

Methods to find genes in Mendelian diseases: software implementation

Iuliana Ionita-Laza

October 2, 2011

We provide C++ code that implements the method for the case-control and affected relative pairs designs discussed in the accompanying paper. We also provide R code to visualize the final results.

1 C++ Programs - Compute Analytical P-value

1.1 Affected individuals are all unrelated `software_mendelian_CC.cpp`

1.1.1 Input and Output Files

Input Files There are three input files:

1. **data.ped** is a standard pedigree file. The first 6 columns are:

- Family_ID
- Individual_ID
- Father_ID
- Mother_ID
- Gender - 1 for male/ 2 for female
- Phenotype - 1 for control/2 for case

The remaining columns encode the genotypes at individual SNV positions. Each genotype is coded in an additive fashion: 0, 1 or 2 (i.e. the number of minor alleles). See Table 1 for a toy example of a pedigree file with 6 SNVs and 10 individuals: 5 cases and 5 controls.

Note: We have an accompanying script to convert a VCF file format to a pedigree file format.

Table 1: Toy pedigree file: 10 individuals (5 cases and 5 controls), and 6 SNVs.

1	1	0	0	1	2	2	0	0	0	0	0
2	1	0	0	2	2	0	0	0	1	0	0
3	1	0	0	1	2	0	1	0	0	0	0
4	1	0	0	2	2	1	0	0	0	0	2
5	1	0	0	2	2	1	0	0	0	0	0
6	1	0	0	1	1	0	0	1	0	1	0
7	1	0	0	2	1	0	0	0	0	1	0
8	1	0	0	1	1	0	1	1	0	0	0
9	1	0	0	1	1	0	0	0	1	0	0
10	1	0	0	2	1	0	0	1	0	0	0

2. **genes.txt** contains information about the grouping of individual SNVs into units to be tested together, such as genes. Each line in this file corresponds to one gene and contains the name of the gene, the number of the first SNV in the gene, and the number of the last SNV in the gene. These start and end positions correspond to the ordering of SNVs in the pedigree file. For example for the pedigree file above we can have two genes, the first contains SNVs 1 and 2, and the second contains SNVs 3, 4, 5, and 6.

Table 2: File with start and end positions of each gene

Gene1	1	2
Gene2	3	6

3. **weights.txt** contains information about whether variants are present in previous public databases (dbSNP, 1000 Genomes) and also weights (regarding functional information) of individual SNVs. For each variant in the ped file there is one line in the weights.txt file.
 - i. The first number in each line signifies whether the variant has been discovered before. One possible coding is 0 (already discovered, hence should not be used), or 1 (new variant, hence should be considered).
 - ii. The second number can be a functionality score, and can be any positive real number. Usually, if only non-synonymous variants are tested then weights are 1 for non-synonymous variants and 0 for synonymous variants.

An example file is in Table 3.

Output Files There is only one output file: **results.txt**. This file contains information that is needed by the R code to visualize the results. Each line corresponds to one gene and contains the name of the gene, P-value for WS-R, P-value for WS-N, and S_{filter} (i.e. the number of affected individuals that carry, e.g., novel, non-synonymous variants), in this order.

Table 3: Example of weights.txt file for the ped file above with 6 SNVs.

1	1
0	0
1	0
1	1
0	1
0	0

1.1.2 Running the program

1. **Source Code.** The source code is in the file called: **software_mendelian_CC.cpp**.
2. **Compilation.** To compile the program type the following command (the GSL library is needed):

```
g++ -lm -lgsl -lgslcblas -o software_mendelian_CC software_mendelian_CC.cpp
```

3. **Execution.** The resulting executable can be run by typing:

```
./software_mendelian_CC nind nsnvs ngenes
```

where

- **nind** is the number of individuals (number of lines) in the pedigree file, data.ped.
- **nsnvs** is the number of SNVs in the pedigree file, data.ped.
- **ngenes** is the number of genes in the gene file, genes.txt.

1.2 Affected individuals are affected relative pairs software_mendelian_ARP.cpp

If data are available for affected relatives, affected relatives can be analyzed as well. The current version of the software only supports two affected relatives per family. The data for the two relatives in a family needs to be two consecutive lines in the pedigree file (this formatting requirement will be fixed soon).

