INSTITUTE OF BIOINFORMATICS, ZHEJIANG UNIVERSITY

GWAS-GMDR User Manual

GWAS-GMDR 1.0 beta

Zhixiang Zhu

2011/8/29

This document describes how to use GWAS-GMDR to analyze GWAS data.

Table of Content

1 Introduction to GWAS-GMDR

1.1 What is GMDR?

Generalized Multifactor Dimensionality Reduction (GMDR) [3] is a nonparametric and genetic model-free alternative to linear or logistic regression for detecting nonlinear interactions among discrete genetic and environmental attributes. It begins with a training stage to search all attribute combinations to calculate their training accuracies. In training stage, the data are divided into multiple training sets. Then in each training set, the training accuracy of each attribute combination is calculated, and the attribute combination having the highest training accuracy is selected. After the training stage, GMDR enters the testing stage to perform permutation tests for those selected attribute combinations, to calculate their P values based on their testing accuracies. Compared with the MDR method[1][2], the GMDR method permits adjustment for discrete and quantitative covariates and is applicable to both dichotomous and continuous phenotypes in different population-based study designs.

1.2 What is GWAS-GMDR?

GWAS-GMDR is a command-line program which implements the GMDR algorithm specifically for analyzing Genome-Wide Association Analysis (GWAS) data. GWAS-GMDR assumes that all the attributes are SNPs, and accelerates the computation speed by the many-core parallel computation technique of graphics processing units (GPU), to overcome the memory and computation burdens of exhaustively searching among millions of SNPs, so that detecting gene-gene interactions in real GWAS data by GMDR becomes feasible. It also makes some improvements on the original GMDR algorithm: users can select multiple best SNP combinations in every training set instead of only one best SNP combination; a partial search approach is developed to detect high dimensional interactions in real GWAS data.

1.3 Overview of GWAS-GMDR

1.3.1 Hardware Requirement

- x86 or x64 architecture CPU
- GWAS-GMDR has no explicit requirement on the amount of memory. The bigger the data is, the more memory it is required to analyze the data. In practice,

GWAS-GMDR can load a data consisting of one million SNPs and five thousand subjects with 1.5GB RAM. Therefore, 1.5GB RAM is enough for most cases

- GWAS-GMDR does not necessarily require a GPU to run. However, if your analysis task meets one of the following conditions, it is highly recommended that you run GWAS-GMDR on a computer equipped with at least one CUDA-enabled GPU such as Tesla C2050:
  - The data consists of more than 5,000 SNPs
  - You want to perform 3 or higher dimensional exhaustive search
  - You want to perform more than 10,000 permutations for each selected SNP combination in testing stage

1.3.2 Software Requirement

- Operating System: Windows XP/Vista/7, Linux or Mac OS X (10.6.8 or later)
- On Windows operating system, you also need to install Microsoft Visual C++ 2008 Redistributable Package (x86), which can be downloaded from http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=29
- If you want to run GWAS-GMDR on GPU, you need to install NVidia CUDA 4.0 or higher driver. For Windows desktop or server environment, 270.81 or later NVidia driver is required; for Windows notebook environment, 275.33 or later NVidia driver is required. You can download the latest NVidia driver from http://www.nvidia.com/Download/index.aspx

1.3.3 Using GWAS-GMDR

The software package you downloaded and extracted should contain an executable file and several dynamic libraries. The executable file is **`gmdr`**, for Windows version it is **`gmdr.exe`**. The dynamic libraries are in the form of **`*.dll`** for Windows version, **`*.so`** for Linux version, and **`*.dylib`** for Mac OS X version.

Before using the software package, you need to add the path of the software package directory, i.e. the directory containing the executable file and the dynamic libraries, to the environment variable **PATH**. On Windows operating systems, you can do this in the following way:

1. Open the **System Properties** dialog. On Windows XP, you can do this by opening **Control Panel** -> **Performance and Maintenance** -> **System**, or right-clicking **My Computer** on desktop and choosing **Properties**. On Windows Vista/7, you can do this by clicking the **Start** button, right-clicking

the **Computer** item, and then clicking **Properties** -> **Advanced system settings**.

