Package ‘DAAG’

July 21, 2025

Version 1.25.6

Title Data Analysis and Graphics Data and Functions

Author
John H Maindonald <john@statsresearch.co.nz> and W. John Braun <john.braun@ubc.ca>

Maintainer W. John Braun <john.braun@ubc.ca>

Description Functions and data sets used in examples and exercises in the text Maindonald, J.H. and Braun, W.J. (2003, 2007, 2010) ``Data Analysis and Graphics Using R'', and in an upcoming Maindonald, Braun, and Andrews text that builds on this earlier text.

LazyLoad true

LazyData true

Depends R (>= 3.5.0)

Imports lattice, latticeExtra, methods, Rdpack

RdMacros Rdpack

Suggests leaps, oz, lme4, knitr, boot, rpart, randomForest, MASS, survival, mgcv, rmarkdown, bookdown, statmod, qgam

ZipData yes

URL https://gitlab.com/daagur/DAAG

VignetteBuilder knitr

Repository CRAN

NeedsCompilation no

Date/Publication 2024-05-26 04:40:04 UTC

Contents

`DAAG-package` *The R DAAG Package*

Description

Various data sets and functions used or referred to in the book Maindonald, J.H. and Braun, W.J. (3rd edn 2010) "Data Analysis and Graphics Using R", plus other selected datasets and functions.

Details

For a list of , use `library(help="DAAG")`.

Author(s)

Author: John H Maindonald

Maintainer: W John Braun `<braun@stats.uwo.ca>`

`ACF1` *Aberrant Crypt Foci in Rat Colons*

Description

Numbers of aberrant crypt foci (ACF) in the section 1 of the colons of 22 rats subjected to a single dose of the carcinogen azoxymethane (AOM), sacrificed at 3 different times.

Usage

```
ACF1
```

Format

This data frame contains the following columns:

count Observed number of ACF in section 1 of each rat colon

endtime Time of sacrifice, in weeks following injection of AOM

Source

Ranjana P. Bird, Faculty of Human Ecology, University of Manitoba, Winnipeg, Canada.

References

E.A. McLellan, A. Medline and R.P. Bird. Dose response and proliferative characteristics of aberrant crypt foci: putative preneoplastic lesions in rat colon. Carcinogenesis, 12(11): 2093-2098, 1991.

Examples

```
sapply(split(ACF1$count,ACF1$endtime),var)
plot(count ~ endtime, data=ACF1, pch=16)
pause()
print("Poisson Regression - Example 8.3")
ACF.glm0 <- glm(formula = count ~ endtime, family = poisson, data = ACF1)
summary(ACF.glm0)

# Is there a quadratic effect?
pause()

ACF.glm <- glm(formula = count ~ endtime + I(endtime^2),
  family = poisson, data = ACF1)
summary(ACF.glm)

# But is the data really Poisson?  If not, try quasipoisson:
pause()

ACF.glm <- glm(formula = count ~ endtime + I(endtime^2),
  family = quasipoisson, data = ACF1)
summary(ACF.glm)
```

`ais` *Australian athletes data set*

Description

These data were collected in a study of how data on various characteristics of the blood varied with sport, body size, and sex of the athlete.

Usage

```
data(ais)
```

Format

A data frame with 202 observations on the following 13 variables.

rcc red blood cell count, in $10^{12} l^{-1}$

wcc while blood cell count, in 10^{12} per liter

hc hematocrit, percent

hg hemaglobin concentration, in g per decaliter

ferr plasma ferritins, ng dl^{-1}

bmi Body mass index, kg $cm^{-2}10^2$

ssf sum of skin folds

pcBfat percent Body fat

lbm lean body mass, kg

ht height, cm

wt weight, kg

sex a factor with levels `f` `m`

sport a factor with levels `B_Ball` `Field` `Gym` `Netball` `Row` `Swim` `T_400m` `T_Sprnt` `Tennis` `W_Polo`

Details

Do blood hemoglobin concentrations of athletes in endurance-related events differ from those in power-related events?

Source

These data were the basis for the analyses that are reported in Telford and Cunningham (1991).

References

Telford, R.D. and Cunningham, R.B. 1991. Sex, sport and body-size dependency of hematology in highly trained athletes. Medicine and Science in Sports and Exercise 23: 788-794.

`align2D` *Function to align points from ordination with known locations*

Description

Find the linear transformation which, applied to one set of points in the (x, y) plane, gives the best match in a least squares sense to a second set of points.

Usage

```
align2D(lat, long, x1, x2, wts=NULL)
```

Arguments

`lat`	Latitude or other co-ordinate of point to align to
`long`	Longitude or other co-ordinate of point to align to
`x1`	First coordinate of point to align
`x2`	First coordinate of point to align
`wts`	If non-NULL, specifies weights for the points.

Details

Achieves the best match, in a least squares sense, between an ordination and known locations in two-dimensionaL space.

Value

`fitlat`	Fitted values of `lat`
`fitlong`	Fitted values of `long`
`lat`	Input values of `lat`
`long`	Input values of `long`

Note

An ordination that is designed to reproduce distances between points is specified only to within an arbitrary rotation about the centroid. What linear transformation of the points ($x1$, $x2$) given by the ordination gives the best match to the known co-ordinates?

Author(s)

John H Maindonald

Examples

```
if(require(DAAG)&require(oz)){
aupts <- cmdscale(audists)
xy <- align2D(lat = aulatlong$latitude, long = aulatlong$longitude,
              x1 = aupts[, 1], x2 = aupts[, 2], wts = NULL)
oz()
fitcoords <- align2D(lat=aulatlong$latitude,
                      long=aulatlong$longitude,
                      x1=aupts[,1], x2 = aupts[,2],
                      wts=NULL)
x <-with(fitcoords,
         as.vector(rbind(lat, fitlat, rep(NA,length(lat)))))
y <-with(fitcoords,
         as.vector(rbind(long, fitlong, rep(NA,length(long)))))
points(aulatlong, col="red", pch=16, cex=1.5)
lines(x, y, col="gray40", lwd=3)
}

## The function is currently defined as
function(lat, long, x1, x2, wts=NULL){
    ## Get best fit in space of (latitude, longitude)
    if(is.null(wts))wts <- rep(1,length(x1))
    fitlat <- predict(lm(lat ~ x1+x2, weights=wts))
    fitlong <- predict(lm(long ~ x1+x2, weights=wts))
    list(fitlat = fitlat, fitlong=fitlong, lat=lat, long=long)
}
```

allbacks *Measurements on a Selection of Books*

Description

The allbacks data frame gives measurements on the volume and weight of 15 books, some of which are softback (pb) and some of which are hardback (hb). Area of the hardback covers is also included.

Usage

```
allbacks
```

Format

This data frame contains the following columns:

volume book volumes in cubic centimeters

area hard board cover areas in square centimeters

weight book weights in grams

cover a factor with levels hb hardback, pb paperback

Source

The bookshelf of J. H. Maindonald.

Examples

```
print("Multiple Regression - Example 6.1")
attach(allbacks)
volume.split <- split(volume, cover)
weight.split <- split(weight, cover)
plot(weight.split$hb ~ volume.split$hb, pch=16, xlim=range(volume), ylim=range(weight),
     ylab="Weight (g)", xlab="Volume (cc)")
points(weight.split$pb ~ volume.split$pb, pch=16, col=2)
pause()

allbacks.lm <- lm(weight ~ volume+area)
summary(allbacks.lm)
detach(allbacks)
pause()

anova(allbacks.lm)
pause()

model.matrix(allbacks.lm)
pause()
```

```
print("Example 6.1.1")
allbacks.lm0 <- lm(weight ~ -1+volume+area, data=allbacks)
summary(allbacks.lm0)
pause()

print("Example 6.1.2")
oldpar <- par(mfrow=c(2,2))
plot(allbacks.lm0)
par(oldpar)
allbacks.lm13 <- lm(weight ~ -1+volume+area, data=allbacks[-13,])
summary(allbacks.lm13)
pause()

print("Example 6.1.3")
round(coef(allbacks.lm0),2)  # Baseline for changes
round(lm.influence(allbacks.lm0)$coef,2)
```

`anesthetic` *Anesthetic Effectiveness*

Description

Thirty patients were given an anesthetic agent maintained at a predetermined level (conc) for 15 minutes before making an incision. It was then noted whether the patient moved, i.e. jerked or twisted.

Usage

```
anesthetic
```

Format

This data frame contains the following columns:

move a binary numeric vector coded for patient movement (0 = no movement, 1 = movement)

conc anesthetic concentration

logconc logarithm of concentration

nomove the complement of move

Details

The interest is in estimating how the probability of jerking or twisting varies with increasing concentration of the anesthetic agent.

Source

unknown

Examples

```
print("Logistic Regression - Example 8.1.4")

z <- table(anesthetic$nomove, anesthetic$conc)
tot <- apply(z, 2, sum)         # totals at each concentration
prop <- z[2,  ]/(tot)           # proportions at each concentration
oprop <- sum(z[2,  ])/sum(tot)  # expected proportion moving if concentration had no effect
conc <- as.numeric(dimnames(z)[[2]])
plot(conc, prop, xlab = "Concentration", ylab = "Proportion", xlim = c(.5,2.5),
    ylim = c(0, 1), pch = 16)
chw <- par()$cxy[1]
text(conc - 0.75 * chw, prop, paste(tot), adj = 1)
abline(h = oprop, lty = 2)

pause()

anes.logit <- glm(nomove ~ conc, family = binomial(link = logit),
  data = anesthetic)
anova(anes.logit)
summary(anes.logit)
```

`antigua` *Averages by block of yields for the Antigua Corn data*

Description

These data frames have yield averages by blocks (parcels). The `ant111b`dataset is a subset that has block averages of corn yields for treatment 111 only

Usage

```
data(antigua)
data(ant111b)
```

Format

A data frame with 324 observations on 7 variables.

id a numeric vector

site a factor with 8 levels.

block a factor with levels `I` `II` `III` `IV`

plot a numeric vector

trt a factor consisting of 12 levels

ears a numeric vector; note that -9999 is used as a missing value code.

harvwt a numeric vector; the average yield

Source

Andrews DF; Herzberg AM, 1985. Data. A Collection of Problems from Many Fields for the Student and Research Worker. Springer-Verlag. (pp. 339-353)

`appletaste` *Tasting experiment that compared four apple varieties*

Description

Each of 20 tasters each assessed three out of the four varieties. The experiment was conducted according to a balanced incomplete block design.

Usage

```
data(appletaste)
```

Format

A data frame with 60 observations on the following 3 variables.

aftertaste a numeric vectorApple samples were rated for `aftertaste`, by making a mark on a continuous scale that ranged from 0 (extreme dislike) to 150 (like very much).

panelist a factor with levels `a` `b` `c` `d` `e` `f` `g` `h` `i` `j` `k` `l` `m` `n` `o` `p` `q` `r` `s` `t`

product a factor with levels `298` `493` `649` `937`

Examples

```
data(appletaste)
appletaste.aov <- aov(aftertaste ~ panelist + product, data=appletaste)
termplot(appletaste.aov)
```

`audists` *Road distances between 10 Australian cities*

Description

Distances between the Australian cities of Adelaide, Alice, Brisbane, Broome, Cairns, Canberra, Darwin, Melbourne, Perth and Sydney

Usage

```
audists
```

Format

The format is: Class 'dist', i.e., a distance matrix.

Source

Australian road map

Examples

```
data(audists)
## Not run:
audists.cmd <- cmdscale(audists)
library(lattice)
xyplot(audists.cmd[,2] ~ audists.cmd[,1],
       groups=row.names(audists.cmd),
       panel = function(x, y, subscripts, groups)
                        ltext(x = x, y = y, label = groups[subscripts],
                        cex=1, fontfamily = "HersheySans"))

## End(Not run)
```

`aulatlong` *Latitudes and longitudes for ten Australian cities*

Description

Latitudes and longitudes for Adelaide, Alice, Brisbane, Broome, Cairns, Canberra, Darwin, Melbourne, Perth and Sydney; i.e., for the cities to which the road distances in `audists` relate.

Usage

```
aulatlong
```

Format

A data frame with 10 observations on the following 2 variables.

`latitude` Latitude, as a decimal number

`longitude` Latitude, as a decimal number

Source

Map of Australia showing latitude and longitude information.

Examples

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

`austpop` *Population figures for Australian States and Territories*

Description

Population figures for Australian states and territories for 1917, 1927, ..., 1997.

Usage

```
austpop
```

Format

This data frame contains the following columns:

year a numeric vector

NSW New South Wales population counts

Vic Victoria population counts

Qld Queensland population counts

SA South Australia population counts

WA Western Australia population counts

Tas Tasmania population counts

NT Northern Territory population counts

ACT Australian Capital Territory population counts

Aust Population counts for the whole country

Source

Australian Bureau of Statistics

Examples

```
print("Looping - Example 1.7")

growth.rates <- numeric(8)
for (j in seq(2,9)) {
    growth.rates[j-1] <- (austpop[9, j]-austpop[1, j])/austpop[1, j] }
growth.rates <- data.frame(growth.rates)
row.names(growth.rates) <- names(austpop[c(-1,-10)])
  # Note the use of row.names() to name the rows of the data frame
growth.rates

pause()
print("Avoiding Loops - Example 1.7b")

sapply(austpop[,-c(1,10)], function(x){(x[9]-x[1])/x[1]})
```

```
pause()
print("Plot - Example 1.8a")
attach(austpop)
plot(year, ACT, type="l") # Join the points ("l" = "line")
detach(austpop)

pause()
print("Exerice 1.12.9")
attach(austpop)
oldpar <- par(mfrow=c(2,4))
for (i in 2:9){
plot(austpop[,1], log(austpop[, i]), xlab="Year",
    ylab=names(austpop)[i], pch=16, ylim=c(0,10))}
par(oldpar)
detach(austpop)
```

`bestsetNoise` *Best Subset Selection Applied to Noise*

Description

Best subset selection applied to completely random noise. This function demonstrates how variable selection techniques in regression can often err in including explanatory variables that are indistinguishable from noise.

Usage

```
bestsetNoise(m = 100, n = 40, method = "exhaustive", nvmax = 3,
              X = NULL, y=NULL, intercept=TRUE,
              print.summary = TRUE, really.big = FALSE, ...)

bestset.noise(m = 100, n = 40, method = "exhaustive", nvmax = 3,
              X = NULL, y=NULL, intercept=TRUE,
              print.summary = TRUE, really.big = FALSE, ...)

bsnCV(m = 100, n = 40, method = "exhaustive", nvmax = 3,
              X = NULL, y=NULL, intercept=TRUE, nfolds = 2,
              print.summary = TRUE, really.big = FALSE)

bsnOpt(X = matrix(rnorm(25 * 10), ncol = 10), y = NULL, method = "exhaustive",
              nvmax = NULL, nbest = 1, intercept = TRUE, criterion = "cp",
              tcrit = NULL, print.summary = TRUE, really.big = FALSE,
         ...)

bsnVaryNvar(m = 100, nvar = nvmax:50, nvmax = 3, method = "exhaustive",
              intercept=TRUE,
```

```
              plotit = TRUE, xlab = "# of variables from which to select",
              ylab = "p-values for t-statistics", main = paste("Select 'best'",
                                                  nvmax, "variables"),
              details = FALSE, really.big = TRUE, smooth = TRUE, ...)
```

Arguments

m	the number of observations to be simulated, ignored if X is supplied.
n	the number of predictor variables in the simulated model, ignored if X is supplied.
method	Use `exhaustive` search, or `backward` selection, or `forward` selection, or `sequential` replacement.
nvmax	Number of explanatory variables in model.
X	Use columns from this matrix. Alternatively, X may be a data frame, in which case a model matrix will be formed from it. If not `NULL`, `m` and `n` are ignored.
y	If not supplied, random normal noise will be generated.
nbest	Number of models, for each choice of number of columns of explanatory variables, to return (`bsnOpt`). If `tcrit` is non-NULL, it may be important to set this greater than one, in order to have a good chance of finding models with minimum absolute *t*-statistic greater than `tcrit`.
intercept	Should an intercept be added?
nvar	range of number of candidate variables (`bsnVaryVvar`).
nfolds	For splitting the data into training and text sets, the number of folds.
criterion	Criterion to use in choosing between models with different numbers of explanatory variables (`bsnOpt`). Alternatives are "bic", or "cip" or "adjr2".
tcrit	Consider only those models for which the minimum absolute *t*-statistic is greater than `tcrit`.
print.summary	Should summary information be printed.
plotit	Plot a graph? (`bsnVaryVvar`)
xlab	*x*-label for graph (`bsnVaryVvar`)
ylab	*y*-label for graph (`bsnVaryVvar`.)
main	main title for graph (`bsnVaryVvar`.)
details	Return detailed output list (`bsnVaryVvar`)
really.big	Set to `TRUE` to allow (currently) for more than 50 explanatory variables.
smooth	Fit smooth to graph? (`bsnVaryVvar`).
...	Additional arguments, to be passed through to `regsubsets()`.

Details

If `X` is not supplied, and in any case for `bsnVaryNvar`, a set of `n` predictor variables are simulated as independent standard normal, i.e. N(0,1), variates. Additionally a N(0,1) response variable is simulated. The function `bsnOpt` selects the 'best' model with `nvmax` or fewer explanatory variables, where the argument `criterion` specifies the criterion that will be used to choose between models with different numbers of explanatory columns. Other functions select the 'best' model with nvmax explanatory columns. In any case, the selection is made using the regsubsets() function from the leaps package. (The leaps package must be installed for this function to work.)

The function bsnCV splits the data (randomly) into nfolds (2 or more) parts. It puts each part aside in turn for use to fit the model (effectively, test data), with the remaining data used for selecting the variables that will be used for fitting. One model fit is returned for each of the nfolds parts.

The function bsnVaryVvar makes repeated calls to bestsetNoise

Value

bestsetNoise returns the lm model object for the "best" model with nvmax explanatory columns.

bsnCV returns as many models as there are folds.

bsnVaryVvar silently returns either (details=FALSE) a matrix that has *p*-values of the coefficients for the 'best' choice of model for each different number of candidate variables, or (details=TRUE) a list with elements:

coef	A matrix of sets of regression coefficients
SE	A matrix of standard errors
pval	A matrix of *p*-values

Matrices have one row for each choice of nvar. The statistics returned are for the 'best' model with nvmax explanatory variables.

bsnOpt silently returns a list with elements:

u1	'best' model (lm object) with nvmax or fewer columns of predictors. If tcrit is non-NULL, and there is no model for which all coefficients have *t*-statistics less than tcrit in absolute value, u1 will be NULL.
tcrit	For each model, the minimum of the absolute values of the *t*-statistics.
regsubsets_obj	The object returned by the call to regsubsets.

Note

These functions are primarily designed to demonstrate the biases that can be expected, relative to theoretical estimates of standard errors of parameters and other fitted model statistics, when there is prior selection of the columns that are to be included in the model. With the exception of bsnVaryNvar, they can also be used with an X and y for actual data. In that case, the *p*-values should be compared with those obtained from repeated use of the function where y is random noise, as a check on the extent of selection effects.

Author(s)

J.H. Maindonald

See Also

lm

Examples

```
leaps.out <- try(require(leaps, quietly=TRUE))
leaps.out.log <- is.logical(leaps.out)
if ((leaps.out.log==TRUE)&(leaps.out==TRUE)){
bestsetNoise(20,6) # `best' 3-variable regression for 20 simulated observations
                   # on 7 unrelated variables (including the response)
bsnCV(20,6) # `best' 3-variable regressions (one for each fold) for 20
                   # simulated observations on 7 unrelated variables
                   # (including the response)
bsnVaryNvar(m = 50, nvar = 3:6, nvmax = 3, method = "exhaustive",
            plotit=FALSE, details=TRUE)
bsnOpt()
}
```

---

biomass *Biomass Data*

Description

The `biomass` data frame has 135 rows and 8 columns. The `rainforest` data frame is a subset of this one.

Usage

```
biomass
```

Format

This data frame contains the following columns:

dbh a numeric vector

wood a numeric vector

bark a numeric vector

fac26 a factor with 3 levels

root a numeric vector

rootsk a numeric vector

branch a numeric vector

species a factor with levels `Acacia mabellae`, `C. fraseri`, `Acmena smithii`, `B. myrtifolia`

Source

J. Ash, Australian National University

References

Ash, J. and Helman, C. (1990) Floristics and vegetation biomass of a forest catchment, Kioloa, south coastal N.S.W. Cunninghamia, 2: 167-182.

bomregions2021 *Australian and Related Historical Annual Climate Data, by Region*

Description

Australian regional temperature data, Australian regional rainfall data, and Annual SOI, are given for the years 1900-2021. The regional rainfall and temperature data are area-weighted averages for the respective regions. The Southern Oscillation Index (SOI) is the difference in barometric pressure at sea level between Tahiti and Darwin.

Usage

```
data("bomregions2021")
```

Format

These data frames contains the following columns:

Year Year

seAVt Southeastern region average temperature (degrees C)

southAVt Southern temperature

eastAVt Eastern temperature

northAVt Northern temperature

swAVt Southwestern temperature

qldAVt temperature

nswAVt temperature

ntAVt temperature

saAVt temperature

tasAVt temperature

vicAVt temperature

waAVt temperature

mdbAVt Murray-Darling basin temperature

ausAVt Australian average temperature, area-weighted mean

seRain Southeast Australian annual rainfall (mm)

southRain Southern rainfall

eastRain Eastern rainfall

northRain Northern rainfall

swRain Southwest rainfall

qldRain Queensland rainfall

nswRain NSW rainfall

ntRain Northern Territory rainfall

saRain South Australian rainfall

tasRain Tasmanian rainfall

vicRain Victorian rainfall

waRain West Australian rainfall

mdbRain Murray-Darling basin rainfall

ausRain Australian average rainfall, area weighted

SOI Annual average Southern Oscillation Index

sunspot Yearly mean sunspot number

co2mlo Moana Loa CO2 concentrations, from 1959

co2law Moana Loa CO2 concentrations, 1900 to 1978

CO2 CO2 concentrations, composite series

avDMI Annual average Dipole Mode Index, for the Indian Ocean Dipole, from 1950

Source

Australian Bureau of Meteorology web pages:

Go to the url http://www.bom.gov.au/climate/change/, choose timeseries to display, then click "Download data"

For the SOI data, go to the url http://www.bom.gov.au/climate/enso/.

The CO2 series `co2law`, for Law Dome ice core data. is from `https://data.ess-dive.lbl.gov/portals/CDIAC/`.

The Moana Loa CO2 series `co2mlo` is from Dr. Pieter Tans, NOAA/ESRL (`https://gml.noaa.gov/ccgg/trends/`)

The series `CO2` is a composite series, obtained by adding 0.46 to the Law data for 1900 to 1958, then following this with the Moana Loa data that is avaiable from 1959. The addition of 0.46 brings the average of the Law data into agreement with that for the Moana Loa data for the period 1959 to 1968.

The yearly mean sunspot number is a subset of one of several sunspot series that are available from WDC-SILSO, Royal Observatory of Belgium, Brussels. `https://www.sidc.be/silso/datafiles/`

The dipole mode index data are from `https://ds.data.jma.go.jp/tcc/tcc/products/elnino/index/Readme_iod.txt`. Note also `https://stateoftheocean.osmc.noaa.gov/sur/ind/dmi.php`, which has details of several other such series.

References

D.M. Etheridge, L.P. Steele, R.L. Langenfelds, R.J. Francey, J.-M. Barnola and V.I. Morgan, 1998, *Historical CO2 records from the Law Dome DE08, DE08-2, and DSS ice cores*, in Trends: A Compendium of Data on Global Change, on line at Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, U.S. Department of Energy, Oak Ridge, Tenn., U.S.A.

Lavery, B., Joung, G. and Nicholls, N. 1997. An extended high-quality historical rainfall dataset for Australia. Australian Meteorological Magazine, 46, 27-38.

Nicholls, N., Lavery, B., Frederiksen, C.\ and Drosdowsky, W. 1996. Recent apparent changes in relationships between the El Nino – southern oscillation and Australian rainfall and temperature. Geophysical Research Letters 23: 3357-3360.

SIDC-team, World Data Center for the Sunspot Index, Royal Observatory of Belgium, Monthly Report on the International Sunspot Number, online catalogue of the sunspot index: https://www.sidc.be/silso/datafiles

Examples

```
plot(ts(bomregions2021[, c("mdbRain","SOI")], start=1900),
     panel=function(y,...)panel.smooth(bomregions2021$Year, y,...))
avrain <- bomregions2021[,"mdbRain"]
xbomsoi <- with(bomregions2021, data.frame(Year=Year, SOI=SOI,
                cuberootRain=avrain^0.33))
xbomsoi$trendSOI <- lowess(xbomsoi$SOI, f=0.1)$y
xbomsoi$trendRain <- lowess(xbomsoi$cuberootRain, f=0.1)$y
xbomsoi$detrendRain <-
  with(xbomsoi, cuberootRain - trendRain + mean(trendRain))
xbomsoi$detrendSOI <-
  with(xbomsoi, SOI - trendSOI + mean(trendSOI))
## Plot time series avrain and SOI: ts object xbomsoi
plot(ts(xbomsoi[, c("cuberootRain","SOI")], start=1900),
     panel=function(y,...)panel.smooth(xbomsoi$Year, y,...),
     xlab = "Year", main="", ylim=list(c(250, 800),c(-20,25)))
par(mfrow=c(1,2))
rainpos <- pretty(xbomsoi$cuberootRain^3, 6)
plot(cuberootRain ~ SOI, data = xbomsoi,
     ylab = "Rainfall (cube root scale)", yaxt="n")
axis(2, at = rainpos^0.33, labels=paste(rainpos))
mtext(side = 3, line = 0.8, "A", adj = -0.025)
with(xbomsoi, lines(lowess(cuberootRain ~ SOI, f=0.75)))
plot(detrendRain ~ detrendSOI, data = xbomsoi,
     xlab="Detrended SOI", ylab = "Detrended rainfall", yaxt="n")
axis(2, at = rainpos^0.33, labels=paste(rainpos))
with(xbomsoi, lines(lowess(detrendRain ~ detrendSOI, f=0.75)))
mtext(side = 3, line = 0.8, "B", adj = -0.025)
par(mfrow=c(1,1))
```

---

bomsoi *Southern Oscillation Index Data*

Description

The Southern Oscillation Index (SOI) is the difference in barometric pressure at sea level between Tahiti and Darwin. Annual SOI and Australian rainfall data, for the years 1900-2005, are given. Australia's annual mean rainfall is an area-weighted average of the total annual precipitation at approximately 370 rainfall stations around the country.

Usage

```
bomsoi
```

Format

This data frame contains the following columns:

Year a numeric vector

Jan average January SOI values for each year

Feb average February SOI values for each year

Mar average March SOI values for each year

Apr average April SOI values for each year

May average May SOI values for each year

Jun average June SOI values for each year

Jul average July SOI values for each year

Aug average August SOI values for each year

Sep average September SOI values for each year

Oct average October SOI values for each year

Nov average November SOI values for each year

Dec average December SOI values for each year

SOI a numeric vector consisting of average annual SOI values

avrain a numeric vector consisting of a weighted average annual rainfall at a large number of Australian sites

NTrain Northern Territory rain

northRain north rain

seRain southeast rain

eastRain east rain

southRain south rain

swRain southwest rain

Source

Australian Bureau of Meteorology web pages:

http://www.bom.gov.au/climate/change/rain02.txt and http://www.bom.gov.au/climate/current/soihtm1.shtml

References

Nicholls, N., Lavery, B., Frederiksen, C.\ and Drosdowsky, W. 1996. Recent apparent changes in relationships between the El Nino – southern oscillation and Australian rainfall and temperature. Geophysical Research Letters 23: 3357-3360.

