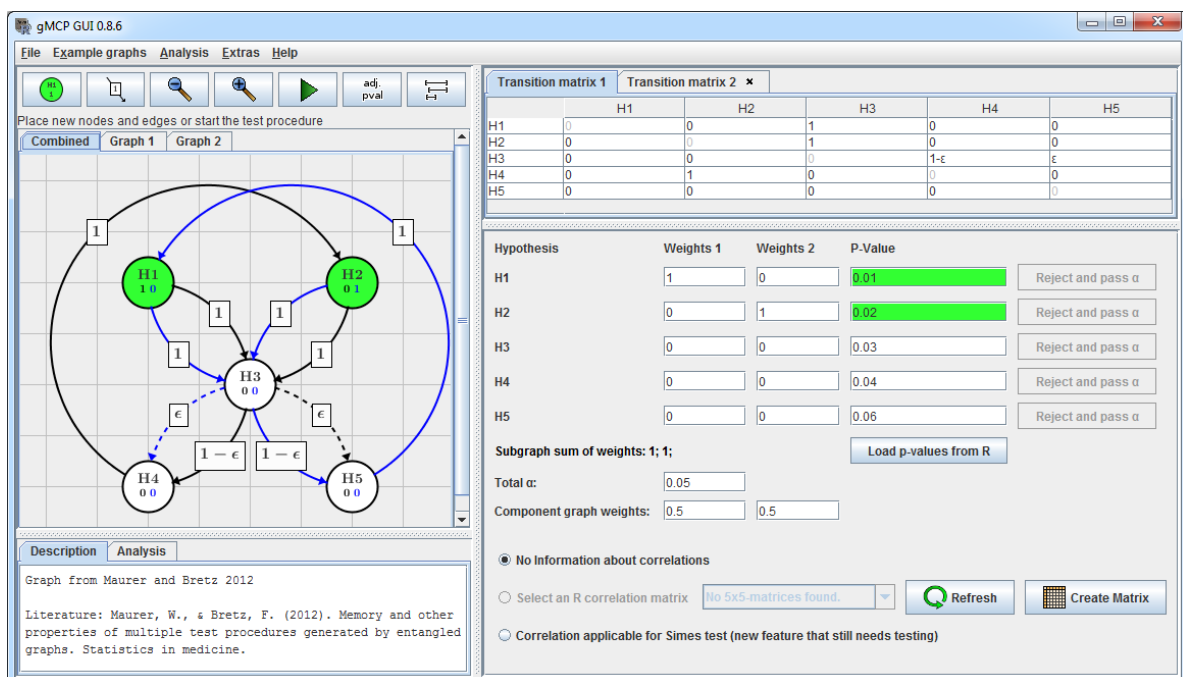


gMCP - an R package for a graphical approach to weighted multiple test procedures

Kornelius Rohmeyer

March 30, 2014



Contents

1	Introduction	4
1.1	Installation	4
1.2	Basic Theoretical Background	4
1.3	Example and diving in	5
2	Creating Weighted Graphs	7
2.1	Using R	7
2.2	Using the GUI	9
3	The sequentially rejective MTP	10
3.1	Using R	10
3.1.1	Adjusted p-values and simultaneous confidence intervals	12
3.2	Using the GUI	13
4	Weighted parametric and Simes tests	13
4.1	Correlation matrix creation	14
5	Epsilon edges	14
6	Entangled graphs	16
7	Power Simulations	18
7.1	Using the R command line	18
7.2	Variable edge weights	18
8	Options and Import/Export	20
8.1	Options	20
8.1.1	Privacy	21
8.2	Import/Exports	21
8.3	Important TikZ commands for optimizing the reports	22
9	Case Studies	23
9.1	Identifying effective and/or safe doses by stepwise confidence intervals for ratios	24
9.2	Testing strategies in multi-dose experiments including active control	24
A	Appendix - Example graphs	26
B	Appendix - Multiple Testing Basics	35
C	Appendix - Graph Theory Basics	36
	Index	37
	Literatur	39

1 Introduction

This package provides functions and graphical user interfaces for graph based multiple test procedures. By defining weighted directed graphs one also defines a weighting strategy for all subsets of null hypotheses. Weighted tests can be performed on these subsets and following the closed test procedure this leads to a multiple test procedure controlling the family wise error rate in the strong sense. In some cases shortcuts are available, one example is the weighted Bonferroni procedure that leads to a sequentially rejective multiple test procedure.

For all steps either graphical user interfaces or the R Console with S4 objects and methods can be used.

1.1 Installation

If you don't already have R (R Core Team [2014]) on your system, you can download a bundled version of R and gMCP from <http://www.algorithm-forge.com/gMCP/bundle/>.

Otherwise open R and type `install.packages("gMCP")`, select an arbitrary mirror and gMCP will be downloaded and installed.

Once it is installed, whenever you start R you can load the gMCP package by entering `library(gMCP)` into the R Console. The graphical user interface is started with the command `graphGUI()`.

If you run into problems, see <http://cran.r-project.org/web/packages/gMCP/INSTALL> or please write us an email at help@small-projects.de. We are eager to help and to learn about existing problems.

1.2 Basic Theoretical Background

Graph based multiple test procedures are closed test procedures, i.e. for a family $\{H_i \mid i \in I\}$, $I = \{1, \dots, n\}$ of elementary hypotheses each intersection $\bigcap_{j \in J} H_j$, $J \subset I$ is tested with a local level α test. Following the closed testing principle one can derive a multiple test procedure that controls the family-wise error rate (FWER) at level α .

The local level α tests in gMCP are weighted tests, where the weights are derived from a directed weighted graph G_I . Examples of weighed tests that are available in gMCP are the weighted Bonferroni, parametric and Simes tests.

For each intersection $\bigcap_{j \in J} H_j$, $J \subset I$ a graph G_J can be derived from G_I and the weights for the weighted local test for $\bigcap_{j \in J} H_j$ are the weights of the nodes of G_J . To derive graph G_J remove all nodes that are not in J and update the edges of the graph according to Algorithm 1 (the order does not matter). For a more detailed version please take a look at the article from Bretz et al. [2011b] that is freely available as an OnlineOpen Article.

Algorithm 1 Removing node i , passing the weight and updating the graph edges

```
for  $l \in I$  do
   $w_l \leftarrow w_l + w_i \cdot g_{il}$ 
  for  $k \in I$  do
    if  $l \neq k$  and  $g_{lj} \cdot g_{jl} \neq 1$  then
       $g_{lk} \leftarrow \frac{g_{lk} + g_{lj} \cdot g_{jk}}{1 - g_{lj} \cdot g_{jl}}$ 
    else
       $g_{lk} \leftarrow 0$ 
    end if
  end for
end for
```

1.3 Example and diving in

Let's start with a well-known procedure and see how it fits into this graphical approach to weighted multiple test procedures: The Bonferroni-Holm-Procedure from Holm [1979].

Theorem 1.1 (Bonferroni-Holm-Procedure). *Let T_1, \dots, T_m be test statistics for $m \in \mathbb{N}$ null hypotheses H_1, \dots, H_m and p_1, \dots, p_m the associated p -values. Then the following test will control the familywise error rate at level $\alpha \in]0, 1[$ in the strong sense:*

Denote the ordered p -values by $p^{(1)} < p^{(2)} < \dots < p^{(m)}$ and the corresponding hypotheses by $H^{(1)}, H^{(2)}, \dots, H^{(m)}$. Reject $H^{(1)}, H^{(2)}, \dots, H^{(j)}$ such that

$$p^{(i)} \leq \frac{\alpha}{n - i + 1} \quad \text{for all } 1 \leq i \leq j.$$

The corresponding graph for the Bonferroni-Holm-Procedure for three hypotheses is given in Figure 1. We see a fully connected graph, where each node represents a hypothesis and the nodes and edges have weights.

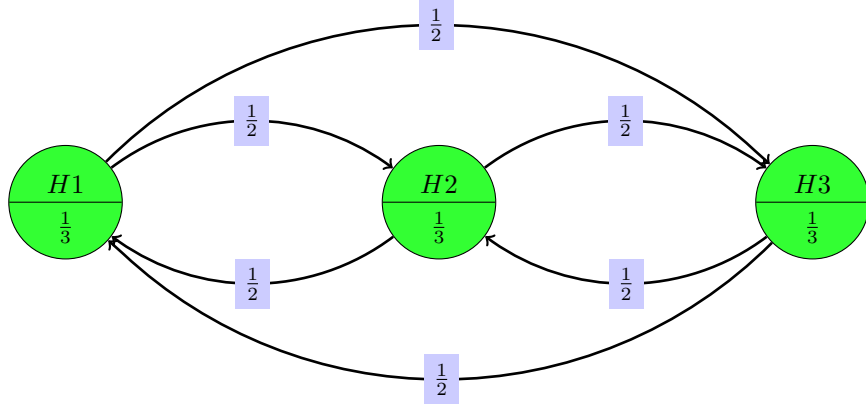


Figure 1: Graph representing the Bonferroni-Holm-Procedure for three hypotheses.

A null hypothesis can be rejected, when the p -value is less than the alpha level of the corresponding node. In this case the graph will be updated and the alpha level of this node is passed according to the edge weights.

Example 1.2. We give an example for the Bonferroni-Holm-Procedure. Of course this package is made for more advanced tests (you find a selection in appendix A with applied examples in section 9), but since most readers are already familiar with this procedure, for a first introduction of gMCP, we stick to this simple example.

Let $p_1 = 0.01$, $p_2 = 0.07$ and $p_3 = 0.02$ be three p -values and $\alpha = 0.05$. In the first step H_1 can be rejected since $p_1 < \alpha/3$. The updated graph can be seen in figure 2 and now also H_3 can be rejected since $p_1 < \alpha/2$. Again the graph is updated, but H_2 can not be rejected.

Let's reproduce this with the gMCP package. We start R and enter:

```
library(gMCP)
graphGUI()
```

The GUI seen in Figure 4 is shown and we select from the menu "Example graphs" the entry "Bonferroni-Holm Test". We enter the three p -values in the respective fields on the right side. By clicking on the button with the green arrow we start the test procedure and can sequentially reject all three hypotheses.

If we don't want to use the GUI we can also use R:

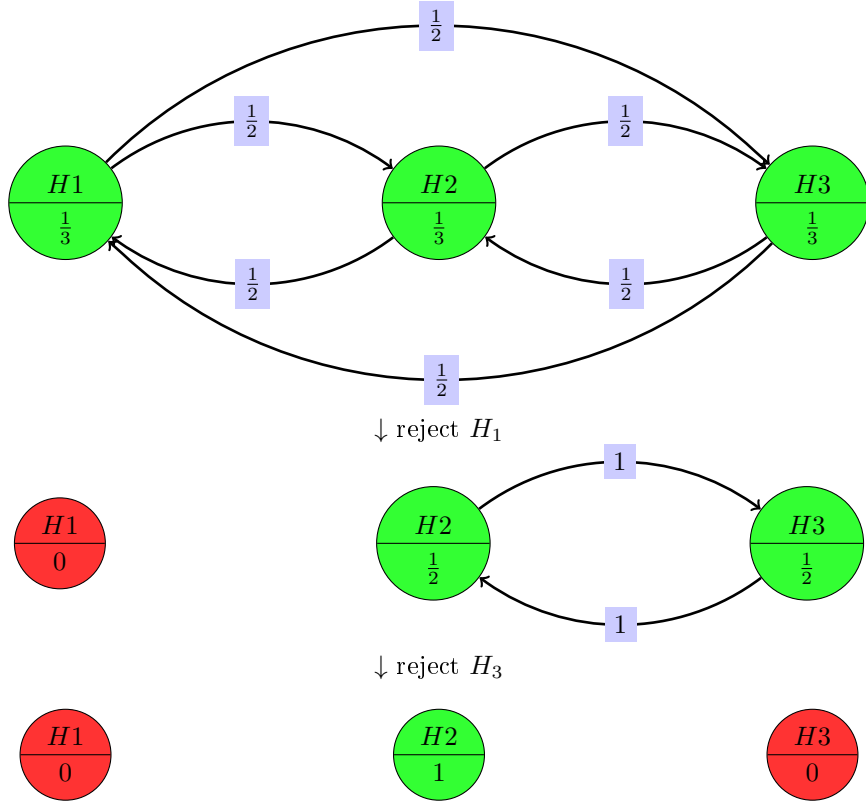


Figure 2: Example showing how two null hypotheses can be rejected with p-values $p_1 = 0.01$, $p_2 = 0.07$ and $p_3 = 0.02$.

```
library(gMCP)
graph <- BonferroniHolm(3)
gMCP(graph, pvalues = c(0.01, 0.07, 0.02), alpha = 0.05)

## gMCP-Result
##
## Initial graph:
## A graphMCP graph
## H1 (weight=0.3333)
## H2 (weight=0.3333)
## H3 (weight=0.3333)
## Edges:
## H1 -( 0.5 )-> H2
## H1 -( 0.5 )-> H3
## H2 -( 0.5 )-> H1
## H2 -( 0.5 )-> H3
## H3 -( 0.5 )-> H1
## H3 -( 0.5 )-> H2
##
##
## P-values:
##   H1   H2   H3
## 0.01 0.07 0.02
##
## Adjusted p-values:
##   H1   H2   H3
## 0.03 0.07 0.04
##
## Alpha: 0.05
##
```

```
## Hypothesis rejected:
##   H1   H2   H3
##  TRUE FALSE TRUE
##
## Final graph after 2 steps:
## A graphMCP graph
## H1 (rejected, weight=0)
## H2 (weight=1)
## H3 (rejected, weight=0)
## No edges.
```

2 Creating Weighted Graphs

In the first step a graph that describes the multiple test procedures must be created.