2. Click the **Advanced** tab of the **System Properties** dialog, and then click the button **Environment Variables**, as shown in Figure 1.

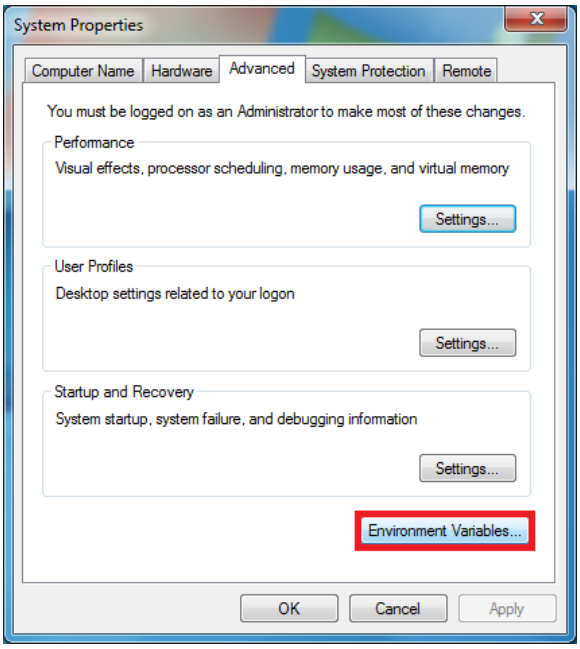

Figure 1 System Properties Dialog

3. Double click the **Path** variable in the **System variables** section of the pop-up **Environment Variables** dialog, as shown in Figure 2.

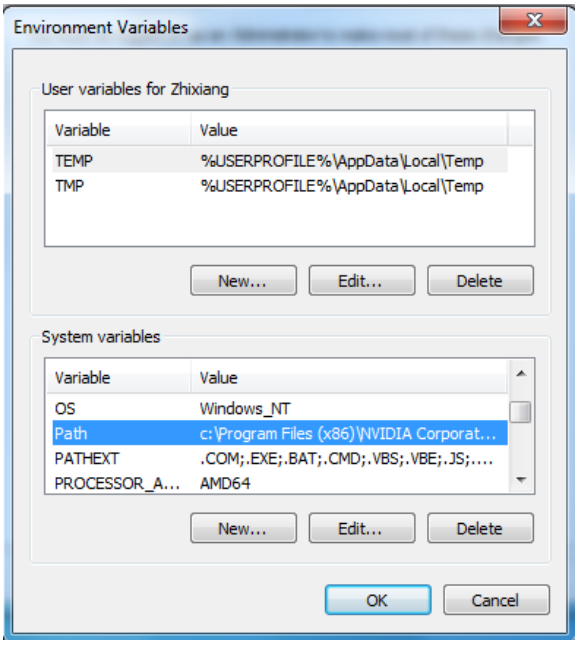

Figure 2 Environment Variables Dialog

4. In the pop-up **Edit System Variable** dialog, place the path of your GWAS-GMDR directory at the end of **Variable value**. For example, if you downloaded the GWAS-GMDR software package and extracted it as directory **`C:\gwasgmdr`**, then place **`C:\gwasgmdr`** at the end of **Variable value**, as shown in Figure 3. Notice that you should separate the old value of **Path** and the directory path you put in by a semicolon (;).

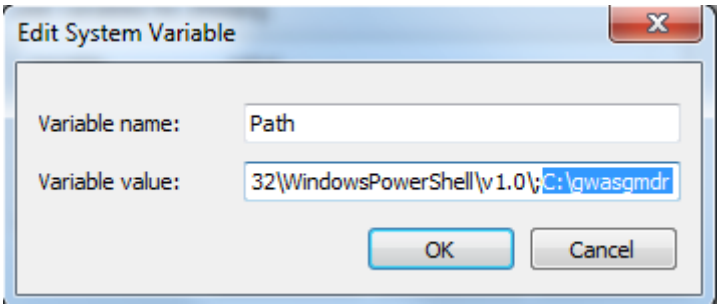

Figure 3 Edit the environment variable Path

5. Click OK button all the way to close the **System Properties** dialog.

On non-Windows operating systems, you also need to add the GWAS-GMDR directory to the environment variable **DYLD_LIBRARY_PATH** (on Mac OS X) or **LD_LIBRARY_PATH** (on Linux). Suppose that the GWAS-GMDR software package is extracted as **`~/gwasgmdr`**, the typical way to setup **PATH** and **DYLD_LIBRARY_PATH** is typing the following commands in your terminal window:

```
export PATH=~/gwasgmdr:$PATH
export DYLD_LIBRARY_PATH=~/gwasgmdr:$DYLD_LIBRARY_PATH
```

To make these settings permanent, place them in **`~/.bash_profile`**.

