Package 'stuRpkg'

April 1, 2018

Type Package

Title The Stu R Package

Version 1.3

Description Small suite of useful functions, subroutines, and data objects for data set manipulation and analysis in R.

License GPL-3 | file LICENSE

Depends R (>= 3.1.0),
magrittr

Imports VennDiagram,
dplyr,
tibble,
grid

Suggests gplots,
testthat

BuildVignettes false

URL https://github.com/stufield/stuRpkg

X-CRAN-COMMENT *Extra comments can go here*

LazyLoad yes

LazyData yes

Language en-US

RoxygenNote 6.0.1

R topics documented:

analyzeProjectionMatrix
Linear Matrix Analysis

Description

A quick and dirty function for the analysis of a projection matrix. Includes sensitivity and elasticity of a linear (projection) matrix model.

Usage

```
analyzeProjectionMatrix(A, initial, Gen = 25, key.thresh = 0.1,
  plots = TRUE)
```

Arguments

A	A population projection matrix.
initial	A numeric vector of the initial stage structured population. Must be the same length as the number of columns of A
Gen	An integer of the number of generations to run the projection in calculating matrix solutions
key.thresh	Numeric. The cutoff for how to limit most influential parameters. Based on Elasticities
plots	Should plots be included with the output?

Value

A list containing:

Matrix	The projection matrix
Projection	The population projection of solutions
StageProportions	The proportion each stage of the total population
lambda	The dominant eigenvalue corresponding to the growth rate of the population
SDD	The Stable Stage Distribution corresponding to the right eigenvector of the dominant eigenvalue
RV	The reproductive value corresponding to the left eigenvector of the dominant eigenvalue
DampingRatio	Ratio of the dominant eigenvalue to the sub-dominant eigenvalue
Time2Eqm	How long does it take for the population to reach equilibrium, defined as when the dominant eigenvalue is 20x the sub-dominant eigenvalue
Sensitivity	Entry-wise sensitivities of each of the matrix parameter entries, the absolute change in lambda with changes in the parameter

`SensitivityZero`
: Entry-wise sensitivities as above, for *only* non-zero entries with respect to `A`, i.e. parameters with an actual value in the projection matrix

`Elasticity`
: Entry-wise elasticities of each of the matrix parameter entries, the proportional change in lambda with changes in the parameter

`KeyPars`
: Which parameters have Elasticities greater than the argument determined by `key.thresh`, sorted by decreasing "values". Theta is the vector of model parameters

Author(s)

Stu Field

References

Caswell, H. Matrix Population Models. 2001.

See Also

`calcMatrixSensitivity`, `eigen`, `eigen.analysis`, `popbio`

Examples

```
A <- diag(1:5 / 10)
A[cbind(2:5, 1:4)] <- 3:6 / 10
A[1, 5] <- 5
analyzeProjectionMatrix(A, initial=c(1,3,5,2,1), Gen=100)
analyzeProjectionMatrix(A, initial=c(1,3,5,2,1), key.thresh=0.05, Gen=25, plots=FALSE)
```

---

assignRData *Assign Rdata Contents to a Variable*

Description

Assigns the contents of a binary `*.Rdata` or `*.rda` file to a variable rather than loading it directly into the global environment. Useful for avoiding unexpected collisions with variables in the current global environment.

Usage

```
assignRData(file)
```

Arguments

`file`
: The character path to an `*.Rdata` or `*.rda` file.

Value

If the binary Rdata file only contains a single object then this object is returned. If the file contains more than one object, a named list is returned.

Author(s)

Mike Mehan

See Also

load.

Examples

```
## Not run:
assignRData("path/to/myfile.rda")

## End(Not run)
```

BlockMat *Create a Block Matrix*

Description

Assembles a block matrix from sub-matrices mimicking coding capabilities of Matlab.

Usage

```
BlockMat(x, b)
```

Arguments

x	A list of the sub-matrices written in order they are to appear by row
b	Numeric. The number of "block" columns. The list `length(x)` must be a multiple of b.

Value

The assembled block matrix

Note

Combine sub-Matrices into larger matrix; a mimic of Matlab. Matrices *must* be as a list

Author(s)

Stu Field

Examples

```
A <- diag(1:4)
A
B <- diag(9:12)
B
I <- diag(4)
I
blocks <- list(A, B, A*B, B-I)
blocks
BlockMat(blocks, b=2)
BlockMat(blocks, b=4)
```

---

bootstrap *Generic Bootstrapping Subroutine*

Description

Provide a numeric or character vector and create bootstrap samples (with estimates) of the original vector. Also calculates CI95 using the quantile() function.

Usage

```
bootstrap(x, boot = 1000, FUN, up = 0.975, lo = 0.025)
```

Arguments

x	Character or Numeric. The original data to be bootstrapped.
boot	Number of bootstraps to perform.
FUN	Function desired for the point estimate of the original data vector (if numeric data)
up	Upper confidence limit
lo	Lower confidence limit

Value

A list containing:

BootSamples	List of the bootstrap populations created during the simulation.
BootEstimates	List of the various point estimates of each of the bootstrap samples.
CI95	Vector of the point estimate and upper & lower CI95 produced via the bootstrap samples.
SE	The standard error based on the original data. Could be used to calculate CI95 via 1.96*SE if so desired.

Author(s)

Stu Field

See Also

CI95se, quantile

Examples

```
bootstrap(x=round(runif(25,1,100)), boot=50, FUN=mean) # numeric
bootstrap(x=LETTERS[1:26], boot=50)                    # character
z <- factor(sample(c("stu","is","cool"), 10, replace=TRUE))
# bootstrap(x=z, boot=50)      # factor (ask to convert to character)
```

---

calcMatrixSensitivity *Linear Sensitivity Analysis*

Description

Sensitivity analysis of linear maps, all in one function.

Usage

```
calcMatrixSensitivity(A)
```

Arguments

A	A projection matrix to be analyzed.

