Package 'doBy'

March 31, 2016

Version 4.5-15

Title Groupwise Statistics, LSmeans, Linear Contrasts, Utilities

Author Søren Højsgaard <sorenh@math.aau.dk> and Ulrich Halekoh <uhalekoh@health.sdu.dk>

Maintainer Søren Højsgaard <sorenh@math.aau.dk>

Description The facilities can roughly be grouped as:
1) Facilities for groupwise computations of summary statistics and other facilities for working with grouped data: 'do' something to data stratified 'by' some variables.
2) LSmeans (least-squares means), general linear contrasts.
3) Miscellaneous other utilities.

Encoding latin1

URL http://people.math.aau.dk/~sorenh/software/doBy/

ZipData no

License GPL (>= 2)

Depends R (>= 3.2.0), methods

Imports MASS, Matrix

Suggests pbkrtest (>= 0.4-6), ggplot2, multcomp, geepack, lme4, survival

NeedsCompilation no

Repository CRAN

Date/Publication 2016-03-31 17:38:50

R topics documented:

`beets` *Yield and sugar percentage in sugar beets from a split plot experiment.*

Description

Data is obtained from a split plot experiment. There are 3 blocks and in each of these the harvest time defines the "whole plot" and the sowing time defines the "split plot". Each plot was $25m^2$ and the yield is recorded in kg. See 'details' for the experimental layout.

Usage

```
data(beets)
```

Format

The format is: chr "beets"

Details

```
Experimental plan
Sowing times            1        4. april
                        2       12. april
                        3       21. april
                        4       29. april
                        5       18. may
Harvest times           1        2. october
                        2       21. october
Plot allocation:
              Block 1     Block 2     Block 3
            +-----------|-----------|-----------+
      Plot  | 1 1 1 1 1 | 2 2 2 2 2 | 1 1 1 1 1 | Harvest time
       1-15 | 3 4 5 2 1 | 3 2 4 5 1 | 5 2 3 4 1 | Sowing time
            |-----------|-----------|-----------|
      Plot  | 2 2 2 2 2 | 1 1 1 1 1 | 2 2 2 2 2 | Harvest time
      16-30 | 2 1 5 4 3 | 4 1 3 2 5 | 1 4 3 2 5 | Sowing time
            +-----------|-----------|-----------+
```

Examples

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

beets$bh <- with(beets, interaction(block, harvest))
summary(aov(yield~block+sow+harvest+Error(bh), beets))
summary(aov(sugpct~block+sow+harvest+Error(bh), beets))
```

---

breastcancer *Gene expression signatures for p53 mutation status in 250 breast cancer samples*

Description

Perturbations of the p53 pathway are associated with more aggressive and therapeutically refractory tumours. We preprocessed the data using Robust Multichip Analysis (RMA). Dataset has been truncated to the 1000 most informative genes (as selected by Wilcoxon test statistics) to simplify computation. The genes have been standardised to have zero mean and unit variance (i.e. z-scored).

Usage

