Package ‘kyotil’

May 14, 2021

LazyLoad yes
LazyData yes
Version 2021.5-14
Title Utility Functions for Statistical Analysis Report Generation and Monte Carlo Studies
Depends R (>= 3.6)
Imports methods
Suggests RUnit, R.rsp, lme4, nlme, xtable, MASS, splines, survival,
        abind, pracma, VGAM, copula, mvtnorm, Hmisc, RColorBrewer, zoo,
        doParallel, Exact, survey
Description Helper functions for creating formatted summary of regression models, writing publication-ready tables to latex files, and running Monte Carlo experiments.
VignetteBuilder R.rsp
License GPL (>= 2)
NeedsCompilation yes
Author Youyi Fong [cre],
        Krisztian Sebestyen [aut],
        Han Sunwoo [aut],
        Jason Becker [ctb],
        Bendix Carstensen [ctb],
        Daryl Morris [ctb],
        Josh Pasek [ctb],
        Dennis Chao [ctb],
        Andri Signorell [ctb],
        Sue Li [ctb],
        Jonathan Bartlett [ctb],
        Christophe Dutang [ctb]
Maintainer Youyi Fong <youyifong@gmail.com>
Repository CRAN
Date/Publication 2021-05-14 19:20:03 UTC
### Description

Calculate age, by Jason P Becker, modified very slightly in how arguments are passed to the function.

### Usage

```r
age_calc(dob, enddate = Sys.Date(), units = c("days","months","years"), precise = TRUE)
```

### Arguments

- **dob**: POSIXlt or Date. Birthday
- **enddate**: POSIXlt or Date. Date to compute age
- **units**: string. Choose a unit.
- **precise**: Boolean.
**base.functions**

**Author(s)**

Jason P Becker

**References**


**Examples**

```r
age_calc (dob=strptime("29OCT2002", format="%d%b%Y"),
         enddate=strptime("30OCT2003", format="%d%b%Y"), units='years', precise=TRUE)
age_calc (dob=strptime("29OCT2002", format="%d%b%Y"),
         enddate=strptime("30DEC2003", format="%d%b%Y"), units='years', precise=FALSE)
```

---

**Some Base Functions**

**Description**

cbinduneven binds together a list of matrixes/dataframes of different lengths, rows are matched by names binary returns binary representation of an integer. binary2 returns binary representation of an integer with leading 0, the length of string is n. mysystem can call any exe file that is in the PATH f2c convert temperature from f to c/

**Usage**

```r
cbinduneven(li)
binary(i)
multi.outer (f, ... )
myreshapelong(dat, cols.to.be.stacked, label.cols.to.be.stacked, new.col.name)
binary2(i, n)
f2c(f)
ftoi(f)
keepWarnings(expr)
meanmed(x, na.rm = FALSE)
methos4(classes, super = FALSE, ANY = FALSE)
```
base.functions

myaggregate(x, by, FUN, new.col.name = "aggregate.value", ...)

myreshapeWide(formula, dat, idvar, keep.extra.col=FALSE)

mysapply(X, FUN, ..., simplify = TRUE, USE.NAMES = TRUE, ret.mat = TRUE)

myscale(x)

mysystem(cmd, ...)

mytapply(X, INDEX, FUN = NULL, ..., simplify = TRUE)

read.sv(file, header = TRUE, ...)

read.tsv(file, header = TRUE, sep = "\t", ...)

table.prop(x,y=NULL,digit=1,style=2,whole.table.add.to.1=FALSE,useNA="ifany", add.perc=FALSE, add.total.column = FALSE)

table.cases (case,group,include.all=TRUE,desc="cases")
table.cases.3(case,group1,group2)

unix()

mycor (x, use = "everything", method = c("pearson", "kendall", "spearman"),
   alternative = c("two.sided", "less", "greater"), exact = NULL,
   conf.level = 0.95, continuity = FALSE,
   digits.coef=2, digits.pval=3,
   ...)

Arguments

add.total.column

use

method

alternative

exact

conf.level

continuity

digits.coef

digits.pval

cols.to.be.stacked

label.cols.to.be.stacked
In multi.out, f is a function.

vector of 0/1

vector of multi-group indicators

a formula object.

a formula object.


Boolean
Examples

```r
binary(5) ### 101
binary2(5, 4)

a=data.frame("x"=1:2)
b=data.frame("y"=3:5);#rownames(b)[3]="
cbinduneven(list(a,b))

## Not run:
# the formula in myreshapewide can only have one variable in the right hand side
myreshapewide(fi~week, dat, c("ptid","stim"))

myreshapelong(dat.201.neut, cols.to.be.stacked=c("MN.3","SF162","SVA.MLV"),
  label.cols.to.be.stacked="antigen", new.col.name="y")

myaggregate(subset(dat.poc, select=c(HIV, trt)), list(dat.poc$f), function(x)
  with(x, c(fisher.test(HIV, trt)$estimate, fisher.test(HIV, trt)$p.value)))

## End(Not run)
```

---

**binaryloess**

*Using loess to Check Functional Form for Logistic Regression*

**Description**

This function plots a smoothed line of how the average value of Y changes with X in order to check functional form for logistic regression.

**Usage**

`binaryloess(x, y, scale = c("logit", "linear"), span = 0.7, weights = NULL, ...)`

**Arguments**

- `x`  
- `y`  
- `scale`  
- `span` smoothing parameter, passed to loess. If less than 1, the neighbourhood includes proportion \( a \) of the points. If greater than 1, all points are used, with the maximum distance assumed to be \( a^{1/p} \) times the actual maximum distance for \( p \) explanatory variables. Missing records are removed first.
- `weights` sampling weights, passed to loess
- `...` passed to plotting function
Details


Examples

```r
set.seed(1234)
n <- 1000
x <- rnorm(n)
xb <- -2+x
pr <- exp(xb)/(1+exp(xb))
y=rbern(n, pr)

par(mfrow=c(1,2))
binaryloess(x, y, scale = "logit", span = 0.7, weights = NULL, ylab="logit(p)")
binaryloess(x, y, scale = "linear", span = 0.7, weights = NULL, ylab="prob")
```

description

A slightly modified test of the proportional hazards assumption for a Cox regression model fit (coxph). This version corrects some conservativeness of the test.

Usage

```r
cox.zph.2(fit, transform = "km", global = TRUE, exact=TRUE)
```

Arguments

- `fit` 
- `transform` 
- `global` 
- `exact` Boolean. If FALSE, this function is an identical copy of cox.zph. If TRUE, it computes the variance of the test statistic exactly, instead of approximately.

Details

When the model uses time-dependent covariates, the approximation used in Grambsch and Therneau resulted in conservativeness of the test. This is "fixed" here at a cost of up to 2.5 times longer execution time.
References

Fong, Y. and Halloran, M Elizabeth and Gilbert, P. Using Time-Dependent Age Group in Cox Regression Analysis of Vaccine Efficacy Trials, Just Another Epi Journal, in prep.

See Also

cox.zph

Examples

library(survival)
fit <- coxph(Surv(futime, fustat) ~ age + ecog.ps,
data=ovarian)
temp <- cox.zph(fit)
print(temp)
temp.2 <- cox.zph.2(fit)
print(temp.2)

---

Deming

*Fit Deming regression.*

Description

Deming regression fit. Assume x and y variances are the same. Slightly modified from MethComp R package.

