob = TRUE, breaks = seq(0.5, max(y1) + 0.5, by = 1),
         border = "blue"))
x <- seq(1, with(odata, max(y1)), by = 1)
with(odata, lines(x, dotpospois(x, Coef(ofit)[1]), col = "orange", type = "h", lwd = 2))
## End(Not run)
```

Otzeta *One-truncated Zeta Distribution*

Description

Density, distribution function, quantile function, and random generation for the one-truncated zeta distribution.

Usage

```
dotzeta(x, shape, log = FALSE)
potzeta(q, shape, log.p = FALSE)
qotzeta(p, shape)
rotzeta(n, shape)
```

Arguments

x, q, p, n	Same as in runif.
shape	The positive shape parameter described in in zetaff. Here it is called shape because it is positive.
log, log.p	Same as in runif.

Details

The one-truncated zeta distribution is a zeta distribution but with the probability of a one being zero. The other probabilities are scaled to add to unity. Some more details are given in zetaff.

Value

dotzeta gives the density, potzeta gives the distribution function, qotzeta gives the quantile function, and rotzeta generates random deviates.

Note

Given some response data, the **VGAM** family function otzeta estimates the parameter shape.

Author(s)

T. W. Yee

See Also

Otzeta, zetaff, Oizeta.

Examples

```
dotzeta(1:20, 0.5)
rotzeta(20, 0.5)

## Not run:  shape <- 0.8; x <- 1:10
plot(x, dotzeta(x, shape = shape), type = "h", ylim = 0:1,
     sub = "shape=0.8", las = 1, col = "blue", ylab = "Probability",
     main = "1-truncated zeta distribution: blue=PMF; orange=CDF")
lines(x + 0.1, potzeta(x, shape = shape), col = "orange", lty = 3, type = "h")
## End(Not run)
```

---

otzeta *One-truncated Zeta Distribution Family Function*

Description

Estimates the parameter of the 1-truncated zeta distribution.

Usage

```
otzeta(lshape = "loglink", ishape = NULL, gshape = exp((-4:3)/4), zero = NULL)
```

Arguments

`lshape, ishape, gshape, zero`
: Same as zetaff.

Details

The 1-truncated zeta distribution is the ordinary zeta distribution but with the probability of one being 0. Thus the other probabilities are scaled up (i.e., divided by $1 - P[Y = 1]$). The mean is returned by default as the fitted values. More details can be found at zetaff. Multiple responses are handled.

Value

An object of class "vglmff" (see vglmff-class). The object is used by modelling functions such as vglm, and vgam.

Author(s)

T. W. Yee

See Also

Otzeta, zetaff, oizeta, diffzeta, zeta, dzeta, hzeta, zipf.

Examples

```
## Not run: odata <- data.frame(x2 = runif(nn <- 1000))  # Artificial data
odata <- transform(odata, shape = loglink(-0.25 + x2, inverse = TRUE))
odata <- transform(odata, y1 = rotzeta(nn, shape))
with(odata, table(y1))
ofit <- vglm(y1 ~ x2, otzeta, data = odata, trace = TRUE, crit = "coef")
coef(ofit, matrix = TRUE)

## End(Not run)
```

---

pirates1 *Personal data of the executed pirates associated with Bartholomew Roberts*

Description

The age, names and habitation of 52 pirates who were found guilty of piracy and executed, after the ships associated with Bartholomew Roberts were captured.

Usage

```
data(pirates1)
```

Format

A data frame with the following 3 variables.

`age` a numeric vector, their age in years at the time of trial. Bartholomew Roberts himself was 39 years old at his death.

`name` character.

`habitation` character.

Details

According to Wiki, in February 1722 Captain Ogle was sent by the British Government to find and capture the notorious pirate Bartholomew Roberts (real name: John Roberts, but also known later as Black Bart). When his warship caught up with the *Royal Fortune* he attacked and Bartholomew Roberts was the first to fall, followed by 2 others. The remaining pirates surrendered soon afterwards. A total of 272 men were captured, and of these, 65 were black, and they were sold into

slavery. The remainder were taken to Cape Coast Castle, apart from those who died on the voyage back. The trial was held in April, 1722, and 54 were condemned to death, of whom 52 were hanged and two were reprieved. Of those executed, their personal data (name, age, habitation) were recorded.

Source

Pages 248–249 of Johnson, Captain Charles, (1955) (Editor: Arthur L. Hayward). *A General History of the Robberies and Murders of the Most Notorious Pirates*, London: Routledge and Kegan Paul Ltd. This edition was first published in 1926. The earliest manuscript of the book dates back to 1724.

This data was entered into R by Lucia Pilleri.

See Also

pirates2.

Examples

```
summary(pirates1)
```

`pirates2` *Personal data of the crew of the ship Ranger, associated with the pirate Edward Low*

Description

A data frame containing the age, name, birthplace and verdict of 35 members of a pirate ship associated with Edward Low, who were taken to trial on 10th to 12th July, 1723.

Usage

```
data(pirates2)
```

Format

The variables `age` and `name` are analagous to `pirates1`. The variable `guilty` is binary and 1 means yes, 0 means not guilty. Guilty crew members were executed except for two: John Brown and Patrick Cunningham; they were respited for one year and recommended to the King's favour.

Details

Starting on the 10th July, 1723, the crew of the *Ranger* were judged. The captain of the ship was Charles Harris, and this ship was one of two pirate ships under Captain Edward Low. Their personal data (name, age, place of birth) and verdicts are recorded in Johnson (1955). This data was constructed from pp.295–296 of that book and includes those who were not found guilty (and therefore were not executed). The execution of the 25 men were performed on 19 July near Newport, Rhode Island, USA. The notorious pirate Edward Low himself was brought to trial in 1724 under different circumstances and was hanged in Martinique.

Source

Same as pirates1. This data was entered into R by Lucia Pilleri.

See Also

pirates1.

Examples

```
summary(pirates2)
```

Posbinom *Positive-Binomial Distribution*

Description

Density, distribution function, quantile function and random generation for the positive-binomial distribution.

Usage

```
dposbinom(x, size, prob, log = FALSE)
pposbinom(q, size, prob)
qposbinom(p, size, prob)
rposbinom(n, size, prob)
```

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
n	number of observations. Fed into runif.
size	number of trials. It is the N symbol in the formula given in posbinomial and should be positive.
prob	probability of success on each trial. Should be in $(0, 1)$.
log	See dbinom.

Details

The positive-binomial distribution is a binomial distribution but with the probability of a zero being zero. The other probabilities are scaled to add to unity. The mean therefore is

$$\mu/(1-(1-\mu)^N)$$

where μ is the argument prob above. As μ increases, the positive-binomial and binomial distributions become more similar. Unlike similar functions for the binomial distribution, a zero value of prob is not permitted here.

Value

dposbinom gives the density, pposbinom gives the distribution function, qposbinom gives the quantile function, and rposbinom generates random deviates.

Note

These functions are or are likely to be deprecated. Use Gaitdbinom instead.

For dposbinom(), if arguments size or prob equal 0 then a NaN is returned.

The family function posbinomial estimates the parameters by maximum likelihood estimation.

Author(s)

T. W. Yee.

See Also

posbinomial, dposbern, Gaitdbinom, zabinomial, zibinomial, Binomial.

Examples

```
prob <- 0.2; size <- 10
table(y <- rposbinom(n = 1000, size, prob))
mean(y)  # Sample mean
size * prob / (1 - (1 - prob)^size)  # Population mean

(ii <- dposbinom(0:size, size, prob))
cumsum(ii) - pposbinom(0:size, size, prob)  # Should be 0s
table(rposbinom(100, size, prob))

table(qposbinom(runif(1000), size, prob))
round(dposbinom(1:10, size, prob) * 1000)  # Should be similar

## Not run:  barplot(rbind(dposbinom(x = 0:size, size, prob),
                           dbinom(x = 0:size, size, prob)),
        beside = TRUE, col = c("blue", "green"),
        main = paste("Positive-binomial(", size, ",",
                     prob, ") (blue) vs",
        " Binomial(", size, ",", prob, ") (green)", sep = ""),
        names.arg = as.character(0:size), las = 1)
## End(Not run)

# Simulated data example
nn <- 1000; sizeval1 <- 10; sizeval2 <- 20
pdata <- data.frame(x2 = seq(0, 1, length = nn))
pdata <- transform(pdata, prob1  = logitlink(-2 + 2 * x2, inverse = TRUE),
                          prob2  = logitlink(-1 + 1 * x2, inverse = TRUE),
                          sizev1 = rep(sizeval1, len = nn),
                          sizev2 = rep(sizeval2, len = nn))
pdata <- transform(pdata, y1 = rposbinom(nn, size = sizev1, prob = prob1),
                          y2 = rposbinom(nn, size = sizev2, prob = prob2))
with(pdata, table(y1))
```

```
with(pdata, table(y2))
# Multiple responses
fit2 <- vglm(cbind(y1, y2) ~ x2, posbinomial(multiple.responses = TRUE),
             trace  = TRUE, data = pdata, weight = cbind(sizev1, sizev2))
coef(fit2, matrix = TRUE)
```

`Posnegbin` *Positive-Negative Binomial Distribution*

Description

Density, distribution function, quantile function and random generation for the positive-negative binomial distribution.

Usage

