Package 'kulife'

October 1, 2013

Version 0.1-14

Title Datasets and functions from the (now non-existing) Faculty of Life Sciences, University of Copenhagen

Author Claus Ekstrom <ekstrom@sund.ku.dk>, Ib M. Skovgaard <ims@life.ku.dk>, Torben Martinussen <tma@sund.ku.dk>

Maintainer Claus Ekstrom <ekstrom@sund.ku.dk>

Suggests lme4, XML

Description Provides various functions and data sets from experiments at the Faculty of Life Sciences, University of Copenhagen. This package will be discontinued and archived, and the functions and datasets will be maintained and updated in the MESS package

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2013-10-01 19:30:50

R topics documented:

auc *Compute the area under the curve for two vectors.*

Description

Compute the area under the curve using linear interpolation for two vectors where one corresponds to the x values and the other corresponds to the y values.

Usage

```
auc(x, y, from = min(x), to = max(x), ...)
```

Arguments

x	a numeric vector of x values.
y	a numeric vector of y values of the same length as x.
from	The value from where to start calculating the area under the curve. Defaults to the smallest x value.
to	The value from where to end the calculation of the area under the curve. Defaults to the smallest y value.
...	additional arguments passed on to approxfun. In particular rule can be set to determine how values outside the range of x is handled.

Details

auc is implemented using the approx function together with the composite trapezoid rule. approx creates a function that performs the linear interpolation between points and the trapezoid rule calculates the numerical integral, and by combining these we can handle unsorted time values, missing observations, ties for the time values, and integrating over part of the area or even outside the area.

Value

The value of the area under the curve.

Author(s)

Claus Ekstrom <ekstrom@life.ku.dk>

See Also

approx

Examples

```
x <- 1:4
y <- c(0, 1, 1, 5)
auc(x, y)

# AUC from 0 to max(x) where we allow for extrapolation
auc(x, y, from=0, rule=2)

# Use value 0 to the left
auc(x, y, from=0, rule=2, yleft=0)

# Use 1/2 to the left
auc(x, y, from=0, rule=2, yleft=.5)
```

---

bees *Bee data. Number of different types of bees caught.*

Description

Number of different types of bees caught in plates of different colours. There are four locations and within each location there are three replicates consisting of three plates of the three different colours (yellow, white and blue). Data are collected at 5 different dates over the summer season. Only data from one date available until data has been published.

Usage

```
data(bees)
```

Format

A data frame with 72 observations on the following 7 variables.

`Locality` a factor with levels `Havreholm` `Kragevig` `Saltrup` `Svaerdborg`. Four different localities in Denmark.

`Replicate` a factor with levels `A` `B` `C`

`Color` a factor with levels `Blue` `White` `Yellow`. Colour of plates

`Time` a factor with levels `july1` `july14` `june17` `june3` `june6`. Data collected at different dates in summer season. Only one day is present in the current data frame until the full data has been released.

`Type` a factor with levels `Bumblebees` `Solitary`. Type of bee.

`Number` a numeric vector. The response variable with number of bees catched.

`id` a numeric vector. The id of the clusters (each containg three plates).

Source

Data were kindly provided by Casper Ingerslev Henriksen, Department of Agricultural Sciences, KU-LIFE. Added by Torben Martinussen <tma@life.ku.dk>

Examples

```
data(bees)
model <- glm(Number ~ Locality + Type*Color,
             family=poisson, data=bees)
```

`clotting` *Blood clotting for 158 rats*

Description

Blood clotting activity (PCA) is measured for 158 Norway rats from two locations just before (baseline) and four days after injection of an anticoagulant (bromadiolone). Normally this would cause reduced blood clotting after 4 days compared to the baseline, but these rats are known to possess anticoagulent resistence to varying extent. The purpose is to relate anticoagulent resistence to gender and location and perhaps weight. Dose of injection is, however, admistered according to weight and gender.

Usage

```
data(clotting)
```

Format

A data frame with 158 observations on the following 6 variables.

`rat` a numeric vector

`locality` a factor with levels `Loc1` `Loc2`

`sex` a factor with levels `F` `M`

`weight` a numeric vector

`PCA0` a numeric vector with percent blood clotting activity at baseline

`PCA4` a numeric vector with percent blood clotting activity on day 4

Source

Ann-Charlotte Heiberg, project at The Royal Veterinary and Agricultural University, 1999.
Added by Ib M. Skovgaard <ims@life.ku.dk>

Examples

```
data(clotting)
dim(clotting)
head(clotting)
day0= transform(clotting, day=0, pca=PCA0)
day4= transform(clotting, day=4, pca=PCA4)
day.both= rbind(day0,day4)
m1= lm(pca ~ rat + day*locality + day*sex, data=day.both)
anova(m1)
```

```
 summary(m1)
 m2= lm(pca ~ rat + day, data=day.both)
 anova(m2)
## Log transformation suggested.
## Random effect of rat.
## maybe str(clotting) ; plot(clotting) ...
```

---

greenland *Average yearly summer air temperature for Tasiilaq, Greenland*

Description

Average yearly summer (June, July, August) air temperature for Tasiilaq, Greenland

Usage

```
data(greenland)
```

Format

A data frame with 51 observations on the following 2 variables.