Usage

Deming(x, y, vr = sdr^2, sdr = sqrt(vr), boot = TRUE, keep.boot = FALSE, alpha = 0.05)

Arguments

x

y

vr

sdr

boot

keep.boot

alpha
Examples

```r
## Not run:
set.seed(1)
x=rnorm(100,0,1)
y=x+rnorm(100,0,.5)
x=x+rnorm(100,0,.5)
fit=Deming(x,y, boot=TRUE)
summary(fit)
plot(x,y)
abline(fit)
# compare with lm fit
fit.1=lm(y~x, data.frame(x,y))
summary(fit.1)
abline(fit.1, col=2)
## End(Not run)
```

---

**DMHeatMap**

Better Heatmap Function

**Description**

Makes a heatmap representation of correlation coefficients easier.

**Usage**

```r
DMHeatMap(x, Rowv = TRUE, Colv = if (symm) "Rowv" else TRUE,
    distfun = dist, hclustfun = hclust, dendrogram =
    c("both", "row", "column", "none"), symm = FALSE,
    scale = c("none", "row", "column"), na.rm = TRUE, revC
    = identical(Colv, "Rowv"), add.expr, breaks, symbreaks
    = min(x < 0, na.rm = TRUE) || scale != "none", col =
    "heat.colors", colsep, rowsep, sepcolor = "white",
    sepwidth = c(0.05, 0.05), cellnote, notecex = 1,
    notecol = "cyan", na.color = par("bg"), trace =
    c("column", "row", "both", "none"), tracecol = "cyan",
    hline = median(breaks), vline = median(breaks),
    linecol = tracecol, margins = c(5, 5), ColSideColors,
    RowSideColors, cexRow = 0.2 + 1/log10(nr), cexCol =
    0.2 + 1/log10(nc), labRow = NULL, labCol = NULL,
    labColor = NULL, axis = TRUE, heatmapOnly = FALSE, key
    = TRUE, keysize = 1.5, density.info = c("histogram",
    "density", "none"), denscol = tracecol, symkey = min(x
    < 0, na.rm = TRUE) || symbreaks, densadj = 0.25, main
    = NULL, xlab = NULL, ylab = NULL, lmat = NULL, lhei
    = NULL, lwid = NULL, lower.left.only = TRUE, legend =
    TRUE, legend.x = "topright", verbose = FALSE, ...)
```
Arguments

x
axis
heatmapOnly
verbose
legend.x
legend
Rowv
Colv
distfun
hclustfun
dendrogram
symm
scale
na.rm
revC
add.expr
breaks
symbreaks
col
colsep
rowsep
sepcolor
sepwidth
cellnote
notecex
notecol
na.color
trace
tracecol
hline
vline
linecol
margins
ColSideColors
RowSideColors
cexRow
get.sim.res

Examples

cor=matrix(runif(15),5,3)
breaks=c(-1,-.7,-.5,-.3,-.1,.1,.3,.5,.7,1)

hU=DMHeatMap(cor, trace="none", symm=FALSE, dendrogram="none", col=RColorBrewer::brewer.pal(
  length(breaks)-1,"RdYlGn"), distfun = function(c) as.dist(1 - c), cexRow =1.5, cexCol =1.5,
  lmat=rbind( c(2, 1), c(4,3) ), lhei=c(4, 1 ), breaks=breaks, margins=c(2,2), key = FALSE,
  Rowv=NA, lower.left.only=FALSE)

Description

Go through a folder and read all files and combine the results into a multidimensional array.
Usage

get.sim.res (dir, res.name="res", verbose=TRUE)
MCSummary (dir, res.name = "res", exclude.some = TRUE,
               exclude.col = 1, verbose = TRUE)
getFormattedMCSummary (path, sim, nn, fit.method, exclude.some = TRUE,
                        exclude.col = 1, verbose = TRUE, coef.0 = NULL, digit1
                        = 2, sum.est = c("mean", "median"), sum.sd =
                        c("median", "mean"), style = 1, keep.intercept =
                        FALSE)

Arguments

dir directory of MC result files
path partial path to the directory of MC result files
res.name name of the R object saved in the files, default is res, but may be others
verbose Boolean
sim a string to denote simulation setting
nn a vector of sample sizes
fit.method a string to denote fitting method. sim, nn and fit.method together forms the name
            of the directory containing MC result files
exclude.col column number
exclude.some whether to exclude MC results that are extreme
coef.0 simulation truth
digit1 digits
sum.est use mean or median as location estimate summary
sum.sd use mean or median as sd estimate summary
style integer
keep.intercept whether to include intercept in the table

Details

Depends on package abind to combine arrays from files.

Value

A multidimensional array.
getK

Description

getK calculates the kernel matrix between X and itself and returns a n by n matrix. Alternatively, it calculates the kernel matrix between X and X2 and returns a n by n2 matrix.

Usage

getK (X, kernel, para=NULL, X2=NULL, C = NULL)

Arguments

X covariate matrix with dimension n by d. Note this is not the paired difference of covariate matrix.

kernel string specifying type of kernel: polynomial or p (1 + <x,y>)^para, rbf or r exp(-para*||x-y||^2), linear or l <x,y>, ibs or i 0.5*mean(2.0 - |x-y|) or sum(w*(2.0 - |x-y|))/sum(w), with x[i],y[i] in 0,1,2 and weights 'w' given in 'para'. hamming or h for sum(x == y) with x[i],y[i] binary, no default.

para parameter of the kernel function. for ibs or hamming, para can be a vector of weights.

X2 optional second covariate matrix with dimension n2 by d

C logical. If TRUE, kernels are computed by custom routines in C, which may be more memory efficient, and faster too for ibs and hamming kernels.

Details

IBS stands for ‘Identical By State’. If ’x’, ’y’ are in in 0,1,2 then
IBS(x,y) = 0 if |x-y|=2, 1 if |x-y|=1, 2 if |x-y|=0, or IBS(x,y) = 2.0 - |x-y|.
K(u,v) = sum(IBS(u[i],v[i])) / 2K where K = length(u).
The ‘hamming’ kernel is the equivalent of the ‘ibs’ kernel for binary data. Note that ‘hamming’ kernel is based on hamming similarity(!), not on dissimilarity distance.
Within in the code, C is default to TRUE for ibs and hamming kernels and FALSE otherwise.

Value

A kernel matrix.

Author(s)

Youyi Fong <youyifong@gmail.com>
Krisztian Sebestyen <ksebestyen@gmail.com>
Shuxin Yin <>
Examples

X = cbind(x1=rnorm(n=5), x2=rnorm(n=5))
dim(X)
X2 = cbind(x1=rnorm(n=3), x2=rnorm(n=3))
dim(X2)

K = getK(X,"linear")
dim(K)
K = getK(X,"linear",X2=X2)
dim(K)
K1 = getK(X2,"1",X2=X)
dim(K1)
all(K==t(K1))

# RBF kernel
K = getK(X,"rbf",para=1,X2=X2)
K1 = getK(X2,"r",para=1,X2=X)
all(K==t(K1))

# IBS kernel for ternary data
X <- as.matrix(expand.grid(0:2,0:2))
K = getK(X,kernel = 'ibs')

# add weight
w = runif(ncol(X))
K = getK(X,kernel = 'ibs',para = w)

# IBS kernel for binary data via option 'h' for 'hamming similarity measure'
X <- as.matrix(expand.grid(0:1,0:1))
K=getK(X,kernel = 'h')

iorw

Causal Mediation Analysis of Cowling et al.