```
dposnegbin(x, size, prob = NULL, munb = NULL, log = FALSE)
pposnegbin(q, size, prob = NULL, munb = NULL,
           lower.tail = TRUE, log.p = FALSE)
qposnegbin(p, size, prob = NULL, munb = NULL)
rposnegbin(n, size, prob = NULL, munb = NULL)
```

Arguments

`x, q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. Fed into `runif`.
`size, prob, munb, log`	Same arguments as that of an ordinary negative binomial distribution (see `dnbinom`). Some arguments have been renamed slightly. Short vectors are recycled. The parameter `1/size` is known as a dispersion parameter; as `size` approaches infinity, the negative binomial distribution approaches a Poisson distribution. Note that `prob` must lie in $(0, 1)$, otherwise a `NaN` is returned.
`log.p, lower.tail`	Same arguments as that of an ordinary negative binomial distribution (see `pnbinom`).

Details

The positive-negative binomial distribution is a negative binomial distribution but with the probability of a zero being zero. The other probabilities are scaled to add to unity. The mean therefore is

$$\mu/(1 - p(0))$$

where μ the mean of an ordinary negative binomial distribution.

Value

dposnegbin gives the density, pposnegbin gives the distribution function, qposnegbin gives the quantile function, and rposnegbin generates n random deviates.

Note

These functions are or are likely to be deprecated. Use Gaitdnbinom instead.

Author(s)

T. W. Yee

References

Welsh, A. H., Cunningham, R. B., Donnelly, C. F. and Lindenmayer, D. B. (1996). Modelling the abundances of rare species: statistical models for counts with extra zeros. *Ecological Modelling*, **88**, 297–308.

See Also

Gaitdnbinom, posnegbinomial, zanegbinomial, zinegbinomial, rnbinom.

Examples

```
munb <- 5; size <- 4; n <- 1000
table(y <- rposnegbin(n, munb = munb, size = size))
mean(y)  # sample mean
munb / (1 - (size / (size + munb))^size)  # population mean
munb / pnbinom(0, mu = munb, size = size, lower.tail = FALSE)  # same as before

x <- (-1):17
(ii <- dposnegbin(x, munb = munb, size = size))
max(abs(cumsum(ii) - pposnegbin(x, munb = munb, size = size)))  # Should be 0

## Not run:
x <- 0:10
barplot(rbind(dposnegbin(x, munb = munb, size = size),
                dnbinom(x, mu   = munb, size = size)),
        beside = TRUE, col = c("blue","green"),
        main = paste("dposnegbin(munb = ", munb, ", size = ", size, ") (blue) vs",
                     " dnbinom(mu = ", munb, ", size = ", size, ") (green)", sep = ""),
        names.arg = as.character(x))
## End(Not run)

# Another test for pposnegbin()
nn <- 5000
mytab <- cumsum(table(rposnegbin(nn, munb = munb, size = size))) / nn
myans <- pposnegbin(sort(as.numeric(names(mytab))), munb = munb, size = size)
max(abs(mytab - myans))  # Should be 0
```

Pospois *Positive-Poisson Distribution*

Description

Density, distribution function, quantile function and random generation for the positive-Poisson distribution.

Usage

```
dpospois(x, lambda, log = FALSE)
ppospois(q, lambda)
qpospois(p, lambda)
rpospois(n, lambda)
```

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
n	number of observations. Fed into runif.
lambda	vector of positive means (of an ordinary Poisson distribution). Short vectors are recycled.
log	logical.

Details

The positive-Poisson distribution is a Poisson distribution but with the probability of a zero being zero. The other probabilities are scaled to add to unity. The mean therefore is

$$\lambda/(1-\exp(-\lambda)).$$

As λ increases, the positive-Poisson and Poisson distributions become more similar. Unlike similar functions for the Poisson distribution, a zero value of lambda returns a NaN.

Value

dpospois gives the density, ppospois gives the distribution function, qpospois gives the quantile function, and rpospois generates random deviates.

Note

These functions are or are likely to be deprecated. Use Gaitdpois instead.

The family function pospoisson estimates λ by maximum likelihood estimation.

Author(s)

T. W. Yee

See Also

Gaitdpois, pospoisson, zapoisson, zipoisson, rpois.

Examples

```
lambda <- 2; y = rpospois(n = 1000, lambda)
table(y)
mean(y)  # Sample mean
lambda / (1 - exp(-lambda))  # Population mean

(ii <- dpospois(0:7, lambda))
cumsum(ii) - ppospois(0:7, lambda)  # Should be 0s
table(rpospois(100, lambda))

table(qpospois(runif(1000), lambda))
round(dpospois(1:10, lambda) * 1000)  # Should be similar

## Not run:  x <- 0:7
barplot(rbind(dpospois(x, lambda), dpois(x, lambda)),
        beside = TRUE, col = c("blue", "orange"),
        main = paste("Positive Poisson(", lambda, ") (blue) vs",
        " Poisson(", lambda, ") (orange)", sep = ""),
        names.arg = as.character(x), las = 1, lwd = 2)
## End(Not run)
```

---

prison.us *US Prison Data*

Description

Number of prisoners in each North American state, and the populations of those states from years 1977 to 2010

Usage

```
data(prison.us)
```

Format

A data frame with 34 observations on the following 103 variables.

Year a numeric vector, the year

AL.num, AL.pop numeric vectors

AK.num, AK.pop, AZ.num numeric vectors

AZ.pop, AR.num, AR.pop numeric vectors

CA.num, CA.pop, CO.num numeric vectors

CO.pop, CT.num, CT.pop numeric vectors

DE.num, DE.pop, FL.num numeric vectors

FL.pop, GA.num, GA.pop numeric vectors

HI.num, HI.pop, ID.num numeric vectors

ID.pop, IL.num, IL.pop numeric vectors

IN.num, IN.pop, IA.num numeric vectors

IA.pop, KS.num, KS.pop numeric vectors

KY.num, KY.pop, LA.num numeric vectors

LA.pop, ME.num, ME.pop numeric vectors

MD.num, MD.pop, MA.num numeric vectors

MA.pop, MI.num, MI.pop numeric vectors

MN.num, MN.pop, MS.num numeric vectors

MS.pop, MO.num, MO.pop numeric vectors

MT.num, MT.pop, NE.num numeric vectors

NE.pop, NV.num, NV.pop numeric vectors

NH.num, NH.pop, NJ.num numeric vectors

NJ.pop, NM.num, NM.pop numeric vectors

NY.num, NY.pop, NC.num numeric vectors

NC.pop, ND.num, ND.pop numeric vectors

OH.num, OH.pop, OK.num numeric vectors

OK.pop, OR.num, OR.pop numeric vectors

PA.num, PA.pop, RI.num numeric vectors

RI.pop, SC.num, SC.pop numeric vectors

SD.num, SD.pop, TN.num numeric vectors

TN.pop, TX.num, TX.pop numeric vectors

UT.num, UT.pop, VT.num numeric vectors

VT.pop, VA.num, VA.pop numeric vectors

WA.num, WA.pop, WV.num numeric vectors

WV.pop, WI.num, WI.pop numeric vectors

WY.num, WY.pop numeric vectors

US.pop, US.num numeric vectors, overall counts for the whole country

Details

This is a data set of the number of prisoners in each American state and the populations of those states, from 1977 to 2010. The number of prisoners are taken from December 31st, while the populations are estimates taken from July 1st based on the previous Census, except for pop.1980, which uses exact census data from 1980/04/01.

Warning: a scatterplot of `US.pop` shows a discontinuity around 2000.

Source

The prisoner data was compiled from: Bureau of Justice Statistics, http://www.bjs.gov/index.cfm. Downloaded in September 2013 and formatted into R by J. T. Gray, jamsgry@gmail.com.

The population data was compiled from: United States Census Bureau, http://www.census.gov/popest/data. Downloaded in September 2013 by J. T. Gray. This site may have become stale.

Examples

```
summary(prison.us)
## Not run:  # This plot shows a discontinuity around 2000.
plot(US.pop / 1e6 ~ Year, prison.us, main = "US population (millions)",
     las = 1, type = "b", col = "blue")
## End(Not run)
```

---

profs.nz *Professors of Statistics in New Zealand*

Description

This data set contains information on about 22 past or present professors of statistics in New Zealand universities.

Usage

```
data(profs.nz)
```

Format

A data frame with 22 observations on the following 7 variables.

`pubtotal` a numeric vector, the total number of publications.

`cites` a numeric vector, the number of citations.

`initials` character, first and middle and surname initials.

`Surname` character, the surname.

`firstyear` a numeric vector, the earliest indexed publication.

`ID` a numeric vector, the unique MR Author ID for each professor.

`pub1stAuthor` a numeric vector, the total number of publications which are first authored by the person.

`ARPtotal` a numeric vector, the total number of author/related publications.

`institution` character, with values `"MU"`, `"UA"`, `"UC"`, `"UO"`, `"UW"`, `"VU"`, the university affiliation. The abbreviations are for: Massey University, University of Auckland, University of Canterbury, University of Otago, University of Waikato and Victoria University Wellington.

Details

This data set contains information taken from the MathSciNet database on professors of statistics (and some related fields) affiliated with New Zealand universities.

In the future the following names may be added: C. F. Ansley, P. C. B. Phillips, B. S. Weir, C. S. Withers.

Source

The data is compiled from https://mathscinet.ams.org/mathscinet/ by J. T. Gray in April 2014.

Examples

```
profs.nz
profs.nz[order(with(profs.nz, pubtotal), decreasing = TRUE), ]
## Not run:
plot(pub1stAuthor / pubtotal ~ pubtotal,
     main = "Professors of Statistics in NZ",
     xlab = "Number of publications in MathSciNet",
     ylab = "Proportion of first-authored papers",
     data = profs.nz, col = "blue", las = 1, type = "n")
with(profs.nz, text(pubtotal, y = pub1stAuthor / pubtotal,
     labels = initials, col = "blue", las = 1))

## End(Not run)
```

---

rugby *Wins, Losses and Draws Between the Top 10 Rugby Teams*

Description

The number of wins, losses and draws for each of the top 10 rugby teams agains each other

Usage