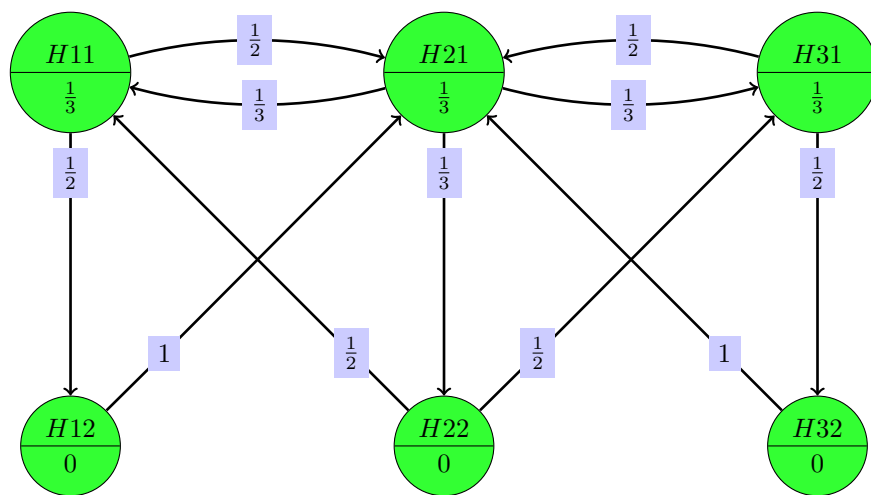


Figure 3: Example graph from Bretz et al. [2011a] that we will create in this vignette.

2.1 Using R

The most convenient way to create a graph in R is to use the functions `matrix2graph` and `setWeights`. As an example we create the graph from Bretz et al. [2011a] that you can see in figure 3.

```
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))

graph <- matrix2graph(m)
graph <- setWeights(graph, c(1/3, 1/3, 1/3, 0, 0, 0))
```

For accessing the weights and transition matrix of an existing graph the functions `getWeights` and `getMatrix` are provided.

Let's print the newly created graph:

```
print(graph)

## A graphMCP graph
## H11 (weight=0.3333)
## H21 (weight=0.3333)
## H31 (weight=0.3333)
## H12 (weight=0)
## H22 (weight=0)
## H32 (weight=0)
## Edges:
## H11 -( 0.5 )-> H21
## H11 -( 0.5 )-> H12
## H21 -( 0.3333333333333333 )-> H11
## H21 -( 0.3333333333333333 )-> H31
## H21 -( 0.3333333333333333 )-> H22
## H31 -( 0.5 )-> H21
## H31 -( 0.5 )-> H32
## H12 -( 1 )-> H21
## H22 -( 0.5 )-> H11
## H22 -( 0.5 )-> H31
## H32 -( 1 )-> H21
```

Since we also want to visualize the graph, we set two node attributes **X** and **Y** (for further information see the manual pages of method **nodeAttr**).

```
graph@nodeAttr$X <- c(H11 = 100, H21 = 300, H31 = 500, H12 = 100, H22 = 300, H32 = 500)
graph@nodeAttr$Y <- c(H11 = 100, H21 = 100, H31 = 100, H12 = 300, H22 = 300, H32 = 300)
```

For placement of the nodes in a matrix pattern, the function **placeNodes** is helpful. The following code does the same as the two lines of R code above.

```
graph <- placeNodes(graph, nrow = 2)
```

Coordinates are interpreted as pixels in the GUI and big points in \LaTeX (72 bp = 1 inch).

Let's take a look at the graph in \LaTeX rendered with package TikZ from Tantau [2008] (figure 3 shows the compiled result):

```
cat(graph2latex(graph))

## \begin{tikzpicture}
##
## \node (H11) at (125bp,-125bp)[draw,circle split,fill=green!80] {H11 \nodepart{lower}  $\frac{1}{3}$ };
## \node (H21) at (325bp,-125bp)[draw,circle split,fill=green!80] {H21 \nodepart{lower}  $\frac{1}{3}$ };
## \node (H31) at (525bp,-125bp)[draw,circle split,fill=green!80] {H31 \nodepart{lower}  $\frac{1}{3}$ };
## \node (H12) at (125bp,-325bp)[draw,circle split,fill=green!80] {H12 \nodepart{lower} 0};
## \node (H22) at (325bp,-325bp)[draw,circle split,fill=green!80] {H22 \nodepart{lower} 0};
## \node (H32) at (525bp,-325bp)[draw,circle split,fill=green!80] {H32 \nodepart{lower} 0};
## \draw [->,line width=1pt] (H11) to[bend left=15] node[near start,above,fill=blue!20] { $\frac{1}{2}$ } (H21);
## \draw [->,line width=1pt] (H11) to[auto] node[near start,above,fill=blue!20] { $\frac{1}{2}$ } (H12);
## \draw [->,line width=1pt] (H21) to[bend left=15] node[near start,above,fill=blue!20] { $\frac{1}{3}$ } (H11);
## \draw [->,line width=1pt] (H21) to[bend left=15] node[near start,above,fill=blue!20] { $\frac{1}{3}$ } (H31);
## \draw [->,line width=1pt] (H21) to[auto] node[near start,above,fill=blue!20] { $\frac{1}{3}$ } (H22);
## \draw [->,line width=1pt] (H31) to[bend left=15] node[near start,above,fill=blue!20] { $\frac{1}{2}$ } (H21);
## \draw [->,line width=1pt] (H31) to[auto] node[near start,above,fill=blue!20] { $\frac{1}{2}$ } (H32);
## \draw [->,line width=1pt] (H12) to[auto] node[near start,above,fill=blue!20] {1} (H21);
## \draw [->,line width=1pt] (H22) to[auto] node[near start,above,fill=blue!20] { $\frac{1}{2}$ } (H11);
## \draw [->,line width=1pt] (H22) to[auto] node[near start,above,fill=blue!20] { $\frac{1}{2}$ } (H31);
## \draw [->,line width=1pt] (H32) to[auto] node[near start,above,fill=blue!20] {1} (H21);
## \end{tikzpicture}
```


We can even change the position of the edge labels for further fine tuning of the graphical representation. With the following command we place the label for the edge from H1 to H2 at position (200, 80):

```
edgeAttr(graph, "H11", "H21", "labelX") <- 200
edgeAttr(graph, "H11", "H21", "labelY") <- 80
```

2.2 Using the GUI

The creation of **graphMCP** objects as seen in the last section with basic R commands is very straight forward, but still takes some time and typos may occur. More convenient for the most users is the use of the graphical user interface for creating and editing MCP graphs that the **gMCP** package includes.

It is called by the command **graphGUI()** and takes as optional argument a variable name, given as a character string, of the graph to edit.

```
graphGUI("graph")
```

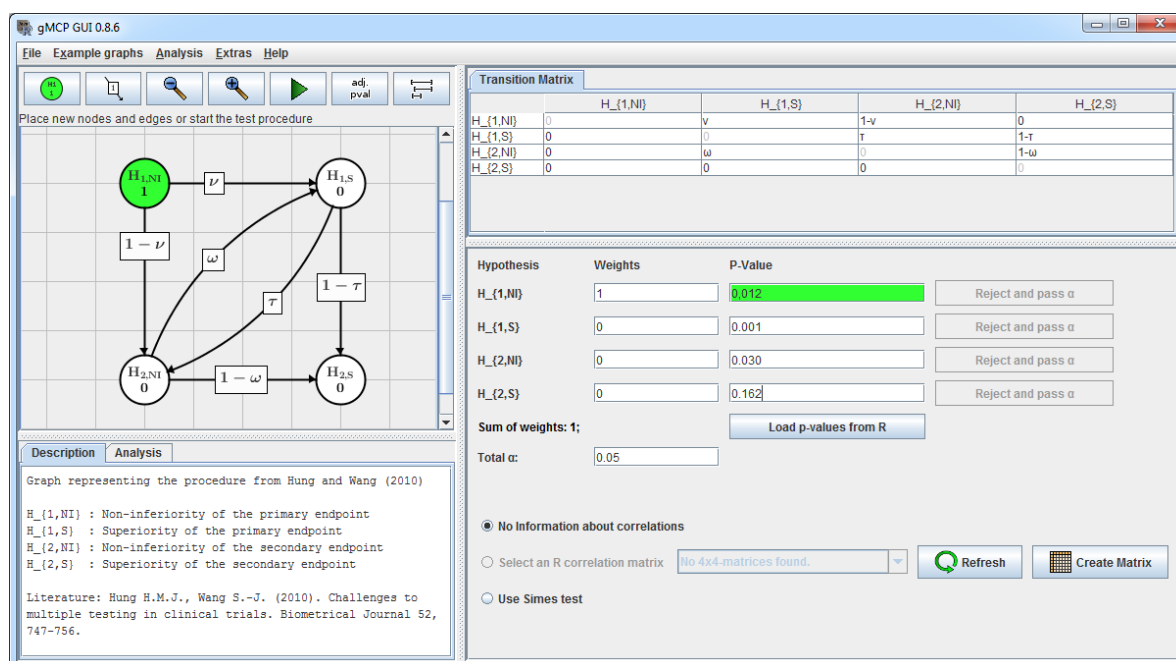








Figure 4: The graphical user interface allows testing, calculation of confidence intervals and adjusted p-values.

Let's take a look at the icon panel:

-  This button lets you add a new node to the graph. After pressing the button click somewhere on the graph panel and a new node will appear at this place.
-  This button lets you add a new edge between two nodes. After pressing the button click on the node the edge should start and after that on the node the edge should end.
-  For really big graphs the ability to zoom in and out is usefull.
-  Starts the testing procedure / goes back to the graph modification.
-  Calculates the adjusted p-values.
-  Calculates simultaneous confidence intervals.

With drag and drop you can move nodes and also adjust edges. Double clicking the nodes or edges will open a property dialog that can also be accessed via the right-click context menu. These dialogs and context menus also give you the option to delete the selected node/edge.

As seen in figure 5 the GUI will show you R code to reproduce results.

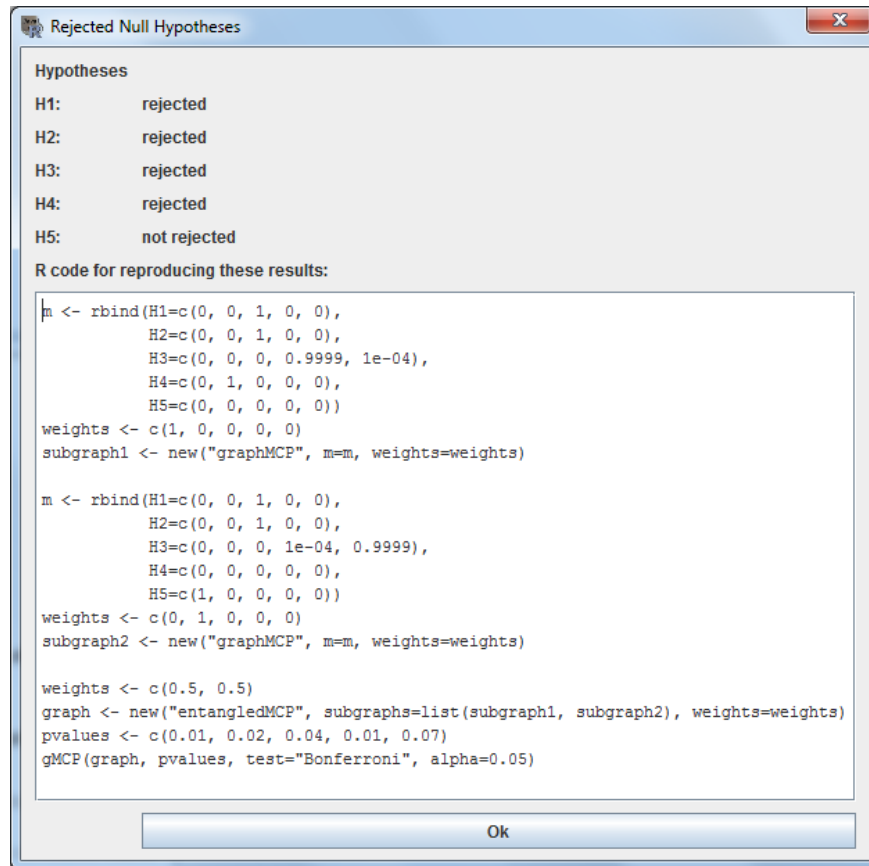


Figure 5: R Code generated and shown by the GUI to reproduce the results.