```
data(breastcancer)
```

Format

A data frame with 250 observations on the following 1001 variables.

`A.1053_at` a numeric vector

A.200039_s_at a numeric vector

A.200053_at a numeric vector

A.200079_s_at a numeric vector

A.200628_s_at a numeric vector

A.200639_s_at a numeric vector

A.200670_at a numeric vector

A.200687_s_at a numeric vector

A.200710_at a numeric vector

A.200740_s_at a numeric vector

A.200773_x_at a numeric vector

A.200783_s_at a numeric vector

A.200795_at a numeric vector

A.200810_s_at a numeric vector

A.200811_at a numeric vector

A.200822_x_at a numeric vector

A.200840_at a numeric vector

A.200853_at a numeric vector

A.200854_at a numeric vector

A.200855_at a numeric vector

A.200996_at a numeric vector

A.201041_s_at a numeric vector

A.201077_s_at a numeric vector

A.201088_at a numeric vector

A.201104_x_at a numeric vector

A.201114_x_at a numeric vector

A.201115_at a numeric vector

A.201124_at a numeric vector

A.201140_s_at a numeric vector

A.201170_s_at a numeric vector

A.201195_s_at a numeric vector

A.201196_s_at a numeric vector

A.201197_at a numeric vector

A.201201_at a numeric vector

A.201202_at a numeric vector

A.201236_s_at a numeric vector

A.201263_at a numeric vector

A.201266_at a numeric vector

A.201281_at a numeric vector

A.201291_s_at a numeric vector

A.201292_at a numeric vector

A.201327_s_at a numeric vector

A.201342_at a numeric vector

A.201384_s_at a numeric vector

A.201394_s_at a numeric vector

A.201395_at a numeric vector

A.201397_at a numeric vector

A.201455_s_at a numeric vector

A.201479_at a numeric vector

A.201508_at a numeric vector

A.201557_at a numeric vector

A.201584_s_at a numeric vector

A.201591_s_at a numeric vector

A.201663_s_at a numeric vector

A.201664_at a numeric vector

A.201685_s_at a numeric vector

A.201694_s_at a numeric vector

A.201710_at a numeric vector

A.201755_at a numeric vector

A.201761_at a numeric vector

A.201770_at a numeric vector

A.201790_s_at a numeric vector

A.201791_s_at a numeric vector

A.201795_at a numeric vector

A.201833_at a numeric vector

A.201890_at a numeric vector

A.201896_s_at a numeric vector

A.201897_s_at a numeric vector

A.201930_at a numeric vector

A.201970_s_at a numeric vector

A.201977_s_at a numeric vector

A.201978_s_at a numeric vector

A.202011_at a numeric vector

A.202088_at a numeric vector

A.202095_s_at a numeric vector

A.202105_at a numeric vector

A.202107_s_at a numeric vector

A.202109_at a numeric vector

A.202117_at a numeric vector

A.202174_s_at a numeric vector

A.202188_at a numeric vector

A.202200_s_at a numeric vector

A.202233_s_at a numeric vector

A.202240_at a numeric vector

A.202270_at a numeric vector

A.202276_at a numeric vector

A.202307_s_at a numeric vector

A.202338_at a numeric vector

A.202370_s_at a numeric vector

A.202371_at a numeric vector

A.202409_at a numeric vector

A.202487_s_at a numeric vector

A.202503_s_at a numeric vector

A.202580_x_at a numeric vector

A.202589_at a numeric vector

A.202590_s_at a numeric vector

A.202613_at a numeric vector

A.202666_s_at a numeric vector

A.202690_s_at a numeric vector

A.202705_at a numeric vector

A.202725_at a numeric vector

A.202754_at a numeric vector

A.202779_s_at a numeric vector

A.202808_at a numeric vector

A.202815_s_at a numeric vector

A.202854_at a numeric vector

A.202858_at a numeric vector

A.202870_s_at a numeric vector

A.202954_at a numeric vector

A.202962_at a numeric vector

A.203022_at a numeric vector

A.203046_s_at a numeric vector

A.203066_at a numeric vector

A.203071_at a numeric vector

A.203126_at a numeric vector

A.203130_s_at a numeric vector

A.203139_at a numeric vector

A.203145_at a numeric vector

A.203187_at a numeric vector

A.203188_at a numeric vector

A.203208_s_at a numeric vector

A.203213_at a numeric vector

A.203214_x_at a numeric vector

A.203223_at a numeric vector

A.203249_at a numeric vector

A.203265_s_at a numeric vector

A.203266_s_at a numeric vector

A.203276_at a numeric vector

A.203287_at a numeric vector

A.203296_s_at a numeric vector

A.203345_s_at a numeric vector

A.203347_s_at a numeric vector

A.203358_s_at a numeric vector

A.203362_s_at a numeric vector

A.203380_x_at a numeric vector

A.203405_at a numeric vector

A.203418_at a numeric vector

A.203422_at a numeric vector

A.203428_s_at a numeric vector

A.203432_at a numeric vector

A.203438_at a numeric vector

A.203439_s_at a numeric vector

A.203463_s_at a numeric vector

A.203484_at a numeric vector

A.203554_x_at a numeric vector

A.203560_at a numeric vector

A.203594_at a numeric vector

A.203625_x_at a numeric vector

A.203693_s_at a numeric vector

A.203696_s_at a numeric vector

A.203702_s_at a numeric vector

A.203744_at a numeric vector

A.203755_at a numeric vector

A.203764_at a numeric vector

A.203799_at a numeric vector

A.203801_at a numeric vector

A.203805_s_at a numeric vector

A.203806_s_at a numeric vector

A.203856_at a numeric vector

A.203859_s_at a numeric vector

A.203906_at a numeric vector

A.203929_s_at a numeric vector

A.203930_s_at a numeric vector

A.203967_at a numeric vector

A.203968_s_at a numeric vector

A.204020_at a numeric vector

A.204023_at a numeric vector

A.204026_s_at a numeric vector

A.204033_at a numeric vector

A.204045_at a numeric vector

A.204072_s_at a numeric vector

A.204092_s_at a numeric vector

A.204115_at a numeric vector

A.204126_s_at a numeric vector

A.204127_at a numeric vector

A.204146_at a numeric vector

A.204162_at a numeric vector

A.204170_s_at a numeric vector

A.204203_at a numeric vector

A.204217_s_at a numeric vector

A.204244_s_at a numeric vector

A.204267_x_at a numeric vector

A.204290_s_at a numeric vector

A.204315_s_at a numeric vector

A.204317_at a numeric vector

A.204318_s_at a numeric vector

A.204441_s_at a numeric vector

A.204444_at a numeric vector

A.204482_at a numeric vector

A.204510_at a numeric vector

A.204533_at a numeric vector

A.204558_at a numeric vector

A.204592_at a numeric vector

A.204603_at a numeric vector

A.204640_s_at a numeric vector

A.204641_at a numeric vector

A.204649_at a numeric vector

A.204675_at a numeric vector

A.204686_at a numeric vector

A.204695_at a numeric vector

A.204702_s_at a numeric vector

A.204709_s_at a numeric vector

A.204732_s_at a numeric vector

A.204766_s_at a numeric vector

A.204767_s_at a numeric vector

A.204768_s_at a numeric vector

A.204800_s_at a numeric vector

A.204817_at a numeric vector

A.204822_at a numeric vector

A.204825_at a numeric vector

A.204863_s_at a numeric vector

A.204881_s_at a numeric vector

A.204962_s_at a numeric vector

A.205024_s_at a numeric vector

A.205034_at a numeric vector

A.205046_at a numeric vector

A.205074_at a numeric vector

A.205158_at a numeric vector

A.205167_s_at a numeric vector

A.205186_at a numeric vector

A.205214_at a numeric vector

A.205225_at a numeric vector

A.205240_at a numeric vector

A.205279_s_at a numeric vector

A.205280_at a numeric vector

A.205282_at a numeric vector

A.205296_at a numeric vector

A.205339_at a numeric vector

A.205354_at a numeric vector

A.205392_s_at a numeric vector

A.205393_s_at a numeric vector

A.205394_at a numeric vector

A.205436_s_at a numeric vector

A.205440_s_at a numeric vector

A.205471_s_at a numeric vector

A.205472_s_at a numeric vector

A.205569_at a numeric vector

A.205611_at a numeric vector

A.205628_at a numeric vector

A.205644_s_at a numeric vector

A.205672_at a numeric vector

A.205710_at a numeric vector

A.205711_x_at a numeric vector

A.205733_at a numeric vector

A.205776_at a numeric vector

A.205779_at a numeric vector

A.205794_s_at a numeric vector

A.205883_at a numeric vector

A.205898_at a numeric vector

A.205933_at a numeric vector

A.205943_at a numeric vector

A.205967_at a numeric vector

A.206055_s_at a numeric vector

A.206074_s_at a numeric vector

A.206091_at a numeric vector

A.206102_at a numeric vector

A.206134_at a numeric vector

A.206197_at a numeric vector

A.206245_s_at a numeric vector

A.206364_at a numeric vector

A.206481_s_at a numeric vector

A.206499_s_at a numeric vector

A.206600_s_at a numeric vector

A.206632_s_at a numeric vector

A.206702_at a numeric vector

A.206794_at a numeric vector

A.206869_at a numeric vector

A.207165_at a numeric vector

A.207542_s_at a numeric vector

A.207783_x_at a numeric vector

A.207788_s_at a numeric vector

A.207828_s_at a numeric vector

A.208029_s_at a numeric vector

A.208079_s_at a numeric vector

A.208103_s_at a numeric vector

A.208140_s_at a numeric vector

A.208184_s_at a numeric vector

A.208190_s_at a numeric vector

A.208308_s_at a numeric vector

A.208374_s_at a numeric vector

A.208433_s_at a numeric vector

A.208503_s_at a numeric vector

A.208511_at a numeric vector

A.208517_x_at a numeric vector

A.208614_s_at a numeric vector

A.208627_s_at a numeric vector

A.208628_s_at a numeric vector

A.208682_s_at a numeric vector

A.208691_at a numeric vector

A.208693_s_at a numeric vector

A.208696_at a numeric vector

A.208709_s_at a numeric vector

A.208718_at a numeric vector

A.208767_s_at a numeric vector

A.208794_s_at a numeric vector

A.208804_s_at a numeric vector

A.208807_s_at a numeric vector

A.208870_x_at a numeric vector

A.208905_at a numeric vector

A.208952_s_at a numeric vector

A.208969_at a numeric vector

A.209002_s_at a numeric vector

A.209019_s_at a numeric vector

A.209026_x_at a numeric vector

A.209068_at a numeric vector

A.209070_s_at a numeric vector

A.209071_s_at a numeric vector

A.209075_s_at a numeric vector

A.209172_s_at a numeric vector

A.209189_at a numeric vector

A.209195_s_at a numeric vector

A.209218_at a numeric vector

A.209311_at a numeric vector

A.209336_at a numeric vector

A.209341_s_at a numeric vector

A.209358_at a numeric vector

A.209375_at a numeric vector

A.209408_at a numeric vector

A.209459_s_at a numeric vector

A.209460_at a numeric vector

A.209464_at a numeric vector

A.209482_at a numeric vector

A.209523_at a numeric vector

A.209603_at a numeric vector

A.209604_s_at a numeric vector

A.209608_s_at a numeric vector

A.209623_at a numeric vector

A.209625_at a numeric vector

A.209642_at a numeric vector

A.209680_s_at a numeric vector

A.209709_s_at a numeric vector

A.209714_s_at a numeric vector

A.209737_at a numeric vector

A.209740_s_at a numeric vector

A.209747_at a numeric vector

A.209753_s_at a numeric vector

A.209773_s_at a numeric vector

A.209825_s_at a numeric vector

A.209832_s_at a numeric vector

A.209869_at a numeric vector

A.209891_at a numeric vector

A.209897_s_at a numeric vector

A.210052_s_at a numeric vector

A.210092_at a numeric vector

A.210093_s_at a numeric vector

A.210108_at a numeric vector

A.210137_s_at a numeric vector

A.210163_at a numeric vector

A.210346_s_at a numeric vector

A.210365_at a numeric vector

A.210416_s_at a numeric vector

A.210466_s_at a numeric vector

A.210559_s_at a numeric vector

A.210567_s_at a numeric vector

A.210766_s_at a numeric vector

A.210778_s_at a numeric vector

A.210821_x_at a numeric vector

A.210881_s_at a numeric vector

A.211000_s_at a numeric vector

A.211034_s_at a numeric vector

A.211042_x_at a numeric vector

A.211072_x_at a numeric vector

A.211080_s_at a numeric vector

A.211378_x_at a numeric vector

A.211519_s_at a numeric vector

A.211714_x_at a numeric vector

A.211725_s_at a numeric vector

A.211765_x_at a numeric vector

A.211939_x_at a numeric vector

A.211943_x_at a numeric vector

A.211967_at a numeric vector

A.211986_at a numeric vector

A.212020_s_at a numeric vector

A.212022_s_at a numeric vector

A.212023_s_at a numeric vector

A.212048_s_at a numeric vector

A.212151_at a numeric vector

A.212195_at a numeric vector

A.212196_at a numeric vector

A.212209_at a numeric vector

A.212219_at a numeric vector

A.212228_s_at a numeric vector

A.212242_at a numeric vector

A.212247_at a numeric vector

A.212266_s_at a numeric vector

A.212284_x_at a numeric vector

A.212296_at a numeric vector

A.212299_at a numeric vector

A.212319_at a numeric vector

A.212419_at a numeric vector

A.212423_at a numeric vector

A.212430_at a numeric vector

A.212448_at a numeric vector

A.212492_s_at a numeric vector

A.212494_at a numeric vector

A.212495_at a numeric vector

A.212496_s_at a numeric vector

A.212541_at a numeric vector

A.212581_x_at a numeric vector

A.212660_at a numeric vector

A.212680_x_at a numeric vector

A.212695_at a numeric vector

A.212705_x_at a numeric vector

A.212708_at a numeric vector

A.212744_at a numeric vector

A.212745_s_at a numeric vector

A.212779_at a numeric vector

A.212815_at a numeric vector

A.212846_at a numeric vector

A.212865_s_at a numeric vector

A.212869_x_at a numeric vector

A.212914_at a numeric vector

A.212936_at a numeric vector

A.212948_at a numeric vector

A.212949_at a numeric vector

A.212956_at a numeric vector

A.212970_at a numeric vector

A.212978_at a numeric vector

A.212985_at a numeric vector

A.213007_at a numeric vector

A.213008_at a numeric vector

A.213011_s_at a numeric vector

A.213018_at a numeric vector

A.213058_at a numeric vector

A.213063_at a numeric vector

A.213088_s_at a numeric vector

A.213101_s_at a numeric vector

A.213102_at a numeric vector

A.213103_at a numeric vector

A.213137_s_at a numeric vector

A.213175_s_at a numeric vector

A.213195_at a numeric vector

A.213224_s_at a numeric vector

A.213226_at a numeric vector

A.213283_s_at a numeric vector

A.213285_at a numeric vector

A.213366_x_at a numeric vector

A.213397_x_at a numeric vector

A.213427_at a numeric vector

A.213451_x_at a numeric vector

A.213453_x_at a numeric vector

A.213464_at a numeric vector

A.213519_s_at a numeric vector

A.213520_at a numeric vector

A.213523_at a numeric vector

A.213541_s_at a numeric vector

A.213571_s_at a numeric vector

A.213577_at a numeric vector

A.213590_at a numeric vector

A.213599_at a numeric vector

A.213627_at a numeric vector

A.213671_s_at a numeric vector

A.213702_x_at a numeric vector

A.213720_s_at a numeric vector

A.213748_at a numeric vector

A.213832_at a numeric vector

A.213911_s_at a numeric vector

A.213933_at a numeric vector

A.214039_s_at a numeric vector

A.214053_at a numeric vector

A.214061_at a numeric vector

A.214062_x_at a numeric vector

A.214077_x_at a numeric vector

A.214095_at a numeric vector

A.214096_s_at a numeric vector

A.214101_s_at a numeric vector

A.214144_at a numeric vector

A.214164_x_at a numeric vector

A.214188_at a numeric vector

A.214269_at a numeric vector

A.214431_at a numeric vector

A.214435_x_at a numeric vector

A.214437_s_at a numeric vector

A.214440_at a numeric vector

A.214552_s_at a numeric vector

A.214581_x_at a numeric vector

A.214657_s_at a numeric vector

A.214710_s_at a numeric vector

A.214728_x_at a numeric vector

A.214745_at a numeric vector

A.214919_s_at a numeric vector

A.215014_at a numeric vector

A.215146_s_at a numeric vector

A.215223_s_at a numeric vector

A.215300_s_at a numeric vector

A.215303_at a numeric vector

A.215304_at a numeric vector

A.215616_s_at a numeric vector

A.215722_s_at a numeric vector

A.215867_x_at a numeric vector

A.215942_s_at a numeric vector

A.216000_at a numeric vector

A.216088_s_at a numeric vector

A.216109_at a numeric vector

A.216237_s_at a numeric vector

A.216264_s_at a numeric vector

A.216465_at a numeric vector

A.216515_x_at a numeric vector

A.216520_s_at a numeric vector

A.216841_s_at a numeric vector

A.216885_s_at a numeric vector

A.216952_s_at a numeric vector

A.216977_x_at a numeric vector

