SDMinP - Program documentation

Documentation Version: 1.1
Date: 19.04.2005

Program Version: 1.0
Date : 16.03.2005
Purpose : Adjustment of the multiple type I error (FWER)
Programmer: Markus Obreiter
Contact: m.obreiter@dkfz-heidelberg.de
Institute: German Cancer Research Center DKFZ, Heidelberg, Germany

Contents

1 Abstract

Multiple testing requires the control of the multiple type I error rate. Ge *et al*. (2003) presented an improvement of the *Free Step-Down Resampling Method* for controlling the *Family wise error rate* (FWER), originally presented by Westfall and Young (1993). They reduced the re-sampling effort considerably, what made the method computationally feasible. Additionally, they optimized calculation steps and made the algorithm very appealing. Becker and Knapp (2004) presented an approach, in which the re-sampling effort is even more reduced.

SDMinP implements a variation of the algorithm of Ge *et al*. (2003) and Becker and Knapp (2004). It calculates adjusted p-values and returns additionally a global p-value for testing the overall result of the experiment. If the global p-value is significant, there is at least one significant result within the tested hypotheses. The program starts on top of a provided set of permutation test statistics per hypothesis. It does not perform the permutation procedure itself to be independent of the statistical tests, used in the analysis. See Obreiter *et al*. (2005) for more information about the theoretical background of the algorithms and methods, implemented in SDMinP.

2 Environment and Installation

2.1 Python

SDMinP is implemented in python, version 2.3.5, and needs the respective python environment. SDMinP does not use any operation system specific functions, what makes it independent from the operating system.

2.1.1 Installation on Windows

Here is a short guideline about how to install python on a windows operating system. First, download the python installation files, version 2.3.5 or any compatible version, from www.python.org. Run the installation. After the installation add the path to the executable python.exe file to the system or user variable 'PATH' of your system, e.g. "Path = old values; C:/Python23".

2.2 SDMinP installation

Unzip the downloaded file sdminp.zip into a destination directory of your choice. No further system specific steps are necessary.

2.3 Directory structure

SDMinP uses a defined directory structure to operate on. It is:

```
- SDMinP     (the home directory of SDMinP)
   - src     (contains the executable file and private library)
   - conf    (contains the configuration file with parameters)
   - data    (optional directory to store input data files)
   - result  (destination of result files)
   - temp    (destination of split files)
   - log     (destination of log files, if logging is enabled)
   - doc     (program documentation and GNU General Public License)
```

src

The src-directory contains the source files and the executable start file *'main.py'*.

conf

Contains the configuration file configuration.conf, which stores the parameters.

data

Can be used to store the data input files. An example input file comes with the downloaded package, see section 4.1.

result

Result file(s) are stored into this directory(see section 10).

temp

This directory is used to store temporary files, e.g. split files (see section 8).

log

If the logging mechanism is enabled in the configuration file, the log files are stored into the log directory. Each calculation step creates a specific log file. For details see section 9.

doc

Stores the program documentation and the GNU General Public License.

3 Program start

The program is started via command line.

1. Call a command line console of your system (e.g.: a Windows DOS box or a Unix shell).
2. Navigate to the source directory 'src'.
3. call the program by typing 'python main.py %input file path%'. The %input file path% is the path to the data file, which has to be processed, where python must calls the python environment of version 2.3.5 or any compatible version.

   E.g., to process the example file type: 'python main.py ../data/example_input.txt'

4 Input Format

The input for the program consists of a flat text file. Each line corresponds to one hypothesis. The data for each hypothesis consists of the following values:

1. Hypothesis identifier: a character string without spaces
2. The unadjusted raw p-value of the hypothesis or a 'NA' as a placeholder. If a hash is given, the raw p-value will be calculated by the program (see sections 4.3 and 5). Note, that you have to provide either only raw p-values or only hashes for all hypotheses. Mixing up hashes and raw p-values is not allowed.
3. The observed test statistic.
4. Permutation test statistics, divided by spaces or tabs. The same number of permutation test statistics has to be provided for each hypothesis.