Examples

```
plot(ts(bomsoi[, 15:14], start=1900),
     panel=function(y,...)panel.smooth(1900:2005, y,...))
pause()

# Check for skewness by comparing the normal probability plots for
# different a, e.g.
par(mfrow = c(2,3))
for (a in c(50, 100, 150, 200, 250, 300))
qqnorm(log(bomsoi[, "avrain"] - a))
  # a = 250 leads to a nearly linear plot

pause()

par(mfrow = c(1,1))
plot(bomsoi$SOI, log(bomsoi$avrain - 250), xlab = "SOI",
     ylab = "log(avrain = 250)")
lines(lowess(bomsoi$SOI)$y, lowess(log(bomsoi$avrain - 250))$y, lwd=2)
  # NB: separate lowess fits against time
lines(lowess(bomsoi$SOI, log(bomsoi$avrain - 250)))
pause()

xbomsoi <-
  with(bomsoi, data.frame(SOI=SOI, cuberootRain=avrain^0.33))
xbomsoi$trendSOI <- lowess(xbomsoi$SOI)$y
xbomsoi$trendRain <- lowess(xbomsoi$cuberootRain)$y
rainpos <- pretty(bomsoi$avrain, 5)
with(xbomsoi,
     {plot(cuberootRain ~ SOI, xlab = "SOI",
           ylab = "Rainfall (cube root scale)", yaxt="n")
     axis(2, at = rainpos^0.33, labels=paste(rainpos))
## Relative changes in the two trend curves
     lines(lowess(cuberootRain ~ SOI))
     lines(lowess(trendRain ~ trendSOI), lwd=2)
  })
pause()

xbomsoi$detrendRain <-
  with(xbomsoi, cuberootRain - trendRain + mean(trendRain))
xbomsoi$detrendSOI <-
  with(xbomsoi, SOI - trendSOI + mean(trendSOI))
oldpar <- par(mfrow=c(1,2), pty="s")
plot(cuberootRain ~ SOI, data = xbomsoi,
     ylab = "Rainfall (cube root scale)", yaxt="n")
axis(2, at = rainpos^0.33, labels=paste(rainpos))
with(xbomsoi, lines(lowess(cuberootRain ~ SOI)))
plot(detrendRain ~ detrendSOI, data = xbomsoi,
  xlab="Detrended SOI", ylab = "Detrended rainfall", yaxt="n")
axis(2, at = rainpos^0.33, labels=paste(rainpos))
with(xbomsoi, lines(lowess(detrendRain ~ detrendSOI)))
pause()
```

```
par(oldpar)
attach(xbomsoi)
xbomsoi.ma0 <- arima(detrendRain, xreg=detrendSOI, order=c(0,0,0))
# ordinary regression model

xbomsoi.ma12 <- arima(detrendRain, xreg=detrendSOI,
                      order=c(0,0,12))
# regression with MA(12) errors -- all 12 MA parameters are estimated
xbomsoi.ma12
pause()

xbomsoi.ma12s <- arima(detrendRain, xreg=detrendSOI,
                      seasonal=list(order=c(0,0,1), period=12))
# regression with seasonal MA(1) (lag 12) errors -- only 1 MA parameter
# is estimated
xbomsoi.ma12s
pause()

xbomsoi.maSel <- arima(x = detrendRain, order = c(0, 0, 12),
                        xreg = detrendSOI, fixed = c(0, 0, 0,
                        NA, rep(0, 4), NA, 0, NA, NA, NA, NA),
                        transform.pars=FALSE)
# error term is MA(12) with fixed 0's at lags 1, 2, 3, 5, 6, 7, 8, 10
# NA's are used to designate coefficients that still need to be estimated
# transform.pars is set to FALSE, so that MA coefficients are not
# transformed (see help(arima))

detach(xbomsoi)
pause()

Box.test(resid(lm(detrendRain ~ detrendSOI, data = xbomsoi)),
          type="Ljung-Box", lag=20)

pause()

attach(xbomsoi)
 xbomsoi2.maSel <- arima(x = detrendRain, order = c(0, 0, 12),
                         xreg = poly(detrendSOI,2), fixed = c(0,
                         0, 0, NA, rep(0, 4), NA, 0, rep(NA,5)),
                         transform.pars=FALSE)
 xbomsoi2.maSel
qqnorm(resid(xbomsoi.maSel, type="normalized"))
detach(xbomsoi)
```

---

bostonc *Boston Housing Data – Corrected*

Description

The corrected Boston housing data (from http://lib.stat.cmu.edu/datasets/).

Usage

```
bostonc
```

Format

A single vector containing the contents of "boston_corrected.txt".

Source

Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978. corrected by Kelley Pace (kpace@unix1.sncc.lsu.edu)

---

bounce *Separate plotting positions for labels, to avoid overlap*

Description

Return univariate plotting positions in which neighboring points are separated, if and as necessary, so that they are the specified minimum distance apart.

Usage

```
bounce(y, d, log = FALSE)
```

Arguments

`y`	A numeric vector of plotting positions
`d`	Minimum required distance between neighboring positions
`log`	`TRUE` if values are will be plotted on a logarithmic scale.

Details

The centroid(s) of groups of points that are moved relative to each other remain the same.

Value

A vector of values such that, when plotted along a line, neighboring points are the required minimum distance apart.

Note

If values are plotted on a logarithmic scale, `d` is the required distance apart on that scale. If a base other than 10 is required, set `log` equal to that base. (Note that base 10 is the default for `plot` with `log=TRUE`.)

Author(s)

John Maindonald

See Also

See also onewayPlot

Examples

```
bounce(c(4, 1.8, 2, 6), d=.4)
bounce(c(4, 1.8, 2, 6), d=.1, log=TRUE)
```

---

capstring — *Converts initial character of a string to upper case*

Description

This function is useful for use before plotting, if one wants capitalized axis labels or factor levels.

Usage

```
capstring(names)
```

Arguments

names	a character vector

Value

A character vector with upper case initial values.

Author(s)

W.J. Braun

Examples

```
capstring(names(tinting)[c(3,4)])

library(lattice)
levels(tinting$agegp) <- capstring(levels(tinting$agegp))
xyplot(csoa ~ it | sex * agegp, data=tinting)
```

carprice — *US Car Price Data*

Description

U.S. data extracted from `Cars93`, a data frame in the MASS package.

Usage

```
carprice
```

Format

This data frame contains the following columns:

Type Type of car, e.g. Sporty, Van, Compact

Min.Price Price for a basic model

Price Price for a mid-range model

Max.Price Price for a 'premium' model

Range.Price Difference between Max.Price and Min.Price

RoughRange Rough.Range plus some N(0,.0001) noise

gpm100 The number of gallons required to travel 100 miles

MPG.city Average number of miles per gallon for city driving

MPG.highway Average number of miles per gallon for highway driving

Source

MASS package

References

Venables, W.N.\ and Ripley, B.D., 4th edn 2002. Modern Applied Statistics with S. Springer, New York.

See also 'R' Complements to Modern Applied Statistics with S-Plus, available from http://www.stats.ox.ac.uk/pub/MASS3/

Examples

```
print("Multicollinearity - Example 6.8")
pairs(carprice[,-c(1,8,9)])

carprice1.lm <- lm(gpm100 ~ Type+Min.Price+Price+Max.Price+Range.Price,
    data=carprice)
round(summary(carprice1.lm)$coef,3)
pause()
```

```
alias(carprice1.lm)
pause()

carprice2.lm <- lm(gpm100 ~ Type+Min.Price+Price+Max.Price+RoughRange, data=carprice)
round(summary(carprice2.lm)$coef, 2)
pause()

carprice.lm <- lm(gpm100 ~ Type + Price, data = carprice)
round(summary(carprice.lm)$coef,4)
pause()

summary(carprice1.lm)$sigma  # residual standard error when fitting all 3 price variables
pause()

summary(carprice.lm)$sigma    # residual standard error when only price is used
pause()

vif(lm(gpm100 ~ Price, data=carprice)) # Baseline Price
pause()

vif(carprice1.lm)    # includes Min.Price, Price & Max.Price
pause()

vif(carprice2.lm)    # includes Min.Price, Price, Max.Price & RoughRange
pause()

vif(carprice.lm)     # Price alone
```

Cars93.summary *A Summary of the Cars93 Data set*

Description

The Cars93.summary data frame has 6 rows and 4 columns created from information in the Cars93 data set in the Venables and Ripley MASS package. Each row corresponds to a different class of car (e.g. Compact, Large, etc.).

Usage

```
Cars93.summary
```

Format

This data frame contains the following columns:

Min.passengers minimum passenger capacity for each class of car

Max.passengers maximum passenger capacity for each class of car

No.of.cars number of cars in each class

abbrev a factor with levels C Compact, L Large, M Mid-Size, Sm Small, Sp Sporty, V Van

Source

Lock, R. H. (1993) 1993 New Car Data. Journal of Statistics Education 1(1)

References

MASS library

Examples

```
type <- Cars93.summary$abbrev
type <- Cars93.summary[,4]
type <- Cars93.summary[,"abbrev"]
type <- Cars93.summary[[4]] # Take the object that is stored
                            # in the fourth list element.
type
pause()

attach(Cars93.summary)
  # R can now access the columns of Cars93.summary directly
abbrev
detach("Cars93.summary")
pause()

#  To change the name of the \verb!abbrev! variable (the fourth column)
names(Cars93.summary)[4] <- "code"
pause()

#  To change all of the names, try
names(Cars93.summary) <- c("minpass","maxpass","number","code")
```

`cerealsugar` *Percentage of Sugar in Breakfast Cereal*

Description

Measurements of sugar content in frosted flakes breakfast cereal.

Usage

```
cerealsugar
```

Format

A vector of 100 measurements.

cfseal *Cape Fur Seal Data*

Description

The `cfseal` data frame has 30 rows and 11 columns consisting of weight measurements for various organs taken from 30 Cape Fur Seals that died as an unintended consequence of commercial fishing.

Usage

```
cfseal
```

Format

This data frame contains the following columns:

age a numeric vector

weight a numeric vector

heart a numeric vector

lung a numeric vector

liver a numeric vector

spleen a numeric vector

stomach a numeric vector

leftkid a numeric vector

rightkid a numeric vector

kidney a numeric vector

intestines a numeric vector

Source

Stewardson, C.L., Hemsley, S., Meyer, M.A., Canfield, P.J. and Maindonald, J.H. 1999. Gross and microscopic visceral anatomy of the male Cape fur seal, Arctocephalus pusillus pusillus (Pinnepedia: Otariidae), with reference to organ size and growth. Journal of Anatomy (Cambridge) 195: 235-255. (WWF project ZA-348)

Examples

```
print("Allometric Growth - Example 5.7")

cfseal.lm <- lm(log(heart) ~ log(weight), data=cfseal); summary(cfseal.lm)
plot(log(heart) ~ log(weight), data = cfseal, pch=16, xlab = "Heart Weight (g, log scale)",
ylab = "Body weight (kg, log scale)", axes=FALSE)
heartaxis <- 100*(2^seq(0,3))
bodyaxis <- c(20,40,60,100,180)
axis(1, at = log(bodyaxis), lab = bodyaxis)
```

```
axis(2, at = log(heartaxis), lab = heartaxis)
box()
abline(cfseal.lm)
```

---

cities *Populations of Major Canadian Cities (1992-96)*

Description

Population estimates for several Canadian cities.

Usage

```
cities
```

Format

This data frame contains the following columns:

CITY a factor, consisting of the city names

REGION a factor with 5 levels (ATL=Atlantic, ON=Ontario, QC=Quebec, PR=Prairies, WEST=Alberta and British Columbia) representing the location of the cities

POP1992 a numeric vector giving population in 1000's for 1992

POP1993 a numeric vector giving population in 1000's for 1993

POP1994 a numeric vector giving population in 1000's for 1994

POP1995 a numeric vector giving population in 1000's for 1995

POP1996 a numeric vector giving population in 1000's for 1996

Source

Statistics Canada

Examples

```
cities$have <- factor((cities$REGION=="ON")|(cities$REGION=="WEST"))
plot(POP1996~POP1992, data=cities, col=as.integer(cities$have))
```

`codling` *Dose-mortality data, for fumigation of codling moth with methyl bromide*

Description

Data are from trials that studied the mortality response of codling moth to fumigation with methyl bromide.

Usage

```
data(codling)
```

Format

A data frame with 99 observations on the following 10 variables.

dose Injected dose of methyl bromide, in gm per cubic meter

tot Number of insects in chamber

dead Number of insects dying

pobs Proportion dying

cm Control mortality, i.e., at dose 0

ct Concentration-time sum

Cultivar a factor with levels `BRAEBURN` `FUJI` `GRANNY` `Gala` `ROYAL` `Red Delicious` `Splendour`

gp a factor which has a different level for each different combination of `Cultivar`, `year` and `rep` (replicate).

year a factor with levels `1988` `1989`

numcm a numeric vector: total number of control insects

Details

The research that generated these data was in part funded by New Zealand pipfruit growers. The published analysis was funded by New Zealand pipfruit growers. See also `sorption`.

Source

Maindonald, J.H.; Waddell, B.C.; Petry, R.J. 2001. Apple cultivar effects on codling moth (Lepidoptera: Tortricidae) egg mortality following fumigation with methyl bromide. Postharvest Biology and Technology 22: 99-110.

`compareTreecalcs` *Error rate comparisons for tree-based classification*

Description

Compare error rates, between different functions and different selection rules, for an approximately equal random division of the data into a training and test set.

Usage

```
compareTreecalcs(x = yesno ~ ., data = DAAG::spam7, cp = 0.00025, fun = c("rpart",
"randomForest"))
```

Arguments

`x`	model formula
`data`	an data frame in which to interpret the variables named in the formula
`cp`	setting for the cost complexity parameter `cp`, used by rpart()
`fun`	one or both of "rpart" and "randomForest"

Details

Data are randomly divided into two subsets, I and II. The function(s) are used in the standard way for calculations on subset I, and error rates returined that come from the calculations carried out by the function(s). Predictions are made for subset II, allowing the calculation of a completely independent set of error rates.

Value

If `rpart` is specified in `fun`, the following:

`rpSEcvI`	the estimated cross-validation error rate when `rpart()` is run on the training data (I), and the one-standard error rule is used
`rpcvI`	the estimated cross-validation error rate when `rpart()` is run on subset I, and the model used that gives the minimum cross-validated error rate
`rpSEtest`	the error rate when the model that leads to `rpSEcvI` is used to make predictions for subset II
`rptest`	the error rate when the model that leads to `rpcvI` is used to make predictions for subset II
`nSErule`	number of splits required by the one standard error rule
`nREmin`	number of splits to give the minimum error

If `rpart` is specified in `fun`, the following:

`rfcvI`	the out-of-bag (OOB) error rate when `randomForest()` is run on subset I
`rftest`	the error rate when the model that leads to `rfcvI` is used to make predictions for subset II

Author(s)

John Maindonald

---

component.residual *Component + Residual Plot*

Description

Component + Residual plot for a term in a `lm` model.

Usage

```
component.residual(lm.obj, which = 1, xlab = "Component",
    ylab = "C+R")
```

Arguments

`lm.obj`	A `lm` object
`which`	numeric code for the term in the `lm` formula to be plotted
`xlab`	label for the x-axis
`ylab`	label for the y-axis

Value

A scatterplot with a smooth curve overlaid.

Author(s)

J.H. Maindonald

See Also

`lm`

Examples

```
mice12.lm <- lm(brainwt ~ bodywt + lsize, data=litters)
oldpar <- par(mfrow = c(1,2))
component.residual(mice12.lm, 1, xlab = "Body weight", ylab= "t(Body weight) + e")
component.residual(mice12.lm, 2, xlab = "Litter size", ylab= "t(Litter size) + e")
par(oldpar)
```

`confusion` *Given actual and predicted group assignments, give the confusion matrix*

Description

Given actual and predicted group assignments, give the confusion matrix

Usage

```
confusion(actual, predicted, gpnames = NULL, rowcol=c("actual", "predicted"),
printit = c("overall","confusion"), prior = NULL, digits=3)
```

Arguments

`actual`	Actual (prior) group assigments
`predicted`	Predicted group assigments.
`gpnames`	Names for groups, if different from `levels(actual)`
`rowcol`	For predicted categories to appear as rows, specify `rowcol="predicted"`
`printit`	Character vector. Print `"overall"`, or `"confusion"` matrix, or both.
`prior`	Prior probabilities for groups, if different from the relative group frequencies
`digits`	Number of decimal digits to display in printed output

Details

Predicted group assignments should be estimated from cross-validation or from bootstrap out-of-bag data. Better still, work with assignments for test data that are completely separate from the data used to dervive the model.

Value

A list with elements overall (overall accuracy), confusion (confusion matrix) and prior (prior used for calculation of overall accuracy)

Author(s)

John H Maindonald

References

Maindonald and Braun: 'Data Analysis and Graphics Using R', 3rd edition 2010, Section 12.2.2

Examples

```
library(MASS)
library(DAAG)
cl <- lda(species ~ length+breadth, data=cuckoos, CV=TRUE)$class
confusion(cl, cuckoos$species)

## The function is currently defined as
function (actual, predicted, gpnames = NULL,
            rowcol = c("actual", "predicted"),
            printit = c("overall","confusion"),
            prior = NULL, digits = 3)
{
  if (is.null(gpnames))
    gpnames <- levels(actual)
  if (is.logical(printit)){
    if(printit)printit <- c("overall","confusion")
    else printit <- ""
  }
  tab <- table(actual, predicted)
  acctab <- t(apply(tab, 1, function(x) x/sum(x)))
  dimnames(acctab) <- list(Actual = gpnames, `Predicted (cv)` = gpnames)
  if (is.null(prior)) {
    relnum <- table(actual)
    prior <- relnum/sum(relnum)
    acc <- sum(tab[row(tab) == col(tab)])/sum(tab)
  }
  else {
    acc <- sum(prior * diag(acctab))
  }
  names(prior) <- gpnames
  if ("overall"%in%printit) {
    cat("Overall accuracy =", round(acc, digits), "\n")
    if(is.null(prior)){
      cat("This assumes the following prior frequencies:",
          "\n")
      print(round(prior, digits))
    }
  }
  if ("confusion"%in%printit) {
    cat("\nConfusion matrix", "\n")
    print(round(acctab, digits))
  }
  invisible(list(overall=acc, confusion=acctab, prior=prior))
}
```

`coralPval` *P-values from biological expression array data*

Description

P-values were calculated for each of 3072 genes, for data that compared expression values between post-settlement coral larvae and pre-settlement coral larvae.

Usage

```
data("coralPval")
```

Format

The format is: num [1:3072, 1] 8.60e-01 3.35e-08 3.96e-01 2.79e-01 6.36e-01 ...

Details

t-statistics, and hence p-values, were derived from five replicate two-colour micro-array slides. Details are in a vignette that accompanies the **DAAGbio** package.

Source

See the `?DAAGbio::coralRG`

References

Grasso, L. C.; Maindonald, J.; Rudd, S.; Hayward, D. C.; Saint, R.; Miller, D. J.; and Ball, E. E., 2008. Microarray analysis identifies candidate genes for key roles in coral development. BMC Genomics, 9:540.

Examples

```
## From p-values, calculate Benjamini-Hochberg false discrimination rates
fdr <- p.adjust(DAAG::coralPval, method='BH')
## Number of genes identified as differentially expressed for FDR = 0.01
sum(fdr<=0.01)
```

`cottonworkers` *Occupation and wage profiles of British cotton workers*

Description

Numbers are given in different categories of worker, in each of two investigations. The first source of information is the Board of Trade Census that was conducted on 1886. The second is a relatively informal survey conducted by US Bureau of Labor representatives in 1889, for use in official reports.

Usage

```
data(cottonworkers)
```

Format

A data frame with 14 observations on the following 3 variables.

census1886 Numbers of workers in each of 14 different categories, according to the Board of Trade wage census that was conducted in 1886

survey1889 Numbers of workers in each of 14 different categories, according to data collected in 1889 by the US Bureau of Labor, for use in a report to the US Congress and House of Representatives

avwage Average wage, in pence, as estimated in the US Bureau of Labor survey

Details

The data in `survey1889` were collected in a relatively informal manner, by approaching individuals on the street. Biases might therefore be expected.

Source

United States congress, House of Representatives, Sixth Annual Report of the Commissioner of Labor, 1890, Part III, Cost of Living (Washington D.C. 1891); idem., Seventh Annual Report of the Commissioner of Labor, 1891, Part III, Cost of Living (Washington D.C. 1892)

Return of wages in the principal textile trades of the United Kingdom, with report therein. (P.P. 1889, LXX). United Kingdom Official Publication.

References

Boot, H. M. and Maindonald, J. H. 2007. New estimates of age- and sex- specific earnings and the male-female earnings gap in the British cotton industry, 1833-1906. *Economic History Review*. Published online 28-Aug-2007 doi: 10.1111/j.1468-0289.2007.00398.x

Examples

```
data(cottonworkers)
str(cottonworkers)
plot(survey1889 ~ census1886, data=cottonworkers)
plot(I(avwage*survey1889) ~ I(avwage*census1886), data=cottonworkers)
```

`cricketer` *Lifespans of UK 1st class cricketers born 1840-1960*

Description

Year and birth, lifespan, etc, of British first class cricketers, born 1840-1960, whose handedness could be determined from information in the Who's who of cricketers. The status (alive=0, dead =1), and lifetime or lifespan, is for 1992.

Usage

```
data(cricketer)
```

Format

A data frame with 5960 observations on the following 8 variables.

`left` a factor with levels `right` `left`

`year` numeric, year of birth

`life` numeric, lifetime or lifespan to 1992

`dead` numeric (0 = alive (censored), 1 = dead, in 1992)

`acd` numeric (0 = not accidental or not dead, 1 = accidental death)

`kia` numeric (0 = not killed in action, 1 = killed in action)

`inbed` numeric (0 = did not die in bed, 1 = died in bed)

`cause` a factor with levels `alive` `acd` (accidental death) `inbed` (died in bed)

Details

Note that those 'killed in action' (mostly during World Wars I and II) form a subset of those who died by accident.

Source

John Aggleton, Martin Bland. Data were collated as described in Aggleton et al.

References

Aggleton JP, Bland JM, Kentridge RW, Neave NJ 1994. Handedness and longevity: an archival study of cricketers. British Medical Journal 309, 1681-1684.

Bailey P, Thorne P, Wynne-Thomas P. 1993. Who's Who of Cricketers. 2nd ed, London, Hamlyn.

Bland M and Altman D. 2005. Do the left-handed die young? Significance 2, 166-170.

See Also

`earlycrcktr`.

Examples

```
data(cricketer)
numLH <- xtabs(~ left+year, data=cricketer)
propLH <- prop.table(numLH, margin=2)[2,]
yr <- as.numeric(colnames(numLH))
plot(propLH ~ yr)
cricketer$lh <- unclass(cricketer$left)-1
left2.hat <- fitted(lm(lh ~ poly(year,2), data=cricketer))
ord <- order(cricketer$year)
lines(left2.hat[ord] ~ cricketer$year[ord])
library(splines)
ns3.hat <- fitted(lm(lh ~ ns(year,3), data=cricketer))
lines(ns3.hat[ord] ~ cricketer$year[ord], col="red")
require(survival)
summary(coxph(Surv(life, kia) ~ bs(year,3) +left, data=cricketer))
```

```
cricketer$notacdDead <- with(cricketer, {dead[acd==1]<-0; dead})
summary(coxph(Surv(life, notacdDead) ~ ns(year,2) +left, data=cricketer))
```

---

cuckoohosts *Comparison of cuckoo eggs with host eggs*

Description

These data compare mean length, mean breadth, and egg color, between cuckoos and their hosts.

Usage

```
cuckoohosts
```

Format

A data frame with 10 observations on the following 12 variables.

clength mean length of cuckoo eggs in given host's nest

cl.sd standard deviation of cuckoo egg lengths

cbreadth mean breadth of cuckoo eggs in given host's nest

cb.sd standard deviation of cuckoo egg breadths

cnum number of cuckoo eggs

hlength length of host eggs

hl.sd standard deviation of host egg lengths

hbreadth breadth of host eggs

hb.sd standard deviation of host egg breadths

hnum number of host eggs

match number of eggs where color matched

nomatch number where color did not match

Details

Although from the same study that generated data in the data frame `cuckoos`, the data do not match precisely. The cuckoo egg lengths and breadths are from the tables on page 168, the host egg lengths and breadths from Appendix IV on page 176, and the color match counts from the table on page 171.

Source

Latter, O.H., 1902. The egg of *cuculus canorus*. an inquiry into the dimensions of the cuckoo's egg and the relation of the variations to the size of the eggs of the foster-parent, with notes on coloration, &c. *Biometrika*, 1:164–176.

Examples

```
cuckoohosts
str(cuckoohosts)
plot(cuckoohosts)
with(cuckoohosts,
     plot(c(clength,hlength),c(cbreadth,hbreadth),col=rep(1:2,c(6,6))))
```

`cuckoos` *Cuckoo Eggs Data*

Description

Length and breadth measurements of 120 eggs lain in the nests of six different species of host bird.

Usage

```
cuckoos
```

Format

This data frame contains the following columns:

length the egg lengths in millimeters

breadth the egg breadths in millimeters

species a factor with levels `hedge.sparrow`, `meadow.pipit`, `pied.wagtail`, `robin`, `tree.pipit`, `wren`

id a numeric vector

Source

Latter, O.H. (1902). The eggs of Cuculus canorus. An Inquiry into the dimensions of the cuckoo's egg and the relation of the variations to the size of the eggs of the foster-parent, with notes on coloration, &c. Biometrika i, 164. Tippett (1931) gives summary details of the data.

References

Tippett, L.H.C. 1931: "The Methods of Statistics". Williams & Norgate, London.

Examples

```
print("Strip and Boxplots - Example 2.1.2")

attach(cuckoos)
oldpar <- par(las = 2) # labels at right angle to axis.
stripchart(length ~ species)
boxplot(split(cuckoos$length, cuckoos$species),
        xlab="Length of egg", horizontal=TRUE)
detach(cuckoos)
```

```
par(oldpar)
pause()

print("Summaries - Example 2.2.2")
sapply(split(cuckoos$length, cuckoos$species), sd)
pause()

print("Example 4.1.4")
wren <- split(cuckoos$length, cuckoos$species)$wren
median(wren)
n <- length(wren)
sqrt(pi/2)*sd(wren)/sqrt(n)  # this s.e. computation assumes normality
```

CVbinary *Cross-Validation for Regression with a Binary Response*

Description

These functions give training (internal) and cross-validation measures of predictive accuracy for regression with a binary response. The data are randomly divided between a number of 'folds'. Each fold is removed, in turn, while the remaining data are used to re-fit the regression model and to predict at the omitted observations.

Usage

```
CVbinary(obj, rand=NULL, nfolds=10, print.details=TRUE)

cv.binary(obj, rand=NULL, nfolds=10, print.details=TRUE)
```

Arguments

`obj`	a `glm` object
`rand`	a vector which assigns each observation to a fold
`nfolds`	the number of folds
`print.details`	logical variable (TRUE = print detailed output, the default)

Value

`cvhat`	predicted values from cross-validation
`internal`	internal or (better) training predicted values
`training`	training predicted values
`acc.cv`	cross-validation estimate of accuracy
`acc.internal`	internal or (better) training estimate of accuracy
`acc.training`	training estimate of accuracy

Note

The term 'training' seems preferable to the term 'internal' in connection with predicted values, and the accuracy measure, that are based on the observations used to derive the model.

Author(s)

J.H. Maindonald

See Also

glm

Examples

```
frogs.glm <- glm(pres.abs ~ log(distance) + log(NoOfPools),
                 family=binomial,data=frogs)
CVbinary(frogs.glm)
mifem.glm <- glm(outcome ~ ., family=binomial, data=mifem)
CVbinary(mifem.glm)
```

CVlm — *Cross-Validation for Linear Regression*

Description

This function gives internal and cross-validation measures of predictive accuracy for multiple linear regression. (For binary logistic regression, use the CVbinary function.) The data are randomly assigned to a number of 'folds'. Each fold is removed, in turn, while the remaining data is used to re-fit the regression model and to predict at the deleted observations.

Usage

```
CVlm(data = DAAG::houseprices, form.lm = formula(sale.price ~ area),
             m = 3, dots = FALSE, seed = 29, plotit = c("Observed","Residual"),
              col.folds=NULL,
              main="Small symbols show cross-validation predicted values",
              legend.pos="topleft",
              printit = TRUE, ...)
cv.lm(data = DAAG::houseprices, form.lm = formula(sale.price ~ area),
             m = 3, dots = FALSE, seed = 29, plotit = c("Observed","Residual"),
              col.folds=NULL,
              main="Small symbols show cross-validation predicted values",
              legend.pos="topleft", printit = TRUE, ...)
```

Arguments

data	a data frame
form.lm	a formula or lm call or lm object
m	the number of folds
dots	uses pch=16 for the plotting character
seed	random number generator seed
plotit	This can be one of the text strings "Observed", "Residual", or a logical value. The logical TRUE is equivalent to "Observed", while FALSE is equivalent to "" (no plot)
col.folds	Per fold color settings
main	main title for graph
legend.pos	position of legend: one of "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right", "center".
printit	if TRUE, output is printed to the screen
...	Other arguments, to be passed through to the function legend()

Details

When plotit="Residual" and there is more than one explanatory variable, the fitted lines that are shown for the individual folds are approximations.

Value

The input data frame is returned, with additional columns Predicted (Predicted values using all observations) and cvpred (cross-validation predictions). The cross-validation residual sum of squares (ss) and degrees of freedom (df) are returned as attributes of the data frame.

Author(s)

J.H. Maindonald

See Also

lm, CVbinary

Examples

```
CVlm()
## Not run:
CVlm(data=nihills, form.lm=formula(log(time)~log(climb)+log(dist)),
          plotit="Observed")
CVlm(data=nihills, form.lm=formula(log(time)~log(climb)+log(dist)),
     plotit="Residual")
out <- CVlm(data=nihills, form.lm=formula(log(time)~log(climb)+log(dist)),
               plotit="Observed")
out[c("ms","df")]

## End(Not run)
```

DAAGtheme — *Function to generate lattice themes for graphs.*

Description

This generates themes for use in "A Practical Guide to Data Analysis Using R".