Description

Estimate the total, direct, and indirect effects using IORW method (inverse odds ratio weighting) and compute 95

Usage

iorw(formula.effect, formula.mediators, data, family = NULL, nboot = 10000, numCores = 1, save.steps = FALSE,
iorw

verbose = FALSE)

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

Arguments

formula.effect a formula object for the total and direct effect regression. The first term on the right is assumed to be the binary treatment/exposure variable.

formula.mediators a formula object for logistic regression. It should be of the form: ~ mediation marker1 + mediation marker2.

data a data frame.

family if Cox regression, leave as NULL; otherwise, it will be passed to glm().

nboot an integer. Number of bootstrap replicates.

numCores an interger. Number of cores to use for parallel procesing.

save.steps boolean. Whether or not to save the fits from the three steps and the weights.

x Object of type iorw

verbose boolean.

... Additional arguments passed to the print function.

Details

Code by Cowling and Lim was downloaded from https://datadryad.org/stash/dataset/doi:10.5061/dryad.cv37539

If a bootstrap replicate generates warnings during regression, NA will be returned for that replicate. The number of such occurrences is recorded in an attribute of boot.perc in the return value.

It does not handle sampling weights yet.

Value

Point estimates and percentile bootstrap confidence intervals.

Author(s)

Youyi Fong, based on code by Cowling and Lim

References


Examples

###### Cox regression

# without adjusting for baseline markers
library(survival)
formula.effect = Surv(surv_time, flu) ~ vaccine + age
formula.mediators = ~log2(postvax.B.Brisbane/5)
res.1 = iorw(formula.effect, formula.mediators, kid, nboot=10, numCores=1); res.1
stopifnot(max(abs(res.1$boot[,1] - c(0.2029779, 0.6070105, 0.3039110, 0.4283389, 0.2124268))) < 1e-6)

# adjust for baseline markers
formula.effect = Surv(surv_time, flu) ~ vaccine + log2(prevax.B.Brisbane) + age
formula.mediators = ~log2(postvax.B.Brisbane/5)
res.2 = iorw(formula.effect, formula.mediators, kid, nboot=10, numCores=1); res.2

###### Logistic regression

# without adjusting for baseline markers
formula.effect = flu ~ vaccine + age
formula.mediators = ~log2(postvax.B.Brisbane/5)
res.3 = iorw(formula.effect, formula.mediators, kid, family=binomial(), nboot=10, numCores=1); res.3
stopifnot(max(abs(res.3$boot[,1] - c(0.1960024, 0.6154349, 0.2937164, 0.4145470, 0.2168644))) < 1e-6)

# adjust for baseline markers
formula.effect = flu ~ vaccine + log2(prevax.B.Brisbane) + age
formula.mediators = ~log2(postvax.B.Brisbane/5)
res.4 = iorw(formula.effect, formula.mediators, kid, family=binomial(), nboot=10, numCores=1); res.4

---

**kid**  
*Dataset from Cowling et al.*

**Description**

Influenza immune response biomarkers dataset.

**Usage**

data("kid")
**Format**

A data frame with 736 observations on the following 10 variables.

- hhID  a numeric vector
- age   a numeric vector
- intervention    a character vector
- vaccine  a numeric vector
- vaccine.date a Date
- postvax.date a Date
- prevax.B.Brisbane a numeric vector
- postvax.B.Brisbane a numeric vector
- surv_time a numeric vector
- flu    a numeric vector

**References**


**Description**

Utility functions by Youyi Fong and Krisz Sebestyen, and some functions copied from other packages for convenience (acknowledged on their manual pages).

Most useful functions: mypostscript/mypdf, mytex,

See the Index link below for a list of available functions.

The package depends on Hmisc. The main reason for that, besides the usefulness of the package, is Hmisc depends on ggplot2, which also define
Description
Returns a data frame that is suitable for time-dependent covariate Cox model fit.

Usage
make.timedep.dataset(dat, X, d, baseline.ageyrs, t.1, t.2 = NULL)

Arguments
- dat: data frame
- X: string. Name of the followup time column in dat. Unit needs to be years.
- d: string. Name of the followup time column in dat.
- baseline.ageyrs: string. Name of the followup time column in dat.
- t.1: numerical. Cutoff for age group
- t.2: numerical. Second cutoff for age group

Details
The function assumes that the followup length is such that only one change of age group is possible.

Value
Returns a data frame with the following columns added: tstart, tstop, .timedep.agegrp, .baseline.agegrp
- tstart: left bound of time interval
- tstop: right bound of time interval
- .timedep.agegrp: time-dependent age group
- .baseline.agegrp: baseline age group

Author(s)
Youyi Fong

References
Therneau, T. and Crowson, C. Using Time Dependent Covariates and Time Dependent Coefficients in the Cox Model. A vignette from the R package survival.
Examples

```
library(survival)

n=3000; followup.length=5; incidence.density=0.015; age.sim="continuous"

dat.0=sim.dat.tvarying.two(n, followup.length, incidence.density, age.sim, seed=1)
dat=subset(dat.0, for.non.tvarying.ana, select=c(ptid, X, d, baseline.age, trt))
dat.timedep = make.timedep.dataset (dat, "X", "d", "baseline.age", 6)
coxph(Surv(tstart,tstop,d) ~ trt*.timedep.agegrp, dat.timedep)
```

---

**math.functions**

### Description

H calculates entropy.

### Usage

```
as.binary(n, base = 2, r = FALSE)
binom.coef(n, m)
expit(x)
logDiffExp(logx1, logx2)
logit(x)
logMeanExp(logx, B = NULL)
logSumExp(logx)
logSumExpFor2(logx, logy)
permn(x, fun = NULL, ...)
Stirling2(n, m)
interpolate(pt1, pt2, x)
```
matrix.array.functions

Matrix and Array Functions

Description
concatList returns a string that concatenates the elements of the input list or array

Usage
AR1(p, w)
concatList(lis, sep = "")
EXCH(p, rho)
fill.jagged.array(a)
getMidPoints(x)
getUpperRight(matri, func = NULL)
matrix.array.functions

last(x, n = 1, ...)
mix(a, b)

## S3 method for class 'data.frame'
rep(x, times = 1, ...)

## S3 method for class 'matrix'
rep(x, times = 1, each = 1, by.row = TRUE, ...)

## S3 method for class 'matrix.block'
rep(x, times = 2, ...)

shift.left(x, k = 1)
shift.right(x, k = 1)

thin.rows(dat, thin.factor = 10)
ThinRows(dat, thin.factor = 10)

tr(m)

Arguments

- p
- w
- lis (list or array)
- sep
- rho
- a
- x
- matri
- func
- n
- ...
- b
- times
- each
- by.row
- k
- dat
- thin.factor
- m
Examples

```
concatList(1:3, "_")
```

Description

DXD computes $D \times X \times D$, where $D$ is a diagonal matrix. tXDX computes $t(X) \times D \times X$. symprod computes $S \times X$ for symmetric $S$. txSy computes $t(x) \times S \times y$ for symmetric $S$.

