Non-technical introduction to multiple testing

The μTOSS project

Examples

High-level description

Simulation tool

μTOSS - Multiple hypotheses testing in an open software system

Gilles Blanchard\textsuperscript{2}, Thorsten Dickhaus\textsuperscript{1}, Niklas Hack\textsuperscript{3}, Frank Konietschke\textsuperscript{4}, Kornelius Rohmeyer\textsuperscript{5}, Jonathan Rosenblatt\textsuperscript{6}, Marsel Scheer\textsuperscript{7}, Wiebke Werft\textsuperscript{8}

\textsuperscript{1}Berlin Institute of Technology \textsuperscript{2}Weierstrass Institute for Applied Analysis and Stochastics Berlin \textsuperscript{3}Medical University of Vienna \textsuperscript{4}Georg-August-University Göttingen \textsuperscript{5}Leibniz University Hannover \textsuperscript{6}Tel Aviv University \textsuperscript{7}German Diabetes Center Düsseldorf \textsuperscript{8}German Cancer Research Center Heidelberg

WAPA 2010, Cumberland Lodge, 01-02.09.2010
Overview

Non-technical introduction to multiple testing

The $\mu$TOSS project

Workflow examples

High-level description

Simulation tool
## The $\mu$TOSS project

<table>
<thead>
<tr>
<th>Format</th>
<th>Harvest Programme</th>
</tr>
</thead>
</table>
| **Coordinators** | Thorsten Dickhaus, TU Berlin  
Gilles Blanchard, WIAS Berlin  
Klaus-Robert Müller, TU Berlin |
| **Coding team** | Niklas Hack, Vienna  
Frank Konietschke, Göttingen  
Kornelius Rohmeyer, Hannover  
Jonathan Rosenblatt, Tel Aviv  
Marsel Scheer, Düsseldorf  
Wiebke Werft, Heidelberg |
| **Hosting site** | Berlin Institute of Technology, Machine Learning Department |
Statistical (single) testing

- **goal**: check whether observed data is compatible with some assumption $H$ (null hypothesis) on its distribution $P$.

- **example**: measure a relevant quantity before $(X_b)$ and after $(X_a)$ some treatment (paired sample test) Null hypothesis: $E(X_a - X_b) = 0$ (no effect of treatment)

- **generic template for a testing procedure**:
  
  Data $\rightarrow$ Scalar quantity $S$ $\rightarrow$ If $S > t^*$ declare $H$ false

- **Rejecting** $H$ is also often called a discovery.

- **Threshold** $t^*$ is chosen such that the type I error control holds

  $$P(H \text{ rejected while true}) \leq \alpha,$$

  where $\alpha$ is a user-specified level.
Statistical (single) testing

- goal: check whether observed data is compatible with some assumption $H$ (null hypothesis) on its distribution $P$.
- example: measure a relevant quantity before ($X_b$) and after ($X_a$) some treatment (paired sample test) Null hypothesis: $E(X_a - X_b) = 0$ (no effect of treatment)
- generic template for a testing procedure:
  \[
  \text{Data} \rightarrow \text{Scalar quantity } S \rightarrow \text{If } S > t^* \text{ declare } H \text{ false}
  \]
- Rejecting $H$ is also often called a discovery.
- Threshold $t^*$ is chosen such that the type I error control holds
  \[
P(H \text{ rejected while true}) \leq \alpha,
  \]
  where $\alpha$ is a user-specified level.
Statistical (single) testing

- goal: check whether observed data is compatible with some assumption $H$ (null hypothesis) on its distribution $P$.
- example: measure a relevant quantity before ($X_b$) and after ($X_a$) some treatment (paired sample test) Null hypothesis:
  $E(X_a - X_b) = 0$ (no effect of treatment)
- generic template for a testing procedure:
  \[
  \text{Data} \quad \rightarrow \quad \text{Scalar quantity } S \quad \rightarrow \quad \text{If } S > t^* \text{ declare } H \text{ false}
  \]
- Rejecting $H$ is also often called a discovery.
- Threshold $t^*$ is chosen such that the type I error control holds
  \[
P(H \text{ rejected while true}) \leq \alpha,
  \]
  where $\alpha$ is a user-specified level.
\( p \)-values