Usage

```
DAAGtheme(fontsize = list(text = 10, points = 6), box = "gray40", color=TRUE,
          sides = list(tck = 0.6, pad1 = 0.75, pad2 = 0.75),...)
```

Arguments

`fontsize`	Fontize for text and points. Specify as, e.g., `list(text = 10, points = 6)`.
`box`	Color for the panel and strip borders
`color`	Logical, determining whether graph will be colored or grayscale
`sides`	List, with elements `tck` (Tick length, as fraction of lattice default), and margin paddings `pad1`, `pad2`, `pad3`, and `pad4`. Margin paddings set the distance, in lines, from the tick marks to the tick labels.
`...`	Settings that will be passed to `simpleTheme()`.

Details

Setting the color of the bounding box and of the strip boxes to gray, which is the default, reduces the focus on them.

Value

A list which can be used as the `par.settings` argument to lattice graphics functions, or as the `theme` argument to `trellis.par.set()`.

Note

The code provides an example of the creation of a functions that generates themes that are tuned to specific user requirements. In this connection, see also `theEconomist.theme`.

Author(s)

John Maindonald.

See Also

`standard.theme`, `simpleTheme`, `theEconomist.theme`, `custom.theme`

Examples

```
bwtheme <- DAAGtheme(pch=2:4)
lattice::xyplot(csoa ~ age | target, groups=sex,
                data=DAAG::tinting, par.settings=bwtheme)
```

DAAGxdb *List, each of whose elements hold rows of a file, in character format*

Description

This is the default database for use with the function `datafile`, which uses elements of this list to place files in the working directory.

Usage

```
data(DAAGxdb)
```

Format

Successive elements in this list hold character vectors from which the corresponding files can be generated. The names of the list elements are fuel, fuel.csv, oneBadRow, scan-demo, molclock1, molclock2, and travelbooks.

Details

The files fuel.txt and fuel.csv are used in Chapter 1 of DAAGUR, while the files oneBadRow.txt and scan-demo.txt are used in Chapter 14 of DAAGUR.

References

Maindonald, J.H. and Braun, W.J. 2007. Data Analysis and Graphics Using R: An Example-Based Approach. 2nd edn, Cambridge University Press (DAAGUR).

Examples

```
data(DAAGxdb)
names(DAAGxdb)
```

datafile *Write an ASCII data file to the working directory.*

Description

Invoking this function writes one or more nominated files to the working directory. In particular, it may be used to write the files 'fuel.txt' and 'fuel.csv' that are used in Chapter 1 of DAAGUR, and the files 'oneBadRow.txt' and 'scan-demo.txt' that are used in Chapter 14 of DAAGUR.

Usage

```
datafile(file = c("fuel", "travelbooks"), datastore =
                 DAAG::DAAGxdb, altstore = DAAG::zzDAAGxdb, showNames =
                 FALSE)
```

Arguments

file	character; with the defaults for datastore and altstore the options are "fuel", for fuel.txt; "fuel.csv", for fuel.csv; "oneBadRow", for oneBadRow.txt; "scan-demo", for scan-demo.txt; "molclock1", for molclock1.txt; "molclock2", for molclock2.txt; "travelbooks", for travelbooks.txt; "bestTimes", for bestTimes.txt; "bostonc", for bostonc.txt
datastore	Each element of this list is a character vector that holds the rows of a file.
altstore	An alternative list. The default alternative list is used for the two files that are more than a few lines.
showNames	if TRUE, returns the names of available datasets.

Value

An ASCII file is output to the current working directory. The names of all available datasets are returned invisibly.

Author(s)

J.H. Maindonald

Examples

```
datafile(file="", showNames=TRUE)
```

dengue *Dengue prevalence, by administrative region*

Description

Data record, for each of 2000 administrative regions, whether or not dengue was recorded at any time between 1961 and 1990.

Usage

```
data(dengue)
```

Format

A data frame with 2000 observations on the following 13 variables.

humid Average vapour density: 1961-1990

humid90 90th percentile of `humid`

temp Average temperature: 1961-1990

temp90 90th percentile of `temp`

h10pix maximum of `humid`, within a 10 pixel radius

h10pix90 maximum of `humid90`, within a 10 pixel radius

trees Percent tree cover, from satellite data

trees90 90th percentile of `trees`

NoYes Was dengue observed? (1=yes)

Xmin minimum longitude

Xmax maximum longitude

Ymin minimum latitude

Ymax maximum latitude

Details

This is derived from a data set in which the climate and tree cover information were given for each half degree of latitude by half degreee of longitude pixel. The variable `NoYes` was given by administrative region. The climate data and tree cover data given here are 50th or 90th percentiles, where percetiles were calculates across pixels for an administrative region.

Source

Simon Hales, Environmental Research New Zealand Ltd.

References

Hales, S., de Wet, N., Maindonald, J. and Woodward, A. 2002. Potential effect of population and climate change global distribution of dengue fever: an empirical model. The Lancet 2002; 360: 830-34.

Examples

```
str(dengue)
glm(NoYes ~ humid, data=dengue, family=binomial)
glm(NoYes ~ humid90, data=dengue, family=binomial)
```

`dewpoint` *Dewpoint Data*

Description

The `dewpoint` data frame has 72 rows and 3 columns. Monthly data were obtained for a number of sites (in Australia) and a number of months.

Usage

```
dewpoint
```

Format

This data frame contains the following columns:

maxtemp monthly minimum temperatures

mintemp monthly maximum temperatures

dewpt monthly average dewpoint for each combination of minimum and maximum temperature readings (formerly dewpoint)

Source

Dr Edward Linacre, visiting fellow in the Australian National University Department of Geography.

Examples

```
print("Additive Model - Example 7.5")
require(splines)
attach(dewpoint)
ds.lm <- lm(dewpt ~ bs(maxtemp,5) + bs(mintemp,5), data=dewpoint)
ds.fit <-predict(ds.lm, type="terms", se=TRUE)
oldpar <- par(mfrow=c(1,2))
plot(maxtemp, ds.fit$fit[,1], xlab="Maximum temperature",
     ylab="Change from dewpoint mean",type="n")
lines(maxtemp,ds.fit$fit[,1])
lines(maxtemp,ds.fit$fit[,1]-2*ds.fit$se[,1],lty=2)
lines(maxtemp,ds.fit$fit[,1]+2*ds.fit$se[,1],lty=2)
plot(mintemp,ds.fit$fit[,2],xlab="Minimum temperature",
     ylab="Change from dewpoint mean",type="n")
ord<-order(mintemp)
lines(mintemp[ord],ds.fit$fit[ord,2])
lines(mintemp[ord],ds.fit$fit[ord,2]-2*ds.fit$se[ord,2],lty=2)
```

```
lines(mintemp[ord],ds.fit$fit[ord,2]+2*ds.fit$se[ord,2],lty=2)
detach(dewpoint)
par(oldpar)
```

droughts *Periods Between Rain Events*

Description

Data collected at Winnipeg International Airport (Canada) on periods (in days) between rain events.

Usage

```
droughts
```

Format

This data frame contains the following columns:

length the length of time from the completion of the last rain event to the beginning of the next rain event.

year the calendar year.

Examples

```
 boxplot(length ~ year, data=droughts)
 boxplot(log(length) ~ year, data=droughts)
 hist(droughts$length, main="Winnipeg Droughts", xlab="length (in days)")
hist(log(droughts$length), main="Winnipeg Droughts", xlab="length (in days, log scale)")
```

edcCO2 *EPICA Dome C Ice Core 800KYr Carbon Dioxide Data*

Description

Carbon dioxide record from the EPICA (European Project for Ice Coring in Antarctica) Dome C ice core covering 0 to 800 kyr BP.

Usage

```
data(edcCO2)
```

Format

A data frame with 1096 observations on the following 2 variables.

`age` Age in years before present (BP)

`co2` CO2 level (ppmv)

Details

Data are a composite series.

Source

Go to the url https://www.ncei.noaa.gov/products/paleoclimatology/ice-core/

References

Luthi, D., M. et al. 2008. High-resolution carbon dioxide concentration record 650,000-800,000 years before present. Nature, Vol. 453, pp. 379-382, 15 May 2008. doi:10.1038/nature06949

Indermuhle, A., E. et al, 1999, Atmospheric CO2 concentration from 60 to 20 kyr BP from the Taylor Dome ice core, Antarctica. Geophysical Research Letters, 27, 735-738.

Monnin, E., A. et al. 2001. Atmospheric CO2 concentrations over the last glacial termination. Science, Vol. 291, pp. 112-114.

Petit, J.R. et al. 1999. Climate and atmospheric history of the past 420,000 years from the Vostok ice core, Antarctica. Nature 399: 429-436.

Siegenthaler, U. et al. 2005. Stable Carbon Cycle-Climate Relationship During the Late Pleistocene. Science, v. 310 , pp. 1313-1317, 25 November 2005.

Examples

```
data(edcCO2)
```

---

edcT *EPICA Dome C Ice Core 800KYr Temperature Estimates*

Description

Temperature record, using Deuterium as a proxy, from the EPICA (European Project for Ice Coring in Antarctica) Dome C ice core covering 0 to 800 kyr BP.

Usage

```
data(edcT)
```

Format

A data frame with 5788 observations on the following 5 variables.

`Bag` Bag number

`ztop` Top depth (m)

`Age` Years before 1950

`Deuterium` Deuterium dD data

`dT` Temperature difference from the average of the last 1000 years ~ -54.5degC

Details

Temperature was estimated from the deuterium data, after making various corrections.

Source

Go to the url https://www.ncei.noaa.gov/products/paleoclimatology/ice-core/

References

Jouzel, J., et al. 2007. EPICA Dome C Ice Core 800KYr Deuterium Data and Temperature Estimates. IGBP PAGES/World Data Center for Paleoclimatology Data Contribution Series # 2007-091. NOAA/NCDC Paleoclimatology Program, Boulder CO, USA.

Jouzel, J., et al. 2007. Orbital and Millennial Antarctic Climate Variability over the Past 800,000 Years. Science, Vol. 317, No. 5839, pp.793-797, 10 August 2007.

Examples

```
data(edcT)
```

---

elastic1 *Elastic Band Data Replicated*

Description

Both datasets give, for each amount by which an elastic band is stretched over the end of a ruler, the distance that the band traveled when released. The `elastic1` data frame has 7 rows. The `elastic2` data frame, whose data span a wider range of stretches and distances, has 9 rows.

Usage

```
data(elastic1)
data(elastic2)
```

Format

These data frames contains the following columns:

stretch the amount by which the elastic band was stretched

distance the distance traveled

Source

J. H. Maindonald

Examples

```
plot(elastic1)
sapply(elastic1, mean)
pause()
sapply(elastic1, function(x)mean(x))
pause()
sapply(elastic1, function(x)sum(log(x)))
pause()
yrange <- range(c(elastic1$distance, elastic2$distance))
xrange <- range(c(elastic1$stretch, elastic2$stretch))
plot(distance ~ stretch, data = elastic1, pch = 16, ylim = yrange, xlim =
xrange)
points(distance ~ stretch, data = elastic2, pch = 15, col = 2)
legend(xrange[1], yrange[2], legend = c("Data set 1", "Data set 2"), pch =
c(16, 15), col = c(1, 2))
elastic1.lm <- lm(distance ~ stretch, data = elastic1)
elastic2.lm <- lm(distance ~ stretch, data = elastic2)
abline(elastic1.lm)
abline(elastic2.lm, col = 2)
summary(elastic1.lm)
summary(elastic2.lm)
pause()
predict(elastic1.lm, se.fit=TRUE)
predict(elastic2.lm, se.fit=TRUE)
```

`elasticband` *Elastic Band Data*

Description

The `elasticband` data frame has 7 rows and 2 columns giving, for each amount by which an elastic band is stretched over the end of a ruler, the distance that the band traveled when released.

Usage

```
elasticband
```

Format

This data frame contains the following columns:

stretch the amount by which the elastic band was stretched

distance the distance traveled

Source

J. H. Maindonald

Examples

```
print("Example 1.8.1")

attach(elasticband)     # R now knows where to find stretch and distance
plot(stretch, distance) # Alternative: plot(distance ~ stretch)
detach(elasticband)

print("Lists - Example 12.7")

elastic.lm <- lm(distance ~ stretch, data=elasticband)
 names(elastic.lm)
 elastic.lm$coefficients
elastic.lm[["coefficients"]]
pause()

elastic.lm[[1]]
pause()

elastic.lm[1]
pause()

options(digits=3)
elastic.lm$residuals
pause()

elastic.lm$call
pause()

 mode(elastic.lm$call)
```

---

errorsINseveral *Simulation of classical errors in x model, with multiple explanatory variables.*

Description

Simulates $y-$ and $x-$values for a classical "errors in x" linear regression model. One or more $x-$values are subject to random measurement error, independently of the corresponding covariate values that are measured without error.

Usage

```
errorsINseveral(n = 1000, a0 = 2.5, beta = c(1.5, 0), mu = 12.5, SDyerr = 0.5,
default.Vpar = list(SDx = 2, rho = -0.5, timesSDx = 1.5),
V = with(default.Vpar, matrix(c(1, rho, rho, 1), ncol = 2) * SDx^2),
xerrV = with(default.Vpar, matrix(c(1, 0, 0, 0), ncol = 2) * (SDx * timesSDx)^2),
parset = NULL, print.summary = TRUE, plotit = TRUE)
```

Arguments

`n`	Number of observations
`a0`	Intercept in linear regression model
`beta`	Regression coefficients. If one coefficient only is given, this will be repeated as many times as necessary
`mu`	Vector of covariate means.
`SDyerr`	SD of y, conditional on the covariates measured without error
`default.Vpar`	Parameters for the default model with two explanatory variables,
`V`	Variance-covariance matrix for the z's, measured without error. (These are generated from a multivariate normal distribution, mainly as a matter of convenience)
`xerrV`	Variance-covariance matrix for the added "errors in x"
`parset`	Parameter list (theme) in a form suitable for supplying to `trellis.par.set()`.
`print.summary`	If `TRUE`, print summary details of the regression results from the simulation.
`plotit`	If `TRUE`, plot the fitted values for the model with covariates with error, against the fitted values for covariates without error.

Details

With default arguments, simulates a model in which two covariates are in contention, the first measured without error, and the second with coefficient 0 in the model that includes both covariates measured without error.

Value

`ERRfree`	Data frame holding covariates without error, plus y
`addedERR`	Data frame holding covariates with error, plus y

Author(s)

John Maindonald

References

Data Analysis and Graphics Using R, 3rd edn, Section 6.8.1

See Also

`errorsINx`

Examples

```
library(lattice)
function(n=1000, a0=2.5, beta=c(1.5,0), mu=12.5, SDyerr=0.5,
           default.Vpar=list(SDx=2, rho=-0.5, timesSDx=1.5),
           V=with(default.Vpar, matrix(c(1,rho,rho,1), ncol=2)*SDx^2),
           xerrV=with(default.Vpar, matrix(c(1,0,0,0), ncol=2)*(SDx*timesSDx)^2),
           parset=NULL, print.summary=TRUE, plotit=TRUE){
    m <- dim(V)[1]
    if(length(mu)==1)mu <- rep(mu,m)
    ow <- options(warn=-1)
    xxmat <- sweep(matrix(rnorm(m*n, 0, 1), ncol=m) %*% chol(V), 2, mu, "+")
    errxx <- matrix(rnorm(m*n, 0, 1), ncol=m) %*% chol(xerrV, pivot=TRUE)
    options(ow)
    dimnames(xxmat)[[2]] <- paste("z", 1:m, sep="")
    xxWITHerr <- xxmat+errxx
    xxWITHerr <- data.frame(xxWITHerr)
    names(xxWITHerr) <- paste("xWITHerr", 1:m, sep="")
    xxWITHerr[, "y"] <- a0 + xxmat %*% matrix(beta,ncol=1) + rnorm(n, sd=SDyerr)
    err.lm <- lm(y ~ ., data=xxWITHerr)
    xx <- data.frame(xxmat)
    names(xx) <- paste("z", 1:m, sep="")
    xx$y <- xxWITHerr$y
    xx.lm <- lm(y ~ ., data=xx)
    B <- coef(err.lm)
    b <- coef(xx.lm)
    SE <- summary(err.lm)$coef[,2]
    se <- summary(xx.lm)$coef[,2]
    if(print.summary){
      beta0 <- c(mean(xx$y)-sum(beta*apply(xx[,1:m],2,mean)), beta)
      tab <- rbind(beta0, b, B)
      dimnames(tab) <- list(c("Values for simulation",
                              "Estimates: no error in x1",
                              "LS Estimates: error in x1"),
                            c("Intercept", paste("b", 1:m, sep="")))
      tabSE <- rbind(rep(NA,m+1),se,SE)
      rownames(tabSE) <- rownames(tab)
      colnames(tabSE) <- c("SE(Int)", paste("SE(", colnames(tab)[-1],")", sep=""))
      tab <- cbind(tab,tabSE)
      print(round(tab,3))
    }
    if(m==2 & print.summary){
      tau <- default.Vpar$timesSDx
      s1 <- sqrt(V[1,1])
      s2 <- sqrt(V[2,2])
      rho <- default.Vpar$rho
      s12 <- s1*sqrt(1-rho^2)
      lambda <- (1-rho^2)/(1-rho^2+tau^2)
      gam12 <- rho*sqrt(V[1,1]/V[2,2])
      expB2 <- beta[2]+beta[1]*(1-lambda)*gam12
      print(c("Theoretical attenuation of b1" = lambda, "Theoretical b2" = expB2))
    }
    if(is.null(parset))parset <- simpleTheme(col=c("gray40","gray40"),
```

```
                                          col.line=c("black","black"))
    if(plotit){
      library(lattice)
      zhat <- fitted(xx.lm)
      xhat <- fitted(err.lm)
      plt <- xyplot(xhat ~ zhat, aspect=1, scales=list(tck=0.5),
                    panel=function(x,y,...){
                      panel.xyplot(x,y,type="p",...)
                      panel.abline(lm(y ~ x), lty=2)
                      panel.abline(0,1)
                    },
                    xlab="Fitted values; regress on exact z",
                    ylab="Fitted values; regress on x = xWITHerr",
                    key=list(space="top", columns=2,
                      text=list(lab=c("Line y=x", "Regression fit to points")),
                      lines=list(lty=1:2)),
                    par.settings=parset
                    )
      print(plt)}
    invisible(list(ERRfree=xx, addedERR=xxWITHerr))
  }
```

---

errorsINx *Simulate data for straight line regression, with "errors in x".*

Description

Simulates $y-$ and $x-$values for the straight line regression model, but with $x-$values subject to random measurement error, following the classical “errors in x” model. Optionally, the x-values can be split into two groups, with one group shifted relative to the other

Usage

```
errorsINx(mu = 12.5, n = 200, a = 15, b = 1.5, SDx=2, SDyerr = 1.5,
          timesSDx=(1:5)/2.5, gpfactor=if(missing(gpdiff))FALSE else TRUE,
          gpdiff=if(gpfactor) 1.5 else 0, layout=NULL,
          parset = simpleTheme(alpha = 0.75, col = c("black","gray45"),
            col.line = c("black","gray45"), lwd=c(1,1.5), pch=c(1,2),
          lty=c(1,2)), print.summary=TRUE, plotit=TRUE, xrelation="same")
```

Arguments

mu	Mean of z
n	Number of points
a	Intercept in model where z is measured without error
b	Slope in model where z is measured without error
SDx	SD of z-values, measured without error

SDyerr	SD of error term in y where z is measured without error
timesSDx	SD of measurement error is timesSDx, as a multiple of SDx
gpfactor	Should x-values be split into two groups, with one shifted relative to the other?
gpdiff	Amount of shift of one group of z-values relative to the other
layout	Layout for lattice graph, if requested
parset	Parameters to be supplied to the lattice plot, if any
print.summary	Print summary information on fits?
plotit	logical: plot the data?
xrelation	character: sets the x-axis relation component of scales to "same" or "free" or (though this does not make make sense here) "sliced".

Details

The argument timesSDx can be a numeric vector. One set of x-values that are contaminated with measurement error is simulated for each element of timesSDx.

Value

gph	the trellis graphics object
mat	A matrix, with length(timesSDx)+2 columns. Values of z are in the first column. There is one further column (x with error) for each element of timesSDx, followed by a column for y. If there is a grouping variable, a further column identifies the groups.

Author(s)

John Maindonald

References

Data Analysis and Graphics Using R, 3rd edn, Section 6.7

Examples

```
library(lattice)
errorsINx()
errorsINx(gpdiff=2, timesSDx=1.25, SDyerr=2.5, n=80)
```

`excessRisk` *Create and analyze multiway frequency or weighted frequency table*

Description

This function creates a multi-way table of counts for the `response` given a set of classifying factors. Output facilitates a check on how the factor specified as `margin` may, after accounting for other classifying factors, affect the response.

Usage

```
excessRisk(form = weight ~ seatbelt + airbag, response = "dead", margin = "airbag",
data = DAAG::nassCDS, decpl = 4, printResults = TRUE)
```

Arguments

`form`	`form` is a formula in which classifying factors appear on the right, with an optional weight variable on the left.
`response`	`response` is a binary variable or two-level factor such that the response of interest is the relative number in the two levels.
`margin`	`margin` is the factor whose effect on the response, after accounting for other classifying factors, is of interest
`data`	`data` is a data frame in which variables and factors may be found
`decpl`	`decpl` is the number of decimal places in proportions that appear in the output
`printResults`	if `TRUE`, a tabular summary is printed.

Details

The best way to understand what this function does may be to run it with the default parameters, and/or with examples that appear below.

Value

The function returns a data frame, with one row for each combination of levels of factors on the right of the formula, but excluding the factor specified as `margin`. The final three columns show the count for level 1 as a fraction of the margin by total, the count for level 2 as a fraction of the margin by total, and the excess count for level 2 of response in the row, under the assumption that, in that row, there is no association between `response` and `margin`. This is the observed response (for the default arguments, number of dead) for level 2 (airbag deployed), less the number that would have been expected if the proportion had been that for level 1. (Negative values favor airbags.)

Author(s)

John Maindonald

References

See help(nassCDS)

See Also

xtabs

Examples

```
excessRisk()
excessRisk(weight ~ airbag+seatbelt+dvcat)
UCB <- as.data.frame.table(UCBAdmissions)
excessRisk(Freq~Gender, response="Admit", margin="Gender",data=UCB)
excessRisk(Freq~Gender+Dept, response="Admit", margin="Gender",data=UCB)
```

---

fossilfuel *Fossil Fuel Data*

Description

Estimates of total worldwide carbon emissions from fossil fuel use.

Usage

```
fossilfuel
```

Format

This data frame contains the following columns:

year a numeric vector giving the year the measurement was taken.

carbon a numeric vector giving the total worldwide carbon emissions from fossil fuel use, in millions of tonnes.

Details

Data for the years 1751 through to 2014 is available from Data for the years 2014 https://data.ess-dive.lbl.gov/portals/CDIAC

Source

Boden T A; Marland G; Andres R J (1999): Global, Regional, and National Fossil-Fuel CO2 Emissions (1751 - 2014) (V. 2017). Carbon Dioxide Information Analysis Center (CDIAC), Oak Ridge National Laboratory (ORNL), Oak Ridge, TN (United States), ESS-DIVE repository. Dataset. doi:10.3334/CDIAC/00001_V2017

Examples

```
plot(fossilfuel)
```

frogs *Frogs Data*

Description

The `frogs` data frame has 212 rows and 11 columns. The data are on the distribution of the Southern Corroboree frog, which occurs in the Snowy Mountains area of New South Wales, Australia.

Usage

```
frogs
```

Format

This data frame contains the following columns:

pres.abs 0 = frogs were absent, 1 = frogs were present

northing reference point

easting reference point

altitude altitude , in meters

distance distance in meters to nearest extant population

NoOfPools number of potential breeding pools

NoOfSites (number of potential breeding sites within a 2 km radius

avrain mean rainfall for Spring period

meanmin mean minimum Spring temperature

meanmax mean maximum Spring temperature

Source

Hunter, D. (2000) The conservation and demography of the southern corroboree frog (Pseudophryne corroboree). M.Sc. thesis, University of Canberra, Canberra.

Examples

```
print("Multiple Logistic Regression - Example 8.2")

plot(northing ~ easting, data=frogs, pch=c(1,16)[frogs$pres.abs+1],
  xlab="Meters east of reference point", ylab="Meters north")
pairs(frogs[,4:10])
attach(frogs)
pairs(cbind(altitude,log(distance),log(NoOfPools),NoOfSites),
  panel=panel.smooth, labels=c("altitude","log(distance)",
  "log(NoOfPools)","NoOfSites"))
detach(frogs)

frogs.glm0 <- glm(formula = pres.abs ~ altitude + log(distance) +
```

```
  log(NoOfPools) + NoOfSites + avrain + meanmin + meanmax,
  family = binomial, data = frogs)
summary(frogs.glm0)

frogs.glm <- glm(formula = pres.abs ~ log(distance) + log(NoOfPools) +
meanmin +
  meanmax, family = binomial, data = frogs)
oldpar <- par(mfrow=c(2,2))
termplot(frogs.glm, data=frogs)

termplot(frogs.glm, data=frogs, partial.resid=TRUE)

cv.binary(frogs.glm0)   # All explanatory variables
pause()

cv.binary(frogs.glm)    # Reduced set of explanatory variables

for (j in 1:4){
 rand <- sample(1:10, 212, replace=TRUE)
 all.acc <- cv.binary(frogs.glm0, rand=rand, print.details=FALSE)$acc.cv
 reduced.acc <- cv.binary(frogs.glm, rand=rand, print.details=FALSE)$acc.cv
 cat("\nAll:", round(all.acc,3), "  Reduced:", round(reduced.acc,3))
}
```

`frostedflakes` *Frosted Flakes data*

Description

The `frosted flakes` data frame has 100 rows and 2 columns giving the sugar concentration (in percent) for 25 g samples of a cereal as measured by 2 methods – high performance liquid chromatography (a slow accurate lab method) and a quick method using the infra-analyzer 400.

Usage

```
frostedflakes
```

Format

This data frame contains the following columns:

Lab careful laboratory analysis measurements using high performance liquid chromatography

IA400 measurements based on the infra-analyzer 400

Source

W. J. Braun

`fruitohms` *Electrical Resistance of Kiwi Fruit*

Description

Data are from a study that examined how the electrical resistance of a slab of kiwifruit changed with the apparent juice content.

Usage

```
fruitohms
```

Format

This data frame contains the following columns:

juice apparent juice content (percent)

ohms electrical resistance (in ohms)

Source

Harker, F. R. and Maindonald J.H. 1994. Ripening of nectarine fruit. *Plant Physiology* 106: 165 - 171.

Examples

```
plot(ohms ~ juice, xlab="Apparent juice content (%)",ylab="Resistance (ohms)", data=fruitohms)
lines(lowess(fruitohms$juice, fruitohms$ohms), lwd=2)
pause()

require(splines)
attach(fruitohms)
plot(ohms ~ juice, cex=0.8, xlab="Apparent juice content (%)",
     ylab="Resistance (ohms)", type="n")
fruit.lmb4 <- lm(ohms ~ bs(juice,4))
ord <- order(juice)
lines(juice[ord], fitted(fruit.lmb4)[ord], lwd=2)
ci <- predict(fruit.lmb4, interval="confidence")
lines(juice[ord], ci[ord,"lwr"])
lines(juice[ord], ci[ord,"upr"])
```

gaba *Effect of pentazocine on post-operative pain (average VAS scores)*

Description

The table shows, separately for males and females, the effect of pentazocine on post-operative pain profiles (average VAS scores), with (mbac and fbac) and without (mpl and fpl) preoperatively administered baclofen. Pain scores are recorded every 20 minutes, from 10 minutes to 170 minutes.

Usage

```
gaba
```

Format

A data frame with 9 observations on the following 7 variables.

`min` a numeric vector

`mbac` a numeric vector

`mpl` a numeric vector

`fbac` a numeric vector

`fpl` a numeric vector

`avbac` a numeric vector

`avplac` a numeric vector

Details

15 females were given baclofen, as against 3 males. 7 females received the placebo, as against 16 males. Averages for the two treatments (baclofen/placebo), taken over all trial participants and ignoring sex, are misleading.

Source

Gordon, N. C. et al.(1995): 'Enhancement of Morphine Analgesia by the GABA_B against Baclofen'. *Neuroscience* 69: 345-349.

Examples

```
data(gaba)
mr <- range(gaba$min)
tran <- range(gaba[, c("mbac","mpl","fbac","fpl")])
## Means by treatment and sex
par(mfrow=c(1,2))
plot(mr, tran, xlab = "Time post pentazocine (min)",
     ylab = "Reduction in VAS pain rating",
     type = "n", xlim = c(0, 170), ylim = tran)
points(gaba$min, gaba$fbac, pch = 1, col = 8, lwd = 2, lty = 2,
```

```
        type = "b")
points(gaba$min, gaba$fpl, pch = 0, col = 8, lwd = 2, lty = 2,
       type = "b")
points(gaba$min, gaba$mbac, pch = 16, col = 8, lty = 2, type = "b")
points(gaba$min, gaba$mpl, pch = 15, col = 8, lty = 2, type = "b")
box()
## Now plot means, by treatment, averaged over all participants
plot(mr, tran, xlab = "Time post pentazocine (min)",
     ylab = "Reduction in VAS pain rating",
     type = "n", xlim = c(0, 170), ylim = tran)
bac <- (15 * gaba$fbac + 3 * gaba$mbac)/18
plac <- (7 * gaba$fpl + 9 * gaba$mpl)/16
points(gaba$min, plac, pch = 15, lty = 1, col=1, type = "b")
points(gaba$min, bac, pch = 16, lty = 1, col=1, type = "b")
box()
par(mfrow=c(1,1))
```

---

geophones *Seismic Timing Data*

Description

The geophones data frame has 56 rows and 2 columns. Thickness of a layer of Alberta substratum as measured by a line of geophones.

