

# Package ‘gMCP’

August 26, 2014

**Type** Package

**Title** Graph Based Multiple Comparison Procedures

**Version** 0.8-7

**Maintainer** Kornelius Rohmeyer <rohmeier@small-projects.de>

**Description** This package provides functions and a graphical user interface for graphical described multiple test procedures.

**Depends** R (>= 2.8.0), rJava (>= 0.6-3), JavaGD, methods, xlsxjars (>= 0.6.1)

**Imports** MASS, PolynomF, multcomp (>= 1.1), mvtnorm, Matrix,CommonJavaJars (>= 1.0.5), stats4

**Suggests** RUnit, Deducer, knitr, graph (>= 1.20), mutoss, boot

**SystemRequirements** Java (>= 5.0)

**License** GPL (>= 2)

**URL** <http://gsrmtpl.r-forge.r-project.org/>

**BugReports** <https://github.com/kornl/gMCP/issues>

**Author** Kornelius Rohmeyer [aut, cre], Florian Klinglmueller [aut]

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2014-08-26 13:33:31

## R topics documented:

gMCP-package . . . . .	2
bdiagNA . . . . .	4
calcPower . . . . .	5
corMatWizard . . . . .	7
doInterim . . . . .	8
entangledMCP-class . . . . .	10

exampleGraphs . . . . .	11
extractPower . . . . .	14
generateBounds . . . . .	15
generatePvals . . . . .	17
generateTest . . . . .	19
generateWeights . . . . .	21
getJavaInfo . . . . .	22
gMCP . . . . .	22
gMCPReport . . . . .	25
gMCPResult-class . . . . .	26
gPADInterim-class . . . . .	26
graph2latex . . . . .	27
graphAnalysis . . . . .	28
graphGUI . . . . .	29
graphMCP-class . . . . .	31
graphTest . . . . .	32
hydroquinone . . . . .	34
joinGraphs . . . . .	35
matrix2graph . . . . .	36
placeNodes . . . . .	37
plotSimCI . . . . .	38
rejectNode . . . . .	39
replaceVariables . . . . .	40
rqmvnorm . . . . .	41
secondStageTest . . . . .	42
simConfint . . . . .	43
simvastatin . . . . .	45
subgraph . . . . .	46
substituteEps . . . . .	47
unitTestsGMCP . . . . .	48
<b>Index</b>	<b>49</b>

gMCP-package

*Graphical described Multiple Comparison Procedures***Description**

This package provides functions and graphical user interfaces for graphical described multiple comparison procedures.

**Details**

Package: gMCP  
 Type: Package  
 License: GPL (>= 2)  
 LazyLoad: yes

The package gMCP helps with the following steps of performing a multiple test procedure:

1. Creating a object of `graphMCP` that represents a sequentially rejective multiple test procedure. This can be done directly via the new method or a graphical user interface by calling `graphGUI`.
2. Calling `gMCP` or `graphGUI`.
3. Exporting the results (optional with all sequential steps) as LaTeX or PDF report.

## Note

We use the following Java libraries:

- Apache Commons Logging under the Apache License, Version 2.0, January 2004, <http://commons.apache.org/logging/>, Copyright 2001-2007 The Apache Software Foundation
- Apache jog4j under Apache License 2.0, <http://logging.apache.org/log4j/>, Copyright 2007 The Apache Software Foundation
- Apache Commons Lang under Apache License 2.0, <http://commons.apache.org/lang/>, Copyright 2001-2011 The Apache Software Foundation
- JLaTeXMath under GPL  $\geq$  2.0, <http://forge.scilab.org/index.php/p/jlatexmath/>, Copyright 2004-2007, 2009 Calixte, Coolsaet, Cleemput, Vermeulen and Universiteit Gent
- JRI under Lesser General Public License (LGPL) 2.1, <http://www.rforge.net/rJava/>, Copyright 2004-2007 Simon Urbanek
- iText 2.1.4 under LGPL, <http://itextpdf.com/>, Copyright by Bruno Lowagie
- SwingWorker under LGPL, <https://swingworker.dev.java.net/>, Copyright (c) 2005 Sun Microsystems
- JXLayer under BSD License, <https://jxlayer.dev.java.net/>, Copyright 2006-2009, Alexander Potochkin
- JGoodies Forms under BSD License, <http://www.jgoodies.com/freeware/forms/>, Copyright JGoodies Karsten Lentzsch
- AFCommons under BSD License, <http://www.algorithm-forge.com/afcommons/>, Copyright (c) 2007-2010 by Kornelius Rohmeyer and Bernd Bischl
- JHLIR under BSD License, <http://jhlir.r-forge.r-project.org/>, Copyright (c) 2008-2010 by Bernd Bischl and Kornelius Rohmeyer
- SGTK under BSD License, <http://www.algorithm-forge.com/rjavaclient/>, Copyright (c) 2007-2010 by Bernd Bischl and Kornelius Rohmeyer

## Author(s)

Kornelius Rohmeyer, R code for correlated tests and adaptive designs from Florian Klinglmueller.

Maintainer: Kornelius Rohmeyer <rohrmeyer@small-projects.de>

## References

Frank Bretz, Martin Posch, Ekkehard Glimm, Florian Klinglmueller, Willi Maurer, Kornelius Rohmeyer (2011): Graphical approaches for multiple comparison procedures using weighted Bonferroni, Simes or parametric tests. *Biometrical Journal* 53 (6), pages 894-913, Wiley. <http://onlinelibrary.wiley.com/doi/10.1002/bimj.201000239/full>

Frank Bretz, Willi Maurer, Werner Brannath, Martin Posch: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)

### Examples

```
g5 <- BonferroniHolm(5)
## Not run:
graphGUI("g5")
## End(Not run)
gMCP(g5, pvalues=c(0.1,0.2,0.4,0.4,0.4))
```

---

bdiagNA

*Create a Block Diagonal Matrix with NA outside the diagonal*

---

### Description

Build a block diagonal matrix with NA values outside the diagonal given several building block matrices.

### Usage

```
bdiagNA(...)
```

### Arguments

... individual matrices or a list of matrices.

### Details

This function is useful to build the correlation matrices, when only partial knowledge of the correlation exists.

### Value

A block diagonal matrix with NA values outside the diagonal.

### Author(s)

Kornelius Rohmeyer <rohmeier@small-projects.de>

### See Also

[gMCP](#)

### Examples

```
bdiagNA(diag(3), matrix(1/2,nr=3,nc=3), diag(2))
```

calcPower

*Calculate power values***Description**

Given the distribution under the alternative (assumed to be multivariate normal), this function calculates the power to reject at least one hypothesis, the local power for the hypotheses as well as the expected number of rejections.

**Usage**

```
calcPower(weights, alpha, G, mean = rep(0, nrow(sigma)),
  sigma = diag(length(mean)), cr = NULL, nSim = 10000,
  type = c("quasirandom", "pseudorandom"), f = list(),
  test)
```

**Arguments**

weights	Initial weight levels for the test procedure, see graphTest function.
alpha	Overall alpha level of the procedure, see graphTest function. (For entangled graphs alpha should be a numeric vector of length equal to the number of graphs, each element specifying the partial alpha for the respective graph. The overall alpha level equals sum(alpha).)
G	Matrix determining the graph underlying the test procedure. Note that the diagonal need to contain only 0s, while the rows need to sum to 1. When multiple graphs should be used this needs to be a list containing the different graphs as elements.
mean	Mean under the alternative
sigma	Covariance matrix under the alternative.
cr	Correlation matrix that should be used for the parametric test. If cr==NULL the Bonferroni based test procedure is used. Can contain NAs.
type	What type of random numbers to use. quasirandom uses a randomized Lattice rule, and should be more efficient than pseudorandom that uses ordinary (pseudo) random numbers.
nSim	Monte Carlo sample size. If type = "quasirandom" this number is rounded up to the next power of 2, e.g. 1000 is rounded up to $1024 = 2^{10}$ and at least 1024.
f	List of user defined power functions (or just a single power function). If one is interested in the power to reject hypotheses 1 and 3 one could specify: <code>f=function(x) {x[1] &amp;&amp; x[3]}</code> . If the power of rejecting hypotheses 1 and 2 is also of interest one would use a (optionally named) list: <code>f=list(power1and3=function(x) {x[1] &amp;&amp; x[3]},          power1and2=function(x) {x[1] &amp;&amp; x[2]})</code> . If the list has no names, the functions will be referenced to as "func1", "func2", etc. in the output.

**test** In the parametric case there is more than one way to handle subgraphs with less than the full alpha. If the parameter `test` is missing, the tests are performed as described by Bretz et al. (2011), i.e. tests of intersection null hypotheses always exhaust the full alpha level even if the sum of weights is strictly smaller than one. If `test="simple-parametric"` the tests are performed as defined in Equation (3) of Bretz et al. (2011).

## Value

A list containing three elements

**LocalPower** A numeric giving the local powers for the hypotheses  
**ExpRejections** The expected number of rejections  
**PowAt1st1** The power to reject at least one hypothesis

## References

Bretz, F., Maurer, W., Brannath, W. and Posch, M. (2009) A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine*, 28, 586–604

Bretz, F., Maurer, W. and Hommel, G. (2010) Test and power considerations for multiple endpoint analyses using sequentially rejective graphical procedures, to appear in *Statistics in Medicine*

## Examples

```
## reproduce example from Stat Med paper (Bretz et al. 2010, Table I)
## first only consider line 2 of Table I
## significance levels
weights <- c(1/2, 1/2, 0, 0)
## graph
G <- rbind(c(0, 0.5, 0.5, 0),
           c(0.5, 0, 0, 0.5),
           c(0, 1, 0, 0),
           c(1, 0, 0, 0))
## or equivalent:
G <- simpleSuccessiveII()@m
## alternative (mvn distribution)
corMat <- rbind(c(1, 0.5, 0.5, 0.5/2),
               c(0.5, 1, 0.5/2, 0.5),
               c(0.5, 0.5/2, 1, 0.5),
               c(0.5/2, 0.5, 0.5, 1))
theta <- c(3, 0, 0, 0)
calcPower(weights, alpha=0.025, G, theta, corMat, nSim = 100000)

## now reproduce all 14 simulation scenarios
## different graphs
weights1 <- c(rep(1/2, 12), 1, 1)
weights2 <- c(rep(1/2, 12), 0, 0)
eps <- 0.01
gam1 <- c(rep(0.5, 10), 1-eps, 0, 0, 0)
gam2 <- gam1
```

```

## different multivariate normal alternatives
rho <- c(rep(0.5, 8), 0, 0.99, rep(0.5,4))
th1 <- c(0, 3, 3, 3, 2, 1, rep(3, 7), 0)
th2 <- c(rep(0, 6), 3, 3, 3, 3, 0, 0, 0, 3)
th3 <- c(0, 0, 3, 3, 3, 3, 0, 2, 2, 2, 3, 3, 3, 3)
th4 <- c(0,0,0,3,3,3,0,2,2,2,0,0,0,0)

## function that calculates power values for one scenario
simfunc <- function(nSim, a1, a2, g1, g2, rh, t1, t2, t3, t4, Gr){
  a1 <- c(a1, a2, 0, 0)
  G <- rbind(c(0, g1, 1-g1, 0), c(g2, 0, 0, 1-g2), c(0, 1, 0, 0), c(1, 0, 0, 0))
  corMat <- rbind(c(1, 0.5, rh, rh/2), c(0.5,1,rh/2,rh), c(rh,rh/2,1,0.5), c(rh/2,rh,0.5,1))
  mean <- c(t1, t2, t3, t4)
  calcPower(a1, alpha=0.025, G, mean, corMat, nSim = nSim)
}

## calculate power for all 14 scenarios
outList <- list()
for(i in 1:14){
  outList[[i]] <- simfunc(10000, weights1[i], weights2[i],
                        gam1[i], gam2[i], rho[i], th1[i], th2[i], th3[i], th4[i])
}

## summarize data as in Stat Med paper Table I
atlst1 <- as.numeric(lapply(outList, function(x) x$PowAtlst1))
locpow <- do.call("rbind", lapply(outList, function(x) x$LocalPower))

round(cbind(atlst1, locpow), 5)

```

---

corMatWizard

*Graphical User Interface for the creation of correlation matrices*


---

## Description

Starts a graphical user interface for the correlation matrices.