Find here an example for one hypothesis with identifier m1, the pre-calculated raw p-value 0.01 of the observed test statistic with value 3.00 and 9 permutation test statistics:

```
1)  2)   3)    <---------------------- 4)---------------------->
m1  0.01 3.00  2.01  3.12  2.99  3.02  1.44  0.99 0.78 0.13  0.55
```

The values have to be separated by spaces or tabs.

Note

1. *Floating Point Numbers* have to be written with a point as a separator between full numbers and decimal digits. E.g., '5.123'.
2. Do not enter empty lines between hypotheses. The program includes hypotheses into the calculation set until the first empty line is found.

4.1 Example Input File

An example input file 'example_input.txt' is added to the download package. It is located in the 'data' folder. It contains example data for three tightly linked markers and 10 permutation replicates, presented in Becker and Knapp (2004). The data is provided in Format II (see below), i.e. without provided raw p-values for the observed test statistic.

4.2 Data Format I

Data Format I denotes the format where the raw p-value per hypothesis test is provided and based on preliminary calculations by the user (note: these raw p-values must not be calculated on basis of the provided permutation test statistics). The following example shows 7 hypotheses with provided raw p-values and 10 permutation test statistics. Here, the raw p-values were randomly determined as the example merely serves for demonstrating the format:

```
m1 0.01 3.00 2.01 3.12 2.99 3.02 1.44 0.99 0.78 0.13 0.55 0.42
m2 0.04 2.12 0.01 4.12 3.12 1.02 1.12 1.11 0.12 0.11 0.02 0.10
m3 0.08 2.00 2.02 2.12 3.22 2.22 1.99 0.74 0.75 1.00 1.88 1.07
m4 0.07 4.48 1.42 5.44 1.44 1.43 1.41 1.40 4.44 1.34 0.44 1.24
m5 0.09 2.44 0.00 0.44 4.44 5.44 0.03 0.04 0.34 1.44 3.44 0.24
m6 0.08 3.03 3.01 7.33 3.02 6.33 3.00 2.99 2.87 2.84 1.33 1.04
m7 0.05 6.47 4.47 9.47 8.47 5.47 0.02 0.37 0.47 1.47 2.47 3.47
```

4.3 Data Format II

Data Format II denotes the format where the raw p-value is not provided. Instead, the placeholder ’NA’ has to be set. The program will calculate it on basis of the provided permutation test statistics. In the following example the raw p-values were omitted and have to be determined by SDMinP (see section 5).

```
m1 NA 3.00  2.01  3.12  2.99  3.02  1.44  0.99  0.78 0.13 0.55 0.42
m2 NA 2.12  0.01  4.12  3.12  1.02  1.12  1.11  0.12 0.11 0.02 0.10
m3 NA 2.00  2.02  2.12  3.22  2.22  1.99  0.74  0.75 1.00 1.88 1.07
m4 NA 4.48  1.42  5.44  1.44  1.43  1.41  1.40  4.44 1.34 0.44 1.24
m5 NA 2.44  0.00  0.44  4.44  5.44  0.03  0.04  0.34 1.44 3.44 0.24
m6 NA 3.03  3.01  7.33  3.02  6.33  3.00  2.99  2.87 2.84 1.33 1.04
m7 NA 6.47  4.47  9.47  8.47  5.47  0.02  0.37  0.47 1.47 2.47 3.47
```

NOTE

Depending on the chosen data format, the formula for calculating raw p-values for permutation test statistics changes slightly. See section 5.

5 Data Format and unadjusted (raw) p-values

Depending on the provided data format within the input file (see section 4) the formula which calculates the permutation raw p-values changes slightly.

Data Format I

For data sets with provided raw p-values, the approach of Ge *et al*. (2003) is taken. The formula for calculating the permuation based raw p-values is described in section 13.2.2.