Usage

```
geophones
```

Format

This data frame contains the following columns:

distance location of geophone.

thickness time for signal to pass through substratum.

Examples

```
plot(geophones)
lines(lowess(geophones, f=.25))
```

greatLakes *Yearly averages of Great Lake heights: 1918 - 2009*

Description

Heights, stored as a multivariate time series, are for the lakes Erie, Michigan/Huron, Ontario and St Clair

Usage

```
data(greatLakes)
```

Format

The format is: mts [1:92, 1:4] 174 174 174 174 174 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr [1:4] "Erie" "michHuron" "Ontario" "StClair" - attr(*, "tsp")= num [1:3] 1918 2009 1 - attr(*, "class")= chr [1:2] "mts" "ts"

Details

For more details, go to the website that is the source of the data.

Source

https://www.lre.usace.army.mil/Missions/Great-Lakes-Information/Great-Lakes-Information-2/Water-Level-Data/

Examples

```
data(greatLakes)
plot(greatLakes)
## maybe str(greatLakes)
```

grog *Alcohol consumption in Australia and New Zealand*

Description

Data are annual apparent alcohol consumption in Australia and New Zealand, in liters of pure alcohol content per annum, separately for beer, wine, and spirits (including spirit-based products).

Usage

```
data(grog)
```

Format

A data frame with 18 observations on the following 5 variables.

`Beer` liters per annum

`Wine` liters per annum

`Spirit` liters per annum

`Country` a factor with levels `Australia` `NewZealand`

`Year` Year ending in June of the given year

Details

Data are total available pure alcohol content, for the three categories, divided by numbers of persons aged 15 years or more. The source data for New Zealand included quarterly figures from December 1997, and annual data to December for all years. The annual New Zealand figure to June 1998 required an estimate for September 1997 that was obtained by extrapolating back the third quarter trend line from later years.

Source

Australian data are from https://www.abs.gov.au. For the most recent data, go to https://www.abs.gov.au/statistics/health/health-conditions-and-risks/apparent-consumption-alcohol-australia For New Zealand data, go to https://infoshare.stats.govt.nz/ Click on 'Industry sectors' and then on 'Alcohol Available for Consumption - ALC'.

Examples

```
data(grog)
library(lattice)
xyplot(Beer+Wine+Spirit ~ Year | Country, data=grog)
xyplot(Beer+Wine+Spirit ~ Year, groups=Country, data=grog, outer=TRUE)
```

---

hardcopy *Graphical Output for Hardcopy*

Description

This function streamlines graphical output to the screen, pdf or ps files. File names for hard copy devices can be generated automatically from function names of the form `g3.2` or `fig3.2` (the choice of alphabetic characters prior to `3.2` is immaterial).

Usage

```
hardcopy(width = 3.75, height = 3.75, color = FALSE, trellis = FALSE,
                  device = c("", "pdf", "ps"), path = getwd(), file =
                  NULL, format = c("nn-nn", "name"), split = "\\.",
                  pointsize = c(8, 4), fonts=NULL, horiz = FALSE, ...)
```

Arguments

`width`	width of plot in inches (sic!)
`height`	height of plot in inches (sic!)
`color`	(lattice plots only) TRUE if plot is not black on white only
`trellis`	TRUE if plot uses trellis graphics
`device`	screen `""`, pdf or ps
`path`	external path name
`file`	name of file to hold output, else `NULL`
`format`	Alternatives are `"nn-nn"` and `"name"`.
`split`	character on which to split function name (file=NULL)
`pointsize`	Pointsize. For trellis devices a vector of length 2 giving font sizes for text and for points respectively
`fonts`	For postscript devices, specify families that will be used in addition to the intial device
`horiz`	FALSE for landscape mode; applies only to postscript files
`...`	Other arguments for passing to the `pdf` or `postscript`

Details

If a file name (`file`, without extension) is not supplied, the `format` argument determines how the name is constructed. With `format="name"`, the function name is used. With `format="nn-nn"` and `dotsplit` unchanged from the default, a function name of the form g3.1 leads to the name `03-01`. Here `g` can be replaced by any other non-numeric characters; the result is the same. The relevant extension is in any case added.

Value

Graphical output to screen, pdf or ps file.

Author(s)

J.H. Maindonald

See Also

`postscript`

`headInjury` *Minor Head Injury (Simulated) Data*

Description

The `headInjury` data frame has 3121 rows and 11 columns. The data were simulated according to a simple logistic regression model to match roughly the clinical characteristics of a sample of individuals who suffered minor head injuries.

Usage

```
headInjury
```

Format

This data frame contains the following columns:

age.65 age factor (0 = under 65, 1 = over 65).

amnesia.before amnesia before impact (less than 30 minutes = 0, more than 30 minutes =1).

basal.skull.fracture (0 = no fracture, 1 = fracture).

GCS.decrease Glasgow Coma Scale decrease (0 = no deterioration, 1 = deterioration).

GCS.13 initial Glasgow Coma Scale (0 = not '13', 1 = '13').

GCS.15.2hours Glasgow Coma Scale after 2 hours (0 = not '15', 1 = '15').

high.risk assessed by clinician as high risk for neurological intervention (0 = not high risk, 1 = high risk).

loss.of.consciousness (0 = conscious, 1 = loss of consciousness).

open.skull.fracture (0 = no fracture, 1 = fracture)

vomiting (0 = no vomiting, 1 = vomiting)

clinically.important.brain.injury any acute brain finding revealed on CT (0 = not present, 1 = present).

References

Stiell, I.G., Wells, G.A., Vandemheen, K., Clement, C., Lesiuk, H., Laupacis, A., McKnight, R.D., Verbee, R., Brison, R., Cass, D., Eisenhauer, M., Greenberg, G.H., and Worthington, J. (2001) The Canadian CT Head Rule for Patients with Minor Head Injury, The Lancet. 357: 1391-1396.

`hills` *Scottish Hill Races Data*

Description

The record times in 1984 (`hills`) for 35 Scottish hill races, or in 2000 (`hills2000`) for 56 hill races. The `hills2000` dataset is the subset of `races2000` for which `type` is `hill`.

Usage

```
data(hills)
data(hills2000)
```

Format

dist distance, in miles (on the map)

climb total height gained during the route, in feet

time record time in hours

timef record time in hours for females, in the `hills2000` dataset.

Source

A.C. Atkinson (1986) Comment: Aspects of diagnostic regression analysis. Statistical Science 1, 397-402.

Also, in MASS library, with time in minutes.

The Scottish Running Resource, http://www.hillrunning.co.uk

References

A.C. Atkinson (1988) Transformations unmasked. Technometrics 30, 311-318. ["corrects" the time for Knock Hill, in the `hills` dataset, from 78.65 to 18.65. It is unclear if this based on the original records.]

Examples

```
print("Transformation - Example 6.4.3")
pairs(hills, labels=c("dist\n\n(miles)", "climb\n\n(feet)",
"time\n\n(hours)"))
pause()

pairs(log(hills), labels=c("dist\n\n(log(miles))", "climb\n\n(log(feet))",
  "time\n\n(log(hours))"))
pause()

hills0.loglm <- lm(log(time) ~ log(dist) + log(climb), data = hills)
oldpar <- par(mfrow=c(2,2))
```

```
plot(hills0.loglm)
pause()


hills.loglm <- lm(log(time) ~ log(dist) + log(climb), data = hills[-18,])
summary(hills.loglm)
plot(hills.loglm)
pause()

hills2.loglm <- lm(log(time) ~ log(dist)+log(climb)+log(dist):log(climb),
data=hills[-18,])
anova(hills.loglm, hills2.loglm)
pause()

step(hills2.loglm)
pause()

summary(hills.loglm, corr=TRUE)$coef
pause()

summary(hills2.loglm, corr=TRUE)$coef
par(oldpar)
pause()

print("Nonlinear - Example 6.9.4")
hills.nls0 <- nls(time ~ (dist^alpha)*(climb^beta), start =
   c(alpha = .909, beta = .260), data = hills[-18,])
summary(hills.nls0)
plot(residuals(hills.nls0) ~ predict(hills.nls0)) # residual plot
pause()

hills$climb.mi <- hills$climb/5280
hills.nls <- nls(time ~ alpha + beta*dist + gamma*(climb.mi^delta),
  start=c(alpha = 1, beta = 1, gamma = 1, delta = 1), data=hills[-18,])
summary(hills.nls)
plot(residuals(hills.nls) ~ predict(hills.nls)) # residual plot
```

---

hotspots *Hawaian island chain hotspot Potassium-Argon ages*

Description

K-Ar Ages (millions of years) and distances (km) from Kilauea along the trend of the chain of Hawaian volcanic islands and other seamounts that are believed to have been created by a moving "hot spot". The age of Kilauea is given as 0-0.4 Ma.

Usage

```
data(hotspots)
```

Format

A data frame with 36 observations on the following 6 variables.

`ID` Volcano identifier

`name` Name

`distance` Distance in kilometers

`age` K-Ar age in millions of years

`error` Standard error of estimate?

`source` Data source; see information on web site below.

Details

For details of the way that errors werre calculated, refer to the original papers. See also the comments under `hotspots2006`. In general, errors do not account for geological uncertainty.

Source

http://www.soest.hawaii.edu/GG/HCV/haw_formation.html

Examples

```
data(hotspots)
plot(age ~ distance, data=hotspots)
abline(lm(age ~ distance, data=hotspots))
```

`hotspots2006` *Hawaian island chain hotspot Argon-Argon ages*

Description

Ar-Ar Ages (millions of years) and distances (km) from Kilauea along the trend of the chain of Hawaian volcanic islands and other seamounts that are believed to have been created by a moving "hot spot".

Usage

```
data(hotspots2006)
```

Format

A data frame with 10 observations on the following 6 variables.

`age` Ar-Ar age

`CI95lim` Measurement error; 95% CI

`geoErr` Geological Uncertainty

`totplus` Total uncertainty (+)

`totminus` Total uncertainty (-)

`distance` Distance in kilometers

Details

Note that measurement error is small relative to geological uncertainty. Geological uncertainty arises because lavas are likely to have erupted, over a period of up to 2 million years, somewhat after passage over the hot spot's centre. Dredging or drilling will in general have accessed larvas from the younger half of this interval. Hence the asymmetry in the geological uncertainty.

Source

Warren D. Sharp and David A. Clague, 50-Ma initiation of Hawaiian-Emperor bend records major change in Pacific Plate motion. Science 313: 1281-1284 (2006).

Examples

```
data(hotspots2006)
```

---

houseprices *Aranda House Prices*

Description

The `houseprices` data frame consists of the floor area, price, and the number of bedrooms for a sample of houses sold in Aranda in 1999. Aranda is a suburb of Canberra, Australia.

Usage

```
houseprices
```

Format

This data frame contains the following columns:

area a numeric vector giving the floor area

bedrooms a numeric vector giving the number of bedrooms

sale.price a numeric vector giving the sale price in thousands of Australian dollars

Source

J.H. Maindonald

Examples

```
plot(sale.price~area, data=houseprices)
pause()

coplot(sale.price~area|bedrooms, data=houseprices)
pause()

print("Cross-Validation - Example 5.5.2")
```

```
houseprices.lm <- lm(sale.price ~ area, data=houseprices)
summary(houseprices.lm)$sigma^2
pause()

CVlm()
pause()

print("Bootstrapping - Example 5.5.3")
houseprices.fn <- function (houseprices, index){
house.resample <- houseprices[index,]
house.lm <- lm(sale.price ~ area, data=house.resample)
coef(house.lm)[2]    # slope estimate for resampled data
}
require(boot)       # ensure that the boot package is loaded
houseprices.boot <- boot(houseprices, R=999, statistic=houseprices.fn)

houseprices1.fn <- function (houseprices, index){
house.resample <- houseprices[index,]
house.lm <- lm(sale.price ~ area, data=house.resample)
predict(house.lm, newdata=data.frame(area=1200))
}

houseprices1.boot <- boot(houseprices, R=999, statistic=houseprices1.fn)
boot.ci(houseprices1.boot, type="perc") # "basic" is an alternative to "perc"
houseprices2.fn <- function (houseprices, index){
house.resample <- houseprices[index,]
house.lm <- lm(sale.price ~ area, data=house.resample)
houseprices$sale.price-predict(house.lm, houseprices)  # resampled prediction errors
}

n <- length(houseprices$area)
R <- 200
houseprices2.boot <- boot(houseprices, R=R, statistic=houseprices2.fn)
house.fac <- factor(rep(1:n, rep(R, n)))
plot(house.fac, as.vector(houseprices2.boot$t), ylab="Prediction Errors",
xlab="House")
pause()

plot(apply(houseprices2.boot$t,2, sd)/predict.lm(houseprices.lm, se.fit=TRUE)$se.fit,
     ylab="Ratio of Bootstrap SE's to Model-Based SE's", xlab="House", pch=16)
abline(1,0)
```

---

humanpower *Oxygen uptake versus mechanical power, for humans*

Description

Data are from Daedalus project; see the reference below.

Usage

```
data(humanpower1)
```

Format

A data frame with 28 observations on the following 3 variables.

wattsPerKg a numeric vector: watts per kilogram of body weight

o2 a numeric vector: ml/min/kg

id a factor with levels 1 - 5 (`humanpower1`) or 1 - 4 (`humanpower2`), identifying the different athletes

Details

Data in `humanpower1` are from investigations (Bussolari 1987) designed to assess the feasibility of a proposed 119 kilometer human powered flight from the island of Crete – in the initial phase of the Daedalus project. Data are for five athletes – a female hockey player, a male amateur tri-athlete, a female amateur triathlete, a male wrestler and a male cyclist – who were selected from volunteers who were recruited through the news media, Data in `humanpower2) are for four out of the 25 applicants who were selected for further testing, in the lead-up to the eventual selection of a pilot for the Daedalus project (Nadel and Bussolari 1988).`

Source

Bussolari, S.R.(1987). Human factors of long-distance human-powered aircraft flights. Human Power 5: 8-12.

Nadel and Bussolari, S.R.(1988). The Daedalus project: physiological problems and solutions. American Scientist 76: 351-360.

References

Nadel and Bussolari, S.R.(1989). The physiological limits of long-duration human-power production – lessons learned from the Daedalus project. Human Power 7: 7-10.

Examples

```
str(humanpower1)
plot(humanpower1)
lm(o2 ~ id + wattsPerKg:id, data=humanpower1)
lm(o2 ~ id + wattsPerKg:id, data=humanpower2)
```

`hurricNamed` *Named US Atlantic Hurricanes*

Description

Details are given of atmospheric pressure at landfall, estimated damage in millions of dollars, and deaths, for named hurricanes that made landfall in the US mainland from 1950 through to 2012.

Usage

```
data("hurricNamed")
```

Format

A data frame with 94 observations on the following 11 variables.

`Name` Hurricane name

`Year` Numeric

`LF.WindsMPH` Maximum sustained windspeed (>= 1 minute) to occur along the US coast. Prior to 1980, this is estimated from the maximum windspeed associated with the Saffir-Simpson index at landfall. If 2 or more landfalls, the maximum is taken

`LF.PressureMB` Atmospheric pressure at landfall in millibars. If 2 or more landfalls, the minimum is taken

`LF.times` Date of first landfall

`BaseDam2014` Property damage (millions of 2014 US dollars)

`BaseDamage` Property damage (in millions of dollars for that year)

`NDAM2014` Damage, had hurricane appeared in 2014

`AffectedStates` Affected states (2-digit abbreviations), pasted together

`firstLF` Date of first landfall

`deaths` Number of continental US direct and indirect deaths

`mf` Gender of name; a factor with levels `f` `m`

Details

An earlier version of these data was the subject of a controversial paper that claimed to have found that hurricanes with female names, presumably because taken less seriously, did more human damage after adjusting for the severity of the storm than those with male names.

Source

https://www.icat.com/storms/catastrophe-resources Deaths except for Audrey and Katrina, are in the Excel file that is available from the url https://www.pnas.org/doi/10.1073/pnas.1402786111

NOAA Monthly Weather Reports (MWRs) supplied the numbers of deaths for all except Donna, Celia, Audrey and Katrina. The figure for Celia is from https://www.nhc.noaa.gov/pdf/NWS-TPC-5.pdf. For the other three hurricanes, it is from the Atlantic hurricane list in Wikipedia (see the references.)

References

https://www.aoml.noaa.gov/hrd/hurdat/mwr_pdf/ https://en.wikipedia.org/wiki/List_of_Atlantic_hurricanes https://www.nhtsa.gov/file-downloads

Jung, Kiju, et al. "Female hurricanes are deadlier than male hurricanes." Proceedings of the National Academy of Sciences 111.24 (2014): 8782-8787.

Examples

```
data(hurricNamed)
str(hurricNamed)
plot(log(deaths+0.5) ~ log(NDAM2014), data=hurricNamed)
with(hurricNamed, lines(lowess(log(deaths+0.5) ~ log(NDAM2014))))
plot(log(deaths+0.5) ~ I(NDAM2014^0.14), data=hurricNamed)
with(hurricNamed, lines(lowess(log(deaths+0.1) ~ I(NDAM2014^0.14))))
```

---

intersalt — *Blood pressure versus Salt; inter-population data*

Description

Median blood pressure, as a fuction of salt intake, for each of 52 human populations.

Usage

```
intersalt
```

Format

A data frame with 52 observations on the following 4 variables.

`b` a numeric vector

`bp` mean diastolic blood pressure (mm Hg)

`na` mean sodium excretion (mmol/24h)

`country` a character vector

Details

For each population took a sample of 25 males and 25 females from each decade in the age range 20 - 50, i.e. 200 individuals in all.

Source

Intersalt Cooperative Research Group. 1988. Intersalt: an international study of electrolyte excretion and blood pressure: results for 24 hour urinary sodium and potassium excretion. *British Medical Journal* 297: 319-328.

References

Maindonald, J.H. *The Design of Research Studies ? A Statistical Perspective*, viii + 109pp. Graduate School Occasional Paper 00/2, Australian National University 2000.

Examples

```
data(intersalt)
plot(bp ~ na, data=intersalt, xlab="Median sodium excretion (mmol/24h)",
     ylab="Median diatoluc blood pressure (mm Hg)")
```

---

ironslag *Iron Content Measurements*

Description

The `ironslag` data frame has 53 rows and 2 columns. Two methods for measuring the iron content in samples of slag were compared, a chemical and a magnetic method. The chemical method requires greater effort than the magnetic method.

Usage

```
ironslag
```

Format

This data frame contains the following columns:

chemical a numeric vector containing the measurements coming from the chemical method

magnetic a numeric vector containing the measurments coming from the magnetic method

Source

Hand, D.J., Daly, F., McConway, K., Lunn, D., and Ostrowski, E. eds (1993) A Handbook of Small Data Sets. London: Chapman & Hall.

Examples

```
iron.lm <- lm(chemical ~ magnetic, data = ironslag)
oldpar <- par(mfrow = c(2,2))
plot(iron.lm)
par(oldpar)
```

jobs *Canadian Labour Force Summary Data (1995-96)*

Description

The number of workers in the Canadian labour force broken down by region (BC, Alberta, Prairies, Ontario, Quebec, Atlantic) for the 24-month period from January, 1995 to December, 1996 (a time when Canada was emerging from a deep economic recession).

Usage

```
jobs
```

Format

This data frame contains the following columns:

BC monthly labour force counts in British Columbia

Alberta monthly labour force counts in Alberta

Prairies monthly labour force counts in Saskatchewan and Manitoba

Ontario monthly labour force counts in Ontario

Quebec monthly labour force counts in Quebec

Atlantic monthly labour force counts in Newfoundland, Nova Scotia, Prince Edward Island and New Brunswick

Date year (in decimal form)

Details

These data have been seasonally adjusted.

Source

Statistics Canada

Examples

```
print("Multiple Variables and Times - Example 2.1.4")
sapply(jobs, range)
pause()

matplot(jobs[,7], jobs[,-7], type="l", xlim=c(95,97.1))
 # Notice that we have been able to use a data frame as the second argument to matplot().
 # For more information on matplot(), type help(matplot)
text(rep(jobs[24,7], 6), jobs[24,1:6], names(jobs)[1:6], adj=0)
pause()

sapply(log(jobs[,-7]), range)
```

```
apply(sapply(log(jobs[,-7]), range), 2, diff)
pause()

oldpar <- par(mfrow=c(2,3))
range.log <- sapply(log(jobs[,-7], 2), range)
maxdiff <- max(apply(range.log, 2, diff))
range.log[2,] <- range.log[1,] + maxdiff
titles <- c("BC Jobs","Alberta Jobs","Prairie Jobs",
   "Ontario Jobs", "Quebec Jobs", "Atlantic Jobs")
for (i in 1:6){
plot(jobs$Date, log(jobs[,i], 2), type = "l", ylim = range.log[,i],
    xlab = "Time", ylab = "Number of jobs", main = titles[i])
}
par(oldpar)
```

---

kiwishade *Kiwi Shading Data*

Description

The kiwishade data frame has 48 rows and 4 columns. The data are from a designed experiment that compared different kiwifruit shading treatments. There are four vines in each plot, and four plots (one for each of four treatments: none, Aug2Dec, Dec2Feb, and Feb2May) in each of three blocks (locations: west, north, east). Each plot has the same number of vines, each block has the same number of plots, with each treatment occurring the same number of times.

Usage

kiwishade

Format

This data frame contains the following columns:

yield Total yield (in kg)

plot a factor with levels east.Aug2Dec, east.Dec2Feb, east.Feb2May, east.none, north.Aug2Dec, north.Dec2Feb, north.Feb2May, north.none, west.Aug2Dec, west.Dec2Feb, west.Feb2May, west.none

block a factor indicating the location of the plot with levels east, north, west

shade a factor representing the period for which the experimenter placed shading over the vines; with levels: none no shading, Aug2Dec August - December, Dec2Feb December - February, Feb2May February - May

Details

The northernmost plots were grouped together because they were similarly affected by shading from the sun in the north. For the remaining two blocks shelter effects, whether from the west or from the east, were thought more important.

Source

Snelgar, W.P., Manson. P.J., Martin, P.J. 1992. Influence of time of shading on flowering and yield of kiwifruit vines. Journal of Horticultural Science 67: 481-487.

References

Maindonald J H 1992. Statistical design, analysis and presentation issues. New Zealand Journal of Agricultural Research 35: 121-141.

Examples

```
print("Data Summary - Example 2.2.1")
attach(kiwishade)
kiwimeans <- aggregate(yield, by=list(block, shade), mean)
names(kiwimeans) <- c("block","shade","meanyield")

kiwimeans[1:4,]
pause()

print("Multilevel Design - Example 9.3")
kiwishade.aov <- aov(yield ~ shade+Error(block/shade),data=kiwishade)
summary(kiwishade.aov)
pause()


sapply(split(yield, shade), mean)

pause()

kiwi.table <- t(sapply(split(yield, plot), as.vector))
kiwi.means <- sapply(split(yield, plot), mean)
kiwi.means.table <- matrix(rep(kiwi.means,4), nrow=12, ncol=4)
kiwi.summary <- data.frame(kiwi.means, kiwi.table-kiwi.means.table)
names(kiwi.summary)<- c("Mean", "Vine 1", "Vine 2", "Vine 3", "Vine 4")
kiwi.summary
mean(kiwi.means) # the grand mean (only for balanced design)

if(require(lme4, quietly=TRUE)) {
kiwishade.lmer <- lmer(yield ~ shade + (1|block) + (1|block:plot),
                       data=kiwishade)
## block:shade is an alternative to block:plot

kiwishade.lmer


##                  Residuals and estimated effects
library(lattice)
xyplot(residuals(kiwishade.lmer) ~ fitted(kiwishade.lmer)|block,
                data=kiwishade, groups=shade,
                layout=c(3,1), par.strip.text=list(cex=1.0),
                xlab="Fitted values (Treatment + block + plot effects)",
                ylab="Residuals", pch=1:4, grid=TRUE,
```

```
              scales=list(x=list(alternating=FALSE), tck=0.5),
              key=list(space="top", points=list(pch=1:4),
                       text=list(labels=levels(kiwishade$shade)),columns=4))
ploteff <- ranef(kiwishade.lmer, drop=TRUE)[[1]]
qqmath(ploteff, xlab="Normal quantiles", ylab="Plot effect estimates",
       scales=list(tck=0.5))
}
```

`leafshape` *Full Leaf Shape Data Set*

Description

Leaf length, width and petiole measurements taken at various sites worldwide. The `leafshape17` data frame is the subset that has data for North Queensland sites.

Usage

```
data(leafshape)
data(leafshape17)
```

Format

This data frame contains the following columns:

bladelen leaf length (in mm)

petiole a numeric vector

bladewid leaf width (in mm)

latitude latitude

logwid natural logarithm of width

logpet logarithm of petiole

loglen logarithm of length

arch leaf architecture (0 = plagiotropic, 1 = orthotropic

location a factor with levels `Sabah`, `Panama`, `Costa Rica`, `N Queensland`, `S Queensland`, `Tasmania`

Source

King, D.A. and Maindonald, J.H. 1999. Tree architecture in relation to leaf dimensions and tree stature in temperate and tropical rain forests. Journal of Ecology 87: 1012-1024.

Examples

```
library(MASS)
leaf17.lda <- lda(arch ~ logwid+loglen, data=leafshape17)
leaf17.hat <- predict(leaf17.lda)
leaf17.lda
 table(leafshape17$arch, leaf17.hat$class)
pause()

tab <- table(leafshape17$arch, leaf17.hat$class)
 sum(tab[row(tab)==col(tab)])/sum(tab)
leaf17cv.lda <- lda(arch ~ logwid+loglen, data=leafshape17, CV=TRUE)
tab <- table(leafshape17$arch, leaf17cv.lda$class)
pause()

leaf17.glm <- glm(arch ~ logwid + loglen, family=binomial, data=leafshape17)
 options(digits=3)
summary(leaf17.glm)$coef
pause()

leaf17.one <- cv.binary(leaf17.glm)
table(leafshape17$arch, round(leaf17.one$internal))     # Resubstitution
pause()

table(leafshape17$arch, round(leaf17.one$cv))           # Cross-validation
```

`leaftemp` *Leaf and Air Temperature Data*

Description

Data are measurements of vapour pressure and of the difference between leaf and air temperature.

Usage

```
leaftemp
```

Format

This data frame contains the following columns:

CO2level Carbon Dioxide level `low`, `medium`, `high`

vapPress Vapour pressure

tempDiff Difference between leaf and air temperature

BtempDiff a numeric vector

Source

Katharina Siebke and Susan von Cammerer, Australian National University.

Examples

```
print("Fitting Multiple Lines - Example 7.3")

leaf.lm1 <- lm(tempDiff ~ 1 , data = leaftemp)
leaf.lm2 <- lm(tempDiff ~ vapPress, data = leaftemp)
leaf.lm3 <- lm(tempDiff ~ CO2level + vapPress, data = leaftemp)
leaf.lm4 <- lm(tempDiff ~ CO2level + vapPress + vapPress:CO2level,
  data = leaftemp)

anova(leaf.lm1, leaf.lm2, leaf.lm3, leaf.lm4)

summary(leaf.lm2)
plot(leaf.lm2)
```

---

leaftemp.all — *Full Leaf and Air Temperature Data Set*

Description

The `leaftemp.all` data frame has 62 rows and 9 columns.

Usage

```
leaftemp.all
```

Format

This data frame contains the following columns:

glasshouse a factor with levels `A`, `B`, `C`

CO2level a factor with Carbon Dioxide Levels: `high`, `low`, `medium`

day a factor

light a numeric vector

CO2 a numeric vector

tempDiff Difference between Leaf and Air Temperature

BtempDiff a numeric vector

airTemp Air Temperature

vapPress Vapour Pressure

Source

J.H. Maindonald

litters *Mouse Litters*

Description

Data on the body and brain weights of 20 mice, together with the size of the litter. Two mice were taken from each litter size.

Usage

```
litters
```

Format

This data frame contains the following columns:

lsize litter size

bodywt body weight

brainwt brain weight

Source

Wainright P, Pelkman C and Wahlsten D 1989. The quantitative relationship between nutritional effects on preweaning growth and behavioral development in mice. Developmental Psychobiology 22: 183-193.