Value

A list of the following:

Matrix	The original projection matrix (map)
Sensitivity	Matrix of linear sensitivity based on perturbations of parameters to lambda
Elasticity	Matrix of linear elasticity (proportional change) based on perturbations of parameters to lambda
lambda	The dominant eigenvalue of A. Projection matrix population "growth"
w	The right eigenvector or stable stage distribution of the population
v	The left eigenvector or the reproductive "value" of each stage in the population

Author(s)

Stu Field

References

Caswell, H. Matrix Population Models. 2001. Sensitivity & Elasticity.

See Also

analyzeProjectionMatrix, eigen, eigen.analysis, popbio

Examples

```
A <- diag(1:5 / 10)
A[cbind(2:5, 1:4)] <- 3:6 / 10
A[1, 5] <- 5

calcMatrixSensitivity(A)
```

---

calcRo *Net Reproductive Rate (Ro)*

Description

Calculate the Ro of a matrix, the basic reproductive ratio.

Usage

```
calcRo(TM, FM)
```

Arguments

TM	Matrix. The transition matrix, separated transition and survivorship probabilities (vital rates)
FM	Matrix. The fecundity matrix, typically non-zero entries in the first row

Value

The scalar Net Reproductive Rate (Ro)

Author(s)

Stu Field

References

Calculation of Ro from de-Camino-Beck & Lewis. 2007.

See Also

eigen, solve

Examples

```
parS <- c(0.75, 0,   0,
          0.2,  0,   0,
          0.33, 0.4, 0)
parF <- c(0, 0, 3,
          0, 0, 2,
          0, 0, 1)
TM <- matrix(parS, 3, 3, byrow = TRUE)
TM
FM <- matrix(parF, 3, 3, byrow = TRUE)
FM
A <- TM + FM
A
eigen(A)$values[1]   # Dom. eigenvalue of A

calcRo(TM, FM)
```

---

capwords *Capitalize Title Format*

Description

Change case to capitalize first letter of each word in a character string.

Usage

```
capwords(s, strict = FALSE)
```

Arguments

s	A character string in the form of a sentence to be converted to "title case" (i.e. first letter capitalized).
strict	Logical. Should first letter capitalization be *strictly* applied? See example.

Value

A character string with "title case" conversion.

Author(s)

Stu Field

See Also

toupper, tolower, strsplit

Examples

```
capwords("using AIC for model selection")
## ->  [1] "Using AIC For Model Selection"
capwords(c("using AIC", "for MODEL selection"), strict=TRUE)
## ->  [1] "Using Aic"  "For Model Selection"
##                ^^^        ^^^^^
##              "bad"       "good"
```

`CI95se` *Calculate SEM 95% Confidence Intervals*

Description

Uses standard error (of the mean) calculation to determine the 95 of a vector of data. Does not use a bootstrapping of empirical data, but the Gaussian approximation.

Usage

```
CI95se(x)
```

Arguments

`x`	Numeric. A vector of data to calculate the CI95

Value

A vector with 3 entries:

`lower`	the lower CI95
`mean`	the arithmetic mean
`upper`	the upper CI95

Author(s)

Stu Field

See Also

`bootstrap` to create CI95 via bootstrapping method.

Examples

```
CI95se(rnorm(100))
```

coefRi *Calculate Interclass Correlation Coefficient (ICC)*

Description

The Intraclass correlation coefficient (r_i; aka ICC) can be used to estimate the repeatability of a method. The value 0 -> 1. Depending on how the groups are set up, you want all your variation to be among groups (individuals), not within groups (repeats) so you want this value to be high if individuals are your groups, and low if your repeated measurements are the groups. When ICC is high, it means most of the variation is *between* treatment groups.

Usage

```
coefRi(x, groups, do.log = TRUE)
```

Arguments

x	A matrix or data frame containing the raw data of the various treatments you are testing in the ANOVA
groups	A vector of the factor groupings for x
do.log	Should data be performed on log10-transformed?

Value

A list containing:

model	Resulting ANOVA table
ICC	The intraclass correlation coefficient, the measure of similarity among individuals *within* a treatment group relative to the differences found *among* groups

Author(s)

Stu Field

References

Sokal & Rohlf (Biometry; 3rd ed.) 210-214. Sokal & Rohlf (Biometry; 2rd ed.) 211-216.

See Also

aov

Examples

```
head(Ri_data)    # internal data
x <- as.vector(as.matrix(Ri_data))[ !is.na(as.vector(as.matrix(Ri_data))) ]
coefRi(x, groups=rep(names(Ri_data), c(8, 10, 13, 6)))
```

collapse2df *Collapse Vector List to Data Frame*

Description

Function collapses a list of vectors, or a list of 1-row data frames, into a n x m data frame, where n equals the list length and m equals the vector length. All vectors must be of the same length.

Usage

```
collapse2df(x)
```

Arguments

x	A *named* list (which become the rows) containing numeric vectors of the same length. Vectors can be in the form of a numeric vector or a 1-row data frame

Details

Uses `rbind` and `Reduce` and to preform the collapse.

Value

A data frame object of the vertically collapsed vectors

Note

Functions similarly to `do.call`.

Author(s)

Stu Field

See Also

`rbind`, `Reduce`

Examples

```
tmp <- lapply(1:3, function(...) rnorm(4))
names(tmp) <- head(LETTERS, 3)
collapse2df(tmp)
tmp2 <- lapply(tmp, function(x) { names(x) <- head(letters,4); x })
collapse2df(tmp2)
```

crossTab *Cross Tabulate Summary Counts*

Description

Create a contingency table of counts generated by cross-classifying factors from groups splitting on the by= argument, and an optional secondary splitting variable. The sums of each row and column are added to the result.

Usage

```
crossTab(x, by)
```

Arguments

x	A "data.frame" or "tibble" object containing the data from which counts are desired.
by	Character. The grouping variable(s). Can be of length=1 or length=2, if length=2 a 2-dim table will be returned.

Value

A table of grouped counts based on splitting variables with sums from each factor.