Usage

```
DXD(d1, X, d2)
tXDX(X, D)
symprod(S, X)
txSy(x, S, y)
.as.double(x, stripAttributes = FALSE)
```

Arguments

- `d1`: a diagonal matrix or an array
- `d2`: a diagonal matrix or an array
- `x`: array
- `y`: array
- `S`: symmetric matrix
- `X`: matrix
- `D`: matrix
- `stripAttributes`: boolean

Details

.as.double does not copying whereas as.double(x) for older versions of R when using .C(DUP = FALSE) make duplicate copy of x. In addition, even if x is a 'double', since x has attributes (dim(x)) as.double(x) duplicates.

The functions do not check whether S is symmetric. If it is not symmetric, then the result will be wrong. DXD offers a big gain, while symprod and txSy gains are more incremental.
Author(s)

Krisztian Sebestyen

Examples

\begin{verbatim}
d1=1:3
d2=4:6
X=matrix(1:9,3,3)
all(DXD(d1, X, d2) == diag(d1) %*% X %*% diag(d2))

S=matrix(c(1,2,3,2,4,5,3,5,8),3,3)
X=matrix(1:9,3,3)
all( symprod(S, X) == S %*% X )

x=1:3
y=4:6
S=matrix(c(1,2,3,2,4,5,3,5,8),3,3)
txSy(x, S, y) == drop(t(x)%*%S%*%y)
\end{verbatim}

misc

Misc Functions

Description

Misc functions. summ computes iterative sum, sort of like diff.

Usage

\begin{verbatim}
pava (x, wt = rep(1, length(x)))
summ(x)
sample.for.cv (dat, v, seed)
empty2na(x)
## S3 method for class 'pcc'
predict(object, newdat, ...)
rank.inv.norm(x)
INT(x)
get.kfold.splits (dat, k, seed)
\end{verbatim}

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dat</td>
<td>a data frame. One of the columns must be named y and y should be 0/1 with 1 for case and 0 for control</td>
</tr>
</tbody>
</table>
p.adj.perm

v

v-fold cross validation

seed

seed for random number generators

x

wt

k

object

e newdat

...

Details

case and controls are sampled separately. rank.inv.norm: rank-based inverse normal/gaussian transformation

Value

sample.for.cv returns a list of two vector of integers: train and test, which refer to the rows of dat

p.adj.perm

Permutation-based Multitesting P Values Adjustment

Description

An implementation of Westfall and Young

Usage

p.adj.perm(p.unadj, p.perms, alpha = 0.05)

Arguments

p.unadj

p.perms

alpha

Details

This implementation is not as fast as the implementation from the package multtest. But usually the step to create p.perms is the rate-limiting step.

The smallest of the Westfall and Young FWER-controlling multitesting adjusted p values coincides with the p value for testing a global null without any assumptions. But for the multitesting adjustment to hold, it requires the subset pivotality condition.
Author(s)
Sue Li, sli@fredhutch.org

References

plotting

Plotting Functions

Description

mypostscript and mypdf sets the width and height based on mfrow input.

Usage

myplot (object, ...)  
## S3 method for class 'loess'
myplot(object, xlab="x", ylab="fitted", ...)  

whiskers (x, s, ...)

abline.pt.slope(pt1, slope, x2=NULL, ...)

abline.pts(pt1, pt2 = NULL)

butterfly.plot(dat, dat2 = NULL, add = FALSE, xaxislabels = rep("", 4), x.ori = 0, xlab = ", " , ylab = "", cex.axis = 1, ...)

empty.plot()

add.mtext.label (text, cex = 1.4, adj = -0.2)

mydev.off(file = "temp", ext = c("pdf"), res = 200, mydev = NULL)

getMfrow(len)

myhist (x, add.norm=TRUE, col.norm="blue", ...)

myforestplot(dat, xlim = NULL, xlab = ", " , main = ", " , col.1 = "red", col.2 = "blue", plot.labels = TRUE, order = FALSE, decreasing = FALSE, vline = TRUE, cols = NULL, log =
my.interaction.plot(dat, x.ori = 0, xaxislabels = rep("", 2), cex.axis = 1, add = FALSE, 
xtab = "", ylab = "", pcol = NULL, lcol = NULL, ...) 

myboxplot(object, ...) 

## S3 method for class 'formula'
myboxplot(formula, data, cex = 0.5, xlab = "", ylab = "", main = 
"", box = TRUE, at = NULL, na.action = NULL, p.val = 
NULL, pch = 1, col = 1, test = "", 
friedman.test.formula = NULL, reshape.formula = NULL, 
reshape.id = NULL, jitter = TRUE, add.interaction = 
FALSE, drop.unused.levels = TRUE, bg.pt = NULL, add = 
FALSE, seed = 1, write.p.at.top = FALSE, ...) 

## S3 method for class 'data.frame'
myboxplot(object, cex = 0.5, ylab = "", xlab = "", main = "", 
box = TRUE, at = NULL, pch = 1, col = 1, test = "", 
paired = FALSE, ...) 

## S3 method for class 'list'
myboxplot(object, paired = FALSE, ...) 

abline.shade.2(x, col=c(0,1,0)) 
abline.shade(pt, type = 5, col = c(0, 1, 0), alpha = 0.3) 

mylegend(legend, x, y=NULL, lty = NULL, bty = "n", ...) 

mymatplot(x, y=NULL, lty = NULL, bty = "n", ...) 

abline.shade.2(x, col=c(0,1,0)) 
abline.shade(pt, type = 5, col = c(0, 1, 0), alpha = 0.3) 

mylegend(legend, x, y=NULL, lty = NULL, bty = "n", ...) 

mymatplot(x, y=NULL, lty = NULL, bty = "n", ...) 

mypairs(dat, ladder = FALSE, show.data.cloud = TRUE, 
ladder.add.line = T, ladder.add.text = T, ...) 

wtd.hist (x, breaks = "Sturges", freq = NULL, probability = !freq, 
include.lowest = TRUE, right = TRUE, density = NULL, angle = 45, 
col = NULL, border = NULL, main = paste("Histogram of", xname), 
xlim = range(breaks), ylim = NULL, xlab = xname, ylab, axes = TRUE, 
plot = TRUE, labels = FALSE, nclass = NULL, weight = NULL,
plotting

...)

mylines(x, y, type = "l", ...)

myfigure(mfrow = c(1, 1), mfcol = NULL, width = NULL,
  height = NULL, oma = NULL, mar = NULL, main.outer = FALSE, bg=NULL)

mypdf(...)

mypng(...)
mytiff(...)

mypostscript(file = "temp", mfrow = c(1, 1), mfcol = NULL, width = NULL,
  height = NULL, ext = c("eps", "pdf", "png", "tiff"), oma = NULL,
  mar = NULL, main.outer = FALSE, save2file = TRUE, res = 200,
  ...)

panel.cor(x, y, digits = 2, prefix = ",", cex.cor, cor., leading0
  = FALSE, cex.cor.dep = TRUE, ...)

panel.hist(x, ...)

panel.nothing(x, ...)

corplot(object, ...)