Examples

```
print("Multiple Regression - Example 6.2")

pairs(litters, labels=c("lsize\n\n(litter size)", "bodywt\n\n(Body Weight)",
                        "brainwt\n\n(Brain Weight)"))
  # pairs(litters) gives a scatterplot matrix with less adequate labeling

mice1.lm <- lm(brainwt ~ lsize, data = litters) # Regress on lsize
mice2.lm <- lm(brainwt ~ bodywt, data = litters) #Regress on bodywt
mice12.lm <- lm(brainwt ~ lsize + bodywt, data = litters) # Regress on lsize & bodywt

summary(mice1.lm)$coef # Similarly for other coefficients.
# results are consistent with the biological concept of brain sparing

pause()

hat(model.matrix(mice12.lm))  # hat diagonal
pause()

plot(lm.influence(mice12.lm)$hat, residuals(mice12.lm))

print("Diagnostics - Example 6.3")
```

```
mice12.lm <- lm(brainwt ~ bodywt+lsize, data=litters)
oldpar <-par(mfrow = c(1,2))
bx <- mice12.lm$coef[2]; bz <- mice12.lm$coef[3]
res <- residuals(mice12.lm)
plot(litters$bodywt, bx*litters$bodywt+res, xlab="Body weight",
  ylab="Component + Residual")
panel.smooth(litters$bodywt, bx*litters$bodywt+res) # Overlay
plot(litters$lsize, bz*litters$lsize+res, xlab="Litter size",
  ylab="Component + Residual")
panel.smooth(litters$lsize, bz*litters$lsize+res)
par(oldpar)
```

---

lmdiags — *Return data required for diagnostic plots*

Description

This extracts the code that provides the major part of the statistical information used by plot.lm, leaving out the code used to provide the graphs

Usage

```
lmdiags(x, which = c(1L:3L, 5L), cook.levels = c(0.5, 1), hii=NULL)
```

Arguments

`x`	This must be an object of class `lm` object, or that inherits from an object of class `lm`.
`which`	a subset of the numbers '1:6', indicating the plots for which statistical information is required
`cook.levels`	Levels for contours of `cook.levels`, by default `c(0.5,1)`
`hii`	Diagonal elements for the hat matrix. If not supplied (`hii=NULL`), they will be calculated from the argument `x`.

Details

See `plot.lm` for additional information.

Value

`yh`	fitted values
`rs`	standardized residuals (for `glm` models standardized deviance residuals)
`yhn0`	As yh, but omitting observations with zero weight
`cook`	Cook's statistics
`rsp`	standardized residuals (for `glm` models standardized Pearson residuals)

Note

This function is designed, in the first place, for use in connection with plotSimDiags, used to give simulations of diagnostic plots for lm objects.

Author(s)

John Maindonald, using code that John Maindonald, Martin Maechler and others had contributed to plot.lm

References

See references for plot.lm

See Also

plotSimDiags, plot.lm

Examples

```
women.lm <- lm(weight ~ height, data=women)
veclist <- lmdiags(x=women.lm)
## Returns the statistics that are required for graphs 1, 2, 3, and 5
```

---

logisticsim — *Simple Logistic Regression Data Simulator*

Description

This function simulates simple regression data from a logistic model.

Usage

```
logisticsim(x = seq(0, 1, length=101), a = 2, b = -4, seed=NULL)
```

Arguments

x	a numeric vector representing the explanatory variable
a	the regression function intercept
b	the regression function slope
seed	numeric constant

Value

a list consisting of

x	the explanatory variable vector
y	the Poisson response vector

Examples

```
logisticsim()
```

`Lottario` *Ontario Lottery Data*

Description

The data frame `Lottario` is a summary of 122 weekly draws of an Ontario lottery, beginning in November, 1978. Each draw consists of 7 numbered balls, drawn without replacement from an urn consisting of balls numbered from 1 through 39.

Usage

```
Lottario
```

Format

This data frame contains the following columns:

Number the integers from 1 to 39, representing the numbered balls

Frequency the number of occurrences of each numbered ball

Source

The Ontario Lottery Corporation

References

Bellhouse, D.R. (1982). Fair is fair: new rules for Canadian lotteries. Canadian Public Policy - Analyse de Politiques 8: 311-320.

Examples

```
order(Lottario$Frequency)[33:39]  # the 7 most frequently chosen numbers
```

`lung` *Cape Fur Seal Lung Measurements*

Description

The `lung` vector consists of weight measurements of lungs taken from 30 Cape Fur Seals that died as an unintended consequence of commercial fishing.

Usage

```
lung
```

Manitoba.lakes *The Nine Largest Lakes in Manitoba*

Description

The Manitoba.lakes data frame has 9 rows and 2 columns. The areas and elevations of the nine largest lakes in Manitoba, Canada. The geography of Manitoba (a relatively flat province) can be divided crudely into three main areas: a very flat prairie in the south which is at a relatively high elevation, a middle region consisting of mainly of forest and Precambrian rock, and a northern region which drains more rapidly into Hudson Bay. All water in Manitoba, which does not evaporate, eventually drains into Hudson Bay.

Usage

```
Manitoba.lakes
```

Format

This data frame contains the following columns:

elevation a numeric vector consisting of the elevations of the lakes (in meters)

area a numeric vector consisting of the areas of the lakes (in square kilometers)

Source

The CANSIM data base at Statistics Canada.

Examples

```
plot(Manitoba.lakes)
plot(Manitoba.lakes[-1,])
```

mdbAVtJtoD *Murray-Darling basin monthly temperatures*

Description

Australian Murray-Darling basin monthly temperatures

Usage

```
data("mdbAVtJtoD")
```

Format

The format is: Time-Series [1:867] from 1950 to 2022: 27.44 26.84 24.4 22.27 8.41 ...

Source

Australian Bureau of Meteorology web pages:

Go to the url http://www.bom.gov.au/climate/change/, choose timeseries to display, then click "Download data"

The website gives anomalies from 1961-1990 averages. The monthly means have been added, in order to obtain a series. The monthly means are shown along with plots for the individual months.

Examples

```
data(mdbAVtJtoD)
plot(window(mdbAVtJtoD, start=c(2000,1)), ylab="Mean monthly data")
```

---

measles *Deaths in London from measles*

Description

Deaths in London from measles: 1629 – 1939, with gaps.

Usage

```
data(measles)
```

Format

The format is: Time-Series [1:311] from 1629 to 1939: 42 2 3 80 21 33 27 12 NA NA ...

Source

Guy, W. A. 1882. Two hundred and fifty years of small pox in London. Journal of the Royal Statistical Society 399-443.

Stocks, P. 1942. Measles and whooping cough during the dispersal of 1939-1940. Journal of the Royal Statistical Society 105:259-291.

References

Lancaster, H. O. 1990. Expectations of Life. Springer.

medExpenses — *Family Medical Expenses*

Description

The `medExpenses` data frame contains average weekly medical expenses including drugs for 33 families randomly sampled from a community of 600 families which contained 2700 individuals. These data were collected in the 1970's at an unknown location.

Usage

```
medExpenses
```

Format

familysize number of individuals in a family

expenses average weekly cost for medical expenses per family member

Examples

```
with(medExpenses, weighted.mean(expenses, familysize))
```

mignonette — *Darwin's Wild Mignonette Data*

Description

Data compare the heights of crossed plants with self-fertilized plants of the wild mignonette *reseda lutea*. Plants were paired within the pots in which they were grown, with one on one side and one on the other.

Usage

```
mignonette
```

Format

This data frame contains the following columns:

cross heights of the crossed plants

self heights of the self-fertilized plants

Source

Darwin, Charles. 1877. The Effects of Cross and Self Fertilisation in the Vegetable Kingdom. Appleton and Company, New York, page 118.

Examples

```
print("Is Pairing Helpful? - Example 4.3.1")

attach(mignonette)
plot(cross ~ self, pch=rep(c(4,1), c(3,12))); abline(0,1)
abline(mean(cross-self), 1, lty=2)
detach(mignonette)
```

---

milk *Milk Sweetness Study*

Description

The milk data frame has 17 rows and 2 columns. Each of 17 panelists compared two milk samples for sweetness.

Usage

```
milk
```

Format

This data frame contains the following columns:

four a numeric vector consisting of the assessments for four units of additive

one a numeric vector while the is the assessment for one unit of additive

Source

J.H. Maindonald

Examples

```
print("Rug Plot - Example 1.8.1")
xyrange <- range(milk)
plot(four ~ one, data = milk, xlim = xyrange, ylim = xyrange, pch = 16)
rug(milk$one)
rug(milk$four, side = 2)
abline(0, 1)
```

`modelcars` *Model Car Data*

Description

The `modelcars` data frame has 12 rows and 2 columns. The data are for an experiment in which a model car was released three times at each of four different distances up a 20 degree ramp. The experimenter recorded distances traveled from the bottom of the ramp across a concrete floor.

Usage

```
modelcars
```

Format

This data frame contains the following columns:

distance.traveled a numeric vector consisting of the lengths traveled (in cm)

starting.point a numeric vector consisting of the distance of the starting point from the top of the ramp (in cm)

Source

W.J. Braun

Examples

```
plot(modelcars)
modelcars.lm <- lm(distance.traveled ~ starting.point, data=modelcars)
aov(modelcars.lm)
pause()

print("Response Curves - Example 4.6")
attach(modelcars)
stripchart(distance.traveled ~ starting.point, vertical=TRUE, pch=15,
           xlab = "Distance up ramp", ylab="Distance traveled")
detach(modelcars)
```

monica *WHO Monica Data*

Description

The monica data frame has 6357 rows and 12 columns. The dataset mifem (1295 rows) is the subset that has data for females.

Usage

```
data(monica)
data(mifem)
```

Format

Columns are:

outcome mortality outcome, a factor with levels live, dead

age age at onset

sex m = male, f = female

hosp y = hospitalized, n = not hospitalized

yronset year of onset

premi previous myocardial infarction event, a factor with levels y, n, nk not known

smstat smoking status, a factor with levels c current, x ex-smoker, n non-smoker, nk not known

diabetes a factor with levels y, n, nk not known

highbp high blood pressure, a factor with levels y, n, nk not known

hichol high cholesterol, a factor with levels y, n nk not known

angina a factor with levels y, n, nk not known

stroke a factor with levels y, n, nk not known

Source

Newcastle (Australia) centre of the Monica project; see the web site http://www.ktl.fi/monica

Examples

```
print("CART - Example 10.7")
summary(monica)
pause()

library(rpart)
monica.rpart <- rpart(outcome ~ ., data = monica, cp = 0.0025)
plotcp(monica.rpart)
printcp(monica.rpart)
```

```
pause()
monicab.rpart <- prune(monica.rpart, cp=0.006)
print(monicab.rpart)
summary(mifem)
pause()
mifem.rpart <- rpart(outcome ~ ., data = mifem, cp = 0.0025)
plotcp(mifem.rpart)
printcp(mifem.rpart)
pause()

mifemb.rpart <- prune(mifem.rpart, cp=0.006)
print(mifemb.rpart)
```

---

moths *Moths Data*

Description

The moths data frame has 41 rows and 4 columns. These data are from a study of the effect of habitat on the densities of two species of moth (A and P). Transects were set across the search area. Within transects, sections were identified according to habitat type.

Usage

```
moths
```

Format

This data frame contains the following columns:

meters length of transect

A number of type A moths found

P number of type P moths found

habitat a factor with levels Bank, Disturbed, Lowerside, NEsoak, NWsoak, SEsoak, SWsoak, Upperside

Source

Sharyn Wragg, formerly of Australian National University

Examples

```
print("Quasi Poisson Regression - Example 8.3")
rbind(table(moths[,4]), sapply(split(moths[,-4], moths$habitat), apply,2,
sum))
A.glm <- glm(formula = A ~ log(meters) + factor(habitat), family =
quasipoisson, data = moths)
summary(A.glm)
  # Note the huge standard errors
```

```
moths$habitat <- relevel(moths$habitat, ref="Lowerside")
A.glm <- glm(A ~ habitat + log(meters), family=quasipoisson, data=moths)
summary(A.glm)$coef
## Consider as another possibility
A2.glm <- glm(formula = A ~ sqrt(meters) + factor(habitat), family =
                  quasipoisson(link=sqrt), data = moths)
summary(A2.glm)
```

`multilap` *Data Filtering Function*

Description

A subset of data is selected for which the treatment to control ratio of non-binary covariates is never outside a specified range.

Usage

```
multilap(df = DAAG::nsw74psid1, maxf = 20, colnames = c("educ",
                  "age", "re74", "re75", "re78"))
```

Arguments

`df`	a data frame
`maxf`	filtering parameter
`colnames`	columns to be compared for filtering

Author(s)

J.H. Maindonald

`nassCDS` *Airbag and other influences on accident fatalities*

Description

US data, for 1997-2002, from police-reported car crashes in which there is a harmful event (people or property), and from which at least one vehicle was towed. Data are restricted to front-seat occupants, include only a subset of the variables recorded, and are restricted in other ways also.

Usage

```
nassCDS
```

Format

A data frame with 26217 observations on the following 15 variables.

dvcat ordered factor with levels (estimated impact speeds) `1-9km/h`, `10-24`, `25-39`, `40-54`, `55+`

weight Observation weights, albeit of uncertain accuracy, designed to account for varying sampling probabilities.

dead factor with levels `alive` `dead`

airbag a factor with levels `none` `airbag`

seatbelt a factor with levels `none` `belted`

frontal a numeric vector; 0 = non-frontal, 1=frontal impact

sex a factor with levels `f` `m`

ageOFocc age of occupant in years

yearacc year of accident

yearVeh Year of model of vehicle; a numeric vector

abcat Did one or more (driver or passenger) airbag(s) deploy? This factor has levels `deploy` `nodeploy` `unavail`

occRole a factor with levels `driver` `pass`

deploy a numeric vector: 0 if an airbag was unavailable or did not deploy; 1 if one or more bags deployed.

injSeverity a numeric vector; 0:none, 1:possible injury, 2:no incapacity, 3:incapacity, 4:killed; 5:unknown, 6:prior death

caseid character, created by pasting together the populations sampling unit, the case number, and the vehicle number. Within each year, use this to uniquely identify the vehicle.

Details

Data collection used a multi-stage probabilistic sampling scheme. The observation weight, called national inflation factor (`national`) in the data from NASS, is the inverse of an estimate of the selection probability. These data include a subset of the variables from the NASS dataset. Variables that are coded here as factors are coded as numeric values in that dataset.

Source

https://www.stat.colostate.edu/~meyer/airbags.htm\ https://www.nhtsa.gov/file-downloads

See also\ https://maths-people.anu.edu.au/~johnm/datasets/airbags/

References

Meyer, M.C. and Finney, T. (2005): *Who wants airbags?*. Chance 18:3-16.

Farmer, C.H. 2006. *Another look at Meyer and Finney's 'Who wants airbags?'*. Chance 19:15-22.

Meyer, M.C. 2006. *Commentary on "Another look at Meyer and Finney's 'Who wants airbags?'*. Chance 19:23-24.

For analyses based on the alternative FARS (Fatal Accident Recording System) data, and associated commentary, see:

Cummings, P; McKnight, B, 2010. *Accounting for vehicle, crash, and occupant characteristics in traffic crash studies.* Injury Prevention 16: 363-366. [The relatively definitive analyses in this paper use a matched cohort design,

Olson, CM; Cummings, P, Rivara, FP, 2006. *Association of first- and second-generation air bags with front occupant death in car crashes: a matched cohort study.* Am J Epidemiol 164:161-169. [The relatively definitive analyses in this paper use a matched cohort design, using data taken from the FARS (Fatal Accident Recording System) database.]

Braver, ER; Shardell, M; Teoh, ER, 2010. *How have changes in air bag designs affected frontal crash mortality?* Ann Epidemiol 20:499-510.

The web page https://www-fars.nhtsa.dot.gov/Main/index.aspx has a menu-based interface into the FARS (Fatality Analysis Recording System) data. The FARS database aims to include every accident in which there was at least one fatality.

Examples

```
data(nassCDS)
xtabs(weight ~ dead + airbag, data=nassCDS)
xtabs(weight ~ dead + airbag + seatbelt + dvcat, data=nassCDS)
tab <- xtabs(weight ~ dead + abcat, data=nassCDS,
             subset=dvcat=="25-39"&frontal==0)[, c(3,1,2)]
round(tab[2, ]/apply(tab,2,sum)*100,2)
```

`nasshead` *Documentation of names of columns in nass9702cor*

Description

`SASname` and `longname` are from the SAS XPT file nass9702cor.XPT that is available from the website noted below. The name `shortname` is the name used in the data frame `nass9702cor`, not included in this package, but available from my website that is noted below. It is also used in `nassCDS`, for columns that `nassCDS` includes.

Usage

```
data(nasshead)
```

Format

A data frame with 56 observations on the following 3 variables.

shortname a character vector

SASname a character vector

longname a character vector

Details

For full details of the coding of values in columns of `nass9702cor`, consult one of the SAS format files that can be obtained by following the instructions on Dr Meyer's web site that is noted below.

Source

https://www.stat.colostate.edu/~meyer/airbags.htm\https://www.nhtsa.gov/file-downloads\

See also https://maths-people.anu.edu.au/~johnm/datasets/airbags/

References

Meyer, M.C. and Finney, T. (2005): *Who wants airbags?*. Chance 18:3-16.

Farmer, C.H. 2006. *Another look at Meyer and Finney's 'Who wants airbags?'*. Chance 19:15-22.

Meyer, M.C. 2006. *Commentary on "Another look at Meyer and Finney's 'Who wants airbags?'"*. Chance 19:23-24.

Examples

```
data(nasshead)
```

---

nihills *Record times for Northern Ireland mountain running events*

Description

Data were from the 2007 calendar for the Northern Ireland Mountain Running Association.

Usage

```
data(nihills)
data(lognihills)
```

Format

A data frame with 23 observations on the following 4 variables.

`dist` distances in miles

`climb` amount of climb in feet

`time` record time in hours for males

`timef` record time in hours for females

`logdist` distances, log(miles)

`logclimb` climb, log(feet)

`logtime` record time for males, log(hours)

`logtimef` record time for females, log(hours)

Details

These data make an interesting comparison with the dataset `hills2000` in the DAAG package.

Source

For more recent information, see https://www.nimra.org.uk/index.php/fixtures/

Examples

```
data(nihills)
lm(formula = log(time) ~ log(dist) + log(climb), data = nihills)
lm(formula = log(time) ~ log(dist) + log(climb/dist), data = nihills)
lm(formula = logtime ~ logdist + I(logclimb-logdist), data = lognihills)
```

---

nsw74demo *Labour Training Evaluation Data*

Description

This nsw74demo data frame, with 445 rows and 10 columns, is the subset of the nswdemo dataset for which 1974 earnings are available. Data are for the male experimental control and treatment groups, in an investigation of the effect of training on changes, between 1974-1975 and 1978, in the earnings of individuals who had experienced employment difficulties.

Likewise, nsw74psid1 (2675 rows) is the subset of the nswpsid1 data, and nsw74psid3 (313 rows) is the subset of the nswpsid3 data, for which 1974 income is available. NB, also, the nsw74psidA data set.

Usage

```
data(nsw74demo)
data(nsw74psid1)
data(nsw74psid3)
data(nsw74psidA)
```

Format

Columns are:

trt a numeric vector identifying the study in which the subjects were enrolled (0 = PSID, 1 = NSW).

age age (in years).

educ years of education.

black (0 = not black, 1 = black).

hisp (0 = not hispanic, 1 = hispanic).

marr (0 = not married, 1 = married).

nodeg (0 = completed high school, 1 = dropout).

re74 real earnings in 1974.

re75 real earnings in 1975.

re78 real earnings in 1978.

Details

The `nsw74psidA` data set (252 rows) was obtained from `nsw74psid1` using:

```
here <- age <= 40 & re74<=5000 & re75 <= 5000 & re78 < 30000
nsw74psidA <- nsw74psid1[here, ]
```

Source

http://www.columbia.edu/~rd247/nswdata.html

References

Dehejia, R.H. and Wahba, S. 1999. Causal effects in non-experimental studies: re-evaluating the evaluation of training programs. Journal of the American Statistical Association 94: 1053-1062.

Lalonde, R. 1986. Evaluating the economic evaluations of training programs. American Economic Review 76: 604-620.

`nswdemo` *Labour Training Evaluation Data*

Description

The `nswdemo` data frame contains 722 rows and 10 columns. These data are pertinent to an investigation of the way that earnings changed, between 1974-1975 and 1978, for an experimental treatment who were given job training as compared with a control group who did not receive such training.

The `psid1` data set is an alternative non-experimental "control" group. `psid2` and `psid3` are subsets of `psid1`, designed to be better matched to the experimental data than `psid1`. Note also the `cps1`, `cps2` and `cps3` datasets (**DAAGxtras**) that have been proposed as non-experimental controls.

Usage

```
data(nswdemo)
```

Format

This data frame contains the following columns:

trt a numeric vector identifying the study in which the subjects were enrolled (0 = Control, 1 = treated).

age age (in years).

educ years of education.

black (0 = not black, 1 = black).

hisp (0 = not hispanic, 1 = hispanic).

marr (0 = not married, 1 = married).

nodeg (0 = completed high school, 1 = dropout).

re74 real earnings in 1974.

re75 real earnings in 1975.

re78 real earnings in 1978.

Source

https://users.nber.org/~rdehejia/nswdata.html

References

Dehejia, R.H. and Wahba, S. 1999. Causal effects in non-experimental studies: re-evaluating the evaluation of training programs. Journal of the American Statistical Association 94: 1053-1062.

Lalonde, R. 1986. Evaluating the economic evaluations of training programs. American Economic Review 76: 604-620.

Smith, J. A. and Todd, P.E. 2005,"Does Matching overcome. LaLonde?s critique of nonexperimental estimators", *Journal of Econometrics* 125: 305-353.

Dehejia, R.H. 2005. Practical propensity score matching: a reply to Smith and Todd. *Journal of Econometrics* 125: 355-364.

See Also

psid1, psid2, psid3

---

nswpsid1 — *Labour Training Evaluation Data*

Description

The cps1 (15992 rows) and psid1 (2490 rows) datasets are from non-experimental "control" groups, used in various studies of the effect of a labor training program, alternative to the experimental control group in nswdemo. The cps2 (2369 rows) and cps3 (429 rows) subsets of cps1 are designed to be better matched to the experimental data than cps1. Likewise, psid2 (253 rows) and psid3 (128 rows) are subsets of psid1 that are designed to be better matched to the experimental data than psid1. The nswpsid1 dataset (2675 rows) combines the experimental treatment group in nswdemo with the psid1 control data from the Panel Study of Income Dynamics (PSID) study.

Usage

```
data(psid1)
data(nswpsid1)
```

Format

Columns are:

trt a numeric vector identifying the study in which the subjects were enrolled (0 = Control, 1 = treated).

age age (in years).

educ years of education.

black (0 = not black, 1 = black).

hisp (0 = not hispanic, 1 = hispanic).

marr (0 = not married, 1 = married).

nodeg (0 = completed high school, 1 = dropout).

re74 real earnings in 1974.

re75 real earnings in 1975.

re78 real earnings in 1978.

Details

The `cps1` and `psid1` data sets are two non-experimental "control" groups, alternative to that in `nswdemo`, used in investigating whether use of such a non-experimental control group can be satisfactory. `cps2` and `cps3` are subsets of `cps1`, designed to be better matched to the experimental data than `cps1`. Similary `psid2` and `psid3` are subsets of `psid1`, designed to be better matched to the experimental data than `psid1`. `nswpsid1` combines data for the experimental treatment group in `nswdemo` with the `psid1` control data from the Panel Study of Income Dynamics (PSID) study.

Source

https://users.nber.org/~rdehejia/nswdata.html

References

Dehejia, R.H. and Wahba, S. 1999. Causal effects in non-experimental studies: re-evaluating the evaluation of training programs. *Journal of the American Statistical Association* 94: 1053-1062.

Lalonde, R. 1986. Evaluating the economic evaluations of training programs. *American Economic Review* 76: 604-620.

Smith, J. A. and Todd, P.E. "Does Matching overcome. LaLonde?s critique of nonexperimental estimators", *Journal of Econometrics* 125: 305-353.

Dehejia, R.H. 2005. Practical propensity score matching: a reply to Smith and Todd. *Journal of Econometrics* 125: 355-364.

obounce *Bounce - obsolete*

Description

A utility function for oneway.plot

Author(s)

J.H. Maindonald

oddbooks *Measurements on 12 books*

Description

Data giving thickness (mm), height (cm), width (cm) and weight (g), of 12 books. Books were selected so that thickness decreased as page area increased

Usage

```
data(oddbooks)
```

Format

A data frame with 12 observations on the following 4 variables.

thick a numeric vector

height a numeric vector

breadth a numeric vector

weight a numeric vector

Source

JM took books from his library.

Examples

```
data(oddbooks)
str(oddbooks)
plot(oddbooks)
```

onesamp *Paired Sample t-test*

Description

This function performs a t-test for the mean difference for paired data, and produces a scatterplot of one column against the other column, showing whether there was any benefit to using the paired design.

Usage

```
onesamp(dset, x="unsprayed", y="sprayed", xlab=NULL, ylab=NULL,
        dubious=NULL, conv=NULL, dig=2, ...)
```

Arguments

`dset`	a matrix or dataframe having two columns
`x`	name of column to play the role of the 'predictor'
`y`	name of column to play the role of the 'response'
`xlab`	horizontal axis label
`ylab`	vertical axis label
`dubious`	vector of logical (`FALSE`/`TRUE`) values, specifying points that are to be omitted
`conv`	scaling factor that should be applied to data
`dig`	round SE to this number of digits for dispplay on graph
`...`	Further arguments, to be passed to `plot()`

Value

A scatterplot of `y` against `x` together with estimates of standard errors and standard errors of the difference (`y-x`).

Also produced is a confidence interval and p-value for the test.

Author(s)

J.H. Maindonald

Examples

```
onesamp(dset = pair65, x = "ambient", y = "heated", xlab =
        "Amount of stretch (ambient)", ylab =
        "Amount of stretch (heated)")
```

`onet.permutation` *One Sample Permutation t-test*

Description

This function computes the p-value for the one sample t-test using a permutation test. The permutation density can also be plotted.

Usage

```
onet.permutation(x = DAAG::pair65$heated - DAAG::pair65$ambient, nsim = 2000,
plotit = TRUE)
```

Arguments

`x`	a numeric vector containing the sample values (centered at the null hypothesis value)
`nsim`	the number of permutations (randomly selected)
`plotit`	if TRUE, the permutation density is plotted

Value

The p-value for the test of the hypothesis that the mean of `x` differs from 0

Author(s)

J.H. Maindonald

References

Good, P. 2000. Permutation Tests. Springer, New York.

Examples

```
onet.permutation()
```

`onetPermutation` *One Sample Permutation t-test*

Description

This function computes the p-value for the one sample t-test using a permutation test. The permutation density can also be plotted.

Usage

```
onetPermutation(x = DAAG::pair65$heated - DAAG::pair65$ambient, nsim = 2000,
plotit = TRUE)
```

Arguments

`x`	a numeric vector containing the sample values (centered at the null hypothesis value)
`nsim`	the number of permutations (randomly selected)
`plotit`	if TRUE, the permutation density is plotted

Details

This function calculates only a one-sided p-value. The `EnvStats::oneSamplePermutationTest` in the **EnvStats** package offers a choice between two-sided and one-sided tests. If the **statmod** package is available, a correction will be applied that accounts for a small bias that results when permutations are sampled.

Value

The p-value for the test of the hypothesis that the mean of `x` differs from 0

Author(s)

J.H. Maindonald

References

Good, P. 2000. Permutation Tests. Springer, New York.

Examples

```
onetPermutation()
```

onewayPlot *Display of One Way Analysis Results*

Description

A line plot of estimates for unstructured comparison of factor levels

Usage

```
onewayPlot(obj, trtnam = "trt", axisht = 4.5, xlim = NULL,
    xlab = NULL, lsdht = 1.5, hsdht = 0.5, textht = axisht -
        2.5, oma = rep(1, 4), angle = 80, alpha = 0.05)
oneway.plot(obj, trtnam = "trt", axisht = 4.5, xlim = NULL,
    xlab = NULL, lsdht = 1.5, hsdht = 0.5, textht = axisht -
        2.5, oma = rep(1, 4), angle = 80, alpha = 0.05)
```

Arguments

obj	One way analysis of variance object (from aov)
trtnam	name of factor for which line plot is required
axisht	Axis height
xlim	Range on horizontal axis
xlab	Horizontal axis label
lsdht	Height adjustment parameter for display of LSD
hsdht	Height adjustment parameter for display of Tukey's HSD
textht	Height of text
oma	Outer margin area
angle	Text angle (in degrees)
alpha	Test size

Value

Estimates, labeled with level names, are set out along a line

Author(s)

J.H. Maindonald

Examples

```
rice.aov <- aov(ShootDryMass ~ trt, data=rice)
onewayPlot(obj=rice.aov)
```

`orings` *Challenger O-rings Data*

Description

Record of the number and type of O-ring failures prior to the tragic Challenger mission in January, 1986.