## Usage

```

corMatWizard(n = dim(matrix)[1],
             matrix = paste("diag(", n, ")"),
             names = paste("H", 1:n, sep = ""), envir = globalenv())

```

## Arguments

<code>n</code>	Square root of the dimension of the $n \times n$ -Matrix.
<code>matrix</code>	Matrix of dimension $n \times n$ to start with.
<code>names</code>	Row and column names.
<code>envir</code>	Environment where the object <i>matrix</i> is located and/or it should be saved (default is the global environment).

**Value**

The function itself returns NULL. But with the dialog a symmetric matrix of dimension  $n \times n$  can be created or edited that will be available in R under the specified variable name after saving.

**Author(s)**

Kornelius Rohmeyer <rohmeier@small-projects.de>

**Examples**

```
## Not run:
corMatWizard(5)
C <- cor(matrix(rnorm(100),10), matrix(rnorm(100),10))
corMatWizard(matrix="C")

## End(Not run)
```

---

doInterim	<i>EXPERIMENTAL: Evaluate conditional errors at interim for a pre-planned graphical procedure</i>
-----------	---

---

**Description**

Computes partial conditional errors (PCE) for a pre-planned graphical procedure given information fractions and first stage z-scores. - Implementation of adaptive procedures is still in an early stage and may change in the near future

**Usage**

```
doInterim(graph, z1, v, alpha = 0.025)
```

**Arguments**

graph	A graph of class <a href="#">graphMCP</a> .
z1	A numeric vector giving first stage z-scores.
v	A numeric vector giving the proportions of pre-planned measurements collected up to the interim analysis. Will be recycled of length different than the number of elementary hypotheses.
alpha	A numeric specifying the maximal allowed type one error rate.

**Details**

For details see the given references.



**Value**

An object of class `gPADInterim`, more specifically a list with elements

`Aj` a matrix of PCEs for all elementary hypotheses in each intersection hypothesis  
`BJ` a numeric vector giving sum of PCEs per intersection hypothesis  
`preplanned` Pre planned test represented by an object of class `graphMCP`

**Author(s)**

Florian Klinglmueller <float@lefant.net>

**References**

Frank Bretz, Willi Maurer, Werner Brannath, Martin Posch: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)

Frank Bretz, Martin Posch, Ekkehard Glimm, Florian Klinglmueller, Willi Maurer, Kornelius Rohmeyer (2011): Graphical approaches for multiple comparison procedures using weighted Bonferroni, Simes or parametric tests. *Biometrical Journal* 53 (6), pages 894-913, Wiley. <http://onlinelibrary.wiley.com/doi/10.1002/bimj.201000239/full>

Posch M, Futschik A (2008): A Uniform Improvement of Bonferroni-Type Tests by Sequential Tests *JASA* 103/481, 299-308

Posch M, Maurer W, Bretz F (2010): Type I error rate control in adaptive designs for confirmatory clinical trials with treatment selection at interim *Pharm Stat* 10/2, 96-104

**See Also**

[graphMCP](#), [secondStageTest](#)

**Examples**

```
## Simple successive graph (Maurer et al. 2011)
## two treatments two hierarchically ordered endpoints
a <- .025
G <- simpleSuccessiveI()
## some z-scores:

p1=c(.1, .12, .21, .16)
z1 <- qnorm(1-p1)
p2=c(.04, 1, .14, 1)
z2 <- qnorm(1-p2)
v <- c(1/2, 1/3, 1/2, 1/3)

intA <- doInterim(G, z1, v)

## select only the first treatment
fTest <- secondStageTest(intA, c(1, 0, 1, 0))
```

---

entangledMCP-class      *Class entangledMCP*

---

### Description

A entangledMCP object describes ... TODO

### Slots

**subgraphs:** A list of graphs of class graphMCP.

**weights:** A numeric.

**graphAttr:** A list for graph attributes like color, etc.

### Methods

**print** signature(object = "entangledMCP"): A method for printing the data of the entangled graph to the R console.

**getMatrices** signature(object = "entangledMCP"): A method for getting the list of transition matrices of the entangled graph.

**getWeights** signature(object = "entangledMCP"): A method for getting the matrix of weights of the entangled graph.

**getRejected** signature(object = "entangledMCP"): A method for getting the information whether the hypotheses are marked in the graph as already rejected. If a second optional argument node is specified, only for these nodes the boolean vector will be returned.

**getXCoordinates** signature(object = "entangledMCP"): A method for getting the x coordinates of the graph. If a second optional argument node is specified, only for these nodes the x coordinates will be returned. If x coordinates are not yet set, NULL is returned.

**getYCoordinates** signature(object = "entangledMCP"): A method for getting the y coordinates of the graph. If a second optional argument node is specified, only for these nodes the y coordinates will be returned. If y coordinates are not yet set, NULL is returned.

### Author(s)

Kornelius Rohmeyer <rohmeier@small-projects.de>

### See Also

[graphMCP](#)

### Examples

```
g1 <- BonferroniHolm(2)
g2 <- BonferroniHolm(2)

graph <- new("entangledMCP", subgraphs=list(g1,g2), weights=c(0.5,0.5))
```

```
getMatrices(graph)
getWeights(graph)
```

---

exampleGraphs

*Functions that create different example graphs*

---

### **Description**

Functions that creates example graphs, e.g. graphs that represents a Bonferroni-Holm adjustment, parallel gatekeeping or special procedures from selected papers.

### **Usage**

```
BonferroniHolm(n)
BretzEtAl2011()
BauerEtAl2001()
BretzEtAl2009a()
BretzEtAl2009b()
BretzEtAl2009c()
HommelEtAl2007()
HommelEtAl2007Simple()
parallelGatekeeping()
improvedParallelGatekeeping()
fallback(weights)
fixedSequence(n)
simpleSuccessiveI()
simpleSuccessiveII()
truncatedHolm()
generalSuccessive(weights = c(1/2, 1/2))
```

```

HuqueAloshEtBhore2011()
HungEtWang2010()
MaurerEtAl1995()
cycleGraph(nodes, weights)
improvedFallbackI(weights = rep(1/3, 3))
improvedFallbackII(weights = rep(1/3, 3))
FerberTimeDose2011(times, doses, w = "\\nu")
Ferber2011()
Entangled1Maurer2012()
Entangled2Maurer2012()

```

### Arguments

n	Number of hypotheses.
nodes	Character vector of node names.
weights	Numeric vector of node weights.
times	Number of time points.
doses	Number of dose levels.
w	Further variable weight(s) in graph.

### Details

We are providing functions and not the resulting graphs directly because this way you have additional examples: You can look at the function body with `body` and see how the graph is built.

**list("BonferroniHolm")** Returns a graph that represents a Bonferroni-Holm adjustment. The result is a complete graph, where all nodes have the same weights and each edge weight is  $\frac{1}{n-1}$ .

**list("BretzEtAl2011")** Graph in figure 2 from Bretz et al. See references (Bretz et al. 2011).

**list("HommelEtAl2007")** Graph from Hommel et al. See references (Hommel et al. 2007).

**list("parallelGatekeeping")** Graph for parallel gatekeeping. See references (Dmitrienko et al. 2003).

**list("improvedParallelGatekeeping")** Graph for improved parallel gatekeeping. See references (Hommel et al. 2007).

**list("HungEtWang2010")** Graph from Hung et Wang. See references (Hung et Wang 2010).

**list("MaurerEtAl1995")** Graph from Maurer et al. See references (Maurer et al. 1995).

- list("cycleGraph")** Cycle graph. The weight `weights[i]` specifies the edge weight from node  $i$  to node  $i + 1$  for  $i = 1, \dots, n - 1$  and `weight[n]` from node  $n$  to node 1.
- list("improvedFallbackI")** Graph for the improved Fallback Procedure by Wiens & Dmitrienko. See references (Wiens et Dmitrienko 2005).
- list("improvedFallbackII")** Graph for the improved Fallback Procedure by Hommel & Bretz. See references (Hommel et Bretz 2008).
- list("Ferber2011")** Graph from Ferber et al. See references (Ferber et al. 2011).
- list("FerberTimeDose2011")** Graph from Ferber et al. See references (Ferber et al. 2011).
- list("Entangled1Maurer2012")** Entangled graph from Maurer et al. TODO: Add references as soon as they are available.

### Value

A graph of class `graphMCP` that represents a sequentially rejective multiple test procedure.

### Author(s)

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

### References

- Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* 6, 65-70.
- Dmitrienko, A., Offen, W., Westfall, P.H. (2003). Gatekeeping strategies for clinical trials that do not require all primary effects to be significant. *Statistics in Medicine*. 22, 2387-2400.
- Bretz, F., Maurer, W., Brannath, W., Posch, M.: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)
- Bretz, F., Maurer, W. and Hommel, G. (2011), Test and power considerations for multiple endpoint analyses using sequentially rejective graphical procedures. *Statistics in Medicine*, 30: 1489–1501.
- Hommel, G., Bretz, F. und Maurer, W. (2007). Powerful short-cuts for multiple testing procedures with special reference to gatekeeping strategies. *Statistics in Medicine*, 26(22), 4063-4073.
- Hommel, G., Bretz, F. (2008): Aesthetics and power considerations in multiple testing - a contradiction? *Biometrical Journal* 50:657-666.
- Hung H.M.J., Wang S.-J. (2010). Challenges to multiple testing in clinical trials. *Biometrical Journal* 52, 747-756.
- W. Maurer, L. Hothorn, W. Lehmacher: Multiple comparisons in drug clinical trials and preclinical assays: a-priori ordered hypotheses. In *Biometrie in der chemisch-pharmazeutischen Industrie*, Vollmar J (ed.). Fischer Verlag: Stuttgart, 1995; 3-18.
- Maurer, W., & Bretz, F. (2012). Memory and other properties of multiple test procedures generated by entangled graphs. *Statistics in medicine*.
- Wiens, B.L., Dmitrienko, A. (2005): The fallback procedure for evaluating a single family of hypotheses. *Journal of Biopharmaceutical Statistics* 15:929-942.
- Ferber, G. Staner, L. and Boeijinga, P. (2011): Structured multiplicity and confirmatory statistical analyses in pharmacodynamic studies using the quantitative electroencephalogram, *Journal of neuroscience methods*, Volume 201, Issue 1, Pages 204-212.