3 The sequentially rejective MTP

For a full description of the sequentially rejective multiple testing procedure take a look at Bretz et al. [2009].

3.1 Using R

You can either specify each rejection step yourself or simply use the method `gmCP`:

```
graph <- BretzEtAl2011()

# We can reject a single node:
print(rejectNode(graph, "H11"))

## A graphMCP graph
## H11 (rejected, weight=0)
## H21 (weight=0.5)
## H31 (weight=0.3333)
## H12 (weight=0.1667)
## H22 (weight=0)
```

```

## H32 (weight=0)
## Edges:
## H21 -( 0.4 )-> H31
## H21 -( 0.2 )-> H12
## H21 -( 0.4 )-> H22
## H31 -( 0.5 )-> H21
## H31 -( 0.5 )-> H32
## H12 -( 1 )-> H21
## H22 -( 0.25 )-> H21
## H22 -( 0.5 )-> H31
## H22 -( 0.25 )-> H12
## H32 -( 1 )-> H21

# Or given a vector of pvalues let the function gMCP do all the work:
pvalues <- c(0.1, 0.008, 0.005, 0.15, 0.04, 0.006)
result <- gMCP(graph, pvalues)
print(result)

## gMCP-Result
##
## Initial graph:
## A graphMCP graph
## H11 (weight=0.3333)
## H21 (weight=0.3333)
## H31 (weight=0.3333)
## H12 (weight=0)
## H22 (weight=0)
## H32 (weight=0)
## Edges:
## H11 -( 0.5 )-> H21
## H11 -( 0.5 )-> H12
## H21 -( 0.3333333333333333 )-> H11
## H21 -( 0.3333333333333333 )-> H31
## H21 -( 0.3333333333333333 )-> H22
## H31 -( 0.5 )-> H21
## H31 -( 0.5 )-> H32
## H12 -( 1 )-> H21
## H22 -( 0.5 )-> H11
## H22 -( 0.5 )-> H31
## H32 -( 1 )-> H21
##
##
## P-values:
##   H11  H21  H31  H12  H22  H32
## 0.100 0.008 0.005 0.150 0.040 0.006
##
## Adjusted p-values:
##   H11  H21  H31  H12  H22  H32
## 0.1200 0.0160 0.0150 0.1500 0.1200 0.0225
##
## Alpha: 0.05
##
## Hypothesis rejected:
##   H11  H21  H31  H12  H22  H32
## FALSE TRUE TRUE FALSE FALSE TRUE
##
## Final graph after 3 steps:
## A graphMCP graph
## H11 (weight=0.6667)
## H21 (rejected, weight=0)

```

```
## H31 (rejected, weight=0)
## H12 (weight=0)
## H22 (weight=0.3333)
## H32 (rejected, weight=0)
## Edges:
## H11 -( 0.666666666666667 )-> H12
## H11 -( 0.333333333333333 )-> H22
## H12 -( 0.5 )-> H11
## H12 -( 0.5 )-> H22
## H22 -( 1 )-> H11
```

We can create a TikZ graphic from the last graph with `graph2latex(result@graphs[[4]])` that is shown in figure 6.

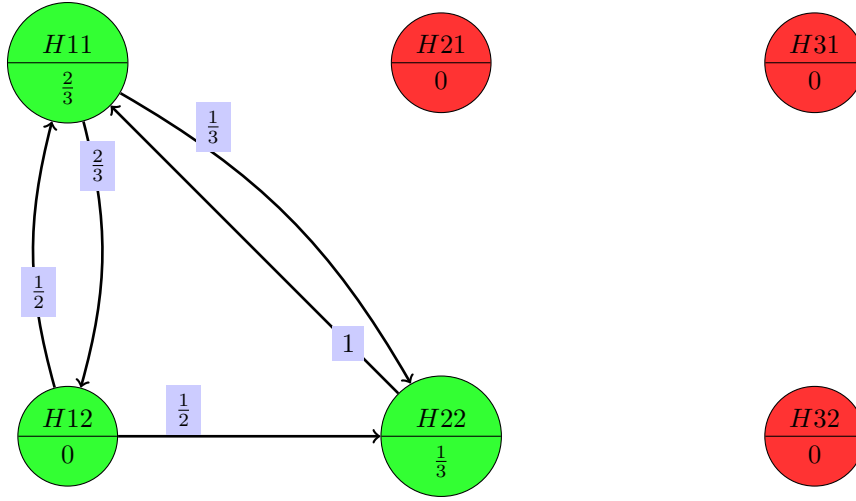


Figure 6: Final graph from the test procedure after rejection of H_{21} , H_{31} and H_{32} .

The command `gMCPReport` generates a full report of the testing procedure:

```
gMCPReport(result, "Report.tex")
```

3.1.1 Adjusted p-values and simultaneous confidence intervals

Also adjusted p-values and simultaneous confidence intervals can be computed.

Let's assume the tests for hypotheses $H1 : \theta_1 \leq 0$, $H2 : \theta_2 \leq 0$ and $H3 : \theta_3 \leq 0$ are three t-tests with degree of freedom 9. The estimates are $\hat{\theta}_1 = 0.981$, $\hat{\theta}_2 = 1.089$ and $\hat{\theta}_3 = 0.8706$, the sample standard deviations $s_1 = 0.876$, $s_2 = 1.291$ and $s_3 = 0.8571$ the t-statistics 3.541, 2.666 and 3.212 and the corresponding p-values 0.0063, 0.02577 and 0.01062. We want to adjust for multiple testing by using the Bonferroni-Holm-Procedure with $\alpha = 0.025$.

```
# Estimates:
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")
```

```
##      lower bound estimate upper bound
## H1      0.0000  0.8604      Inf
## H2     -0.0076  0.9161      Inf
## H3      0.0000  0.9733      Inf

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

##      lower bound estimate upper bound
## [1,]      0.000000  0.8604      Inf
## [2,]    -0.007581  0.9161      Inf
## [3,]      0.000000  0.9733      Inf
```

3.2 Using the GUI

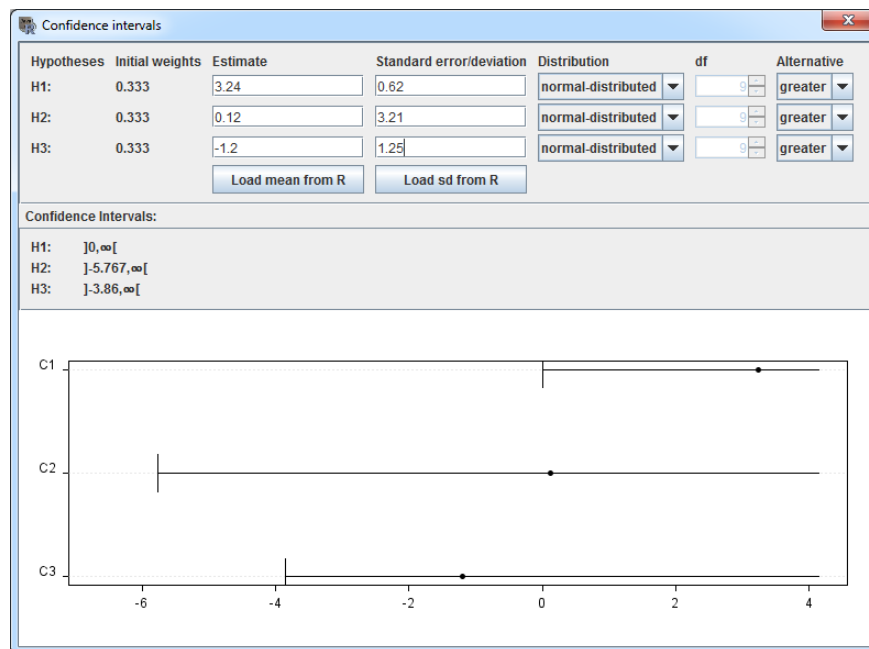


Figure 7: For normal and t-distributions simultaneous CI can be calculated by the GUI.

Use the following two buttons:



See Bretz et al. [2011b].

4 Weighted parametric and Simes tests

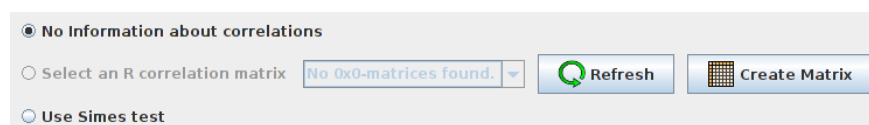


Figure 8: You can also specify a correlation between the tests.

In the lower right panel with p-values, it is also possible to specify a known correlation between the original test statistics (see figure 8). Either you can perform a Simes test or a weighted parametric tests

as described in Bretz et al. [2011b]. For the later 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 Σ where Φ^{-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. For further information please take a look at the vignette "Weighted parametric tests defined by graphs".

4.1 Correlation matrix creation

The GUI features a dialog for easy creation of correlation matrices (see figure 9).

	H1	H2	H3	H4	H5	H6
H1	1	1/2	1/2	0	0	0
H2	1/2	1	1/2	0	0	0
H3	1/2	1/2	1	0	0	0
H4	0	0	0	1	1/2	1/2
H5	0	0	0	1/2	1	1/2
H6	0	0	0	1/2	1/2	1

Figure 9: Dialog for specifying a correlation matrix.

If the entered matrix is not positive semidefinite, i.e. negative eigen values exist, a warning is given. This dialog is perhaps useful on its own and can be opened by calling the function `corMatWizard`.