Note

This is a simple wrapper around table that adds the sums of columns and rows to the final object

Author(s)

Stu Field

See Also

table, addmargins

Examples

```
crossTab(test_data, by = "Sample")
crossTab(test_data, by = c("Sample", "TimePoint"))
```

cumsumWindow *Calculate Limited Cumulative Sum*

Description

Calculate the cumulative sum of a set of numbers within a vector. The difference between this and cumsum is that it is a sliding window approach, so sums are not necessarily calculated over the entire length of the vector. When `cut == length(x)` then it is the same as cumsum. Also, for entries < window, i.e. the beginning, the entries returned will be identical to cumsum.

Usage

```
cumsumWindow(x, window)
```

Arguments

x	The vector to be summed across
window	The length/size of the window to sum within (the moving cutoff)

Details

If `window >= length(x)`, a warning is triggered and cumsumWindow reverts to cumsum.

Value

A vector of the sums of the sliding window for the cumulative sums.

Author(s)

Stu Field

See Also

cumsum

Examples

```
cumsumWindow(1:20, 5)
cumsumWindow(1:20, window=20)
cumsum(1:20)
r.vec <- sample(1:20, 100, replace=TRUE) # random vector
cumsumWindow(r.vec, 5)
```

`cumulative` *Cumulative Wrapper*

Description

Returns cumulative vector values of neighboring vector elements.

Usage

```
cumulative(x)
```

Arguments

`x`	A numeric vector

Value

A list containing:

`cum_min`	The cumulative minima of the elements of `x`
`cum_max`	The cumulative maxima of the elements of `x`
`sum`	The sum of the elements of `x`
`cum_sum`	The cumulative sum of the elements of `x`
`prod`	The product of the elements of `x`
`cum_prod`	The cumulative product of the elements of `x`

Note

Used as a mere exercise in function writing for the R tutorial.

Author(s)

Stu Field

See Also

`cumsum`, `cumprod`

Examples

```
?cumsum
cumulative(1:10)
```

diagR *Create Matrix from Vector*

Description

Matlab mimic function for producing matrices with vectors along the sub- or super-diagonal

Usage

```
diagR(x, k = 0)
```

Arguments

x	Numeric. A Vector to be placed into entries of a matrix along a diagonal.
k	Numeric. The offset from the diagonal. +1 = super-diagonal, -1 = sub-diagonal. Default = 0, which reverts to diag.

Value

A matrix with vector "x" along a diagonal, or the sub- super-diagonal if k!=0

Note

Matlab style matrix diag()

Author(s)

Stu Field

See Also

diag, zeros

Examples

```
diagR(1:7, k=-1)     # sub-diagonal
diagR(1:15, k=0)     # same as diag()
vec <- seq(15, 30, by=3)
diagR(vec, 1)        # super-diagonal
```

DriftSim *Genetic Drift Simulation*

Description

Performs an illustrative simulation of genetic drift.

Usage

```
DriftSim(p.star = 0.5, n = 50, nsim = 50, plot = "b")
```

Arguments

p.star	Initial allelic frequency of the p allele.
n	Integer. Number of individuals in the population.
nsim	Integer. Number of simulations to perform.
plot	Plot both histogram and drift with time (default) or just histogram ("h") or just line graph ("l").

Value

A plot of the simulation is returned.

Note

Simulation written for FEScUE class

Author(s)

Stu Field

References

Department of Biology, Colorado State University, Fort Collins, CO 80523-1878.

See Also

DriftSim2

Examples

```
## Not run:
DriftSim()

## End(Not run)
```

DriftSim2 *Genetic Drift Simulation 2*

Description

Performs a simulation of genetic drift according to the one described in the Evolution & Ecology Excel spreadsheets manuals

Usage

```
DriftSim2(p, Gen, n, trials)
```

Arguments

p	Numeric. Initial allelic frequency in for each of the simulations
Gen	Integer. Generations to run each simulation
n	Integer. Number of individuals in each of the trials
trials	Integer. Number of simulations to run

Value

A list containing:

p.mat	A matrix composed of the simulation of each of the drift simulations following the allelic frequency p.
P.fix	The proportion of the time the p allele becomes fixed in the population. Should approach 0.5 when trials increases.

Author(s)

Stu Field

References

Evolutionary Ecology Tutorials in Excel - Workbook.

See Also

DriftSim

Examples

```
## Not run:
set.seed(501)
DriftSim2(p=0.5, Gen=100, n=10, trials=20)

## End(Not run)
```

`getFileExt` *Get File Extension*

Description

Function to return the file extension of a character string given a path name to a file.

Usage

```
getFileExt(x)
```

Arguments

`x`	Character. String of the file name. Can be absolute or relative path, or just the basename.

Value

A string containing the extension of the file name passed in `x`.

Author(s)

Stu Field

See Also

`regexpr`, `substring`

Examples

```
getFileExt("file.pdf")
getFileExt("/home/full/path/to/file.pdf")
```

`InfectionByCounty` *Infection data by US County*

Description

Infection data describing the infections count data of an infectious disease by geographic position. Includes the following headings:

- CountyNo
- CountyNo
- CountyName
- State
- Fips
- Site
- Lat
- Long
- Population
- Area
- Density
- Infected

Format

A data frame containing 3082 individual cases/records.

Source

Stu Field

References

Infection data originally from Dylan George and used for an exercise data set in the EEID R tutorial (2010).

Examples

```
head(InfectionByCounty)
```

`Mapply` *Convenience Wrapper for mapply()*

Description

A simple wrapper for `mapply` that enables similar syntax to the `lapply` and `sapply` format and includes built in checks for argument formation and function arguments.

Usage

```
Mapply(...)
```

Arguments

`...`	Arguments passed to `mapply`. Typically the first argument is a named list, followed by additional arguments of the same length as the first. The elements of each argument are passed sequentially to the `FUN` argument, which is typically the final argument (but does not have to be).

Details

To ensure fidelity to the original structure of the data in the output, the `SIMPLIFY=FALSE` argument is hard coded but can be set by the user.