Data Format II

For data sets with omitted raw p-values, the approach of Becker and Knapp (2004) is taken. The raw p-value of the observed and permutation test statistics are calculated on basis of the formulas, specified in sections 13.2.3 and 13.2.4.

6 Calculation identifier

The calculation identifier consists of date and time of the moment, when the program is started, and the name of the input file. E.g. 2005126_18751_%Input_File_Name%. It is used to name log and result files.

7 Configuration

The program is configured by the configuration file *'configuration.conf'*. Table 1 shows parameters and their valid value set.

7.1 Program parameters

INTERACTIVE_DIALOG

If INTERACTIVE_DIALOG is set to 1 the program will ask interactively for confirmation of starting calculation and split file creation. If the parameter is set to 0, this dialog is suppressed and the questions are automatically confirmed. This is useful for calling SDMinP automatically via a batch script.

Section	Parameter	Purpose	Values	Default by download
PROGRAM	INTERACTIVE_DIALOG	switch interactive mode on and off	0, 1	1
CALCULATION	TEST_CHARACTER	specifies the test character	LEFT_SIDED RIGHT_SIDED TWO_SIDED	RIGHT_SIDED
”	GLOBAL_PVAL_OPTION	specifies the formula	SIMPLE EXTENDED	SIMPLE
LOG	RAW_P_CALCULATION	enables logging	0, 1	0
”	SORTING	”	0, 1	0
”	STEP_DOWN_PROCEDURE	”	0, 1	0
”	GLOBAL_PVAL_CALCULATION	”	0, 1	0
RESULT	R_FORMATTED	generate additional result file in R format	0, 1	0
SPLITFILES	MAX_LINES	number of lines per split file (if < 1, no split files will be used)	integer	10
”	REMOVE_AFTER_USAGE	defines, whether split files stay or will be deleted	0, 1	1

Table 1: Configuration parameter.

7.2 Calculation parameters

TEST_CHARACTER

Specify here whether the test is left-,right- or two-sided.
For two-sided tests (the symmetric center is 0), the values of the test statistics are internally converted to their absolute value. Then, the calculation formulas for right-sided tests are applied.

GLOBAL_PVAL_CALCULATION

If GLOBAL_PVAL_CALCULATION is set to 'SIMPLE', the global p-value is the same as the smallest adjusted p-value.
If GLOBAL_PVAL_CALCULATION is set to 'EXTENDED', the global p-value is determined by taking the distribution of the smallest and second smallest raw p-values into account (see section 13.4). This is appropriate for relatively small numbers of permutation replicates (Becker and Knapp, 2004).

7.3 Log parameters

Each log parameter enables the logging for a certain processing step. Each processing step is logged into its own log file. The file name consists of the *calculation identifier* and the specifier for the logged processing step. Consider, that activating the logging slows down the program performance. It makes therefore sense to activate it only for small test data sets to follow up the processing steps.

RAW_P_CALCULATION

Logs the calculation of the raw p values.

SORTING

Logs the sorting of the hypotheses, which have to be ordered by their p-value before applying the step-down procedure. For details see section 9.

STEP_DOWN_PROCEDURE

Logs the step-down calculation. For details see section 9.

GLOBAL_PVAL_CALCULATION

Logs the global p-value calculation. For details see section 9.

7.4 Result parameters

R_FORMATTED

If R_FORMATTED is set to 1 an additional result file in R format is created. For details see section 10.

7.5 Split File parameters

MAX_LINES

If MAX_LINES $>$ 0: split files (see section 8) are created with a maximal number of ’MAX_LINES’ lines. If MAX_LINES $\leq$ 0: the program works on top of the original input file. No split files are created. The number of lines per split file influences the performance. The default value is 10.

REMOVE_AFTER_USAGE

Specifies, whether split files are deleted automatically before the program terminates (REMOVE_AFTER_USAGE = 1) or if they stay (REMOVE_AFTER_USAGE = 0).

