GWAtoolbox
An R package for the fast processing of data from Genome-Wide Association Studies

Christian Fuchsberger Daniel Taliun Cristian Pattaro

June 16, 2011

Contents

1 Introduction

GWAtoolbox is an *R* package for processing data originated from Genome-Wide Association Studies (GWAS). GWAS have become increasingly popular in the last years, leading to the discovery of hundreds of common genetic variants affecting the risk of diseases (such as diabetes, hypertension, chronic kidney disease, etc.) or the level of quantitative biological parameters.

Results from GWAS typically consist in large files where, for each single nucleotide polymorphisms (SNP), statistics related to the association between the SNP and the studied trait are stored. The number of SNPs which is currently being analyzed in most GWAS is in excess of 2.5 Million and is expected to increase rapidly. For each individual SNP, the minimal information stored consists of the SNP identification code (SNPID), chromosomal position, genotype (reference and non-reference alleles), frequency of the reference allele, and SNP effect size and its standard error. Additional information such as p-value, minor allele frequency (MAF), and an imputation quality index are often provided. As a consequence, the typical dimension of GWAS result files is of >2.5 Million rows by >9 columns, for a total file size which is often larger than 300 Mbytes.

With the aim of detecting common or less common genetic variants with modest effects, it is now common practice to pool results from individual studies into meta-analysis efforts which not rarely involve dozens of studies. In these consortia initiatives, each individual study contributes several files either because multiple traits are being analyzed or because different analyses on the same trait are needed. Consequently, statisticians working in consortia have to deal with a massive amount of files which need to be quality controlled to avoid problems during the meta-analysis process. As a result of the quality control (QC) process, some files could be found to be corrupted or erroneous so that new data upload is needed from individuals studies. In this way, the loop between the consortium and the individual study analyst originates multiple file checks, until a satisfactory data quality is achieved.

When working with such large datasets in R, simple operations such as the uploading files into the R working space, file management, and data plotting, can take considerable time, and a systematic QC of hundreds of files can be unfeasible or may require several weeks.

With the *GWAtoolbox* we provide a set of instruments to simplify the data handling in the framework of meta-analyses of GWA data. The function *gwasqc()* is capable to process a high number of GWAS data files in a single run, and producing several QC reports and figures. A routine for the between-study comparison is also provided to check systematic difference between files. In addition, the package contains annotation and graphical tools to help the result interpretation.

2 Installation

GWAtoolbox package can be downloaded from `http://www.eurac.edu/GWAtoolbox.html`. It requires R version 2.9.2 or higher. The installation of package varies depending on your host operating system and user privileges. In this section we provide detailed installation instructions for a wide range of settings.

2.1 Windows

GWAtoolbox for Windows is distributed in compiled binary form. The following steps describe the installation procedure:

1. Download the latest package version *GWAtoolbox_X.Y.Z.zip*.
2. Start the R program.

3a. If you have administrator privileges (you can install packages to the main R library):

   i. Execute the command:

   ```
   install.packages("path/to/GWAtoolbox_X.Y.Z.zip",
   repos=NULL)
   ```

   where `path/to` is the directory of the downloaded package.

   ii. Now you can load the package in R with the command:

   ```
   library(GWAtoolbox)
   ```

3b. If you do NOT have sufficient privileges to install packages to the main R library directory:

   i. Execute the command:

   ```
   install.packages("path/to/GWAtoolbox_X.Y.Z.zip",
   lib="path/to/install/directory",
   repos=NULL)
   ```

   where `path/to/install/directory` is the path with your install directory.

   ii. Now you can load the package in R with command:

   ```
   library(GWAtoolbox, lib.loc =
   "path/to/install/directory")
   ```

2.2 Unix

GWAtoolbox for Unix is distributed in source form and, therefore, it is compiled on the user machine. This requires the following tools to be installed:

- C/C++ compilers
- GNU Scientific Library (GSL)* version 1.8

When these requirements are fulfilled, the following steps will guide you through the package installation process:

1. Download the latest package version *GWAtoolbox_X.Y.Z.tar.gz*.

2a. If you have administrator privileges (you can install packages to the main R library):

   i. In the Unix shell execute the command:

*http://www.gnu.org/software/gsl/

```
R CMD INSTALL path/to/GWAtoolbox_X.Y.Z.tar.gz
```

where `path/to` is the directory of the downloaded package.

ii. Now you can start the R program and load the package with the command:

```
library(GWAtoolbox)
```

2b. If you do NOT have sufficient privileges to install packages to the main R library directory:

i. In the Unix shell execute the single line command:

```
R CMD INSTALL path/to/GWAtoolbox_X.Y.Z.tar.gz
-l path/to/install/directory
```

where `path/to/install/directory` is the path with your install directory.

ii. Now you can start the R program and load the package with the command:

```
library(GWAtoolbox, lib.loc="path/to/install/directory")
```

2.3 Mac OS X

GWAtoolbox for Mac OS X is distributed in compiled binary form. The following steps describe the installation procedure:

1. Download the latest package version *GWAtoolbox_X.Y.Z.tar.gz*.

2a. To install from the Mac OS X shell (*Terminal*):

3a. If you have administrator privileges (you can install packages to the main R library):

i. In the Mac OS X shell execute the command:

```
R CMD INSTALL path/to/GWAtoolbox_X.Y.Z.tar.gz
```

where `path/to` is the directory of the downloaded package.

ii. Now you can start the R program and load the package with the command:

```
library(GWAtoolbox)
```