5 Epsilon edges

The GUI supports epsilon edges. You can enter the weights in R syntax, e.g. $1 - 2\epsilon + \frac{1}{3}\epsilon^2$ for $1 - 2\epsilon + \frac{1}{3}\epsilon^2$.

```
m <- rbind(H1=c(0,      0,      0.5,      0.5      ),
           H2=c(0,      0,      0.5,      0.5      ),
           H3=c("\\epsilon", 0,      0,      "1-\\epsilon"),
           H4=c(0,      "\\epsilon", "1-\\epsilon", 0      ))

graph <- matrix2graph(m)
#graph <- improvedParallelGatekeeping()
graph

## A graphMCP graph
## H1 (weight=0.25)
## H2 (weight=0.25)
```

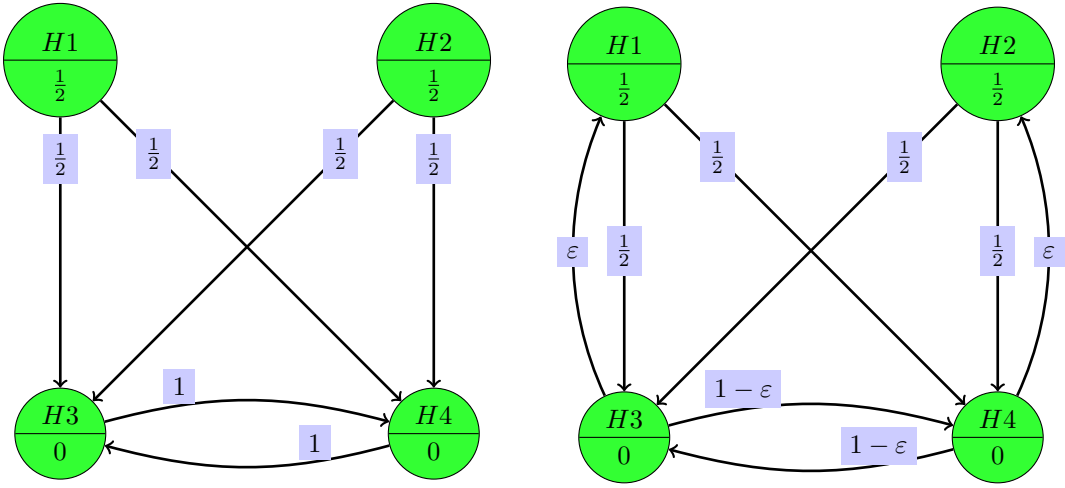


Figure 10: The Parallel Gatekeeping and the Improved Parallel Gatekeeping Procedure.

```
## H3 (weight=0.25)
## H4 (weight=0.25)
## Edges:
## H1 -( 0.5 )-> H3
## H1 -( 0.5 )-> H4
## H2 -( 0.5 )-> H3
## H2 -( 0.5 )-> H4
## H3 -( \epsilon )-> H1
## H3 -( 1-\epsilon )-> H4
## H4 -( \epsilon )-> H2
## H4 -( 1-\epsilon )-> H3

substituteEps(graph, eps=0.001)

## A graphMCP graph
## H1 (weight=0.25)
## H2 (weight=0.25)
## H3 (weight=0.25)
## H4 (weight=0.25)
## Edges:
## H1 -( 0.5 )-> H3
## H1 -( 0.5 )-> H4
## H2 -( 0.5 )-> H3
## H2 -( 0.5 )-> H4
## H3 -( 0.001 )-> H1
## H3 -( 0.999 )-> H4
## H4 -( 0.001 )-> H2
## H4 -( 0.999 )-> H3

gMCP(graph, pvalues=c(0.02, 0.04, 0.01, 0.02), eps=0.001)

## gMCP-Result
##
## Initial graph:
## A graphMCP graph
## H1 (weight=0.25)
## H2 (weight=0.25)
## H3 (weight=0.25)
## H4 (weight=0.25)
## Edges:
## H1 -( 0.5 )-> H3
```

```
## H1 -( 0.5 )-> H4
## H2 -( 0.5 )-> H3
## H2 -( 0.5 )-> H4
## H3 -( 0.001 )-> H1
## H3 -( 0.999 )-> H4
## H4 -( 0.001 )-> H2
## H4 -( 0.999 )-> H3
##
##
## P-values:
##   H1   H2   H3   H4
## 0.02 0.04 0.01 0.02
##
## Adjusted p-values:
##       H1       H2       H3       H4
## 0.04002 0.04002 0.04000 0.04002
##
## Alpha: 0.05
##
## Hypothesis rejected:
##   H1   H2   H3   H4
## TRUE TRUE TRUE TRUE
##
## Final graph after 4 steps:
## A graphMCP graph
## H1 (rejected, weight=0)
## H2 (rejected, weight=1)
## H3 (rejected, weight=0)
## H4 (rejected, weight=0)
## No edges.
```

6 Entangled graphs

As a new feature we support entangled graphs (see Maurer and Bretz [2013]), which allow us to describe an even bigger class of multiple test procedures in the familiar graphical way. Most notably these test procedures can have some kind of memory when passing the local significance levels to unrejected null hypotheses, i.e. for the step-wise rejective Bonferroni based test procedure the passing of alpha levels when a null hypothesis is rejected can be different depending from which previous rejected nodes this alpha level originates.

Definition 6.1 (Entangled Graph). An *entangled graph* is a pair (\mathcal{G}, w) with $\mathcal{G} = (\mathcal{G}_1, \dots, \mathcal{G}_n)$ a n -tuple of graphs and $w \dots$. We call $\mathcal{G}_1, \dots, \mathcal{G}_n$ the *component graphs* of the entangled graph \mathcal{G} .

You can add a second component graph by selecting "*Extras → Add entangled graph*" from the menu. Should be "*Extras → Add another component graph*"?

For each graph a new tab in the graph panel and transition matrices panel is created (see figure 11 for example).

```
m <- rbind(H1=c(0, 0, 1, 0, 0),
           H2=c(0, 0, 1, 0, 0),
           H3=c(0, 0, 0, 0.9999, 1e-04),
           H4=c(0, 1, 0, 0, 0),
           H5=c(0, 0, 0, 0, 0))
weights <- c(1, 0, 0, 0, 0)
```

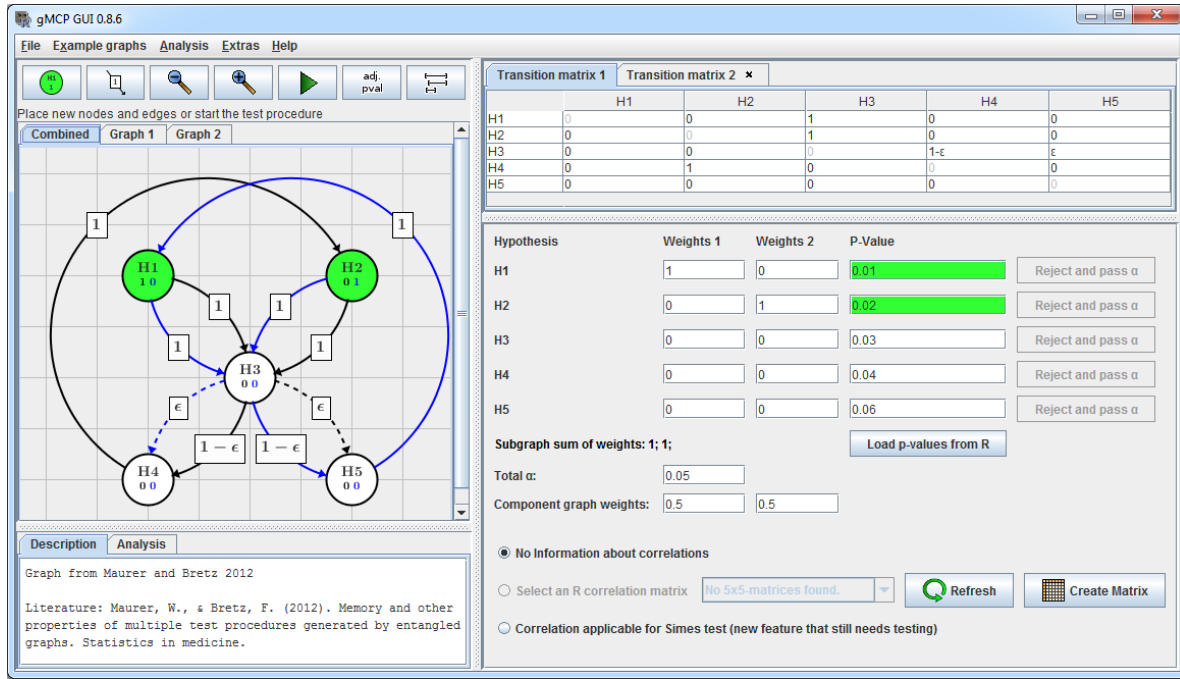



Figure 11: Different tabs show the different entangled graphs / transition matrices.

```
subgraph1 <- new("graphMCP", m=m, weights=weights)

m <- rbind(H1=c(0, 0, 1, 0, 0),
           H2=c(0, 0, 1, 0, 0),
           H3=c(0, 0, 0, 1e-04, 0.9999),
           H4=c(0, 0, 0, 0, 0),
           H5=c(1, 0, 0, 0, 0))
weights <- c(0, 1, 0, 0, 0)
subgraph2 <- new("graphMCP", m=m, weights=weights)

weights <- c(0.5, 0.5)
graph <- new("entangledMCP", subgraphs=list(subgraph1, subgraph2), weights=weights)
```

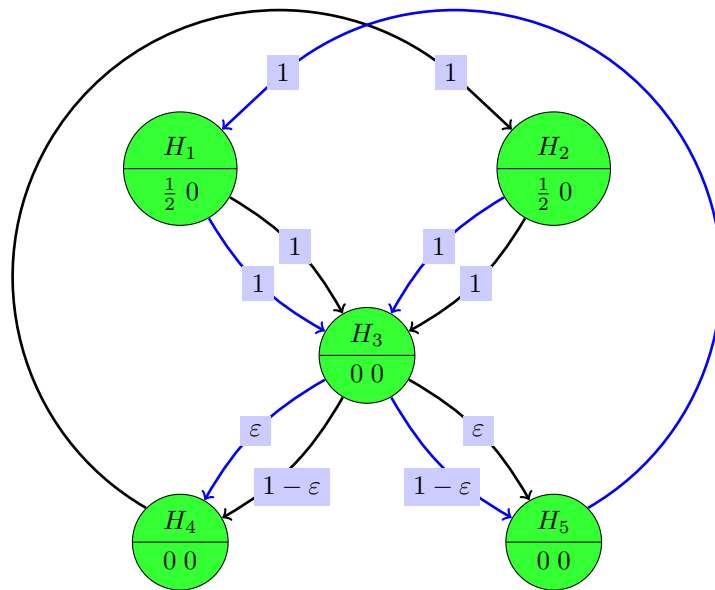


Figure 12: Entangled graph from Maurer et Bretz

7 Power Simulations

7.1 Using the R command line

```
# Bonferroni adjustment
G <- diag(2)
weights <- c(0.5,0.5)
corMat <- diag(2)+matrix(1,2,2)
theta <- c(1,2)
calcPower(weights, alpha=0.025, G, theta, corMat)

## [1] 1 2
##      [,1] [,2]
## [1,]    2    1
## [2,]    1    2
## [1] 10000
## $LocalPower
##      H1      H2
## 0.1898 0.4322
##
## $ExpRejections
## [1] 0.622
##
## $PowAt1st1
## [1] 0.4837
##
## $RejectAll
## [1] 0.1383

calcPower(weights, alpha=0.025, G, 2*theta, 2*corMat)

## [1] 2 4
##      [,1] [,2]
## [1,]    4    2
## [2,]    2    4
## [1] 10000
## $LocalPower
##      H1      H2
## 0.4520 0.8104
##
## $ExpRejections
## [1] 1.262
##
## $PowAt1st1
## [1] 0.8436
##
## $RejectAll
## [1] 0.4188
```

7.2 Variable edge weights

Apart from latin letters the following greek letters can be used to name a variable¹. Please enter them with a leading backslash so that they are recognized:

`\alpha`, `\beta`, `\gamma`, `\delta`, `\epsilon`, `\zeta`, `\eta`, `\theta`, `\iota`, `\kappa`, `\lambda`, `\mu`, `\nu`,
`\xi`, `\pi`, `\rho`, `\sigma`, `\tau`, `\upsilon`, `\phi`, `\chi`, `\psi`, `\omega`.

¹Note that omicron is not allowed since it can not be distinguished from the latin character "o".

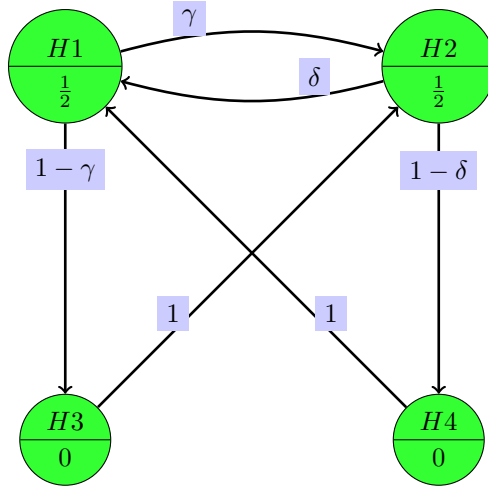


Figure 13: Graph from Bretz et al. (2009)

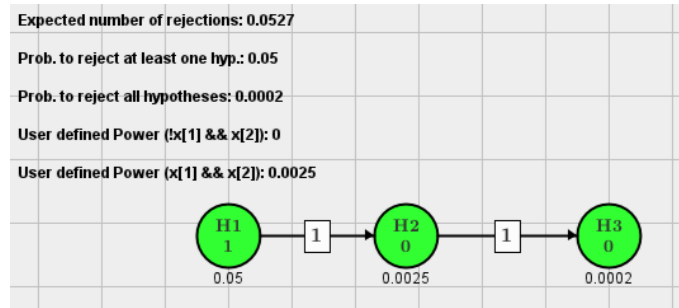


Figure 14: Local power and some (trivial) user defined power functions.

These are shown in the GUI as α , β , γ , δ , ε , ζ , η , θ , ι , κ , λ , μ , ν , ξ , π , ρ , σ , τ , v , φ , χ , ψ and ω .