From the `mapply` description: `mapply()` calls `FUN` for the values of `...` which are re-cycled to the length of the longest, unless any have length zero. In this wrapper, recycling is *not* allowed, and all arguments must have equal length.

Value

If `SIMPLIFY=FALSE`, typically a list object with each entry the result of applying the function argument to each element of the `...`. If `SIMPLIFY=TRUE`, an attempt is made to reduce the result to a vector or matrix in a similar fashion to the `simplify=` argument of `sapply`.

Author(s)

Stu Field

See Also

`mapply`, `sapply`

Examples

```
tmp <- list(A = 5, B = 8, C = 10)
f <- function(x, y, z) { print(sprintf("%s-%s", z, y)); x + y }
out <- Mapply(tmp, 1:length(tmp), names(tmp), f)
out
```

mapVector *Map Vector*

Description

Map a factor or character vector (1-to-1 or many-2-one) to another new vector.

Usage

```
mapVector(v, from.list, to.vec, nomatch = NULL)
```

Arguments

v	The source vector to perform the mapping. Must be a character or factor vector, or be able to be converted to one. In general, the returned value is of the same class (character/factor) as the input vector.
from.list	A list of the associated levels to map in the order they will be mapped to in to.vec. Alternatively, a character vector of the same length as to.vec of only 1-to-1 mapping is desired.
to.vec	The new mapping in the same order as they appear in from.list.
nomatch	Value that unmapped values in the source vector should take. By default, NULL, unmapped values are themselves returned unchanged.
...	Additional arguments passed to internals of specific S3 methods.

Details

If v is a factor, the returned vector is also a factor (and the levels of the returned value are determined by the to.vec argument), otherwise class of the returned vector is determined by the to.vec argument. If the class of the returned vector changes, a warning is flagged.

factor -> factor character -> factor, numeric, or integer

Value

A factor (or character) vector with the new mapping applied.

Author(s)

Stu Field

See Also

setdiff, factor

Examples

```
# factor
mapVector(test_data$TimePoint,
          list(c("baseline","24 months"),c("6 months","12 months")),
          c("one","two"))

# numeric -> character (with warning)
mapVector(rep(1:4, each=10), list(c(1,2), c(3,4)), c("one","two"))

# character ("C" is un-mapped)
mapVector(head(LETTERS, 5), list(c("A","B"), c("D","E")), c("one","two"))

# character with missing (nomatch)
mapVector(head(LETTERS, 5), list(c("A","B"), c("D","E")), c("one","two"), nomatch="No")

# integer
mapVector(1:5, list(c(1,3,5), c(2,4)), c(10,100))

# character -> integer
mapVector(head(LETTERS, 5), list(c("A","B"), c("C","D","E")), 1:2)
```

`matPower` *Matrix Power*

Description

Calculate the result of exponentiation of a matrix of the form X^n, where X is a matrix and n is an integer.

Usage

```
matPower(X, n)
```

Arguments

`X`	A matrix to be multiplied.
`n`	An integer to place in the exponent (power).

Details

This technique is referred to the Power Method and is sometimes used to estimate Lambda, the growth rate of a projection matrix.

Value

A matrix, the result of X^n.

Author(s)

Stu Field

References

Stolen from somewhere online.

Examples

```
M <- diag(c(1.3, 0.4, 0.5, 0.9))
matPower(M, 10)
```

---

matrixX *Matrix Multiplication*

Description

Mimic of proper matrix multiplication without using the R syntax of

Usage

```
matrixX(A, B)
```

Arguments

A	Pre-multiplied Matrix.
B	Post-multiplied Matrix.

Details

For matrix multiplication, the ncol(A) must equal nrow(B)! and you get a nrow(A) x ncol(B) matrix as a result. Must be matrices not vectors. If vector is desired, use 1 row/col matrix.

Value

A matrix (or vector) result of matrix multiplication.

Note

This is a mimic of the MatLab version of matrix multiplication, A * B, where here is it is `matrixX(A, B)`.

Author(s)

Stu Field

References

put references to the literature/web site here

See Also

%*%

Examples

```
M <- diag(1:4)
N <- matrix(1:16, ncol = 4)
M%*%N            # R syntax
matrixX(M, N)    # using function
M * N            # Hadamard Product
```

MC_sims *Monte-Carlo Integration Simulation*

Description

Vector containing 1000 simulation estimates of the Monte-Carlo integral estimate (area under curve) for a given function. The simulation is time greedy so these data are saved as an object.

Format

Vector containing the estimates of the Monte-Carlo simulation of the integral of a given function

Source

Bill Black

References

Stu Field

Examples

```
head(MC_sims)
tail(MC_sims)
mean(MC_sims)
hist(MC_sims)
```

`mergeMetaData` *Merge Meta Data*

Description

Merge additional meta data to an existing data frame based on unique row sample identifiers. This is a wrapper around the existing R function `left_join`.

Usage

```
mergeMetaData(df, meta, ...)
```

Arguments

`df`	The existing data frame to which the meta data is to be merged.
`meta`	Can be either: 1. A data frame of additional meta data, with at least one common column name to match the samples OR 2. A file name (and path) pointing to where to get a "*.csv" file containing meta data (this option uses a call to `read.csv`).
`...`	Additional arguments passed to `left_join`.

Value

A tibble object with extra columns corresponding to the meta data added and indexed to the appropriate sample IDs.

Author(s)

Stu Field

See Also

`left_join`, `read.csv`

Examples

```
set.seed(101)
new <- data.frame(pid = sample(test_data$pid, 10), new.clin = rnorm(10))
mergeMetaData(test_data, new)            # use default 'by'
mergeMetaData(test_data, new, by ="pid") # same
```

`MonteCarloIntegral` *Monte-Carlo Integration*

Description

Solving indefinite integrals via Monte-Carlo Monte-Carlo integration of any unknown function of "x" via simulation. Calculates the area under the curve by picking numerous points and deciding if that point falls above or below the curve. This demo shows the method of Monte-Carlo and how it can be used to estimate the integral (area under a function) over a given interval of "x". Simulation mimics a random dart thrown on a dartboard style algorithm.