- \( p \)-values are a “normalizing” monotone transform of the original testing statistics.
- central object in statistical testing literature
- **defining property**: if hypothesis \( H \) is true, then the \( p \)-value \( p_H \) has a uniform distribution
- rejecting \( H \) iff \( p_H < \alpha \) has type I error control at level \( \alpha \)
Non-technical introduction to multiple testing

**Multiple testing**

- possibly very large set $\mathcal{H}$ of hypotheses to test simultaneously
- for each hypothesis $H$, a corresponding statistic $S_H$
- need to control the error at family level
- several existing error criteria:
  - Family-wise error rate (FWER): probability of one or more type I errors over the family
  - False discovery rate (FDR): average proportion of type I errors in the reported discoveries
- cannot simply reject hypotheses $H$ having $p_H < \alpha$!
- needed: compute a suitable rejection threshold, or alternatively compute *adjusted* $p$-values
- other goals: estimate the proportion of true null hypotheses; simultaneous confidence intervals if relevant
Non-technical introduction to multiple testing

The $\mu$TOSS project

Examples

High-level description

Simulation tool

Multiple testing workflow(s)

Data + model

Statistics

$p_1$

$p_2$

$p_K$

Adjusted p-values

Rejected hypotheses

$S_1$

$S_2$

$\cdots$

$S_K$

$\tilde{p}_1$

$\tilde{p}_2$

$\tilde{p}_K$
Goals and overview of \( \mu \)TOSS

1. A unified software basis for multiple comparison procedures (MCPs), for developers and end users
2. Source code-open implementation in R
3. Easy to add new procedures
4. Graphical user interface
5. Simulation tool for facilitating comparison of procedures
6. Online user’s guide on which procedure to use according to the user’s specification of the test problem
7. Inclusion of a large part of the known MCP methods
8. Ongoing maintenance
GUI Example


```r
> notterman <- read.table("CarcinomaNormalDataset CancerResearch.txt")
> dim(notterman)
[1] 7464  36
> notterman.classlabel <- c(rep(1,18), rep(0,18))
> library(mutossGUI)
> mutossGUI()
```
Non-technical introduction to multiple testing

µTOSS project

Examples

High-level description

Simulation tool

MuToss GUI

Data

Load R Object

Info

Model

Hypotheses

n/a

p-Values

Please load either data or p-values!

Error Rate

n/a

P0

n/a

Adjusted p-Values

n/a

Confidence Intervals

n/a

Rejected

n/a

Assumptions

Output

R Connection established
Non-technical introduction to multiple testing

µTOSS project

Examples

High-level description

Simulation tool

FILE

Please load either data or p-values!

Model

n/a

Hypotheses

n/a

p-Values

Please load either data or p-values!

Error Rate

n/a

P0

n/a

Adjusted p-Values

n/a

Confidence Intervals

n/a

Rejected

n/a

Choose R Object

Output

R Connection established

Assumptions
Non-technical introduction to multiple testing

- µTOSS project
- Examples
- High-level description
- Simulation tool
Paired sample test

The robust version uses the Wilcoxon test, otherwise a t-test will be performed.

The input for this paired sample model is a data matrix whose columns represent the samples and the rows represent the multiple endpoints. E.g. for genomics this would be a gene matrix, where each row gives the expression for a single gene.