Usage

```
orings
```

Format

This data frame contains the following columns:

Temperature O-ring temperature for each test firing or actual launch of the shuttle rocket engine

Erosion Number of erosion incidents

Blowby Number of blowby incidents

Total Total number of incidents

Source

Presidential Commission on the Space Shuttle Challenger Accident, Vol. 1, 1986: 129-131.

References

Tufte, E. R. 1997. Visual Explanations. Graphics Press, Cheshire, Connecticut, U.S.A.

Examples

```
oldpar <- par(mfrow=c(1,2))
plot(Total~Temperature, data = orings[c(1,2,4,11,13,18),]) # the
               # observations included in the pre-launch charts
plot(Total~Temperature, data = orings)
par(oldpar)
```

overlapDensity *Overlapping Density Plots*

Description

Densities for two distinct samples are estimated and plotted.

Usage

```
overlapDensity(x0, x1, ratio = c(0.05, 20), ratio.number = FALSE,
       plotvalues = c("Density", "Numbers"), gpnames = c("Control", "Treatment"),
          cutoffs = c(lower = TRUE, upper = TRUE), bw = FALSE,
          xlab = "Score", ylab = NULL,
          col = 1:2, lty = 1:2, lwd = c(1, 1), ...)

overlap.density(x0, x1, ratio = c(0.05, 20), ratio.number = FALSE,
       plotvalues = c("Density", "Numbers"), gpnames = c("Control", "Treatment"),
          cutoffs = c(lower = TRUE, upper = TRUE), bw = FALSE,
          xlab = "Score", ylab = NULL,
          col = 1:2, lty = 1:2, lwd = c(1, 1), ...)
```

Arguments

`x0`	control group measurements
`x1`	treatment group measurements
`ratio`	if not `NULL`, the range within which the relative number per unit interval (`ratio.number=TRUE`) or relative probability density (`ratio.number=FALSE`) of observations from the two groups are required to lie will be used to determine lower and upper bounds on the values of `x0` and `x1`. [The relative numbers at any point are estimated from (density1*n1)/(density0*x0)]
`ratio.number`	If TRUE (default), then `ratio` is taken as the ratio of number of points per unit interval
`plotvalues`	If set to `Number` then the y-axis scale is chosen so that total area undere the curve is equal to the sample size; otherwise (`plotvalues="Density"`) total area under each cueve is 1. Any other setting does not give a plot.
`gpnames`	Names of the two samples
`cutoffs`	logical vector, indicating whether density estimates should be truncated below (lower=TRUE) or above (upper=TRUE)
`bw`	logical, indicates whether to overwrite with a gray scale plot
`xlab`	Label for x-axis
`ylab`	Label for y-axis
`col`	standard color parameter
`lty`	standard line type preference
`lwd`	standard line width preference
`...`	Other parameters to be passed to `plot()`

Author(s)

J.H. Maindonald

See Also

`t.test`

Examples

```
attach(two65)
overlapDensity(ambient,heated)
t.test(ambient,heated)
```

`ozone` *Ozone Data*

Description

Monthly provisional mean total ozone (in Dobson units) at Halley Bay (approximately corrected to Bass-Paur).

Usage

```
ozone
```

Format

This data frame contains the following columns:

Year the year

Aug August mean total ozone

Sep September mean total ozone

Oct October mean total ozone

Nov November mean total ozone

Dec December mean total ozone

Jan January mean total ozone

Feb February mean total ozone

Mar March mean total ozone

Apr April mean total ozone

Annual Yearly mean total ozone

Source

Shanklin, J. (2001) Ozone at Halley, Rothera and Vernadsky/Faraday.

http://www.antarctica.ac.uk/met/jds/ozone/data/zoz5699.dat

References

Christie, M. (2000) The Ozone Layer: a Philosophy of Science Perspective. Cambridge University Press.

Examples

```
AnnualOzone <- ts(ozone$Annual, start=1956)
plot(AnnualOzone)
```

---

pair65 *Heated Elastic Bands*

Description

The pair65 data frame has 9 rows and 2 columns. Eighteen elastic bands were divided into nine pairs, with bands of similar stretchiness placed in the same pair. One member of each pair was placed in hot water (60-65 degrees C) for four minutes, while the other was left at ambient temperature. After a wait of about ten minutes, the amounts of stretch, under a 1.35 kg weight, were recorded.

Usage

```
pair65
```

Format

This data frame contains the following columns:

heated a numeric vector giving the stretch lengths for the heated bands

ambient a numeric vector giving the stretch lengths for the unheated bands

Source

J.H. Maindonald

Examples

```
mean(pair65$heated - pair65$ambient)
sd(pair65$heated - pair65$ambient)
```

`panel.corr` *Scatterplot Panel*

Description

This function produces a bivariate scatterplot with the Pearson correlation. This is for use with the function `panelplot`.

Usage

```
panel.corr(data, ...)
```

Arguments

`data`	A data frame with columns x and y
`...`	Additional arguments

Author(s)

J.H. Maindonald

Examples

```
# correlation between body and brain weights for 20 mice:

weights <- litters[,-1]
names(weights) <-  c("x","y")
weights <- list(weights)
weights[[1]]$xlim <- range(litters[,2])
weights[[1]]$ylim <- range(litters[,3])
panelplot(weights, panel.corr, totrows=1, totcols=1)
```

`panelCorr` *Scatterplot Panel*

Description

This function produces a bivariate scatterplot with the Pearson correlation. This is for use with the function `panelplot`.

Usage

```
panelCorr(data, ...)
```

Arguments

`data`	A data frame with columns x and y
`...`	Additional arguments

Author(s)

J.H. Maindonald

Examples

```
# correlation between body and brain weights for 20 mice:

weights <- litters[,-1]
names(weights) <-  c("x","y")
weights <- list(weights)
weights[[1]]$xlim <- range(litters[,2])
weights[[1]]$ylim <- range(litters[,3])
panelplot(weights, panelCorr, totrows=1, totcols=1)
```

`panelplot` *Panel Plot*

Description

Panel plots of various types.

Usage

```
panelplot(data, panel=points, totrows=3, totcols=2, oma=rep(2.5, 4),
par.strip.text=NULL, ...)
```

Arguments

`data`	A list consisting of elements, each of which consists of x, y, xlim and ylim vectors
`panel`	The panel function to be plotted
`totrows`	The number of rows in the plot layout
`totcols`	The number of columns in the plot layout
`oma`	Outer margin area
`par.strip.text`	A data frame with column cex
`...`	Other parameters to be passed to plotting functions

Author(s)

J.H. Maindonald

Examples

```
x1 <- x2 <- x3 <- (11:30)/5
   y1 <- x1 + rnorm(20)/2
   y2 <- 2 - 0.05 * x1 + 0.1 * ((x1 - 1.75))^4 + 1.25 * rnorm(20)
   r <- round(cor(x1, y2), 3)
   rho <- round(cor(rank(x1), rank(y2)), 3)
   y3 <- (x1 - 3.85)^2 + 0.015 + rnorm(20)/4
   theta <- ((2 * pi) * (1:20))/20
   x4 <- 10 + 4 * cos(theta)
   y4 <- 10 + 4 * sin(theta) + (0.5 * rnorm(20))
   r1 <- cor(x1, y1)
   xy <- data.frame(x = c(rep(x1, 3), x4), y = c(y1, y2, y3, y4),
                    gp = rep(1:4, rep(20, 4)))
   xy <- split(xy,xy$gp)
   xlimdf <- lapply(list(x1,x2,x3,x4), range)
   ylimdf <- lapply(list(y1,y2,y3,y4), range)
   xy <- lapply(1:4, function(i,u,v,w){list(xlim=v[[i]],ylim=w[[i]],
                      x=u[[i]]$x, y=u[[i]]$y)},
                         u=xy, v=xlimdf, w=ylimdf)

   panel.corr <- function (data, ...)
       {
       x <- data$x
       y <- data$y
       points(x, y, pch = 16)
       chh <- par()$cxy[2]
       x1 <- min(x)
       y1 <- max(y) - chh/4
       r1 <- cor(x, y)
       text(x1, y1, paste(round(r1, 3)), cex = 0.8, adj = 0)
   }


   panelplot(xy, panel=panel.corr, totrows=2, totcols=2,oma=rep(1,4))
```

---

pause *Pause before continuing execution*

Description

If a program produces several plots, isertion of `pause()` between two plots suspends execution until the <Enter> key is pressed, to allow inspection of the current plot.

Usage

```
pause()
```

Author(s)

From the ‘sm’ package of Bowman and Azzalini (1997) plotSampDist *Plot(s) of simulated sampling distributions*

Description

Plots are based on the output from `simulateSampDist()`. By default, both density plots and normal probability plots are given, for a sample from the specified population and for samples of the relevant size(s)

Usage

```
plotSampDist(sampvalues, graph = c("density", "qq"), cex = 0.925,
             titletext = "Empirical sampling distributions of the",
             popsample=TRUE, ...)
```

Arguments

`sampvalues`	Object output from `simulateSampDist()`
`graph`	Either or both of `"density"` and `"qq"`
`cex`	Character size parameter, relative to default
`titletext`	Title for graph
`popsample`	If `TRUE` show distribution of random sample from population
`...`	Other graphics parameters

Value

Plots graph(s), as described above.

Author(s)

John Maindonald

References

Maindonald, J.H. and Braun, W.J. (3rd edn, 2010) “Data Analysis and Graphics Using R”, Sections 3.3 and 3.4.

See Also

See Also `help(simulateSampDist)`

Examples

```
## By default, sample from normal population
simAvs <- simulateSampDist()
par(pty="s")
plotSampDist(simAvs)
## Sample from empirical distribution
simAvs <- simulateSampDist(rpop=rivers)
plotSampDist(simAvs)

## The function is currently defined as
function(sampvalues, graph=c("density", "qq"), cex=0.925,
           titletext="Empirical sampling distributions of the",
           popsample=TRUE, ...){
    if(length(graph)==2)oldpar <- par(mfrow=c(1,2), mar=c(3.1,4.1,1.6,0.6),
               mgp=c(2.5, 0.75, 0), oma=c(0,0,1.5,0), cex=cex)
    values <- sampvalues$values
    numINsamp <- sampvalues$numINsamp
    funtxt <- sampvalues$FUN
    nDists <- length(numINsamp)+1
    nfirst <- 2
    legitems <- paste("Size", numINsamp)
    if(popsample){nfirst <- 1
                  legitems <- c("Size 1", legitems)
                 }
    if(match("density", graph)){
      popdens <- density(values[,1], ...)
      avdens <- vector("list", length=nDists)
      maxht <- max(popdens$y)
      ## For each sample size specified in numINsamp, calculate mean
      ## (or other statistic specified by FUN) for numsamp samples
      for(j in nfirst:nDists){
        av <- values[, j]
        avdens[[j]] <- density(av, ...)
        maxht <- max(maxht, avdens[[j]]$y)
      }
    }
    if(length(graph)>0)
      for(graphtype in graph){
        if(graphtype=="density"){
          if(popsample)
          plot(popdens, ylim=c(0, 1.2*maxht), type="l", yaxs="i",
               main="")
          else plot(avdens[[2]], type="n", ylim=c(0, 1.2*maxht),
                    yaxs="i", main="")
          for(j in 2:nDists)lines(avdens[[j]], col=j)
          legend("topleft",
                 legend=legitems,
                 col=nfirst:nDists, lty=rep(1,nDists-nfirst+1), cex=cex)
        }
        if(graphtype=="qq"){
          if(popsample) qqnorm(values[,1], main="")
          else qqnorm(values[,2], type="n")
```

```
          for(j in 2:nDists){
            qqav <- qqnorm(values[, j], plot.it=FALSE)
            points(qqav, col=j, pch=j)
           }
            legend("topleft", legend=legitems,
                   col=nfirst:nDists, pch=nfirst:nDists, cex=cex)
      }
     }
    if(par()$oma[3]>0){
      outer <- TRUE
      line=0
    }  else
    {
      outer <- FALSE
      line <- 1.25
    }
    if(!is.null(titletext))
      mtext(side=3, line=line,
            paste(titletext, funtxt),
            cex=1.1, outer=outer)
    if(length(graph)>1)par(oldpar)
  }
```

`plotSimDiags` *Diagnostic plots for simulated data*

Description

This provides diagnostic plots, closely equivalent to those provided by `plot.lm`, for simulated data. By default, simulated data are for the fitted model. Alternatively, simulated data can be supplied, making it possible to check the effct of fitting, e.g., an AR1 model.

Usage

```
plotSimDiags(obj, simvalues = NULL, seed = NULL,
types = NULL, which = c(1:3, 5), layout = c(4, 1), qqline=TRUE,
cook.levels = c(0.5, 1), caption = list("Residuals vs Fitted",
"Normal Q-Q", "Scale-Location", "Cook's distance", "Residuals vs Leverage",
expression("Cook's dist vs Leverage  " * h[ii]/(1 - h[ii]))),
...)
```

Arguments

`obj`	Fitted model object - `lm` or an object inheriting from `lm`
`simvalues`	Optional matrix of simulated data.
`seed`	Random number seed - set this to make results repeatable.
`types`	If set, this should be a list with six elements, ordinarily with each list element either `"p"` or `c("p","smooth")` or (which=2, which=6) NULL or (which=4) "h"

`which`	Set to be a subset of the numbers 1 to 6, as for `plot.lm`
`layout`	Controls the number of simulations and the layout of the plots. For example `layout=c(3,4)` will give 12 plots in a 3 by 4 layout.
`qqline`	logical: add line to normal Q-Q plot
`cook.levels`	Levels of Cook's statistics for which contours are to be plotted.
`caption`	list: Captions for the six graphs
`...`	Other parameters to be passed to plotting functions

Details

Diagnotic plots from repeated simulations from the fitted model provide a useful indication of the range of variation in the model diagnistics that are consistent with the fitted model.

Value

A list of lattice graphics objects is returned, one for each value of `which`. List elements for which a graphics object is not returned are set to NULL. Or if `which` is of length 1, a lattice graphics object.

`residVSfitted`	Residuals vs fitted
`normalQQ`	Normal quantile-quantile plot
`scaleVSloc`	Scale versus location
`CookDist`	Cook's distance vs observation number
`residVSlev`	Standardized residuals (for GLMs, standardized Pearson residuals) vs leverage
`CookVSlev`	Cook's distance vs leverage

For the default `which=c(1:3,5)`, list items 1, 2, 3 and 5 above contain graphics objects, with list elements 4 and 6 set to `NULL`.

Note

The graphics objects contained in individual list elements can be extracted for printing, or updating and printing, as required. If the value is returned to the command line, list elements that are not `NULL` will be printed in turn.

Author(s)

John Maindonald, with some code chunks adapted from `plot.lm`

References

See `plot.lm`

See Also

`plot.lm`, `lmdiags`

Examples

```
women.lm <- lm(height ~ weight, data=women)
gphlist <- plotSimDiags(obj=women.lm, which=c(1:3,5))
```

---

plotSimScat *Simulate scatterplots, from* lm *object with a single explanatory variable.*

Description

This plots simulated y-values, or residuals from such simulations, against x-values .

Usage

```
plotSimScat(obj, sigma = NULL, layout = c(4, 1), type = c("p", "r"),
show = c("points", "residuals"), ...)
```

Arguments

obj	An lm object with a single explanatory variable.
sigma	Standard deviation, if different from that for the supplied lm object.
layout	Columns by Rows layout for plots from the simulations.
type	See type as in plot.lm.
show	Specify points or residuals.
...	Other parameters to be passed to plotting functions

Value

A lattice graphics object is returned.

Author(s)

J H Maindonald

See Also

plotSimDiags

Examples

```
nihills.lm <- lm(timef~time, data=nihills)
plotSimDiags(nihills.lm)

## The function is currently defined as
function (obj, sigma = NULL, layout = c(4, 1), type = c("p",
    "r"), show = c("points", "residuals"))
{
    nsim <- prod(layout)
    if (is.null(sigma))
        sigma <- summary(obj)[["sigma"]]
    hat <- fitted(obj)
    xnam <- all.vars(formula(obj))[2]
    ynam <- all.vars(formula(obj))[1]
    df <- data.frame(sapply(1:nsim, function(x) rnorm(length(hat),
        sd = sigma)))
    if (show[1] == "points")
        df <- df + hat
    simnam <- names(df) <- paste("Simulation", 1:nsim, sep = "")
    df[, c(xnam, ynam)] <- model.frame(obj)[, c(xnam, ynam)]
    if (show[1] != "points") {
        df[, "Residuals"] <- df[, ynam] - hat
        ynam <- "Residuals"
        legadd <- "residuals"
    }
    else legadd <- "data"
    leg <- list(text = paste(c("Simulated", "Actual"), legadd),
        columns = 2)
    formula <- formula(paste(paste(simnam, collapse = "+"), "~",
        xnam))
    parset <- simpleTheme(pch = c(16, 16), lty = 2, col = c("black",
        "gray"))
    gph <- xyplot(formula, data = df, outer = TRUE, par.settings = parset,
        auto.key = leg, lty = 2, layout = layout, type = type)
    formxy <- formula(paste(ynam, "~", xnam))
    addgph <- xyplot(formxy, data = df, pch = 16, col = "gray")
    gph + as.layer(addgph, under = TRUE)
  }
```

---

poissonsim *Simple Poisson Regression Data Simulator*

Description

This function simulates simple regression data from a Poisson model. It also has the option to create over-dispersed data of a particular type.

Usage

```
poissonsim(x = seq(0, 1, length=101), a = 2, b = -4, intcp.sd=NULL,
            slope.sd=NULL, seed=NULL)
```

Arguments

`x`	a numeric vector representing the explanatory variable
`a`	the regression function intercept
`b`	the regression function slope
`intcp.sd`	standard deviation of the (random) intercept
`slope.sd`	standard deviation of the (random) slope
`seed`	numeric constant

Value

a list consisting of

`x`	the explanatory variable vector
`y`	the Poisson response vector

Examples

```
poissonsim()
```

`possum` *Possum Measurements*

Description

The `possum` data frame consists of nine morphometric measurements on each of 104 mountain brushtail possums, trapped at seven Australian sites from Southern Victoria to central Queensland. See `possumsites` for further details. The `fossum` data frame is the subset of `possum` that has measurements for the 43 females.

Usage

```
data(possum)
data(fossum)
```

Format

This data frame contains the following columns:

case observation number

site one of seven locations where possums were trapped. The sites were, in order,Cambarville, Bellbird, Whian Whian, Byrangery, Conondale, Allyn River and Bulburin

Pop a factor which classifies the sites as `Vic` Victoria, `other` New South Wales or Queensland

sex a factor with levels `f` female, `m` male

age age

hdlngth head length

skullw skull width

totlngth total length

taill tail length

footlgth foot length

earconch ear conch length

eye distance from medial canthus to lateral canthus of right eye

chest chest girth (in cm)

belly belly girth (in cm)

Source

Lindenmayer, D. B., Viggers, K. L., Cunningham, R. B., and Donnelly, C. F. 1995. Morphological variation among columns of the mountain brushtail possum, Trichosurus caninus Ogilby (Phalangeridae: Marsupiala). Australian Journal of Zoology 43: 449-458.

Examples

```
boxplot(earconch~sex, data=possum)
pause()

sex <- as.integer(possum$sex)
oldpar <- par(oma=c(2,4,5,4))
pairs(possum[, c(9:11)], pch=c(0,2:7), col=c("red","blue"),
  labels=c("tail\nlength","foot\nlength","ear conch\nlength"))
chh <- par()$cxy[2]; xleg <- 0.05; yleg <- 1.04
oldpar <- par(xpd=TRUE)
legend(xleg, yleg, c("Cambarville", "Bellbird", "Whian Whian  ",
  "Byrangery", "Conondale  ","Allyn River", "Bulburin"), pch=c(0,2:7),
  x.intersp=1, y.intersp=0.75, cex=0.8, xjust=0, bty="n", ncol=4)
text(x=0.2, y=yleg - 2.25*chh, "female", col="red", cex=0.8, bty="n")
text(x=0.75, y=yleg - 2.25*chh, "male", col="blue", cex=0.8, bty="n")
par(oldpar)
pause()

sapply(possum[,6:14], function(x)max(x,na.rm=TRUE)/min(x,na.rm=TRUE))
pause()

here <- na.omit(possum$footlgth)
possum.prc <- princomp(possum[here, 6:14])
pause()

plot(possum.prc$scores[,1] ~ possum.prc$scores[,2],
  col=c("red","blue")[as.numeric(possum$sex[here])],
  pch=c(0,2:7)[possum$site[here]], xlab = "PC1", ylab = "PC2")
  # NB: We have abbreviated the axis titles
chh <- par()$cxy[2]; xleg <- -15; yleg <- 20.5
oldpar <- par(xpd=TRUE)
legend(xleg, yleg, c("Cambarville", "Bellbird", "Whian Whian  ",
```

```
  "Byrangery", "Conondale  ","Allyn River", "Bulburin"), pch=c(0,2:7),
  x.intersp=1, y.intersp=0.75, cex=0.8, xjust=0, bty="n", ncol=4)
text(x=-9, y=yleg - 2.25*chh, "female", col="red", cex=0.8, bty="n")
summary(possum.prc, loadings=TRUE, digits=2)
par(oldpar)
pause()

require(MASS)
here <- !is.na(possum$footlgth)
possum.lda <- lda(site ~ hdlngth+skullw+totlngth+ taill+footlgth+
  earconch+eye+chest+belly, data=possum, subset=here)
options(digits=4)
possum.lda$svd   # Examine the singular values
plot(possum.lda, dimen=3)
  # Scatterplot matrix - scores on 1st 3 canonical variates (Figure 11.4)
possum.lda
pause()
boxplot(fossum$totlngth)
```

possumsites *Possum Sites*

Description

The `possumsites` data frame consists of Longitudes, Latitudes, and altitudes for the seven sites from Southern Victoria to central Queensland where the `possum` observations were made.

Usage

```
possumsites
```

Format

This data frame contains the following columns:

Longitude a numeric vector

Latitude a numeric vector

altitude in meters

Source

Lindenmayer, D. B., Viggers, K. L., Cunningham, R. B., and Donnelly, C. F. 1995. Morphological variation among columns of the mountain brushtail possum, Trichosurus caninus Ogilby (Phalangeridae: Marsupiala). Australian Journal of Zoology 43: 449-458.

Examples

```
require(oz)
oz(sections=c(3:5, 11:16))
attach(possumsites)
points(Longitude, Latitude, pch=16, col=2)
chw <- par()$cxy[1]
chh <- par()$cxy[2]
posval <- c(2,4,2,2,4,2,2)
text(Longitude+(3-posval)*chw/4, Latitude, row.names(possumsites), pos=posval)
```

---

powerplot *Plot of Power Functions*

Description

This function plots powers of a variable on the interval [0,10].

Usage

```
powerplot(expr="x^2", xlab="x", ylab="y", ...)
```

Arguments

expr	Functional form to be plotted
xlab	x-axis label
ylab	y-axis label
...	Further arguments, to be passed to plot()

Details

Other expressions such as "sin(x)" and "cos(x)", etc. could also be plotted with this function, but results are not guaranteed.

Value

A plot of the given expression on the interval [0,10].

Author(s)

J.H. Maindonald

Examples

```
    oldpar <- par(mfrow = c(2, 3), mar = par()$mar - c(
         1, 1, 1.0, 1),  mgp = c(1.5, 0.5, 0),  oma=c(0,1,0,1))
#     on.exit(par(oldpar))
     powerplot(expr="sqrt(x)", xlab="")
     powerplot(expr="x^0.25", xlab="", ylab="")
     powerplot(expr="log(x)", xlab="", ylab="")
     powerplot(expr="x^2")
     powerplot(expr="x^4", ylab="")
     powerplot(expr="exp(x)", ylab="")
par(oldpar)
```

---

poxetc *Deaths from various causes, in London from 1629 till 1881, with gaps*

Description

Deaths from "flux" or smallpox, measles, all causes, and ratios of the the first two categories to total deaths.

Usage

```
data(poxetc)
```

Format

This is a multiple time series consisting of 5 series: `fpox`, `measles`, `all`, `fpox2all`, `measles2all`.

Source

Guy, W. A. 1882. Two hundred and fifty years of small pox in London. Journal of the Royal Statistical Society 399-443.

References

Lancaster, H. O. 1990. Expectations of Life. Springer.

Examples

```
data(poxetc)
str(poxetc)
plot(poxetc)
```

`press` *Predictive Error Sum of Squares*

Description

Allen's PRESS statistic is computed for a fitted model.

Usage

```
press(obj)
```

Arguments

`obj`	A `lm` object

Value

A single numeric value.

Author(s)

W.J. Braun

See Also

`lm`

Examples

```
litters.lm <- lm(brainwt ~ bodywt + lsize, data = litters)
press(litters.lm)
litters.lm0 <- lm(brainwt ~ bodywt + lsize -1, data=litters)
press(litters.lm0) # no intercept
litters.lm1 <- lm(brainwt ~ bodywt, data=litters)
press(litters.lm1) # bodywt only
litters.lm2 <- lm(brainwt ~ bodywt + lsize + lsize:bodywt, data=litters)
press(litters.lm2) # include an interaction term
```

`primates` *Primate Body and Brain Weights*

Description

A subset of `Animals` data frame from the MASS library. It contains the average body and brain measurements of five primates.

Usage

```
primates
```

Format

This data frame contains the following columns:

Bodywt a numeric vector consisting of the body weights (in kg) of five different primates

Brainwt a numeric vector consisting of the corresponding brain weights (in g)

Source

P. J. Rousseeuw and A. M. Leroy (1987) Robust Regression and Outlier Detection. Wiley, p. 57.

Examples

```
attach(primates)
plot(x=Bodywt, y=Brainwt, pch=16,
       xlab="Body weight (kg)", ylab="Brain weight (g)",
       xlim=c(5,300), ylim=c(0,1500))
chw <- par()$cxy[1]
chh <- par()$cxy[2]
text(x=Bodywt+chw, y=Brainwt+c(-.1,0,0,.1,0)*chh,
       labels=row.names(primates), adj=0)
detach(primates)
```

`progression` *Progression of Record times for track races, 1912 - 2008*

Description

Progression in world record times for track and road races.

Usage

```
data(progression)
```

Format

A data frame with 227 observations on the following 4 columns.

`year` Year that time was first recorded

`Distance` distance in kilometers

`Time` time in minutes

`race` character; descriptor for event (100m, mile, ...)

Details

Record times for men's track events, from 1912 onwards. The series starts with times that were recognized as record times in 1912, where available.

Source

Links to sources for the data are at

https://en.wikipedia.org/wiki/Athletics_world_record

Examples

```
data(progression)
plot(log(Time) ~ log(Distance), data=progression)
res <- resid(lm(log(Time) ~ log(Distance), data=progression))
plot(res ~ log(Distance), data=progression,
     ylab="Residuals from regression line on log scales")
library(lattice)
xyplot(log(Time) ~ log(Distance), data=progression, type=c("p","r"))
xyplot(log(Time) ~ log(Distance), data=progression,
       type=c("p","smooth"))
```

`qreference` *Simulate QQ reference plots*

Description

This function computes the QQ plot for given data and specified distribution, then repeating the comparison for data simulated from the specified distribution. The plots for simulated data give an indication of the range of variation that is to expected, and thus calibrate the eye.

Usage

```
qreference(test = NULL, m = 30, nrep = 6, pch=c(16,2), distribution = function(x) qnorm(x,
    mean = ifelse(is.null(test), 0, mean(test)), sd = ifelse(is.null(test),
    1, sd(test))), seed = NULL, nrows = NULL, cex.strip = 0.75,
    xlab = NULL, ylab = NULL)
```

Arguments

`test`	a vector containing a sample to be tested; if not supplied, all qq-plots are for data simulated from the reference distribution
`m`	the sample size for the reference samples; default is test sample size if test sample is supplied
`nrep`	the total number of samples, including reference samples and test sample if any
`pch`	plot character(s)
`distribution`	reference distribution; default is standard normal
`seed`	the random number generator seed
`nrows`	number of rows in the plot layout
`cex.strip`	character expansion factor for labels
`xlab`	label for x-axis
`ylab`	label for y-axis

Value

QQ plots of the sample (if test is non-null) and all reference samples

Author(s)

J.H. Maindonald

Examples

```
# qreference(rt(30,4))

# qreference(rt(30,4), distribution=function(x) qt(x, df=4))

# qreference(rexp(30), nrep = 4)

# toycars.lm <- lm(distance ~ angle + factor(car), data = toycars)
# qreference(residuals(toycars.lm), nrep = 9)
```

`races2000` *Scottish Hill Races Data - 2000*

Description

The record times in 2000 for 77 Scottish long distance races. We believe the data are, for the most part, trustworthy. However, the `dist` variable for Caerketton (record 58) seems to have been variously recorded as 1.5 mi and 2.5 mi.

Usage

```
races2000
```

Format

This data frame contains the following columns:

dist distance, in miles (on the map)

climb total height gained during the route, in feet

time record time in hours

timef record time in hours for females

type a factor, with levels indicating type of race, i.e. hill, marathon, relay, uphill or other

Source

The Scottish Running Resource, http://www.hillrunning.co.uk

Examples

```
pairs(races2000[,-5])
```

`rainforest` *Rainforest Data*

Description

The `rainforest` data frame has 65 rows and 7 columns.