Usage

```
MonteCarloIntegral(n = 10000, interval, FUN, force = FALSE, quick = TRUE,
  plot = FALSE)
```

Arguments

`n`	Integer. How many points to use in the estimation. Defaults to 10000
`interval`	Range of "x" over which the function should be evaluated
`FUN`	The function to integrate. Can be any function
`force`	Logical. Should a brute force method be used to find the maximum of "y". This can sometimes be useful to avoid missing the global maximum via `optimize`, but is slower since many points along the function are independently evaluated
`quick`	Logical. Should a vectorized implementation be used?
`plot`	Logical. Should the simulation be plotted upon completion? This option is ignored if `quick=FALSE`

Details

When `quick=TRUE`, a quicker implementation is used which uses vectorized random number generation and optional plotting (plotting is slow). This vectorized version is >100x faster than the "slow" version when plotting turned off.

Value

Numeric estimation of the integral (area below the curve). If `quick=FALSE`, a real-time plot of the function with randomly added points colored by above and below the function is generated

Note

This demo is for students to visually see how the Monte-Carlo Integration works. Includes an iteration counter & a run time indicator.

Author(s)

Stu Field, William Black IV!

References

put references to the literature/web site here

See Also

optimize

Examples

```
# set objective function to optimize

mysteryFun <- function(x) {
  20*dnorm(x, mean=-1, sd=5) +
    ifelse(x > -1.1,
    6*dgamma(x=x+1,shape=2,scale=0.5),
    1.5*dgamma(x=-x,shape=5,scale=0.2)) +
    2*dgamma(x=2.75-x, shape=3, scale=0.25)
}

# compare quick=T vs. quick=F

## Not run:
MonteCarloIntegral(n=10000, interval=c(-2.98, 2.98), FUN=mysteryFun, quick=FALSE)
MonteCarloIntegral(n=10000, interval=c(-2.98, 2.98), FUN=mysteryFun)

## End(Not run)

# 1000 simulations
head(MC_sims)
mean(MC_sims)

# Plot histogram of 1000 estimates
hist(MC_sims, col="gray75", prob=TRUE, xlab="Area", main="", breaks=15)
box()
lines(density(MC_sims))
lines(density(MC_sims, adjust=1.75), lty="dotted", col=2)
par <- density(MC_sims)$x[which.max(density(MC_sims)$y)]
abline(v=par, col=4, lty="dotted")
legend("topleft", legend=sprintf("Area Est ~ 0.3%f", par),
       bg="gray75", cex=0.75)

# check that histogram sums to 1
H <- hist(MC_sims, plot=FALSE)
print(sum((H$breaks[2]-H$breaks[1])*H$density), digits=10)
```

NormFun *Normal Distribution Histogram*

Description

Describes a Gaussian (Normal) distribution given various arguments and plots them in a histogram along with a line composed by dnorm.

Usage

```
NormFun(n = 1000, mu = 400, sd = 25, bks = 25, ...)
```

Arguments

n	Numeric. The number of observations.
mu	Numeric. A vector of means.
sd	Numeric. A vector of standard deviations.
bks	How to set the breaks for plotting the histogram.
...	Additional arguments passed to hist.

Details

Neat function for making histogram with density line as well.

Value

A histogram with the random observations produced in rnorm() with a line produced with dnorm.

Note

Used as an example to demo how to create functions in the R tutorial.

Author(s)

Stu Field

See Also

hist, rnorm, dnorm, curve

Examples

```
NormFun()
```

optimizer *Optimization Function*

Description

Given a function, determine the set of parameters that maximizes (default) or minimizes it over a given interval via brute force.

Usage

```
optimizer(fn, I, max = TRUE, ...)
```

Arguments

fn	Function to be evaluated. Must return a scalar value
I	Numeric. Interval over which to find the max (or min)
max	Logical. If FALSE, the Min is computed
...	Arguments passed to the function

Value

A plot containing the functional curve, max/min, y-value at Max/Min

Note

This differs from the optim function but essentially does the same. This is a brute force method.

Author(s)

Stu field

See Also

optimize, optim

Examples

```
myfun <- function(x, a, b, t) {
   -t * (x-a)^2 + b
}

optimizer(fn=myfun, I=c(-10,10), max=TRUE, a=2, b=12, t=0.1)
optimizer(fn=myfun, I=c(-10,10), max=FALSE, a=2, b=12, t=0.1)
```

path2file *Determine Path to File*

Description

Determine the path of a defined file via brute force search of root directory and optionally change the R working directory to the directory containing that file. Useful in setting the WD prior to a `read.csv` call.

Usage

```
path2file(file, root.dir = Sys.getenv("HOME"), switchDir = FALSE)
```

Arguments

`file`	Character string of the desired file, typically `"*.R"` or `"*.csv"`
`root.dir`	The path of the root directory. The higher up the root to travel the longer the search will take If you narrow down the path, more specific, it will take less time. The default is a Mac/Unix formulation. For Windows/PC machines, `root.dir` must be specified
`switchDir`	Logical. Set the working directory to the path of the file in `file`, if found. Uses `setwd`.

Details

If the file in `filename` is not found, function is stopped and warning printed.

Value

The path of the file specified in `file`

Author(s)

Stu Field, Steven Mosher

References

Assistance from Steven Mosher via StackOverflow

See Also

`list.files`, `setwd`

Examples

```
## Not run:
filename <- "TreeData.csv"
path2file(filename)
path2file(filename, switchDir = TRUE)
read.csv(filename)

## End(Not run)
```

`plotErrorBars` *Plot Error Bars*

Description

Produce a barplot (or points) containing plot error bars.