A.217297_s_at a numeric vector

A.217346_at a numeric vector

A.217398_x_at a numeric vector

A.217427_s_at a numeric vector

A.217499_x_at a numeric vector

A.217714_x_at a numeric vector

A.217733_s_at a numeric vector

A.217755_at a numeric vector

A.217834_s_at a numeric vector

A.217838_s_at a numeric vector

A.217844_at a numeric vector

A.217889_s_at a numeric vector

A.217906_at a numeric vector

A.217925_s_at a numeric vector

A.217938_s_at a numeric vector

A.217990_at a numeric vector

A.218007_s_at a numeric vector

A.218009_s_at a numeric vector

A.218039_at a numeric vector

A.218065_s_at a numeric vector

A.218074_at a numeric vector

A.218080_x_at a numeric vector

A.218115_at a numeric vector

A.218131_s_at a numeric vector

A.218146_at a numeric vector

A.218193_s_at a numeric vector

A.218211_s_at a numeric vector

A.218213_s_at a numeric vector

A.218219_s_at a numeric vector

A.218239_s_at a numeric vector

A.218259_at a numeric vector

A.218260_at a numeric vector

A.218295_s_at a numeric vector

A.218308_at a numeric vector

A.218349_s_at a numeric vector

A.218350_s_at a numeric vector

A.218353_at a numeric vector

A.218355_at a numeric vector

A.218374_s_at a numeric vector

A.218384_at a numeric vector

A.218418_s_at a numeric vector

A.218437_s_at a numeric vector

A.218447_at a numeric vector

A.218471_s_at a numeric vector

A.218474_s_at a numeric vector

A.218483_s_at a numeric vector

A.218497_s_at a numeric vector

A.218499_at a numeric vector

A.218512_at a numeric vector

A.218538_s_at a numeric vector

A.218542_at a numeric vector

A.218552_at a numeric vector

A.218584_at a numeric vector

A.218585_s_at a numeric vector

A.218588_s_at a numeric vector

A.218662_s_at a numeric vector

A.218663_at a numeric vector

A.218684_at a numeric vector

A.218726_at a numeric vector

A.218728_s_at a numeric vector

A.218741_at a numeric vector

A.218755_at a numeric vector

A.218769_s_at a numeric vector

A.218795_at a numeric vector

A.218834_s_at a numeric vector

A.218844_at a numeric vector

A.218856_at a numeric vector

A.218875_s_at a numeric vector

A.218877_s_at a numeric vector

A.218883_s_at a numeric vector

A.218894_s_at a numeric vector

A.218936_s_at a numeric vector

A.218966_at a numeric vector

A.218976_at a numeric vector

A.218982_s_at a numeric vector

A.218987_at a numeric vector

A.219000_s_at a numeric vector

A.219004_s_at a numeric vector

A.219010_at a numeric vector

A.219031_s_at a numeric vector

A.219061_s_at a numeric vector

A.219065_s_at a numeric vector

A.219091_s_at a numeric vector

A.219105_x_at a numeric vector

A.219132_at a numeric vector

A.219148_at a numeric vector

A.219197_s_at a numeric vector

A.219252_s_at a numeric vector

A.219286_s_at a numeric vector

A.219304_s_at a numeric vector

A.219306_at a numeric vector

A.219396_s_at a numeric vector

A.219413_at a numeric vector

A.219417_s_at a numeric vector

A.219436_s_at a numeric vector

A.219440_at a numeric vector

A.219455_at a numeric vector

A.219463_at a numeric vector

A.219469_at a numeric vector

A.219490_s_at a numeric vector

A.219493_at a numeric vector

A.219494_at a numeric vector

A.219510_at a numeric vector

A.219555_s_at a numeric vector

A.219588_s_at a numeric vector

A.219646_at a numeric vector

A.219650_at a numeric vector

A.219686_at a numeric vector

A.219687_at a numeric vector

A.219689_at a numeric vector

A.219713_at a numeric vector

A.219787_s_at a numeric vector

A.219833_s_at a numeric vector

A.219918_s_at a numeric vector

A.219922_s_at a numeric vector

A.219978_s_at a numeric vector

A.219990_at a numeric vector

A.220011_at a numeric vector

A.220060_s_at a numeric vector

A.220094_s_at a numeric vector

A.220173_at a numeric vector

A.220239_at a numeric vector

A.220276_at a numeric vector

A.220324_at a numeric vector

A.220540_at a numeric vector

A.220651_s_at a numeric vector

A.220789_s_at a numeric vector

A.220865_s_at a numeric vector

A.220917_s_at a numeric vector

A.221203_s_at a numeric vector

A.221207_s_at a numeric vector

A.221221_s_at a numeric vector

A.221258_s_at a numeric vector

A.221272_s_at a numeric vector

A.221275_s_at a numeric vector

A.221276_s_at a numeric vector

A.221436_s_at a numeric vector

A.221505_at a numeric vector

A.221520_s_at a numeric vector

A.221521_s_at a numeric vector

A.221562_s_at a numeric vector

A.221588_x_at a numeric vector

A.221590_s_at a numeric vector

A.221676_s_at a numeric vector

A.221677_s_at a numeric vector

A.221685_s_at a numeric vector

A.221740_x_at a numeric vector

A.221856_s_at a numeric vector

A.221913_at a numeric vector

A.221922_at a numeric vector

A.221951_at a numeric vector

A.222010_at a numeric vector

A.222011_s_at a numeric vector

A.222039_at a numeric vector

A.222043_at a numeric vector

A.222077_s_at a numeric vector

A.222118_at a numeric vector

A.222195_s_at a numeric vector

A.36129_at a numeric vector

A.37152_at a numeric vector

A.38158_at a numeric vector

A.39854_r_at a numeric vector

A.40016_g_at a numeric vector

A.40093_at a numeric vector

A.45297_at a numeric vector

A.45633_at a numeric vector

A.57703_at a numeric vector

A.AFFX.HUMGAPDH.M33197_3_at a numeric vector

A.AFFX.HUMGAPDH.M33197_M_at a numeric vector

B.200039_s_at a numeric vector

B.200076_s_at a numeric vector

B.200079_s_at a numeric vector

B.200086_s_at a numeric vector

B.222396_at a numeric vector

B.222423_at a numeric vector

B.222453_at a numeric vector

B.222602_at a numeric vector

B.222606_at a numeric vector

B.222608_s_at a numeric vector

B.222640_at a numeric vector

B.222680_s_at a numeric vector

B.222740_at a numeric vector

B.222752_s_at a numeric vector

B.222767_s_at a numeric vector

B.222820_at a numeric vector

B.222835_at a numeric vector

B.222843_at a numeric vector

B.222848_at a numeric vector

B.222889_at a numeric vector

B.222958_s_at a numeric vector

B.222962_s_at a numeric vector

B.223054_at a numeric vector

B.223056_s_at a numeric vector

B.223062_s_at a numeric vector

B.223068_at a numeric vector

B.223096_at a numeric vector

B.223100_s_at a numeric vector

B.223119_s_at a numeric vector

B.223126_s_at a numeric vector

B.223186_at a numeric vector

B.223204_at a numeric vector

B.223225_s_at a numeric vector

B.223229_at a numeric vector

B.223234_at a numeric vector

B.223274_at a numeric vector

B.223307_at a numeric vector

B.223315_at a numeric vector

B.223348_x_at a numeric vector

B.223361_at a numeric vector

B.223381_at a numeric vector

B.223387_at a numeric vector

B.223452_s_at a numeric vector

B.223480_s_at a numeric vector

B.223491_at a numeric vector

B.223515_s_at a numeric vector

B.223556_at a numeric vector

B.223570_at a numeric vector

B.223666_at a numeric vector

B.223700_at a numeric vector

B.223864_at a numeric vector

B.224217_s_at a numeric vector

B.224320_s_at a numeric vector

B.224333_s_at a numeric vector

B.224428_s_at a numeric vector

B.224468_s_at a numeric vector

B.224471_s_at a numeric vector

B.224521_s_at a numeric vector

B.224523_s_at a numeric vector

B.224566_at a numeric vector

B.224578_at a numeric vector

B.224610_at a numeric vector

B.224674_at a numeric vector

B.224686_x_at a numeric vector

B.224699_s_at a numeric vector

B.224752_at a numeric vector

B.224753_at a numeric vector

B.224903_at a numeric vector

B.224944_at a numeric vector

B.225004_at a numeric vector
B.225064_at a numeric vector
B.225071_at a numeric vector
B.225083_at a numeric vector
B.225092_at a numeric vector
B.225099_at a numeric vector
B.225100_at a numeric vector
B.225142_at a numeric vector
B.225162_at a numeric vector
B.225191_at a numeric vector
B.225268_at a numeric vector
B.225291_at a numeric vector
B.225300_at a numeric vector
B.225309_at a numeric vector
B.225327_at a numeric vector
B.225365_at a numeric vector
B.225379_at a numeric vector
B.225402_at a numeric vector
B.225454_at a numeric vector
B.225468_at a numeric vector
B.225501_at a numeric vector
B.225592_at a numeric vector
B.225611_at a numeric vector
B.225613_at a numeric vector
B.225629_s_at a numeric vector
B.225655_at a numeric vector
B.225656_at a numeric vector
B.225676_s_at a numeric vector
B.225686_at a numeric vector
B.225687_at a numeric vector
B.225723_at a numeric vector
B.225748_at a numeric vector
B.225777_at a numeric vector
B.225804_at a numeric vector
B.225834_at a numeric vector
B.225841_at a numeric vector
B.225866_at a numeric vector

B.226030_at a numeric vector

B.226108_at a numeric vector

B.226118_at a numeric vector

B.226198_at a numeric vector

B.226274_at a numeric vector

B.226298_at a numeric vector

B.226303_at a numeric vector

B.226308_at a numeric vector

B.226344_at a numeric vector

B.226346_at a numeric vector

B.226349_at a numeric vector

B.226358_at a numeric vector

B.226410_at a numeric vector

B.226437_at a numeric vector

B.226439_s_at a numeric vector

B.226456_at a numeric vector

B.226466_s_at a numeric vector

B.226473_at a numeric vector

B.226506_at a numeric vector

B.226519_s_at a numeric vector

B.226522_at a numeric vector

B.226582_at a numeric vector

B.226661_at a numeric vector

B.226831_at a numeric vector

B.226833_at a numeric vector

B.226846_at a numeric vector

B.226914_at a numeric vector

B.226915_s_at a numeric vector

B.226936_at a numeric vector

B.226974_at a numeric vector

B.226977_at a numeric vector

B.226980_at a numeric vector

B.226992_at a numeric vector

B.227021_at a numeric vector

B.227047_x_at a numeric vector

B.227068_at a numeric vector

B.227081_at a numeric vector

B.227103_s_at a numeric vector

B.227165_at a numeric vector

B.227182_at a numeric vector

B.227198_at a numeric vector

B.227211_at a numeric vector

B.227212_s_at a numeric vector

B.227227_at a numeric vector

B.227232_at a numeric vector

B.227279_at a numeric vector

B.227350_at a numeric vector

B.227363_s_at a numeric vector

B.227419_x_at a numeric vector

B.227423_at a numeric vector

B.227436_at a numeric vector

B.227478_at a numeric vector

B.227512_at a numeric vector

B.227533_at a numeric vector

B.227651_at a numeric vector

B.227700_x_at a numeric vector

B.227703_s_at a numeric vector

B.227718_at a numeric vector

B.227762_at a numeric vector

B.227809_at a numeric vector

B.227811_at a numeric vector

B.227873_at a numeric vector

B.227874_at a numeric vector

B.227920_at a numeric vector

B.227928_at a numeric vector

B.227982_at a numeric vector

B.228044_at a numeric vector

B.228069_at a numeric vector

B.228081_at a numeric vector

B.228252_at a numeric vector

B.228273_at a numeric vector

B.228281_at a numeric vector

B.228318_s_at a numeric vector

B.228323_at a numeric vector

B.228327_x_at a numeric vector

B.228361_at a numeric vector

B.228468_at a numeric vector

B.228469_at a numeric vector

B.228476_at a numeric vector

B.228504_at a numeric vector

B.228505_s_at a numeric vector

B.228528_at a numeric vector

B.228550_at a numeric vector

B.228554_at a numeric vector

B.228559_at a numeric vector

B.228560_at a numeric vector

B.228597_at a numeric vector

B.228692_at a numeric vector

B.228718_at a numeric vector

B.228729_at a numeric vector

B.228730_s_at a numeric vector

B.228750_at a numeric vector

B.228799_at a numeric vector

B.228811_at a numeric vector

B.228854_at a numeric vector

B.228868_x_at a numeric vector

B.228915_at a numeric vector

B.228931_at a numeric vector

B.228955_at a numeric vector

B.229030_at a numeric vector

B.229062_at a numeric vector

B.229097_at a numeric vector

B.229127_at a numeric vector

B.229150_at a numeric vector

B.229169_at a numeric vector

B.229170_s_at a numeric vector

B.229181_s_at a numeric vector

B.229342_at a numeric vector

B.229381_at a numeric vector

B.229437_at a numeric vector

B.229466_at a numeric vector

B.229490_s_at a numeric vector

B.229538_s_at a numeric vector

B.229551_x_at a numeric vector

B.229610_at a numeric vector

B.229764_at a numeric vector

B.229886_at a numeric vector

B.230021_at a numeric vector

B.230123_at a numeric vector

B.230142_s_at a numeric vector

B.230165_at a numeric vector

B.230250_at a numeric vector

B.230451_at a numeric vector

B.230469_at a numeric vector

B.230863_at a numeric vector

B.230966_at a numeric vector

B.231002_s_at a numeric vector

B.231034_s_at a numeric vector

B.231472_at a numeric vector

B.231577_s_at a numeric vector

B.232065_x_at a numeric vector

B.232210_at a numeric vector

B.232238_at a numeric vector

B.232278_s_at a numeric vector

B.232286_at a numeric vector

B.232307_at a numeric vector

B.232398_at a numeric vector

B.232459_at a numeric vector

B.232596_at a numeric vector

B.232855_at a numeric vector

B.232944_at a numeric vector

B.232968_at a numeric vector

B.233110_s_at a numeric vector

B.233388_at a numeric vector

B.233413_at a numeric vector

B.233520_s_at a numeric vector

B.234222_at a numeric vector

B.234294_x_at a numeric vector

B.234749_s_at a numeric vector

B.234863_x_at a numeric vector

B.234944_s_at a numeric vector

B.234954_at a numeric vector

B.234992_x_at a numeric vector

B.235040_at a numeric vector

B.235046_at a numeric vector

B.235117_at a numeric vector

B.235124_at a numeric vector

B.235181_at a numeric vector

B.235193_at a numeric vector

B.235363_at a numeric vector

B.235369_at a numeric vector

B.235425_at a numeric vector

B.235542_at a numeric vector

B.235545_at a numeric vector

B.235547_at a numeric vector

B.235570_at a numeric vector

B.235572_at a numeric vector

B.235609_at a numeric vector

B.235662_at a numeric vector

B.235709_at a numeric vector

B.235771_at a numeric vector

B.235786_at a numeric vector

B.235789_at a numeric vector

B.235800_at a numeric vector

B.236050_at a numeric vector

B.236064_at a numeric vector

B.236312_at a numeric vector

B.236641_at a numeric vector

B.236787_at a numeric vector

B.236953_s_at a numeric vector

B.237086_at a numeric vector

B.237168_at a numeric vector

B.237301_at a numeric vector

B.237339_at a numeric vector

B.238001_at a numeric vector

B.238045_at a numeric vector
B.238075_at a numeric vector
B.238077_at a numeric vector
B.238116_at a numeric vector
B.238425_at a numeric vector
B.238447_at a numeric vector
B.238656_at a numeric vector
B.238756_at a numeric vector
B.238781_at a numeric vector
B.238898_at a numeric vector
B.239002_at a numeric vector
B.239349_at a numeric vector
B.239432_at a numeric vector
B.239758_at a numeric vector
B.239890_s_at a numeric vector
B.240099_at a numeric vector
B.240422_at a numeric vector
B.241310_at a numeric vector
B.241408_at a numeric vector
B.241577_at a numeric vector
B.241789_at a numeric vector
B.241937_s_at a numeric vector
B.242218_at a numeric vector
B.242255_at a numeric vector
B.242323_at a numeric vector
B.242560_at a numeric vector
B.242657_at a numeric vector
B.242890_at a numeric vector
B.243754_at a numeric vector
B.243806_at a numeric vector
B.244116_at a numeric vector
B.244375_at a numeric vector
B.244571_s_at a numeric vector
B.244677_at a numeric vector
B.244696_at a numeric vector
B.AFFX.HUMGAPDH.M33197_3_at a numeric vector
B.AFFX.HUMGAPDH.M33197_M_at a numeric vector
code a factor with levels case control

Details

The factor code defines whether there was a mutation in the p53 sequence (code=case) or not (code=control).

Source

Dr. Chris Holmes, c.holmes at stats dot. ox . ac .uk

References

Miller et al (2005, PubMed ID:16141321)

Examples

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

---

budworm *Effect of Insecticide on survivial of tobacco budworms*

Description

Number of killed budworms after exposure to an insecticide.

Usage