Usage

```
rainforest
```

Format

This data frame contains the following columns:

dbh a numeric vector

wood a numeric vector

bark a numeric vector

root a numeric vector

rootsk a numeric vector

branch a numeric vector

species a factor with levels `Acacia mabellae`, `C. fraseri`, `Acmena smithii`, `B. myrtifolia`

Source

J. Ash, Australian National University

References

Ash, J. and Helman, C. (1990) Floristics and vegetation biomass of a forest catchment, Kioloa, south coastal N.S.W. Cunninghamia, 2: 167-182.

Examples

```
table(rainforest$species)
```

`rareplants` *Rare and Endangered Plant Species*

Description

These data were taken from species lists for South Australia, Victoria and Tasmania. Species were classified as CC, CR, RC and RR, with C denoting common and R denoting rare. The first code relates to South Australia and Victoria, and the second to Tasmania. They were further classified by habitat according to the Victorian register, where D = dry only, W = wet only, and WD = wet or dry.

Usage

```
rareplants
```

Format

The format is: chr "rareplants"

Source

Jasmyn Lynch, Department of Botany and Zoology at Australian National University

Examples

```
chisq.test(rareplants)
```

`repPsych` *Summary results from Reproducibility Study: Psychology*

Description

The chief interest, in collating this dataset, was in the measures of effect size, for the originl study and for the replication.

Usage

```
data("repPsych")
```

Format

A data frame with 97 observations on the following 12 variables.

`stat` Test statistic. Character

`Journal` Where published. Character.

`Discipline` Cognitive or Social. Character.

`reportedP.O` Reported p-value. Character.

`effSizeO` Original effect size. Character.

`T_r.O` Original effect size, as correlation. Numeric.

`T_r.R` Replication effect size, as correlation. Numeric.

`efftype` a character vector

`tlike` Was test statistic t or F(1, m). Logical.

`d_O` Original effect size, on Cohen's d scale. Numeric.

`d_R` Replication effect size, on Cohen's d scale. Numeric.

Details

Effect estimates on a correlation scale were converted to a Cohen's d scale using `d = 2r/sqrt(1-r^2)`.

Source

https://osf.io/fgjvw/

References

https://osf.io/ezum7/ https://osf.io/z7aux Open Science Collaboration, 2015. Estimating the reproducibility of psychological science. Science, 349(6251), p.aac4716.

Examples

```
data(repPsych)
```

`rice` *Genetically Modified and Wild Type Rice Data*

Description

The `rice` data frame has 72 rows and 7 columns. The data are from an experiment that compared wild type (wt) and genetically modified rice plants (ANU843), each with three different chemical treatments (F10, NH4Cl, and NH4NO3).

Usage

```
rice
```

Format

This data frame contains the following columns:

PlantNo a numeric vector

Block a numeric vector

RootDryMass a numeric vector

ShootDryMass a numeric vector

trt a factor with levels `F10`, `NH4Cl`, `NH4NO3`, `F10 +ANU843`, `NH4Cl +ANU843`, `NH4NO3 +ANU843`

fert a factor with levels `F10` `NH4Cl` `NH4NO3`

variety a factor with levels `wt` `ANU843`

Source

Perrine, F.M., Prayitno, J., Weinman, J.J., Dazzo, F.B. and Rolfe, B. 2001. Rhizobium plasmids are involved in the inhibition or stimulation of rice growth and development. Australian Journal of Plant Physiology 28: 923-927.

Examples

```
print("One and Two-Way Comparisons - Example 4.5")
attach(rice)
oldpar <- par(las = 2)
stripchart(ShootDryMass ~ trt, pch=1, cex=1, xlab="Level of factor 1")
detach(rice)
pause()

rice.aov <- aov(ShootDryMass ~ trt, data=rice); anova(rice.aov)
anova(rice.aov)
pause()

summary.lm(rice.aov)$coef
pause()

rice$trt <- relevel(rice$trt, ref="NH4Cl")
  # Set NH4Cl as the baseline

fac1 <- factor(sapply(strsplit(as.character(rice$trt)," \\+"), function(x)x[1]))
anu843 <- sapply(strsplit(as.character(rice$trt), "\\+"),
function(x)c("wt","ANU843")[length(x)])
anu843 <- factor(anu843, levels=c("wt", "ANU843"))
attach(rice)
interaction.plot(fac1, anu843, ShootDryMass)
detach(rice)
par(oldpar)
```

rockArt *Pacific Rock Art features*

Description

Data characterise rock art at 103 sites in the Pacific.

Usage

```
rockArt
```

Format

A data frame with 103 observations on the following 641 variables.

`Site.No.` a numeric vector

`Site.Name` a character vector

`Site.Code` a character vector

`District` a character vector

`Island` a character vector

`Country` a character vector

`Technique` a character vector

`Engtech` a character vector

`red` a numeric vector

`black` a numeric vector

`yellow` a numeric vector

`white` a numeric vector

`green` a numeric vector

`red.blk` a numeric vector

`red.wh` a numeric vector

`red.yell` a numeric vector

`r.w.y` a numeric vector

`black.white` a numeric vector

`blue` a numeric vector

`Geology` a character vector

`Topography` a character vector

`Location` a character vector

`Proxhab.km.` a character vector

`Proxcoast.km.` a numeric vector

`Maxheight.m.` a numeric vector

`Language` a character vector

`No.motif` a character vector

`Ca1` a numeric vector

`Ca2` a numeric vector

`Ca3` a numeric vector

`Ca4` a numeric vector

`Cb5` a numeric vector

`Cb6` a numeric vector

`Cc7` a numeric vector

`Cc8` a numeric vector

`Cc9` a numeric vector

`Cc10` a numeric vector

`Cc11` a numeric vector

`Cc12` a numeric vector

`Cc13` a numeric vector

`Cc14` a numeric vector

`Cc15` a numeric vector

`Cc16` a numeric vector

`Cc17` a numeric vector

`Cc18` a numeric vector

`Cc19` a numeric vector

`Cc20` a numeric vector

`Cd21` a numeric vector

`Cd22` a numeric vector

`Cd23` a numeric vector

`Cd24` a numeric vector

`Cd25` a numeric vector

`Cd26` a numeric vector

`Cd27` a numeric vector

`Ce28` a numeric vector

`Ce29` a numeric vector

`Cf30` a numeric vector

`Cf31` a numeric vector

`Cf32` a numeric vector

`Cf33` a numeric vector

`Cf34` a numeric vector

`Cf35` a numeric vector

Cf36 a numeric vector

Cf37 a numeric vector

Cf38 a numeric vector

Cg39 a numeric vector

Cg40 a numeric vector

Ch41 a numeric vector

Ch42 a numeric vector

Ci43 a numeric vector

Ci44 a numeric vector

Cj45 a numeric vector

Ck46 a numeric vector

Ck47 a numeric vector

Cl48 a numeric vector

Cm49 a numeric vector

Cm50 a numeric vector

Cm51 a numeric vector

Cm52 a numeric vector

Cm53 a numeric vector

Cm54 a numeric vector

Cm55 a numeric vector

Cm56 a numeric vector

Cm57 a numeric vector

Cm58 a numeric vector

Cn59 a numeric vector

Cn60 a numeric vector

Cn61 a numeric vector

Cn62 a numeric vector

Cn63 a numeric vector

Cn64 a numeric vector

Cn65 a numeric vector

Cn66 a numeric vector

Cn67 a numeric vector

Cn68 a numeric vector

Cn69 a numeric vector

Cn70 a numeric vector

Cn71 a numeric vector

Co72 a numeric vector

Co73 a numeric vector
Co74 a numeric vector
Co75 a numeric vector
Co76 a numeric vector
Co77 a numeric vector
Co78 a numeric vector
Co79 a numeric vector
Cp80 a numeric vector
Cq81 a numeric vector
Cq82 a numeric vector
Cq83 a numeric vector
Cq84 a numeric vector
Cq85 a numeric vector
Cq86 a numeric vector
Cq87 a numeric vector
Cq88 a numeric vector
Cq89 a numeric vector
Cq90 a numeric vector
Cq91 a numeric vector
Cq92 a numeric vector
Cq93 a numeric vector
Cq94 a numeric vector
Cq95 a numeric vector
Cq96 a numeric vector
Cq97 a numeric vector
Cr98 a numeric vector
Cr99 a numeric vector
Cr100 a numeric vector
Cr101 a numeric vector
Cs102 a numeric vector
Cs103 a numeric vector
Cs104 a numeric vector
Cs105 a numeric vector
Cs106 a numeric vector
Ct107 a numeric vector
C108 a numeric vector
C109 a numeric vector

C110 a numeric vector
C111 a numeric vector
SSa1 a numeric vector
SSd2 a numeric vector
SSd3 a numeric vector
SSd4 a numeric vector
SSd5 a numeric vector
SSd6 a numeric vector
SSd7 a numeric vector
SSd8 a numeric vector
SSf9 a numeric vector
SSg10 a numeric vector
SSj11 a numeric vector
SSj12 a numeric vector
SSj13 a numeric vector
SSl14 a numeric vector
SSm15 a numeric vector
SSm16 a numeric vector
SSn17 a numeric vector
SSn18 a numeric vector
SSn19 a numeric vector
SSn20 a numeric vector
SSn21 a numeric vector
SSn22 a numeric vector
SSn23 a numeric vector
SSn24 a numeric vector
SSn25 a numeric vector
SSn26 a numeric vector
SSn27 a numeric vector
SSn28 a numeric vector
SSn29 a numeric vector
SSn30 a numeric vector
SSn31 a numeric vector
SSn32 a numeric vector
SSn33 a numeric vector
SSn34 a numeric vector
SSn35 a numeric vector

SSo36 a numeric vector

SSo37 a numeric vector

SSp38 a numeric vector

SSq39 a numeric vector

SSq40 a numeric vector

SSt41 a numeric vector

SSu42 a numeric vector

Oa1 a numeric vector

Oc2 a numeric vector

Od3 a numeric vector

Od4 a numeric vector

Oe5 a numeric vector

Of6 a numeric vector

Of7 a numeric vector

Of8 a numeric vector

Of9 a numeric vector

Og10 a numeric vector

Og11 a numeric vector

Og12 a numeric vector

Og13 a numeric vector

Og14 a numeric vector

Og15 a numeric vector

Oi16 a numeric vector

Om17 a numeric vector

Om18 a numeric vector

Om19 a numeric vector

Om20 a numeric vector

Om21 a numeric vector

On22 a numeric vector

On23 a numeric vector

On24 a numeric vector

Oq25 a numeric vector

Oq26 a numeric vector

Oq27 a numeric vector

.u28 a numeric vector

Ov29 a numeric vector

Ov30 a numeric vector

O31 a numeric vector
O32 a numeric vector
O33 a numeric vector
Sa1 a numeric vector
Sb2 a numeric vector
Sb3 a numeric vector
Sd4 a numeric vector
Sd5 a numeric vector
Sd6 a numeric vector
Sd7 a numeric vector
Se8 a numeric vector
Si9 a numeric vector
Sm10 a numeric vector
Sm11 a numeric vector
S12 a numeric vector
S13 a numeric vector
Sx14 a numeric vector
Sx15 a numeric vector
Sx16 a numeric vector
Sx17 a numeric vector
Sy18 a numeric vector
Sz19 a numeric vector
S20 a numeric vector
S21 a numeric vector
S22 a numeric vector
S23 a numeric vector
S24 a numeric vector
S25 a numeric vector
SCd1 a numeric vector
SCd2 a numeric vector
SCd3 a numeric vector
SCd4 a numeric vector
SCd5 a numeric vector
SCd6 a numeric vector
SCd7 a numeric vector
SCm8 a numeric vector
SCn9 a numeric vector

SCn10 a numeric vector

SCw11 a numeric vector

SCx12 a numeric vector

SCx13 a numeric vector

SCx14 a numeric vector

SCx15 a numeric vector

SCx16 a numeric vector

SCy17 a numeric vector

SCy18 a numeric vector

SC19 a numeric vector

SC20 a numeric vector

SC21 a numeric vector

SC22 a numeric vector

SC23 a numeric vector

SC24 a numeric vector

SC25 a numeric vector

SC26 a numeric vector

SRd1 a numeric vector

SRd2 a numeric vector

SRd3 a numeric vector

SRd4 a numeric vector

SRf5 a numeric vector

SRf6 a numeric vector

SRf7 a numeric vector

SRj8 a numeric vector

SR9 a numeric vector

SR10 a numeric vector

Bd1 a numeric vector

Bn2 a numeric vector

Bn3 a numeric vector

Bn4 a numeric vector

Bt5 a numeric vector

Bx6 a numeric vector

Ha1 a numeric vector

Hg2 a numeric vector

Hn3 a numeric vector

Hq4 a numeric vector

`Hq5` a numeric vector

`TDd1` a numeric vector

`TDf2` a numeric vector

`TDj3` a numeric vector

`TDn4` a numeric vector

`TDq5` a numeric vector

`TD6` a numeric vector

`TD7` a numeric vector

`TD8` a numeric vector

`TD9` a numeric vector

`Dc1` a numeric vector

`Dg2` a numeric vector

`Dh3` a numeric vector

`Dk4` a numeric vector

`Dm5` a numeric vector

`Dm6` a numeric vector

`D7` a numeric vector

`D8` a numeric vector

`D9` a numeric vector

`D10` a numeric vector

`D11` a numeric vector

`D12` a numeric vector

`D13` a numeric vector

`Ta1` a numeric vector

`Tc2` a numeric vector

`Tc3` a numeric vector

`Tc4` a numeric vector

`Td5` a numeric vector

`Tf6` a numeric vector

`Tf7` a numeric vector

`Tg8` a numeric vector

`Th9` a numeric vector

`To10` a numeric vector

`T11` a numeric vector

`T12` a numeric vector

`T13` a numeric vector

`T14` a numeric vector

T15 a numeric vector

T16 a numeric vector

CNg1 a numeric vector

CN2 a numeric vector

CN3 a numeric vector

CN4 a numeric vector

CN5 a numeric vector

CN6 a numeric vector

CN7 a numeric vector

CN8 a numeric vector

Ld1 a numeric vector

Lf2 a numeric vector

Lg3 a numeric vector

Lp4 a numeric vector

L5 a numeric vector

L6 a numeric vector

L7 a numeric vector

L8 a numeric vector

L9 a numeric vector

L10 a numeric vector

L11 a numeric vector

LS1 a numeric vector

LS2 a numeric vector

LL1 a numeric vector

LL2 a numeric vector

LL3 a numeric vector

LL4 a numeric vector

LL5 a numeric vector

EGd1 a numeric vector

EGf2 a numeric vector

CCd1 a numeric vector

CCn2 a numeric vector

CCn3 a numeric vector

EMc1 a numeric vector

EMd2 a numeric vector

EMd3 a numeric vector

EMf4 a numeric vector

`EMf5` a numeric vector

`EMn6` a numeric vector

`EMx7` a numeric vector

`EM8` a numeric vector

`EM9` a numeric vector

`EM10` a numeric vector

`EM11` a numeric vector

`EM12` a numeric vector

`TE1` a numeric vector

`TE2` a numeric vector

`TE3` a numeric vector

`TE4` a numeric vector

`TE5` a numeric vector

`BWe1` a numeric vector

`BWn2` a numeric vector

`BWn3` a numeric vector

`TS1` a numeric vector

`TS2` a numeric vector

`TS3` a numeric vector

`TS4` a numeric vector

`TS5` a numeric vector

`TS6` a numeric vector

`TS7` a numeric vector

`TS8` a numeric vector

`TS9` a numeric vector

`Pg1` a numeric vector

`Pg2` a numeric vector

`Pg3` a numeric vector

`DUaa1` a numeric vector

`DUw2` a numeric vector

`DU3` a numeric vector

`CP1` a numeric vector

`CP2` a numeric vector

`CP3` a numeric vector

`CP4` a numeric vector

`CP5` a numeric vector

`CP6` a numeric vector

CP7 a numeric vector
CP8 a numeric vector
CP9 a numeric vector
CP10 a numeric vector
CP11 a numeric vector
CP12 a numeric vector
STd1 a numeric vector
STd2 a numeric vector
STd3 a numeric vector
STg4 a numeric vector
STaa5 a numeric vector
STaa6 a numeric vector
STaa7 a numeric vector
STaa8 a numeric vector
ST9 a numeric vector
ST10 a numeric vector
ST11 a numeric vector
ST12 a numeric vector
Wd1 a numeric vector
Wd2 a numeric vector
Wd3 a numeric vector
Wd4 a numeric vector
Wn5 a numeric vector
Waa6 a numeric vector
Waa7 a numeric vector
W8 a numeric vector
W9 a numeric vector
W10 a numeric vector
W11 a numeric vector
W12 a numeric vector
W13 a numeric vector
Zd1 a numeric vector
Zd2 a numeric vector
Zn3 a numeric vector
Zw4 a numeric vector
Zw5 a numeric vector
Zaa6 a numeric vector

Z7 a numeric vector

Z8 a numeric vector

Z9 a numeric vector

Z10 a numeric vector

Z11 a numeric vector

Z12 a numeric vector

CLd1 a numeric vector

CLd2 a numeric vector

CLd3 a numeric vector

CLd4 a numeric vector

CLd5 a numeric vector

CLd6 a numeric vector

CLd7 a numeric vector

CLd8 a numeric vector

CLd9 a numeric vector

CLd10 a numeric vector

CLd11 a numeric vector

CLd12 a numeric vector

CLd13 a numeric vector

CLd14 a numeric vector

CLd15 a numeric vector

CLd16 a numeric vector

CLd17 a numeric vector

CLd18 a numeric vector

CLd19 a numeric vector

CLd20 a numeric vector

CLd21 a numeric vector

CLd22 a numeric vector

CLd23 a numeric vector

CLd24 a numeric vector

CLd25 a numeric vector

CLd26 a numeric vector

CLd27 a numeric vector

CLd28 a numeric vector

CLd29 a numeric vector

CLd30 a numeric vector

CLd31 a numeric vector

CLd32 a numeric vector

CLd33 a numeric vector

CLd34 a numeric vector

CLd35 a numeric vector

CLd36 a numeric vector

CLd37 a numeric vector

CLd38 a numeric vector

CLn39 a numeric vector

CLn40 a numeric vector

CLn41 a numeric vector

CLn42 a numeric vector

CLn43 a numeric vector

CLn44 a numeric vector

CLn45 a numeric vector

CLn46 a numeric vector

CLn47 a numeric vector

CLn48 a numeric vector

CLw49 a numeric vector

CL50 a numeric vector

CL51 a numeric vector

CL52 a numeric vector

CL53 a numeric vector

CL54 a numeric vector

CL55 a numeric vector

CL56 a numeric vector

CL57 a numeric vector

CL58 a numeric vector

CL59 a numeric vector

Xd1 a numeric vector

Xd2 a numeric vector

Xd3 a numeric vector

Xd4 a numeric vector

Xd5 a numeric vector

Xd6 a numeric vector

Xd7 a numeric vector

Xd8 a numeric vector

Xd9 a numeric vector

Xd10 a numeric vector
Xd11 a numeric vector
Xd12 a numeric vector
Xd13 a numeric vector
Xf14 a numeric vector
Xk15 a numeric vector
Xn16 a numeric vector
Xn17 a numeric vector
Xn18 a numeric vector
Xn19 a numeric vector
Xn20 a numeric vector
Xn21 a numeric vector
Xn22 a numeric vector
Xn23 a numeric vector
Xn24 a numeric vector
Xn25 a numeric vector
Xn26 a numeric vector
Xn27 a numeric vector
Xn28 a numeric vector
Xn29 a numeric vector
Xn30 a numeric vector
Xn31 a numeric vector
Xn32 a numeric vector
Xp33 a numeric vector
Xp34 a numeric vector
Xp35 a numeric vector
Xq36 a numeric vector
Xq37 a numeric vector
Xq38 a numeric vector
X39 a numeric vector
X40 a numeric vector
X41 a numeric vector
X42 a numeric vector
X43 a numeric vector
X44 a numeric vector
X45 a numeric vector
X46 a numeric vector

X47 a numeric vector

X48 a numeric vector

X49 a numeric vector

X50 a numeric vector

Qd1 a numeric vector

Qe2 a numeric vector

Qe3 a numeric vector

Qh4 a numeric vector

Qh5 a numeric vector

Qh6 a numeric vector

Qh7 a numeric vector

Qh8 a numeric vector

Qh9 a numeric vector

Qn10 a numeric vector

Qn11 a numeric vector

Qt12 a numeric vector

Q13 a numeric vector

Q14 a numeric vector

Q15 a numeric vector

Q16 a numeric vector

Q17 a numeric vector

Q18 a numeric vector

Q19 a numeric vector

Q20 a numeric vector

Q21 a numeric vector

Q22 a numeric vector

TZd1 a numeric vector

TZf2 a numeric vector

TZh3 a numeric vector

TZ4 a numeric vector

CRd1 a numeric vector

CR2 a numeric vector

CR3 a numeric vector

EUd1 a numeric vector

EUd2 a numeric vector

EUg3 a numeric vector

EUm4 a numeric vector

EUw5 a numeric vector
EU6 a numeric vector
Ud1 a numeric vector
Ud2 a numeric vector
Ud3 a numeric vector
Uaa4 a numeric vector
U5 a numeric vector
Vd1 a numeric vector
V2 a numeric vector
V3 a numeric vector
V4 a numeric vector
V5 a numeric vector
LWE1 a numeric vector
LWE2 a numeric vector
Ad1 a numeric vector
Al2 a numeric vector
Am3 a numeric vector
An4 a numeric vector
Aw5 a numeric vector
Aaa6 a numeric vector
A7 a numeric vector
A8 a numeric vector
A9 a numeric vector
EVd1 a numeric vector
EVg2 a numeric vector
TK1 a numeric vector
ECL1 a numeric vector
EFe1 a numeric vector
EFm2 a numeric vector
EFm3 a numeric vector
EF4 a numeric vector
LPo1 a numeric vector
LPq2 a numeric vector
LP3 a numeric vector
LP4 a numeric vector
LP5 a numeric vector
PT1 a numeric vector

`CSC` a numeric vector

`CSR` a numeric vector

`CCRC` a numeric vector

`SA` a numeric vector

`Anthrop` a numeric vector

`Turtle` a numeric vector

`Boat` a numeric vector

`Canoe` a numeric vector

`Hand` a numeric vector

`Foot` a numeric vector

`Lizard` a numeric vector

`Crocodile` a numeric vector

`Jellyfish` a numeric vector

`Bird` a numeric vector

`Anthrobird` a numeric vector

`Axe` a numeric vector

`Marine` a numeric vector

`Face` a numeric vector

`Zoo1` a numeric vector

`Zoo2` a numeric vector

`Zoo3` a numeric vector

`Zoo4` a numeric vector

`Zoo5` a numeric vector

`Zoo6` a numeric vector

Details

Note the vignette **rockArt**.

Source

Meredith Wilson: *Picturing Pacific Pre-History* (PhD thesis), 2002, Australian National University.

References

Meredith Wilson: Rethinking regional analyses of Western Pacific rock-art. *Records of the Australian Museum*, Supplement 29: 173-186.

Examples

```
data(rockArt)
rockart.dist <- dist(x = as.matrix(rockArt[, 28:641]), method = "binary")
sum(rockart.dist==1)/length(rockart.dist)
plot(density(rockart.dist, to = 1))
rockart.cmd <- cmdscale(rockart.dist)
tab <- table(rockArt$District)
district <- as.character(rockArt$District)
district[!(rockArt$District %in% names(tab)[tab>5])] <- "other"
## Not run:
xyplot(rockart.cmd[,2] ~ rockart.cmd[,1], groups=district,
       auto.key=list(columns=5),
       par.settings=list(superpose.symbol=list(pch=16)))
library(MASS)
## For sammon, need to avoid zero distances
omit <- c(47, 54, 60, 63, 92)
rockart.dist <- dist(x = as.matrix(rockArt[-omit, 28:641]), method = "binary")
rockart.cmd <- cmdscale(rockart.dist)
rockart.sam <- sammon(rockart.dist, rockart.cmd)
xyplot(rockart.sam$points[,2] ~ rockart.sam$points[,1],
       groups=district[-omit], auto.key=list(columns=5),
       par.settings=list(superpose.symbol=list(pch=16)))
## Notice the very different appearance of the Sammon plot

## End(Not run)
```

`roller` *Lawn Roller Data*

Description

The `roller` data frame has 10 rows and 2 columns. Different weights of roller were rolled over different parts of a lawn, and the depression was recorded.

Usage

```
roller
```

Format

This data frame contains the following columns:

weight a numeric vector consisting of the roller weights

depression the depth of the depression made in the grass under the roller

Source

Stewart, K.M., Van Toor, R.F., Crosbie, S.F. 1988. Control of grass grub (Coleoptera: Scarabaeidae) with rollers of different design. N.Z. Journal of Experimental Agriculture 16: 141-150.

Examples

```
plot(roller)
roller.lm <- lm(depression ~ weight, data = roller)
plot(roller.lm, which = 4)
```

---

sampdist *Plot sampling distribution of mean or other sample statistic.*

Description

The function sampvals generates the data. A density plot of a normal probability plot is provided, for one or mare sample sizes. For a density plot, the density estimate for the population is superimposed in gray. For the normal probability plot, the population plot is a dashed gray line. Default arguments give the sampling distribution of the mean, for a distribution that is mildly positively skewed.

Usage

```
sampdist(sampsize = c(3, 9, 30), seed = NULL, nsamp = 1000, FUN = mean,
         sampvals = function(n) exp(rnorm(n, mean = 0.5, sd = 0.3)),
         tck = NULL, plot.type = c("density", "qq"), layout = c(3, 1))
```

Arguments

sampvals	Function that generates the data. For sampling from existing data values, this might be function that generates bootstrap samples.
sampsize	One or more sample sizes. A plot will be provided for each different sample size.
seed	Specify a seed if it is required to make the exact set(s) of sample values reproducible.
nsamp	Number of samples.
FUN	Function that calculates the sample statistic.
plot.type	Specify density, or qq. Or if no plot is required, specify "".
tck	Tick size on lattice plots, by default 1, but 0.5 may be suitable for plots that are, for example, 50% of the default dimensions in each direction.
layout	Layout on page, e.g. c(3,1) for a 3 columns by one row layout.

Value

Data frame

Author(s)

John Maindonald.

Examples

```
sampdist(plot.type="density")
sampdist(plot.type="qq")

## The function is currently defined as
  function (sampsize = c(3, 9, 30), seed = NULL, nsamp = 1000, FUN = mean,
            sampvals = function(n) exp(rnorm(n, mean = 0.5, sd = 0.3)),
            tck = NULL, plot.type = c("density", "qq"), layout = c(3,
                                                          1))
{
  if (!is.null(seed))
    set.seed(seed)
  ncases <- length(sampsize)
  y <- sampvals(nsamp)
  xlim = quantile(y, c(0.01, 0.99))
  xlim <- xlim + c(-1, 1) * diff(xlim) * 0.1
  samplingDist <- function(sampsize=3, nsamp=1000, FUN=mean)
    apply(matrix(sampvals(sampsize*nsamp), ncol=sampsize), 1, FUN)
  df <- data.frame(sapply(sampsize, function(x)samplingDist(x, nsamp=nsamp)))
  names(df) <- paste("y", sampsize, sep="")
  form <- formula(paste("~", paste(names(df), collapse="+")))
  lab <- lapply(sampsize, function(x) substitute(A, list(A = paste(x))))
  if (plot.type[1] == "density")
    gph <- densityplot(form, data=df, layout = layout, outer=TRUE,
                       plot.points = FALSE, panel = function(x, ...) {
                         panel.densityplot(x, ..., col = "black")
                         panel.densityplot(y, col = "gray40", lty = 2,
                                           ...)
                       }, xlim = xlim, xlab = "", scales = list(tck = tck),
                       between = list(x = 0.5), strip = strip.custom(strip.names = TRUE,
                       factor.levels = as.expression(lab), var.name = "Sample size",
                                                    sep = expression(" = ")))
  else if (plot.type[1] == "qq")
    gph <- qqmath(form, data = df, layout = layout, plot.points = FALSE,
                  outer=TRUE,
                  panel = function(x, ...) {
                    panel.qqmath(x, ..., col = "black", alpha=0.5)
                    panel.qqmath(y, col = "gray40", lty = 2, type = "l",
                                 ...)
                  }, xlab = "", xlim = c(-3, 3), ylab = "", scales = list(tck = tck),
                  between = list(x = 0.5), strip = strip.custom(strip.names = TRUE,
                  factor.levels = as.expression(lab), var.name = "Sample size",
                                               sep = expression(" = ")))
  if (plot.type[1] %in% c("density", "qq"))
    print(gph)
  invisible(df)
}
```

---

science *School Science Survey Data*

Description

The `science` data frame has 1385 rows and 7 columns.

The data are on attitudes to science, from a survey where there were results from 20 classes in private schools and 46 classes in public schools.