Usage

```
plotErrorBars(x, lo, up, plotfun = graphics::barplot, bar.col = 1,
  ylims = NULL, bar.lty = 1, bar.cex = 0.33, bar.lwd = 1, pad = 0.05,
  ...)
```

Arguments

`x`	Summary data to plot into a barplot, typically a vector of the bar heights or points.
`lo`	Numeric. A vector of heights of the *lower* error bars to be drawn. Must be computed externally and passed into the function.
`up`	Numeric. A vector of heights of the *upper* error bars to be drawn. Must be computed externally and passed into the function.
`plotfun`	Currently one of: `barplot, points, or plot`. Note: plot or points produce the same output.
`bar.col`	Color of the error bars and lines.
`ylims`	The y-axis limits to pass to `plotfun`.
`bar.lty`	Line type for the error bars. Argument is passed eventually to `lines`, `options=1:5`.
`bar.cex`	Character expansion for the width of the error bars (hat/base) in units of *bar widths*.
`bar.lwd`	Line width for the error bars (`=lwd`) passed to `lines`.
`pad`	Additional spacing, as a proportion of the y-value range, to pad the y-axis limits.
`...`	Additional arguments passed to either `barplot` , `plot`, or `plot`.

Value

A plot with error bars plotted on top defined by up and lo.

Author(s)

Stu Field

References

Heavily modified from The R Book. 2007. Michael Crowley.

See Also

plotCI, barplot, plot, lines

Examples

```
set.seed(101)
x <- rnorm(10)
plotErrorBars(x, lo = x - sd(x), up = x + sd(x))

# Barplot with Error Bars (SEMs & CIs)
set.seed(101)
f <- factor(rep(LETTERS[1:4], each = 20))
x <- runif(80)
data <- data.frame(x, f)
GroupMeans <- tapply(data$x, data$f, mean)
GroupSD <- tapply(data$x, data$f, sd)
sem <- GroupSD / sqrt(length(GroupMeans))
ci <- (GroupMeans + (sem * 1.96)) - (GroupMeans - (sem * 1.96))

par(mfrow=c(1, 2))
plotErrorBars(GroupMeans, lo = GroupMeans - sem, up = GroupMeans+sem)
plotErrorBars(GroupMeans, lo = GroupMeans - ci, up = GroupMeans+ci)

# Points with Error Bars
plotErrorBars(GroupMeans, lo = GroupMeans - sem, up = GroupMeans+sem, plotfun = plot,
              bar.col = 2, bar.lwd = 2, pch = 21, bg = 4, cex = 2)
```

---

popdata *6 Class Population Projection*

Description

This data set it the data produced by the projection for the 6 class example used in the R tutorial

Format

A matrix of the 6-class population projection with classes as columns and the map/step solutions as the rows

Source

Stu Field

References

Stu Field, The R Tutorial.

Examples

```
head(popdata, 10)
```

`print2file` *Print Object to File*

Description

A convenient wrapper around the `sink` function to print the resulting `print` command to a corresponding file connection. Useful for sharing data only results with collaborators or medical group.

Usage

```
print2file(..., file, width = 250)
```

Arguments

`...`	Object(s) to be printed to file as standard output.
`file`	File name for the output.
`width`	Numeric. Controls the maximum number of columns on a line used in printing vectors, matrices and arrays, and when filling by `cat`. See also `getOption("width")`.

Author(s)

Stu Field

See Also

`sink`, `print`

Examples

```
tab <- crossTab(test_data, c("Pop", "Sample"))
print2file(tab, file="table_file.txt")
```

PunnettPack *Packing Punnett Map Solutions (FEScUE)*

Description

Used to pack and unpack the "Punnett Square" formulation for Tribolium model in FEScUE. Each slice of the 3D array is a "punnett square" that is:

$$\begin{pmatrix} AA & Aa \\ aA & aa \end{pmatrix}$$

Usage

```
PunnettPack(x, nallele, nstages)
```

Arguments

x	The object to be packed into an array. Typically a vector of length = nallele^2 * nstages.
nallele	Number of alleles
nstages	Number of base stages.

Value

A 3-dim array with `dim = c(nallele, nallele, nstages)`.

Note

Based on the Jesse Drendel formulation of the Tribolium model in FEScUE for the Tribolium Project.

Author(s)

Stu Field, Jesse Drendel

Examples

```
PunnettPack(1:24, nallele=2, nstages=6)
```

`removeColumns` *Remove Column Containing Given Entry(s)*

Description

Remove an entire column of a matrix of data frame whose entries contain a given value. Created in SNP analysis data and is similar to `na.omit` except the columns (default) are removed is a match occurs in that margin.

Usage

```
removeColumns(x, index = 2, search)
```

Arguments

`x`	A matrix or data frame
`index`	Numeric (1,2). The margin to index (columns = 2; rows = 1)
`search`	Numeric or character. The exact match to be searched. If contained within any column or row, it will be removed

Value

Matrix or data frame with either row(s) or column(s) removed as a result of matching `search`

Note

Used in SNP analysis to remove nucleotide positions containing a given character

Author(s)

Stu Field

See Also

`na.omit`

Examples

```
M <- matrix(1:25, ncol=5)
removeColumns(M, index=2, search=20)
M <- as.data.frame(M)
M[4, 4] <- "A"
M
removeColumns(M, search=c("A", 17, 23))
```

Ri_data *Coefficient Interclass Correlation Data Example*

Description

Coefficient Interclass Correlation Data Example from Sokal & Rohlf (Biometry; 3rd ed.), pages 210-214.

Format

A data frame of data with treatments as columns and rows as cases

Source

Stu Field

References

Sokal & Rohlf (Biometry; 3rd ed.), p. 210-214.

Examples

```
Ri_data
```

rMat *Create Random Matrix*

Description

Generate a matrix with random numbers in its entries.

Usage

```
rMat(m, n, min, max, replace = FALSE, ...)
```

Arguments

m	Numeric. Row dimension of the matrix to be produced
n	Numeric. Column dimension of the matrix to be produced
min	Numeric. Minimum random number in range
max	Numeric. Maximum random number in range
replace	Logical. Should the sampled numbers be repeated?
...	Additional arguments passed to matrix

Value

A matrix with m x n dimensions full of random numbers on the interval [min, max]

Note

Never know when you need a matrix of random numbers!