**Examples**

```

g <- BonferroniHolm(5)

## Not run:
# If Rgraphviz is installed, we can take a look at the graph:
library(Rgraphviz)
renderGraph(layoutGraph(g))

## End(Not run)

gMCP(g, pvalues=c(0.1, 0.2, 0.4, 0.4, 0.7))

```

---

extractPower

*Calculate power values*


---

**Description**

Calculates local power values, expected number of rejections, the power to reject at least one hypothesis and the power to reject all hypotheses.

**Usage**

```
extractPower(x, f = list())
```

**Arguments**

x	A matrix containing the rejected hypothesis, as produces by the graphTest function.
f	List of user defined power functions. If one is interested in the power to reject hypotheses 1 and 3 one could specify <code>function(x) {x[1] &amp;&amp; x[3]}</code>

**Value**

A list containing at least the following four elements and an element for each element in the parameter f.

- **LocPower** A numeric giving the local powers for the hypotheses
- **ExpNrRej** The expected number of rejections
- **PowAtlst1** The power to reject at least one hypothesis
- **RejectAll** The power to reject all hypotheses

---

generateBounds	<i>generateBounds</i>
----------------	-----------------------

---

### Description

compute rejection bounds for z-scores of each elementary hypotheses within each intersection hypotheses

### Usage

```
generateBounds(g, w, cr, al = 0.05,
  hint = generateWeights(g, w), exhaust = F)
```

### Arguments

g	graph defined as a matrix, each element defines how much of the local alpha reserved for the hypothesis corresponding to its row index is passed on to the hypothesis corresponding to its column index
w	vector of weights, defines how much of the overall alpha is initially reserved for each elementary hypothesis
cr	correlation matrix if p-values arise from one-sided tests with multivariate normal distributed test statistics for which the correlation is partially known. Unknown values can be set to NA. (See details for more information)
al	overall alpha level at which the family error is controlled
hint	if intersection hypotheses weights have already been computed (output of <a href="#">generateWeights</a> ) can be passed here otherwise will be computed during execution
exhaust	if FALSE (default) the parametric test is performed at the reduced level alpha of $\text{sum}(w) \cdot \text{alpha}$ . (See details)

### Details

It is assumed that under the global null hypothesis  $(\Phi^{-1}(1 - p_1), \dots, \Phi^{-1}(1 - p_m))$  follow a multivariate normal distribution with correlation matrix cr where  $\Phi^{-1}$  denotes the inverse of the standard normal distribution function.

For example, this is the case if  $p_1, \dots, p_m$  are the raw p-values from one-sided z-tests for each of the elementary hypotheses where the correlation between z-test statistics is generated by an overlap in the observations (e.g. comparison with a common control, group-sequential analyses etc.). An application of the transformation  $\Phi^{-1}(1 - p_i)$  to raw p-values from a two-sided test will not in general lead to a multivariate normal distribution. Partial knowledge of the correlation matrix is supported. The correlation matrix has to be passed as a numeric matrix with elements of the form:  $\text{correlation}[i, i] = 1$  for diagonal elements,  $\text{correlation}[i, j] = \rho_{ij}$ , where  $\rho_{ij}$  is the known value of the correlation between  $\Phi^{-1}(1 - p_i)$  and  $\Phi^{-1}(1 - p_j)$  or NA if the corresponding correlation is unknown. For example  $\text{correlation}[1,2]=0$  indicates that the first and second test statistic are uncorrelated, whereas  $\text{correlation}[2,3] = \text{NA}$  means that the true correlation between statistics two and three is unknown and may take values between -1 and 1. The correlation has to be specified for

complete blocks (ie.: if  $\text{cor}(i,j)$ , and  $\text{cor}(i,k)$  for  $i \neq j \neq k$  are specified then  $\text{cor}(j,k)$  has to be specified as well) otherwise the corresponding intersection null hypotheses tests are not uniquely defined and an error is returned.

The parametric tests in (Bretz et al. (2011)) are defined such that the tests of intersection null hypotheses always exhaust the full alpha level even if the sum of weights is strictly smaller than one. This has the consequence that certain test procedures that do not test each intersection null hypothesis at the full level alpha may not be implemented (e.g., a single step Dunnett test). If exhaust is set to FALSE (default) the parametric tests are performed at a reduced level alpha of  $\text{sum}(w) * \alpha$  and p-values adjusted accordingly such that test procedures with non-exhaustive weighting strategies may be implemented. If set to TRUE the tests are performed as defined in Equation (3) of (Bretz et al. (2011)).

### Value

Returns a matrix of rejection bounds. Each row corresponds to an intersection hypothesis. The intersection corresponding to each line is given by conversion of the line number into binary (eg. 13 is binary 1101 and corresponds to (H1,H2,H4))

### Author(s)

Florian Klinglmueller

### References

Bretz F, Maurer W, Brannath W, Posch M; (2008) - A graphical approach to sequentially rejective multiple testing procedures. - Stat Med - 28/4, 586-604

Frank Bretz, Martin Posch, Ekkehard Glimm, Florian Klinglmueller, Willi Maurer, Kornelius Rohmeyer (2011): Graphical approaches for multiple comparison procedures using weighted Bonferroni, Simes or parametric tests. Biometrical Journal 53 (6), pages 894-913, Wiley. <http://onlinelibrary.wiley.com/doi/10.1002/bimj.201000239/full>

### Examples

```
## Define some graph as matrix
g <- matrix(c(0,0,1,0,
              0,0,0,1,
              0,1,0,0,
              1,0,0,0), nrow = 4,byrow=TRUE)

## Choose weights
w <- c(.5,.5,0,0)
## Some correlation (upper and lower first diagonal 1/2)
c <- diag(4)
c[1:2,3:4] <- NA
c[3:4,1:2] <- NA
c[1,2] <- 1/2
c[2,1] <- 1/2
c[3,4] <- 1/2
c[4,3] <- 1/2

## Boundaries for correlated test statistics at alpha level .05:
```



```
generateBounds(g,w,c,.05)
```

---

```
generatePvals          generatePvals
```

---

### Description

compute adjusted p-values either for the closed test defined by the graph or for each elementary hypotheses within each intersection hypotheses

### Usage

```
generatePvals(g, w, cr, p, adjusted = TRUE,
              hint = generateWeights(g, w), exhaust = FALSE)
```

### Arguments

g	graph defined as a matrix, each element defines how much of the local alpha reserved for the hypothesis corresponding to its row index is passed on to the hypothesis corresponding to its column index
w	vector of weights, defines how much of the overall alpha is initially reserved for each elementary hypothesis
cr	correlation matrix if p-values arise from one-sided tests with multivariate normal distributed test statistics for which the correlation is partially known. Unknown values can be set to NA. (See details for more information)
p	vector of observed unadjusted p-values, that belong to test-statistics with a joint multivariate normal null distribution with (partially) known correlation matrix cr
adjusted	logical, if TRUE (default) adjusted p-values for the closed test are returned, else a matrix of p-values adjusted only for each intersection hypothesis is returned
hint	if intersection hypotheses weights have already been computed (output of <a href="#">generateWeights</a> ) can be passed here otherwise will be computed during execution
exhaust	if FALSE (default) the p-values are additionally adjusted for the case that non-exhaustive weights are specified. (See details)

### Details

It is assumed that under the global null hypothesis  $(\Phi^{-1}(1 - p_1), \dots, \Phi^{-1}(1 - p_m))$  follow a multivariate normal distribution with correlation matrix cr where  $\Phi^{-1}$  denotes the inverse of the standard normal distribution function.

For example, this is the case if  $p_1, \dots, p_m$  are the raw p-values from one-sided z-tests for each of the elementary hypotheses where the correlation between z-test statistics is generated by an overlap in the observations (e.g. comparison with a common control, group-sequential analyses etc.). An application of the transformation  $\Phi^{-1}(1 - p_i)$  to raw p-values from a two-sided test will not in general lead to a multivariate normal distribution. Partial knowledge of the correlation

matrix is supported. The correlation matrix has to be passed as a numeric matrix with elements of the form:  $cr[i, i] = 1$  for diagonal elements,  $cr[i, j] = \rho_{ij}$ , where  $\rho_{ij}$  is the known value of the correlation between  $\Phi^{-1}(1 - p_i)$  and  $\Phi^{-1}(1 - p_j)$  or NA if the corresponding correlation is unknown. For example  $cr[1,2]=0$  indicates that the first and second test statistic are uncorrelated, whereas  $cr[2,3] = NA$  means that the true correlation between statistics two and three is unknown and may take values between -1 and 1. The correlation has to be specified for complete blocks (ie.: if  $cor(i,j)$ , and  $cor(i,k)$  for  $i \neq j \neq k$  are specified then  $cor(j,k)$  has to be specified as well) otherwise the corresponding intersection null hypotheses tests are not uniquely defined and an error is returned.

The parametric tests in (Bretz et al. (2011)) are defined such that the tests of intersection null hypotheses always exhaust the full alpha level even if the sum of weights is strictly smaller than one. This has the consequence that certain test procedures that do not test each intersection null hypothesis at the full level alpha may not be implemented (e.g., a single step Dunnett test). If exhaust is set to FALSE (default) the parametric tests are performed at a reduced level alpha of  $sum(w) * alpha$  and p-values adjusted accordingly such that test procedures with non-exhaustive weighting strategies may be implemented. If set to TRUE the tests are performed as defined in Equation (3) of (Bretz et al. (2011)).

### Value

If adjusted is set to true returns a vector of adjusted p-values. Any elementary null hypothesis is rejected if its corresponding adjusted p-value is below the predetermined alpha level. For adjusted set to false a matrix with p-values adjusted only within each intersection hypotheses is returned. The intersection corresponding to each line is given by conversion of the line number into binary (eg. 13 is binary 1101 and corresponds to (H1,H2,H4)). If any adjusted p-value within a given line falls below alpha, then the corresponding intersection hypotheses can be rejected.