	H1	H2	H3	H4
H1	0	γ	$1-\gamma$	0
H2	δ	0	0	$1-\delta$
H3	0	1	0	0
H4	1	0	0	0

```
graph <- generalSuccessive()
graph

## A graphMCP graph
## H1 (weight=0.5)
## H2 (weight=0.5)
## H3 (weight=0)
## H4 (weight=0)
## Edges:
## H1 -( \gamma )-> H2
## H1 -( 1-\gamma )-> H3
## H2 -( \delta )-> H1
## H2 -( 1-\delta )-> H4
## H3 -( 1 )-> H2
## H4 -( 1 )-> H1
```

8 Options and Import/Export

8.1 Options

Grid For easier placement of nodes a grid can be used that aligns the nodes to its intersections. You can specify a positive integer that sets the grid size, i.e. the width in pixels between two proximate parallel lines. If you set the grid size to 1 this would allow unrestricted placement and therefore disables the grid.

Number of digits Number of digits to be shown at various places. In this version not every part of the GUI will use this value, but this will improve in further versions.

Line width Especially if you want to use exported PNG graphics in other documents, you may want to adjust the line width of edges and nodes, when borders look to thin or thick.

Font Size Font size of the text in the GUI widgets.

Look'n'Feel The way the widgets of a GUI look and how they behave is called "look and feel" in Java. Depending on your operating system and classpath several Look'n'Feel implementations may be available (e.g. Metal (Java default), Windows, Mac OS, Motif and/or System/GTK). If you are used to a particular Look'n'Feel, you can select it here. But if you have problems with the graphical interface, please try to use the default Metal theme to check whether it could be a problem with the selected Look'n'Feel.

Colored image files and pdf reports Colors are used to highlight different conditions in the graph like hypotheses that could be rejected. While these colors are helpful in the GUI, you perhaps prefer black and white PNG image files and PDF reports.

Show rejected nodes in GUI When using the GUI to for stepwise rejection of hypotheses, this options determines whether rejected nodes should "disappear" or whether they remain on the screen and are only marked as rejected.

Use JLaTeXMath There are not many reasons not to use the free Java library JLaTeXMath to render numbers, symbols and formulas in the GUI. The option is mainly provided in case that errors occur displaying the numbers and formulas.

Show epsilon edges as dashed lines You can set whether epsilon edges should been shown as dashed or solid lines.

Show fractions instead of decimal numbers Floating point numbers are used for all calculations and values like $1/3$ would be normally shown as 0.333333. When this option is active the method fractions from package MASS is used to display fractions whenever the floating point numbers are close to a fraction that looks right.

Use epsilon approximation In this version this value can not be changed. No calculations with infinitesimal small values are done but instead the epsilon is approximated by a small real number.

Epsilon The small real value that should be used to approximate the infinitesimal small epsilon. Default is 10^{-3} .

Verbose output of algorithms If checked the selected the algorithms produce a verbose output that is shown in the GUI. For example the Simes test specifies for each intersection of elementar hypotheses whether and why it could be rejected.

Monte Carlo sample size for power The Monte Carlo sample size for power calculations. Default is 10000.

Type of random numbers You can select quasirandom or pseudorandom numbers for power calculations. The quasirandom option uses a randomized Lattice rule, and should be more efficient than the pseudorandom option that uses ordinary (pseudo) random numbers.

Check online for updates On start-up gMCP can check automatically whether a new version of gMCP is available. Only your version of R (like 2.13.1), the version of gMCP (like 0.7-5) and a random number (to distinguish different requests) are transmitted.

Weights of subgraphs are upscaled to 1 If 'No' is selected then for each intersection of hypotheses (i.e. each subgraph) a weighted test is performed at the possibly reduced level alpha of $\text{sum}(w) * \alpha$, where $\text{sum}(w)$ is the sum of all node weights in this subset. If 'Yes' is selected all weights are upscaled, so that $\text{sum}(w) = 1$.

Export images with transparent background If checked the background of exported PNG graphics will be transparent. Otherwise the graphs are displayed on a white background.

If a node is dragged also all edges to this node follow If selected the edges will always repositioned whenever a node is dragged. Otherwise only newly added edges behave that way and edges that have been dragged themselves are considered "anchored" and will stay with the edge weight label at the same position.

Automatically enter the editing mode, whenever a table cell gets the focus

Enable highly experimental feature The gMCP GUI often contains new features that are not that well tested. If you want to use or take a look at them, activate this option. But be prepared that things might go wrong.

Show R code to reproduce results in R After performing a test in the GUI the result dialog does not only show the pure results, but also R code to reproduce these results in R. If you are not interested in this feature you can disable it here.

8.1.1 Privacy

The GUI always asks before sending data to our server. It will do so to

1. check whether a new version of gMCP exists,
2. submit/load a graph to the online archive
3. send bug reports if an error occurs.

Only in the last case of a bug report some information about your computer is collected that can be reviewed by the user before sending the bug report. If you do not agree with sending this data, simply don't send a problematic bug report or if you never want to send bug reports, disable the option in the options menu.

8.2 Import/Exports

This subsection is work in progress, but fortunately the menu entries in figure 15 should be fairly self-explanatory.

You can export graphs to png files. The background of these png files will be made transparent, so that they will fit into whichever document you insert them. Note that some image viewers visualize transparency

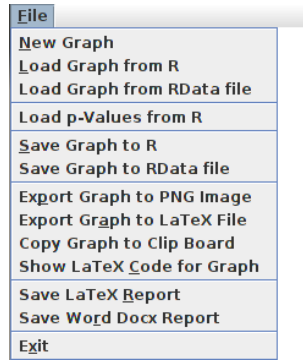


Figure 15: Import and export of graphs.

with a checkerboard pattern. In some Windows application the transparent background may be wrongly shown as black - if this happens or you generally prefer a non-transparent white background, you can toggle the option "*Export images with transparent background*" in the options dialog.

Since version 0.8-7 gMCP also can create word documents (Office Open XML / docx).

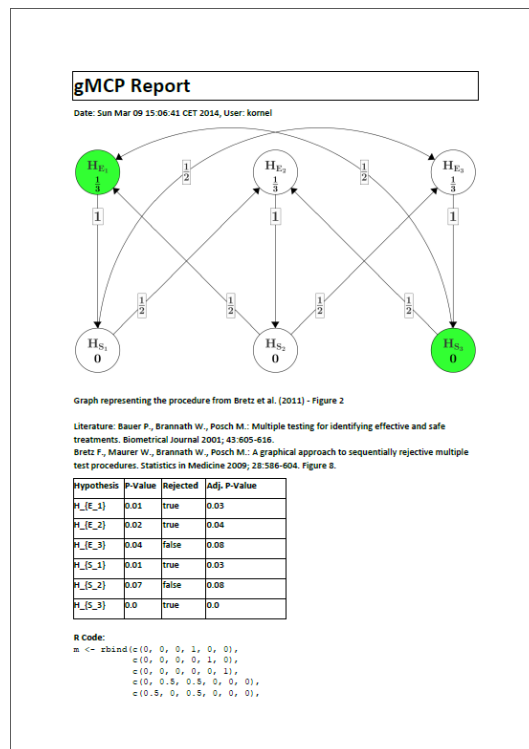


Figure 16: Example Word export - first page.

8.3 Important TikZ commands for optimizing the reports

A clear automatic placement of edges and weight labels without overlapping is a very difficult task and for complicated graphs the gMCP package will often fail to accomplish this automatically. If the result of `graph2latex` does not give you an acceptable layout, simply load the graph into the GUI and use the mouse to drag the edge labels around until you are satisfied with the placement. Save the graph and the TikZ output will be pretty close to the graph seen in the GUI.

PGF/TikZ is very useful L^AT_EX-package, we recommend it for many purposes and it's totally worth reading its 560 pages manual (Tantau [2008]), but if you don't have the time right now, we will give you short

overview of the most important commands for our kind of graphs so that you can easily adapt the output from `graph2latex`.

You also perhaps want to use an TikZ editor with a preview pane like `qtikz` or `ktikz`².

Let's start with this graph in figure 17:

```
\begin{tikzpicture}[scale=1]
\node (H11) at (200bp,200bp) [draw,circle split,fill=green!80] {$H11$ \nodepart{lower}  $\frac{8}{15}$ };
...
\draw [->,line width=1pt] (H11) to[bend left=15] node[near start,above,fill=blue!20] {0.667} (H12);
...
\end{tikzpicture}
```

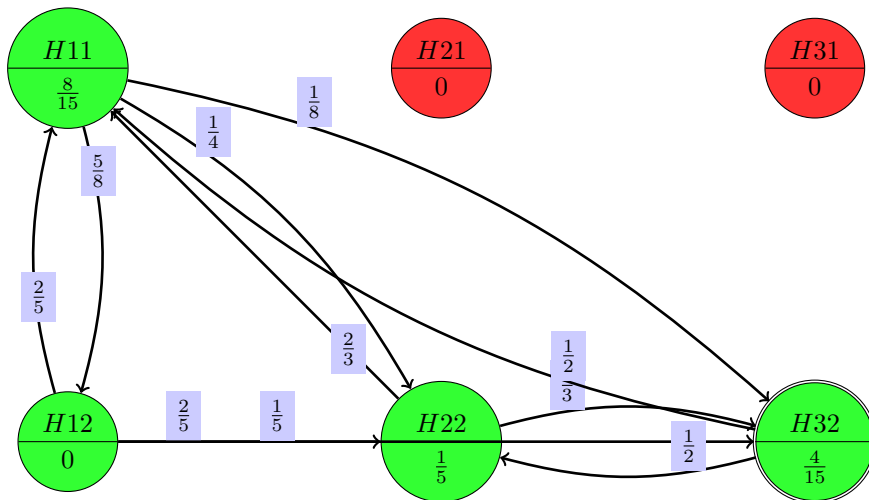


Figure 17: Graph from `graph2latex` that does not look optimal.

You can scale the TikZ graphic by changing the `[scale=1]` option. By default `graph2latex` doesn't scale TikZ graphics, but has an optional parameter `scale`.

For an explanation what `green!80` means and how you can specify other colors, please take a look at the `xcolor` manual from Kern [2007].

You can choose between the following label positions `above`, `below`, `right`, `left`, `above right`, `above left`, `below right`, and `below left`. In addition these positions can take an optional dimension argument, so that for example `below=1pt` can be used to place a label below and additionally shift it 1pt downwards.

You can change the position where the edge weight label is placed to `at start`, `very near start`, `near start`, `midway`, `near end`, `very near end` and `at end` or simply use something like `pos=0.5`. If you add an argument `sloped`, the text label will be rotated so that a parallel line to the base line becomes a tangent to the edge.

Often it is useful to reduce the bending angle in `[bend left=15]` below 15. You could also specify and change `out=15` and `in=165` separately.

A powerful feature is the use of styles, since this will effect all objects of a given class. But for this please take a look directly at the TikZ manual (Tantau [2008]).

9 Case Studies

This section is work in progress.

²<http://www.hackenberger.at/blog/ktikz-editor-for-the-tikz-language/>

9.1 Identifying effective and/or safe doses by stepwise confidence intervals for ratios

In this subsection we show how to use gMCP to reproduce the results of the paper from Bretz et al. [2003] with the same title.