```
data(budworm)
```

Format

This data frame contains 12 rows and 4 columns:

sex: sex of the budworm

dose: dose of the insecticide trans-cypermethrin in μg

ndead: budworms killed in a trial

ntotal: total number of budworms exposed per trial

Details

Mortality of the moth tobacco budworm 'Heliothis virescens' for 6 doses of the pyrethroid trans-cypermethrin differentiated with respect to sex.

Source

Collet, D. (1991) Modelling Binary Data, Chapman & Hall, London, Example 3.7

References

Venables, W.N; Ripley, B.D.(1999) Modern Applied Statistics with S-Plus, Heidelberg, Springer, 3rd edition, chapter 7.2

Examples

```
data(budworm)
## function to caclulate the empirical logits
empirical.logit<- function(nevent,ntotal) {
  y<-log ((nevent+0.5)/(ntotal-nevent+0.5))
  y
}

## plot the empirical logits against log-dose

log.dose  <- log(budworm$dose)
emp.logit <- empirical.logit(budworm$ndead,budworm$ntotal)
plot(log.dose,emp.logit,type='n',xlab='log-dose',ylab='emprirical logit')
title('budworm: emprirical logits of probability to die ')

male   <- budworm$sex=='male'
female <- budworm$sex=='female'
lines(log.dose[male],emp.logit[male],type='b',lty=1,col=1)
lines(log.dose[female],emp.logit[female],type='b',lty=2,col=2)
legend(0.5,2,legend=c('male','female'),lty=c(1,2),col=c(1,2))
```

---

carcass *Lean meat contents of 344 pig carcasses*

Description

Measurement of lean meat percentage of 344 pig carcasses together with auxillary information collected at three Danish slaughter houses

Usage

```
data(carcass)
data(carcassall)
```

Format

carcassall: A data frame with 344 observations on the following 17 variables.

`weight` Weight of carcass

`lengthc` Length of carcass from back toe to head (when the carcass hangs in the back legs)

`lengthf` Length of carcass from back toe to front leg (that is, to the shoulder)

`lengthp` Length of carcass from back toe to the pelvic bone

`Fat02, Fat03, Fat11, Fat12, Fat13, Fat14, Fat16` Thickness of fat layer at different locations on the back of the carcass (FatXX refers to thickness at (or rather next to) rib no. XX. Notice that 02 is closest to the head

`Meat11, Meat12, Meat13` Thickness of meat layer at different locations on the back of the carcass, see description above

`LeanMeat` Lean meat percentage determined by dissection

`slhouse` Slaughter house; a factor with levels `a b c`

`sex` Sex of the pig; a factor with `a b c`. Notice that it is no an error to have three levels; the third level refers to castrates

carcass: Contains only the variables Fat11, Fat12, Fat13, Meat11, Meat12, Meat13, LeanMeat

Source

Busk, H., Olsen, E. V., Brøndum, J. (1999) Determination of lean meat in pig carcasses with the Autofom classification system, Meat Science, 52, 307-314

`codstom` *Diet of Atlantic cod in the Gulf of St. Lawrence (Canada)*

Description

Stomach content data for Atlantic cod (Gadus morhua) in the Gulf of St.Lawrence, Eastern Canada. Note: many prey items were of no interest for this analysis and were regrouped into the "Other" category.

Usage

```
data(codstom)
```

Format

A data frame with 10000 observations on the following 10 variables.

`region` a factor with levels `SGSL NGSL` representing the southern and northern Gulf of St. Lawrence, respectively

`ship.type` a factor with levels `2 3 31 34 90 99`

`ship.id` a factor with levels `11558 11712 136148 136885 136902 137325 151225 151935 99433`

`trip` a factor with levels `10 11 12 179 1999 2 2001 20020808 3 4 5 6 7 8 88 9 95`

`set` a numeric vector

`fish.id` a numeric vector

`fish.length` a numeric vector, length in mm

`prey.mass` a numeric vector, mass of item in stomach, in g

`prey.type` a factor with levels `Ammodytes_sp Argis_dent Chion_opil Detritus Empty Eualus_fab Eualus_mac Gadus_mor Hyas_aran Hyas_coar Lebbeus_gro Lebbeus_pol Leptocl_mac Mallot_vil Megan_norv Ophiuroidea Other Paguridae Pandal_bor Pandal_mon Pasiph_mult Sabin_sept Sebastes_sp Them_abys Them_comp Them_lib`

Details

Cod are collected either by contracted commerical fishing vessels (ship.type 90 or 99) or by research vessels. Commercial vessels are identified by a unique ship.id.

Either one research vessel or several commercial vessels conduct a survey (trip), during which a trawl, gillnets or hooked lines are set several times. Most trips are random stratified surveys (depth-based stratification).

Each trip takes place within one of the regions. The trip label is only guaranteed to be unique within a region and the set label is only guaranteed to be unique within a trip.

For each fish caught, the fish.length is recorded and the fish is allocated a fish.id, but the fish.id is only guaranteed to be unique within a set. A subset of the fish caught are selected for stomach analysis (stratified random selection according to fish length; unit of stratification is the set for research surveys, the combination ship.id and stratum for surveys conducted by commercial vessels, although strata are not shown in codstom).

The basic experimental unit in this data set is a cod stomach (one stomach per fish). Each stomach is uniquely identified by a combination of region, ship.type, ship.id, trip, set, and fish.id.

For each prey item found in a stomach, the species and mass of the prey item are recorded, so there can be multiple observations per stomach. There may also be several prey items with the same prey.type in the one stomach (for example many prey.types have been recoded Other, which produced many instances of Other in the same stomach).

If a stomach is empty, a single observation is recorded with prey.type Empty and a prey.mass of zero.

Source

Small subset from a larger dataset (more stomachs, more variables, more prey.types) collected by D. Chabot and M. Hanson, Fisheries & Oceans Canada (chabotd@dfo-mpo.gc.ca).

Examples