To use GWAS-GMDR, open a terminal window, type

```
gmdr -h
```

and press Enter, you will see a detailed description of the command-line options.

As a simple example, type

```
gmdr -d 2 -g Genotype.txt -p Phenotype.txt
```

and press Enter, GWAS-GMDR will perform two-dimensional exhaustive search for the data consisting of a genotype file **`Genotype.txt`** and a phenotype file **`Phenotype.txt`** in the current directory and display the result in the terminal window.

2 Data Formats

GWAS-GMDR supports two kinds of data format, which is native format and QTLNetwork-like format. The software package you downloaded and extracted should contain a subdirectory called **SampleData**, in which there are sample files demonstrating the data formats described here.

2.1 Native Format

A native format data consists of a combination of the following four files: genotype file, covariate file, phenotype file and score file. Genotype file is a necessary component of a native format data.

2.1.1 Genotype File

The text file **Native_Genotype.txt** contained in subdirectory **SampleData** is a demonstration of genotype file of a native format data. A genotype file is a white-space (space or tab) delimited text file, i.e. a table format file, consisting of three components:

The initial row of the genotype file is a group of space or tab delimited SNP names one for each SNP column.

The next two rows indicate the physical locations in the chromosomes of all the SNPs. Specifically, the second row of the genotype file is a group of space or tab delimited chromosome numbers one for each SNP column; the third row of the genotype file is a group of space or tab delimited base pair (bp) locations one for each SNP column. Notice that X chromosome and Y chromosome should be coded as 23 and 24, respectively. If either the chromosome number or bp location of a SNP is unknown, its chromosome number and bp location should both be set to 0.

Starting from the fourth row, every row records the genotypes of a subject (or person). The genotypes should be in biallelic form. All SNPs must have two alleles specified, where each allele is coded by a character (e.g. 1, 2, 3, 4 or A, T, C, G or anything else) except 0, which is the missing allele character. The two alleles of a genotype can be separated by one or more spaces or tabs, or not separated. The genotypes of different SNPs must be separated by spaces or tabs.

As an example, here are two subjects typed for 3 SNPs, which are SNP1, SNP2 and SNP3:

```
SNP1    SNP2    SNP3
1       23      0
123     500     0
A A     CG      1 2
A 0     GG      0 0
```

In the above example, SNP1 is located at 123bp of chromosome 1, SNP2 is located at 500bp of chromosome 23 (i.e. chromosome X), the physical location of SNP3 is unknown.

2.1.2 Covariate File

The text file **Native_Covariate.txt** contained in subdirectory **SampleData** is a demonstration of covariate file of a native format data. A covariate file must be provided if you want to correct for any covariate. The covariate file is also in table format. The first row is a group of space or tab delimited covariate names one for each covariate. Starting from the second row, every row records the covariate values of a subject and is a group of space or tab delimited integers or floating point numbers. Notice that missing covariate values are not allowed.

As an example, here are two subjects investigated for three covariates:

```
Sex     Age     Height
1       72.1    1.740
2       40.5    1.687
```

2.1.3 Phenotype File

The text file **Native_Phenotype_Quantitative.txt** contained in subdirectory **SampleData** is a demonstration of phenotype file of a native format data in which the phenotype is quantitative; while the text file **Native_Phenotype_Dichotomous.txt** contained in subdirectory **SampleData** is a demonstration of phenotype file of a native format data in which the phenotype is dichotomous. A phenotype file consists of only one column with NO HEADER. Every row is an integer or a floating point number which is the phenotype value of a subject. Notice that if the phenotype is dichotomous, i.e. case-control style, then cases should be coded by 1, and controls should be coded by 0. Also, missing phenotype values are not allowed.