The input and output files are similar as for the case-control design above, except that the pedigree file now contains affected relative pairs (of the same type) and unrelated controls. To compile and execute the program, the following steps are necessary:

1. **Source Code.** The source code is in the file called: **software_mendelian_ARP.cpp**.
2. **Compilation.** To compile the program type the following command:

```
g++ -lm -lgsl -lgslcblas -o software_mendelian_ARP software_mendelian_ARP.cpp
```

3. **Execution.** The resulting executable can be run by typing:

```
./software_mendelian_ARP nind nsnvs ngenes kin
```

where

- **nind** is the number of individuals (number of lines) in the pedigree file, data.ped.
- **nsnvs** is the number of SNVs in the pedigree file, data.ped.
- **ngenes** is the number of genes in the gene file, genes.txt.
- **kin** encodes the relationship between the two relatives in a pair. 1 for sibs, 2 for uncle-nephew, 3 for first cousins and 4 for second cousins.

4. **Running Time.** The provided C++ code should run very fast on any reasonable machine. Our applications to the three Mendelian diseases each took ≈ 45 seconds on a normal desktop.

2 R program - Graphical Display

The accompanying R code is in the file **graph_mendelian.R** and contains code to produce graphical output of the results (as Figure 2 in the manuscript, or Figure 1 below). In addition, there is code to calculate the Joint-Rank for each gene, and output the top genes according to their Joint-Rank value.

Input Files Two input files are necessary, one is the **results.txt** produced above, and a second one **genes_position_pvalues.txt** contains for each gene, the following three fields: (1) chromosome number, (2) a single number representing the gene position on chromosome (e.g. the mid-position), and (3) the P-value from the analytic test (column 3 in **results.txt**). This latter file is necessary to produce the manhattan-style plot.

The R code provided produces two figures side-by-side: the first one is a manhattan-style plot showing the $-\log_{10}(\text{P-values})$ for all genes. The function **mhtplot2.R** that is needed to produce the manhattan-style plot is also provided and needs to be loaded beforehand. The second figure shows the $-\log_{10}(\text{P-value})$ against the number of affected individuals that carry rare, novel variants (as in the filter-based approach) for each gene.

There is also code that allows for calculation of the Joint-Rank statistic for each gene, and also outputs the top (e.g. 20) genes according to their Joint-Rank value.

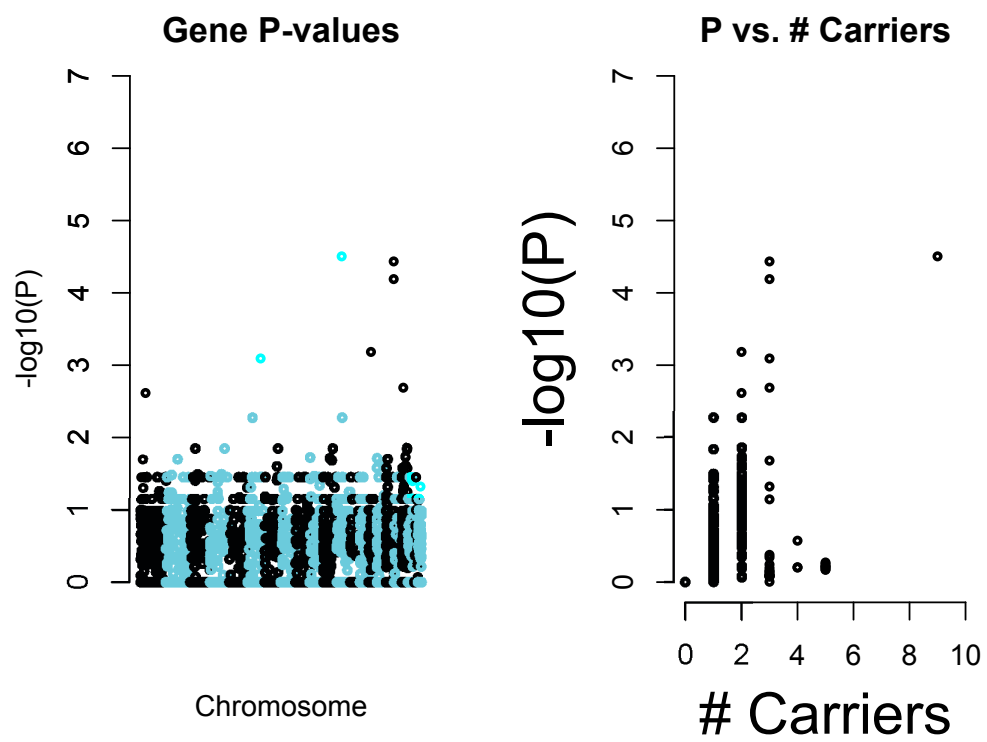


Figure 1: Kabuki Syndrome application.