9.2 Testing strategies in multi-dose experiments including active control

Bauer et al. [1998]

```
data(hydroquinone)
pvalues <- c()
x <- hydroquinone$micronuclei[hydroquinone$group == "C-"]
for (dose in c("30 mg/kg", "50 mg/kg", "75 mg/kg", "100 mg/kg", "C+")) {
  y <- hydroquinone$micronuclei[hydroquinone$group == dose]
  result <- wilcox.test(x, y, alternative = "less", correct = TRUE)
  pvalues <- c(result$p.value, pvalues)
}
pvalues

## [1] 0.004929 0.002634 0.002634 0.004319 0.066255

library(coin, quietly = TRUE)
pvalues <- c()
for (dose in c("30 mg/kg", "50 mg/kg", "75 mg/kg", "100 mg/kg", "C+")) {
  subdata <- droplevels(hydroquinone[hydroquinone$group %in% c("C-", dose), ])
  result <- wilcox_test(micronuclei ~ group, data = subdata, distribution = "exact")
  pvalues <- c(pvalue(result), pvalues)
}
pvalues

## [1] 0.006061 0.001263 0.001263 0.005051 0.135101
```

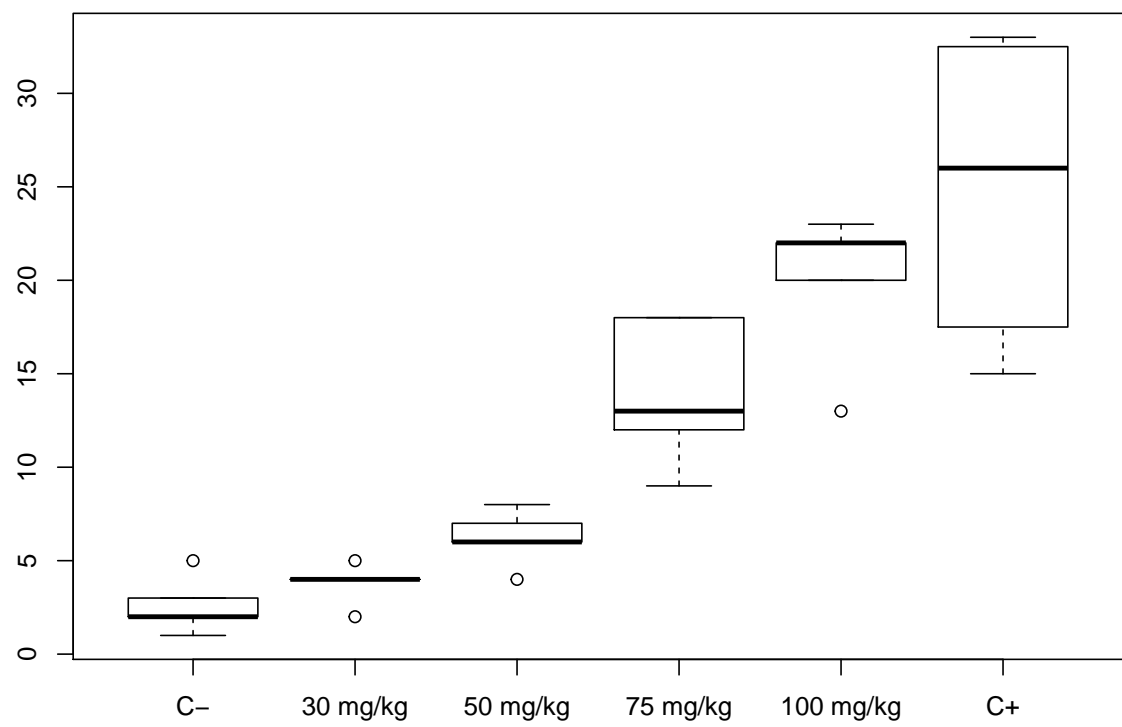
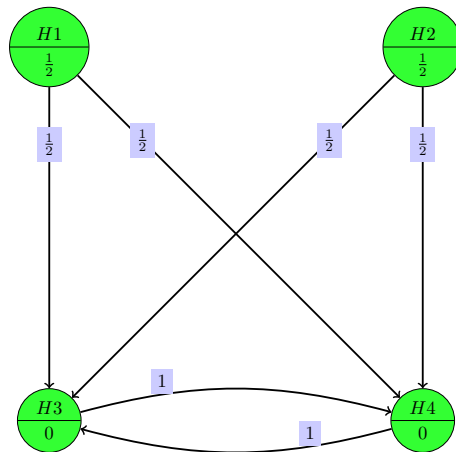
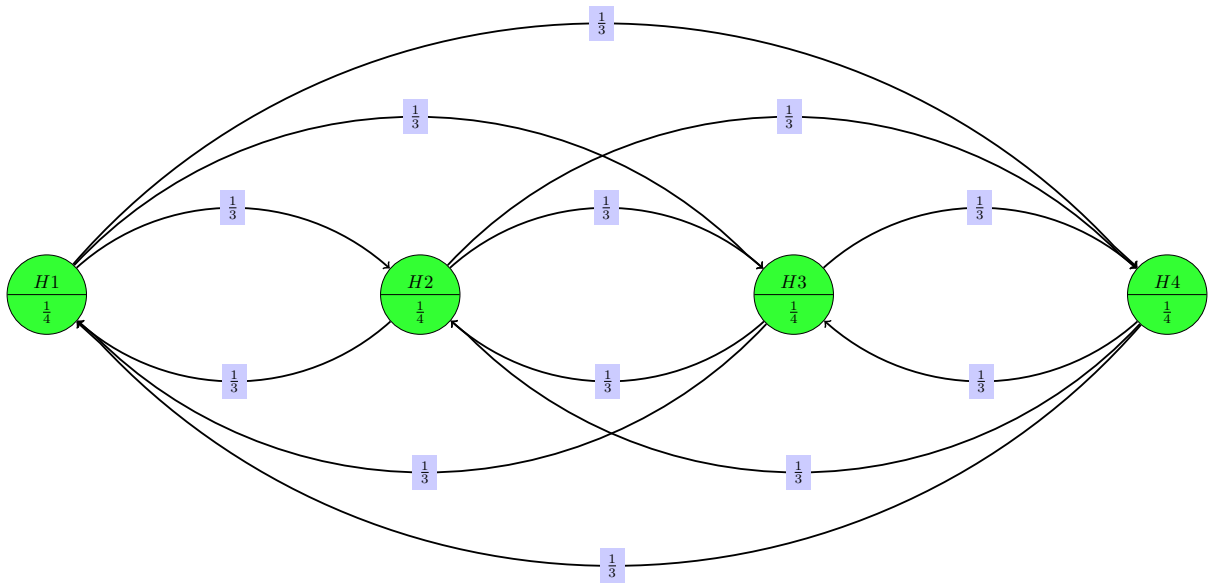
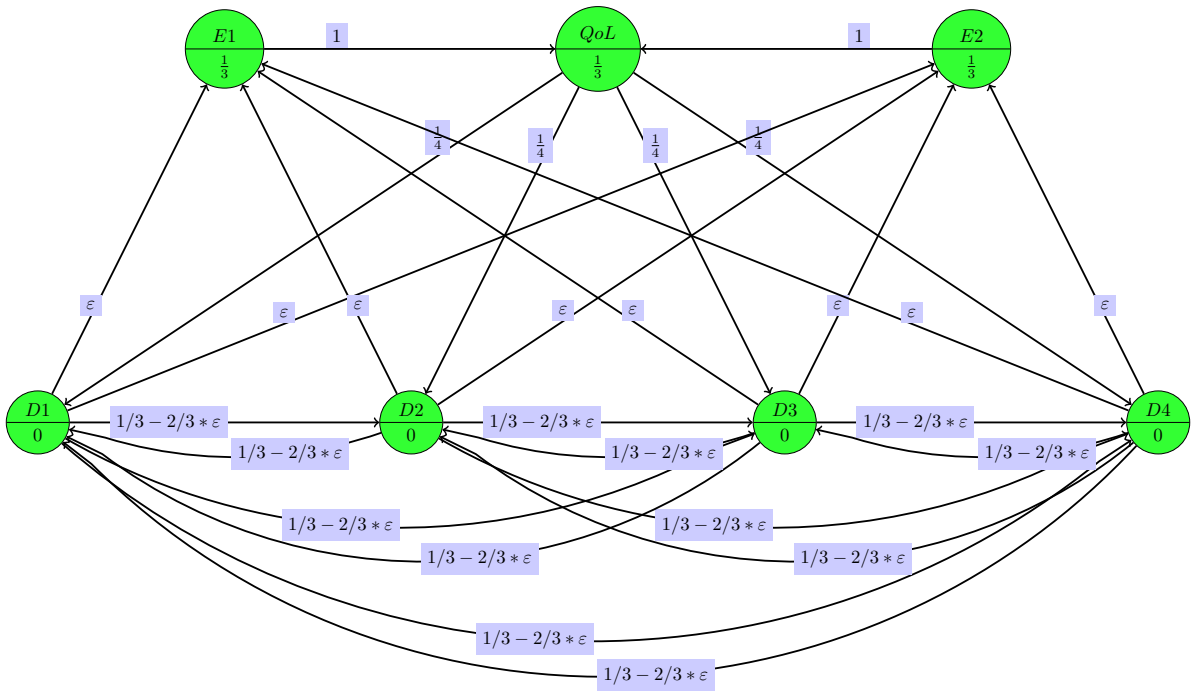
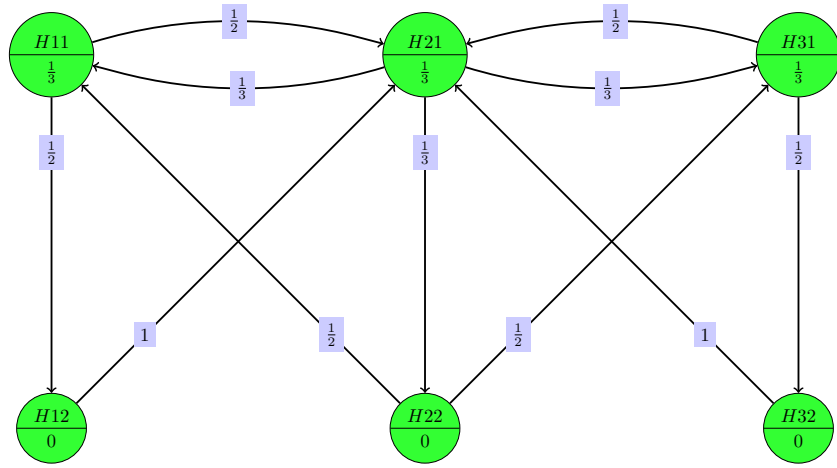
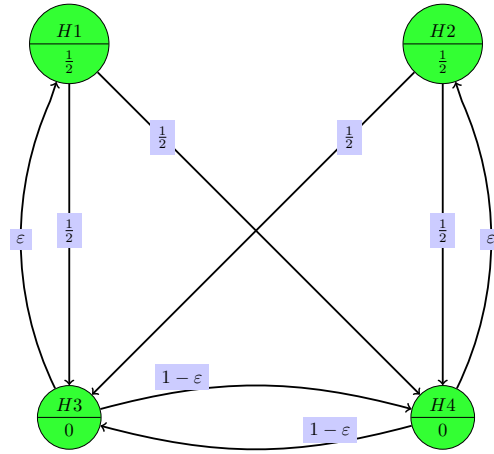
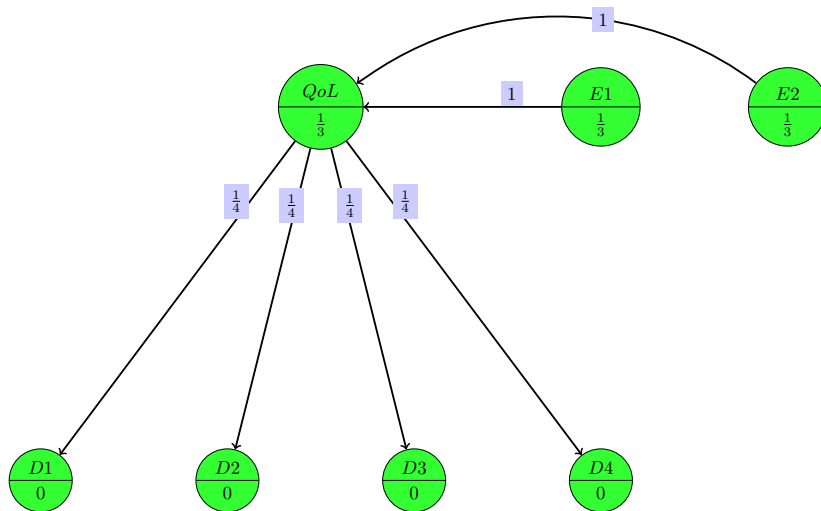