8 Split Files

In order to improve the program performance, the input data file is divided into split files. These can be browsed faster, what improves performance of the program.
The reason for the creation of split files is that the program often has to access a specific line directly. Instead of browsing through all preceding lines of the original input file, it can refer directly to a much shorter split file with the contained required line. Here, the program has maximally to browse through a defined maximum number of lines (see section 7.5).
The split files are stored in the temp-directory, which is always cleaned before new split files for a new input file will be created. Split files can automatically be deleted directly after the calculation. This can be controlled via the configuration file 7.5). The file extension for split files is ’.split’.
NOTE: The usage of split files requires for their storage at least the same disc space as the input data file. Make sure, that you have enough space, when using split files. SDMinP does not check for free disc space, but it asks in the interactive mode (see configuration) explicitly for a confirmation, before split files are created.

9 Log Files

If you enable the logging of specific calculation steps, log files are written into the log directory. The log file name consists of the calculation tag and the specifier for what has been logged. E.g.:

1. 2005126_16533_%Input File Name%_log_calcRawP.txt
2. 2005126_16533_%Input File Name%_log_sorting.txt
3. 2005126_16533_%Input File Name%_log_stepDown.txt
4. 2005126_16533_%Input File Name%_log_globalPval.txt

9.1 Log the raw p value calculation

The observed and permutation test statistic are shown per hypothesis test, together with their calculated empirical raw p value. Additionally depicted are the used approach (see in section 5) and hypothesis test character.

9.2 Log the sorting of hypotheses

Before the step-down algorithm starts, the tests have to be ordered by size of the raw p-values of the observed test statistic. The algorithm starts with the less significant p-value, such that the hypotheses tests are ordered descending by the raw p-value.

9.3 Log the step down algorithm & the enforced monotonicity step

Each step of the the step down algorithm is shown. Thus it is possible to follow up the changing values of the ’q-vector’ (see Ge *et al*. (2003)) and the intermediate adjustment of the empirical p-value of the observed statistic. The change of the q-vector of second smallest raw p-values is stored additionally, if the formula for the global p-value is chosen to be ’EXTENDED’ (see section 7.2).
The ’enforcing monotonicity step’ (see Ge *et al*. (2003)) is attached at the end of this logging file.

9.4 Log the global p-value calculation

If the formula for the global p-value calculation was chosen to be ’SIMPLE’, the q-vector of the smallest permutation raw p-values per permutation replicate is depicted, sorted ascending. The minimum raw p-value of the observed test statistics is depicted, as well as the result of the global p-value.
If the calculation option was set to ’EXTENDED’, the sorted q-vector of the second smallest permutation raw p-values per permutation replicate and the second smallest p-value of the observed test statistics are additionally depicted.

10 Result Files

10.1 Default result file

The result file stores the calculation results for each hypothesis. Additionally the calculation parameters, the input file and calculation time are given. E.g.:

```
InputFile:  C:\SDMinP\data\example.txt

Number of hypotheses tests: 7
Number of perm. test statistics:    10
Start-Time: 15-50-55
End -Time: 15-50-57

TestCharacter        :  RIGHT_SIDED
RawP - Formula       :  BECKER
```

```
Global pval - Formula:  SIMPLE



ID     Tval  RawP AdjP  Global
m1     3.0   0.2  0.4   NA
m2     2.12  0.2  0.4   NA
m3     2.0   0.4  0.50  NA
m4     4.48  0.1  0.3   NA
m5     2.44  0.3  0.5   NA
m6     3.03  0.2  0.4   NA
m7     6.47  0.2  0.4   NA
Global NA    NA   NA    0.3
```