Furthermore, a classlabel needs to be provided to distinguish the two paired groups. The arrangement of group indices does not matter, as long as the columns are arranged in the same corresponding order between groups. For example, if group 1 is code as 0 and group 2 is coded as 1, for 3 pairs of data, it does not matter if the classlabel is coded as (0,0,1,1,1)
Non-technical introduction to multiple testing

- µTOSS project
- Examples
- High-level description
- Simulation tool

Method: Paired sample test

Choose R Object

- amod
- grouplabels
- InsectSprays
- iris
- noterman
- noterman.classlabel
- noterman.pvalue
- pvalues
- X

Paired sample test

Choose classlabel

- noterman.classlabel

Example, if group 1 is coded as 0 and group 2 is coded as 1, for 3 pairs of data, it does not matter if the classlabel is coded as (0,0,0,1,1,1)
Pairing example: Paired sample test
Non-technical introduction to multiple testing

The TOSS project Examples High-level description Simulation tool

Method: Paired sample test

The robust version uses the Wilcoxon test, otherwise a paired t-test will be performed.

A vector of class labels needs to be provided to distinguish the two paired groups. The arrangement of group indices does not matter, as long as the columns are arranged in the same corresponding order between groups. For example, if group 1 is code as 0 and group 2 is coded as 1, for 3 pairs of data, it does not matter if the class label is coded as (0,0,0,1,1,1) or (1,1,0,0,0) or (0,1,0,1,0,1) or (1,0,1,0,1,0), the paired differences between groups will be calculated as group2 - group1.

You could either choose a valid R object to load as class labels or you could provide it
Non-technical introduction to multiple testing

The \( \mu \) TOSS project

Examples

High-level description

Simulation tool

R Connection established

**Paired sample test**

**Paired sample test**
Non-technical introduction to multiple testing

The TOSS project

Examples

High-level description

Simulation tool
Non-technical introduction to multiple testing

µTOSS project Examples

Simulation tool

![Diagram of ecdf graph with x-axis from 0.0 to 1.0 and y-axis from 0.0 to 1.0]
Non-technical introduction to multiple testing

- µTOSS project
- Examples
- High-level description
- Simulation tool

FWER

Family wise error rate alpha: 0.05
Non-technical introduction to multiple testing

- µTOSS project
- Examples
- High-level description
- Simulation tool

![Graphical User Interface (GUI) of MuToss]

- Data: RObj: noterman
- Model: Paired sample test
- Hypotheses: n/a
- p-Values: Paired sample test
- Error Rate: FWER (alpha=0.05)
- P0: n/a
- Adjusted p-Values: n/a
- Confidence Intervals
- Rejected
  - Info
    - Bonferroni correction
    - Hochberg (1988) step-up
    - Holm’s (1979) step-down
    - Hommel (1988) adjustment
    - Multiple Contrast Tests
    - Nonparametric Multiple contrast tests
    - Nonparametric relative contrast effects
    - Resampling-based F test
    - Resampling-based one sample test
    - Resampling-based paired sample test
    - Resampling-based two sample test
    - Ryan / Einot and Gabriel / Welsch test
    - Sidak correction
    - Sidak-like (1987) step-down
    - Tukey HSD Test

- Output: ecdf
- P-Value Plot

![Graphical representation of ecdf]

- x-axis: 0.0 to 1.0
- y-axis: 0.0 to 1.0

The graph shows the empirical cumulative distribution function (ecdf) with a comparison to a theoretical line.
Holm's step-down-procedure

Holm's step-down-procedure is applied to p-values. It controls the FWER in the strong sense under arbitrary dependency.

Holm's procedure uses the same critical values as the Hochberg's procedure, namely \( c(i) = \alpha / (m - i + 1) \), but is a step-down version while Hochberg's method is a step-up version of the Bonferroni test. Holm's method is based on the Bonferroni inequality and is valid regardless of the joint distribution of the test statistics, whereas Hochberg's method relies on the assumption that Simes' inequality holds for the joint null distribution of the test statistics. If this assumption is met, Hochberg's step-up procedure is more powerful than Holm's step-down procedure.