3b. If you do NOT have sufficient privileges to install packages to the main R library directory:

i. In the Mac OS X shell execute the single line command:

```
R CMD INSTALL path/to/GWAtoolbox_X.Y.Z.tar.gz
-l path/to/install/directory
```

where `path/to/install/directory` is the path with your install directory.

ii. Now you can start the R program and load the package with the command:

```
library(GWAtoolbox, lib.loc="path/to/install/directory")
```

2b. To install from R:

3. Start the R program.

4a. If you have administrator privileges (you can install packages to the main R library):

   i. Execute the command:

      ```
      install.packages("path/to/GWAtoolbox_X.Y.Z.tar.gz",
      repos=NULL)
      ```

      where `path/to` is the directory of the downloaded package.

   ii. Now you can load the package in R with the command:

      ```
      library(GWAtoolbox)
      ```

4b. If you do NOT have sufficient privileges to install packages to the main R library directory:

   i. Execute the command:

      ```
      install.packages("path/to/GWAtoolbox_X.Y.Z.tar.gz",
      lib="path/to/install/directory",
      repos=NULL)
      ```

      where `path/to/install/directory` is the path with your install directory.

   ii. Now you can load the package in R with command:

      ```
      library(GWAtoolbox, lib.loc =
      "path/to/install/directory")
      ```

3 The Quality Control Workflow

A careful and thorough data QC should be performed before starting any meta-analysis of GWAS data, especially when many studies are involved. In this framework, we identified three objectives of a good QC analysis:

1. formal checking: whether all files that will be entered in the meta-analysis process fulfill the format guidelines. This includes:

   - consistency of column names with meta-analysis guidelines;
   - presence of the minimal required information;
   - the number of chromosomes is as expected;
   - data are in a format that can be analyzed (numeric, character, factor);
   - all SNP identification numbers are unique;
   - alleles are coded in letters/numbers as expected;
   - missing values are coded in a consistent way;
   - the field separator is as expected;
   - strand assessment;

2. quality checking: evaluating the quality of data in each single file. This includes:

   - presence of unexpected values for some of the items required for the meta-analysis (e.g.: negative p-values or standard errors);

- p-value inflation and p-value distribution;

3. global checking: identification of any systematic biases that can disturb the analysis. It is aimed to uncover studies that are systematically different from the others. This may happen when, for instance, analysts of one study forget to log-transform the phenotype or apply the wrong model to the data.

Formal checks and quality checks of individual studies are performed in *GWAtoolbox* using the *gwasqc()* function. *gwasqc()* was built to include the following features:

1. rapid file processing and reporting;
2. eliminate routine user operations;
3. multi-format reporting which includes *HTML*, *CSV*, and text files.

The complete QC workflow can be summarized in four basic steps (see Figure 1):

1. collect the GWAS data files;
2. write an input script to process of all GWAS files with the *gwasqc()* function;
3. run the QC using *gwasqc()*;
4. analyze the QC results to uncover errors or inconsistencies.

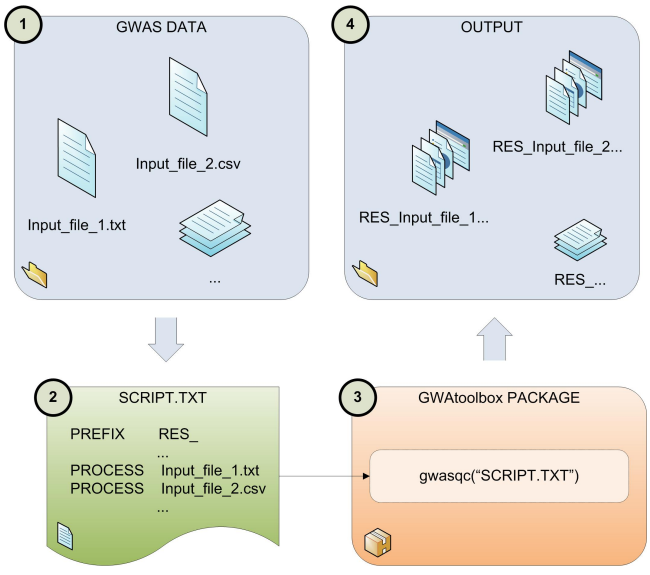

Figure 1: The quality control workflow.

In the next sections we cover each of the four steps and describe the requirements for the input files and the precise content of all output files.

4 GWAS data files

GWAS data are usually stored as delimited text files. The first line of the file is the header row that describes the content of every column. The field separator for the columns can be any among *whitespace*, *tabulation*, *comma*, or *semicolon*. The field separator must be the same for every row in the file, including the header.

There is a minimum set of columns, that every GWAS data file should contain. In *GWAtoolbox*, the following information is required for every file:

- Marker name
- Chromosome number or name
- Marker position
- Coded and non-coded allele
- Allele frequency for coded allele
- Strand
- Imputation label
- Imputation quality
- Effect size
- Standard error
- P-value

The *gwasqc()* function will take full responsibility for checking if an input file contains all the information and will report about missing data.

More non-mandatory items can be included in the data file as, for example, the study sample size, the SNP call rate for genotyped SNPs, the p-value of the Hardy-Weinberg equilibrium test for genotyped SNPs, etc.

5 The Input Script

gwasqc() can analyze several GWAS data files consecutively. Instructions are provided using a script in a text file. The format of the script file resembles the one of the METAL input files[†].