```
data(rugby)
data(rugby.ties)
```

Format

The format is as two matrices.

Details

The first matrix is of the number of games won by each team against each of the other teams. The other matrix is the number of draws (ties) between each team. This data is current as of 2013-10-07.

Source

The match statistics are compiled from http://www.rugbydata.com/ on 2013-10-07 by J. T. Gray, jamsgry@gmail.com.

The top ten teams are determined by the International Rugby Board world rankings, https://www.world.rugby/.

Examples

```
data(rugby); data(rugby.ties)
rugby
rugby.ties
```

SardiniaHotels *Data from hotels in Sardinia, Italy*

Description

This data set contains information and satisfaction scores appearing on the TripAdvisor website between the years 2008 and 2016 regarding hotels in Sardinia, Italy.

The satisfaction data refer to the reputation of hotel located along Sardinian coasts, as expressed by clients with respect to different services (e.g., breakfast, restaurant, swimming pool) offered by the hotel.

Usage

```
data(SardiniaHotels)
```

Format

A data frame with 518 rows and 43 columns (variables). Each row refers to a single hotel.

The following variables are included in the dataset:

`municipality` a factor, the municipality where the hotel is located.

`stars` an ordered factor with levels:

- `1OR2stars` for 1 star or 2 star hotels,
- `3stars` 3 star hotels,
- `residence`,
- `4stars`, 4 star hotels,
- `5starsORresort`, 5 star hotels or resorts.

`area` a factor with levels related to the area of the Sardinian coast where each single hotel is located: `AlgheroSassari`, `CagliariVillasimius`, `CostaSmeralda`, `DorgaliOrosei`, `Gallura`, `NurraAnglona`, `Ogliastra`, `Olbia`, `OristanoBosa`, `PulaChia`, `Sarrabus`, `Sulcis`.

`seaLocation` a factor with levels `yes` (if the hotel is located close to the sea) and `no` (otherwise).

`excellent` a numeric vector, the number of people that expressed the highest level of satisfaction.

`good` a numeric vector, the number of people that expressed a good level of satisfaction.

`average` a numeric vector, the number of people that expressed an average level of satisfaction.

`bad` a numeric vector, the number of people that expressed a bad level of satisfaction.

`poor` a numeric vector, the number of people that expressed the lowest level of satisfaction.

`family` a numeric vector, the number of people travelling with family.

`couple` a numeric vector, the number of people travelling with their partner.

`single` a numeric vector, the number of people travelling alone.

`business` a numeric vector, the number of people travelling for work.

`MarMay` a numeric vector, the number of people travelling during the period March to May.

`JunAug` a numeric vector, the number of people travelling during the period June to August.

`SepNov` a numeric vector, the number of people travelling during the period September to November.

`DecFeb` a numeric vector, the number of people travelling during the period December to February.

`location` a numeric vector, the satisfaction score expressed by tourists towards the location.

`sleepQuality` a numeric vector, the satisfaction score expressed by tourists towards the sleep quality.

`room` a numeric vector, the satisfaction score expressed by tourists towards the comfort and quality of the room.

`services` a numeric vector, the satisfaction score expressed by tourists towards the quality of the services.

`priceQualityRate` a numeric vector, the satisfaction score expressed by tourists towards ratio between price and quality.

`cleaning` a numeric vector, the satisfaction score expressed by tourists towards level of room and hotel cleaning.

`bt1` a factor with levels `breakfast`, `cleaning`, `location`, `overall`, `price`, `restaurant`, `room`, `services`, `staff`, `structure` and `Wi-Fi`.

It expresses the 1st most used word in reviews for a hotel.

`ratebt1` a factor with levels `-1` (if the satisfaction score espressed in `bt1` is prevalently negative) and `1` (if the satisfaction score espressed in `bt1` is prevalently positive).

`bt2` a factor with levels `breakfast`, `cleaning`, `location`, `overall`, `price`, `restaurant`, `room`, `services`, `staff`, `structure` and `Wi-Fi`.

It expresses the 2nd most used word in reviews for a hotel.

`ratebt2` a factor with levels `-1` (if the satisfaction score espressed in `bt2` is prevalently negative) and `1` (if the satisfaction score espressed in `bt2` is prevalently positive).

`bt3` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt4` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt5` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt6` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt7` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt8` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt9` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`bt10` similar to `bt1` and `bt2`, but with a corresponding different ranking.

`ratebt3` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt4` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt5` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt6` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt7` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt8` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt9` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

`ratebt10` similar to `ratebt1` and `ratebt2`, but with a corresponding different ranking.

Details

These data were manually collected during March–June 2016 by students of the class of "Statistics for Turism" at the University of Cagliari, Italy (Bachelor's degree in Tourism Economics and Managment), under the supervision of Prof. Claudio Conversano and Dr. Giulia Contu.

Many of the variables fall into several natural groups, e.g., [`municipality`, `stars`, `area`, `seaLocation`]; [`excellent`, `good`, `average`, `bad`, `poor`]; [`MarMay`, `JunAug`, `SepNov`, `DecFeb`]; [`family`, `couple`, `single`, `business`]; [`location`,...`cleaning`]; [`bt1`,...`bt10`]; [`ratebt1`,...`ratebt10`].

Source

TripAdvisor, https://www.tripadvisor.it/.

Examples

```
data(SardiniaHotels)
summary(SardiniaHotels)
```

---

students.tw *Taiwanese students answer a multiple response question*

Description

This data is a subset from a survey of 49609 first-year college students in Taiwan collected in the year 2003 about their preferences for college study.

Usage

```
data(students.tw)
```

Format

A data frame with 49609 observations on the following 12 response variables. For binary variables, a `"1"` means `yes`, and `"0"` means `no`. See below for exact wording (translated from the original Chinese).

`ID` a numeric vector, a unique identification number for each student in the survey.

`read` Read Chinese and foreign classics.

`t.travel` Travel around Taiwan.

`conference` Present academic papers in conferences.

`act.leader` Lead large-scale activities.

`team` Be on a school team.

`stu.leader` Be a student association leader.

`intern` Participate internship programs.

`love` Fall in love.

`sex` Have sexual experience.

`o.travel` Travel around the world.

`friends` Make many friends.

`other` Other experience which is not included in the survey.

Details

This data frame is a subset of a larger data set where any student with any missing value was deleted. The remaining data set contains of 32792 students. Unfortunately, other variables such as age and sex were not made available.

Each student was asked the following multiple response question.

Question : What kind of experience do you expect to receive during the period of college study? (Select at least one response)

1. Read Chinese and foreign classics
2. Travel around Taiwan
3. Present academic papers in conferences
4. Lead large-scale activities
5. Be on a school team
6. Be a student association leader
7. Participate internship programs
8. Fall in love
9. Have sexual experience
10. Travel around the world
11. Make many friends
12. Other

Source

Originally, the data set for was downloaded from a survey center of Academia Sinica `https://srda.sinica.edu.tw/news`. It now seems unavailable.

References

Wang, H. and Huang, W. H. (2013) Bayesian Ranking Responses in Multiple Response Questions. *Journal of the Royal Statistical Society, Series A*, **(to appear)**.

Help from Viet Hoang Quoc is gratefully acknowledged.

Examples

```
data(students.tw)
summary(students.tw)

with(students.tw, table(love, sex))
## Not run:
plot(jitter(sex) ~ jitter(love), data = students.tw, col = "blue",
     main = "Taiwanese students")

## End(Not run)
```

Tikuv *A Short-tailed Symmetric Distribution*

Description

Density, cumulative distribution function, quantile function and random generation for the short-tailed symmetric distribution of Tiku and Vaughan (1999).

Usage

```
dtikuv(x, d, mean = 0, sigma = 1, log = FALSE)
ptikuv(q, d, mean = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qtikuv(p, d, mean = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE, ...)
rtikuv(n, d, mean = 0, sigma = 1, Smallno = 1.0e-6)
```

Arguments

`x, q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. Same as in `runif`.
`d, mean, sigma`	arguments for the parameters of the distribution. See `tikuv` for more details. For `rtikuv`, arguments `mean` and `sigma` must be of length 1.
`Smallno`	Numeric, a small value used by the rejection method for determining the lower and upper limits of the distribution. That is, `ptikuv(L)` < `Smallno` and `ptikuv(U)` > `1-Smallno` where `L` and `U` are the lower and upper limits respectively.

`...` Arguments that can be passed into `uniroot`.

`log` Logical. If `log = TRUE` then the logarithm of the density is returned.

`lower.tail, log.p`
Same meaning as in `pnorm` or `qnorm`.

Details

See `tikuv` for more details.

Value

`dtikuv` gives the density, `ptikuv` gives the cumulative distribution function, `qtikuv` gives the quantile function, and `rtikuv` generates random deviates.

Author(s)

T. W. Yee and Kai Huang

See Also

`tikuv`.

Examples

```
## Not run:  par(mfrow = c(2, 1))
x <- seq(-5, 5, len = 401)
plot(x, dnorm(x), type = "l", col = "black", ylab = "", las = 1,
     main = "Black is standard normal, others are dtikuv(x, d)")
lines(x, dtikuv(x, d = -10), col = "orange")
lines(x, dtikuv(x, d = -1 ), col = "blue")
lines(x, dtikuv(x, d =  1 ), col = "green")
legend("topleft", col = c("orange","blue","green"), lty = rep(1, len = 3),
       legend = paste("d =", c(-10, -1, 1)))

plot(x, pnorm(x), type = "l", col = "black", ylab = "", las = 1,
     main = "Black is standard normal, others are ptikuv(x, d)")
lines(x, ptikuv(x, d = -10), col = "orange")
lines(x, ptikuv(x, d = -1 ), col = "blue")
lines(x, ptikuv(x, d =  1 ), col = "green")
legend("topleft", col = c("orange","blue","green"), lty = rep(1, len = 3),
       legend = paste("d =", c(-10, -1, 1)))
## End(Not run)

probs <- seq(0.1, 0.9, by = 0.1)
ptikuv(qtikuv(p = probs, d =  1), d = 1) - probs  # Should be all 0
```

tikuv *Short-tailed Symmetric Distribution Family Function*

Description

Fits the short-tailed symmetric distribution of Tiku and Vaughan (1999).

Usage