2.1.4 Score File

Usually users don't have to care about score[3] files. But those who are experts on GMDR methodology may implement their own link functions[3] to calculate scores from the covariates and phenotypes. In this way, they should provide a score file. Also, if you are using PGMDR[4] to perform family studies, you will also need to provide the score file generated by PGMDR. The format of a score file is the same as that of a phenotype file. It contains only one column with NO HEADER. Therefore, if your score file is generated by PGMDR, you should remove the header in the generated file. Every row is an integer or a floating point number which is the score value of a subject. Missing score values are not allowed.

NOTE:

1. Native format is the only way to provide scores directly by users to GWAS-GMDR.
2. You should not provide any covariate or phenotype file if a score file is provided.

2.2 QTLNetwork-Like Format

A data in QTLNetwork-like format consists of a map file and a data file, or just a data file if all the SNP locations are unknown.

2.2.1 Map File

The text file **`QTLNetwork_Map.txt`** contained in subdirectory **`SampleData`** is a demonstration of map file of a QTLNetwork-like format data. A map file of a QTLNetwork-like format data provides the SNP locations in the data. It begins with a line which itself begins with the word `_MarkerNumbers`, followed by one or more positive integers indicating the number of markers (SNPs) on each chromosome. The order of the numbers must be consistent with that of the SNP location columns in the map body, which follows the `_MarkerNumbers` line and is described below.

The map body follows the `_MarkerNumbers` line, starts with a line whose content is `*MapBegin*` and ends with a line whose content is `*MapEnd*`. The content between `*MapBegin*` and `*MapEnd*` is arranged by columns which are delimited by white-spaces. The leftmost column begins with the header `#Marker`. Following the header `#Marker` in the leftmost column are the orders of SNPs on each chromosome. The maximum order is equal to the number of markers on the chromosome that has the most markers among all the chromosomes. Each of the other columns begins with a header in the form of `Ch<n>` indicating the chromosome number, where `<n>` is a non-negative integer. Notice that chromosome X and Y should be named as Ch23 and Ch24, respectively. Following the header `Ch<n>`

are the physical (base pair) locations of all the SNPs on that chromosome. If there's any SNP whose location is unknown, they should be grouped into the column whose header is Ch0, and all their base pair locations should be coded as 0.

As an example, the following map file content shows the locations of 9 SNPs, of which 4 SNP locations are unknown, 2 SNPs are on chromosome 1, and 3 SNPs are on chromosome 8:

```
_MarkerNumbers 4   2   3
*MapBegin*
#Marker    Ch0    Ch1    Ch8
   1       0      1200   1000
   2       0      1450   2000
   3       0             3500
   4       0
*MapEnd*
```

Besides the `_MarkerNumbers` line and the map body, you can add other content as comment in the map file. GWAS-GMDR will ignore them. But notice that the comment you add should not begin with `_MarkerNumbers` or `*MapBegin*`.

2.2.2 Data File

The two text files **`QTLNetwork_Data_MK.txt`** and **`QTLNetwork_Data_SUB.txt`** contained in subdirectory **`SampleData`** are two demonstrations of data file of a QTLNetwork-like format data. A data file contains genotype, covariate and phenotype information of the population sample. It consists of two necessary parts, which are the marker body and the trait body; and one optional part, which is the covariate body. The three parts can appear in any order. Each of them must appear no more than once.

Marker Body

The marker body begins with a line whose content is `*MarkerBegin*`, and ends with a line whose content is `*MarkerEnd*`. The content between `*MarkerBegin*` and `*MarkerEnd*` is a table whose cells are delimited by white-spaces. The last cell of each row must be a semicolon (;).

If the first cell of the first row is `#Mk`, then following `#Mk` in the first row are the SNP names one for each SNP column, and following `#Mk` in the first column are the subject names one for each subject row. For the remaining cells, each row is the genotypes of a subject. The genotypes should be in biallelic form. All SNPs must have

two alleles specified, where each allele is coded by a character (e.g. 1, 2, 3, 4 or A, T, C, G or anything else) except 0, which is the missing allele character. The two alleles of a genotype can be separated by one or more spaces or tabs, or not separated. The genotypes of different SNPs must be separated by spaces or tabs. Notice that the order of the SNP columns must be consistent with the order of SNPs on each chromosome determined in the map file.