10.2 Result file in R-format

If enabled in the configuration file, an additional result file in R-format is created. This file can be imported into the statistic program R (R Development Core Team, 2004) by applying the command 'read.table(%file-path%)'. E.g.:

```
    ID   Tval   RawP   AdjP   Global
1   m1   3.0    0.2    0.4  NA
2   m2     2.12   0.2    0.4  NA
3   m3     2.0    0.4    0.5  NA
4   m4     4.48   0.1    0.3  NA
5   m5     2.44   0.3    0.5  NA
6   m6     3.03   0.2    0.4  NA
7   m7     6.47   0.2    0.4  NA
8   Global NA     NA     NA   0.3
```

10.3 Plotting the results in R

The following routines plot the results in R:

```
# read the input result
SDMinPin <- read.table("%path to R-formatted result file%")

# data.frame without global values, to get rid of the NA's
newDF <- data.frame(ID = SDMinPin$ID[1:dim(SDMinPin)[1]-1],
RawP = SDMinPin$RawP[1:dim(SDMinPin)[1]-1],
AdjP = SDMinPin$AdjP[1:dim(SDMinPin)[1]-1],
Tval = SDMinPin$Tval[1:dim(SDMinPin)[1]-1])

# prepare data objects
identif <-as.character(newDF[,1])
idpl <- c(1:(dim(SDMinPin)[1]-1))
```

```
globalP <- SDMinPin$Global[dim(SDMinPin)[1]]
logGlobalP <- -log(globalP,10)
logRawP <- -log(newDF$RawP[1:dim(SDMinPin)[1]-1],10)
logAdjP <- -log(newDF$AdjP[1:dim(SDMinPin)[1]-1],10)
tvalues <- newDF$Tval

# plot the graphics

# plot 3 graphics next to each other
# if you want to display only one graphic at
# once, comment out the following line
op <- par(mfcol=c(1,3))

# plot test statistics
plot(idpl,tvalues, type="p", bty="n",axes=F, xlab="Hypotheses",
ylab="Test statistics",
main="Test statistics")
axis(1, labels = as.character(newDF[,1]))
axis(2)
box()

# plot raw p-values
plot(idpl,logRawP, xlab="Hypotheses", axes=F, ylab="-log10(p)",
main="Raw p-values")
axis(1, labels = as.character(newDF[,1]))
axis(2)
box()

# plot adjusted p-values
plot(idpl,logAdjP, xlab="Hypotheses", axes=F, ylab="-log10(p)",
main="Adjusted p-values")
axis(1, labels = as.character(newDF[,1]))
axis(2)
box()

# draw a red line, corresponding to the global p-value
abline(logGlobalP ,0, col = "red")
```

11 Step-by-Step example

This section serves as a guideline how to use SDMinP for obtaining adjusted p-values. We have three hypotheses in the multiple testing experiment, and for each hypothesis, we have one observed test statistic and three permutation test statistics.

1. Preparation of the input file

   The next step is to write these values to a file, accordingly to the file format definitions in section 4. Each line stands for one hypothesis, and contains the unique identifier, the empirical p-value or the placeholder 'NA', the observed test statistic and the permutation test statistics per permutation. The placeholder for the raw p-value of the observed test statistic is set. In this example SDMinP calculates the corresponding raw p-values.

   ```
   id1  NA  1.3  1.2  1.8  9.4
   id2  NA  7.3  5.6  1.0  4.5
   id3  NA  5.2  10.0 2.6  9.3
   ```

   We save the file under the name 'ourdata.txt' in the directory 'C:/our_data'.

   Hint: if you use R and have the data in an data.frame object, you can use the R command "write" to store the data to a file.

   ```
   my.data.frame
         [,1]  [,2]  [,3]
    [1,] "id1"  "id2"  "id3"
    [2,] NA     NA     NA
    [3,] "1.3"  "7.3"  "5.2"
    [4,] "1.2"  "5.6"  "10"
    [5,] "1.8"  "1"    "2.6"
    [6,] "9.4"  "4.5"  "9.3"
   write(my.data.frame, ncolumns =
                      dim(my.data.frame)[1], "c:/our_data/ourdata.txt")
   ```

2. Set the configuration parameters

   Now, we have to prepare the configuration file, located in the *conf* directory and to set the calculation parameters (see section 7).