Reference:

Non-technical introduction to multiple testing

The µTOSS project

Examples

High-level description

Simulation tool
Non-technical introduction to multiple testing

The µTOSS project

Examples

High-level description

Simulation tool
Holm's (1979) step-down

Holm's (1979) step-down Procedure

Number of hyp.: 7464
Number of rej.: 113

<table>
<thead>
<tr>
<th>rejected</th>
<th>pValues</th>
<th>adjPValues</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7264</td>
<td>1.032963e-08</td>
</tr>
<tr>
<td>2</td>
<td>7248</td>
<td>1.467169e-08</td>
</tr>
<tr>
<td>3</td>
<td>106</td>
<td>3.124752e-08</td>
</tr>
<tr>
<td>4</td>
<td>7371</td>
<td>4.855137e-08</td>
</tr>
<tr>
<td>5</td>
<td>107</td>
<td>5.527837e-08</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>5.861399e-08</td>
</tr>
<tr>
<td>7</td>
<td>68</td>
<td>7.605817e-08</td>
</tr>
<tr>
<td>8</td>
<td>7299</td>
<td>9.480653e-08</td>
</tr>
<tr>
<td>9</td>
<td>7431</td>
<td>1.122460e-07</td>
</tr>
<tr>
<td>10</td>
<td>7451</td>
<td>1.468294e-07</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>1.682927e-07</td>
</tr>
<tr>
<td>12</td>
<td>8</td>
<td>1.717905e-07</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>1.789076e-07</td>
</tr>
<tr>
<td>14</td>
<td>7459</td>
<td>1.859696e-07</td>
</tr>
</tbody>
</table>
GUI Example 2: Keuls (Euphytica, 1952)

Goal: pairwise comparison of coordinate means
Non-technical introduction to multiple testing

µTOSS project

Examples

High-level description

Simulation tool
Non-technical introduction to multiple testing

Examples

High-level description

Simulation tool

Tukey HSD test and simultaneous confidence intervals in parametric factorial designs

With this function, it is possible to compute all pair comparisons for expectations and simultaneous confidence intervals in factorial linear models. Hereby, the all-pair comparisons can be performed for user given effects. The overall variance is estimated by the linear model as well as the degree of Freedom used by the studentized range distribution.

Reference:

Tukey HSD Test

```r
Tukey HSD Test

estimates  confintL  confintU  pvalues
Yar02-Yar01 -0.64,666667 -0.279918e+01  0.279918e+01  0.7849074e-05
Yar03-Yar01 -0.78,333333 -1.116667e+01  0.346667e+01  0.2899999e-07
Yar04-Yar01 -0.97,333333 -1.266667e+01  0.273333e+01  0.0000000e+00
Yar05-Yar01 -0.55,333333 -0.886667e+01  0.333333e+01  0.0000000e+00
Yar06-Yar01 -0.44,000000 -0.733333e+01  0.300000e+01  0.0000000e+00
Yar07-Yar01 -0.34,000000 -0.666667e+01  0.340000e+01  0.0000000e+00
Yar08-Yar01 -0.75,333333 -1.086667e+01  0.483333e+01  0.0000000e+00
Yar09-Yar01 -0.45,000000 -0.783333e+01  0.376667e+01  0.0000000e+00
Yar10-Yar01 -0.51,666667 -0.846667e+01  0.346667e+01  0.0000000e+00
Yar11-Yar01 -0.23,333333 -0.516667e+01  0.293333e+01  0.0000000e+00
Yar12-Yar01 -0.53,333333 -0.846667e+01  0.316667e+01  0.0000000e+00
Yar13-Yar01 -0.97,000000 -0.416667e+01  0.593333e+01  0.0000000e+00
Yar03-Yar02 -1.33,333333 -0.666667e+01  0.333333e+01  0.0000000e+00
Yar04-Yar02 0.17,333333 -0.599999e+01  0.846667e+01  0.0000000e+00
Yar05-Yar02 -0.83,333333 -0.126667e+01  0.383333e+01  0.0000000e+00
Yar06-Yar02 0.16,666667 -0.400000e+01  0.716667e+01  0.0000000e+00
Yar07-Yar02 0.30,333333 -0.299999e+01  0.603333e+01  0.0000000e+00
Yar08-Yar02 -0.10,666667 -0.399999e+01  0.196667e+01  0.0000000e+00
Yar09-Yar02 1.99,999999 -0.165556e+01  0.621667e+01  0.0000000e+00
Yar10-Yar02 0.59,000000 -0.203333e+01  0.463333e+01  0.0000000e+00
Yar11-Yar02 0.41,333333 -0.600000e+01  0.383333e+01  0.0000000e+00
Yar12-Yar02 0.44,333333 -0.600000e+01  0.383333e+01  0.0000000e+00
Yar13-Yar02 0.24,666667 -0.666667e+01  0.466667e+01  0.0000000e+00
Yar11-Yar03 0.33,000000 -0.219999e+01  0.199999e+01  0.0000000e+00
Yar12-Yar03 0.51,000000 -0.199999e+01  0.299999e+01  0.0000000e+00
Yar13-Yar03 0.31,000000 -0.219999e+01  0.199999e+01  0.0000000e+00
Yar05-Yar04 -0.80,000000 -0.433333e+01  0.333333e+01  0.0000000e+00
Yar06-Yar04 0.31,000000 -0.219999e+01  0.199999e+01  0.0000000e+00
Yar07-Yar04 0.13,000000 -0.219999e+01  0.269999e+01  0.0000000e+00
Yar08-Yar04 -0.28,000000 -0.616667e+01  0.346667e+01  0.0000000e+00
Yar09-Yar04 2.13,333333 -0.300000e+01  0.366667e+01  0.0000000e+00
Yar10-Yar04 -4.11,111111 -3.766667e+01  0.346667e+01  0.0000000e+00
Yar11-Yar04 -24.000000 -0.346667e+01  0.346667e+01  0.0000000e+00
Yar12-Yar04 2.20,000000 -0.133333e+01  0.276667e+01  0.0000000e+00
Yar13-Yar04 0.13,333333 -0.219999e+01  0.333333e+01  0.0000000e+00
Yar06-Yar05 0.51,111111  0.219999e+01  0.533333e+01  0.0000000e+00
Yar07-Yar05 -2.13,333333 -0.219999e+01  0.269999e+01  0.0000000e+00
Yar08-Yar05 -10.000000 -0.616667e+01  0.346667e+01  0.0000000e+00
Yar09-Yar05 1.01,111111  0.219999e+01  0.533333e+01  0.0000000e+00
Yar10-Yar05 2.66,666667 -0.966667e+01  0.366667e+01  0.0000000e+00
Yar11-Yar05 3.11,111111 -0.616667e+01  0.346667e+01  0.0000000e+00
```
Non-technical introduction to multiple testing

µTOSS project

Examples

High-level description

Simulation tool
Available Adjustment Methods

- Bonferroni correction
- Hochberg (1988) step-up
- Holm’s (1979) step-down
- Hommel (1988) adjustment
- Sidak correction
- Sidak-like (1987) step-down
Available Adjustment Methods

Non-technical introduction to multiple testing  The $\mu$TOSS project  Examples  High-level description  Simulation tool
Available Adjustment Methods

**MuToss GUI**

- **Data**: Please load either data or p-values!
- **Model**: n/a
- **Hypotheses**: n/a
- **p-Values**: RObj::pvalues
- **Error Rate**: FDR (alpha=0.05)
- **P0**: n/a
- **Adjusted p-Values**: n/a
- **Confidence Intervals**: n/a
- **Rejected**

**Assumptions**

- Asymptotically optimal rejection curve (2009)
- Benjamini-Liu (1999) step-down
- Benjamini-Yekutieli (2001) step-up
- Blanchard-Roquain 2-stage adaptive step-up (2009)
- Blanchard-Roquain adaptive step-up (2009)
- Local FDR (fdr)
μtoss standard specification

