installReps

Install all repository packages

Description
Using reposTools, install/update all packages from given repositories.

Usage
installReps(repNames = "aData", lib = .libPaths()[1],
            type =getOption("pkgType"))

Arguments
repNames A character vector containing repository names. A listing of known repository
          names can be found with the getReposOption() function.
lib A path to install/update the packages. If this directory does not exist, it will be
     created (if possible).
type type of package do download

Author(s)
Colin A. Smith (webbioc@colinsmith.org)
**mt.wrapper**

*Multiple Testing Wrapper Function*

**Description**

A wrapper for some of the functionality for the multtest package. It also includes hooks to calculate q-values with John D. Storey’s ‘q-value.R’ code.

**Usage**

```
mt.wrapper(proc, X, classlabel, test="t", rawpcalc="Parametric", side="abs", ...)
```

**Arguments**

- `proc`: A character string containing the name of the multiple testing procedure for which adjusted *p*-values are to be computed. This vector should include any of the following: "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BH", "BY", "maxT", "minP", "q".

- `X`: A data frame or matrix, with *m* rows corresponding to variables (hypotheses) and *n* columns to observations. In the case of gene expression data, rows correspond to genes and columns to mRNA samples. The data can be read using `read.table`.

- `classlabel`: A vector of integers corresponding to observation (column) class labels. For *k* classes, the labels must be integers between 0 and *k* − 1. For the `blockf` test option, observations may be divided into *n*/*k* blocks of *k* observations each. The observations are ordered by block, and within each block, they are labeled using the integers 0 to *k* − 1.

- `test`: A character string specifying the statistic to be used to test the null hypothesis of no association between the variables and the class labels. If `test="t"`, the tests are based on two-sample Welch t-statistics (unequal variances). If `test="t.equalvar"`, the tests are based on two-sample t-statistics with equal variance for the two samples. The square of the t-statistic is equal to an F-statistic for *k* = 2. If `test="wilcoxon"`, the tests are based on standardized rank sum Wilcoxon statistics. If `test="f"`, the tests are based on F-statistics. If `test="pairt"`, the tests are based on paired t-statistics. The square of the paired t-statistic is equal to a block F-statistic for *k* = 2. If `test="blockf"`, the tests are based on F-statistics which adjust for block differences (cf. two-way analysis of variance).

- `rawpcalc`: A character string specifying how to calculate nominal/raw *p*-values. The possible choices are "Parametric" or "Permutation".

- `side`: A character string specifying the type of rejection region. If `side="abs"`, two-tailed tests, the null hypothesis is rejected for large absolute values of the test statistic. If `side="upper"`, one-tailed tests, the null hypothesis is rejected for large values of the test statistic. If `side="lower"`, one-tailed tests, the null hypothesis is rejected for small values of the test statistic.
Further arguments for `mt.maxT, mt.minP, qvalue`.

**Value**

A data frame with components

- **index**: Vector of row indices, between 1 and `nrow(X)`, where rows are sorted first according to their adjusted $p$-values, next their unadjusted $p$-values, and finally their test statistics.

- **teststat**: Vector of test statistics, ordered according to `index`. To get the test statistics in the original data order, use `teststat[order(index)]`.

- **rawp**: Vector of raw (unadjusted) $p$-values, ordered according to `index`.

- **adjp**: Vector of adjusted $p$-values, ordered according to `index`.

- **plower**: For `mt.minP` function only, vector of "adjusted $p$-values", where ties in the permutation distribution of the successive minima of raw $p$-values with the observed $p$-values are counted only once. Note that procedures based on `plower` do not control the FWER. Comparison of `plower` and `adjp` gives an idea of the discreteness of the permutation distribution. Values in `plower` are ordered according to `index`.

**Author(s)**

Colin A. Smith (webbioc@colinsmith.org)
Index

*Topic file
  installReps, 1
  mt.wrapper, 2

installReps, 1

mt.minP, 3
mt.wrapper, 2

read.table, 2