## Default S3 method:
corplot(object, y, ...)

## S3 method for class 'formula'
corplot(formula, data, main = ",", method = c("pearson", "spearman"),
  col = 1, cex = 0.5, add.diagonal.line = TRUE, add.lm.fit = FALSE, add.loess.fit = FALSE,
  col.lm = 2, add.deming.fit = FALSE, col.deming = 4,
  add = FALSE, log = ",", same.xylim = FALSE, xlim =
  NULL, ylim = NULL, ...)

Arguments

cex.cor.dep
add.loess.fit
leading0
null.val
write.p.at.top
text.width
text
cex
adj
file
ext
res                resolution.
add.norm                Boolean, whether to add normal approximation density line
col.norm                string, color of added normal density line
pt1
s
ladder
slope
friedman.test.formula

reshape.id
impute.missing.for.line

cor.
mydev
jitter                Boolean
add.interaction                Boolean
...
xaxt
breaks
freq
bg.pt
probability
include.lowest
right
density
angle
border
axes
plot
labels
nclass
weight
pt2
pt
alpha
dat
lwd
x.intersp
y.intersp
legend.inset
dat2
add
log
add.lm.fit
add.deming.fit
col.lm
col.deming
reshape.formula
xaxislabels
x ori
xlab
ylab
cex.axis
len
same.xylim
xlim
ylim
main
col.1
col.2
pcol
lcol
object
formula
data
box
at
line width.
controls the look of legend.
controls the look of legend.
legend inset
a formula object.
Boolean. Whether xlim and ylim should be the same
plotting

pch
col
test string. For example, "t","w","f","k","tw"
legend
x
lty
bty
type
make.legend
legend.x
legend.title
legend.cex
draw.x.axis
bg
method
mfrow
mfcol
width
height
oma
mar
main.outer
save2file
y
digits
prefix
cex.cor
plot.labels Boolean
order Boolean
decreasing Boolean
add.diagonal.line

x2
vline
cols
na.action
drop.unused.levels
plotting

p.val
seed
paired
show.data.cloud

ladder.add.line
ladder.add.text

Details
myboxplot shows data points along with boxes. The data points are jittered and the pattern of jittering is made reproducible in repeated calls. The test can only take one type of test currently.

myforestplot is modified from code from Allan deCamp/SCHARP. dat should have three columns. first column should be point estimate, second and third lci and uci, fourth p value. col.1 is the color used for CIs that do not include null, col.2 is used for CIs that do include null. If order is TRUE, the rows are ordered by the first column of dat. decreasing can be used to change the behavior of order.

corplot.formula uses MethComp::Deming by Bendix Carstensen to fit Deming regression.

wtd.hist is copied from weights package, author: Josh Pasek.

mymatplot will use na.approx (zoo) to fill in NA before plotting in order to draw continuous lines. The filled-in values will not be shown as points.

Examples

```r
set.seed(1)
x=1:50+rnorm(50,0,4)
y=1:50+rnorm(50,0,4)
dat=data.frame(x, y)
corplot(y~x,dat,add.lm.fit=TRUE,add.deming.fit=TRUE,col.lm="red",col.deming="blue")

dat=data.frame(y=c(1:10,2:11), x=rep(c("a","b"),each=10), ptid=c(1:10,1:10))
par(mfrow=c(1,2))
myboxplot(y~x, dat, test="w", jitter=FALSE)
myboxplot(y~x, dat, test="f", add.interaction=TRUE, reshape.formula=y~x, reshape.id="ptid")

myboxplot(list(jitter(1:10), jitter(3:12)), test="w")
myboxplot(list(jitter(1:10), jitter(3:12)), test="w", paired=TRUE)

## Not run:
myfigure(mfrow=c(1,2))
    plot(1:10)
    plot(1:10)
mydev.off(ext="png,pdf", file="tmp")
## End(Not run)
```
Description

roundup prints a specified number of digits after decimal point even if 0s are needed at the end.
formatInt prints a specified number of digits before decimal point even if 0s are needed at the beginning.

Usage

myprint(object, ...)

## Default S3 method:
myprint(..., newline = TRUE, digits = 3, print.name=TRUE)

## S3 method for class 'matrix'
myprint(object, ...)

formatInt(x, digits, fill = "0", ...)

make.latex.coef.table(models, model.names = NULL, row.major = FALSE, round.digits = NULL)

mysanitize.text(str)
mysanitize.numbers(x)

mytex(dat = NULL, file.name = "temp", digits = NULL, display = NULL, align = "r", include.rownames = TRUE, include.colnames = TRUE, col.headers = NULL, comment = FALSE, floating = FALSE, lines = TRUE, hline.after = NULL, add.to.row = NULL, sanitize.text.function = NULL, append = FALSE, preamble = "", input.foldername = NULL, save2input.only = NULL, caption = NULL, label = paste("tab", last(strsplit(file.name, "/"))[1]), sep = " ", table.placement = "h!", add.clear.page.between.tables = FALSE, longtable = FALSE, verbose = FALSE, ...)

mytex.begin(file.name, preamble = "")
mytex.end(file.name)

mywrite(x, ...)

mywrite.csv(x, file = "tmp", row.names = FALSE, digits = NULL, ...)

roundup (value, digits, na.to.empty=TRUE, remove.leading0=TRUE)

formatDouble(value, digits, na.to.empty=TRUE, remove.leading0=TRUE)

**Arguments**

- input.foldername
- object
- newline
- print.name
- save2input.only
  - Boolean
- include.colnames
  - Boolean
- col.headers
  - string. Column headers
- comment
  - Boolean, whether to include the version and timestamp comment
- hline.after
  - vector
- add.to.row
  - a list
- sanitize.text.function
  - a function
  - str
- remove.leading0

- caption
- longtable
- label
  - default to be the same as file.name stem
- table.placement

- na.to.empty
- value
- digits
- fill
- models
- model.names
- row.major
Examples

roundup (3.1, 2) # 3.10
formatInt(3, 2) # 03

## Not run:
# demo of dimnames
tab=diag(1:4); rownames(tab)<-colnames(tab)<-1:4; names(dimnames(tab))=c("age","height")
# for greek letter in the labels, we need sanitize.text.function=identity
rownames(tab)[1]="$\alpha$
# note that to use caption, floating needs to be TRUE
mytex (tab, file="tmp!", sanitize.text.function=identity,
  caption="This is a caption .........................", caption.placement="top",
  floating=TRUE)

# col.headers has to have the RIGHT number of columns
# but align is more flexible, may not need to include the rownames col
# but diag(1:4) has 4 columns
mytex (tab, file="tmp", include.rownames = TRUE,
  align=c("c","c","c|","c","c"), col.headers="\hline\n & \multicolumn{2}{c|}{Vaccine} & \multicolumn{2}{c}{Control} \ \ \n")
# not include rownames
mytex (tab, file="tmp", include.rownames = FALSE,
  align=c("c","c","c","c","c"), col.headers=...
The code snippet demonstrates the use of the `mytex` function to create a LaTeX table with specific row and column headers, including an example of adding custom row content.