• at its core, toolbox of (command-line) functions
• following a fixed convention for input and output parameter names
• these functions can then be accessed:
  • as simple command-line R functions
  • by applying them on objects of class Mutoss through the wrapper function `mutoss.apply`
  • through the GUI.
The object class Mutoss

- the class Mutoss:
  > slotNames("Mutoss")
  
  [1] "data"       "model"       "description"  "statistic"
  [5] "hypotheses" "hypNames"    "criticalValues" "pValues"
  [9] "adjPValues" "errorControl" "rejected"     "qValues"
 [13] "locFDR"    "pi0"         "confIntervals" "commandHistory"

- used with the wrapper function mutoss.apply:

  (Mutoss object, standardized function) $\mapsto$ Mutoss object

- read/writes slots that are relevant for the function
- allows a consistent workflow and records internally command history
Adding new methods

Only need to write a function following the standard specification:

```r
> bonferroni <- function(pValues, alpha) {
    adjPValues = sapply(pValues*length(pValues), function(x) {min(x,1)})
    if (missing(alpha)) {
        return(list(adjPValues=adjPValues))
    } else {
        rejected <- (adjPValues<=alpha)
        return(list(adjPValues=adjPValues, rejected=rejected,
                     errorControl = new(Class='ErrorControl',
                                         type="FWER", alpha=alpha))
    }
}
```
Adding new methods

Define function metadata:

```r
> mutoss.bonferroni <- function() {
  return(new(Class="MutossMethod",
             label="Bonferroni correction",
             errorControl="FWER",
             callFunction="bonferroni",
             output=c("adjPValues", "rejected", "errorControl"),
             info="<h2>Bonferroni correction</h2>",
             parameters=list(pValues=list(type="numeric"),
                             alpha=list(type="numeric", optional=TRUE)))
}
```
Simulation tool (R command-line tool)

Composed of only two functions:

- `simulation(nreplications, DataGen, listOfProcedures)`:
  loops over data generation parameters / processing functions (’map’ step)

- `gatherStatistics(SimOutput, ProcSingle, ProcGroup)`:
  groups the result of the first step over identical parameters and computes some statistics (’reduce’ step)

- Output: data frame
Simulation tool (R command-line tool)

```r
> simulation
function(replications, DataGen, listOfProcedures){ ... }

> sim <- simulation(
   replications = 1000,
   DataGen = list(funName="MyDataGenFun",
                   fun=EquiCorrelatedPValues,
                   sampleSize=100, pi0=1.0,
                   rho=c(0.1, 0.9)),
   listOfProcedures =
     list(list(funName="Bonferroni",
                fun = bonferroni,
                alpha = c(0.05, 0.01))))

> print(length(sim))
[1] 4000
```
Simulation tool (R command-line tool)

> result <- gatherStatistics(
    listOfObjects = sim,
    listOfStatisticFunctions =
        list(V.ge.0 = V.greater.Zero),
    listOfAvgFunctions = list(MEAN = mean))

> print(result)

$statisticDF
   sampleSize method      rho alpha V.ge.0.MEAN
1        100  Bonferroni 0.1  0.05    0.049
2        100  Bonferroni 0.1  0.01    0.008
3        100  Bonferroni 0.9  0.05    0.052
4        100  Bonferroni 0.9  0.01    0.004
Installation

- **mutoss:**
  
  ```r
  install.packages("mutoss")
  
  Dependencies: multcomp, mvtnorm, plotrix, multtest, Biobase
  ```

- **mutossGUI:**
  
  ```r
  install.packages("mutossGUI")
  
  Dependencies: rJava, JavaGD
  ```

  To start the GUI in R:

  ```r
  > library(mutossGUI)
  > mutossGUI()
  ```
Links and references

• μTOSS R-Forge Website with released R-packages: http://mutoss.r-forge.r-project.org

• Official μTOSS Website: http://mutoss.eu

• Wishlist for requesting methods and data types: http://mutoss.eu/mutoss-wishlist.html