Within the input script file, the user can list all GWAS file names to be analyzed and specify the format of each single GWAS file, including column names, field separator, etc. In the case that more GWAS files are in the same format, file specifications can be entered only once, before listing the file names. Example 1 illustrates the content of a hypothetical input script file.

Example 1

[†]http://www.sph.umich.edu/csg/abecasis/metal/

```
# Description of input data columns
MARKER        SNPID
CHR           Chromosome
POSITION      Position
N             n_total
ALLELE        coded_allele noncoded_allele
STRAND        strand
EFFECT        beta
STDERR        se
PVALUE        pval
FREQLABEL     allele_freq_coded_allele
IMPUTED       imputed
IMP_QUALITY   oevar_imp

# High quality filters
HQ_SNP   0.01   0.3

# Plotting filters
MAF      0.01   0.05
IMP      0.3    0.6

# Prefix for output files
PREFIX   res_

# Input file with GWA data
PROCESS  input_file.txt
```

◁

5.1 Specifying Input Data Files

The names of the GWAS data files are specified in the input script with the command **PROCESS**[‡] If multiple files have to be checked, multiple **PROCESS** lines must be specified.

Example 2 The input script contains the following two lines:

```
PROCESS   input_file_1.txt
PROCESS   /dir_1/dir_2/input_file_2.csv
```

Then, QC is applied first to *input_file_1.txt* and then to *input_file_2.csv*. As used in the example, when files reside in different directories, the full path must be specified. ◁

5.2 Describing Input Data Columns

5.2.1 Field Separator

The field separator may be different for each GWAS data file. The *gwasqc()* function automatically detects the separator field for each input file *based on*

[‡] *GWAtoolbox* supports single line feed ('\n') character or carriage return character ('\r') followed by line feed character as the line terminators in the input files.

the first 10 rows. However, the user has the possibility to specify the separator manually for each individual file using the command **SEPARATOR**. Table 1 lists all supported separators.

Argument	Separator
COMMA	*comma*
TAB	*tabulation*
WHITESPACE	*whitespace*
SEMICOLON	*semicolon*

Table 1: The list of arguments for the SEPARATOR command.

Example 3 In the following input script:

```
PROCESS      input_file_1.txt
SEPARATOR    TAB
PROCESS      input_file_2.csv
PROCESS      input_file_3.txt
```

the field separator for the input file *input_file_1.txt* is determined automatically by *gwasqc()*, but for the input files *input_file_2.csv* and *input_file_3.txt* the separator is manually set to tabulation. ◁

5.2.2 Missing Values

By default *gwasqc()* assumes that missing values are labeled as *NA*. However, the label for missing value can be specified manually by the user with the command **MISSING**.

Example 4 Let's assume the following input script:

```
MISSING      -
PROCESS      input_file_1.txt
MISSING      NA
PROCESS      input_file_2.csv
```

For the file *input_file_1.txt* the *hyphen* symbol is set as symbol for missing value. Afterwards, it is changed to *NA* and is used to process *input_file_2.csv*. ◁

5.2.3 Column Names

In table 2 the complete list of the default column names for a GWAS data file is reported. These names identify uniquely the items in the GWAS data file.

Given that different names can be provided with the GWAS data files, *gwasqc()* allows to redefine the default values for every input file in the input script. The redefinition command consists of the default column name followed by a new column name. To redefine the default column names for *coded* and *non-coded* alleles, the command **ALLELE** is followed by two new column names.

Default column name(s)	Description
MARKER	Marker name
CHR	Chromosome number or name
POSITION	Marker position
ALLELE1, ALLELE2	Coded and non-coded alleles
FREQLABEL	Allele frequency for the coded allele
STRAND	Strand
IMPUTED	Label value indicating if the marker was imputed (1) or genotyped (0)
IMP_QUALITY	Imputation quality statistics; this can be different depending on the software used for imputation: MACH's *Rsq*, IMPUTE's *properinfo*, ...
EFFECT	Effect size
STDERR	Standard error
PVALUE	P-value
HWE_PVAL	Hardy-Weinberg equilibrium p-value
CALLRATE	Genotype callrate
N	Sample size
USED_FOR_IMP	Label value indicating if a marker was used for imputation (1) or not (0)

Table 2: The default column names.

Example 5 Let's assume to have two input files, *input_file_1.txt* and *input_file_2.txt*. In *input_file_1.txt*, the column names for effect size and standard error are *beta* and *SE*, respectively. In the *input_file_2.txt*, the column name for the effect size is the same as in *input_file_1.txt*, but the column name for the standard error is *STDERR*. The correct column redefinitions are as follows:

```
EFFECT     beta
STDERR     SE
PROCESS    input_file_1.txt
STDERR     STDERR
PROCESS    input_file_2.csv
```

First, we redefine column names for the input file *input_file_1.txt*. We note that the column *beta* doesn't need to be redefined for the input file *input_file_2.csv*. However, for this file we need to redefine the column *STDERR*, returning it to the default column naming. ◁

Example 6 Consider an input file *input_file_1.txt* with the following names for ALLELE1 and ALLELE2: *myRefAllele* and *myNonRefAllele*. The new column definition is applied as follows:

```
ALLELE    myRefAllele myNonRefAllele
PROCESS   input_file_1.txt
```

◁

5.2.4 Case Sensitivity

By default the *gwasqc()* function assumes that column names in the input files are case insensitive. For example, the column names *STDERR*, *StdErr*, and *STDErr* are all perfectly equivalent. This behaviour can be changed for every input file in the input script using the command **CASESENSITIVE**, that controls case sensitivity for the column names. Table 3 lists all possible arguments.