### Author(s)

Florian Klingmueller

### References

Bretz F, Maurer W, Brannath W, Posch M; (2008) - A graphical approach to sequentially rejective multiple testing procedures. - Stat Med - 28/4, 586-604  
 Bretz F, Posch M, Glimm E, Klingmueller F, Maurer W, Rohmeyer K; (2011) - Graphical approaches for multiple endpoint problems using weighted Bonferroni, Simes or parametric tests - to appear

### Examples

```
## Define some graph as matrix
g <- matrix(c(0,0,1,0, 0,0,0,1, 0,1,0,0, 1,0,0,0), nrow = 4,byrow=TRUE)
## Choose weights
w <- c(.5,.5,0,0)
## Some correlation (upper and lower first diagonal 1/2)
c <- diag(4)
c[1:2,3:4] <- NA
c[3:4,1:2] <- NA
c[1,2] <- 1/2
c[2,1] <- 1/2
c[3,4] <- 1/2
```

```

c[4,3] <- 1/2
## p-values as Section 3 of Bretz et al. (2011),
p <- c(0.0121,0.0337,0.0084,0.0160)

## Boundaries for correlated test statistics at alpha level .05:
generatePvals(g,w,c,p)

```

---

generateTest

*generateTest*


---

## Description

generates a test function for the multiple comparison procedure with correlated test statistics defined by a graph

## Usage

```
generateTest(g, w, cr, al)
```

## Arguments

g	graph defined as a matrix, each element defines how much of the local alpha reserved for the hypothesis corresponding to its row index is passed on to the hypothesis corresponding to its column index
w	vector of weights, defines how much of the overall alpha is initially reserved for each elementary hypothesis
cr	correlation matrix if p-values arise from one-sided tests with multivariate normal distributed test statistics for which the correlation is partially known. Unknown values can be set to NA. (See details for more information)
al	overall alpha level at which the family error is controlled

## Details

It is assumed that under the global null hypothesis  $(\Phi^{-1}(1 - p_1), \dots, \Phi^{-1}(1 - p_m))$  follow a multivariate normal distribution with correlation matrix  $cr$  where  $\Phi^{-1}$  denotes the inverse of the standard normal distribution function.

For example, this is the case if  $p_1, \dots, p_m$  are the raw p-values from one-sided z-tests for each of the elementary hypotheses where the correlation between z-test statistics is generated by an overlap in the observations (e.g. comparison with a common control, group-sequential analyses etc.). An application of the transformation  $\Phi^{-1}(1 - p_i)$  to raw p-values from a two-sided test will not in general lead to a multivariate normal distribution. Partial knowledge of the correlation matrix is supported. The correlation matrix has to be passed as a numeric matrix with elements of the form:  $cr[i, i] = 1$  for diagonal elements,  $cr[i, j] = \rho_{ij}$ , where  $\rho_{ij}$  is the known value of the correlation between  $\Phi^{-1}(1 - p_i)$  and  $\Phi^{-1}(1 - p_j)$  or NA if the corresponding correlation is unknown. For example  $cr[1,2]=0$  indicates that the first and second test statistic are uncorrelated, whereas  $cr[2,3] = NA$  means that the true correlation between statistics two and three is unknown and may take values between -1 and 1. The correlation has to be specified for complete blocks (ie.:

if  $\text{cor}(i,j)$ , and  $\text{cor}(i,k)$  for  $i \neq j \neq k$  are specified then  $\text{cor}(j,k)$  has to be specified as well) otherwise the corresponding intersection null hypotheses tests are not uniquely defined and an error is returned.

The parametric tests in (Bretz et al. (2011)) are defined such that the tests of intersection null hypotheses always exhaust the full alpha level even if the sum of weights is strictly smaller than one. This has the consequence that certain test procedures that do not test each intersection null hypothesis at the full level alpha may not be implemented (e.g., a single step Dunnett test). If exhaust is set to FALSE (default) the parametric tests are performed at a reduced level alpha of  $\text{sum}(w) * \alpha$  and p-values adjusted accordingly such that test procedures with non-exhaustive weighting strategies may be implemented. If set to TRUE the tests are performed as defined in Equation (3) of (Bretz et al. (2011)).

### Value

Returns a function that will take a vector of z-scores to which the test will be applied. This function in turn will return a boolean vector with elements false if the particular elementary hypothesis can not be rejected and true otherwise.

### Author(s)

Florian Klinglmueller

### References

Bretz F, Maurer W, Brannath W, Posch M; (2008) - A graphical approach to sequentially rejective multiple testing procedures. - Stat Med - 28/4, 586-604  
 Bretz F, Posch M, Glimm E, Klinglmueller F, Maurer W, Rohmeyer K; (2011) - Graphical approaches for multiple endpoint problems using weighted Bonferroni, Simes or parametric tests - to appear

### Examples

```
## Define some graph as matrix
g <- matrix(c(0,0,1,0,
              0,0,0,1,
              0,1,0,0,
              1,0,0,0), nrow = 4,byrow=TRUE)

## Choose weights
w <- c(.5,.5,0,0)

## Some correlation (upper and lower first diagonal 1/2)
c <- diag(4)
c[1:2,3:4] <- NA
c[3:4,1:2] <- NA
c[1,2] <- 1/2
c[2,1] <- 1/2
c[3,4] <- 1/2
c[4,3] <- 1/2

## Test function for further use:
myTest <- generateTest(g,w,c,.05)
myTest(c(3,2,1,2))
```

---

generateWeights      *generateWeights*

---

**Description**

compute Weights for each intersection Hypotheses in the closure of a graph based multiple testing procedure

**Usage**

```
generateWeights(g, w)
```

**Arguments**

**g** graph defined as a matrix, each element defines how much of the local alpha reserved for the hypothesis corresponding to its row index is passed on to the hypothesis corresponding to its column index

**w** vector of weights, defines how much of the overall alpha is initially reserved for each elementary hypothesis

**Value**

Returns matrix with each row corresponding to one intersection hypothesis in the closure of the multiple testing problem. The first half of elements indicate whether an elementary hypothesis is in the intersection (1) or not (0). The second half of each row gives the weights allocated to each elementary hypotheses in the intersection.

**Author(s)**

Florian Klinglmueller

**References**

Bretz F, Maurer W, Brannath W, Posch M; (2008) - A graphical approach to sequentially rejective multiple testing procedures. - Stat Med - 28/4, 586-604  
Bretz F, Posch M, Glimm E, Klinglmueller F, Maurer W, Rohmeyer K; (2011) - Graphical approaches for multiple endpoint problems using weighted Bonferroni, Simes or parametric tests - to appear

**Examples**

```
g <- matrix(c(0,0,1,0,
             0,0,0,1,
             0,1,0,0,
             1,0,0,0), nrow = 4,byrow=TRUE)
## Choose weights
w <- c(.5,.5,0,0)
## Weights of conventional gMCP test:
generateWeights(g,w)
```

---

`getJavaInfo`*Get Memory and Runtime Info from JVM*

---

**Description**

Get Memory and Runtime Info from JVM

**Usage**

```
getJavaInfo()
```

**Value**

character vector of length 1 containing the memory and runtime info.

**Author(s)**

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

**Examples**

```
cat(getJavaInfo())
```

---

`gMCP`*Graph based Multiple Comparison Procedures*

---

**Description**

Performs a graph based multiple test procedure for a given graph and unadjusted p-values.

**Usage**

```
gMCP(graph, pvalues, test, correlation, alpha = 0.05,  
      approxEps = TRUE, eps = 10-3, ..., useC = FALSE,  
      verbose = FALSE, keepWeights = TRUE, adjPValues = TRUE)
```

**Arguments**

<code>graph</code>	A graph of class <code>graphMCP</code> .
<code>pvalues</code>	A numeric vector specifying the p-values for the graph based MCP. Note the assumptions in the details section for the parametric tests, when a correlation is specified.

test	By default the Bonferroni-based test procedures are used if no correlation is specified and the algorithm from Bretz et al. 2011 if a correlation is specified. If test is set to "Simes" the weighted Simes test will be performed for each subset of hypotheses. If test equals "Bretz2011" the weighted parametric test described in Bretz et al. 2011 is performed. If test equals "simple-parametric" for each subset a weighted parametric test is performed at the reduced level alpha of $\sum(w) \cdot \alpha$ , where $\sum(w)$ is the sum of all node weights in this subset. Otherwise the procedure is the same as described in Bretz et al. 2011.
correlation	Optional correlation matrix. If the weighted Simes test is performed, it is checked whether type I error rate can be ensured and a warning is given if this is not the case. For parametric tests the p-values must arise from one-sided tests with multivariate normal distributed test statistics for which the correlation is (partially) known. In that case a weighted parametric closed test is performed (also see <a href="#">generatePvals</a> ). Unknown values can be set to NA. (See details for more information)
alpha	A numeric specifying the maximal allowed type one error rate.
approxEps	A boolean specifying whether epsilon values should be substituted with the value given in the parameter eps.
eps	A numeric scalar specifying a value for epsilon edges.
...	Test specific arguments can be given here.
useC	Logical scalar. If TRUE neither adjusted p-values nor intermediate graphs are returned, but the calculation is sped up by using code written in C. THIS CODE IS NOT FOR PRODUCTIVE USE YET! If approxEps is FALSE and the graph contains epsilon edges, a warning is thrown and useC will be ignored.
verbose	Logical scalar. If TRUE verbose output is generated during sequentially rejection steps.
keepWeights	Logical scalar. If FALSE the weight of a node without outgoing edges is set to 0 if it is removed. Otherwise it keeps its weight.
adjPValues	Logical scalar. If FALSE no adjusted p-values will be calculated. Especially for the weighted Simes test this will result in significantly less calculations in most cases.

## Details

For the Bonferroni procedure the p-values can arise from any statistical test, but if you improve the test by specifying a correlation matrix, the following assumptions apply:

It is assumed that under the global null hypothesis  $(\Phi^{-1}(1 - p_1), \dots, \Phi^{-1}(1 - p_m))$  follow a multivariate normal distribution with correlation matrix `correlation` where  $\Phi^{-1}$  denotes the inverse of the standard normal distribution function.

For example, this is the case if  $p_1, \dots, p_m$  are the raw p-values from one-sided z-tests for each of the elementary hypotheses where the correlation between z-test statistics is generated by an overlap in the observations (e.g. comparison with a common control, group-sequential analyses etc.). An application of the transformation  $\Phi^{-1}(1 - p_i)$  to raw p-values from a two-sided test will not in general lead to a multivariate normal distribution. Partial knowledge of the correlation matrix is supported. The correlation matrix has to be passed as a numeric matrix with elements of the form:

$correlation[i, i] = 1$  for diagonal elements,  $correlation[i, j] = \rho_{ij}$ , where  $\rho_{ij}$  is the known value of the correlation between  $\Phi^{-1}(1 - p_i)$  and  $\Phi^{-1}(1 - p_j)$  or NA if the corresponding correlation is unknown. For example  $correlation[1,2]=0$  indicates that the first and second test statistic are uncorrelated, whereas  $correlation[2,3] = NA$  means that the true correlation between statistics two and three is unknown and may take values between -1 and 1. The correlation has to be specified for complete blocks (ie.: if  $cor(i,j)$ , and  $cor(i,j')$  for  $i!=j!=j'$  are specified then  $cor(j,j')$  has to be specified as well) otherwise the corresponding intersection null hypotheses tests are not uniquely defined and an error is returned.