Usage

```
science
```

Format

This data frame contains the following columns:

State a factor with levels `ACT` Australian Capital Territory, `NSW` New South Wales

PrivPub a factor with levels `private` school, `public` school

school a factor, coded to identify the school

class a factor, coded to identify the class

sex a factor with levels `f`, `m`

like a summary score based on two of the questions, on a scale from 1 (dislike) to 12 (like)

Class a factor with levels corresponding to each class

Source

Francine Adams, Rosemary Martin and Murali Nayadu, Australian National University

Examples

```
classmeans <- with(science, aggregate(like, by=list(PrivPub, Class), mean))
names(classmeans) <- c("PrivPub","Class","like")
dim(classmeans)

attach(classmeans)
boxplot(split(like, PrivPub), ylab = "Class average of attitude to science score", boxwex = 0.4)
rug(like[PrivPub == "private"], side = 2)
rug(like[PrivPub == "public"], side = 4)
detach(classmeans)
if(require(lme4, quietly=TRUE)) {
science.lmer <- lmer(like ~ sex + PrivPub + (1 | school) +
                     (1 | school:class), data = science,
                     na.action=na.exclude)
summary(science.lmer)
science1.lmer <- lmer(like ~ sex + PrivPub + (1 | school:class),
                      data = science, na.action=na.exclude)
summary(science1.lmer)
ranf <- ranef(obj = science1.lmer, drop=TRUE)[["school:class"]]
flist <- science1.lmer@flist[["school:class"]]
privpub <- science[match(names(ranf), flist), "PrivPub"]
num <- unclass(table(flist)); numlabs <- pretty(num)
## Plot effect estimates vs numbers
```

```
plot(sqrt(num), ranf, xaxt="n", pch=c(1,3)[as.numeric(privpub)],
     xlab="# in class (square root scale)",
     ylab="Estimate of class effect")
lines(lowess(sqrt(num[privpub=="private"]),
             ranf[privpub=="private"], f=1.1), lty=2)
lines(lowess(sqrt(num[privpub=="public"]),
             ranf[privpub=="public"], f=1.1), lty=3)
axis(1, at=sqrt(numlabs), labels=paste(numlabs))
}
```

---

seedrates *Barley Seeding Rate Data*

Description

The seedrates data frame has 5 rows and 2 columns on the effect of seeding rate of barley on yield.

Usage

```
seedrates
```

Format

This data frame contains the following columns:

rate the seeding rate

grain the number of grain per head of barley

Source

McLeod, C.C. 1982. Effect of rates of seeding on barley grown for grain. New Zealand Journal of Agriculture 10: 133-136.

References

Maindonald J H 1992. Statistical design, analysis and presentation issues. New Zealand Journal of Agricultural Research 35: 121-141.

Examples

```
plot(grain~rate,data=seedrates,xlim=c(50,180),ylim=c(15.5,22),axes=FALSE)
new.df<-data.frame(rate=(2:8)*25)
seedrates.lm1<-lm(grain~rate,data=seedrates)
seedrates.lm2<-lm(grain~rate+I(rate^2),data=seedrates)
hat1<-predict(seedrates.lm1,newdata=new.df,interval="confidence")
hat2<-predict(seedrates.lm2,newdata=new.df,interval="confidence")
axis(1,at=new.df$rate); axis(2); box()
z1<-spline(new.df$rate, hat1[,"fit"]); z2<-spline(new.df$rate,
```

```
hat2[,"fit"])
rate<-new.df$rate; lines(z1$x,z1$y)
lines(spline(rate,hat1[,"lwr"]),lty=1,col=3)
lines(spline(rate,hat1[,"upr"]),lty=1,col=3)
lines(z2$x,z2$y,lty=4)
lines(spline(rate,hat2[,"lwr"]),lty=4,col=3)
lines(spline(rate,hat2[,"upr"]),lty=4,col=3)
```

`show.colors` *Show R's Colors*

Description

This function displays the built-in colors.

Usage

```
show.colors(type=c("singles", "shades", "gray"), order.cols=TRUE)
```

Arguments

`type`	type of display - single, multiple or gray shades
`order.cols`	Arrange colors in order

Value

A plot of colors for which there is a single shade (type = "single"), multiple shades (type = "multiple"), or gray shades (type = "gray")

Author(s)

J.H. Maindonald

Examples

```
require(MASS)
show.colors()
```

simulateLinear *Simulation of Linear Models for ANOVA vs. Regression Comparison*

Description

This function simulates a number of bivariate data sets in which there are replicates at each level of the predictor. The p-values for ANOVA and for the regression slope are compared, and a lattice graphics object returned.

Usage

```
simulateLinear(sd=2, npoints=5, nrep=4, nsets=200, graphtype="xy",
seed=21, ...)
```

Arguments

`sd`	The error standard deviation
`npoints`	Number of distinct predictor levels
`nrep`	Number of replications at each level
`nsets`	Number of simulation runs
`graphtype`	Type of graph; x-y plot (`graphtype="xy"`), overlaid density plots (`graphtype="density"`), or density plot for x-y difference (`graphtype="density-diff"`)
`seed`	Random Number generator seed
`...`	Additional arguments, to be passed through to the lattice function that is called

Value

A lattice graphics object.

Author(s)

J.H. Maindonald

Examples

```
simulateLinear()
```

`simulateSampDist` *Simulated sampling distribution of mean or other statistic*

Description

Simulates the sample distribution of the specified statistic, for samples of the size(s) specified in `numINsamp`. Additionally a with replacement) sample is drawn from the specified population.

Usage

```
simulateSampDist(rpop = rnorm, numsamp = 100, numINsamp = c(4, 16),
                 FUN = mean, seed=NULL
      )
```

Arguments

`rpop`	Either a function that generates random samples from the specified distribution, or a vector of values that define the population (i.e., an empirical distribution)
`numsamp`	Number of samples that should be taken. For close approximation of the asymptotic distribution (e.g., for the mean) this number should be large
`numINsamp`	Size(s) of each of the `numsamp` sample(s)
`FUN`	Function to calculate the statistic whose sampling distribution is to be simulated
`seed`	Optional seed for random number generation

Value

List, with elements `values`, `numINsamp` and `FUN`

`values`	Matrix, with dimensions `numsamp` by `numINsamp + 1`. The first column has a random with replacement sample from the population, while the remaining `length(numINsamp)` columns hold simulated values from sampling distributions with samples of the specified size(s)
`numINsamp`	Input value of `numINsamp`
`numsamp`	Input value of `numsamp`

Author(s)

John Maindonald

References

Maindonald, J.H. and Braun, W.J. (3rd edn, 2010) *Data Analysis and Graphics Using R*, 3rd edn, Sections 3.3 and 3.4

See Also

`help(plotSampDist)`

Examples

```
## By default, sample from normal population
simAvs <- simulateSampDist()
par(pty="s")
plotSampDist(simAvs)
## Sample from empirical distribution
simAvs <- simulateSampDist(rpop=rivers)
plotSampDist(simAvs)


## The function is currently defined as
function(rpop=rnorm, numsamp=100, numINsamp=c(4,16), FUN=mean,
seed=NULL){
    if(!is.null(seed))set.seed(seed)
    funtxt <- deparse(substitute(FUN))
    nDists <- length(numINsamp)+1
    values <- matrix(0, nrow=numsamp, ncol=nDists)
    if(!is.function(rpop)) {
      x <- rpop
      rpop <- function(n)sample(x, n, replace=TRUE)
    }
    values[,1] <- rpop(numsamp)
    for(j in 2:nDists){
      n <- numINsamp[j-1]
      for(i in 1:numsamp)values[i, j] <- FUN(rpop(n))
    }
    colnames(values) <- paste("Size", c(1, numINsamp))
    invisible(list(values=values, numINsamp=numINsamp, FUN=funtxt))
  }
```

---

socsupport *Social Support Data*

Description

Data from a survey on social and other kinds of support.

Usage

`socsupport`

Format

This data frame contains the following columns:

gender a factor with levels `female`, `male`

age age, in years, with levels `18-20`, `21-24`, `25-30`, `31-40`,`40+`

country a factor with levels `australia`, `other`

marital a factor with levels married, other, single

livewith a factor with levels alone, friends, other, parents, partner, residences

employment a factor with levels employed fulltime, employed part-time, govt assistance, other, parental support

firstyr a factor with levels first year, other

enrolment a factor with levels full-time, part-time, <NA>

emotional summary of 5 questions on emotional support availability

emotionalsat summary of 5 questions on emotional support satisfaction

tangible summary of 4 questions on availability of tangible support

tangiblesat summary of 4 questions on satisfaction with tangible support

affect summary of 3 questions on availability of affectionate support sources

affectsat summary of 3 questions on satisfaction with affectionate support sources

psi summary of 3 questions on availability of positive social interaction

psisat summary of 3 questions on satisfaction with positive social interaction

esupport summary of 4 questions on extent of emotional support sources

psupport summary of 4 questions on extent of practical support sources

supsources summary of 4 questions on extent of social support sources (formerly, socsupport)

BDI Score on the Beck depression index (summary of 21 questions)

Source

Melissa Manning, Psychology, Australian National University

Examples

```
attach(socsupport)

not.na <- apply(socsupport[,9:19], 1, function(x)!any(is.na(x)))
ss.pr1 <- princomp(as.matrix(socsupport[not.na, 9:19]), cor=TRUE)
pairs(ss.pr1$scores[,1:3])
sort(-ss.pr1$scores[,1])        # Minus the largest value appears first
pause()

not.na[36] <- FALSE
ss.pr <- princomp(as.matrix(socsupport[not.na, 9:19]), cor=TRUE)
summary(ss.pr)          # Examine the contribution of the components
pause()

# We now regress BDI on the first six principal components:
ss.lm <- lm(BDI[not.na] ~ ss.pr$scores[, 1:6], data=socsupport)
summary(ss.lm)$coef
pause()

ss.pr$loadings[,1]
plot(BDI[not.na] ~  ss.pr$scores[ ,1], col=as.numeric(gender),
pch=as.numeric(gender),  xlab ="1st principal component", ylab="BDI")
topleft <- par()$usr[c(1,4)]
legend(topleft[1], topleft[2], col=1:2, pch=1:2, legend=levels(gender))
```

softbacks — *Measurements on a Selection of Paperback Books*

Description

This is a subset of the `allbacks` data frame which gives measurements on the volume and weight of 8 paperback books.

Usage

```
softbacks
```

Format

This data frame contains the following columns:

volume a numeric vector giving the book volumes in cubic centimeters

weight a numeric vector giving the weights in grams

Source

The bookshelf of J. H. Maindonald.

Examples

```
print("Outliers in Simple Regression - Example 5.2")
paperback.lm <- lm(weight ~ volume, data=softbacks)
summary(paperback.lm)
plot(paperback.lm)
```

sorption — *sorption data set*

Description

Concentration-time measurements on different varieties of apples under methyl bromide injection.

Usage

```
data(sorption)
```

Format

A data frame with 192 observations on the following 14 variables.

m5 a numeric vector

m10 a numeric vector

m30 a numeric vector

m60 a numeric vector

m90 a numeric vector

m120 a numeric vector

ct concentration-time

Cultivar a factor with levels `Pacific Rose` `BRAEBURN` `Fuji` `GRANNY` `Gala` `ROYAL` `Red Delicious` `Splendour`

Dose injected dose of methyl bromide

rep replicate number, within Cultivar and year

year a factor with levels `1988` `1989` `1998` `1999`

year.rep a factor with levels `1988:1` `1988:2` `1988:3` `1989:1` `1989:2` `1998:1` `1998:2` `1998:3` `1999:1` `1999:2`

gp a factor with levels `BRAEBURN1` `BRAEBURN2` `Fuji1` `Fuji10` `Fuji2` `Fuji6` `Fuji7` `Fuji8` `Fuji9` `GRANNY1` `GRANNY2` `Gala4` `Gala5` `Pacific Rose10` `Pacific Rose6` `Pacific Rose7` `Pacific Rose8` `Pacific Rose9` `ROYAL1` `ROYAL2` `Red Del10` `Red Del9` `Red Delicious1` `Red Delicious2` `Red Delicious3` `Red Delicious4` `Red Delicious5` `Red Delicious6` `Red Delicious7` `Red Delicious8` `Splendour4` `Splendour5`

inyear a factor with levels `1` `2` `3` `4` `5` `6`

SP500close — *Closing Numbers for S and P 500 Index*

Description

Closing numbers for S and P 500 Index, Jan. 1, 1990 through early 2000.

Usage

```
SP500close
```

Source

Derived from SP500 in the MASS library.

Examples

```
ts.plot(SP500close)
```

`SP500W90` *Closing Numbers for S and P 500 Index - First 100 Days of 1990*

Description

Closing numbers for S and P 500 Index, Jan. 1, 1990 through early 2000.

Usage

```
SP500W90
```

Source

Derived from SP500 in the MASS library.

Examples

```
ts.plot(SP500W90)
```

`spam7` *Spam E-mail Data*

Description

The data consist of 4601 email items, of which 1813 items were identified as spam. This is a subset of the full dataset, with six only of the 57 explanatory variables in the complete dataset.

Usage

```
spam7
```

Format

Columns included are:

crl.tot total length of uninterrupted sequences of capitals

dollar Occurrences of ‘$’, as percent of total number of characters

bang Occurrences of ‘!’, as percent of total number of characters

money Occurrences of ‘money’, as percent of total number of words

n000 Occurrences of the string ‘000’, as percent of total number of words

make Occurrences of ‘make’, as % of total number of words

yesno outcome variable, a factor with levels `n` not spam, `y` spam

Source

George Forman, Hewlett-Packard Laboratories

The complete dataset, and documentation, are available from Spam database

Examples

```
require(rpart)
spam.rpart <- rpart(formula = yesno ~ crl.tot + dollar + bang +
   money + n000 + make, data=spam7)
plot(spam.rpart)
text(spam.rpart)
```

---

stVincent *Averages by block of yields for the St. Vincent Corn data*

Description

These data frames have yield averages by blocks (parcels).

Usage

```
stVincent
```

Format

A data frame with 324 observations on 8 variables.

code a numeric vector

island a numeric vector

id a numeric vector

site a factor with 8 levels.

block a factor with levels `I` `II` `III` `IV`

plot a numeric vector

trt a factor consisting of 12 levels

harvwt a numeric vector; the average yield

Source

Andrews DF; Herzberg AM, 1985. Data. A Collection of Problems from Many Fields for the Student and Research Worker. Springer-Verlag. (pp. 339-353)

sugar — *Sugar Data*

Description

The `sugar` data frame has 12 rows and 2 columns. They are from an experiment that compared an unmodified wild type plant with three different genetically modified forms. The measurements are weights of sugar that were obtained by breaking down the cellulose.

Usage

```
sugar
```

Format

This data frame contains the following columns:

weight weight, in mg

trt a factor with levels `Control` i.e. unmodified Wild form, `A` Modified 1, `B` Modified 2, `C` Modified 3

Source

Anonymous

Examples

```
sugar.aov <- aov(weight ~ trt, data=sugar)
fitted.values(sugar.aov)
summary.lm(sugar.aov)
sugar.aov <- aov(formula = weight ~ trt, data = sugar)
summary.lm(sugar.aov)
```

sumry — *A more flexible alternatives to* `summary`.

Description

At present this has a method only for `glm` objects. The function `print.sumry.glm` allows greater control over what is printed.

Usage

```
sumry(object, ...)
```

Arguments

`object`	An object for with a summary is required. At present, this must be a glm object.
`...`	additional arguments affecting the summary produced.

Value

Returns summary information.

Author(s)

John Maindonald

See Also

`print.sumry.glm`, `sumry`, `glm`

---

sumry.glm — *Summarizing Generalized Linear Model Fits*

Description

These functions are `methods` for class `glm` or `sumry.glm` objects.

Usage

```
## S3 method for class 'glm'
sumry(object, dispersion = NULL, correlation = FALSE,
        symbolic.cor = FALSE, ...)

## S3 method for class 'sumry.glm'
print(x, digits = max(3L, getOption("digits") - 3L),
                          symbolic.cor = FALSE,
                          signif.stars = getOption("show.signif.stars"),
                          call=FALSE, deviance.residuals=FALSE,
                          show.iter=10, ...)
```

Arguments

`object`	an object of class `"glm"`, usually, a result of a call to `glm`.
`x`	an object of class `"summary.glm"`, usually, a result of a call to `summary.glm`.
`dispersion`	the dispersion parameter for the family used. Either a single numerical value or `NULL` (the default), when it is inferred from `object` (see 'Details').
`correlation`	logical; if `TRUE`, the correlation matrix of the estimated parameters is returned and printed.
`digits`	the number of significant digits to use when printing.

symbolic.cor	logical. If TRUE, print the correlations in a symbolic form (see symnum) rather than as numbers.
signif.stars	logical. If TRUE, 'significance stars' are printed for each coefficient.
call	logical. If TRUE, details of the function call are printed.
deviance.residuals	logical. If TRUE, deviance residuals are printed.
show.iter	NULL or integer. If NULL, or if the number of iterations is greater than the specified integer, then the number of iterations will be printed.
...	further arguments passed to or from other methods.

Details

The function print.sumry.glm allows, relative to print.summary.glm, some greater flexibility in what is printed. By default, details of the call to glm are omitted, and details of the number of interations only in the unusual case where this number is greater than 10. See the help page for summary.glm for further details.

Value

sumry.glm returns an object of class "sumry.glm", a list with the same components as summary.glm.

See Also

glm, summary.

Examples

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

---

tinting *Car Window Tinting Experiment Data*

Description

These data are from an experiment that aimed to model the effects of the tinting of car windows on visual performance. The authors were mainly interested in effects on side window vision, and hence in visual recognition tasks that would be performed when looking through side windows.

Usage

```
tinting
```

Format

This data frame contains the following columns:

case observation number

id subject identifier code (1-26)

age age (in years)

sex a factor with levels `f` female, `m` male

tint an ordered factor with levels representing degree of tinting: `no` < `lo` < `hi`

target a factor with levels `locon`: low contrast, `hicon`: high contrast

it the inspection time, the time required to perform a simple discrimination task (in milliseconds)

csoa critical stimulus onset asynchrony, the time to recognize an alphanumeric target (in milliseconds)

agegp a factor with levels `younger`, 21-27, `older`, 70-78

Details

Visual light transmittance (VLT) levels were 100% (tint=none), 81.3% (tint=lo), and 35.1% (tint=hi). Based on these and other data, Burns et al. argue that road safety may be compromised if the front side windows of cars are tinted to 35

Source

Burns, N.R., Nettlebeck, T., White, M. and Willson, J., 1999. Effects of car window tinting on visual performance: a comparison of younger and older drivers. Ergonomics 42: 428-443.

Examples

```
library(lattice)
levels(tinting$agegp) <- capstring(levels(tinting$agegp))
xyplot(csoa ~ it | sex * agegp, data=tinting) # Simple use of xyplot()
pause()

xyplot(csoa ~ it|sex*agegp, data=tinting, panel=panel.superpose, groups=target)
pause()

xyplot(csoa ~ it|sex*agegp, data=tinting, panel=panel.superpose, col=1:2,
  groups=target, key=list(x=0.14, y=0.84, points=list(pch=rep(1,2),
  col=1:2), text=list(levels(tinting$target), col=1:2), border=TRUE))
## Not run:
xyplot(csoa ~ it|sex*agegp, data=tinting, panel=panel.superpose,
  groups=tint, type=c("p","smooth"), span=0.8, col=1:3,
  key=list(x=0.14, y=0.84, points=list(pch=rep(1,2), col=1:3),
  text=list(levels(tinting$tint), col=1:3), border=TRUE))

## End(Not run)
```

`tomato` *Root weights of tomato plants exposed to 4 different treatments*

Description

The `tomato` data frame has 24 rows and 2 columns. They are from an experiment that exposed tomato plants to four different 'nutrients'.

Usage

```
data(tomato)
```

Format

This data frame contains the following columns:

weight weight, in g

trt a factor with levels `water only`, `conc nutrient`, `2-4-D + conc nutrient`, `3x conc nutrient`

Source

Dr Ron Balham, Victoria University of Wellington NZ, sometime in 1971 - 1976.

Examples

```
tomato.aov <- aov(log(weight) ~ trt, data=tomato)
fitted.values(tomato.aov)
summary.lm(tomato.aov)
tomato.aov <- aov(formula = weight ~ trt, data = tomato)
summary.lm(tomato.aov)
```

`toycars` *Toy Cars Data*

Description

The `toycars` data frame has 27 rows and 3 columns. Observations are on the distance traveled by one of three different toy cars on a smooth surface, starting from rest at the top of a 16 inch long ramp tilted at varying angles.

Usage

```
toycars
```

Format

This data frame contains the following columns:

angle tilt of ramp, in degrees

distance distance traveled, in meters

car a numeric code (1 = first car, 2 = second car, 3 = third car)

Examples

```
toycars.lm <- lm(distance ~ angle + factor(car), data=toycars)
summary(toycars.lm)
```

---

two65 *Unpaired Heated Elastic Bands*

Description

Twenty-one elastic bands were divided into two groups.

One of the sets was placed in hot water (60-65 degrees C) for four minutes, while the other was left at ambient temperature. After a wait of about ten minutes, the amounts of stretch, under a 1.35 kg weight, were recorded.

Usage

```
pair65
```

Format

This list contains the following elements:

heated a numeric vector giving the stretch lengths for the heated bands

ambient a numeric vector giving the stretch lengths for the unheated bands

Source

J.H. Maindonald

Examples

```
twot.permutation(two65$ambient,two65$heated) # two sample permutation test
```

twot.permutation *Two Sample Permutation Test - Obsolete*

Description

This function computes the p-value for the two sample t-test using a permutation test. The permutation density can also be plotted.

Usage

```
twot.permutation(x1 = DAAG::two65$ambient, x2 = DAAG::two65$heated, nsim = 2000,
plotit = TRUE)
```

Arguments

`x1`	Sample 1
`x2`	Sample 2
`nsim`	Number of simulations
`plotit`	If TRUE, the permutation density will be plotted

Details

Suppose we have n1 values in one group and n2 in a second, with n = n1 + n2. The permutation distribution results from taking all possible samples of n2 values from the total of n values.

Value

The p-value for the test of the hypothesis that the mean of `x1` differs from `x2`

Author(s)

J.H. Maindonald

References

Good, P. 2000. Permutation Tests. Springer, New York.

Examples

```
twot.permutation()
```

`twotPermutation` *Two Sample Permutation Test*

Description

This function computes the p-value for the two sample t-test using a permutation test. The permutation density can also be plotted.

Usage

```
twotPermutation(x1 = DAAG::two65$ambient, x2 = DAAG::two65$heated, nsim = 2000,
plotit = TRUE)
```

Arguments

`x1`	Sample 1
`x2`	Sample 2
`nsim`	Number of simulations
`plotit`	If TRUE, the permutation density will be plotted

Details

Suppose we have n1 values in one group and n2 in a second, with n = n1 + n2. The permutation distribution results from taking all possible samples of n2 values from the total of n values.

Value

The p-value for the test of the hypothesis that the mean of `x1` differs from `x2`

Author(s)

J.H. Maindonald

References

Good, P. 2000. Permutation Tests. Springer, New York.

Examples

```
twotPermutation()
```

vif — *Variance Inflation Factors*

Description

Variance inflation factors are computed for the standard errors of linear model coefficient estimates.

Usage

```
vif(obj, digits=5)
```

Arguments

obj	A lm object
digits	Number of digits

Value

A vector of variance inflation factors corresponding to the coefficient estimates given in the lm object.

Author(s)

J.H. Maindonald

See Also

lm

Examples

```
litters.lm <- lm(brainwt ~ bodywt + lsize, data = litters)
vif(litters.lm)

carprice1.lm <- lm(gpm100 ~ Type+Min.Price+Price+Max.Price+Range.Price,
    data=carprice)
vif(carprice1.lm)

carprice.lm <- lm(gpm100 ~ Type + Price, data = carprice)
vif(carprice1.lm)
```

`vince111b` *Averages by block of corn yields, for treatment 111 only*

Description

These data frames have averages by blocks (parcels) for the treatment `111`.

Usage

```
vince111b
```

Format

A data frame with 36 observations on 8 variables.

site a factor with levels `AGSV` `CASV` `CPSV` `LPSV` `MPSV` `OOSV` `OTSV` `SSSV` `UISV`

parcel a factor with levels `I` `II` `III` `IV`

code a numeric vector

island a numeric vector

id a numeric vector

plot a numeric vector

trt a numeric vector

harvwt a numeric vector

Source

Andrews DF; Herzberg AM, 1985. Data. A Collection of Problems from Many Fields for the Student and Research Worker. Springer-Verlag. (pp. 339-353)

`vlt` *Video Lottery Terminal Data*

Description

Data on objects appearing in three windows on a video lottery terminal, together with the prize payout (usually 0). Observations were taken on two successive days in late 1994 at a hotel lounge north of Winnipeg, Manitoba. Each observation cost 25 cents (Canadian). The game played was ‘Double Diamond’.

Usage

```
vlt
```

Format

This data frame contains the following columns:

window1 object appearing in the first window.

window2 object appearing in the second window.

window3 object appearing in the third window.

prize cash prize awarded (in Canadian dollars).

night 1, if observation was taken on day 1; 2, if observation was taken on day 2.

Details

At each play, each of three windows shows one of 7 possible objects. Apparently, the three windows are independent of each other, and the objects should appear with equal probability across the three windows. The objects are coded as follows: blank (0), single bar (1), double bar (2), triple bar (3), double diamond (5), cherries (6), and the numeral "7" (7).

Prizes (in quarters) are awarded according to the following scheme: 800 (5-5-5), 80 (7-7-7), 40 (3-3-3), 25 (2-2-2), 10 (1-1-1), 10 (6-6-6), 5 (2 "6"'s), 2 (1 "6") and 5 (any combination of "1", "2" and "3"). In addition, a "5" doubles any winning combination, e.g. (5-3-3) pays 80 and (5-3-5) pays 160.

Source

Braun, W. J. (1995) An illustration of bootstrapping using video lottery terminal data. Journal of Statistics Education http://www.amstat.org/publications/jse/v3n2/datasets.braun.html

Examples

```
vlt.stk <- stack(vlt[,1:3])
table(vlt.stk)
```

---

wages1833 — *Wages of Lancashire Cotton Factory Workers in 1833*

Description

The `wages1833` data frame gives the wages of Lancashire cotton factory workers in 1833.

Usage

```
wages1833
```

Format

This data frame contains the following columns:

age age in years

mnum number of male workers

mwage average wage of male workers

fnum number of female workers

fwage average wage of female workers

Source

Boot, H.M. 1995. How Skilled Were the Lancashire Cotton Factory Workers in 1833? Economic History Review 48: 283-303.

Examples

```
attach(wages1833)
plot(mwage~age,ylim=range(c(mwage,fwage[fwage>0])))
points(fwage[fwage>0]~age[fwage>0],pch=15,col="red")
lines(lowess(age,mwage))
lines(lowess(age[fwage>0],fwage[fwage>0]),col="red")
```

---

whoops *Deaths from whooping cough, in London*

Description

Deaths from whooping cough, in London from 1740 to 1881.

Usage

```
data(whoops)
```

Format

This is a multiple time series consisting of 3 series: `wcough`, `ratio`, and `alldeaths`.

Source

Guy, W. A. 1882. Two hundred and fifty years of small pox in London. Journal of the Royal Statistical Society 399-443.

References

Lancaster, H. O. 1990. Expectations of Life. Springer.

Examples

```
data(whoops)
str(whoops)
plot(whoops)
```

`worldRecords` *Record times for track and road races, at August 9th 2006*

Description

Record times for track and road races, at August 9th 2006

Usage

```
data(worldRecords)
```

Format

A data frame with 40 observations on the following 9 variables.

`Distance` distance in kilometers

`roadORtrack` a factor with levels `road` `track`

`Place` place; a character vector

`Time` time in minutes

`Date` a Date

Details

For further details, and some additional details, see the web site that is the source of the data.

Source

http://www.gbrathletics.com/wrec.htm

Examples

```
data(worldRecords)
library(lattice)
xyplot(log(Time) ~ log(Distance), groups=roadORtrack, data=worldRecords)
xyplot(log(Time) ~ log(Distance), groups=roadORtrack, data=worldRecords,
       type=c("p","r"))
xyplot(log(Time) ~ log(Distance), groups=roadORtrack, data=worldRecords,
       type=c("p","smooth"))
```

zzDAAGxdb *List, each of whose elements hold rows of a file, in character format*

Description

This is the default alternative database for use with the function `datafile`, which uses elements of this list to place files in the working directory. The names of the list elements are bestTimes and bostonc.

Usage

```
data(zzDAAGxdb)
```

Format

Successive elements in this list hold character vectors from which the corresponding files can be readily generated.

Details

The web site given as the source of the data has additional information on the bestTimes data. Records are as at August 7 2006.

Source

http://www.gbrathletics.com/wrec.htm (bestTimes)

http://lib.stat.cmu.edu/datasets/ (bostonc)

References

Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978. corrected by Kelley Pace (kpace@unix1.sncc.lsu.edu)

Examples

```
data(zzDAAGxdb)
names(zzDAAGxdb)
```

Index