Argument	Description
0	Column names in the input file are case insensitive (default)
1	Column names in the input file are case sensitive

Table 3: The list of arguments for CASESENSITIVE command.

Example 7 Consider the following commands:

```
CASESENSITIVE   1
PROCESS         input_file_1.txt
CASESENSITIVE   0
PROCESS         input_file_2.csv
```

In this case, the column names in the input file *input_file_1.txt* are case sensitive and must correspond exactly to the default column names, while the column names in the input file *input_file_2.csv* are case insensitive. ◁

5.3 Specifying Data Filters

5.3.1 Implausible Values Filter

Often, there is the necessity to identify implausible values for the statistics that will be included in the meta-analysis. Implausible values for the effect estimate, for its standard error, and for the p-value are sometimes generated by the software used for the association testing. In case of small numbers, which is typical of a disease outcome with a small number of cases or of a SNP with very small minor allele frequency, statistical packages can report inconsistent results. This is due to statistical algorithms that fail to converge because of data sparseness. Other types of inconsistencies can originate from errors in the file management.

In these situations, it is important to identify the SNPs with inconsistent values, so that they can be removed before starting the meta-analysis. *gwasqc()* can identify these values by using appropriate threshold values. The number of SNPs affected by this kind of problems is reported. In addition, these SNPs are excluded from the calculation of the summary statistics on data quality.
The implausible values filter is used in the *gwasqc()* function to identify implausible data values. Table 4 lists the columns for which the filter is applied and the default thresholds.

Default column name	Default thresholds
STDERR	$[0, 100000]$
IMP_QUALITY	$(0, 1.5)$
PVALUE	$(0, 1)$
FREQLABEL	$(0, 1)$
HWE_PVAL	$(0, 1)$
CALLRATE	$(0, 1)$

Table 4: The default implausible values filter.

The default thresholds can be redefined for every column in the input script. The new threshold values for a column can be specified after the redefinition of the column name (see Section 5.2.3).

Example 8 Let's assume that the input file *input_file_1.txt* has a standard error column called *STDERR* and that the corresponding column in the input file *input_file_2.csv* is called *SE*. In addition, the imputation quality column is defined as *oevar_imp* in both files. The following script shows how the user can redefine the column names while applying different plausibility filters:

```
STDERR        STDERR 0 80000
IMP_QUALITY   oevar_imp 0 1
PROCESS       input_file_1.txt
STDERR        SE 0 100000
PROCESS       input_file_2.csv
```

The file *input_file_1.txt* has new $[0, 80000]$ thresholds for the standard error column and new $(0, 1)$ threshold for the imputation quality. For the file *input_file_2.csv* the thresholds of $[0, 100000]$ will be applied to the standard error column, while for the imputation quality column the same filters as for the *input_file_1.txt* will be applied. $\triangleleft$

5.3.2 High Quality Filters

In many cases, the analysis is restricted to SNPs with high imputation quality and with not too small minor allele frequency. We call these SNPs '*high quality SNPs*', that is SNPs for which results should be quite robust. In the special case, when estimating the inflation factor, *lambda*, to check the presence of cryptic relatedness or hidden population sub-structures, it can be important to remove SNPs that could artificially increase the value of *lambda*. Summary statistics are calculated after excluding SNPs with low quality (*CSV* report files). Table 5 lists the default thresholds for the allele frequency and for the imputation quality.

The default values can be redefined using the command **HQ_SNP** for every input file in the input script. The command is followed by two values: the first one corresponds to the threshold for the minor allele frequency, and the second one corresponds to the threshold for the imputation quality.

Default column name	Default thresholds
FREQLABEL	> 0.01
IMP_QUALITY	> 0.3

Table 5: The default high quality imputation filters.

Example 9 If we want to define 'high quality SNPs' those with minor allele frequency > 0.03 and with imputation quality > 0.4, we would add the following lines to the input script:

```
HQ_SNP      0.03 0.4
PROCESS     input_file_1.txt
```

$\triangleleft$

5.3.3 Plotting Filter

The plotting filter is used to select appropriate data for the QQ-plots, boxplots and histograms. The filter has two threshold levels: each of them is applied dependently on the plot type and column. Figure 2 (see Section 5.4.3) shows what data and filters are used when producing plots. Table 6 lists the default threshold values.

Default column name	Default 1st level thresholds	Default 2nd level thresholds
FREQLABEL	> 0.01	> 0.05
IMP_QUALITY	> 0.3	> 0.6

Table 6: The default plotting filter.

The default threshold values for the coded allele frequency and imputation quality can be redefined accordingly with the commands **MAF** and **IMP** for the every input file in the input script.

Example 10 Assume the input script contains the following commands:

```
MAF         0.02 0.03
IMP         0.3 0.5
PROCESS     input_file_1.txt
```

In this example new plotting filter thresholds are set for the input file *input_file_1.txt*. For the first level threshold the coded allele frequency > 0.02 and the imputation quality > 0.3, while for the second level threshold the coded allele frequency > 0.03 and imputation quality > 0.5. $\triangleleft$

5.4 Specifying Output Files

5.4.1 Output File Name

The output file names are constructed from the input file names by adding the specified prefix. This is done both for the textual output files and image files. The prefix can be specified once for all input files, or for every single input file or groups of input files explicitly using the command **PREFIX**.