```
data(codstom)
str(codstom)
# removes multiple occurences of same prey.type in stomachs
codstom1 <- summaryBy(prey.mass ~
                      region+ship.type+ship.id+trip+set+fish.id+prey.type,
                      data = codstom, id = ~fish.length,
                      keep.names=TRUE, FUN = sum)

# keeps a single line per stomach with the total mass of stomach content
codstom2 <- summaryBy(prey.mass ~ region+ship.type+ship.id+trip+set+fish.id,
                      data = codstom, id = ~fish.length,
                      keep.names=TRUE, FUN = sum)

# mean prey mass per stomach for each trip
codstom3 <- summaryBy(prey.mass ~ region+ship.type+ship.id+trip,
                      data = codstom2, keep.names=TRUE, FUN = mean)

## Not run:
# wide version, one line per stomach, one column per prey type
```

```
library(reshape)
codstom4 <- melt(codstom, id = c(1:7, 9))
codstom5 <- cast(codstom4,
                 region+ship.type+ship.id+trip+set+fish.id+fish.length ~
                 prey.type, sum)
k <- length(names(codstom5))
prey_col <- 8:k
out <- codstom5[,prey_col]
out[is.na(out)] <- 0
codstom5[,prey_col] <- out
codstom5$total.content <- rowSums(codstom5[, prey_col])

## End(Not run)
```

---

createFunBy *A template function for creating groupwise functions*

Description

A template function for creating groupwise functions

Usage

```
formulaFunBy(formula, group, data, FUN, class = NULL, ...)
xyFunBy(xy, group, data, FUN, class = NULL, ...)
```

Arguments

formula	A formula of the form `y ~ x` (which must be variable names in data).
xy	A character vector with one or two elements (which must be variable names in data).
group	A right hand sided formula or a character vector defining the grouping of data
data	A data frame
FUN	The function to be applied
class	The class to give the result of the returned value of the created function.
...	Further arguments passed on to FUN

Value

A function

Note

This function is a recent addition and has not been thoroughly tested. Please report bugs.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

parseGroupFormula

Examples

```
## Example: Create a function for creating groupwise t-tests

mydata <- data.frame(y=rnorm(32), x=rnorm(32),
g1=factor(rep(c(1,2),each=16)), g2=factor(rep(c(1,2), each=8)),
g3=factor(rep(c(1,2),each=4)))


t.testBy1 <- function(formula, group, data, ...){
  formulaFunBy(formula, group, data, FUN=t.test, class="t.testBy1", ...)
}

t.testBy2 <- function(formula, group, data, ...){
  xyFunBy(formula, group, data, FUN=t.test, class="t.testBy1", ...)
}


t.testBy1(y~g1, ~g2+g3, data=mydata)
t.testBy2(y~x, ~g2+g3, data=mydata)
```

---

crimeRate *crimeRate*

Description

Crime rates per 100,000 inhabitants in states of the USA for different crime types.

Usage

```
data(crimeRate)
```

Format

This data frame contains:

State: State of the USA

Murder: crime of murder

Rape:

Robbery:

Assault:

Burglary: residential theft

Larceny: unlawful taking of personal property (pocket picking)

AutoTheft:

Examples

```
data(crimeRate)
```

`descStat` *Computing simple descriptive statistics of a numeric vector.*

Description

Computing simple descriptive statistics of a numeric vector – not unlike what proc means of SAS does

Usage

```
descStat(x, na.rm = TRUE)
```

Arguments

`x`	A numeric vector
`na.rm`	Should missing values be removed

Value

A vector with named elements.

Author(s)

Gregor Gorjanc; gregor.gorjanc <at> bf.uni-lj.si

See Also

`summaryBy`

Examples

```
x <- c(1,2,3,4,NA,NaN)
descStat(x)
```

`dietox` *Growth curves of pigs in a 3x3 factorial experiment*

Description

The `dietox` data frame has 861 rows and 7 columns.

Data contains weight of slaughter pigs measured weekly for 12 weeks. Data also contains the startweight (i.e. the weight at week 1). The treatments are 3 different levels of Evit = vitamin E (dose: 0, 100, 200 mg dl-alpha-tocopheryl acetat /kg feed) in combination with 3 different levels of Cu=copper (dose: 0, 35, 175 mg/kg feed) in the feed. The cumulated feed intake is also recorded. The pigs are littermates.

Usage

```
data(dietox)
```

Format

This data frame contains the following columns:

Weight Weight

Feed Cumulated feed intake

Time Time (in weeks) in the experiment

Pig Id of each pig

Evit Vitamin E dose

Cu Copper dose

Start Start weight in experiment, i.e. weight at week 1.

Litter Id of litter of each pig

Source

Lauridsen, C., Højsgaard, S.,Sørensen, M.T. C. (1999) Influence of Dietary Rapeseed Oli, Vitamin E, and Copper on Performance and Antioxidant and Oxidative Status of Pigs. J. Anim. Sci.77:906-916

Examples

```
data(dietox)
str(dietox) ;
plot(dietox)
```

doBy *Various utilitie. Functions for creating groupwise calculations etc.; calculation of least-squares means; miscellaneous utilities*

Description

The core doBy functions were developed to make it easy to split data into groups (defined by the levels of a set of factors) and performing some actions on each of these groups. Thus, these functions mimic what can be achieved using the BY statement in various SAS procedures.

In addition hereto the doBy package containts various other utilities.

Details

Functions summaryBy, splitBy, orderBy, sampleBy, transformBy etc. are the doBy functions.

`linest()` calculates linear estimates based on a matrix for various model objects. The `esticon()` function has a similar functionality, but it will be removed at some point of time.

`LSmeans()` and `LSmatrix()` are used in connection with calculating least–squares means.

There are various other utility functions in the package.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

summaryBy, orderBy, transformBy, splitBy, sampleBy

Examples

```
data(dietox)

summaryBy(Weight+Feed~Evit+Cu+Time, data=dietox, FUN=c(mean,var), na.rm=TRUE, use="pair")

orderBy(~Time+Evit, data=dietox)

splitBy(formula = ~Evit+Cu, data = dietox)

sampleBy(formula = ~Evit+Cu, frac=.1, data = dietox)
```

esticon *Contrasts for lm, glm, lme, and geeglm objects*

Description

Computes linear functions (i.e. weighted sums) of the estimated regression parameters.

Can also test the hypothesis, that such a function is equal to a specific value.

Usage

```
esticon(obj, cm, beta0, conf.int = TRUE, level=0.95, joint.test = FALSE,...)
```

Arguments

obj	Regression object (of type lm, glm, lme, geeglm)
cm	Matrix specifying linear functions of the regresson parameters (one linear function per row). The number of columns must match the number of fitted regression parameters in the model. See 'details' below.
beta0	A vector of numbers
conf.int	TRUE
level	The confidence level
joint.test	Logical value. If TRUE a 'joint' Wald test for the hypothesis L beta=beta0 is made. Default is that the 'row-wise' tests are made, i.e. (L beta)i=beta0i. If joint.test is TRUE, then no confidence inteval etc. is calculated.
...	Additional arguments; currently not used.

Details

Let the estimated parameters of the model be

$$latex$$

A linear function of the estimates is of the form

$$latex$$

where $latex$ is specified by the user.

The esticon function calculates c, its standard error and by default also a 95 pct confidence interval. It is sometimes of interest to test the hypothesis $latex$ for some value $latex$ given by the user. A test is provided for the hypothesis $latex$ but other values of $latex$ can be specified.

In general, one can specify r such linear functions at one time by speficying cm to be an $latex$ matrix where each row consists of p numbers $latex$. Default is then that $latex$ is a p vector of 0s but other values can be given.

It is possible to test simulatneously that all speficied linear functions are equal to the corresponding values in $latex$.

For computing contrasts among levels of a single factor, 'contrast.lm' may be more convenient.

Value

Returns a matrix with one row per linear function. Columns contain estimated coefficients, standard errors, t values, degrees of freedom, two-sided p-values, and the lower and upper endpoints of the 1-alpha confidence intervals.

Note

'esticon' works on geese/geeglm objects from the geepack package (for Generalized Estimating Equations), on 'lm' and 'glm' objects, and on 'gls' objects.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

Examples

```
data(iris)
lm1  <- lm(Sepal.Length~Sepal.Width+Species+Sepal.Width:Species, data=iris)
## Note that the setosa parameters are set to zero
coef(lm1)

## Estimate the intercept for versicolor
lambda1 <- c(1,0,1,0,0,0)
esticon(lm1,lambda1)

## Estimate the difference between versicolor and virgica intercept
## and test if the difference is 1
lambda2 <- c(0,1,-1,0,0,0)
esticon(lm1,lambda2,beta0=1)

## Do both estimates at one time
esticon(lm1,rbind(lambda1,lambda2),beta0=c(0,1))

## Make a combined test for that the difference between versicolor and virgica intercept
## and difference between versicolor and virginica slope is zero:
lambda3 <- c(0,0,0,0,1,-1)
esticon(lm1,rbind(lambda2,lambda3),joint.test=TRUE)

# Example using esticon on coxph objects (thanks to Alessandro A. Leidi).
# Using dataset 'veteran' in the survival package
# from the Veterans' Administration Lung Cancer study

if (require(survival)){
library(survival);
data(veteran)
sapply(veteran,class)
levels(veteran$celltype)
attach(veteran)
veteran.s<-Surv(time,status)
coxmod<-coxph(veteran.s~age+celltype+trt,method='breslow')
summary(coxmod)
```

```
# compare a subject 50 years old with celltype 1
# to a subject 70 years old with celltype 2
# both subjects on the same treatment
AvB<-c(-20,-1,0,0,0)