For further details see the given references.

### Value

An object of class `gMCPResult`, more specifically a list with elements

<code>graphs</code>	list of graphs
<code>pvalues</code>	p-values
<code>rejected</code>	logical whether hypotheses could be rejected
<code>adjPValues</code>	adjusted p-values

### Author(s)

Kornelius Rohmeyer <rohmeier@small-projects.de>

### References

Frank Bretz, Willi Maurer, Werner Brannath, Martin Posch: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)

Bretz F., Posch M., Glimm E., Klinglmueller F., Maurer W., Rohmeyer K. (2011): Graphical approaches for multiple endpoint problems using weighted Bonferroni, Simes or parametric tests - to appear.

Strassburger K., Bretz F.: Compatible simultaneous lower confidence bounds for the Holm procedure and other Bonferroni based closed tests. *Statistics in Medicine* 2008; 27:4914-4927.

Hommel G., Bretz F., Maurer W.: Powerful short-cuts for multiple testing procedures with special reference to gatekeeping strategies. *Statistics in Medicine* 2007; 26:4063-4073.

Guilbaud O.: Simultaneous confidence regions corresponding to Holm's stepdown procedure and other closed-testing procedures. *Biometrical Journal* 2008; 50:678-692.

### See Also

[graphMCP](#) [graphNEL](#)

### Examples

```
g <- BonferroniHolm(5)
gMCP(g, pvalues=c(0.01, 0.02, 0.04, 0.04, 0.7))
```



---

`gMCPReport`*Automatic Generation of gMCP Reports*

---

**Description**

Creates a LaTeX file with a gMCP Report.

**Usage**

```
gMCPReport(object, file = "", ...)
```

**Arguments**

<code>object</code>	A graph of class <code>graphMCP</code> or an object of class <code>gMCPResult</code> .
<code>file</code>	A connection, or a character string naming the file to print to. If "" (the default), the report is printed to the standard output connection (the console unless redirected by <code>sink</code> ). If it is " <code> cmd</code> ", the output is piped to the command given by <code>cmd</code> , by opening a pipe connection [taken from the manual page of <code>cat</code> , which is called in this function].
<code>...</code>	Arguments to be passed to method <code>graph2latex</code> like <code>package</code> and <code>scale</code> .

**Details**

This function uses `cat` and `graph2latex`.

**Value**

None (invisible NULL).

**Author(s)**

Kornelius Rohmeyer <rohmeyer@small-projects.de>

**References**

The TikZ and PGF Packages Manual for version 2.00, Till Tantau, <http://www.ctan.org/tex-archive/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf>

**See Also**

[cat graph2latex](#)

**Examples**

```
g <- BretzEtAl2011()
result <- gMCP(g, pvalues=c(0.1, 0.008, 0.005, 0.15, 0.04, 0.006))
gMCPReport(result)
```

---

gMCPResult-class      *Class gMCPResult*

---

### Description

A gMCPResult object describes an evaluated sequentially rejective multiple test procedure.

### Slots

graphs: Object of class list.

alpha: A numeric specifying the maximal type I error rate.

pvalues: The numeric vector of pvalues.

rejected: The logical vector of rejected null hypotheses.

adjPValues: The numeric vector of adjusted pvalues.

### Author(s)

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

### See Also

[gMCP](#)

---

gPADInterim-class      *Class gPADInterim*

---

### Description

A gPADInterim object describes an object holding interim information for an adaptive procedure that is based on a preplanned graphical procedure.

### Slots

Aj: Object of class numeric. Giving partial conditional errors (PCEs) for all elementary hypotheses in each intersection hypothesis

BJ: A numeric specifying the sum of PCEs per intersection hypothesis.

z1: The numeric vector of first stage z-scores.

v: A numeric specifying the proportion of measurements collected up to interim

preplanned: Object of class [graphMCP](#) specifying the preplanned graphical procedure.

alpha: A numeric giving the alpha level of the pre-planned test

**Author(s)**

Florian Klinglmueller <float@elefant.net>

**See Also**

[gMCP](#), [doInterim](#), [secondStageTest](#)

---

graph2latex

*Graph2LaTeX*

---

**Description**

Creates LaTeX code that represents the given graph.

**Usage**

```
graph2latex(graph, package = "TikZ", scale = 1,
  alpha = 0.05, pvalues, fontsize, nodeTikZ,
  labelTikZ = "near start,above,fill=blue!20",
  tikzEnv = TRUE, offset = c(0, 0),
  fill = list(reject = "red!80", retain = "green!80"),
  nodeR = 25, scaleText = TRUE)
```

**Arguments**

graph	A graph of class <a href="#">graphMCP</a> .
package	A character string specifying the LaTeX package that should be used. Up to now only TikZ is available.
scale	A numeric scalar specifying a possible scaling of the graph. Note that this does not effect the fontsize of the graph. (Coordinates are interpreted in big points - 72 bp = 1 inch).
alpha	An optional numeric argument to specify the type I error rate.
pvalues	If the optional numeric argument pvalues is given, nodes that can be rejected, will be marked.
fontsize	An optional character vector specifying the fontsize for the graph, must be one of "tiny", "scriptsize", "footnotesize", "small", "normalsize", "large", "Large", "LARGE", "huge" or "Huge".
nodeTikZ	A character string with additional arguments for the TikZ node command like for example nodeTikZ="minimum size=2cm".
labelTikZ	A character string with arguments for the TikZ node command within an edge.
tikzEnv	Logical whether the LaTeX code should be wrapped in a TikZ environment.
offset	A numeric of length 2 specifying the x and y offset in the TikZ environment.
fill	A list containing 2 elements reject and retain specifying node fill colour of rejected and retained (or not yet rejected) nodes.

nodeR	Radius of nodes (pixel in Java, bp in LaTeX).
scaleText	Only relevant if scale is unequal 1. If scaleText is TRUE (the default) a scalebox environment is used. If it is FALSE the optional parameter scale from the tikzpicture environment is used and font size will not change. Note that while you easily can change the scale in the scalebox environment, it is more problematic to adjust the scale in the tikzpicture environment afterwards in the LaTeX document, since for curved edges the parameters are calculated for a certain relative node size which changes if the graph is scaled but the text size stays the same.

**Details**

For details see the given references.

**Value**

A character string that contains LaTeX code representing the given graph.

**Author(s)**

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

**References**

The TikZ and PGF Packages Manual for version 2.00, Till Tantau, <http://www.ctan.org/tex-archive/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf>

**See Also**

[graphMCP](#), [gMCPReport](#)

**Examples**

```
g <- BonferroniHolm(5)
graph2latex(g)
```

---

graphAnalysis

*Analysis of a gMCP-Graph*

---

**Description**

Creates LaTeX code that represents the given graph.

**Usage**

```
graphAnalysis(graph, file = "")
```

**Arguments**

graph	A graph of class <code>graphMCP</code> .
file	A connection, or a character string naming the file to print to. If "" (the default), the analysis is printed to the standard output connection (the console unless redirected by sink). If it is " cmd", the output is piped to the command given by cmd, by opening a pipe connection [taken from the manual page of cat, which is called in this function].

**Details**

In the moment it is only tested whether each node is accessible from each other node. Further analysis will be added in future versions.

**Value**

A character string that contains the printed analysis.

**Author(s)**

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

**See Also**

`graphMCP`

**Examples**

```
g <- BonferroniHolm(5)
graphAnalysis(g)
```

---

graphGUI	<i>Graphical User Interface for graphical described multiple comparison procedures</i>
----------	--

---

**Description**

Starts a graphical user interface for the creation/modification of directed weighted graphs and applying graphical described multiple comparison procedures.

**Usage**

```
graphGUI(graph = "createdGraph", pvalues = numeric(0),
  grid = 0, debug = FALSE, experimentalFeatures = FALSE,
  envir = globalenv())
```

## Arguments

graph	Either a variable name for the graph, given as a character string. (If it is not a syntactically valid name, <code>make.names</code> is called to change it to a valid one.) Or an object of class <code>graphMCP</code> . If the object is modified (even just by updating the class definition or arranging the nodes) it will be saved in the specified environment (default is the global environment).
pvalues	Numeric value that optionally specifies the p-values.
grid	Positive integer that sets the grid size for easier placement of nodes. (Therefore grid size 1 allows unrestricted placement and disables the grid.) The default <code>grid=0</code> uses the last used grid value or if the GUI is started the first time a value of 50.
debug	Logical. If TRUE debug output is printed to the R console.
experimentalFeatures	Logical. If TRUE some unfinished / insufficiently tested experimental features are available in the GUI.
envir	Environment where the object <code>graph</code> is located and/or it should be saved (default is the global environment).

## Details

See the vignette of this package for further details, since describing a GUI interface is better done with a lot of nice pictures.

## Value

The function itself returns NULL. But with the GUI a graph can be created or edited that will be available in R under the specified variable name after saving in the specified environment.

## Author(s)

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

## Examples

```
## Not run:
graphGUI()
pvalues <- c(9.7, 1.5, 0.5, 0.6, 0.4, 0.8, 4)/100
graphGUI(HommelEtAl2007(), pvalues=pvalues)

x <- new.env()
assign("graph", BonferroniHolm(3), envir=x)
graphGUI("graph", envir=x)
## End(Not run)
```

---

graphMCP-class            *Class graphMCP*

---

## Description

A graphMCP object describes a sequentially rejective multiple test procedure.

## Slots

**m**: A transition matrix. Can be either numerical or character depending whether the matrix contains variables or not. Row and column names will be the names of the nodes.

**weights**: A numeric.

**edgeAttr**: A list for edge attributes.

**nodeAttr**: A list for node attributes.

## Methods

**getMatrix** signature(object = "graphMCP"): A method for getting the transition matrix of the graph.

**getWeights** signature(object = "graphMCP"): A method for getting the weights. If a third optional argument node is specified, only for these nodes the weight will be returned.

**setWeights** signature(object = "graphMCP"): A method for setting the weights. If a third optional argument node is specified, only for these nodes the weight will be set.

**getRejected** signature(object = "graphMCP"): A method for getting the information whether the hypotheses are marked in the graph as already rejected. If a second optional argument node is specified, only for these nodes the boolean vector will be returned.

**getXCoordinates** signature(object = "graphMCP"): A method for getting the x coordinates of the graph. If a second optional argument node is specified, only for these nodes the x coordinates will be returned. If x coordinates are not set yet NULL is returned.

**getYCoordinates** signature(object = "graphMCP"): A method for getting the y coordinates of the graph. If a second optional argument node is specified, only for these nodes the y coordinates will be returned. If y coordinates are not set yet NULL is returned.

