

# Manhattan Harvester (MH) 0.1 Documentation

Website: [www.geenivaramu.ee/en/tools/manhattan-harvester](http://www.geenivaramu.ee/en/tools/manhattan-harvester)

Contact: [toomas.haller@ut.ee](mailto:toomas.haller@ut.ee), [toomashaller@gmail.com](mailto:toomashaller@gmail.com)

Reference: *manuscript in preparation*

## **CONTENTS**

I. INTRODUCTION – p. 2

II. HOW TO COMPILE THE PROGRAM – p. 2

III. USER GUIDE – p. 3

3.1. Input file requirements – p. 3

3.2. General workflow – p. 3

TABLE 1: Parameters output by MH – p. 4

3.3. Detailed instructions – p. 5

3.4. Error handling – p. 6

IV. EXAMPLES – p. 6

## I. INTRODUCTION

Manhattan Harvester (MH) is a command line tool for analyzing GWAS results – the Manhattan Plots. It uses the physical position and p-value as input to detect the regions that correspond to Manhattan Plot peaks. It finds the peak borders and height of all peaks and computes an array of parameters for each peak (including a General Quality Score) to allow filtering during runtime or after the runtime as decided by the user. MH comes in handy in situations where the number of GWAS files is too large for evaluation by the human eye.

## II. HOW TO COMPILE THE PROGRAM

You can download the program and instructions from the website (see above). The statically compiled versions are ready to run, dynamically compiled versions require that Qt libraries are installed on your system (Qt 4.3 or newer and gcc 5.2 or newer). Qt libraries are free and can be downloaded and installed from <http://www.qt.io/download/>. The GCC compiler can be downloaded from <https://gcc.gnu.org/>.

In order to compile MH from the source code, place the source code in the folder named “HARVESTER” on the Qt path and execute the following lines (please read all notes below):

```
> qmake -project
> qmake
> make
```

**Note1:** Harvester requires C++11. After you have made the Make file (typed “qmake”), please add ‘-std=c++11’ after the flag CXXFLAGS. You may also need to add ‘-lz’ after LIBS on some systems.

**Note2:** If you are compiling for Windows, open your \*.pro file after typing “qmake -project” and add this line to the \*.pro file:  
**CONFIG+=console**  
**INCLUDEPATH += C:\Qt\4.4.3\src\3rdparty\zlib** (where C:\Qt\4.4.3\ should be replaced with the correct path)

Your \*.exe file will appear in the “release” subfolder.

**Note3:** If you are compiling for Mac, you may need to have Xcode installed (<https://developer.apple.com/xcode/>) if your computer doesn't already have it. Before you install, look under “/Applications” to see if “/Applications/XCode” exists.

If you installed Qt on a Mac using binaries, the program should be located here:  
“~/QtSDK/Desktop/<qt.version>/<compiler>/bin”, if you built it from source code, it should be here:  
“/usr/local/Trolltech/<qt.version>/bin”.

**Note4:** If you are compiling for Mac you need to execute the following lines in order to compile using g++:  
**qmake -project**  
**qmake -spec macx-g++**  
**make**

After you have created your Makefile (before typing “make”) you may need to add “-lz” after “LFLAGS=” in your Makefile.

**Note5:** If you are compiling statically, open your \*.pro file after typing “qmake -project” and add this line to the \*.pro file:  
**CONFIG+=static** or **CONFIG+=staticlib**  
Static compilation is an option only if Qt itself was built statically.  
In your Makefile please add ‘-static-libstdc++’ and ‘-static-libgcc’ behind LIBS.

**Note6:** If you are using Qt5 to compile, you need to add “`QT += widgets`” to your .pro file (after typing “`qmake -project`”)

If you have any problems compiling or running MH, please do not hesitate to contact the author at [toomashaller@gmail.com](mailto:toomashaller@gmail.com) for help.

### III. USER GUIDE

#### 3.1. Input file requirements

MH can read any tabular file that contains chromosome number, physical position and p-value in a tabular format. Any file delimiter is supported file header is optional. The data columns can be in any order and total number of columns can be anything as the columns are detected by their index, values provided by the user. Generally a GWAS output file can be directly used as a MH input. It is required that the physical position values are all valid numbers and in increasing order. P-value column can have missing values.

#### 3.2. General workflow

MH starts by reading the values below a p-value threshold into memory. The recommended p-value threshold is 0.001 and this is the default. Internally all values are converted to  $-\log(P)$ . Subsequently the plot is smoothed by replacing each value with a new value as decided by performing linear regression with 5 data points – 2 before and 2 after the point to be updated. Next the distances between neighboring data points are shrunk based on the average  $-\log(P)$  of the neighboring data points times a constant (2 is the default). For example if the  $-\log(P)$  values are 3 and 7, the distance is computed by dividing the old distance by 10 ( $(3+7)/2 * 2 = 10$ ). After this the p-values are dissociated from the positions and the algorithm continues only with the one dimensional array of position values – essentially a projection of the data onto the position axis. This step is followed by relocating each data point precisely between its neighbors (point centering). This is done so that each point to be moved is re-positioned relative to two stable data points (the ones not redistributed in the same iteration round of redistribution). All these steps result in a one-dimensional array of more and less dense areas. The dense areas correspond to peaks. The peaks are extracted from these arrays by performing an iterative fragmentation of this array. The array is continually fragmented to smaller fragments starting from the least dense region and every time the half with the larger number of data points is carried over to the next round for subsequent fragmentation. The fragmentation is terminated when certain criteria are met (see publication). The detected peak is characterized by turing to the original data set (the one not smoothed, compressed and re-distributed) and the points corresponding to the detected peaks are removed both from the original and the modified data sets. At this point the one dimensional vector fragmentation starts again but using the data set that has the previously detected peak removed. The user can set filters to determine what peaks are listed in the output. The peaks that are listed are done so together with a variety of peak characteristics (Table 1) to let the user perform their own filtering for isolating potentially interesting peaks.