   We run the application in the interactive mode:

   ```
   INTERACTIVE_DIALOG: 1
   ```

   Our test statistic is right-sided, such that we have to set the test character property respectively:

   ```
   TEST_CHARACTER:  RIGHT_SIDED
   ```

   We use the standard formula:

   ```
   GLOBAL_PVAL_OPTION: SIMPLE
   ```

   To visualize the calculation result with the statistic program R, we chose:

```
R_FORMATTED: 1
```

We disable the logging:

```
SORTING                : 0
RAW_P_CALCULATION      : 0
STEP_DOWN_PROCEDURE    : 0
GLOBAL_PVAL_CALCULATION: 0
```

We set the maximum lines per split file to 10. Additionally we want to delete the created split files automatically:

```
MAX_LINES           : 10
DELETE_AFTER_USAGE  : 1
```

3. Program call

   The program must be called in the directory:

   ```
   c:/programs/SDMinP/src
   ```

   We call SDMinP by typing the command ’python main.py %Path to datafile%’:

   ```
   C:\programs\SDMinP\src\python main.py C:/our\_data/ourdata.txt
   ```

4. Interactive mode

   The program mode is configured to run interactively. We confirm all questions by typing ”y”, otherwise the program terminates.

   ```
   Start calculation (type ’y’)? y
   ...
   Split files are going to be created Max line number per split file:
   10 At least 0.000 MB of free disc space are needed.
   Are you sure you have enough space (type ’y’)? y
   ```

   The 0.000MB is due to the very small data input file, which has less than 1KB of size.

5. Program run

   The program starts after the confirmation of all questions. The main steps are prompted to the console.

6. View the result

   At the end, the program displays the paths to the result files, i.e. the default and the R-formatted result file.

```
Result(s) stored in
 --> ../result\2005218_15738_ourdata.result.txt
 --> ../result\2005218_15738_ourdata.result.R.txt
```

The result file contains the information about the input data file:

```
InputFile:  C:\our_data\ourdata.txt
```

the number of hypotheses and permutation test statistics:

```
Number of hypotheses tests     : 3
Number of perm. test statistics: 3
```

the calculation time:

```
Start-Time: 15-29-12 End -Time: 15-29-12
```

the test character, the formula for the raw p-value calculation and the specified formula for the global p-value calculation:

```
TestCharacter        :  RIGHT_SIDED
RawP - Formula       :  BECKER
Global pval - Formula:  SIMPLE
```

and finally the result per hypothesis, consisting of identifier, the observed test statistic, the unadjusted raw p-value and the adjusted p-value. The last row and last column show the global p-value:

```
ID      Tval    RawP    AdjP    Global
id1     1.3     0.67    1.0     NA
id2     7.3     0.0     0.67    NA
id3     5.2     0.67    1.0     NA
Global  NA      NA      NA      0.67
```

7. Load result into R for further processing

   In the R console, we have to load the content of the R-readable result file into an R object. We do this using the command ’my.results <- read.table(”%Path to R-readable resultfile%”)’.

```
 my.results <- read.table(
     "C:/programs/SDMinP/result/2005218_15738_ourdata.result.txt")
```

’my.results’ is an R-object of class ’data.frame’.

12 FAQs

- Which platform is best running SDMinP?

  SDMinP is designed to run on any python supporting platform, i.e. Windows (XP, 2000), Unix, Linux or Macintosh. No specific platform dependent functionalities were used. We tested the program on a Windows XP and a Unix system.

- Problems with Unix-Windows file format?

  We did not have any problems with Unix or Windows file formats.

- How do I format my input files?

  You can use scripting or programming languages, such as Perl, Java or Python, to create the input file. In the step-by-step example we show how to create the input file using the statistic program R (www.r-project.org).

- How can I visualize the results?