**setEdge** signature(from="character", to="character", graph="graphNEL", weights="numeric"): A method for adding new edges with the given weights.

**setEdge** signature(from="character", to="character", graph="graphMCP", weights="character"): A method for adding new edges with the given weights.

## Author(s)

Kornelius Rohmeyer <rohmeier@small-projects.de>

**Examples**

```

m <- rbind(H11=c(0, 0.5, 0, 0.5, 0, 0 ),
H21=c(1/3, 0, 1/3, 0, 1/3, 0 ),
H31=c(0, 0.5, 0, 0, 0, 0.5),
H12=c(0, 1, 0, 0, 0, 0 ),
H22=c(0.5, 0, 0.5, 0, 0, 0 ),
H32=c(0, 1, 0, 0, 0, 0 ))

weights <- c(1/3, 1/3, 1/3, 0, 0, 0)

# Graph creation
graph <- new("graphMCP", m=m, weights=weights)

# Visualization settings
nodeX <- rep(c(100, 300, 500), 2)
nodeY <- rep(c(100, 300), each=3)
graph@nodeAttr$X <- nodeX
graph@nodeAttr$Y <- nodeY

getWeights(graph)

getRejected(graph)

pvalues <- c(0.1, 0.008, 0.005, 0.15, 0.04, 0.006)
result <- gMCP(graph, pvalues)

getWeights(result@graphs[[4]])
getRejected(result@graphs[[4]])

```

---

graphTest

*Multiple testing using graphs*


---

**Description**

Implements the graphical test procedure described in Bretz et al. (2009). Note that the gMCP function in the gMCP package performs the same task.

**Usage**

```

graphTest(pvalues, weights = NULL, alpha = 0.05,
  G = NULL, cr = NULL, graph = NULL, verbose = FALSE,
  test)

```

**Arguments**

**pvalues** Either a vector or a matrix containing the local p-values for the hypotheses in the rows.



weights	Initial weight levels for the test procedure, in case of multiple graphs this needs to be a matrix.
alpha	Overall alpha level of the procedure. For entangled graphs alpha should be a numeric vector of length equal to the number of graphs, each element specifying the partial alpha for the respective graph. The overall alpha level equals <code>sum(alpha)</code> .
G	For simple graphs G should be a numeric matrix determining the graph underlying the test procedure. Note that the diagonal need to contain only 0s, while the rows need to sum to 1. For entangled graphs it needs to be a list containing the different graph matrices as elements.
graph	As an alternative to the specification via <code>weights</code> and <code>G</code> one can also hand over a <code>graphMCP</code> object to the code. <code>graphMCP</code> objects can be created for example with the <code>graphGUI</code> function.
cr	Correlation matrix that should be used for the parametric test. If <code>cr==NULL</code> the Bonferroni based test procedure is used.
verbose	If <code>verbose</code> is <code>TRUE</code> , additional information about the graphical rejection procedure is displayed.
test	In the parametric case there is more than one way to handle subgraphs with less than the full alpha. If the parameter <code>test</code> is missing, the tests are performed as described by Bretz et al. (2011), i.e. tests of intersection null hypotheses always exhaust the full alpha level even if the sum of weights is strictly smaller than one. If <code>test="simple-parametric"</code> the tests are performed as defined in Equation (3) of Bretz et al. (2011).

### Value

A vector or a matrix containing the test results for the hypotheses under consideration. Significant tests are denoted by a 1, non-significant results by a 0.

### References

Bretz, F., Maurer, W., Brannath, W. and Posch, M. (2009) A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine*, 28, 586–604

Bretz, F., Maurer, W. and Hommel, G. (2010) Test and power considerations for multiple endpoint analyses using sequentially rejective graphical procedures, to appear in *Statistics in Medicine*

### Examples

```
#### example from Bretz et al. (2010)
weights <- c(1/3, 1/3, 1/3, 0, 0, 0)
graph <- rbind(c(0,      0.5, 0,      0.5, 0,      0),
              c(1/3,    0,    1/3,    0,    1/3,    0),
              c(0,      0.5, 0,      0,      0,      0.5),
              c(0,      1,    0,      0,      0,      0),
              c(0.5,    0,    0.5,    0,      0,      0),
              c(0,      1,    0,      0,      0,      0))
pvals <- c(0.1, 0.008, 0.005, 0.15, 0.04, 0.006)
graphTest(pvals, weights, alpha=0.025, graph)
```

```

## observe graphical procedure in detail
graphTest(pvals, weights, alpha=0.025, graph, verbose = TRUE)

## now use many p-values (useful for power simulations)
pvals <- matrix(rbeta(6e4, 1, 30), ncol = 6)
out <- graphTest(pvals, weights, alpha=0.025, graph)
head(out)

## example using multiple graphs (instead of 1)
G1 <- rbind(c(0,0.5,0.5,0,0), c(0,0,1,0,0),
            c(0, 0, 0, 1-0.01, 0.01), c(0, 1, 0, 0, 0),
            c(0, 0, 0, 0, 0))
G2 <- rbind(c(0,0,1,0,0), c(0.5,0,0.5,0,0),
            c(0, 0, 0, 0.01, 1-0.01), c(0, 0, 0, 0, 0),
            c(1, 0, 0, 0, 0))
weights <- rbind(c(1, 0, 0, 0, 0), c(0, 1, 0, 0, 0))
pvals <- c(0.012, 0.025, 0.005, 0.0015, 0.0045)
out <- graphTest(pvals, weights, alpha=c(0.0125, 0.0125), G=list(G1, G2), verbose = TRUE)

## now again with many p-values
pvals <- matrix(rbeta(5e4, 1, 30), ncol = 5)
out <- graphTest(pvals, weights, alpha=c(0.0125, 0.0125), G=list(G1, G2))
head(out)

```

---

hydroquinone

*Hydroquinone Mutagenicity Assay*


---

## Description

This data set gives the number of micronuclei per animal and 2000 scored cells for six different groups of differently treated male mice: The negative control (C-), four doses (30, 50, 75, 100 mg hydroquinone / kg) of hydroquinone and an active control (C+) (with 25 mg/kg cyclophosphamide).

## Usage

```
data(hydroquinone)
```

## Format

A data frame with 31 observations on the following 2 variables:

**group** A factor with levels "C-", "30 mg/kg", "50 mg/kg", "75 mg/kg", "100 mg/kg" and "C+" specifying the groups.

**micronuclei** A numeric vector, giving the counts of micronuclei per animal and 2000 scored cells after 24h.

**Source**

Adler, I.-D. and Kliesch, U. (1990): *Comparison of single and multiple treatment regimens in the mouse bone marrow micronucleus assay for hydroquinone and cyclophosphamide*. Mutation Research 234, 115-123.

**References**

Bauer, P., Roehmel, J., Maurer, W., and Hothorn, L. (1998): *Testing strategies in multi-dose experiments including active control*. Statistics in Medicine 17, 2133-2146.

**Examples**

```
data(hydroquinone)
boxplot(micronuclei~group, data=hydroquinone)
```

---

 joinGraphs

*Joins two graphMCP objects*


---

**Description**

Creates a new graphMCP object by joining two given graphMCP objects.

**Usage**

```
joinGraphs(graph1, graph2, xOffset = 0, yOffset = 200)
```

**Arguments**

graph1	A graph of class <a href="#">graphMCP</a> .
graph2	A graph of class <a href="#">graphMCP</a> .
xOffset	A numeric specifying an offset (on the x-axis) for placing the nodes and edge labels of the second graph.
yOffset	A numeric specifying an offset (on the y-axis) for placing the nodes and edge labels of the second graph.

**Details**

If graph1 and graph2 have duplicates in the node names, the nodes of the second graph will be renamed.

If and only if the sum of the weights of graph1 and graph2 exceeds 1, the weights are scaled so that the sum equals 1.

A description attribute of either graph will be discarded.

**Value**

A graphMCP object that represents a graph that consists of the two given graphs.

**Author(s)**

Kornelius Rohmeyer <rohmeier@small-projects.de>

**See Also**

[graphMCP](#)

**Examples**

```
g1 <- BonferroniHolm(2)
g2 <- BonferroniHolm(3)

joinGraphs(g1, g2)
```

---

matrix2graph

*Matrix2Graph and Graph2Matrix*

---

**Description**

Creates a graph of class [graphMCP](#) from a given transition matrix or vice versa.

**Usage**

```
matrix2graph(m, weights = rep(1/dim(m)[1], dim(m)[1]))

graph2matrix(graph)
```

**Arguments**

m	A transition matrix.
weights	A numeric for the initial weights.
graph	A graph of class <a href="#">graphMCP</a> .

**Details**

The hypotheses names are the row names or if these are NULL, the column names or if these are also NULL of type H1, H2, H3, ...

If the diagonal of the matrix is unequal zero, the values are ignored and a warning is given.

**Value**

A graph of class [graphMCP](#) with the given transition matrix for `matrix2graph`. The transition matrix of a [graphMCP](#) graph for `graph2matrix`.

**Author(s)**

Kornelius Rohmeyer <rohmeier@small-projects.de>

## Examples

```
# Bonferroni-Holm:
m <- matrix(rep(1/3, 16), nrow=4)
diag(m) <- c(0, 0, 0, 0)
graph <- matrix2graph(m)
print(graph)
graph2matrix(graph)
```

---

placeNodes

*Placement of graph nodes*

---

## Description

Places the nodes of a graph according to a specified layout.

## Usage

```
placeNodes(graph, nrow, ncol, byrow = TRUE,
            topdown = TRUE, force = FALSE)
```

## Arguments

graph	A graph of class <a href="#">graphMCP</a> or class <a href="#">entangledMCP</a> .
nrow	The desired number of rows.
ncol	The desired number of columns.
byrow	Logical whether the graph is filled by rows (otherwise by columns).
topdown	Logical whether the rows are filled top-down or bottom-up.
force	Logical whether a graph that has already a layout should be given the specified new layout.

## Details

If one of `nrow` or `ncol` is not given, an attempt is made to infer it from the number of nodes of the graph and the other parameter. If neither is given, the graph is placed as a circle.

## Value

The graph with nodes placed according to the specified layout.

## Author(s)

Kornelius Rohmeyer <[rohrmeyer@small-projects.de](mailto:rohrmeyer@small-projects.de)>

## See Also

[graphMCP](#), [entangledMCP](#)

**Examples**

```
g <- matrix2graph(matrix(0, nrow=6, ncol=6))

g <- placeNodes(g, nrow=2, force=TRUE)

## Not run:
graphGUI(g)

## End(Not run)
```

---

plotSimCI

*Plot confidence intervals*

---

**Description**

A function for convenient plotting of confidence intervals.

**Usage**

```
plotSimCI(ci)
```

**Arguments**

**ci** a (named) matrix containing the lower confidence bounds in the first column, the point estimates in the second and the upper confidence bounds in the third column.