```
tikuv(d, lmean = "identitylink", lsigma = "loglink", isigma = NULL,
      zero = "sigma")
```

Arguments

d	The d parameter. It must be a single numeric value less than 2. Then $h = 2 - d > 0$ is another parameter.
lmean, lsigma	Link functions for the mean and standard deviation parameters of the usual univariate normal distribution (see **Details** below). They are μ and σ respectively. See Links for more choices.
isigma	Optional initial value for σ. A NULL means a value is computed internally.
zero	A vector specifying which linear/additive predictors are modelled as intercept-only. The values can be from the set {1,2}, corresponding respectively to μ, σ. If zero = NULL then all linear/additive predictors are modelled as a linear combination of the explanatory variables. For many data sets having zero = 2 is a good idea. See CommonVGAMffArguments for information.

Details

The short-tailed symmetric distribution of Tiku and Vaughan (1999) has a probability density function that can be written

$$f(y) = \frac{K}{\sqrt{2\pi}\sigma}\left[1 + \frac{1}{2h}\left(\frac{y-\mu}{\sigma}\right)^2\right]^2 \exp\left(-\frac{1}{2}(y-\mu)^2/\sigma^2\right)$$

where $h = 2 - d > 0$, K is a function of h, $-\infty < y < \infty$, $\sigma > 0$. The mean of Y is $E(Y) = \mu$ and this is returned as the fitted values.

Value

An object of class "vglmff" (see vglmff-class). The object is used by modelling functions such as vglm, and vgam.

Warning

Under- or over-flow may occur if the data is ill-conditioned, e.g., when d is very close to 2 or approaches -Inf.

Note

The density function is the product of a univariate normal density and a polynomial in the response y. The distribution is bimodal if $d > 0$, else is unimodal. A normal distribution arises as the limit as d approaches $-\infty$, i.e., as h approaches ∞. Fisher scoring is implemented. After fitting the value of d is stored in `@misc` with component name d.

Author(s)

Thomas W. Yee

References

Akkaya, A. D. and Tiku, M. L. (2008). Short-tailed distributions and inliers. *Test*, **17**, 282–296.

Tiku, M. L. and Vaughan, D. C. (1999). A family of short-tailed symmetric distributions. *Technical report, McMaster University, Canada*.

See Also

dtikuv, uninormal.

Examples

```
m <- 1.0; sigma <- exp(0.5)
tdata <- data.frame(y = rtikuv(n = 1000, d = 1, m = m, s = sigma))
tdata <- transform(tdata, sy = sort(y))
fit <- vglm(y ~ 1, tikuv(d = 1), data = tdata, trace = TRUE)
coef(fit, matrix = TRUE)
(Cfit <- Coef(fit))
with(tdata, mean(y))
## Not run:  with(tdata, hist(y, prob = TRUE))
lines(dtikuv(sy, d = 1, m = Cfit[1], s = Cfit[2]) ~ sy, data = tdata, col = "orange")
## End(Not run)
```

---

trapO *Trout Data at the Te Whaiau Trap on Lake Otamangakau*

Description

Rainbow and brown trout trapped at the Te Whaiau Trap at Lake Otamangakau in the central North Island of New Zealand. The data were collected by the Department of Conservation.

Usage

```
data(trapO)
```

Format

A data frame with 1226 observations on the following 15 variables.

Date Date as a class `"Date"` variable.

BFTW, BMTW, RFTW, RMTW numeric vectors, the number of fish trapped daily. `B/R` is for brown/rainbow trout. `F/M` is for female/male. `TW` is for the Te Whaiau trap location (there was another trap just off the Tongariro River).

MinAT, MaxAT numeric vectors, daily minimum and maximum ambient temperatures in Celsius.

Rain numeric vector, daily rainfall that has been scaled between 0 (none) and 100 (flooding situation).

LevelTW numeric vector, water level of the stream that has been scaled between 0 (none) and 100 (flooding situation). In a flooding situation it is possible that some fish going upstream were not caught.

Year, Month, Day numeric vectors, extracted from `Date`.

doy a numeric vector, Julian day of year. The value 1 means 1st of January, and so on up to 365.

f.Year a factor vector, the year as a factor.

fict.Year similar to `Date` but a fictitional year is used for all the data. This allows all the data to be plotted along one calendar year.

Details

These are the daily numbers of fish trapped at the Te Whaiau trap near Lake Otamangakau, during the winter months when spawning is at its peak. These fish were all going upstream. There are two species of trout, split up by males and females, in the data set. The first is brown trout (*Salmo trutta*) and the second is rainbow trout (*Oncorhynchus mykiss*). Information on the movement patterns of brown and rainbow trout in Lake Otamangakau and Lake Te Whaiau can be found in Dedual et al. (2000).

Brown trout are more sedentary compared with rainbow trout, and spawning activities of brown trout occur between May and June whilst peak spawning of rainbow trout occurs between July and August. Furthermore, brown trout have been observed avoiding water above 19 degrees Celsius and optimum temperatures for growth are between 10–15 degrees for brown trout and 16.5–17.2 degrees for rainbow trout.

See also `lake0`.

Source

Many thanks to Dr Michel Dedual (`http://www.doc.govt.nz`) for making this data available. Help from Simeon Pattenwise is also acknowledged.

References

Dedual, M. and Maxwell, I. D. and Hayes, J. W. and Strickland, R. R. (2000). Distribution and movements of brown (*Salmo trutta*) and rainbow trout (*Oncorhynchus mykiss*) in Lake Otamangakau, central North Island, New Zealand. *New Zealand Journal of Marine and Freshwater Research*, **34**: 615–627.

Examples

```
data("trap0")
summary(trap0)
```

`tube10` *London Underground (Tube) Passenger Counts for November 2010*

Description

The data set contains counts of the number of passengers entering London Underground stations (also known as *the Tube*) on a typical day of November 2010 in quarter-hour time blocks.

Usage