**TABLE 1: The parametrs output by the MH**

<b>filename</b>	name of the input file
<b>chrom</b>	chromosome number/name
<b>range</b>	range of the peak (in absolute position units)
<b>max</b>	maximal $-\log P$ value of the peak detected
<b>bestslope0</b>	highest absolute slope value of the peak (position vs. $-\log P$ ) computed using 5-point sliding window
<b>bestslope</b>	same as above but each slope value is multiplied by its corresponding $-\log P$ value
<b>monot</b>	peak monotony (in fractional units) computed relative to the assumption that $-\log P$ values should be monotonously increasing from start to peak and monononously decreasing from peak to end
<b>balance</b>	peak balance; number of points to the left of the $\max(-\log P)$ divided by the number of points to the right of the $\max(-\log P)$ ; non-linear with respect to peak quality
<b>multip</b>	peak multiplicity; indicates what fraction of points have $-\log P = \max(-\log P)$
<b>reps</b>	peak repetitions; indicates what fraction of points have a $-\log P$ equal to that of their neighbor
<b>ratio</b>	height to width ratio calculated as $-\log P * 10^6 / \text{width}$
<b>kolmo</b>	Kolmogorov-Smirnoff test of normality; 0 = normality condition not satisfied, 1= normality condition satisfied; not suitable for comparing peaks of very different size
<b>range</b>	width of the peak in bp
<b>count</b>	number of points in the peak
<b>spacing</b>	mean distance between points
<b>balance</b>	number of all points to the right of $\max(-\log P)$ divided by the number of points to the left of $\max(-\log P)$ ; non-linear with respect to peak quality
<b>skew</b>	all $-\log P$ values to the right of $\max(-\log P)$ divided by all $-\log P$ values to the left of $\max(-\log P)$ ; non-linear with respect to peak quality
<b>vbal1</b>	number of data points with $-\log P$ values $\geq$ $\text{mean}(-\log P)$ divided by number of data points with $-\log P$ values $<$ $\text{mean}(-\log P)$
<b>vbal2</b>	same as above but instead of point counts the cumulative $-\log P$ values are used
<b>maxbyMean</b>	$\max(-\log P)$ divided by $\text{mean}(-\log P)$
<b>GQS</b>	General Quality Score of the peak (1-5)

### 3.3. Detailed instructions

Manhattan harvester makes use of a list of flags that control how the program operated. Each value has a default setting. If the corresponding flag is used the default setting takes effect. The default settings are optimized for a typical GWAS performed on a human genome.

Flags:

#### **-file**

This is the input file name. REQUIRED if “-list” is not specified

#### **-list**

This is the list file that contains the names of all files (one per row) that are analyzed in batch mode. REQUIRED if “-file” is not specified. If both “-list” and “-file” are specified the “-file” tag is simply ignored.

#### **-out**

This is the output file name. If this is omitted the output file name is generated from the input file name by appending the suffix “.out”.

#### **-delim**

Used to define file delimiter. Default = tab. Use here any file delimiter text. This is the list of predefined delimiters: tab, space, comma, semicolon, colon, backslash (means backslash), slash, dash, quote, squote (means single quotes). Example: “-delim comma “

#### **-header**

Define if the file has a header. Default is no header. Options: yes, no. Example: “-header yes”.

#### **-header-size**

The size of the header (default=1). Use this if your header is several rows.

#### **-chrcolumn**

The index of the chromosome column (first column is 1). REQUIRED

#### **-lcolumn**

The index of the physical location column (first column is 1). REQUIRED

### **-pcolumn**

The index of the pvalue column (first column is 1). REQUIRED

### **-missing**

The missing value. Fefault is "NA". Example "-missing NOVALUE"

### **-inlimit**

The pvalue below which the values are considered by MH. The dafault is 0.001. MH is optimized to work with this default value. We do not recommend to change this without a good reason.

### **-shrink**

The compression constant used in the compression step of the algorithm. Default value is 2, we do not recommend to change this without a good reason.

### **-dots**

The number of points each peak needs to contain in order to get reported. Default value is 5

### **-peak-limit**

The minimal height of the peak (in -log10 scale) to be reported. Default value is 5.

## **3.4. Error handling**

MH catches most common errors and provides error messages in standard out.

## **IV. EXAMPLES**

An example file (example.txt) for practicing MH is provided together with the MH download. You can test MH options by varying the input parameters after the flags.

### Minimal example:

```
HARVESTER -chrcolumn 1 -lcolumn 2 -pcolumn 3 -header yes -file example.txt
```

### Full example:

```
HARVESTER -chrcolumn 1 -lcolumn 2 -pcolumn 3 -header yes -header-size 1 -file example.txt -out
```

results.txt -delim tab -missing NA -inlimit 0.001 -peak-limit 5 -dots 5 -shrink 2

Batch mode (minimal) example:

HARVESTER -chrcolumn 1 -lcolumn 2 -pcolumn 3 -header yes -list mylist.txt

(mylist.txt is a list of filenames to be analyzed in batch mode)

***Thank you for reading the Manhattan Harvester user guide!  
Looking forward to your feedback.***