  To visualize the results the statistic program R (www.r-project.org) can be used. SDMinP provides the possibility to generate an R-readable result file, which can be imported into R by using the command 'my.results <- read.table("%Path to R-readable resultfile%")' in the R console. See section 11.3 in the Documentation for a routine to plot the results.

- Which version of Python supports SDMinP?

  SDMinP was developed in Python 2.3.5. You can download this or newer versions from www.python.org. Python is also a standard program in Linux and Unix environments.

- What is the difference between the formulas for the raw p-value calculation?

  There are two formulas used for calculating the empirical p-value of the test statistics. One is provided by Ge *et al*. (2003), which is used by SDMinP to calculate the raw p-values of the permutation test statistics, when the raw p-value of the observed test statistic is provided in the input file (data input format I). The other formula is provided by Becker and Knapp (2004) and is used for calculating the raw p-value of the observed and permutation test statistics, when the data file does not contain the raw p-value of the observed test statistic (data input format II). For more detailed information about this subject refer to Becker and Knapp (2004).

- Is the program limited in the number of hypotheses or in the number of permutations?

  The program is not limited in the number of hypotheses or the number of permutations. A limit is the maximum size of the input file, which is determined by the operating system. All operating systems should be able to handle files of 2GB. If you want to process larger files, see the documentation of your system.

- Why should I use split files?

  Split files make the computation faster, as they contain the content of the input data file, but split into smaller portions. The program does not have to parse the whole original data file from top to the respective position. The program jumps instead to the shorter split file, containing the respective information. You can always use the advantage of split files.

- Which maximum line number is the best for split files?

  Generally, decreasing the split file size speeds up the program retrieval of specific hypotheses. We recommend to set the number of maximum lines to a value between 10 and 100.

13 Formulas

13.1 Notations

These notations are used to explain the formulas in the following sections:
$t_{m,0}$: observed test statistic of hypothesis m
$t_{m,b}$: permutation test statistic of hypothesis m and permutation replicate b
$p_{m,0}$: unadjusted raw p-value of the observed test statistic of hypothesis m
$p_{m,b}$: raw p-value of the permutation test statistic of hypothesis m and permutation replicate b
$p^*_{m,0}$: adjusted p-value of the observed test statistic of hypothesis m
p_0^{min} : smallest raw p-value of observed test statistics
p_b^{min} : smallest raw p-value of permutation replicate b
p_0^{min2} : second smallest raw p-value of observed test statistics
p_b^{min2} : second smallest raw p-value of permutation replicate b
p^{glob} : global p-value
M: number of hypotheses , $m \in \{1..M\}$
B: number of permutation replicates, $b \in \{1..B\}$
H_0^C: the complete null hypothesis, i.e. the assumption that no hypothesis will be significant
P_m: vector of permutation raw p-values for hypothesis m: $[p_{m,1}, p_{m,2}, .., p_{m,B}]$
P^{min}: vector of smallest permutation raw p-values per permutation replicate b: $[p_1^{min}, .., p_b^{min}, .., p_B^{min}]$
P^{min2}: vector of second smallest permutation raw p-values per permutation replicate b: $[p_1^{min2}, .., p_b^{min2}, .., p_B^{min2}]$

13.2 Empirical (permutation based) raw p value

We considered two different approaches for the calculation of raw p-values. Which approach will be chosen by the program depends on the provided data input format (see section 5). One approach is suggested by (Ge *et al.*, 2003), the other by (Becker and Knapp, 2004).
NOTE: the depicted formulas are designed for a right-sided test. For a left-sided test, the operator has to be changed from to '$\geq$' to '$\leq$'. For two-sided tests, the absolute values of the test statistics would have to be taken, by keeping the operator '$\geq$'.

13.2.1 Approach Ge et al. - Raw p-value of the *observed test statistic* is provided

If the raw p-value of the observed test statistics are provided, i.e. Data Format I is chosen, there is no need for SDMinP to calculate the raw p-value on basis of the provided permutation test statistics. The reason for offering the possibility of providing a pre-calculated p-values of the observed test statistics is that their calculation might be based on other permutation replicates than provided here for the step-down minP adjustment.