`year` year

`airtemp` average air temperature (degrees Celcius)

Source

Data provided by Sebastian Mernild.
Originally obtained from http://www.dmi.dk/dmi/index/gronland/vejrarkiv-gl.htm.
Added by Claus Ekstrom <ekstrom@life.ku.dk>

References

Aktuelt Naturvidenskab september 2010.
http://aktuelnaturvidenskab.dk/fileadmin/an/nr-4/an4_2010gletscher.pdf

Examples

```
data(greenland)
model <- lm(airtemp ~ year, data=greenland)
plot(greenland$year, greenland$airtemp, xlab="Year", ylab="Air temperature")
abline(model, col="red")
```

`kulife.colors` *Color palette for faculty of Life Sciences, University of Copenhagen*

Description

Sets the color palette to match the design of the faculty of Life Sciences, University of Copenhagen

Usage

```
kulife.colors(n = 4, KUgrey = FALSE)
```

Arguments

`n`	Positive integer. Number of brown tone colors in the palette.
`KUgrey`	Logical. Should the grey University of Copenhagen color be included in the palette.

Value

Returns the color palette.

Author(s)

Claus Ekstrom <ekstrom@life.ku.dk>

References

http://designguide.ku.dk

See Also

`palette`

Examples

```
kucols <- kulife.colors(3)
pie(1:4, col=kucols)
```

qpcr *Gene expression from real-time quantitative PCR*

Description

Gene expression levels from real-time quantitative polymerase chain reaction (qPCR) experiments on two different plant lines. Each line was used for 7 experiments each with 45 cycles.

Usage

```
data(qpcr)
```

Format

A data frame with 630 observations on the following 4 variables.

`flour`	numeric	Fluorescence level
`line`	factor	Plant lines `rnt` (mutant) and `wt` (wildtype)
`cycle`	numeric	Cycle number for the experiment
`transcript`	factor	Transcript used for the different runs

Source

Data provided by Kirsten Jorgensen <kij@life.ku.dk>.
Added by Claus Ekstrom <ekstrom@life.ku.dk>

References

Morant, M. et al. (2010). Metabolomic, Transcriptional, Hormonal and Signaling Cross-Talk in Superroot2. *Molecular Plant*. 3, p.192–211.

Examples

```
data(qpcr)

#
# Analyze a single run for the wt line, transcript 1
#
run1 <- subset(qpcr, transcript==1 & line=="wt")

model <- nls(flour ~ fmax/(1+exp(-(cycle-c)/b))+fb,
             start=list(c=25, b=1, fmax=100, fb=0), data=run1)

print(model)

plot(run1$cycle, run1$flour, xlab="Cycle", ylab="Fluorescence")
lines(run1$cycle, predict(model))
```

rainman *Perception of points in a swarm*

Description

Five raters were asked to guess the number of points in a swarm for 10 different figures (which - unknown to the raters - were each repeated three times).

Usage

```
data(rainman)
```

Format

A data frame with 30 observations on the following 6 variables.

`SAND` The true number of points in the swarm. Each picture is replicated thrice

`ME` Ratings from judge 1

`TM` Ratings from judge 2

`AJ` Ratings from judge 3

`BM` Ratings from judge 4

`LO` Ratings from judge 5

Details

The raters har approximately 10 seconds to judge each picture, and the thought it was 30 different pictures. Before starting the experiment they were shown 6 (unrelated) pictures and were told the number of points in each of those pictures. The SAND column contains the picture id and the true number of points in the swarm.

Source

Collected by Claus Ekstrom.

Examples

```
data(rainman)
long <- data.frame(stack(rainman[,2:6]), figure=factor(rep(rainman$SAND,5)))
figind <- interaction(long$figure,long$ind)
# Use a linear random effect model from the
# lme4 package if available
if(require(lme4)) {
  model <- lmer(values ~ (1|ind) + (1|figure) + (1|figind), data=long)
}

#
# Point swarms were generated by the following program
```

```
#

set.seed(2) # Original
npoints <- sample(4:30)*4
nplots <- 10
pdf(file="swarms.pdf", onefile=TRUE)

s1 <- sample(npoints[1:nplots])
print(s1)
for (i in 1:nplots) {
  n <- s1[i]
  set.seed(n)
  x <- runif(n)
  y <- runif(n)
  plot(x,y, xlim=c(-.15, 1.15), ylim=c(-.15, 1.15), pch=20, axes=FALSE,
       xlab="", ylab="")
}
s1 <- sample(npoints[1:nplots])
print(s1)
for (i in 1:nplots) {
  n <- s1[i]
  set.seed(n)
  x <- runif(n)
  y <- runif(n)
  plot(y,x, xlim=c(-.15, 1.15), ylim=c(-.15, 1.15), pch=20, axes=FALSE,
       xlab="", ylab="")
}
s1 <- sample(npoints[1:nplots])
print(s1)
for (i in 1:nplots) {
  n <- s1[i]
  set.seed(n)
  x <- runif(n)
  y <- runif(n)
  plot(-x,y, xlim=c(-1.15, .15), ylim=c(-.15, 1.15), pch=20, axes=FALSE,
       xlab="", ylab="")
}
dev.off()
```