As an example, here are two subjects typed for 9 SNPs recorded in a marker body:

```
*MarkerBegin*
#Mk        SNP1  SNP2  SNP3  SNP4  SNP5  SNP6  SNP7  SNP8  SNP9    ;
Subject1   Aa     A C   T G    11    0 0   TA    12    2 2   A 0     ;
Subject2   AA     C C   T G   1 2   CG    TT    22    1 1   A A     ;
*MarkerEnd*
```

If the above marker body example is provided with the map file example shown in section 2.2.1, then the locations of SNP1~SNP4 are unknown, SNP5 is located at 1200bp of chromosome 1, SNP6 is located at 1450bp of chromosome 1, SNP7 is located at 1000bp of chromosome 8, SNP8 is located at 2000bp of chromosome 8, and SNP9 is located at 3500bp of chromosome 8.

If the first cell of the first row is not #Mk, then following #Mk in the first row are the subject names one for each subject column, and following #Mk in the first column are the SNP names one for each SNP row. For the remaining cells, each row is the genotypes of all the subjects of a SNP. The genotypes should be in biallelic form. All SNPs must have two alleles specified, where each allele is coded by a character (e.g. 1, 2, 3, 4 or A, T, C, G or anything else) except 0, which is the missing allele character. The two alleles of a genotype can be separated by one or more spaces or tabs, or not separated. The genotypes of different SNPs must be separated by spaces or tabs. Notice that the order of the SNP rows must be consistent with the order of SNPs on each chromosome determined in the map file.

As an example, here's another marker body recording the same genotype sample as the marker body example given before:

```
*MarkerBegin*
#Sub    Subject1    Subject2    ;
SNP1    Aa          AA          ;
SNP2    A C         C C         ;
SNP3    T G         T G         ;
SNP4    11          1 2         ;
SNP5    0 0         CG          ;
SNP6    TA          TT          ;
SNP7    12          22          ;
SNP8    2 2         1 1         ;
SNP9    A 0         A A         ;
*MarkerEnd*
```

Trait Body

The trait body begins with a line whose content is *TraitBegin*, and ends with a line whose content is *TraitEnd*. Between *TraitBegin* and *TraitEnd* is a two-row table, of which the cells are delimited by white-spaces. The first cell of the first row can be any word. Following the first cell in the first row are the subject names. The first cell of the second row is the trait name, or phenotype name. Following the first cell in the second row are the phenotype values of all the subjects, which should be integers or floating point numbers. Missing phenotype values are not allowed. If the phenotype is dichotomous, i.e. case-control style, then cases should be coded by 1, and controls should be coded by 0. Both the rows should end with a cell whose content is a semicolon (;).

As an example, here's a trait body recording the phenotypes of 3 subjects:

```
*TraitBegin*
SubjectName    Subject1    Subject2    Subject3      ;
TraitName      49.75       13.65       31.7          ;
*TraitEnd*
```

Covariate Body

The covariate body must be provided in the data file if you want to correct for any covariate. The covariate body begins with a line whose content is *CovariateBegin*, and ends with a line whose content is *CovariateEnd*. Between *CovariateBegin* and *CovariateEnd* is a table whose cells are delimited by white-spaces. All the rows must end with a cell whose content is a

semicolon (;). The first cell of the first row can be any word. Following the first cell in the first row are the subject names, one for each subject column; while following the first cell in the first column are the covariate names. For the remaining cells, each row records the covariate values of all the subjects of a covariate, which should be integers or floating point numbers. Missing covariate values are not allowed.

As an example, here's a covariate body recording 2 subjects investigated for 3 covariates:

```
*CovariateBegin*
SubjectName    Subject1    Subject2    ;
Sex            1           2           ;
Age            36          40          ;
Height         1.75        1.65        ;
*CovariateEnd*
```

Besides the marker body, the trait body and the covariate body, you can add any other content as comment in the data file. GWAS-GMDR will ignore the comment you add. But notice that the comment you add should not begin with *MarkerBegin*, *TraitBegin* or *CovariateBegin*.