# compare a subject 40 years old with celltype 2 on treat=0
# to a subject 35 years old with celltype 3 on treat=1
CvB<-c(5,1,-1,0,-1)

esti<-esticon(coxmod,rbind(AvB,CvB))
esti
exp(esti[,c(2,7,8)])
}
```

`firstlastobs` *Locate the index of the first/last unique value*

Description

Locate the index of the first/last unique value in i) a vector or of a variable in a data frame.

Usage

```
## S3 method for class 'formula'
firstobs(formula, data=parent.frame(), ...)
## S3 method for class 'formula'
lastobs(formula, data=parent.frame(), ...)
firstobs(x, ...)
lastobs(x, ...)
```

Arguments

`x`	A vector
`formula`	A formula (only the first term is used, see 'details').
`data`	A data frame
`...`	Currently not used

Details

If writing ~a+b+c as formula, then only a is considered.

Value

A vector.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

Examples

```
x <- c(rep(1,5),rep(2,3),rep(3,7),rep(1,4))

firstobs(x)
lastobs(x)

data(dietox)

firstobs(~Pig, data=dietox)
lastobs(~Pig, data=dietox)
```

---

haldCement *Heat development in cement under hardening.*

Description

Heat development in cement under hardening related to the chemical composition.

Usage

```
data(haldCement)
```

Format

A data frame with 13 observations on the following 5 variables.

`x1` Percentage (weight) of [3Ca0][Al2O3]

`x2` Percentage (weight) of [3Cao][SiO2]

`x3` Percentage (weight) of [4Ca0][Al2O3][Fe03]

`x4` Percentage (weight) of [2Cao][SiO2]

`y` Heat development measured in calories per gram cement after 180 days

References

Anders Hald (1949); Statistiske Metoder; Akademisk Forlag (in Danish), page 509.

Examples

```
data(haldCement)

if( interactive() ){
pairs( haldCement )
}
m <- lm( y~x1+x2+x3+x4, data=haldCement )
summary( m )

# Notice: The model explains practically all variation in data;
# yet none of the explanatory variables appear to be statistically
# significant...
```

`is_estimable` *Determines if contrasts are estimable.*

Description

Determines if contrasts are estimable, that is, if the contrasts can be written as a linear function of the data.

Usage

```
is_estimable(K, null.basis)
```

Arguments

`K`	A matrix.
`null.basis`	A basis for a null space (can be found with `null_basis()`).

Details

Consider the setting $E(Y) = Xb$. A linear function of b, say $l'b$ is estimable if and only if there exists an r such that $r'X = l'$ or equivalently $l = X'r$. Hence l must be in the column space of X', i.e. in the orthogonal complement of the null space of X. Hence, with a basis B for the null space, `is_estimable()` checks if each row l of the matrix K is perpendicular to each column basis vector in B.

Value

A logical vector.

Author(s)

SÃ¸ren HÃ¸jsgaard, <sorenh@math.aau.dk>

References

http://web.mit.edu/18.06/www/Essays/newpaper_ver3.pdf

See Also

null_basis

Examples

```
## TO BE WRITTEN
```

---

lapplyBy *Formula based version of lapply*

Description

This function is a wrapper for calling lapply on the list resulting from first calling splitBy.

Usage

```
lapplyBy(formula, data = parent.frame(), FUN, keep.groupid = FALSE)
```

Arguments

formula	A formula describing how data should be split
data	A dataframe
FUN	A function to be applied to each element in the splitted list, see 'Examples' below.
keep.groupid	If TRUE, the grouping information is stored as a dataframe in an attribute called groupid.

Value

A list.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

orderBy, summaryBy, transformBy, splitBy,

Examples

```
data(dietox)

## Calculate weekwise feed efficiency = weight gain / feed intake
dietox <- orderBy(~Pig+Time, data=dietox)
v<-lapplyBy(~Pig, data=dietox, function(d) c(NA, diff(d$Weight)/diff(d$Feed)))
dietox$FE <- unlist(v)

## Technically this is the same as
dietox <- orderBy(~Pig+Time, data=dietox)
wdata <- splitBy(~Pig, data=dietox)
v <- lapply(wdata, function(d) c(NA, diff(d$Weight)/diff(d$Feed)))
dietox$FE <- unlist(v)
```

---

linest *Compute linear estimates*

Description

Compute linear estimates for a range of models. One example of linear estimates is population means (also known as LSMEANS).

Usage

```
linest(object, K=NULL, level=0.95, ...)
```

Arguments

object	Model object
K	Either NULL or a matrix with p columns where p is the number of parameters in the systematic effects in the model. If NULL then K is taken to be the p times p identity matrix
level	The level of the (asymptotic) confidence interval.
...	Additional arguments; currently not used.

Value

A dataframe with results from computing the contrasts.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

LSmeans LSmatrix

Examples

```
## Make balanced dataset
dat.bal <- expand.grid(list(AA=factor(1:2), BB=factor(1:3), CC=factor(1:3)))
dat.bal$y <- rnorm(nrow(dat.bal))

## Make unbalanced dataset
#   'BB' is nested within 'CC' so BB=1 is only found when CC=1
#   and BB=2,3 are found in each CC=2,3,4
dat.nst <- dat.bal
dat.nst$CC <-factor(c(1,1,2,2,2,2,1,1,3,3,3,3,1,1,4,4,4,4))

mod.bal  <- lm(y ~ AA + BB*CC,    data=dat.bal)
mod.nst  <- lm(y ~ AA + BB : CC, data=dat.nst)

K <- LSmatrix(mod.nst, effect=c("BB","CC"))
linest( mod.nst, K )
```

`lmBy` *List of lm objects with a common model*

Description

The data is split into strata according to the levels of the grouping factors and individual lm fits are obtained for each stratum.

Usage

```
lmBy(formula, data, id = NULL, ...)
```

Arguments

`formula`	A linear model formula object of the form y ~ x1+...+xn | g1+...gm. In the formula object, y represents the response, x1,...,xn the covariates, and the grouping factors specifying the partitioning of the data according to which different lm fits should be performed.
`data`	A dataframe
`id`	A formula describing variables from data which are to be available also in the output.
`...`	Additional arguments passed on to `lm()`.

Value

A list of lm fits.

Note

This is a recent addition to the package; please report bugs.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

Examples

```
bb <- lmBy(1/uptake~log(conc)|Treatment, data=CO2)

coef(bb)

fitted(bb)
residuals(bb)

summary(bb)
coef(summary(bb))
coef(summary(bb), simplify=TRUE)
```

LSmeans *Compute linear estimates, including LS-means (aka population means or marginal means)*

Description

Compute linear estimates for a range of models. One example of linear estimates is LS-means (least squares means, also known as population means and as marginal means).

Usage

```
LSmeans(object, effect = NULL, at = NULL, level=0.95,


      ...)
```

Arguments

object	Model object
effect	A vector of variables. For each configuration of these the estimate will be calculated.
at	A list of values of covariates (including levels of some factors) to be used in the calculations
level	The level of the (asymptotic) confidence interval.
...	Additional arguments; currently not used.

Details

There are restrictions on the formulas allowed in the model object. For example having `y ~ log(x)` will cause an error. Instead one must define the variable `logx = log(x)` and do `y~logx`.

Value

A dataframe with results from computing the contrasts.

Warning

Notice that `LSmeans` and `LSmatrix` fails if the model formula contains an offset (as one would have in connection with e.g. Poisson regression. It is on the todo-list to fix this

Note

The `LSmeans` method is a recent addition to the package, and it will eventually replace the `popMeans` method.

Please report unexpected behaviour.

Some of the code has been inspired by the **lsmeans** package.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

`LSmatrix` `linest`

Examples

```
## Two way anova:

data(warpbreaks)

m0 <- lm(breaks ~ wool + tension, data=warpbreaks)
m1 <- lm(breaks ~ wool * tension, data=warpbreaks)
LSmeans(m0)
LSmeans(m1)

## same as:
K <- LSmatrix(m0);K
linest(m0, K)
K <- LSmatrix(m1);K
linest(m1, K)

LSmatrix(m0, effect="wool")
LSmeans(m0, effect="wool")

LSmatrix(m1, effect="wool")
LSmeans(m1, effect="wool")

LSmatrix(m0, effect=c("wool","tension"))
LSmeans(m0, effect=c("wool","tension"))

LSmatrix(m1, effect=c("wool","tension"))
```

```
LSmeans(m1, effect=c("wool","tension"))


## Regression; two parallel regression lines:

data(Puromycin)

m0 <- lm(rate ~ state + log(conc), data=Puromycin)
## Can not use LSmeans / LSmatrix here because of
## the log-transformation. Instead we must do:
Puromycin$lconc <- log( Puromycin$conc )
m1 <- lm(rate ~ state + lconc, data=Puromycin)

LSmatrix(m1)
LSmeans(m1)

LSmatrix(m1, effect="state")
LSmeans(m1, effect="state")

LSmatrix(m1, effect="state", at=list(lconc=3))
LSmeans(m1, effect="state", at=list(lconc=3))

## Non estimable contrasts

## ## Make balanced dataset
dat.bal <- expand.grid(list(AA=factor(1:2), BB=factor(1:3),
                            CC=factor(1:3)))
dat.bal$y <- rnorm(nrow(dat.bal))

## ## Make unbalanced dataset
#      'BB' is nested within 'CC' so BB=1 is only found when CC=1
#       and BB=2,3 are found in each CC=2,3,4
dat.nst <- dat.bal
dat.nst$CC <-factor(c(1,1,2,2,2,2,1,1,3,3,3,3,1,1,4,4,4,4))

mod.bal  <- lm(y ~ AA + BB*CC,    data=dat.bal)
mod.nst  <- lm(y ~ AA + BB : CC, data=dat.nst)

LSmeans(mod.bal, effect=c("BB", "CC"))
LSmeans(mod.nst, effect=c("BB", "CC"))
LSmeans(mod.nst, at=list(BB=1, CC=1))

LSmeans(mod.nst, at=list(BB=1, CC=2))
## Above: NA's are correct; not an estimable function

if( require( lme4 )){
 warp.mm <- lmer(breaks ~ -1 + tension + (1|wool), data=warpbreaks)
 LSmeans(warp.mm, effect="tension")
 class(warp.mm)
 fixef(warp.mm)
 coef(summary(warp.mm))
 vcov(warp.mm)
 if (require(pbkrtest))
```

```
   vcovAdj(warp.mm)
}
```

`milkman` *Milk yield data for manually milked cows.*

Description

Milk yield data for cows milked manually twice a day (morning and evening).

Usage