`rootonorm` *Hanging rootogram for normal distribution*

Description

Create a hanging rootogram for a quantitative numeric vector and compare it to a Gaussian distribution.

Usage

```
rootonorm(x, breaks = "Sturges", type = c("hanging", "deviation"),
```

```
scale = c("sqrt", "raw"), zeroline = TRUE,
linecol = "red", rectcol = "lightgrey",
xlab = xname, ylab = "Sqrt(frequency)",
yaxt = "n", ylim = NULL,
mu = mean(x), s = sd(x), gap = 0.1, ...)
```

Arguments

x	a numeric vector of values for which the rootogram is desired
breaks	Either the character string 'Sturges' to use Sturges' algorithm to decide the number of breaks or a positive integer that sets the number of breaks.
type	if "hanging" then a hanging rootogram is plotted, and if "deviation" then deviations from zero are plotted.
scale	The type of transformation. Defaults to "sqrt" which takes square roots of the frequencies. "raw" yields untransformed frequencies.
zeroline	logical; if TRUE a horizontal line is added at zero.
linecol	The color of the density line for the normal distribution. The default is to make a red density line.
rectcol	a colour to be used to fill the bars. The default of lightgray yields lightgray bars.
xlab, ylab	plot labels. The xlab and ylab refer to the x and y axes respectively
yaxt	Should y axis text be printed. Defaults to n.
ylim	the range of y values with sensible defaults.
mu	the mean of the Gaussian distribution. Defaults to the sample mean of x.
s	the standard deivation of the Gaussian distribution. Defaults to the sample std.dev. of x.
gap	The distance between the rectangles in the histogram.
...	further arguments and graphical parameters passed to plot.

Details

The mean and standard deviation of the Gaussian distribution are calculated from the observed data unless the mu and s arguments are given.

Value

Returns a vector of counts of each bar. This may be changed in the future. The plot is the primary output of the function.

Author(s)

Claus Ekstrom <ekstrom@life.ku.dk>

References

Tukey, J. W. 1972. *Some Graphic and Semigraphic Displays*. In *Statistical Papers in Honor of George W. Snedecor*, p. 293-316.

Examples

```
oldpar <- par()
par(mfrow=c(2,2))
rootonorm(rnorm(200))
rootonorm(rnorm(200), type="deviation", scale="raw")
rootonorm(rnorm(200), mu=1)
rootonorm(rexp(200), mu=1)
par(oldpar)
```

---

superroot2 *Gene expression data from two-color dye-swap experiment*

Description

Gene expression levels from two-color dye-swap experiment on 6 microarrays. Arrays 1 and 2 represent the first biological sample (ie, the first dye swap), 3 and 4 the second, and arrays 5 and 6 the third.

Usage

```
data(superroot2)
```

Format

A data frame with 258000 observations on the following 5 variables.

`color` a factor with levels `green` `red` representing the dye used for the gene expression

`array` a factor with levels `1` `2` `3` `4` `5` `6` corresponding to the 6 arrays

`gene` a factor with 21500 levels representing the genes on the arrays

`plant` a factor with levels `rnt` `wt` for the two types of plants: runts and wild type

`signal` a numeric vector with the gene expression level (normalized but not log transformed)

Source

Data provided by Soren Bak <bak@life.ku.dk>.
Added by Claus Ekstrom <ekstrom@life.ku.dk>

References

Morant, M. et al. (2010). Metabolomic, Transcriptional, Hormonal and Signaling Cross-Talk in Superroot2. *Molecular Plant*. 3, p.192–211.

Examples

```
data(superroot2)
# Select one gene
g1 <- superroot2[superroot2$gene=="AT2G24000.1",]
model <- lm(log(signal) ~ plant + color + array, data=g1)
summary(model)
```

`write.xml` *Write a data frame in XML format*

Description

Writes the data frame to a file in the XML format.

Usage

```
write.xml(data, file = NULL)
```

Arguments

`data`	the data frame object to save.
`file`	a file name to be written to.

Details

This function requires the **XML** package to be installed to function properly.

Value

'`NULL`'

Author(s)

Claus Ekstrom <ekstrom@life.ku.dk> based on work by Duncan Temple Lang.

Examples

```
data(trees)
write.xml(trees, file="mydata.xml")
```

Index