```
data("tube10")
```

Format

A data frame with 100 observations on the following 270 variables.

`ActonTown` a numeric vector

`Aldgate` a numeric vector

`AldgateEast` a numeric vector

`Alperton` a numeric vector

`Amersham` a numeric vector

`Angel` a numeric vector

`Archway` a numeric vector

`ArnosGrove` a numeric vector

`Arsenal` a numeric vector

`BakerStreet` a numeric vector

`Balham` a numeric vector

`BankAndMonument` a numeric vector

`Barbican` a numeric vector

`Barking` a numeric vector

`Barkingside` a numeric vector

`BaronsCourt` a numeric vector

`Bayswater` a numeric vector

`Becontree` a numeric vector

`BelsizePark` a numeric vector

`Bermondsey` a numeric vector

`BethnalGreen` a numeric vector

Blackfriars a numeric vector

BlackhorseRoad a numeric vector

BondStreet a numeric vector

Borough a numeric vector

BostonManor a numeric vector

BoundsGreen a numeric vector

BowRoad a numeric vector

BrentCross a numeric vector

Brixton a numeric vector

BromleyByBow a numeric vector

BuckhurstHill a numeric vector

BurntOak a numeric vector

CaledonianRoad a numeric vector

CamdenTown a numeric vector

CanadaWater a numeric vector

CanaryWharf a numeric vector

CanningTown a numeric vector

CannonStreet a numeric vector

CanonsPark a numeric vector

ChalfontAndLatimer a numeric vector

ChalkFarm a numeric vector

ChanceryLane a numeric vector

CharingCross a numeric vector

Chesham a numeric vector

Chigwell a numeric vector

ChiswickPark a numeric vector

Chorleywood a numeric vector

ClaphamCommon a numeric vector

ClaphamNorth a numeric vector

ClaphamSouth a numeric vector

Cockfosters a numeric vector

Colindale a numeric vector

ColliersWood a numeric vector

CoventGarden a numeric vector

Croxley a numeric vector

DagenhamEast a numeric vector

DagenhamHeathway a numeric vector

Debden a numeric vector

DollisHill a numeric vector

EalingBroadway a numeric vector

EalingCommon a numeric vector

EarlsCourt a numeric vector

EastActon a numeric vector

EastFinchley a numeric vector

EastHam a numeric vector

EastPutney a numeric vector

Eastcote a numeric vector

Edgware a numeric vector

EdgwareRoadBak a numeric vector

EdgwareRoadCir a numeric vector

ElephantAndCastle a numeric vector

ElmPark a numeric vector

Embankment a numeric vector

Epping a numeric vector

Euston a numeric vector

EustonSquare a numeric vector

Fairlop a numeric vector

Farringdon a numeric vector

FinchleyCentral a numeric vector

FinchleyRoad a numeric vector

FinsburyPark a numeric vector

FulhamBroadway a numeric vector

GantsHill a numeric vector

GloucesterRoad a numeric vector

GoldersGreen a numeric vector

GoldhawkRoad a numeric vector

GoodgeStreet a numeric vector

GrangeHill a numeric vector

GreatPortlandStreet a numeric vector

GreenPark a numeric vector

Greenford a numeric vector

Gunnersbury a numeric vector

Hainault a numeric vector

HammersmithDis a numeric vector

HammersmithHC a numeric vector

Hampstead a numeric vector

HangerLane a numeric vector

Harlesden a numeric vector

HarrowAndWealdstone a numeric vector

HarrowOnTheHill a numeric vector

HattonCross a numeric vector

HeathrowTerminals123 a numeric vector

HeathrowTerminal4 a numeric vector

HeathrowTerminal5 a numeric vector

HendonCentral a numeric vector

HighBarnet a numeric vector

HighStreetKensington a numeric vector

HighburyAndIslington a numeric vector

Highgate a numeric vector

Hillingdon a numeric vector

Holborn a numeric vector

HollandPark a numeric vector

HollowayRoad a numeric vector

Hornchurch a numeric vector

HounslowCentral a numeric vector

HounslowEast a numeric vector

HounslowWest a numeric vector

HydeParkCorner a numeric vector

Ickenham a numeric vector

Kennington a numeric vector

KensalGreen a numeric vector

KensingtonOlympia a numeric vector

KentishTown a numeric vector

Kenton a numeric vector

KewGardens a numeric vector

Kilburn a numeric vector

KilburnPark a numeric vector

KingsCrossStPancras a numeric vector

Kingsbury a numeric vector

Knightsbridge a numeric vector

LadbrokeGrove a numeric vector

LambethNorth a numeric vector

LancasterGate a numeric vector

LatimerRoad a numeric vector

LeicesterSquare a numeric vector

Leyton a numeric vector

Leytonstone a numeric vector

LiverpoolStreet a numeric vector

LondonBridge a numeric vector

Loughton a numeric vector

MaidaVale a numeric vector

ManorHouse a numeric vector

MansionHouse a numeric vector

MarbleArch a numeric vector

Marylebone a numeric vector

MileEnd a numeric vector

MillHillEast a numeric vector

MoorPark a numeric vector

Moorgate a numeric vector

Morden a numeric vector

MorningtonCrescent a numeric vector

Neasden a numeric vector

NewburyPark a numeric vector

NorthActon a numeric vector

NorthEaling a numeric vector

NorthGreenwich a numeric vector

NorthHarrow a numeric vector

NorthWembley a numeric vector

Northfields a numeric vector

Northolt a numeric vector

NorthwickPark a numeric vector

Northwood a numeric vector

NorthwoodHills a numeric vector

NottingHillGate a numeric vector

Oakwood a numeric vector

OldStreet a numeric vector

Osterley a numeric vector

Oval a numeric vector

OxfordCircus a numeric vector

Paddington a numeric vector

ParkRoyal a numeric vector

ParsonsGreen a numeric vector

Perivale a numeric vector

PiccadillyCircus a numeric vector

Pimlico a numeric vector

Pinner a numeric vector

Plaistow a numeric vector

PrestonRoad a numeric vector

PutneyBridge a numeric vector

QueensPark a numeric vector

Queensbury a numeric vector

Queensway a numeric vector

RavenscourtPark a numeric vector

RaynersLane a numeric vector

Redbridge a numeric vector

RegentsPark a numeric vector

Richmond a numeric vector

Rickmansworth a numeric vector

RodingValley a numeric vector

RoyalOak a numeric vector

Ruislip a numeric vector

RuislipGardens a numeric vector

RuislipManor a numeric vector

RussellSquare a numeric vector

SevenSisters a numeric vector

ShepherdsBushCen a numeric vector

ShepherdsBushHC a numeric vector

SloaneSquare a numeric vector

Snaresbrook a numeric vector

SouthEaling a numeric vector

SouthHarrow a numeric vector

SouthKensington a numeric vector

SouthKenton a numeric vector

SouthRuislip a numeric vector

SouthWimbledon a numeric vector

SouthWoodford a numeric vector

Southfields a numeric vector

Southgate a numeric vector

Southwark a numeric vector

StJamessPark a numeric vector

StJohnsWood a numeric vector

StPauls a numeric vector

StamfordBrook a numeric vector

Stanmore a numeric vector

StepneyGreen a numeric vector

Stockwell a numeric vector

StonebridgePark a numeric vector

Stratford a numeric vector

SudburyHill a numeric vector

SudburyTown a numeric vector

SwissCottage a numeric vector

Temple a numeric vector

TheydonBois a numeric vector

TootingBec a numeric vector

TootingBroadway a numeric vector

TottenhamCourtRoad a numeric vector

TottenhamHale a numeric vector

TotteridgeAndWhetstone a numeric vector

TowerHill a numeric vector

TufnellPark a numeric vector

TurnhamGreen a numeric vector

TurnpikeLane a numeric vector

Upminster a numeric vector

UpminsterBridge a numeric vector

Upney a numeric vector

UptonPark a numeric vector

Uxbridge a numeric vector

Vauxhall a numeric vector

Victoria a numeric vector

WalthamstowCentral a numeric vector

Wanstead a numeric vector

WarrenStreet a numeric vector

`WarwickAvenue` a numeric vector

`Waterloo` a numeric vector

`Watford` a numeric vector

`WembleyCentral` a numeric vector

`WembleyPark` a numeric vector

`WestActon` a numeric vector

`WestBrompton` a numeric vector

`WestFinchley` a numeric vector

`WestHam` a numeric vector

`WestHampstead` a numeric vector

`WestHarrow` a numeric vector

`WestKensington` a numeric vector

`WestRuislip` a numeric vector

`WestbournePark` a numeric vector

`Westminster` a numeric vector

`WhiteCity` a numeric vector

`Whitechapel` a numeric vector

`WillesdenGreen` a numeric vector

`WillesdenJunction` a numeric vector

`Wimbledon` a numeric vector

`WimbledonPark` a numeric vector

`WoodGreen` a numeric vector

`WoodLane` a numeric vector

`Woodford` a numeric vector

`WoodsidePark` a numeric vector

`Total` a numeric vector; the total over all stations.

`mins24` a numeric vector; minutes on a 24 hour clock, e.g., 0 is midnight, 120 is 2am.

Details

Each cell contains the number of passengers entering a station during a quarter-hour period of time on a typical day during November 2010. The column names of the data frame are the station names and the most of the rows are the start time of each quarter-hour time block given in 24 hour time, e.g., 2215 means 10:15pm to 10:29pm. The last four rows are `"Total"`, `"A.M.Peak"`, `"Interpeak"`, `"P.M.Peak"`.

The data is adjusted to remove the effect of abnormal circumstances that many affect passeger numbers such as industrial action.

Source

The data came from the UK Government Transport for London website https://www.tfl.gov.uk. Downloaded in December 2013 and formatted into R by J. T. Gray (and slightly edited by T. W. Yee).

Examples

```
## Not run:
data(tube10)
waterloo <- tube10[1:(4*24), "Waterloo"]  # Omit the totals and the peaks
barplot(unlist(waterloo))
barplot(log10(1 + unlist(waterloo)), col = "lightblue",
        ylab = "log10(1+.)", las = 1)

## End(Not run)
```

---

ugss *Undergraduate Statistics Students Lifestyle Questionnaire*

Description

About 800 students studying undergraduate statistics were asked many lifestyle questions.

Usage

```
data(ugss)
```

Format

A data frame with 804 observations on the following 29 variables.

`sex` Gender, a factor, (female or male)

`age` age in years, a numeric vector

`eyes` eye colour, a factor, (blue, brown, green, hazel or other)

`piercings` Number of body piercings, a numeric vector

`pierced` Any body piercings? a factor, (Yes or No)

`tattoos` Number of tattoos, a numeric vector

`tattooed` Any tattoos? a factor, (Yes or No)

`glasses` Wears glasses etc.? a factor, (Yes or No)

`sleep` Average number of hours of sleep per night, a numeric vector

`study` Average number of hours of study per week, a numeric vector

`tv` Average number of hours watching TV per week, a numeric vector

`movies` Number of movies seen at a cinema during the last 3 months, a numeric vector

`movies3m` Seen movies in last 3 months? a factor, (Yes or No)

`sport` Favourite sport, a factor, about 19 of them

`entertainment` Favourite entertainment, a factor, about 15 of them

`fruit` Favourite fruit a factor, about 13 of them

`income` Average income during semester per week, a numeric vector

`rent` Amount spent on rent or room and board per week, a numeric vector

`clothes` Average amount spent on clothes per month, a numeric vector

`hair` Average cost to get a hair-cut, a numeric vector

`tobacco` Average amount spent on tobacco per week, a numeric vector

`smokes` Smokes? a factor, (Yes or No)

`alcohol` Average amount spent on alcohol per week, a numeric vector

`buy.alcohol` Buys (purchases) alcohol? a factor, (Yes or No)

`sendtxt` Average number text messages sent per day, a numeric vector.

`receivetxt` Average number text messages received per day, a numeric vector.

`txts` Uses text messaging? a factor, (Yes or No)

`country` Country of birth, a factor, about 54 of them

`status` Student status, a factor, (International, NZ.Citizen, NZ.Resident)

Details

This data was collected online and anonymously in 2010. The respondents were students studying an undergraduate statistics course at a New Zealand university. Possibly there are duplicate students (due to failing and re-enrolling). All monies are in NZD. Note the data has had minimal checking. Most numerical variables tend to have measurement error, and all of them happen to be all integer-valued.

Examples

```
summary(ugss)
```

---

vtinpat *Vermont Hospital Inpatient Data*

Description

Information on inpatients discharged from hospitals in Vermont, USA, 2012.

Usage