---

### random.functions

#### Description

Generate samples from random variables.

#### Usage

- `dbern(x, prob, log = FALSE)`
- `dcorbern(x, p, a, log = FALSE)`
- `dmixnorm(x, mix.p, sd1, sd2, log = FALSE)`
- `dnorm.norm.gamma(x, p, same.distr = FALSE, log = FALSE)`
- `rbern(n, prob, generalized = FALSE)`
- `rbigamma(n, shape.1, shape.2, rate.1, rate.2, rho)`
- `rbilogistic(n, loc.1, loc.2, scale.1, scale.2, rho)`
- `rejective.sampling(N, n, pik)`
rnorm.ar(n, sd, rho)

rnorm.norm.gamma(n, mu.0, lambda, alpha, beta)

rmixnorm (n, mix.p, mu1, mu2, sd1, sd2)

rdoublexp(n, location=0, scale=1)
ddoublexp(x, location=0, scale=1)
qdoublexp(p, location=0, scale=1)
pdoublexp(q, location=0, scale=1)

rbidoublexp(n, loc.1, loc.2, scale.1, scale.2, rho)

**Arguments**

q
location
scale
x
prob
log
p
a
mix.p
sd1
sd2
same.distr
n
generalized
N
pik
mu
mu1
mu2
sd
alpha
mu.0
lambda
beta
Loc. 1
Loc. 2
Scale. 1
Scale. 2
Rate. 1
Rate. 2
Shape. 1
Shape. 2
Rho

Details
rbern generates Bernoulli random variables.
rbilogistic generates a bivariate logistic distribution for correlation coefficient 0.5, or [-0.271, 0.478].
In the former case it is generated by calling rbilogis, part of the VGAM package; in the latter case it is generated via the AMH copular.
rnorm.ar simulate autoregressive normal random variables, correlation is rho^d between x_1 and x_(1+d)

Examples

set.seed(1)
rbern(n=10, p=1/2)
rbern(n=2, p=c(.999,.001))

## Not run:
tmp=replicate(1e4, rnorm.cor(10, 1, .81))
round(cor(t(tmp)),2)

## End(Not run)
Usage

getFormattedSummary(fits, type = 12, est.digits = 2, se.digits = 2, robust, random = FALSE, VE = FALSE, to.trim = FALSE, rows = NULL, coef.direct = FALSE, trunc.large.est = TRUE, scale.factor = 1, p.digits = 3, remove.leading0 = FALSE, p.adj.method = "fdr", ...)

defineComponent(object, ...)

defixedEf(object, ...)

risk.cal(risk, binary.outcome, weights = NULL, ngroups = NULL, cuts = NULL, main = "", add = FALSE, show.emp.risk = TRUE, lcol = 2, ylim = NULL, scale = c("logit", "risk")
interaction.table(fit, v1, v2, v1.type = "continuous", v2.type = "continuous", logistic.regression = TRUE)

## S3 method for class 'coxph'
defixedEf(object, exp=FALSE,robust=FALSE, ...)

## S3 method for class 'gam'
defixedEf(object, ...)

## S3 method for class 'gee'
defixedEf(object, exp = FALSE, ...)

## S3 method for class 'geese'
defixedEf(object, robust = TRUE, ...)

## S3 method for class 'tps'
defixedEf(object, exp=FALSE, robust=TRUE, ...)

## S3 method for class 'glm'
defixedEf(object, exp = FALSE, robust = TRUE, ret.robcov = FALSE, ...)

## S3 method for class 'svyglm'
defixedEf(object, exp = FALSE, robust = TRUE, ...)

## S3 method for class 'svy_vglm'
defixedEf(object, exp = FALSE, robust = TRUE, ...)

## S3 method for class 'svycoxph'
defixedEf(object, exp = FALSE, robust = TRUE, ...)

## S3 method for class 'inla'
defixedEf(object, ...)
## S3 method for class 'lm'
getFixedEf(object, ...)

## S3 method for class 'lme'
getFixedEf(object, ...)

## S3 method for class 'logistf'
getFixedEf(object, exp = FALSE, ...)

## S3 method for class 'matrix'
getFixedEf(object, ...)

## S3 method for class 'MIresult'
getFixedEf(object, ...)

## S3 method for class 'hyperpar.inla'
getVarComponent(object, transformation = NULL, ...)

## S3 method for class 'matrix'
getVarComponent(object, ...)

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

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

## S3 method for class 'geese'
predict(object, x, ...)

## S3 method for class 'tps'
predict(object, newdata = NULL, type = c("link", "response"), ...)

## S3 method for class 'geese'
residuals(object, y, x, ...)

## S3 method for class 'geese'
vcov(object, ...)

## S3 method for class 'tps'
vcov(object, robust, ...)

## S3 method for class 'logistf'
vcov(object, ...)

### Arguments

...  
object  
fit  
coef.direct
robust  
  Boolean, whether to return robust variance estimate
exp
remove.leading0

p.adj.method
cuts
ret.robcov
fits
type
est.digits
se.digits
p.digits
random
VE
transformation
weights
v1
v2
v1.type
v2.type
logistic.regression

newdata
x
y
to.trim
rows
risk
binary.outcome
ngroups
main
add
show.emp.risk
lcol
ylim
scale
trunc.large.est

scale.factor
Details

getFormattedSummary: from a list of fits, say lmer, inla fits, return formatted summary controlled
by "type". For a matrix, return Monte Carlo variance random=TRUE returns variance components
type=1: est type=2: est (se) type=3: est (2.5 percent, 97.5 percent) type=4: est se
getFixedEf returns a matrix, first column coef, second column se,
getFixedEf.matrix used to get mean and sd from a jags or winbugs sample, getVarComponent.matrix
and getFixedEf.matrix do the same thing. Each column of samples is a variable
interaction.table expects coef and vcov to work with fit.

Examples

## Annette Dobson (1990) "An Introduction to Generalized Linear Models".
## Page 9: Plant Weight Data.
ctl <- c(4.17,5.58,5.18,6.11,4.50,4.61,5.17,4.53,5.33,5.14)
trt <- c(4.81,4.17,4.41,3.59,5.87,3.83,6.03,4.89,4.32,4.69)
group <- gl(2, 10, 20, labels = c("Ctl","Trt"))
weight <- c(ctl, trt)
mlm.D9 <- lm(weight ~ group)
glm.D9 <- glm(weight ~ group)
getFormattedSummary (list(lm.D9, glm.D9), robust=FALSE)

sim.dat.tvarying.three Simulation Functions for Time-dependent Proportional Hazard Model

Description

sim.dat.tvarying.three simulates from a model with time varing age group variale of three levels,
sim.dat.tvarying.two two.