**Author(s)**

Code adapted from plotCII from Frank Schaarschmidt

**Examples**

```
est <- c("H1"=0.860382, "H2"=0.9161474, "H3"=0.9732953)
# Sample standard deviations:
ssd <- c("H1"=0.8759528, "H2"=1.291310, "H3"=0.8570892)

pval <- c(0.01260, 0.05154, 0.02124)/2

ci <- simConfint(BonferroniHolm(3), pvalues=pval,
  confint="t", df=9, estimates=est, alpha=0.025, alternative="greater")

plotSimCI(ci)
```

---

rejectNode	<i>Rejects a node/hypothesis and updates the graph accordingly.</i>
------------	---

---

**Description**

Rejects a node/hypothesis and updates the graph accordingly.

**Usage**

```
rejectNode(graph, node, verbose = FALSE,  
           keepWeights = TRUE)
```

**Arguments**

graph	A graph of class <code>graphMCP</code> or <code>entangledMCP</code> .
node	A character string specifying the node to reject.
verbose	Logical scalar. If TRUE verbose output is generated during sequentially rejection steps.
keepWeights	Logical scalar. If FALSE the weight of a node without outgoing edges is set to 0 if it is removed. Otherwise it keeps its weight.

**Details**

For details see the given references.

**Value**

An updated graph of class `graphMCP` or `entangledMCP`.

**Author(s)**

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

**References**

Frank Bretz, Willi Maurer, Werner Brannath, Martin Posch: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)

**See Also**

[graphMCP](#)

**Examples**

```
g <- BonferroniHolm(5)  
rejectNode(g, "H1")
```

---

replaceVariables	<i>Replaces variables in a general graph with specified numeric values</i>
------------------	--

---

**Description**

Given a list of variables and real values a general graph is processed and each variable replaced with the specified numeric value.

**Usage**

```
replaceVariables(graph, variables = list(), ask = TRUE)
```

**Arguments**

graph	A graph of class <a href="#">graphMCP</a> or class <a href="#">entangledMCP</a> .
variables	A named list with the specified real values, for example <code>list(a=0.5, b=0.8, "tau"=0.5)</code> . If <code>ask=TRUE</code> and this list is missing at all or single variables are missing from the list, the user is asked for the values (if the session is not interactive an error is thrown).
ask	If <code>FALSE</code> all variables that are not specified are not replaced.

**Value**

A graph or a matrix with variables replaced by the specified numeric values.

**Author(s)**

Kornelius Rohmeyer <[rohrmeyer@small-projects.de](mailto:rohrmeyer@small-projects.de)>

**See Also**

[graphMCP](#), [entangledMCP](#)

**Examples**

```
graph <- HungEtWang2010()
## Not run:
replaceVariables(graph)

## End(Not run)
replaceVariables(graph, list("tau"=0.5,"omega"=0.5, "nu"=0.5))
```



---

`rqmvnorm`*Random sample from the multivariate normal distribution*

---

**Description**

Draw a quasi or pseudo random sample from the MVN distribution. For details on the implemented lattice rule for quasi-random numbers see Cools et al. (2006).

**Usage**

```
rqmvnorm(n, mean = rep(0, nrow(sigma)),
         sigma = diag(length(mean)),
         type = c("quasirandom", "pseudorandom"))
```

**Arguments**

<code>n</code>	Number of samples, when <code>type = "quasirandom"</code> is used this number is rounded up to the next power of 2 (e.g. 1000 to $1024=2^{10}$ ) and at least 1024.
<code>mean</code>	Mean vector
<code>sigma</code>	Covariance matrix
<code>type</code>	What type of random numbers to use. <code>quasirandom</code> uses a randomized Lattice rule, and should be more efficient than <code>pseudorandom</code> that uses ordinary (pseudo) random numbers.

**Value**

Matrix of simulated values

**Author(s)**

We thank Dr. Frances Kuo for the permission to use the generating vectors (order 2 lattice rule) obtained from her website <http://web.maths.unsw.edu.au/~fkuo/lattice/>.

**References**

Cools, R., Kuo, F. Y., and Nuyens, D. (2006) Constructing embedded lattice rules for multivariate integration. *SIAM Journal of Scientific Computing*, 28, 2162-2188.

**Examples**

```
sims <- rqmvnorm(100, mean = 1:2, sigma = diag(2))
plot(sims)
```

---

secondStageTest	<i>EXPERIMENTAL: Construct a valid level alpha test for the second stage of an adaptive design that is based on a pre-planned graphical MCP</i>
-----------------	---

---

### Description

Based on a pre-planned graphical multiple comparison procedure, construct a valid multiple level alpha test that conserves the family wise error in the strong sense regardless of any trial adaptations during an unblinded interim analysis. - Implementation of adaptive procedures is still in an early stage and may change in the near future

### Usage

```
secondStageTest(interim, select, matchCE = TRUE,
  zWeights = "reject", G2 = interim@preplanned)
```

### Arguments

interim	An object of class <a href="#">gPADInterim</a> .
select	A logical vector giving specifying which hypotheses are carried forward to the second stage
matchCE	Logical specifying whether second stage weights should be computed proportional to corresponding PCEs
zWeights	Either "reject", "accept", or "strict" giving the rule what should be done in cases where none of the selected hypotheses has positive second stage weight.
G2	An object of class <a href="#">graphMCP</a> laying down the rule to compute second stage weights. Defaults to pre-planned graph.

### Details

For details see the given references.

### Value

A function of signature `function(z2)` with arguments:  
that returns objects of class [gMCPResult](#).

z2	A numeric vector with second stage z-scores. Z-scores of dropped hypotheses should be set no NA,
----	--

### Author(s)

Florian Klingmueller <[float@lefant.net](mailto:float@lefant.net)>

## References

Frank Bretz, Willi Maurer, Werner Brannath, Martin Posch: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)

Bretz F., Posch M., Glimm E., Klinglmueller F., Maurer W., Rohmeyer K. (2011): Graphical approaches for multiple endpoint problems using weighted Bonferroni, Simes or parametric tests - to appear.

Posch M, Futschik A (2008): A Uniform Improvement of Bonferroni-Type Tests by Sequential Tests *JASA* 103/481, 299-308

Posch M, Maurer W, Bretz F (2010): Type I error rate control in adaptive designs for confirmatory clinical trials with treatment selection at interim *Pharm Stat* 10/2, 96-104

## See Also

[graphMCP](#), [doInterim](#)

## Examples

```
## Simple successive graph (Maurer et al. 2011)
## two treatments two hierarchically ordered endpoints
a <- .025
G <- simpleSuccessiveI()
## some z-scores:

p1=c(.1, .12, .21, .16)
z1 <- qnorm(1-p1)
p2=c(.04, 1, .14, 1)
z2 <- qnorm(1-p2)
v <- c(1/2, 1/3, 1/2, 1/3)

intA <- doInterim(G, z1, v)

## select only the first treatment
fTest <- secondStageTest(intA, c(1, 0, 1, 0))
```

---

simConfint

*Simultaneous confidence intervals for sequentially rejective multiple test procedures*

---

## Description

Calculates simultaneous confidence intervals for sequentially rejective multiple test procedures.

## Usage

```
simConfint(object, pvalues, confint, alternative=c("less", "greater"),
           estimates, df, alpha=0.05, mu=0)
```

**Arguments**

object	A graph of class <code>graphMCP</code> .
pvalues	A numeric vector specifying the p-values for the sequentially rejective MTP.
confint	One of the following: A character string "normal", "t" or a function that calculates the confidence intervals. If <code>confint=="t"</code> the parameter <code>df</code> must be specified. If <code>confint</code> is a function it must be of signature ("character", "numeric"), where the first parameter is the hypothesis name and the second the marginal confidence level (see examples).
alternative	A character string specifying the alternative hypothesis, must be "greater" or "less".
estimates	Point estimates for the parameters of interest.
df	Degree of freedom as numeric.
alpha	The overall alpha level as numeric scalar.
mu	The numerical parameter vector under null hypothesis.

**Details**

For details see the given references.

**Value**

A matrix with columns giving lower confidence limits, point estimates and upper confidence limits for each parameter. These will be labeled as "lower bound", "estimate" and "upper bound".

**Author(s)**

Kornelius Rohmeyer <rohrmeyer@small-projects.de>

**References**

Frank Bretz, Willi Maurer, Werner Brannath, Martin Posch: A graphical approach to sequentially rejective multiple test procedures. *Statistics in Medicine* 2009 vol. 28 issue 4 page 586-604. [http://www.meduniwien.ac.at/fwf\\_adaptive/papers/bretz\\_2009\\_22.pdf](http://www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf)

**See Also**

[graphMCP](#)

**Examples**

```
est <- c("H1"=0.860382, "H2"=0.9161474, "H3"=0.9732953)
# Sample standard deviations:
ssd <- c("H1"=0.8759528, "H2"=1.291310, "H3"=0.8570892)

pval <- c(0.01260, 0.05154, 0.02124)/2

simConfint(BonferroniHolm(3), pvalues=pval,
```

```

confint=function(node, alpha) {
  c(est[node]-qt(1-alpha,df=9)*ssd[node]/sqrt(10), Inf)
}, estimates=est, alpha=0.025, mu=0, alternative="greater")

# Note that the sample standard deviations in the following call
# will be calculated from the pvalues and estimates.
ci <- simConfint(BonferroniHolm(3), pvalues=pval,
  confint="t", df=9, estimates=est, alpha=0.025, alternative="greater")
ci

plotSimCI(ci)

```

---

simvastatin

*Simvastatin and Colesevelam Treatment in Patients with Primary Hypercholesterolemia*


---

### Description

This data set gives the results from a study investigating the efficacy and safety of simvastatin and colesevelam treatment in patients with primary hypercholesterolemia. It shows the sample sizes, the mean LDL cholesterol levels and the number of patients with adverse events after 6 weeks. The treatment groups are: The Placebo control, two doses 10 mg and 20 mg of simvastatin and an combined treatment 20 mg + 2.3 g colesevelam.

### Usage

```
data(simvastatin)
```

### Format

A data frame with a summary table for ...:

**group** A factor with levels "Placebo", "10 mg", "20 mg", "20 mg + 2.3 g Colesevelam" specifying the groups.

**sampleSize** A numeric vector, giving the number of patients in the groups.

**means** A numeric vector, giving the mean LDL cholesterol levels.

**sd** A numeric vector, giving the standard deviation of the LDL cholesterol levels.

**adverseEvents** An integer vector, giving the number of patients with adverse events after 6 weeks.

### Source

Knapp, H.H. and Schrott, H. and Ma, P. and Knopp, R. and Chin, B. and Gaziano, J.M. and Donovan, J.M. and Burke, S.K. and Davidson, M.H. (2001): *Efficacy and safety of combination simvastatin and colesevelam in patients with primary hypercholesterolemia* The American journal of medicine 110, 352-360.

## References

Bretz, F., Hothorn, L. A. and Hsu, J. C. (2003): *Identifying effective and/or safe doses by stepwise confidence intervals for ratios* Statistics in Medicine 22, 847-858.