3 Command-Line Options

3.1 GMDR Algorithm Parameters

-d <n> or --dim <n>

This option must be set explicitly by user. It sets the search dimension. **<n>** is a positive integer.

-w <file> or --low <file>

If this option is set, GWAS-GMDR will perform partial search based on the low dimensional interactions having been detected. **<file>** is the path to the result file generated by option **-o** or **--out** in a lower dimensional search run. If this option is not set, GWAS-GMDR will perform exhaustive search.

-i <n> or --dist <n>

If this option is set, any SNP combination containing a pair of SNPs whose physical distance (number of base pairs) is less than **<n>** is ignored in the search, where **<n>** is a non-negative integer.

--pair

If this option is set, then GWAS-GMDR assumes that the data is generated by PGMDR and the scores are thus pairwise. In this case, GWAS-GMDR will perform pairwise permutation test in the testing stage.

-a <n> or --partitions <n>

This option sets the number of data partitions, i.e. the number of training sets and testing sets, which is default to 10. **<n>** should be an integer bigger than 1.

-m <n> or --num-sel-each-train <n>

This option sets the number of best SNP combinations selected in each training set, which is default to 1. **<n>** is a positive integer.

-n <n> or --min-cvc <n>

This option sets the minimum CVC required to show in result, which is default to 1. **<n>** is a positive integer.

-u <n> or --perm-pow <n>

This option sets the number of permutations performed in testing stage for each selected SNP combination. **<n>** is a non-negative integer. If this option is set, then $10^{<n>}$ permutations will be performed for each SNP combination selected in training stage; otherwise 10^3=1,000 permutations will be performed for each SNP combination selected in training stage.

3.2 Computation Device Parameters

-r <n> or --seed <n>

This option sets the random number generator seed. **<n>** is an integer. If this option

is not set, the random number generator seed will be set based on the time when the program is launched.

--cpu

If this option is set, GWAS-GMDR will use CPU for all the computation, even if the computer is equipped with CUDA GPUs.

-e <n> or --dev <n>

This option sets which CUDA GPUs you want to use. **<n>** is a non-negative integer, which is a CUDA GPU ID. For example, if you want to use GPU 0 and GPU 1, then you should provide **-e 0 -e 1** in the command-line. You can use the **deviceQueryDrv** executable delivered within the package to query the CUDA GPUs installed in your computer.

NOTE: You cannot set **--cpu** and **-e/--dev** at the same time. If neither **--cpu** nor **-e/--dev** is set, GWAS-GMDR will first search for the CUDA GPUs installed on your computer and use all the CUDA GPUs it can detect for computation; if no CUDA GPU can be detected, GWAS-GMDR will use CPU for computation.

-b <n> or --blocks <n>

Most users don't need to care about this option. It's reserved for CUDA experts. It sets the number of CUDA blocks per CUDA grid of each CUDA device, which is default to 10 times the number of SMs; a value of 0 indicates the default value. **<n>** is a non-negative integer. Notice that **-e** or **--dev** option must be set if you want to set this option.

For example, if you provide **-e 0 -e 2** in the command-line to use device 0 and 2, and you want device 0 to launch 80 blocks per CUDA grid, and want device 2 to launch the default number of blocks per CUDA grid, then you can provide **-b 80 -b 0** in the command-line. In some rare cases, you may find the GPU doesn't have enough device memory to perform a specific analysis, in these cases you can lower down the memory consumption by setting this option to a smaller number. But notice that fewer CUDA blocks may result in slower running speed.

-t <n> or --threads <n>

Most users don't need to care about this option. It's reserved for CUDA experts. It sets the number of CUDA threads per CUDA blocks of each CUDA device, which is default to 128; a value of 0 indicates the default value. **<n>** is a non-negative integer.