Usage

sim.dat.tvarying.three(n, followup.length, incidence.density,
age.sim = c("tvaryinggroup", "baselinegroup", "continuous","bt"),
random.censoring.rate = 0.05, seed)
sim.dat.tvarying.two(n, followup.length, incidence.density,
age.sim = c("tvaryinggroup", "baselinegroup", "continuous","bt"),
random.censoring.rate = 0.05, seed)

Arguments

n integer. Sample size.
followup.length numeric. Length of followup, in years.
incidence.density
numeric. Incidence rate per year.

age.sim
string. Choose between one of three possibilities. tvaryinggroup: age group is
time-varying covariate; baselinegroup: age group is a baseline covariate; contin-
umous: age is a continuous covariate; bt: age group by treatment interaction uses
baseline age group, while age group main effect uses time-dependent age group

random.censoring.rate
numeric. Amount of random censoring.

seed
integer. Random number generator seed.

Details
In sim.dat.tvarying.three, baseline age is uniformly distributed between 2.0 and 16.0, and divided
into three groups at 6 and 12. In sim.dat.tvarying.two, baseline age is uniformly distributed between
2.0 and 12.0, and divided into two groups at 6.

Value
Return a data frame with the following columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ptid</td>
<td>subject identifier</td>
</tr>
<tr>
<td>trt</td>
<td>treatment indicator 0/1</td>
</tr>
<tr>
<td>for.non.tvarying.ana</td>
<td>Boolean, used to subset dataset for non-time dependent analysis</td>
</tr>
<tr>
<td>C</td>
<td>censoring time</td>
</tr>
<tr>
<td>baseline.age</td>
<td>age years at baseline</td>
</tr>
<tr>
<td>agegrp</td>
<td>a factor with levels [0, 6) [6, 12) [12, 100)</td>
</tr>
<tr>
<td>baseline.agegrp</td>
<td>a factor with levels [0, 6) [6, 12) [12, 100)</td>
</tr>
<tr>
<td>tstart</td>
<td>left bound of time interval</td>
</tr>
<tr>
<td>tstop</td>
<td>right bound of time interval</td>
</tr>
<tr>
<td>d</td>
<td>event indicator</td>
</tr>
<tr>
<td>X</td>
<td>followup time, in years</td>
</tr>
</tbody>
</table>

Author(s)
Youyi Fong

See Also
make.timedep.dataset
Examples

```r
library(survival)

dat=sim.dat.tvarying.three(n=6000,followup.length=3, incidence.density=0.05,
  age.sim="tvaryinggroup", seed=1)
f.tvarying = Surv(tstart,tstop,d) ~ trt*agegrp
f = Surv(X,d) ~ trt*baseline.agegrp
fits=list()
fits["tvarying"]=coxph(f.tvarying, dat)
fits["baseline"]=coxph(f, subset(dat, for.non.tvarying.ana))
fits
```

---

### Stat Functions

#### Description

H calculates entropy.

#### Usage

```r
H(p, logbase = c("e", "2"))
mutual.info(two.way.table, logbase = c("e", "2"))
cor.mixed(x, ...)
```

```r
# Default S3 method:
cor.mixed(x, na.fun, method=c("pearson","spearman"), ...)
# S3 method for class 'vector'
cor.mixed(x, y, na.fun, method=c("pearson","spearman"), ...)
# S3 method for class 'formula'
cor.mixed(formula, data, na.fun, method=c("pearson","spearman"), ...)
```

```r
skew (x, na.rm = FALSE)
info.cor(two.way.table)
yule.y(two.by.two.matrix)
```
kappa.cor(two.by.two.matrix, weight = c(1, 1), maximum = FALSE)

l.measure(two.by.two.matrix)

Arguments

p either a count vector or a probability vector, but can not be a vector of membership indicator

logbase
na.rm
two.way.table
x
...
na.fun
method
y
formula
data
two.by.two.matrix

weight
maximum

Examples

H(rep(1/5,5))
H(rep(3,5))

string.functions String Functions

Description

%+% concatenates its arguments and returns a string.

Usage

a %+% b

contain(s1, s2)
trim(x, trim.trailing=TRUE, trim.leading=TRUE)
escapeUnderline(name)
fileStem(file.name)
firstIndex(s1, s2)
getExt(file.name)
getFileStem(file.name)
getStem(file.name)
lastIndex(s1, s2)
remove.prefix(s, sep = "_")

Arguments

a
b
s1
s2
name
file.name
s
sep
x
trim.leading
trim.trailing

Examples

x=1
x %.% "b" %.% "c"
Description

Testing functions.

Usage

hosmerlem(y, yhat, g = 10)

quick.t.test(x, y, var.equal = FALSE)

sign.test(x)

tukey.mtest(mu, ms, n)

vector.t.test(mean.x, mean.y, var.x, var.y, n)

myfisher.test(x, y, ...)

mycor.test(x, method = c("pearson", "kendall", "spearman"), idx = NULL)

Arguments

...  
y  
yhat  
g  
x  
var.equal  
method  
mu  
ms  
n  
mean.x  
mean.y  
var.x  
var.y  
idx
**VEplot**

**Examples**

```r
signtest(runif(10))
```

---

**Description**

Vaccine efficacy plots.

**Usage**

```r
VEplot(object, ...) 

## S3 method for class 'cox.zph'
VEplot(object, resid = TRUE, se = TRUE, df = 4, nsmo = 40,
       var, ylab="VE", xlab="Time", xaxt="s", cex.axis=1, ...)

## S3 method for class 'glm'
VEplot(object, X1, X2, x, ...)

## S3 method for class 'cox.zph'
myplot(object, resid = TRUE, se = TRUE, df = 4, nsmo = 40, var,
        coef.transform=NULL,
        ylab=NULL,
        xlab="Time", xaxt="s", cex.axis=1,
        ...)
```

**Arguments**

- `object`: An object
- `resid`: Boolean, whether to plot residuals
- `se`: Boolean, whether to plot confidence band
- `df`: degrees of freedom
- `nsmo`: number of points used to plot the fitted spline
- `var`: estimated variance matrix from the Cox model fit
- `xlab`: x label
- `xaxt`: x axis
- `cex.axis`: cex for axis
- `ylab`: y label
- `coef.transform`: a function to transform Cox hazard ratio estimate
**Details**

`VEplot` and `myplot.cox.zph` are extensions of `survival::plot.cox.zph` to plot VE curve and other transformations.

`myplot.cox.zph` adds the following parameters to the original list of parameters in `plot.cox.zph`:
- `coef.transform`: a function to transform the coefficients
- `ylab`: y axis label
- `xlab`: x axis label

`VEplot.glm` computes a series of k VEs: for i in 1...k, VE[i] = P(Y=1|X1[i]) / P(Y=1|X2[i]). It returns a 3 by k matrix, whose first row contains VE estimates and the second and third rows contain lower and upper bounds, respectively.