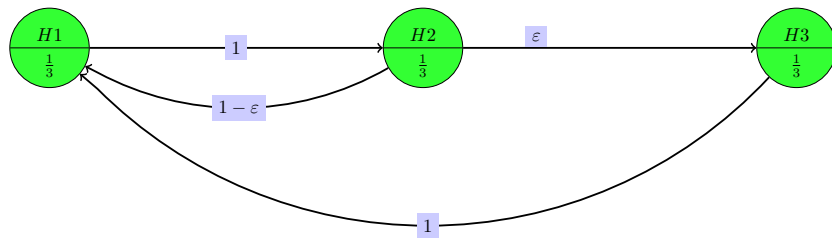
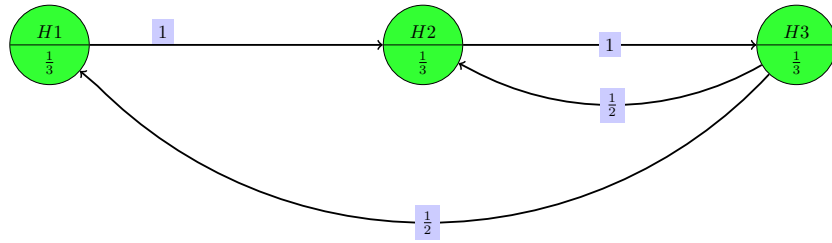
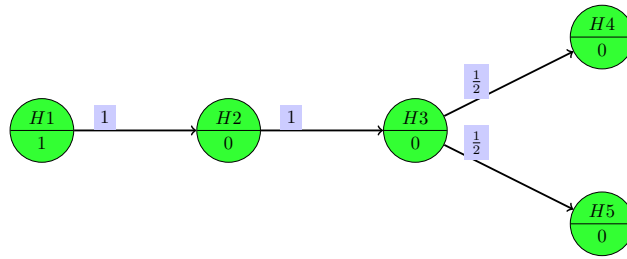
Figure 18: Boxplot of the hydroquinone data set

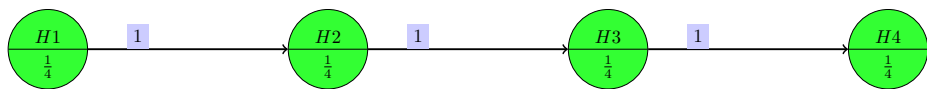
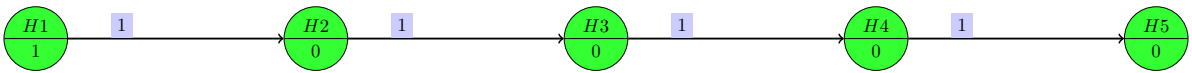
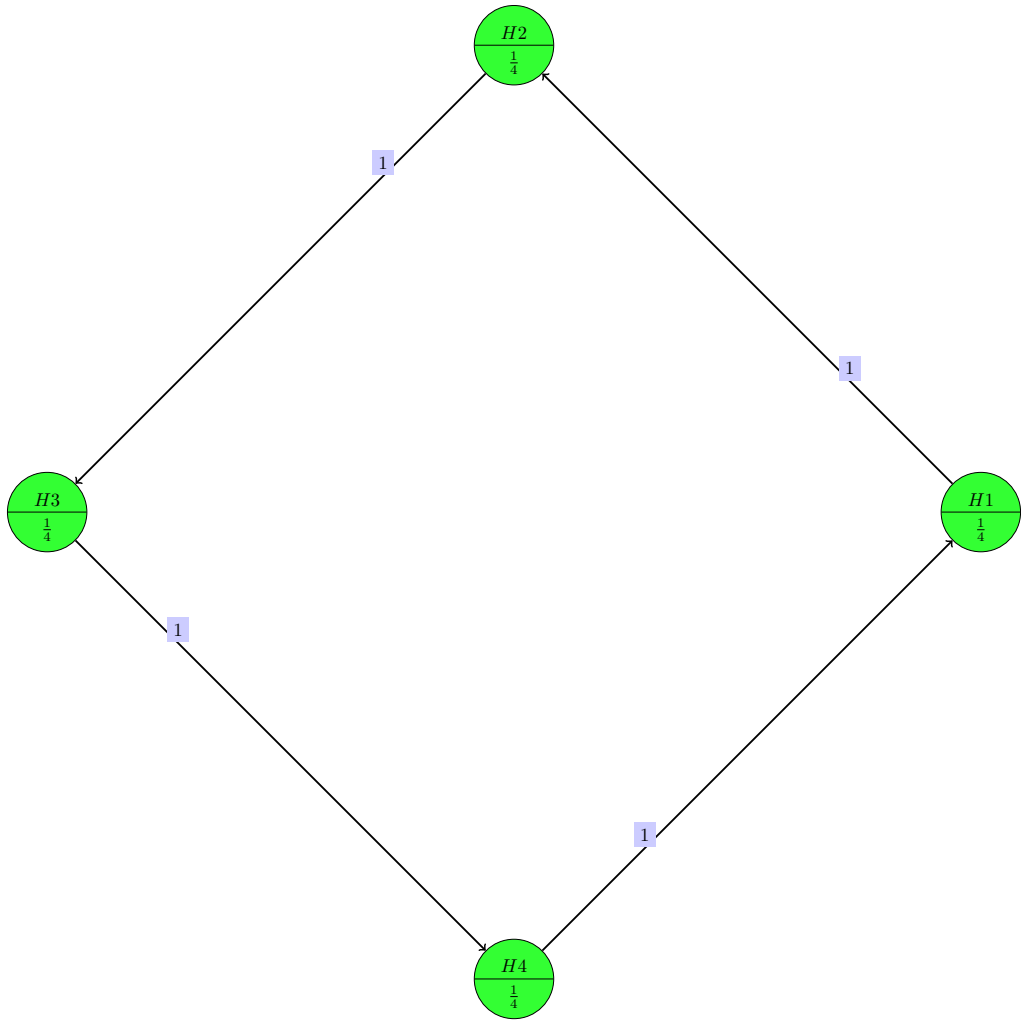
A Appendix - Example graphs

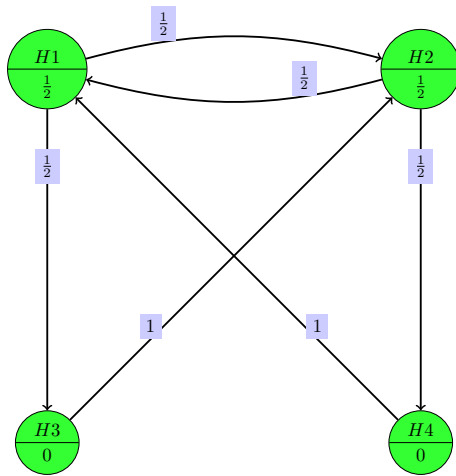
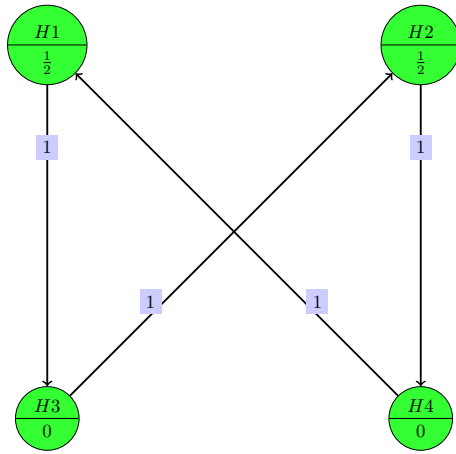


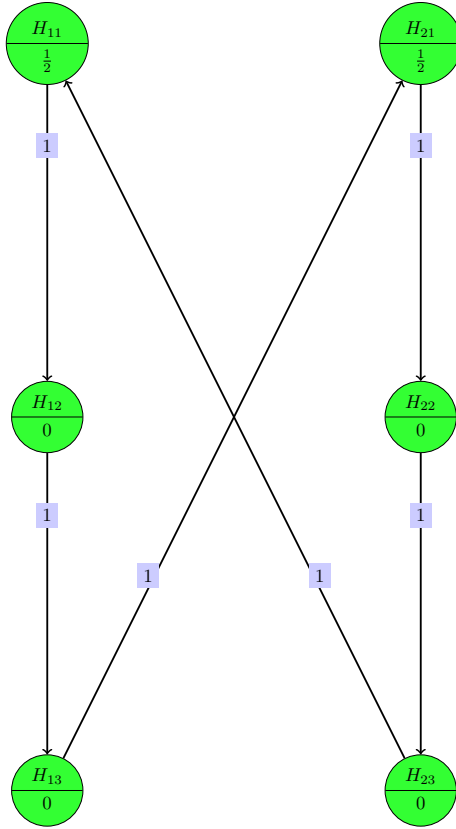
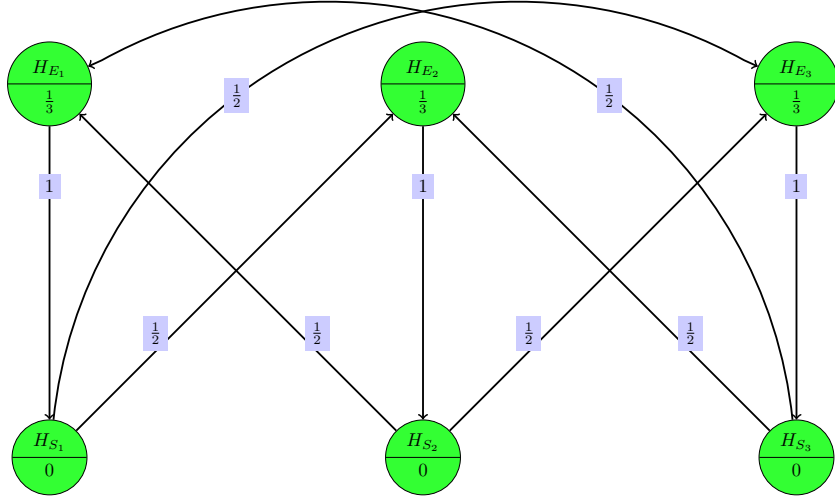


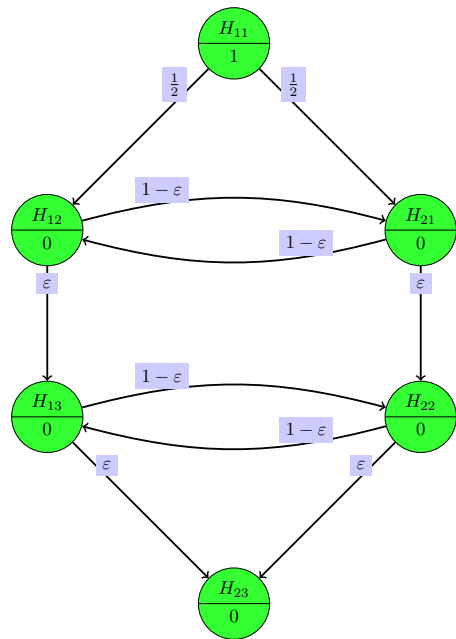
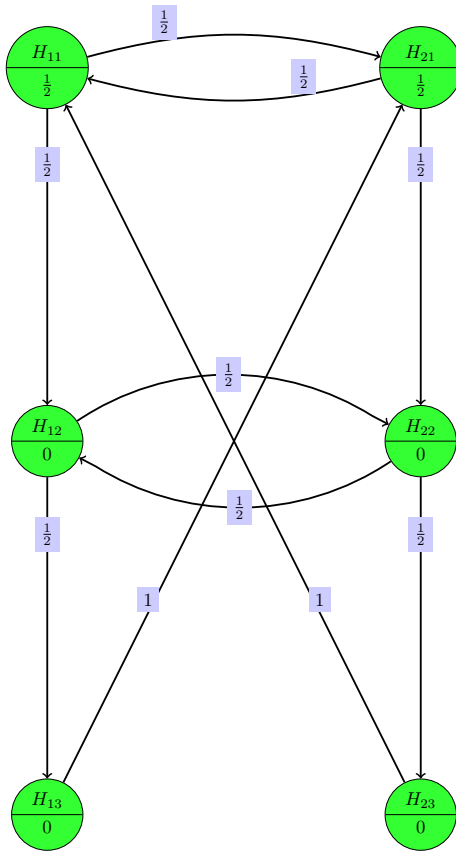