Author(s)

Stu Field

See Also

`sample`

Examples

```
rMat(5, 6, 35, 75)
rMat(5, 6, 35, 75, replace=TRUE)
```

`rotateMatrix` *Rotate a Matrix*

Description

Function rotates a matrix 90 degrees. Direction can be either clockwise or counter-clockwise

Usage

```
rotateMatrix(x, direction = c("clock", "counter-clock"))
```

Arguments

`x`	A matrix object
`direction`	Character. Which direction to rotate 90 degrees. Partial matching allowed

Value

A matrix rotated 90 degrees from the original matrix

Author(s)

Stu Field

See Also

`match.arg`, `t`

Examples

```
m <- matrix(1:9, ncol=3)
m
rotateMatrix(m, "clock")
rotateMatrix(m, "counter")
```

`searchReplace` *Search & Replace*

Description

A global search & replace of entries within a vector, matrix, or data frame

Usage

```
searchReplace(x, s, r)
```

Arguments

`x`	The object to be searched, typically a matrix or data frame but can be a vector of character or numeric class.
`s`	The search index.
`r`	The replace with. Must be same length as `s`.
`...`	Pass through to various S3 methods.

Details

Warning: The lengths of `s` and `r` *must* be identical.

Value

An object of the same dimensions and class as `x`, with the `s=` matches replaced with `r=`.

Author(s)

Stu Field

See Also

`replace`

Examples

```
# matrix
Y <- matrix(1:25, ncol = 5)
Y
searchReplace(Y, s = c(8, 20), r = c(99, 99))

# data.frame
X <- data.frame(x = c(1, 2, 3), y = c(2, 2, 4), z = c(1, 2, 4))
rownames(X) <- c("one", "two", "three")
X
searchReplace(X, s = 1:4, r = c("A", "C", "G", "T"))

# numeric
searchReplace(1:10, s = 4, r = 19)

# character
searchReplace(head(LETTERS, 10), s = "G", r = "Z")
```

---

seconds2time *Determine Time from Seconds (& vice versa)*

Description

time2seconds and seconds2time determine the time (format="hh:mm:ss.ss") from the value in seconds or vice versa. Two digit hour precision optional.

Determine Seconds from Time

Usage

```
seconds2time(x)

time2seconds(x)
```

Arguments

x	Character or Numeric. Of the form "hh:mm:ss.ss" with 2 decimal point precision on the seconds to convert to seconds. If numeric, the seconds to convert to a character string of the form "hh:mm:ss.ss".

Value

Either number of seconds (numeric or the time format (string) in "hh:mm:ss.ss".

Author(s)

Stu Field

See Also

strsplit, gsub, grepl, sprintf

Examples

```
seconds2time(159.72)
time2seconds("3:44:12.04")
time2seconds("15:44:12.04")
```

---

subapply *Apply Function to Subsets of Data*

Description

Apply a function, written on the fly (.fun), to the subsets of data determined by an appropriate subsetting variable (column) If the x is a vector, the index argument must be of the same length as x, and the function reduces to a simple tapply. If x is a data frame, then y must be supplied and corresponds to a column in x.

S3 subapply method for data.frame

Usage

```
subapply(x, index, .fun, ...)

## S3 method for class 'data.frame'
subapply(x, index, .fun, y, longform = FALSE,
  sum.field = deparse(substitute(.fun)), ...)
```

Arguments

x	A vector, data frame, of matrix of values evaluate.
index	If x is a vector, the sub-setting index of same length as x. If x is a data frame, a character string indicating the column name(s) in x to use as indices. Character string currently cannot be more than length 2.
.fun	The function to apply to each subset of the data. Can be written on the fly using the function(x) syntax.
...	Additional arguments passed to tapply
y	Character. Only used if x is a data frame, then a character string corresponding to a column name in x.
longform	Alternative version of output for the returned value. Returns a table.
sum.field	Character. The summary field column name in the returned data frame that summarizes the function called upon each subset of the data. Ignored if longform=FALSE.

Details

THIS FUNCTION WILL EVENTUALLY BE REPLACED BY summarise.

Value

A data frame with summary data, and the result of the applied function.

Author(s)

Stu Field

See Also

tapply, select

Examples

```
# S3 numeric
subapply(test_data$z, index = test_data$Sample, .fun = mean)

# S3 character
set.seed(10)
y <- sample(LETTERS, 10, replace=TRUE)
subapply(y, rep("a",length(y)), duplicated)

# S3 data.frame
subapply(test_data, index = "Sample", .fun = mean, y = "z")  # same as above

# cross tabulated
subapply(test_data, c("Sample", "Response"), .fun = mean, y = "z")

# cross tabulated long-format
subapply(test_data, index = c("Sample","Response"), .fun = mean, y = "z",
         longform = TRUE, sum.field = "Mean")

# S3 matrix
test_data$Sample %<>% as.numeric
m <- test_data[, c(1, 3, 11)] %>% data.matrix
subapply(m, index = "Sample", .fun = mean, y = "z", longform = TRUE) # same as data.frame
```

---

test_data *Sample Test Data Frame*

Description

A quick sample tibble data frame for running examples and checking data frame functionalities. See ?tibble.

Format

test_data a tibble object:

Column	Definition
pid	seq(1041, 1080, 1)
Pop	rep(LETTERS[1:10])
Sample	sample(c("small", "medium", "large"), 20)
TimePoint	rep(c("baseline", "6 mths", "12 mths", "24 mths"), each = 10)
a	1:length(test_data$TimePoint)
b	sample(1:10, 40, replace = TRUE)
ABCD.1234.56.8	rnorm(40, 25, 3)
XYZZ.6969.4.7	rnorm(40, 25, 3)
x	test_data$a * runif (length(test_data$a), 0.25, 0.75)
y	sample(11:20, 40, replace = TRUE)
z	Mod(round(sample(rnorm(40)), 3))
Response	sample(c("control", "disease"), 40, replace = TRUE))

Source

Stu Field

Examples

```
dim(test_data)
head(test_data)
```

TreeData *Tree Data*

Description

Data set containing tree characteristic metric data, that has been blinded and anonymized from its original source. Used heavily in the R tutorial and for general data frame usage (e.g. package testing).