Example 11 Consider the following input script:

```
PREFIX      res_
PROCESS     input_file_1.txt
PROCESS     input_file_2.csv

PREFIX      result_
PROCESS     input_file_3.tab
```

In this example, all the result output files corresponding to the input files *input_file_1.txt* and *input_file_2.csv* will be prefixed with *res_*, while the result output files corresponding to the input file *input_file_3.tab* will be prefixed with *result_*. ◁

5.4.2 Verbosity Level

The *GWAtoolbox* package provides the possibility to control the number of generated output figures using command **VERBOSITY** (see Table 7 for the available options).

Argument	Description
1	The default and the lowest verbosity level.
2	The highest verbosity level.

Table 7: The list of arguments for the VERBOSITY command.

Example 12 Assume the input script contains the following commands:

```
VERBOSITY       2
PROCESS         input_file_1.txt
VERBOSITY       1
PROCESS         input_file_2.csv
```

In this example the input file *input_file_1.txt* is processed with the highest verbosity level and therefore all figures are produced, while the input file *input_file_2.csv* is processed with the lowest verbosity level and less output figures are generated. ◁

5.4.3 Number And Content Of Plots

Number and content of the output plots depend on the setting of the plotting filters (see Section 5.3.3) and on the available columns in the input file. Figure 2 shows the dependencies. If some dependency is not satisfied because of missing column or filter setting, then the corresponding plot is not produced or may be truncated at different levels.

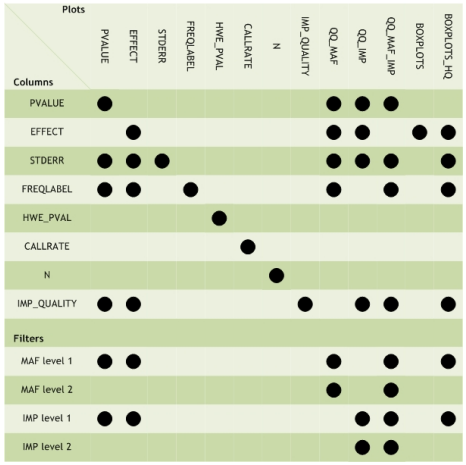

Figure 2: The dependency of plots on columns and filters.

Furthermore, the boxplots comparing the *EFFECT* distributions across studies allow the specification of a **BOXPLOTWIDTH** that can be based on one of the other available information (typically the sample size). As an argument, **BOXPLOTWIDTH** requires one of the default column names. If **BOXPLOTWIDTH** is not specified all boxplots have the same width.

It is also possible to specify labels for every input file, to be used in the plots instead of the full file names, which could be too long and, therefore, clutter the plots.

Example 13 Let *n_total* be the column name which identifies the sample size in the input file *input_file_1.txt*, and *samplesize* the corresponding name in *input_file_2.csv*. Then, consider the following input script:

```
N               n_total
PROCESS         input_file_1.txt first
N               samplesize
PROCESS         /dir_1/dir_2/input_file_2.csv second
BOXPLOTWIDTH    N
```

In this example, the width of the first boxplot for the input file *input_file_1.txt* depends on the *n_total* column, while the width of the second boxplot for the input file *input_file_2.csv* depends on the *samplesize* column. The labels "first" and "second" will be used to label the two studies in the plots. ◁

6 The Output Files

The typical output of the *gwasqc()* function consists of the following files:

1. **Graphical files** (*PNG* file extension) include **histograms** and **boxplots** of the distribution of the main statistics from each GWAS file: effect estimates, imputation quality index, sample size, p-value, allele frequency; **QQ-plots** of the p-value distribution are also provided to investigate the presence of study-design bias. See Figure 2 for an exhaustive list of available plots.

2. **Textual report** (*TXT* file extension) contains information on the GWAS file-format quality and statistics summarizing the distribution of all fields present in the GWAS files (effect estimates, p-values, etc.). The statistics provided in this report can be compared with the graphical output described above.

3. **Comma separated report** (*CSV* file extension) contains statistics limited to the high quality SNPs, as they have been defined by the user when setting the parameter **HQ_SNP** in the QC script (see Section 5.3.2). The tabular format of this file is intended to be useful for users who wants to perform additional analyses and compare results from different GWAS files without having to manage large original GWAS files. Figure 3 shows an example of such file. For each of the variables listed by columns, summary

	N	EFFECT	STDERR	PVALUE	MAF
N	2543887	2542617	2542422	2542617	2543887
N_HQ	2448544	2448544	2448544	2448544	2448544
N_NAs	0	1270	1270	1270	0
Mean	516	-0.00137	0.295076	0.500823	0.230416
StdDev	0	0.333379	0.158119	0.28873	0.143312
Min	516	-8.76038	0.17353	7.34E-08	0.010001
Max	516	6.12165	2.53341	1	0.5
Median	516	-0.00149	0.233707	0.502629	0.217008
Skewness	nan	-0.06933	2.886758	-0.00935	0.214034
Kurtosis	nan	9.109873	11.14145	-1.20085	-1.18454

	IMPUTED	IMP_QUALITY	CALLRATE	HWE_PVAL	STD_EFFECT_0.5
N	2543887	2543887	2543887	NA	NA
N_HQ	2448544	2448544	2448544	NA	NA
N_NAs	0	0	0	NA	NA
Mean	1	0.958868	1	NA	NA
StdDev	0	0.097734	0	NA	NA
Min	1	0.300006	1	NA	NA
Max	1	1.0467	1	NA	NA
Median	1	0.994577	1	NA	NA
Skewness	nan	-3.90226	nan	NA	-0.00587
Kurtosis	nan	16.76683	nan	NA	3.297394