Notice that **-e** or **--dev** option must be set if you want to set this option. For example, if you provide **-e 0 -e 2** in the command-line to use CUDA device 0 and 2, and you want GPU 0 to launch 64 threads per CUDA block, and want GPU 2 to launch default number of blocks per CUDA grid, then you can provide **-t 64 -b 0** in the command-line. In some rare cases, you may find the GPU doesn't have enough device memory to perform a specific analysis, in these cases you can lower down the memory consumption by setting this option to a smaller number. But notice that fewer CUDA threads may result in slower running speed.

3.3 Data Options

-g <file> or --geno <file>

If your data is in native format, you should use this option to provide the genotype file. **<file>** is the path to the genotype file.

-c <file> or --cov <file>

If your data is in native format and you want to correct for covariate in the analysis, you should use this option to provide the covariate file. **<file>** is the path to the covariate file.

-p <file> or --pheno <file>

If your data is in native format and no score file is provided, you should use this option to provide the phenotype file. **<file>** is the path to the phenotype file.

-s <file> or --score <file>

If your data is in native format and you want to provide the scores directly, you should use this option to provide the score file. **<file>** is the path to the score file. Notice that you should not provide any covariate or phenotype file if this option is set.

--qtlnetwork-map <file>

If your data is in QTLNetwork-like format and there's a map file, you should use this option to provide the map file. **<file>** is the path to the map file.

--qtlnetwork-data <file>

If your data is in QTLNetwork-like format, you should use this option to provide the data file. **<file>** is the path to the data file.

NOTE: Do not mix the data options related to different data formats together!

3.4 Result Output Options

-o <file> or --out <file>

This option sets to which file you want to write the result. **<file>** is the path to the file you want to write. If this option is not set, result will be written to the standard output, i.e. the terminal window. The result is displayed in ALL-VERTICAL format, in which all the information, including the selected SNP combinations and their related statistics, are displayed in vertical direction.

--out-snp-only-horizontal <file>

This option outputs the result to a file in SNP-ONLY-HORIZONTAL format, in which only the SNP names contained in the selected SNP combinations are displayed, each selected SNP combination is displayed in horizontal direction, i.e. a line. This format is easy for other programs to parse. **<file>** is the path to the file you want to write.

NOTE: Do not set different result output options to the same file!

3.5 Other Options

-h or --help

Show the help messages.

4 Example Runs

gmdr -d 2 -e 0 -e 1 -e 2 -g genotype.txt -c covariate.txt -p phenotype.txt -o result1.txt

Use CUDA GPU 0, 1 and 2 to exhaustively search for 2D interactions. The data is in native format. Genotypes are provided by **genotype.txt**. Phenotypes are provided

by **phenotype.txt**. Correct the covariates in **covariate.txt**. Output the result to **result1.txt**.

gmdr -d 3 -w result1.txt -e 1 -g genotype.txt -c covariate.txt -p phenotype.txt

Use CUDA GPU 1 to partially search for 3D interactions based on the low-dimensional interactions written in **result1.txt**. The data is in native format. Genotypes are provided by **genotype.txt**. Phenotypes are provided by **phenotype.txt**. Correct the covariates in **covariate.txt**. Output the result to the terminal window.

5 References

[1] Ritchie MD, Hahn LW, Roodi N, Bailey LR, Dupont WD, Parl FF & Moore JH. Multifactor-Dimensionality Reduction Reveals High-Order Interactions among Estrogen-Metabolism Genes in Sporadic Breast Cancer. *Am J Hum Genet* (2001) **69**: pp. 138-147.

[2] Moore JH. Computational analysis of gene-gene interactions using multifactor dimensionality reduction. *Expert Review of Molecular Diagnostics* (2004) **4**: pp. 795-803.

[3] Lou XY, Chen GB, Yan L, Ma JZ, Zhu J, Elston RC & Li MD. A generalized combinatorial approach for detecting gene-by-gene and gene-by-environment interactions with application to nicotine dependence. *Am J Hum Genet* (2007) **80**: pp. 1125-1137.

[4] Lou XY, Chen GB, Yan L, Ma JZ, Mangold JE, Zhu J, Elston RC & Li MD. A Combinatorial Approach to Detecting Gene-Gene and Gene-Environment Interactions in Family Studies. *Am J Hum Genet* (2008) **83**: pp. 457-467.