```
data(vtinpat)
```

Format

A data frame with 52206 observations on the following 7 variables.

`hospital` a factor with levels `1` = Northwestern Medical Center, `2` = North Country Hospital and Health Center, `3` = Northeastern Vermont Regional Hospital, `4` = Copley Hospital, `5` = Fletcher Allen Health Care, `6` = Central Vermont Hospital, `8` = Rutland Regional Medical Center, `9` = Porter Medical Center, `10` = Gifford Memorial Hospital, `11` = Mount Ascutney Hospital and Health Center, `12` = Springfield Hospital, `14` = Grace Cottage Hospital, `15` = Brattleboro Memorial Hospital, `16` = Southwestern Vermont Medical Center

`admit` a factor with levels `1` = Emergency, `2` = Urgent, `3` = Elective, `4`, Newborn, `5` = Trauma

`age.group` a factor with levels `1` = Under 1, `2` = 1-17, `3` = 18-24, `4` = 25-29, `5` = 30-34, `6` = 35-39, `7` = 40-44, `8` = 45-49, `9` = 50-54, `10` = 55-59, `11` = 60-64, `12` = 65-69, `13` = 70-74, `14` = 75+

`sex` a factor with levels `1` = Male, `2` = Female

`discharge` a factor with levels `1` = To another medical facility, `2` = home, `3` = against medical advice, `4` = Died, `5` = To court or law enforcement, `6` = still a patient

`diagnosis` a factor with levels `1` = Brain And C.N.S., `2` = Eye, `3` = Ear, Nose & Throat, `4` = Respiratory, `5` = Heart & Circulatory, `6` = Digestive, `7` = Liver & Pancreas, `8` = Musculoskeletal, `9` = Skin and Breast, `10` = Endocrine, `11` = Kidney & Urinary, `12` = Male Reproductive, `13` = Female Reproductive, `14` = Pregnancy, Childbirth, `15` = Neonatal, `16` = Spleen & Blood, `17` = Lymphatic, `18` = Infection, `19` = Mental Illness, `20` = Substance Abuse, `21` = Injury, Toxic Effects, `22` = Burns, `23` = Other, `24` = Trauma, `25` = H.I.V.

`los` a numeric vector, number of days spent in hospital

Details

This data set contains details on inpatients discharged from hospitals in Vermont, USA, in 2012 as part of the Vermont Uniform Hospital Discharge Data Set. The Vermont Department of Financial Regulation administers this program and the Vermont Department of Health manages the data set.

Source

Vermont department of Health, https://www.healthvermont.gov/stats/systems formatted into R by J. T. Gray in mid-2014.

Examples

```
summary(vtinpat)
```

---

wffc *2008 World Fly Fishing Championships Data*

Description

Capture records of the 2008 FIPS-MOUCHE World Fly Fishing Championships held in Rotorua, New Zealand during 22–30 March 2008.

Usage

```
data(wffc)
```

Format

A data frame with 4267 observations on the following 8 variables. Each row is a recorded capture.

`length` a numeric vector; length of fish in mm.

`water` a factor with levels `Waihou`, `Waimakariri`, `Whanganui`, `Otamangakau`, `Rotoaira`. These are known as Sectors IV, V, I, II, III respectively, and are also represented by the variable `sector`.

`session` a numeric vector; a value from the set 1,2,...,6. These are time ordered, and there were two sessions per competition day.

`sector` a numeric vector; a value from the set 1,2,...,5. Ideally these should be converted to a factor.

`beatboat` a numeric vector; beat or boat number, a value from the set 1,2,...,19. Ideally these should be converted to a factor. For a river though, they are contiguous whereas on a lake it is less so.

`comid` a numeric vector; the competitor's ID number. Uniquely identifies each competitor. These ID numbers actually correspond to their rankings in the individual level.

`iname` a character vector; the individual competitor's name.

`country` a character vector; what country the competitors represented. The countries represented were Australia (AUS), Canada (CAN), Croatia (CRO), Czech Republic (CZE), England (ENG), Finland (FIN), France (FRA), Holland (NED), Ireland (IRE), Italy (ITA), Japan (JPN), Malta (MAL), New Zealand (NZL), Poland (POL), Portugal (POR), South Africa (RSA), Slovakia (SVK), USA (USA), Wales (WAL).

Details

Details may be obtained at Yee (2010) and Yee (2014). Here is a brief summary. The three competition days were 28–30 March. Each session was fixed at 9.00am–12.00pm and 2.30–5.30pm daily. One of the sessions was a rest session. Each of 19 teams had 5 members, called A, B, C, D and E (there was a composite team, actually). The scoring system allocated 100 points to each eligible fish (minimum length was 18 cm) and 20 points for each cm of its length (rounded up to the nearest centimeter). Thus a 181mm or 190mm fish was worth 480 points. Each river was divided into 19 contiguous downstream beats labelled 1,2,...,19. Each lake was fished by 9 boats, each with two competitors except for one boat which only had one. Each competitor was randomly assigned to a beat/boat.

Competitors were ranked according to their placings at each sector-session combination, and then these placings were summed. Those with the minimum total placings were the winners, thus it was not necessarily those who had the maximum points who won. For example, in Session 1 at the Waihou River, each of the 19 competitors was ranked 1 (best) to 19 (worst) according to the point system. This is the "placing" for that session. These placings were added up over the 5 sessions to give the "total placings".

All sectors have naturally wild Rainbow trout (*Oncorhynchus mykiss*) while Lake Otamangakau and the Whanganui River also holds Brown trout (*Salmo trutta*). Only these two species were targetted. The species was not recorded electronically, however a post-analysis of the paper score sheets from the two lakes showed that, approximately, less than 5 percent were Brown trout. It may be safely assumed that all the Waihou and Waimakariri fish were Rainbow trout. The gender of the fish were also not recorded electronically, and anyway, distinguishing between male and female was very difficult for small fish.

Although species and gender data were supposed to have been collected at the time of capture the quality of these variables is rather poor and furthermore they were not recorded electronically.

Note that some fish may have been caught more than once, hence these data do not represent individual fish but rather recorded captures.

Note also that a few internal discrepancies may be found within and between the data frames wffc, wffc.nc, wffc.indiv, wffc.teams. This is due to various reasons, such as competitors being replaced by reserves when sick, fish that were included or excluded upon the local judge's decision, competitors who fished two hours instead of three by mistake, etc. The data has already been cleaned of errors and internal inconsistencies but a few may remain.

Source

This data frame was adapted from the WFFC's spreadsheet. Special thanks goes to Paul Dewar, Jill Mandeno, Ilkka Pirinen, and the other members of the Organising Committee of the 28th FIPS-Mouche World Fly Fishing Championships for access to the data. The assistance and feedback of Colin Shepherd is gratefully acknowledged.

References

Yee, T. W. (2010). VGLMs and VGAMs: an overview for applications in fisheries research. *Fisheries Research*, **101**, 116–126.

Yee, T. W. (2014). Scoring rules, and the role of chance: analysis of the 2008 World Fly Fishing Championships. *Journal of Quantitative Analysis in Sports*. **10**, 397–409.

See Also

wffc.indiv, wffc.teams, wffc.nc, wffc.P1, lake0.

Examples

```
summary(wffc)
with(wffc, table(water, session))

# Obtain some simple plots
waihou <- subset(wffc, water == "Waihou")
waimak <- subset(wffc, water == "Waimakariri")
whang  <- subset(wffc, water == "Whanganui")
otam   <- subset(wffc, water == "Otamangakau")
roto   <- subset(wffc, water == "Rotoaira")
minlength <- min(wffc[, "length"])
maxlength <- max(wffc[, "length"])
nwater <- c("Waihou"    = nrow(waihou), "Waimakariri" = nrow(waimak),
            "Whanganui" = nrow(whang),  "Otamangakau" = nrow(otam),
            "Rotoaira"  = nrow(roto))
## Not run:
par(mfrow = c(2, 3), las = 1)
# Overall distribution of length
with(wffc, boxplot(length/10 ~ water, ylim = c(minlength, maxlength)/10,
                   border = "blue", main = "Length (cm)", cex.axis = 0.5))
```

```
# Overall distribution of LOG length
with(wffc, boxplot(length/10 ~ water, ylim = c(minlength, maxlength)/10,
                   border = "blue", log = "y", cex.axis = 0.5,
                   main = "Length (cm) on a log scale"))

# Overall distribution of number of captures
pie(nwater, border = "blue", main = "Proportion of captures",
    labels = names(nwater), density = 10, col = 1:length(nwater),
    angle = 85+30* 1:length(nwater))

# Overall distribution of number of captures
with(wffc, barplot(nwater, main = "Number of captures", cex.names = 0.5,
                   col = "lightblue"))

# Overall distribution of proportion of number of captures
with(wffc, barplot(nwater / sum(nwater), cex.names = 0.5, col = "lightblue",
                   main = "Proportion of captures"))
# An interesting lake
with(roto, hist(length/10, xlab = "Fish length (cm)", col = "lightblue",
                breaks = seq(18, 70, by = 3), prob = TRUE, ylim = c(0, 0.08),
                border = "blue", ylab = "", main = "Lake Rotoaira", lwd = 2))

## End(Not run)
```

`wffc.indiv` *2008 World Fly Fishing Championships (Individual results) Data*

Description

Individual competitors' results of the 2008 FIPS-MOUCHE World Fly Fishing Championships held in Rotorua, New Zealand during 22–30 March 2008.

Usage

```
data(wffc.indiv)
```

Format

A data frame with 99 observations on the following 8 variables. Some of these variable are described in `wffc`.

`totalPlacings` a numeric vector; these are the summed placings over the 5 sessions.

`points` a numeric vector.

`noofcaptures` a numeric vector.

`longest` a numeric vector.

`individual` a numeric vector; did the competitor fish in a team or as an individual? (one team was made of composite countries due to low numbers).

`country` a character vector.

`iname` a character vector.

`comid` a numeric vector.

Details

This data frame gives the individual results of the competition. See also wffc and wffc.teams for more details and links.

References

Yee, T. W. (2010). VGLMs and VGAMs: an overview for applications in fisheries research. *Fisheries Research*, **101**, 116–126.

Examples

```
summary(wffc.indiv)
```

---

wffc.nc *2008 World Fly Fishing Championships (Number of captures) Data*

Description

Number of captures in the 2008 FIPS-MOUCHE World Fly Fishing Championships held in Rotorua, New Zealand during 22–30 March 2008.

Usage