**Author(s)**

Youyi Fong, Dennis Chao

**References**


**Examples**

```r
library(survival)
vfit <- coxph(Surv(time,status) ~ trt + factor(celltype) +
              karno + age, data=veteran, x=TRUE)
temp <- cox.zph(vfit)
par(mfrow=c(2,2))
for (v in c("trt","age")) {
  VEplot(temp, var=v, resid=FALSE, main=v, ylab="VE", cex.axis=1.5)
  plot(temp, var=v, resid=FALSE, main=v)
}
library(survival)
fit <- glm(status ~ trt + trt*age, data=veteran)
summary(fit)
age=seq(min(veteran$age),max(veteran$age),length=10)
out = VEplot(fit, X1=cbind(1,1,age,1*age), X2=cbind(1,0,age,0*age), x=age)
out
```
Index

* time varying
  - make_timedep_dataset, 18
  - as.double (matrix2), 22
  - .% (string.functions), 44
  - abline.pt.slope (plotting), 25
  - abline.pts (plotting), 25
  - abline.shade (plotting), 25
  - add.mtext.label (plotting), 25
  - age_calc, 2
  - AR1 (matrix.array.functions), 20
  - array.functions (matrix.array.functions), 20
  - as.binary (math.functions), 19
  - base.functions, 3
  - binary (base.functions), 3
  - binary2 (base.functions), 3
  - binaryloess, 6
  - binom.coef (math.functions), 19
  - butterfly.plot (plotting), 25
  - cbinduneven (base.functions), 3
  - coef.Deming (Deming), 8
  - coef.geese (regression.model.functions), 37
  - coef.tps (regression.model.functions), 37
  - concatList (matrix.array.functions), 20
  - contain (string.functions), 44
  - cor.mixed (stat.functions), 43
  - corplot (plotting), 25
  - cox.zph, 8
  - cox.zph.2, 7
  - dbern (random.functions), 35
  - dcorbern (random.functions), 35
  - ddoublexp (random.functions), 35
  - Deming, 8
  - DMHeatMap, 9
  - dmixnorm (random.functions), 35
  - dnorm.norm.gamma (random.functions), 35
  - DXD (matrix2), 22
  - empty.plot (plotting), 25
  - empty2na (misc), 23
  - escapeUnderline (string.functions), 44
  - EXCH (matrix.array.functions), 20
  - expit (math.functions), 19
  - f2c (base.functions), 3
  - fileStem (string.functions), 44
  - fill.jagged.array (matrix.array.functions), 20
  - formattedDouble (print.functions), 32
  - formattedInt (print.functions), 32
  - ftoi (base.functions), 3
  - get.kfold.splits (misc), 23
  - get.sim.res, 11
  - getExt (string.functions), 44
  - getFileStem (string.functions), 44
  - getFixedEf (regression.model.functions), 37
  - getFixedEf.Deming (Deming), 8
  - getFixedEf2 (regression.model.functions), 37
  - getFormattedMCSummary (get.sim.res), 11
  - getFormattedSummary (regression.model.functions), 37
  - getK, 13
  - getMfrow (plotting), 25
  - getMidPoints (matrix.array.functions), 20
  - getStem (string.functions), 44
getVarComponent
  (regression.model.functions), 37
H(stat.functions), 43
hosmerlem(testing.functions), 46
info.cor(stat.functions), 43
INT(misc), 23
interaction.table
  (regression.model.functions), 37
interpolate(math.functions), 19
iorw, 14
kappa.cor(stat.functions), 43
keepWarnings(base.functions), 3
kid, 16
kyotil, 17
l.measure(stat.functions), 43
last(matrix.array.functions), 20
lastIndex(string.functions), 44
logDiffExp(math.functions), 19
logit(math.functions), 19
logMeanExp(math.functions), 19
logSumExp(math.functions), 19
logSumExpFor2(math.functions), 19
make.latex.coef.table
  (print.functions), 32
make.timedep.dataset, 18, 42
math.functions, 19
matrix.array.functions, 20
matrix.functions
  (matrix.array.functions), 20
matrix2, 22
MCsummary(get.sim.res), 11
meanned(base.functions), 3
methods4(base.functions), 3
misc, 23
mix(matrix.array.functions), 20
multi.outer(base.functions), 3
mutual.info(stat.functions), 43
my.interaction.plot(plotting), 25
myaggregate(base.functions), 3
myboxplot(plotting), 25
mycor(base.functions), 3
mycor.test(testing.functions), 46
mydev.off(plotting), 25
myfigure(plotting), 25
myfisher.test(testing.functions), 46
myforestplot(plotting), 25
myhist(plotting), 25
mylegend(plotting), 25
mylines(plotting), 25
mymatplot(plotting), 25
mypairs(plotting), 25
mypdf(plotting), 25
myfile(plotting), 25
myplot(plotting), 25
myplot.cox.zph(VEplot), 47
mypng(plotting), 25
mypostscript(plotting), 25
myprint(print.functions), 32
myreshapelong(base.functions), 3
myreshapewide(base.functions), 3
mysanitize.numbers(print.functions), 32
mysanitize.text(print.functions), 32
mysapply(base.functions), 3
myscale(base.functions), 3
mysystem(base.functions), 3
mytapply(base.functions), 3
mytex(print.functions), 32
mytiff(plotting), 25
mywrite(print.functions), 32
p.adj.perm, 24
panel.cor(plotting), 25
panel.hist(plotting), 25
panel.nothing(plotting), 25
pava(misc), 23
pdoublexp(random.functions), 25
permn(math.functions), 19
plotting, 25
predict.Deming(Deming), 8
predict.geese
  (regression.model.functions), 37
predict.pcc(misc), 23
predict.tps
  (regression.model.functions), 37
print.functions, 32
print.iorw(iorw), 14
qdoublexp(random.functions), 35
quick.t.test (testing.functions), 46
random.functions, 35
rank.inv.norm (misc), 23
rbern (random.functions), 35
rbigamma (random.functions), 35
rbilogistic (random.functions), 35
rdoublexp (random.functions), 35
rbidoublexp (random.functions), 35
read.sv (base.functions), 3
read.tsv (base.functions), 3
regression.model.functions, 37
rejective.sampling (random.functions), 35
remove.prefix (string.functions), 44
rep.data.frame (matrix.array.functions), 20
rep.matrix (matrix.array.functions), 20
residuals.geese (regression.model.functions), 37
risk.cal (regression.model.functions), 37
rmixnorm (random.functions), 35
rnorm.ar (random.functions), 35
rnorm.norm.gamma (random.functions), 35
roundup (print.functions), 32
sample.for.cv (misc), 23
shift.left (matrix.array.functions), 20
shift.right (matrix.array.functions), 20
sign.test (testing.functions), 46
sim.dat.tvarying.three
  (sim.dat.tvarying.two), 41
sim.dat.tvarying.two, 41
skew (stat.functions), 43
stat.functions, 43
Stirling2 (math.functions), 19
string.functions, 44
summ (misc), 23
summary.Deming (Deming), 8
symprod (matrix2), 22
table.cases (base.functions), 3
table.prop (base.functions), 3
testing.functions, 46
thin.rows (matrix.array.functions), 20
ThinRows (matrix.array.functions), 20
tr (matrix.array.functions), 20
trim (string.functions), 44
tukey.mtest (testing.functions), 46
tXDX (matrix2), 22
txSy (matrix2), 22
unix (base.functions), 3
vcov.Deming (Deming), 8
vcov.geese
  (regression.model.functions), 37
vcov.logistf (regression.model.functions), 37
vcov.tps (regression.model.functions), 37
vector.t.test (testing.functions), 46
VEplot, 47
whiskers (plotting), 25
wtd.hist (plotting), 25
yule.y (stat.functions), 43