	STD_EFFECT_0.75	STD_EFFECT_0.95	STD_EFFECT_0.99	STD_EFFECT_1
N	NA	NA	NA	NA
N_HQ	NA	NA	NA	NA
N_NAs	NA	NA	NA	NA
Mean	NA	NA	NA	NA
StdDev	NA	NA	NA	NA
Min	NA	NA	NA	NA
Max	NA	NA	NA	NA
Median	NA	NA	NA	NA
Skewness	-0.01036	0.022903	0.040024	-0.07186
Kurtosis	3.822136	4.712606	5.861049	9.066722

Figure 3: The comma separated report file.

statistics are reported by row. Specifically:

- **N** – number of SNPs with available information (i.e. non-missing values).

- **N_HQ** – number of high quality SNPs with that information available (i.e. non-missing values).
- **N_NAs** – number of SNPs with missing values for the specific field of interest.
- **Mean**, **StdDev**, **Min**, **Max**, **Median**, **Skewness**, and **Kurtosis** are referred to the distribution of the specific field.

Columns include:

- **N** – study sample size.
- **EFFECT**, **STDERR**, and **PVALUE** – summaries of the SNP-phenotype associations.
- **MAF**, **IMPUTED**, **IMP_QUALITY**, **CALLRATE**, and **HWE_PVAL** – summaries of the genotype distribution and quality.
- **STD_EFFECT_0.5**, **STD_EFFECT_0.75**, **STD_EFFECT_0.95**, **STD_EFFECT_0.99**, **STD_EFFECT_1** – these columns are only of interest for the reported skewness and kurtosis of the standardized effect estimates. The numbers 0.5, 0.75, 0.95, 0.99, and 1 are referred to the percentage of SNPs with the highest p-values chosen to study the effect size distribution (see Section 8.1 for additional details).

4. **An HTML report** (*main.html* file) combines both textual and graphical output, allowing the user to easily surfing across the results and across the studies included in the QC analysis. In operating systems allowing graphical interfaces, the *HTML* document should be the first file to be opened to investigate the results.

7 Example

This is an embedded R code example. All input files of this example are located in the subdirectory *doc* of the installed *GWAtoolbox* package.

Consider the following five GWAS data files: *gwa_data_example_1.txt*, *gwa_data_example_2.tbl*, *gwa_data_example_3.csv*, *gwa_data_example_4.txt* and *gwa_data_example_5.csv*. The first file contains 16 columns separated with whitespace:

```
> t <- read.table("gwa_data_example_1.txt", header = T, nrow = 1,
+     sep = " ")
> colnames(t)
```

```
 [1] "SNPID"         "chr"           "position"      "coded_all"
 [5] "noncoded_all"  "strand_genome" "beta"          "SE"
 [9] "pval"          "AF_coded_all"  "n_total"       "oevar_imp"
[13] "avpostprob"    "callrate"      "HWE_pval"      "used_for_imp"
[17] "imputed"
```

At the same time, the second file also contains 16 columns, however separated with tabulation:

```
> t <- read.table("gwa_data_example_2.tbl", header = T, nrow = 1,
+     sep = "\t")
> colnames(t)

 [1] "SNPID"         "chr"           "position"      "coded_all"
 [5] "noncoded_all"  "strand_genome" "beta"          "StdErr"
 [9] "p"             "AF_coded_all"  "n_total"       "oevar_imp"
[13] "avpostprob"    "callrate"      "HWE_pval"      "used_for_imp"
[17] "imputed"
```

Analogously, we can preview the headers of the other three files. Then, in order to perform the quality control check of these files with *GWAtoolbox* package, we prepare a simple input script *GWAS_script.txt*. Below are listed commands, which were inlcuded into the script:

```
> cat(readLines("GWASQC_script.txt"), sep = "\n")

# Column names
ALLELE               coded_all noncoded_all
CALLRATE         callrate
CHR                chr
EFFECT                beta
FREQLABEL         AF_coded_all
HWE_PVAL         HWE_pval
IMPUTED         imputed
IMP_QUALITY        oevar_imp
MARKER               SNPID
N                 n_total
POSITION         position
PVALUE                pval
STRAND                strand_genome
STDERR                SE
USED_FOR_IMP         used_for_imp

# Plotting filters for the coded allele frequency and imputation quality
MAF 0.01    0.05
IMP 0.3     0.5

# Prefix for output files
PREFIX       res_

# Column N controls the width of boxplots
BOXPLOTWIDTH        N

# Input file and its short name for plotting
PROCESS gwa_data_example_1.txt Study1

PVALUE               p
STDERR               StdErr

PROCESS gwa_data_example_2.tbl Study2
```

```
PROCESS gwa_data_example_3.csv Study3

PVALUE                pvalue

PROCESS gwa_data_example_4.txt Study4
PROCESS gwa_data_example_5.csv Study5
```

When the input script was prepared, we load the *GWAtoolbox* library and call the *gwasqc()* function as follows:

```
> library(GWAtoolbox)
> gwasqc("GWASQC_script.txt")
```

8 Between-study comparisons

8.1 Comparing skewness and kurtosis of effect size distribution

Association of genetic markers with continuous or binary phenotypes is generally assessed by the use of linear models, where an effect estimate and its standard error are used to represent the evidence of the association. The effect estimates is usually represented by the beta coefficient of the linear regression model. For binary outcome, this summarized by the log(odds ratio) obtained from logistic regression models. Let's define with θ our parameter of interest, and with $SE(\theta)$ its standard error. Under the null hypothesis of no association, the distribution of $\theta/SE(\theta) \sim N(0,1)$.