13.2.2 Approach Ge et al. - Raw p-value calculation of *permutation test statistics*

As basis of the calculation of the permutation raw p-values $p_{m,i},\ i = 1..B$, serve the permutation test statistics $t_{m,i}$ of hypothesis m. The difference to the approach of Becker and Knapp (2004) is, that the provided observed test statistic is not incorporated into the calculation.

$$p_{m,i} = \frac{\#\{s{:}1 \leq s \leq B, t_{m,s} \geq t_{m,i}\}}{B}$$

13.2.3 Approach Becker - Raw p-value calculation of the *observed test statistic*

If the raw p-value of the observed test statistics was not passed by the data input file (Data Format II), it is calculated by SDMinP by using the formula:

$$p_{m,0} = \frac{\#\{s{:}1 \leq s \leq B, t_{m,s} \geq t_{m,0}\}}{B}$$

13.2.4 Approach Becker - Raw p-value calculation of *permutation test statistics*

Here, the observed test statistic is included into the calculation. The permutation raw p-values $p_{m,i}$, $i = 1..B$, are calculated by the formula:

$$p_{m,i} = \frac{\#\{s{:}0 \leq s \leq B, s \neq i, t_{m,s} \geq t_{m,i}\}}{B}$$

13.3 Step down algorithm

Assume that $p_{r_1,0} \leq p_{r_2,0} \leq .. \leq p_{r_m,0}$ are the ordered raw p-values of the observed test statistics. Then, step-down minP adjusted p-values are defined by the formula (Westfall and Young, 1993):

$$p^*_{r_i,0} = max_{k=1,..,i}\{Pr(min_{l=k,..,m} P_{r_l} \leq p_{r_k,0} | H_0^C)\}$$

13.4 Global p-value

The smallest adjusted p-value of the individual hypotheses can be taken to test the global hypothesis:

$$p^{glob} = \frac{\#\{s{:}1 \leq s \leq B, p_s^{min} \leq p_0^{min}\}}{B}$$

If the improved formula of Becker and Knapp (2004) is used, the distribution of the second smallest permutation raw p-values over all hypotheses is additionally taken into account to calculate the global p-value. This is appropriate for relatively small numbers of permutation replicates.

$$p^{glob} = \frac{\#\{s{:}1 \leq s \leq B, p_s^{min} < p_0^{min} or (p_s^{min} = p_0^{min} and p_s^{min2} \leq p_0^{min2})\}}{B}$$

14 References

Becker,T.,Knapp,M. (2004) A Powerful Strategy to Account for Multiple Testing in the Context of Haplotype Analysis, *Am. J. Hum. Genet.*, **75**, 561-570.

Ge,Y.,Dudoit,S.,Speed,T.P. (2003) Resampling-based Multiple Testing for Microarray Data Analysis, *Test*, **12**, 1-77

Obreiter M., Fischer C., Chang-Claude J., Beckmann L. (2004) SDMinP: a program to control the family wise error rate using step-down minP adjusted P-values. *Bioinformatics.* 2005 Jul 15;21(14):3183-4. Epub 2005 May 6

R Development Core Team (2004). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org.

Westfall,P.H.,Young,S.S. (1993) Resampling-based multiple testing: examples and methods for P-value adjustment *John Wiley & Sons, New York*

15 The GNU General Public License

Version 2, June 1991
Copyright © 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or

a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program “as is” without

WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms. To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> one line to give the program's name and a brief idea of what it does.
> Copyright (C) yyyy name of author
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Also add information on how to contact you by electronic and paper mail.
If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright (C) yyyy name of author
> Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
> This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program
> ‘Gnomovision’ (which makes passes at compilers) written by James Hacker.
>
> signature of Ty Coon, 1 April 1989
> Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.