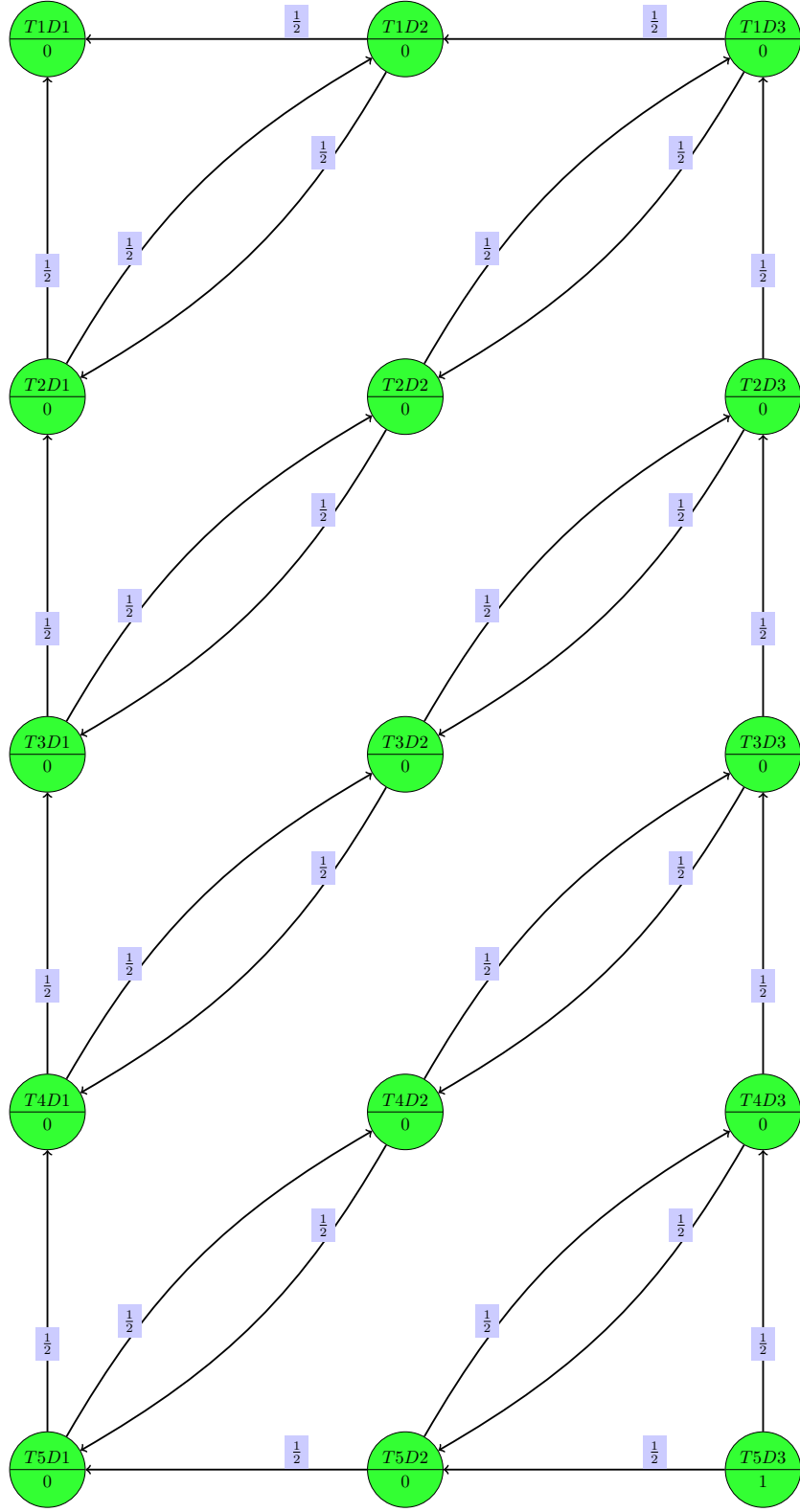












B Appendix - Multiple Testing Basics

Let Θ be a parameter space indexing a family of probabilities $\{P_\theta \mid \theta \in \Theta\}$ and $(\Omega, \mathcal{F}, P_\theta)$ the associated probability spaces. For a family of null hypotheses $H_i \subset \Theta$, $i \in \{1, \dots, n\} =: I$ a multiple test procedure φ is defined as a family of $(\mathcal{F}, \mathfrak{Pot}(\{0, 1\}^n))$ -measurable functions $\{\varphi_J : \Omega \rightarrow \{0, 1\}^n \mid J \subset I\}$. (We'll write φ_j for $\varphi_{\{j\}}$).

The family of hypotheses $\{H_i \mid i \in I\}$ is called *closed* if it is closed under intersection.

Definition B.1 (Familywise Error Rate). Let $H_J := \bigcap_{j \in J} H_j$. The multiple test procedure φ controls the *familywise error rate at level α in the weak sense* if

$$\forall \theta \in H_I : P_\theta(\varphi_J = 1 \text{ for some } J \subset I) \leq \alpha.$$

The multiple test procedure φ controls the *familywise error rate at level α in the strong sense* if

$$\forall \theta \in \Theta : P_\theta \left(\max_{J \subset I, \theta \in H_J} \varphi_J = 1 \right) \leq \alpha.$$

This section is work in progress.

Theorem B.2 (Closed testing principle). *Marcus et al. [1976]*

Definition B.3 (Coherence and Consonance). A multiple test procedure is called *consonant* if

$$\forall J \subset I : (\varphi_J = 1 \Rightarrow \exists j \in J : \varphi_j = 1).$$

A multiple test procedure is called *coherent* if

$$\forall J, J' \subset I : (\varphi_J = 0 \text{ and } J' \subset J \Rightarrow \varphi_{J'} = 0).$$

For further reading see Hochberg and Tamhane [2009] and Gabriel [1969].

Definition B.4.

Theorem B.5 (Simes-Procedure). *Let T_1, \dots, T_m be test statistics for $m \in \mathbb{N}$ null hypotheses H_1, \dots, H_m and p_1, \dots, p_m the associated p-values and $\alpha \in]0, 1[$.*

Denote the ordered p-values by $p^{(1)} < p^{(2)} < \dots < p^{(m)}$ and the corresponding hypotheses by $H^{(1)}, H^{(2)}, \dots, H^{(m)}$.

Reject H_0 if

$$p^{(j)} \leq \frac{j\alpha}{n} \quad \text{for some } 1 \leq j \leq m.$$

For independent tests the FWER is controlled at level α .

Theorem B.6 (Weighted Simes Procedure). *Benjamini and Hochberg (1997)*

Let $\sum_{k=1}^m w_k = m$ and reject H_0 if

$$p^{(j)} \leq \frac{\sum_{k=1}^j w_{(k)}}{m} \cdot \alpha \quad \text{for some } 1 \leq j \leq m.$$

Theorem B.7 (Hommel Procedure). *Hommel 1988*

Theorem B.8 (Hochberg Procedure). *Hochberg 1988*

Adjusted p-values from the Hochberg Procedure are greater as or equal to the Hommel-adjusted p-values.

Adjusted p-Values in the Simes Test

For each set $J \subset I$ we calculate

$$m_J := \min_{j \in J} \left(\frac{p_j}{\sum_{i \in J_j} w_i} \right), \quad J_j = \{k \in J \mid p_k \leq p_j\}.$$

The weighted Simes Test rejects H_J iff $m_J \leq \alpha$.

In a closed testing procedure a hypothesis H_j is rejected iff H_J is rejected for each $J \subset I$ with $j \in J$.

An adjusted p-value p'_j is defined as the minimal α so that the test to global level α rejects H_j .

Therefore $p'_j = \max(m_J \mid j \in J)$.

C Appendix - Graph Theory Basics

When we talk about graphs in the context of gMCP we always mean finite, directed, weighted graphs with no self-loops and no parallel edges:

Definition C.1. In our context a (valid) *graph* G is a triple $G = (V, E, w)$ of a non-empty, finite set V of nodes together with the set of edges $E \subset (V \times V) \setminus \{(v, v) \mid v \in V\}$ and a mapping $w : V \cup E \rightarrow [0, 1]$ that fullfills $\sum_{v \in V} w(v) \leq 1$ and $w(e) > 0$ for each edge $e \in E$.

Isomorphisms.

This section is work in progress.

Index

adjusted p-values, 12

Bonferroni-Holm-Procedure, 5

closed family, 35

closed testing principle, 35

coherence, 35

consonance, 35

coordinates, 8

correlation matrix, 14

edge weights

variable, 18

entangled graphs, 16

epsilon edges, 14

export, 21

familywise error rate, 35

gatekeeping

improved parallel, 15

parallel, 15

graph2latex, 22

graphs

component, 16

entangled, 16

Hochberg procedure, 35

Hommel Procedure, 35

import, 21

matrix2graph, 7

options, 20

parallel gatekeeping, 15

parametric test, 13

power simulation, 18

report generation, 12

setWeights, 7

Simes-Procedure, 13, 35

simultaneous confidence intervals, 12

TikZ, 8, 22

Weighted Simes Procedure, 35

List of Algorithms

1	Removing node i , passing the weight and updating the graph edges	4
---	---	---

List of Figures

1	Graph representing the Bonferroni-Holm-Procedure for three hypotheses.	5
2	Example showing how two null hypotheses can be rejected with p-values $p_1 = 0.01$, $p_2 = 0.07$ and $p_3 = 0.02$	6
3	Example graph from Bretz et al. [2011a] that we will create in this vignette.	7
4	The graphical user interface allows testing, calculation of confidence intervals and adjusted p-values.	9
5	R Code generated and shown by the GUI to reproduce the results.	10
6	Final graph from the test procedure after rejection of H_{21} , H_{31} and H_{32}	12
7	For normal and t-distributions simultaneous CI can be calculated by the GUI.	13
8	You can also specify a correlation between the tests.	13
9	Dialog for specifying a correlation matrix.	14
10	The Parallel Gatekeeping and the Improved Parallel Gatekeeping Procedure.	15
11	Different tabs show the different entangled graphs / transition matrices.	17
12	Entangled graph from Maurer et Bretz	17
13	Graph from Bretz et al. (2009)	19
14	Local power and some (trivial) user defined power functions.	19
15	Import and export of graphs.	22
16	Example Word export - first page.	22
17	Graph from <code>graph2latex</code> that does not look optimal.	23
18	Boxplot of the hydroquinone data set	25

List of Tables

References

- P. Bauer, J. Röhm, W. Maurer, and L. Hothorn. Testing strategies in multi-dose experiments including active control. *Statistics in Medicine*, 17(18):2133–2146, 1998. ISSN 1097-0258.
- F. Bretz, L.A. Hothorn, and J.C. Hsu. Identifying effective and/or safe doses by stepwise confidence intervals for ratios. *Statistics in medicine*, 22(6):847–858, 2003. ISSN 1097-0258.
- F. Bretz, W. Maurer, W. Brannath, and M. Posch. A graphical approach to sequentially rejective multiple test procedures. *Statistics in medicine*, 28(4):586–604, 2009. URL www.meduniwien.ac.at/fwf_adaptive/papers/bretz_2009_22.pdf.
- F. Bretz, W. Maurer, and G. Hommel. Test and power considerations for multiple endpoint analyses using sequentially rejective graphical procedures. *Statistics in medicine*, 30(13):1489–1501, 2011a.
- F. Bretz, M. Posch, E. Glimm, F. Klinglmueller, W. Maurer, and K. Rohmeyer. Graphical approaches for multiple comparison problems using weighted bonferroni, simes or parametric tests. *Biometrical Journal*, 53(6):894–913, 2011b. URL <http://onlinelibrary.wiley.com/doi/10.1002/bimj.201000239/full>.
- K.R. Gabriel. Simultaneous test procedures—some theory of multiple comparisons. *The Annals of Mathematical Statistics*, 40(1):224–250, 1969.
- Y. Hochberg and A.C. Tamhane. *Multiple Comparison Procedures*. Wiley Series in Probability and Statistics. John Wiley & Sons, 2009. ISBN 9780470568330. URL <http://books.google.de/books?id=XC5IPwAACAAJ>.
- S. Holm. A simple sequentially rejective multiple test procedure. *Scand. J. Statist.*, 6:65–70, 1979.
- U. Kern. *Extending LaTeX’s color facilities: the xcolor package*, 2007. URL <http://www.ctan.org/tex-archive/macros/latex/contrib/xcolor/>.
- R. Marcus, P. Eric, and K.R. Gabriel. On closed testing procedures with special reference to ordered analysis of variance. *Biometrika*, 63(3):655, 1976. ISSN 0006-3444.
- W. Maurer and F. Bretz. Memory and other properties of multiple test procedures generated by entangled graphs. *Statistics in medicine*, 32(10):1739–1753, 2013.
- R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. URL <http://www.R-project.org/>.
- T. Tantau. *The Tik Z and PGF Packages Manual for version 2.00*, 2008. URL <http://www.ctan.org/tex-archive/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf>.

Table of Symbols

Sets

\mathbb{R}	set of real numbers
\mathbb{N}_0	set of natural numbers (including 0)
$\mathfrak{P}\text{ot}(X)$	power set of set X , i.e. the set of all subsets of X

Functions

$\langle \cdot, \cdot \rangle$	standard direct product $\langle x, y \rangle = \sum_{j=1}^n x_j \cdot y_j$ for $x, y \in \mathbb{R}^n$
id_X	identity on X , i.e. $\text{id}_X : X \rightarrow X, x \mapsto x$