Format

A data frame containing 20 cases and numerous random variables

Source

Stu Field

References

Originally from Sala et al., but heavily modified to be unrecognizable and used as the main example data set for the R tutorial.

Examples

```
head(TreeData)
summary(TreeData)
suppressWarnings(sapply(TreeData, mean))
```

`TribMate` *Tribolium Mating Subroutine*

Description

A sub-routine used to determine the number of offspring produced by the current population according to Hardy-Weinberg predictions. Used in `Tribolium9()` where only the top class (adults) mates.

Usage

```
TribMate(Y, f)
```

Arguments

`Y`	A matrix containing an intermediate population with the classes as rows & genotypes as cols. Columns should be ordered AA, Aa, aa. Only the final class mates.
`f`	Integer representing the fecundity (i.e. egg production) of the female mating partner. Number of offspring produced per reproductive adult.

Value

A matrix of identical dimensions to Y containing non-zero entries in the first row only representing the newly produced offspring (eggs) in the population. This matrix can then simply be added to the current, or intermediate, solution to obtain the current solution.

Note

Used in FEScUE for `Tribolium9()`.

Author(s)

Stu Field, Jesse Drendel

References

Tribolium Example from Caswell (2008); pg 71. Perturbation analysis of nonlinear matrix popn models. Demographic Research 18: 59-116

See Also

`Tribolium9`

`tryNULL` *Try NULL Function*

Description

A convenient wrapper around either `try` or `tryCatch`. However, rather than of returning the an error signal/warning, `NULL` is returned if an error is encountered. This makes it easier to test the status of the returned object later in the code with `is.null`.

Usage

```
tryNULL(expr, default = NULL, quiet = TRUE)
```

Arguments

`expr`	an R expression to try.
`default`	The value to return if an error is encountered. By default, and according to the function name, `NULL`. This option is included for the rare case when something other than `NULL` is desired.
`quiet`	Logical. Should the error message (if encountered) be printed to the console?

Value

Either the value of the expression in `expr` or, if an error is encountered, the value in `default`.

Author(s)

Stu Field

References

More or less stolen from the `plyr` package, with some modifications.

See Also

`try`, `tryCatch`, `tryNULL`

Examples

```
tryNULL(log("a"))
tryNULL(log("a"), quiet = FALSE)
tryNULL(log("a"), default = 0)
```

---

vennWrapper *Venn Diagram Plotting Routine*

Description

A wrapper function for plotting Venn diagrams comprising 1 - 5 intersections.

Usage

```
vennWrapper(x, ..., edge.col = "transparent", colors = seq(length(x)),
  num.cex = 1, alpha = 0.25, label.col = "gray35", fontfamily = "sans",
  main.fontface = "bold", sub.fontface = "bold", main.col = 1,
  sub.col = "gray35", main.cex = 3, sub.cex = 1.5, margin = 0.01,
  cat.fontface = "bold", cat.cex = 2, cat.col = "black",
  cat.fontfamily = "sans", rotation.degree = 0, filename = NULL)
```

Arguments

x	A named list of vectors containing strings to match intersections.
...	Additional arguments passed to the internal venn_default, which was stolen mostly from VennDiagram.
edge.col	describe
colors	describe
num.cex	describe
alpha	describe
label.col	describe
fontfamily	describe
main.fontface	describe
sub.fontface	describe
main.col	describe
sub.col	describe
main.cex	describe
sub.cex	describe
margin	describe
cat.fontface	describe

`cat.cex`	describe
`cat.col`	describe
`cat.fontfamily`	describe
`rotation.degree`	Numeric (0-360). Rotation for the entire diagram.
`filename`	Character. Path and file name to which to print the figure. Default of `NULL` or `NA` prints to the null device.

Details

More details if necessary

Value

A grob class object which can be plotting to a device of printed to file if `filename` is not `NULL`.

Note

I stole this ...

Author(s)

Stu Field

References

See the `VennDiagram` package.

See Also

`VennDiagram`, `grid`

Examples

```
int.list <- lapply(1:3, function(...) sample(LETTERS[1:10], 6))
names(int.list) <- c("Larry", "Curly", "Mo")
vennWrapper(int.list, num.cex=seq(0.5,2,length=7), cat.cex=c(1,1.5,2),
            main="Title Here", sub="Subtitle")
```

whichEntry *Which Entry(s)*

Description

Determine which entry of a matrix matches a given value. Similar functions exist for a vector, but when applied to a matrix entries are converted to a vector of the matrix columns. Not ideal. This function provides row x column addressing of the entries which agree with the `match` argument.

Usage

```
whichEntry(x, match)
```

Arguments

`x`	Matrix object to be searched
`match`	Index to be matched

Value

A matrix indicating which entries agree with `match=`. Rows are the number of different match(es) and column 1 is the row of a positive match and column 2 is the column of the positive match.

Author(s)

Stu Field

See Also

`which`

Examples

```
M <- matrix(1:25, ncol=5)
M[5,1]  <- 17
M[4, 2] <- 17
whichEntry(M, 17)
```

`zeros` *Diagonal Matrix of Zeros*

Description

Produce a matrix of zeros of dimensions determined by the diagonal argument.

Usage

```
zeros(x)
```

Arguments

`x`	Numeric. The dimensions (m x n) for the square matrix to be produced, or the length of the diagonal of the square matrix

Details

Mimic of the MatLab function of a similar name.

Value

A square matrix of zeros with `m x n` dimensions where both equal `x`

Note

Mimic of MatLab, even though MatLab is bullshit.

Author(s)

Stu Field

See Also

`diag`, `diagR`

Examples

```
zeros(7)
zeros(15)
zeros(c(3, 5))
```

Index