In checking the quality of GWA results, we are interested in assessing potential errors arisen during the analytical process or during the file management process. These errors could origin $\theta/SE(\theta)$ distributions that are systematically biased towards positive (or negative) values, or that are over/under-dispersed. The *kurtosis* and the *skewness* indices are the natural candidates to perform this kind of assessment. Convenient graphical display based on these two indices enables the contemporary plot and comparison of all the studies involved in the meta-analysis, with consequent identification of studies that are systematically different from each other.

Under the forms proposed by Cramer [3], the kurtosis and the skewness indices can be defined as $ku = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^2} - 3$ and $sk = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^3}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^{\frac{3}{2}}}$. The skewness index assesses the symmetry of a distribution around its central value, and the kurtosis index assesses the dispersion of the distribution around its central value. If $\theta/SE(\theta) \sim N(0,1)$, then for large sample sizes, $sk_\theta = sk\big(\theta/SE(\theta)\big) \to 0$ and $ku_\theta = ku\big(\theta/SE(\theta)\big) \to 0$ (Fisher [4]; Joanes and Gill [5]).

In a GWAS setting, we can assume that the 50% of SNPs with largest p-value are not associated with the phenotype of interest and so, they can be used to represent the null situation. Notice that the 50% of SNPs with worst p-values correspond to the concept of the genomic control inflation factor, lambda, which is estimated based on the median chi-square distribution of the p-values' quantiles.

In real world application, distribution of $\theta/SE(\theta)$ for 50% worst SNPs is not normally distributed, because SNPs are not independent. This situation is more pronounced in presence of genotype imputation, with an excess of values that are too close to 0. For this reason, it is not realistic to expect that $ku_\theta \to 0$: the distribution is leptokuric, and so $ku_\theta > 0$. However, as long as the studies involved in the meta-analysis used similar imputation reference platforms, ku_θ should be similar for all studies. For what concerns skewness, there is no good reaseon why sk_θ of the 50% worst SNPs shouldn't approximate 0.

Then, for given K studies, we can estimate $ku_{\theta,k}(50)$ and $sk_{\theta,k}(50)$ for $k = 1, \ldots, K$, and we can plot the two vectors $sk_{\theta,[1,\ldots,K]}(50)$ vs. $ku_{\theta,[1,\ldots,K]}(50)$ in a Cartesian diagram, with every point representing a study. We expect all points (studies) to cluster around the same point at $sk = 0$, with similar kurtosis value. Points (studies) that show strong departures from the main cluster could be submitted to detailed investigation in order to detect the reason of such discrepancy. In general, departures along the sk-axis are more serious than departures on the sk-axis, because the first ones will introduce systematic bias in the meta-analysis with one or a few studies that are systematically different from the others, with effects more often in one direction.

This diagnostic plot is shown in Figure 4 where the set of SNPs with 50% largest p-values is compared with other sets of SNPs with the largest 75%, 95%, and 99% p-values, respectively. Highlighted are points (studies) that are largely difference from the ones in the cluster. In the last scenario, all SNPs are included and the bias is given by the SNPs that are really associated with the phenotype. Outlier studies, that have been identified in the 50% scatterplot, are colored in red also in the other situations for comparison.

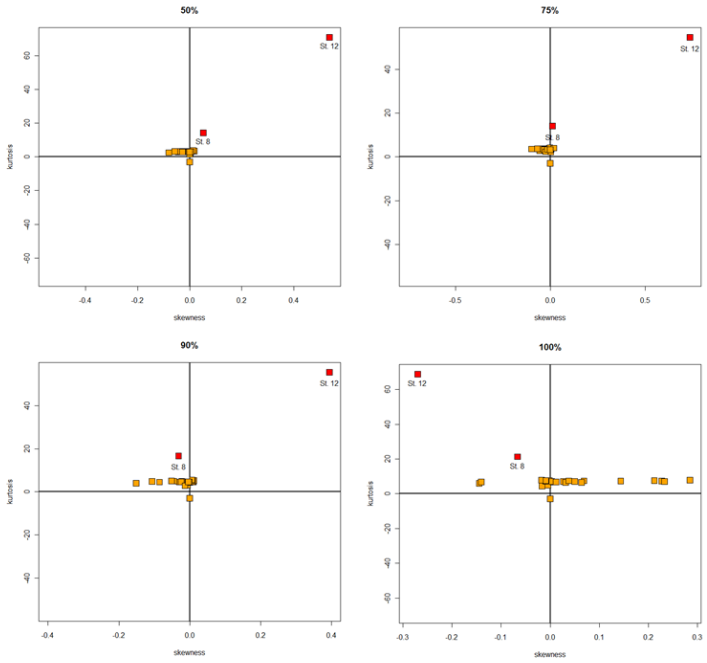

Figure 4: The skewness and kurtosis plot.

The *GWAtoolbox* allows automatic comparison of skewness and kurtosis of effect size distribution between GWA stuties. The *gwasqc()* function estimates the corresponding skewness and kurtosis values during the quality check and includes them into the *CSV* reports. Then, the auxiliary *kusk_check()* function can be used to export this information to the *R* data frame and to produce the diagnostic plots. As input it requires the same script as *gwasqc()* function and assumes that all the *CSV* reports are in the current working directory. Additionally, an optional list consisting of any of integer numbers 50, 75, 95, 99, and 100 can be specified. The numbers correspond to the percentage of SNPs to be considered as representing the null distribution.