```
data(wffc.nc)
```

Format

A data frame with 475 observations on the following 7 variables. Most of these variable are described in wffc. Each row is sorted by sector, session and beat.

`sector` a numeric vector.

`session` a numeric vector.

`beatboat` a numeric vector.

`numbers` a numeric vector.

`comid` a numeric vector.

`iname` a character vector.

`country` a character vector.

Details

This data frame was obtained by processing wffc. The key variable is `numbers`, which is sector-session-beat specific.

Note that some fish may have been caught more than once, hence these data do not represent individual fish.

References

Yee, T. W. (2010). VGLMs and VGAMs: an overview for applications in fisheries research. *Fisheries Research*, **101**, 116–126.

See Also

DeLury, lake0.

Examples

```
xtabs( ~ sector + session, wffc.nc)
```

`wffc.points` *Point System for the 2008 World Fly Fishing Championships*

Description

Point system for the 2008 World Fly Fishing Championships: current and some proposals.

Usage

```
wffc.P1(length, c1 = 100, min.eligible = 0.18, ppm = 2000)
wffc.P2(length, c1 = 100, min.eligible = 0.18, ppm = 2000,
        c.quad = 12700)
wffc.P3(length, c1 = 100, min.eligible = 0.18, ppm = 2000)
wffc.P1star(length, c1 = 100, min.eligible = 0.18, ppm = 2000)
wffc.P2star(length, c1 = 100, min.eligible = 0.18, ppm = 2000,
            c.quad = 12700)
wffc.P3star(length, c1 = 100, min.eligible = 0.18, ppm = 2000)
```

Arguments

`length`	Length of the fish, in meters. Numeric vector.
`c1`	Points added to each eligible fish.
`min.eligible`	The 2008 WFFC regulations stipulated that the smallest eligible fish was 0.180 m, which is 180 mm.
`ppm`	Points per meter of length of the fish.
`c.quad`	Constants for the quadratic terms. The defaults mean that a fish twice the minimum legal size will award about 50 percent more points compared to `wffc.P1()` and `wffc.P1star()`. See below for examples.

Details

The official website contains a document with the official rules and regulations of the competition. The function `wffc.P1()` implements the current WFFC point system, and is 'discrete' in that fish lengths are rounded up to the nearest centimeter (provided it is greater or equal to `min.eligible` m). `wffc.P1star()` is a continuous version of it in that it is piecewise linear with two pieces and it is discontinuous at `min.eligible`.

The function `wffc.P2()` is a new proposal which rewards catching bigger fish. It is based on a quadratic polynomial. `wffc.P2star()` is a continuous version of it.

The function `wffc.P3()` is another new proposal which rewards catching bigger fish. Named a *cumulative linear proposal*, its slope is `ppm` between `min.eligible` and `2 * min.eligible`, its slope is `2 * ppm` between `2 * min.eligible` and `3 * min.eligible`, its slope is `3 * ppm` between `3 * min.eligible` and `4 * min.eligible`, etc. One adds the usual `c1` to each eligible fish. `wffc.P3star()` is a continuous version of `wffc.P3()`.

The function `wffc.P4()` is another new proposal which rewards catching bigger fish. Named a *cumulative linear proposal*, its slope is `ppm` between `min.eligible` and `2 * min.eligible`, its slope is `2 * ppm` between `2 * min.eligible` and `1.5 * min.eligible`, its slope is `3 * ppm` between `1.5 * min.eligible` and `2 * min.eligible`, etc. One adds the usual `c1` to each eligible fish. `wffc.P4star()` is a continuous version of `wffc.P4()`.

Value

A vector with the number of points.

Note

`wffc.P2` and `wffc.P2star` may change in the future, as well as possibly `wffc.P3` and `wffc.P3star` and `wffc.P4` and `wffc.P4star`.

Author(s)

T. W. Yee.

References

Yee, T. W. (2014). Scoring rules, and the role of chance: analysis of the 2008 World Fly Fishing Championships. *Journal of Quantitative Analysis in Sports*. **10**, 397–409.

See Also

`wffc`.

Examples

```
## Not run:  fishlength <- seq(0.0, 0.72, by = 0.001)
plot(fishlength, wffc.P2star(fishlength), type = "l", col = "blue",
     las = 1, lty = "dashed", lwd = 2, las = 1, cex.main = 0.8,
     xlab = "Fish length (m)", ylab = "Competition points",
     main = "Current (red) and proposed (blue and green) WFFC point system")
lines(fishlength, wffc.P1star(fishlength), type = "l", col = "red", lwd = 2)
```

```
lines(fishlength, wffc.P3star(fishlength), type = "l", col = "darkgreen",
      lwd = 2, lty = "dashed")
lines(fishlength, wffc.P4star(fishlength), type = "l", col = "orange",
      lwd = 2, lty = "dashed")
abline(v = (1:4) * 0.18, lty = "dotted")
abline(h = (1:13) * wffc.P1star(0.18), lty = "dotted")
## End(Not run)

# Successive slopes:
(wffc.P1star((2:8)*0.18) - wffc.P1star((1:7)*0.18)) / (0.18 * 2000)
(wffc.P3star((2:8)*0.18) - wffc.P3star((1:7)*0.18)) / (0.18 * 2000)
(wffc.P4star((2:8)*0.18) - wffc.P4star((1:7)*0.18)) / (0.18 * 2000)

# More successive slopes:
MM2 <- 0.18 / 2
ind1 <- 2:12
(wffc.P4star((ind1)*MM2) - wffc.P4star((ind1-1)*MM2)) / (MM2 * 2000)

# About 50 percent more points:
wffc.P2    (2 * 0.18) / wffc.P1    (2 * 0.18)
wffc.P2star(2 * 0.18) / wffc.P1star(2 * 0.18)
```

`wffc.teams` *2008 World Fly Fishing Championships (Team results) Data*

Description

Team results of the 2008 FIPS-MOUCHE World Fly Fishing Championships held in Rotorua, New Zealand during 22–30 March 2008.

Usage

```
data(wffc.teams)
```

Format

A data frame with 18 observations on the following 5 variables. Some of these variable are described in `wffc`.

`country` a character vector.

`totalPlacings` a numeric vector; these are the summed placings over the 5 sessions and 5 team members.

`points` a numeric vector; see `wffc`.

`noofcaptures` a numeric vector.

`longestfish` a numeric vector.

Details

This data frame gives the team results of the competition. See also wffc and wffc.indiv for more details and links.

Examples

```
wffc.teams
```

---

xs.nz *Cross-sectional Data from the New Zealand Population*

Description

A cross-sectional data set of a workforce company, plus another health survey, in New Zealand during the 1990s,

Usage