## Examples

```
data(simvastatin)
barplot(simvastatin$means, names.arg=simvastatin$group)
```

---

subgraph	<i>Get a subgraph</i>
----------	-----------------------

---

## Description

Given a set of nodes and a graph this function creates the subgraph containing only the specified nodes.

## Usage

```
subgraph(graph, subset)
```

## Arguments

graph	A graph of class <a href="#">graphMCP</a> .
subset	A logical or character vector specifying the nodes in the subgraph.

## Value

A subgraph containing only the specified nodes.

## Author(s)

Kornelius Rohmeyer <rohmeier@small-projects.de>

## See Also

[graphMCP](#)

## Examples

```
graph <- improvedParallelGatekeeping()
subgraph(graph, c(TRUE, FALSE, TRUE, FALSE))
subgraph(graph, c("H1", "H3"))
```

---

substituteEps	<i>Substitute Epsilon</i>
---------------	---------------------------

---

**Description**

Substitute Epsilon with a given value.

**Usage**

```
substituteEps(graph, eps = 10^(-3))
```

**Arguments**

graph	A graph of class <a href="#">graphMCP</a> or class <a href="#">entangledMCP</a> .
eps	A numeric scalar specifying a value for epsilon edges.

**Details**

For details see the given references.

**Value**

A graph where all epsilons have been replaced with the given value.

**Author(s)**

Kornelius Rohmeyer <rohmeier@small-projects.de>

**See Also**

[graphMCP](#), [entangledMCP](#)

**Examples**

```
graph <- improvedParallelGatekeeping()  
graph  
substituteEps(graph, eps=0.01)
```

---

unitTestsGMCP

*Run the R unit (and optional the JUnit) test suite for gMCP*


---

### Description

Runs the R unit (and optional the JUnit) test suite for gMCP and prints the results.

### Usage

```
unitTestsGMCP(extended = FALSE, java = FALSE,
              interactive = FALSE, junitLibrary, outputPath)
```

### Arguments

extended	If TRUE (or if the environment variable GMCP_UNIT_TESTS equals "extended" or "all") an extended version of the R unit test suite for gMCP will be used. The run will take significantly longer time. tested in the non-extended version.
java	If TRUE the GUI and its logic is tested with JUnit tests. You need JUnit 4 classes in the classpath or specify the path to a JUnit 4 jar file via the parameter junitLibrary.
interactive	If TRUE the interactive part of the RUnit tests is run. The user have to look at results and answer questions.
junitLibrary	A character String specifying the path to a JUnit 4 jar file to run the JUnit tests. You can download it from <a href="http://www.junit.org/">http://www.junit.org/</a> . Alternatively you can use the environment variable GMCP_JUNIT_LIBRARY to specify the path.
outputPath	During the RUnit tests files maybe produced at this location. If missing the current working directory is used if nothing else is specified in the environment variable GMCP_UNIT_TEST_OPATH.

### Value

None of interest so far - the function prints the results to the standard output. (Perhaps in future versions a value will be returned that can be processed by the GUI.)

### Author(s)

Kornelius Rohmeyer <rohmeier@small-projects.de>

### Examples

```
## Not run:
unitTestsForGMCP()
unitTestsGMCP(extended=TRUE, java=TRUE, interactive=TRUE, outputPath=~"/RUnitTests")

## End(Not run)
```



# Index

- \*Topic **IO**
  - gMCPReport, 25
- \*Topic **datasets**
  - hydroquinone, 34
  - simvastatin, 45
- \*Topic **distribution**
  - rmvnorm, 41
- \*Topic **file**
  - gMCPReport, 25
- \*Topic **graphs**
  - corMatWizard, 7
  - doInterim, 8
  - entangledMCP-class, 10
  - exampleGraphs, 11
  - gMCP, 22
  - gMCP-package, 2
  - gMCPReport, 25
  - gMCPResult-class, 26
  - gPADInterim-class, 26
  - graph2latex, 27
  - graphAnalysis, 28
  - graphGUI, 29
  - graphMCP-class, 31
  - joinGraphs, 35
  - matrix2graph, 36
  - placeNodes, 37
  - rejectNode, 39
  - replaceVariables, 40
  - secondStageTest, 42
  - simConfint, 43
  - subgraph, 46
  - substituteEps, 47
- \*Topic **hplot**
  - plotSimCI, 38
- \*Topic **htest**
  - calcPower, 5
  - doInterim, 8
  - extractPower, 14
  - generateBounds, 15
  - generatePvals, 17
  - generateTest, 19
  - generateWeights, 21
  - gMCP, 22
  - gMCP-package, 2
  - graphTest, 32
  - rejectNode, 39
  - secondStageTest, 42
  - simConfint, 43
- \*Topic **misc**
  - corMatWizard, 7
  - exampleGraphs, 11
  - graphGUI, 29
- \*Topic **package**
  - gMCP-package, 2
- \*Topic **print**
  - gMCPReport, 25
  - graph2latex, 27
  - replaceVariables, 40
  - subgraph, 46
  - substituteEps, 47
- BauerEtAl2001 (exampleGraphs), 11
- bdiagNA, 4
- body, 12
- BonferroniHolm (exampleGraphs), 11
- BretzEtAl2009a (exampleGraphs), 11
- BretzEtAl2009b (exampleGraphs), 11
- BretzEtAl2009c (exampleGraphs), 11
- BretzEtAl2011 (exampleGraphs), 11
- calcPower, 5
- cat, 25
- corMatWizard, 7
- cycleGraph (exampleGraphs), 11
- doInterim, 8, 27, 43
- edgeAttr (graphMCP-class), 31
- edgeAttr, graphMCP, character, character, character-method (graphMCP-class), 31

- edgeAttr<- (graphMCP-class), 31
- edgeAttr<-, graphMCP, character, character, character-method (graphMCP-class), 31
- Entangled1Maurer2012 (exampleGraphs), 11
- Entangled2Maurer2012 (exampleGraphs), 11
- entangledMCP, 37, 39, 40, 47
- entangledMCP (entangledMCP-class), 10
- entangledMCP-class, 10
- exampleGraphs, 11
- extractPower, 14
  
- fallback (exampleGraphs), 11
- Ferber2011 (exampleGraphs), 11
- FerberTimeDose2011 (exampleGraphs), 11
- fixedSequence (exampleGraphs), 11
  
- generalSuccessive (exampleGraphs), 11
- generateBounds, 15
- generatePvals, 17, 23
- generateTest, 19
- generateWeights, 15, 17, 21
- getJavaInfo, 22
- getMatrices (entangledMCP-class), 10
- getMatrices, entangledMCP-method (entangledMCP-class), 10
- getMatrix (graphMCP-class), 31
- getMatrix, graphMCP-method (graphMCP-class), 31
- getNodes (graphMCP-class), 31
- getNodes, entangledMCP-method (entangledMCP-class), 10
- getNodes, graphMCP-method (graphMCP-class), 31
- getRejected (graphMCP-class), 31
- getRejected, entangledMCP-method (entangledMCP-class), 10
- getRejected, gMCPResult-method (gMCPResult-class), 26
- getRejected, gPADInterim-method (gPADInterim-class), 26
- getRejected, graphMCP-method (graphMCP-class), 31
- getWeights (graphMCP-class), 31
- getWeights, entangledMCP-method (entangledMCP-class), 10
- getWeights, gMCPResult-method (gMCPResult-class), 26
- getWeights, gPADInterim-method (gPADInterim-class), 26
- getWeights, graphMCP-method (graphMCP-class), 31
- getXCoordinates (graphMCP-class), 31
- getXCoordinates, entangledMCP-method (entangledMCP-class), 10
- getXCoordinates, graphMCP-method (graphMCP-class), 31
- getYCoordinates (graphMCP-class), 31
- getYCoordinates, entangledMCP-method (entangledMCP-class), 10
- getYCoordinates, graphMCP-method (graphMCP-class), 31
- gMCP, 3, 4, 22, 26, 27
- gMCP-package, 2
- gMCPReport, 25, 28
- gMCPResult, 25, 42
- gMCPResult (gMCPResult-class), 26
- gMCPResult-class, 26
- gPADInterim, 42
- gPADInterim (gPADInterim-class), 26
- gPADInterim-class, 26
- graph2latex, 25, 27
- graph2matrix (matrix2graph), 36
- graphAnalysis, 28
- graphGUI, 3, 29
- graphMCP, 3, 8–10, 13, 22, 24–30, 35–37, 39, 40, 42–44, 46, 47
- graphMCP (graphMCP-class), 31
- graphMCP-class, 31
- graphNEL, 24
- graphTest, 32
  
- HommelEtAl12007 (exampleGraphs), 11
- HommelEtAl12007Simple (exampleGraphs), 11
- HungEtWang2010 (exampleGraphs), 11
- HuqueAloshEtBhore2011 (exampleGraphs), 11
- hydroquinone, 34
  
- improvedFallbackI (exampleGraphs), 11
- improvedFallbackII (exampleGraphs), 11
- improvedParallelGatekeeping (exampleGraphs), 11
  
- joinGraphs, 35
- make.names, 30
- matrix2graph, 36
- MaurerEtAl1995 (exampleGraphs), 11

nodeAttr (graphMCP-class), 31  
nodeAttr, graphMCP, character, character-method  
(graphMCP-class), 31  
nodeAttr<- (graphMCP-class), 31  
nodeAttr<-, graphMCP, character, character-method  
(graphMCP-class), 31

parallelGatekeeping (exampleGraphs), 11  
placeNodes, 37  
plot, gMCPResult, ANY-method  
(gMCPResult-class), 26  
plot, gPADInterim-method  
(gPADInterim-class), 26  
plot, graphMCP, ANY-method  
(graphMCP-class), 31  
plotSimCI, 38  
print, entangledMCP-method  
(entangledMCP-class), 10  
print, gMCPResult-method  
(gMCPResult-class), 26  
print, gPADInterim-method  
(gPADInterim-class), 26  
print, graphMCP-method (graphMCP-class),  
31

rejectNode, 39  
replaceVariables, 40  
rqmnorm, 41

secondStageTest, 9, 27, 42  
setEdge (graphMCP-class), 31  
setEdge, character, character, graphMCP, character-method  
(graphMCP-class), 31  
setEdge, character, character, graphMCP, numeric-method  
(graphMCP-class), 31  
setRejected<- (graphMCP-class), 31  
setRejected<-, graphMCP-method  
(graphMCP-class), 31  
setWeights (graphMCP-class), 31  
setWeights, graphMCP-method  
(graphMCP-class), 31  
simConfint, 43  
simConfint, graphMCP-method  
(simConfint), 43  
simpleSuccessiveI (exampleGraphs), 11  
simpleSuccessiveII (exampleGraphs), 11  
simvastatin, 45  
subgraph, 46  
substituteEps, 47  
truncatedHolm (exampleGraphs), 11  
unitTestsGMCP, 48