```
data(milkman)
```

Format

A data frame with 161836 observations on the following 12 variables.

`cowno` a numeric vector; cow identification

`lactno` a numeric vector; lactation number

`ampm` a numeric vector; milking time: 1: morning; 2: evening

`dfc` a numeric vector; days from calving

`my` a numeric vector; milk yield (kg)

`fatpct` a numeric vector; fat percentage

`protpct` a numeric vector; protein percentage

`lactpct` a numeric vector; lactose percentage

`scc` a numeric vector; somatic cell counts

`race` a factor with levels `RDM` `Holstein` `Jersey`

`ecmy` a numeric vector; energy corrected milk

`cowlact` Combination of cowno and lactno; necessary because the same cow may appear more than once in the dataset (in different lactations)

Details

There are data for 222 cows. Some cows appear more than once in the dataset (in different lactations) and there are 288 different lactations.

Examples

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

NIRmilk *NIRmilk*

Description

Near infra red light (NIR) measurments are made at 152 wavelengths on 17 milk samples. While milk runs through a glass tube, infra red light is sent through the tube and the amount of light passing though the tube is measured at different wavelengths. Each milk sample was additionally analysed for fat, lactose, protein and drymatter.

Usage

```
data(NIRmilk)
```

Format

This data frame contains 18 rows and 158 columns. The first column is the sample number. The columns Xwww contains the infra red light amount at wavelength www. The response variables are fat, protein, lactose and dm (drymatter).

Details

PCA regression

Examples

```
data(NIRmilk)
```

null_basis *Finds the basis of the (right) null space.*

Description

Finds the basis of the (right) null space of a matrix, a vector (a 1-column matrix) or a model object for which a model matrix can be extracted.

Usage

```
null_basis(object)
```

Arguments

`object`	A matrix, a vector (a 1-column matrix) or a model object for which a model matrix can be extracted (using `model.matrix`).

Details

Finds basis for the (right) null space x : Mx = 0.

Value

A matrix (possibly with zero columns if the null space consists only of the zero vector).

Author(s)

SÃ¸ren HÃ¸jsgaard, <sorenh@math.aau.dk>

See Also

Null

Examples

```
M <- matrix(c(1,1,1,1,1,1,0,0,0,0,1,1), nrow=4)
null_basis(M)
MASS::Null(t(M))

M <- c(1,1,1,1)
null_basis(M)
MASS::Null(t(M))

m0 <- lm(breaks ~ wool + tension, data=warpbreaks)
null_basis(m0)
MASS::Null(t(model.matrix(m0)))

## Make balanced dataset
dat.bal   <- expand.grid(list(A=factor(1:2), B=factor(1:3), C=factor(1:3)))
dat.bal$y <- rnorm(nrow(dat.bal))

## Make unbalanced dataset: 'B' is nested within 'C' so B=1 is only
## found when C=1 and B=2,3 are found in each C=2,3,4
dat.nst <- dat.bal
dat.nst$C <-factor(c(1,1,2,2,2,2,1,1,3,3,3,3,1,1,4,4,4,4))
xtabs(y ~ C+B+A , data=dat.nst)

mod.bal  <- lm(y ~ A + B*C, data=dat.bal)
mod.nst  <- lm(y ~ A + B*C, data=dat.nst)

null_basis( mod.bal )
null_basis( mod.nst )

null_basis( model.matrix(mod.bal) )
null_basis( model.matrix(mod.nst) )

MASS::Null( t(model.matrix(mod.bal)) )
MASS::Null( t(model.matrix(mod.nst)) )
```

orderBy *Ordering (sorting) rows of a data frame*

Description

Ordering (sorting) rows of a data frame by the certain variables in the data frame. This function is essentially a wrapper for the order() function - the important difference being that variables to order by can be given by a model formula.

Usage

```
orderBy(formula, data)
```

Arguments

formula	The right hand side of a formula
data	A data frame

Details

The sign of the terms in the formula determines whether sorting should be ascending or decreasing; see examples below

Value

The ordered data frame

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk> and Kevin Wright

See Also

summaryBy, transformBy, splitBy, lapplyBy,

Examples

```
data(dietox)
orderBy(~Time+Evit, data=dietox)
## Sort decreasingly by Time
orderBy(~-Time+Evit, data=dietox)
```

parseGroupFormula *Extract components from a formula with "conditioning bar"*

Description

Extract components from a formula with the form `y~x1+...xn|g1+...gm`

Usage

```
parseGroupFormula(form)
```

Arguments

`form`	A formula of the form `y~x1+...xn|g1+...gm`

Value

If the formula is `y~x1+x2|g1+g2` the result is

`model`	`y~x1+x2`
`groups`	`g1+g2`
`groupFormula`	`~g1+g2`

Note

This function is a recent addition and has not been thoroughly tested. Please report bugs.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

Examples

```
gf<-parseGroupFormula(y~x1+x2|g1+g2)
```

potatoes *Weight and size of 20 potatoes*

Description

Weight and size of 20 potatoes. Weight in grams; size in milimeter. There are two sizes: `length` is the longest length and `width` is the shortest length across a potato.

Usage

```
data("potatoes")
```

Format

A data frame with 20 observations on the following 3 variables.

`weight` a numeric vector

`length` a numeric vector

`width` a numeric vector

Author(s)

SÃ¸ren HÃ¸jsgaard, <sorenh@math.aau.dk>

Source

My own garden; autumn 2015.

Examples

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

recodeVar *Recode values of a vector*

Description

Recodes a vector with values, say 1,2 to a variable with values, say 'a', 'b'

Usage

```
recodeVar(x, src, tgt, default=NULL, keep.na=TRUE)
```

Arguments

x	A vector; the variable to be recoded
src	The source values: a subset of the present values of x
tgt	The target values: the corresponding new values of x
default	Default target value for those values of x not listed in 'src'. When default=NULL, values of x which are not given in 'src' will be kept in the output.
keep.na	If TRUE then NA's in x will be retained in the output

Value

A vector

Warning

Care should be taken if x is a factor. A safe approach may be to convert x to a character vector using as.character.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

cut, factor, recodeVar

Examples

```
x <- c("dec","jan","feb","mar","apr","may")
src1 <- list(c("dec","jan","feb"), c("mar","apr","may"))
tgt1 <- list("winter","spring")
recodeVar(x,src=src1,tgt=tgt1)
#[1] "winter" "winter" "winter" "spring" "spring" "spring"

x <- c(rep(1:3,3))
#[1] 1 2 3 1 2 3 1 2 3

## Simple usage:
recodeVar(x, src=c(1,2), tgt=c("A","B"))
#[1] "A" "B" NA  "A" "B" NA  "A" "B" NA

## Here we need to use lists
recodeVar(x, src=list(c(1,2)), tgt=list("A"))
#[1] "A" "A" NA  "A" "A" NA  "A" "A" NA
recodeVar(x, src=list(c(1,2)), tgt=list("A"), default="L")
#[1] "A" "A" "L" "A" "A" "L" "A" "A" "L"
recodeVar(x, src=list(c(1,2),3), tgt=list("A","B"), default="L")
#[1] "A" "A" "B" "A" "A" "B" "A" "A" "B"

## Dealing with NA's in x
x<-c(NA,rep(1:3,3),NA)
```

```
#[1] NA  1  2  3  1  2  3  1  2  3 NA
recodeVar(x, src=list(c(1,2)), tgt=list("A"))
#[1] NA  "A" "A" NA  "A" "A" NA  "A" "A" NA  NA
recodeVar(x, src=list(c(1,2)), tgt=list("A"), default="L")
#[1] NA  "A" "A" "L" "A" "A" "L" "A" "A" "L" NA
recodeVar(x, src=list(c(1,2)), tgt=list("A"), default="L", keep.na=FALSE)
#[1] "L" "A" "A" "L" "A" "A" "L" "A" "A" "L" "L"

x <- c("no", "yes", "not registered", "no", "yes", "no answer")
recodeVar(x, src = c("no", "yes"), tgt = c("0", "1"), default = NA)
```

`renameCol` *Rename columns in a matrix or a dataframe.*

Description

Rename columns in a matrix or a dataframe.

Usage

```
renameCol(indata, src, tgt)
```

Arguments

`indata`	A dataframe or a matrix
`src`	Source: Vector of names of columns in 'indata' to be renamed. Can also be a vector of column numbers.
`tgt`	Target: Vector with corresponding new names in the output.

Value

A dataframe if 'indata' is a dataframe; a matrix in 'indata' is a matrix.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

`rename.vars`

Examples

```
renameCol(CO2, 1:2, c("kk","ll"))
renameCol(CO2, c("Plant","Type"), c("kk","ll"))

# These fail - as they should:
# renameCol(CO2, c("Plant","Type","conc"), c("kk","ll"))
# renameCol(CO2, c("Plant","Type","Plant"), c("kk","ll"))
```

---

sampleBy *Sampling from a data frame*

Description

A data frame is split according to some variables in a formula, and a sample of a certain fraction of each is drawn.

Usage

```
sampleBy(formula, frac = 0.1, replace=FALSE, data = parent.frame(),systematic=FALSE)
```

Arguments

formula	A formula defining the grouping of the data frame
frac	The part of data to be sampled.
replace	Is the sampling with replacement
data	A data frame
systematic	Should sampling be systematic.

Details

If systematic=FALSE (default) then frac gives the fraction of data sampled. If systematic=TRUE and frac=.2 then every 1/.2 i.e. every 5th observation is taken out.

Value

A data frame

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

orderBy, summaryBy, transformBy, splitBy,

Examples

```
data(dietox)
sampleBy(formula = ~Evit+Cu, frac=.1, data = dietox)
```

`scaleBy` *Groupwise scaling and centering of numeric columns in a dataframe*

Description

Groupwise scaling and centering of numeric columns in a dataframe. Obtained by first splitting a dataframe and then calling scale on each stratum.

Usage

```
scaleBy(formula, data, center = TRUE, scale = TRUE, details=0)
```

Arguments

`formula`	Either a two-sided formula or a list. A dot (.) is allowed on both left and right hand side of formula. See 'details' for the meaning of this.
`data`	A dataframe
`center`	If TRUE then data is centered to have mean zero
`scale`	If TRUE then data is scaled to have variance one
`details`	If larger than zero then information about grouping etc. is printed.

Details

A typical formula is `y1+y2~f1+f2` where y1 and y2 are numeric variables and f1 and f2 can be of any type. For each cross-combination of the values of f1 and f2, the variables (y1,y2) are centered/scaled.

It is valid to write `.~f1+f2`. In this case the variables to be centered/scaled are taken to be all numeric variables in the dataframe except those that a listed on the right hand side of the formula.

It is valid to write `y1+y2~.`. In this case the stratification is taken to be by all non-numeric variables. If there are no non-numeric variables, then no stratification is made and a 'global' centering/scaling is made.

It is valid to write `.~.`. In this case the variables to be centered/scaled are taken to be all numeric variables in the dataframe. The stratification is made by all non-numeric variables. If there are no non-numeric variables, then no stratification is made and a 'global' centering/scaling is made.

Value

A dataframe with the same columns as the input dataframe, but the scaled / centered values are put into the relevant columns.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

applyBy, orderBy,

splitBy, summaryBy, transformBy,

Examples