```
data(xs.nz)
```

Format

A data frame with 10529 observations on the following 59 variables. For binary variables, a "1" or TRUE means yes, and "0" or FALSE means no. Also, "D" means don't know, and "-" means not applicable. The pregnancy questions were administered to women only.

regnum a numeric vector, a unique registration number. This differs from their original registration number, and the rows are sorted by their new registration number.

study1 a logical vector, Study 1 (workforce) or Study 2?

age a numeric vector, age in years.

sex a factor with levels F and M.

pulse a numeric vector, beats per minute.

sbp a numeric vector, systolic blood pressure (mm Hg).

dbp a numeric vector, diastolic blood pressure (mm Hg).

cholest a numeric vector, cholesterol (mmol/L).

height a numeric vector, in m.

weight a numeric vector, in kg.

fh.heartdisease a factor with levels 0, 1, D. Has a family history of heart disease (heart attack, angina, or had a heart bypass operation) within the immediate family (brother, sister, father or mother, blood relatives only)? Note that D means: do not know.

fh.age a factor, following from fh.heartdisease, if yes, how old was the family member when it happened (if more than one family member, give the age of the youngest person)?

fh.cancer a factor with levels 0, 1, D. Has a family history of cancer within the immediate family (blood relatives only)? Note that D means: do not know.

heartattack a numeric vector, have you ever been told by a doctor that you have had a heart attack ("coronary")?

stroke a numeric vector, have you ever been told by a doctor that you have had a stroke?

diabetes a numeric vector, have you ever been told by a doctor that you have had diabetes?

hypertension a numeric vector, have you ever been told by a doctor that you have had high blood pressure (hypertension)?

highchol a numeric vector, have you ever been told by a doctor that you have had high cholesterol?

asthma a numeric vector, have you ever been told by a doctor that you have had asthma?

cancer a numeric vector, have you ever been told by a doctor that you have had cancer?

acne a numeric vector, have you ever received treatment from a doctor for acne (pimples)?

sunburn a numeric vector, have you ever received treatment from a doctor for sunburn?

smokepassive a numeric vector, on average, how many hours each week (at work and at home) would you spend near someone who is smoking? (put `"0"` if none)

smokeever a numeric vector, have you ever smoked tailor-made or roll-you-own cigarettes once a week or more? A 1 means yes and 0 means no.

smokenow a numeric vector, do you smoke tailor-made or roll-you-own cigarettes now? A 1 means yes and 0 means no.

smokeagequit a factor, if no to `smokenow`, how old were you when you stopped smoking? Using `as.numeric(as.character(smokeagequit))` will work for those values which are not `as.character(smokeagequit) == "-"`.

smokeyears a numeric vector, if yes to `smokeever`, for how many years altogether have you smoked tailor-made or roll-you-own cigarettes?

smoketailormade a numeric vector, how many tailor-made cigarettes do you smoke each day?

smokeweekpack a numeric vector, how many packets of *roll-your-own* tobacco do you use each week? (put `"0"` if none)

smokepacketsize a numeric vector, what size packets of *roll-your-own* tobacco do you usually buy? (`"0"` means don't smoke *roll-your-owns*, else 25g or 30g or 35g or 50g)

drinkmonth a numeric vector, do you drink alcohol once a month or more?

drinkfreqweek a numeric vector, if yes to `drinkmonth`, about how often do you drink alcohol (days per week)? Note: 0.25 is once a month, 0.5 is once every two weeks, 1 is once a week, 2.5 is 2-3 days a week, 4.5 is 4-5 days a week, 6.5 is 6-7 days a week.

Further note: 1 can, small bottle or handle of beer or home brew = 1 drink, 1 quart bottle of beer = 2 drinks, 1 jug of beer = 3 drinks, 1 flagon/peter of beer = 6 drinks, 1 glass of wine, sherry = 1 drink, 1 bottle of wine = 6 drinks, 1 double nip of spirits = 1 drink.

drinkweek a numeric vector, how many drinks per week, on average. This is the average daily amount of drinks multiplied by the frequency of drinking per week. See `drinkfreqweek` on what constitutes a 'drink'.

drinkmaxday a numeric vector, in the last three months, what is the largest number of drinks that you had on any one day? Warning: some values are considered unrealistically excessive.

eggs a numeric vector, how many eggs do you eat a week (raw, boiled, scrambled, poached, or in quiche)?

chocbiscuits a numeric vector, how many chocolate biscuits do you usually eat in a week?

pregnant a factor, have you ever been pregnant for more than 5 months?

pregfirst a factor, if yes to pregnant, how old were you when your first baby was born (or you had a miscarriage after 5 months)?

preglast a factor, how old were you when your last baby was born (or you had a miscarriage after 5 months)?

babies numeric, how many babies have you given birth to?

moody a numeric vector, does your mood often go up or down?

miserable a numeric vector, do you ever feel 'just miserable' for no reason?

hurt a numeric vector, are your feelings easily hurt?

fedup a numeric vector, do you often feel 'fed up'?

nervous a numeric vector, would you call yourself a nervous person?

worrier a numeric vector, are you a worrier?

worry a numeric vector, do you worry about awful things that might happen?

tense a numeric vector, would you call yourself tense or 'highly strung'?

embarrassed a numeric vector, do you worry too long after an embarrassing experience?

nerves a numeric vector, do you suffer from 'nerves'?

nofriend a numeric vector, do you have a friend or family member that you can talk to about problems or worries that you may have? The value 1 effectively means "no", i.e., s/he has no friend or friends.

depressed a numeric vector, in your lifetime, have you ever had two weeks or more when nearly every day you felt sad or depressed?

exervig a numeric vector, how many hours per week would you do any vigorous activity or exercise either at work or away from work that makes you breathe hard and sweat? Values here ought be be less than 168.

exermod a numeric vector, how many hours per week would you do any moderate activity or exercise such as brisk walking, cycling or mowing the lawn? Values here ought be be less than 168.

feethour a numeric vector, on an average work day, how long would you spend on your feet, either standing or moving about?

ethnicity a factor with 4 levels, what ethnic group do you belong to? European = European (NZ European or British or other European), Maori = Maori, Polynesian = Pacific Island Polynesian, Other = Other (Chinese, Indian, Other).

sleep a numeric vector, how many hours do you usually sleep each night?

snore a factor with levels 0, 1, D. Do you usually snore? Note that D means: do not know.

cat a numeric vector, do you have a household pet cat?

dog a numeric vector, do you have a household pet dog?

hand a factor with levels right = right, left = left, both = either. Are you right-handed, left-handed, or no preference for left or right?

numhouse an ordered factor with 4 levels: 1 = 1, 2 = 2, 3 = 3, 4+ = four or more; how many people (including yourself) usually live in your house?

`marital` a factor with 4 levels: `single` = single, `married` = married or living with a partner, `separated` = separated or divorced, `widowed` = widowed.

`educ` an ordered factor with 4 levels: `primary` = Primary school, `secondary` = High school/secondary school, `polytechnic` = Polytechnic or similar, `university` = University. What was the highest level of education you received?

Details

The data frame is a subset of the entire data set which was collected from a confidential self-administered questionnaire administered in a large New Zealand workforce observational study conducted during 1992–3. The data were augmented by a second study consisting of retirees. The data can be considered a reasonable representation of the white male New Zealand population in the early 1990s. There were physical, lifestyle and psychological variables that were measured. The psychological variables were headed "Questions about your feelings".

Although some data cleaning was performed and logic checks conducted, anomalies remain. Some variables, of course, are subject to a lot of measurement error and bias. It is conceivable that some participants had poor reading skills! In particular, the smoking variables contain a small percentage of conflicting values, and when `NA`s are taken into account then there would be several different ways the data might be cleaned. If `smokeever == 0` then strictly speaking, only `smokepassive` is the other variable—the other smoking variables should either be `NA` or `0`.

Warning

More variables may be added in the future and these may be placed in any column position. Therefore references such as `xs.nz[,12]` are dangerous. Also, variable names may change in the future as well as their format or internal structure, e.g., `factor` versus `numeric`.

Note

More error checking are needed for the pregnancy and smoking variables.

Source

Originally, Clinical Trials Research Unit, University of Auckland, New Zealand, `http://www.ctru.auckland.ac.nz`. Originally much of the error checking and formatting was performed by Stephen Vander Hoorn. Lately (2014), more changes and error checks were made to the data by James T. Gray.

References

MacMahon, S., Norton, R., Jackson, R., Mackie, M. J., Cheng, A., Vander Hoorn, S., Milne, A., McCulloch, A. (1995). Fletcher Challenge-University of Auckland Heart & Health Study: design and baseline findings. *New Zealand Medical Journal*, **108**, 499–502.

See Also

`chest.nz`.

Examples

```
data(xs.nz)
summary(xs.nz)
```

---

yip88 *Zero-Inflated Poisson Distribution (Yip (1988) algorithm)*

Description

Fits a zero-inflated Poisson distribution based on Yip (1988).

Usage

```
yip88(link = "loglink", n.arg = NULL, imethod = 1)
```

Arguments

link	Link function for the usual λ parameter. See Links for more choices.
n.arg	The total number of observations in the data set. Needed when the response variable has all the zeros deleted from it, so that the number of zeros can be determined.
imethod	Details at CommonVGAMffArguments.

Details

The method implemented here, Yip (1988), maximizes a *conditional* likelihood. Consequently, the methodology used here deletes the zeros from the data set, and is thus related to the positive Poisson distribution (where $P(Y = 0) = 0$).

The probability function of Y is 0 with probability ϕ, and Poisson(λ) with probability $1 - \phi$. Thus

$$P(Y = 0) = \phi + (1 - \phi)P(W = 0)$$

where W is Poisson(λ). The mean, $(1 - \phi)\lambda$, can be obtained by the extractor function fitted applied to the object.

This family function treats ϕ as a scalar. If you want to model both ϕ and λ as a function of covariates, try zipoisson.

Value

An object of class "vglmff" (see vglmff-class). The object is used by modelling functions such as vglm, rrvglm and vgam.

Warning

Under- or over-flow may occur if the data is ill-conditioned. Yip (1988) only considered ϕ being a scalar and not modelled as a function of covariates. To get around this limitation, try zipoisson.

Inference obtained from summary.vglm and summary.vgam may or may not be correct. In particular, the p-values, standard errors and degrees of freedom may need adjustment. Use simulation on artificial data to check that these are reasonable.

Note

The data may be inputted in two ways. The first is when the response is a vector of positive values, with the argument `n` in `yip88` specifying the total number of observations. The second is simply include all the data in the response. In this case, the zeros are trimmed off during the computation, and the `x` and `y` slots of the object, if assigned, will reflect this.

The estimate of ϕ is placed in the `misc` slot as `@misc$pstr0`. However, this estimate is computed only for intercept models, i.e., the formula is of the form `y ~ 1`.

Author(s)

Thomas W. Yee

References

Yip, P. (1988). Inference about the mean of a Poisson distribution in the presence of a nuisance parameter. *The Australian Journal of Statistics*, **30**, 299–306.

Angers, J-F. and Biswas, A. (2003). A Bayesian analysis of zero-inflated generalized Poisson model. *Computational Statistics & Data Analysis*, **42**, 37–46.

See Also

`zipoisson`, `Zipois`, `zapoisson`, `pospoisson`, `poissonff`, `dzipois`.

Examples

```
phi <- 0.35; lambda <- 2  # Generate some artificial data
y <- rzipois(n <- 1000, lambda, phi)
table(y)

# Two equivalent ways of fitting the same model
fit1 <- vglm(y ~ 1, yip88(n = length(y)), subset = y > 0)
fit2 <- vglm(y ~ 1, yip88, trace = TRUE, crit = "coef")
(true.mean <- (1-phi) * lambda)
mean(y)
head(fitted(fit1))
fit1@misc$pstr0  # The estimate of phi

# Compare the ZIP with the positive Poisson distribution
pp <- vglm(y ~ 1, pospoisson, subset = y > 0, crit = "c")
coef(pp)
Coef(pp)
coef(fit1) - coef(pp)            # Same
head(fitted(fit1) - fitted(pp))  # Different

# Another example (Angers and Biswas, 2003) ----------------------
abdata <- data.frame(y = 0:7, w = c(182, 41, 12, 2, 2, 0, 0, 1))
abdata <- subset(abdata, w > 0)

yy <- with(abdata, rep(y, w))
fit3 <- vglm(yy ~ 1, yip88(n = length(yy)), subset = yy > 0)
```

```
fit3@misc$pstr0  # Estimate of phi (they get 0.5154 with SE 0.0707)
coef(fit3, matrix = TRUE)
Coef(fit3)  # Estimate of lambda (they get 0.6997 with SE 0.1520)
head(fitted(fit3))
mean(yy)  # Compare this with fitted(fit3)
```

Index