Example 14 In this example we report the commands to obtain the scatterplot shown in Figure 4, when the 50% SNPs with largest p-values are considered. It can be seen that, currently, no automatic method to identify outlier studies is implemented and that the user needs to define his/her own criteria.

```
> library(GWAtoolbox)
> W <- kusk_check("GWASQC_script.txt", worst = c(50), plot = TRUE)
> points(W$sk50[W$ku50 > 5], W$ku50[W$ku50 > 5], pch = 22, bg = 2,
+     cex = 2)
> text(W$sk50[W$ku50 > 5], W$ku50[W$ku50 > 5], labels = W$study[W$ku50 >
+     5], cex = 1, pos = 4)
```

◁

8.2 Effect estimates precision by sample size

A different graphical test to compare studies against each other is based on the assessment of the distribution of estimates' precision vs. the study sample size. In general, the average $SE(\theta)$ should be inversely proportional to the study sample size. The auxiliary *dispersion_check()* function simply plots a scatterplot of the $mean(SE(\theta))$ vs. the median sample size of all studies, as depicted in the example Figure 5. Over-dispersion is defined as the presence of larger SEs than expected given the sample size of the study and under-dispersion is meant to be the opposite phenomenon. For example, a study with unmodeled relatedness or population stratification may present SEs that are smaller than another study of similar sample size where these issues were accounted for properly.

The *dispersion_check()* function uses *CSV* reports generated by *gwasqc()* function. As input it requires the same script as *gwasqc()* function and assumes that all the reports are in the current working directory. If the study sample size is missing from GWAS files, then the function has an optional parameter allowing to specify a vector with all study sample sizes. The function returns an *R* data frame with the information extracted from the *CSV* reports and produces the diagnostic plot.

Example 15

```
> library(GWAtoolbox)
> Z <- dispersion_check("GWASQC_script.txt", plot = TRUE)
```

```
> Z
```

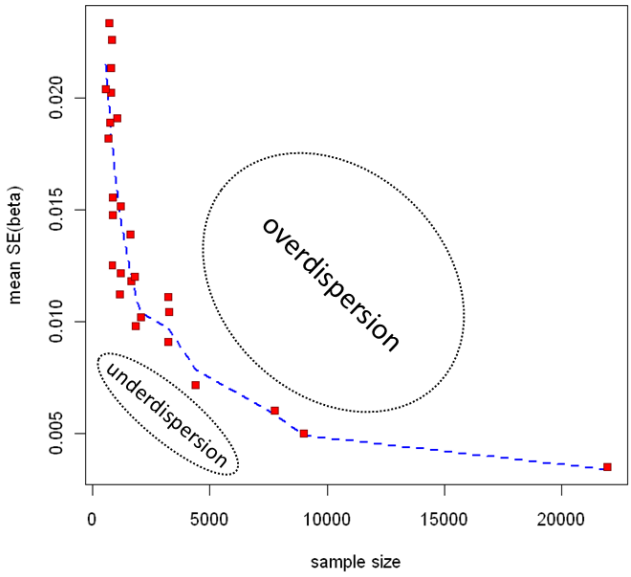

Figure 5: Schematic representation of the dispersion plot and its interpretation.

```
   study    mean_se median_n
1 Study1 0.01188491     1201
2 Study2 0.03206312      201
3 Study3 0.01270057     1000
4 Study4 0.02030638      437
5 Study5 0.01532013      721
```

```
> text(Z$median_n, Z$mean_se, labels = Z$study, pos = c(2, 4, 2,
+     1, 2))
```

```
> Z <- dispersion_check("GWASQC_script.txt", sample_sizes = c(1200,
+     200, 1000, 500, 700), plot = TRUE)
```

```
> Z
```

```
   study    mean_se median_n
1 Study1 0.01188491     1200
2 Study2 0.03206312      200
3 Study3 0.01270057     1000
4 Study4 0.02030638      500
5 Study5 0.01532013      700
```

```
> text(Z$median_n, Z$mean_se, labels = Z$study, pos = c(2, 4, 2,
+     1, 2))
```

◁

9 Additional Tools

In addition to the facilities for the data QC, *GWAtoolbox* also includes facilities including routines for the data manipulation after the GWAS meta-analysis has been done and functions that facilitates result annotation. These extensions are currently being debugged and will be made available soon.

References

[1] Cristen J. Willer, Yun Li, and Gonçalo R. Abecasis. (2010) **METAL: fast and efficient meta-analysis of genomewide association scans**. Bioinformatics 26: 2190-2191.

[2] Paul I.W. de Bakker, Manuel A.R. Ferreira, Xiaoming Jia, Benjamin M. Neale, Soumya Raychaudhuri, and Benjamin F. Voight (2008) **Practical aspects of imputation-driven meta-analysis of genome-wide association studies**. Hum. Mol. Genet. 17: R122-R128.

[3] H. Cramer (1946) **Mathematical Methods of Statistics**. Princeton: Princeton University Press.

[4] R.A. Fisher (1930) **The moments of the distribution for normal samples of measures of departure from normality**. Proc. R. Soc. Series A 130:16-28.

[5] D.N. Joanes, C.A. Gill (1998) **Comparing Measures of Sample Skewness and Kurtosis**. J. Royal Stat. Soc. Series D (The Statistician) 47(1):183-189.

Index