```
## The following forms are equivalent:
scaleBy(conc+rate ~ state, data=Puromycin)
scaleBy(list(c("conc","rate"), "state"), data=Puromycin)
scaleBy(list(c("."), "."), data=Puromycin)
scaleBy(.~., data=Puromycin)

## The same results can be obtained from
lapply(splitBy(~state, data=Puromycin),
function(.dd) scale(.dd[,sapply(Puromycin,class)=="numeric"]))


## The pig growth data 'dietox'
data(dietox)

# "Remove the effect of time" by centering data within each time point.
dietox2 <- scaleBy(Weight~Time, data=dietox, scale=FALSE)

## Not run:
library(lattice)
xyplot(Weight~Time|Evit+Cu, groups=Pig, data=dietox)
xyplot(Weight~Time|Evit+Cu, groups=Pig, data=dietox2)

## End(Not run)
```

---

splitBy *Split a data frame*

Description

Split a dataframe according to the levels of variables in the dataframe. The variables to split by can be given as a formula or as a character vector.

Usage

```
splitBy(formula, data = parent.frame(), drop=TRUE)
```

Arguments

formula	The right hand side of a formula (or a character vector)
data	A data frame
drop	Logical indicating if levels that do not occur should be dropped
...	Additional arguments, currently not used.

Value

A list of dataframes.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

orderBy, summaryBy, transformBy,

lapplyBy, scaleBy

Examples

```
data(dietox, package="doBy")
splitBy(formula = ~Evit+Cu, data = dietox)
splitBy(formula = c("Evit","Cu"), data = dietox)

splitBy(~Month, data=airquality)
splitBy("Month", data=airquality)
```

---

subSeq *Find sub-sequences of identical elements in a vector.*

Description

Find sub-sequences of identical elements in a vector.

Usage

```
subSeq(x, item = NULL)
```

Arguments

x	An atomic vector.
item	Optionally a specific value to look for in 'x'.

Value

A dataframe.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

rle

Examples

```
x <- c(1,1,1,0,0,1,1,1,2,2,2,1,2,2,2,3)
(ans <- subSeq(x))
ans$value
# Notice: Same results below
subSeq(x,item=1)
subSeq(x,item="1")

x <- as.character(c(1,1,1,0,0,1,1,1,2,2,2,1,2,2,2,3))
(ans<-subSeq(x))
ans$value
# Notice: Same results below
subSeq(x,item="1")
subSeq(x,item=1)
```

---

subsetBy *Finds subsets of a dataframe which is split by variables in a formula.*

Description

A data frame is split by a formula into groups. Then subsets are found within each group, and the result is collected into a data frame.

Usage

```
subsetBy(formula, subset, data = parent.frame(), select, drop=FALSE,
join=TRUE, ... )
```

Arguments

formula	A formula to split by
subset	logical expression indicating elements or rows to keep: missing values are taken as false.
data	A data frame
select	expression, indicating columns to select from a data frame.
drop	passed on to [indexing operator.
join	If FALSE the result is a list of data frames (as defined by 'formula'); if TRUE one data frame is returned.
...	further arguments to be passed to or from other methods.

Value

A data frame.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

See Also splitBy

Examples

```
data(dietox)
subsetBy(~Evit, Weight < mean(Weight), data=dietox)
```

---

summaryBy *Function to calculate groupwise summary statistics*

Description

Function to calculate groupwise summary statistics, much like the summary procedure of SAS

Usage

```
summaryBy(formula, data = parent.frame(), id = NULL, FUN = mean,
          keep.names=FALSE, p2d=FALSE, order=TRUE, full.dimension=FALSE,
          var.names=NULL, fun.names=NULL, ...)
```

Arguments

formula	A formula object, see examples below
data	A data frame
id	A formula specifying variables which data are not grouped by but which should appear in the output. See examples below.
FUN	A list of functions to be applied, see examples below.
keep.names	If TRUE and if there is only ONE function in FUN, then the variables in the output will have the same name as the variables in the input, see 'examples'.
p2d	Should parentheses in output variable names be replaced by dots?
order	Should the resulting dataframe be ordered according to the variables on the right hand side of the formula? (using orderBy
full.dimension	If TRUE then rows of summary statistics are repeated such that the result will have the same number of rows as the input dataset.
var.names	Option for user to specify the names of the variables on the left hand side.
fun.names	Option for user to specify function names to apply to the variables on the left hand side.
...	Additional arguments to FUN. This could for example be NA actions.

Details

Extra arguments ('...') are passed onto the functions in FUN. Hence care must be taken that all functions in FUN accept these arguments - OR one can explicitly write a functions which get around this. This can particularly be an issue in connection with handling NAs. See examples below.

Some code for this function has been suggested by Jim Robison-Cox.

Value

A data frame

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

ave, descStat, lapplyBy, orderBy, scaleBy, splitBy, transformBy,

Examples

```
data(dietox)
dietox12    <- subset(dietox,Time==12)

summaryBy(Weight+Feed~Evit+Cu, data=dietox12,
          FUN=c(mean,var,length))

summaryBy(list(c("Weight","Feed"), c("Evit","Cu")), data=dietox12,
          FUN=c(mean,var,length))


summaryBy(Weight+Feed~Evit+Cu+Time, data=subset(dietox,Time>1),
   FUN=c(mean,var,length))

## Calculations on transformed data:

summaryBy(log(Weight)+Feed~Evit+Cu, data=dietox12)

## Calculations on all numerical variables (not mentioned elsewhere):

summaryBy(.~Evit+Cu,               data=dietox12,
   id=~Litter, FUN=mean)

## There are missing values in the 'airquality' data, so we remove these
## before calculating mean and variance with 'na.rm=TRUE'. However the
## length function does not accept any such argument. Hence we get
## around this by defining our own summary function in which length is
## not supplied with this argument while mean and var are:

sumfun <- function(x, ...){
  c(m=mean(x, ...), v=var(x, ...), l=length(x))
```

```
}
summaryBy(Ozone+Solar.R~Month, data=airquality, FUN=sumfun, na.rm=TRUE)

## Using '.' on the right hand side of a formula means to stratify by
## all variables not used elsewhere:

data(warpbreaks)
summaryBy(breaks ~ wool+tension, warpbreaks)
summaryBy(breaks ~., warpbreaks)
summaryBy(.~ wool+tension, warpbreaks)

## Keep the names of the variables (works only if FUN only returns one
## value):

summaryBy(Ozone+Wind~Month, data=airquality,FUN=c(mean),na.rm=TRUE,
  keep.names=TRUE)

## Using full.dimension=TRUE

## Consider:
summaryBy(breaks~wool, data=warpbreaks)
## Rows of result are replicated below
summaryBy(breaks~wool, data=warpbreaks, full.dimension=TRUE)
## Notice: Previous result is effectively the same as
with(warpbreaks, ave(breaks, wool))
## A possible application of full.dimension=TRUE is if we want to
## standardize (center and scale) data within groups:
ss <- summaryBy(breaks~wool, data=warpbreaks, full.dimension=TRUE, FUN=c(mean,sd))
(warpbreaks$breaks-ss$breaks.mean)/ss$breaks.sd
```

`timeSinceEvent` *Calculate "time since event" in a vector.*

Description

Events are coded as 1 in numeric vector (and non-events are coded with values different from 1). `timeSinceEvent` will give the time since event (with and without sign). In a logical vector, events are coded as TRUE and all non-events as FALSE.

Usage

```
timeSinceEvent(yvar, tvar = seq_along(yvar))
```

Arguments

`yvar`	A numerical or logical vector specifying the events
`tvar`	An optional vector specifying time

Value

A dataframe with columns 'yvar', 'tvar', 'abs.tse' (absolute time since nearest event), 'sign.tse' (signed time since nearest event) and 'run' (indicator of the time window around each event).

Note

NA's in yvar are converted to zeros.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

subSeq, rle

Examples

```
## Events:
yvar <- c(0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0)

## Plot results:
tse<- timeSinceEvent(yvar)
plot(sign.tse~tvar, data=tse, type="b")
grid()
rug(tse$tvar[tse$yvar==1], col=4,lwd=4)
points(scale(tse$run), col=tse$run,lwd=2)
lines(abs.tse+.2~tvar, data=tse, type="b",col=3)

## Find times for which time since an event is at most 1:
tse$tvar[tse$abs<=1]

yvar <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
)
tvar <- c(207, 208, 208, 208, 209, 209, 209, 209, 210, 210, 211, 211,
211, 212, 213, 213, 214, 214, 215, 216, 216, 216, 216, 217, 217,
217, 218, 218, 219, 219, 219, 219, 220, 220, 221, 221, 221, 221,
222, 222, 222)

timeSinceEvent(yvar, tvar)
```

transformBy *Function to make groupwise transformations*

Description

Function to make groupwise transformations of data by applying the transform function to subsets of data.

Usage

```
transformBy(formula, data, ...)
```

Arguments

formula	A formula with only a right hand side, see examples below
data	A data frame
...	Further arguments of the form tag=value

Details

The ... arguments are tagged vector expressions, which are evaluated in the data frame data. The tags are matched against names(data), and for those that match, the value replace the corresponding variable in data, and the others are appended to data.

Value

The modified value of the dataframe data.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

orderBy, summaryBy, splitBy, doby.xtabs,

Examples

```
data(dietox)
transformBy(~Pig, data=dietox, minW=min(Weight), maxW=max(Weight),
    gain=sum(range(Weight)*c(-1,1)))
```

which.maxn *Where are the n largest or n smallest elements in a numeric vector ?*

Description

Determines the locations, i.e., indices of the n largest or n smallest elements of a numeric vector.

Usage

```
which.maxn(x, n = 1)
which.minn(x, n = 1)
```

Arguments

x	numeric vector
n	integer >= 1

Value

A vector of length at most n with the indices of the n largest / smaller elements. NAs are discared and that can cause the vector to be smaller than n.

Author(s)

Søren Højsgaard, <sorenh@math.aau.dk>

See Also

which.max, which.min

Examples

```
x <- c(1:4,0:5,11,NA,NA)
ii <- which.minn(x,5)

x <- c(1,rep(NA,10),2)
ii <- which.minn(x,5)
```

Index