

HaploBlockFinder Version 0.7

A software package for analyses of haplotype block structure.

Web-base Program: <http://cgi.uc.edu/cgi-bin/kzhang/haploBlockFinder.cgi>

Codes for standalone program: <http://cgi.uc.edu/~kzhang/HaploBlockFinder-0.7.zip>

LICENSE

HaploBlockFinder is distributed under the GNU GPL license. It can be freely distributed and modified (I hope to be informed in this case), but WITHOUT ANY WARRANTY.

INSTALLATION

The codes were written with ANSI C, and has been successfully compiled under several operating systems, including Tru64 Unix, Solaris, Linux and Windows.

Here are two examples:

(1) Tru64 Unix with cc:

```
cc HaploBlockCore.c HaploBlockFinder.c HaploBlockReport.c -o haploBlockFinder -lm -fast
```

(2) Other Unix with cc:

```
cc HaploBlockCore.c HaploBlockFinder.c HaploBlockReport.c -o haploBlockFinder -lm -O3
```

(3) Unix with gcc:

```
gcc HaploBlockCore.c HaploBlockFinder.c HaploBlockReport.c -o haploBlockFinder -lm -O3
```

The package also includes three perl scripts for format conversion and diagram generation, and another one to run the program on a web server. In order to use the drawBlocks.pl and the showLDmatrix.pl to draw a graphical diagram, you need Perl and two perl modules, GD and Getopt::Long. In windows, you may download ActivePerl from <http://www.activestate.com/>, install Perl and install the GD and Getopt::Long using the ppm3 program distributed with ActivePerl. For Unix system, you need to download the GD and Getopt::Long module from <http://www.cpan.org>, or install it using the cpan utility. In some Linux system, the default Perl might have some problems, in which case the Perl need to be recompiled with the latest source code. For the detailed usage of these perl script, read the documentation imbeded in the perl scripts using the perldoc utility.

HOW TO USE IT

To run haploBlockFinder, a text file contains haplotype data is required as input. Haplotypes should be in a FASTA like format as below:

```
>haplotype1          <---- name of haplotype, less than 50 characters
AGTCGTGTGTCTGTGCTGTG      <-----haplotype
```

Haplotypes can be represented as base code (A,C,G,T,N,I [insert],D [deletion]) or numerical code (0, 1, 2, 3...9; 0 represents ambiguous call in this case). An example haplotype files "sample_haplotypes_fasta.txt" is included. An optional file contains locus ID and locus position can be also submitted to the program, so that some statistics, like average block size can be easily calculated from the report files. This optional file should have only two column: The first one is the locus ID, and the second one is the locus position (bp). If this position file is not provided, the program assumes a default density of 1 SNP/Kb. There should be NO HEADER line in the file. See "locusInfo.txt" for example.

A note on haplotype format: Starting from Version 0.4, haplotype data in a format, such as the one generated by **Merlin** (see sample_haplotypes_Merlin.txt), is accepted. Just run the wrapper program [hbfWrapper.pl](#) instead of the haploBlockFinder (see Example #5). From Version 0.6, the web-based haploBlockFinder supports the haplotype file generated by **Arlequin** and **PHASE**. (If you wish to use the downloaded program on your local machine, there are two subroutines in the haploBlockFinder.cgi, called Arlequin2Fasta and Phase2Fasta, for reformatting, which can be easily pulled out and used as a standalone program).

Arguments accepted by this program are:

- A: block identification algorithm
 - 1 Minimal |D'| range: A block is defined as a region in which pair-wise |D'| values between all SNPs pairs exceed a certain threshold
 - 2 Chromosomal coverage: A block is defined as a region in which certain percentage of chromosomes is represented by less than 4 common haplotypes
 - 3 Four-gamete test: A block is defined as a region in which no within block recombination is found based on the four-gamete test
 - 4 Minimal d^2 range: A block is defined as a region in which pair-wise d^2 values between all SNPs pairs exceed a certain threshold
 - 5 Minimal LLR range: A block is defined as a region in which pair-wise LLR values between all SNPs pairs exceed a certain threshold
 - 6 Minimal r^2 range: A block is defined as a region in which pair-wise r^2 values between all SNPs pairs exceed a certain threshold
- D: threshold value of LD measurements, default value is 0.8 for |D'|, 0.3 for d^2 and 1.0 for LLR. (Optional)
- C: threshold value of chromosome coverage, default value is 0.8. (Optional)
- Q: threshold value for the quality filter, default value is 0.5. The quality filter. (Optional)
 - masks loci that has ambiguous/total ratio exceeding the given threshold, since these loci tends to cause adverse effect on block partitioning.

- I: file that contains haplotypes
- L: file that contains locus name and position (Optional)
- O: directory for report files. (Optional)
- H: higher bound of minor allele frequency for the allele frequency filter (Optional)
- B: lower bound of minor allele frequency for the allele frequency filter (Optional)
- M: LD matrix is generated if set as 1. (Optional, required for showLDmatrix.pl)
- T: Identify htSNPs if set as 1. (Optional, could be slow if the number of SNPs in a block is large)
- P: portion of chromosomes that can be distinguished with htSNPs, default value is 0.8 (Optional)

Note on haplotype block finding based on chromosome coverage:

This program excludes ambiguous haplotypes (haplotypes with more than one "N") for the calculation of chromosome coverage. Besides, blocks that have over 50% of chromosomes with ambiguous haplotypes will not be evaluated.

Example 1:

```
[kzhang@cgi kzhang]$./haploBlockFinder -A1 -D0.9 -Isample_haplotypes_fasta.txt -LlocusInfo.txt -Orpt
```

To find blocks with minimal LD range using 0.9 as threshold, haplotype file is "sample_haplotypes_fasta.txt", and the locus informatio file is "locusInfo.txt", the report files

will be in the ./rpt directory. Note that this directory should be created first, otherwise the program won't run correctly.

Example 2:

```
[kzhang@cgi kzhang]$./haploBlockFinder -A2 -C0.7 -Isample_haplotypes_fasta.txt
```

To find blocks with chromosome coverage threshold as 0.7, haplotype file is "sample_haplotypes_fasta.txt". The report files will be in the current directory.

Example 3:

```
[kzhang@cgi kzhang]$./haploBlockFinder -A3 -Q0.6 -Isample_haplotypes_fasta.txt
```

To find blocks with four-gamete test, haplotype file is "sample_haplotypes_fasta.txt".

Loci that have over 60% of ambiguous calls are masked.

Example 4:

```
[kzhang@cgi kzhang]$./haploBlockFinder -A1 -D0.8 -Isample_haplotypes_fasta.txt -LlocusInfo.txt -B0.15 -M1 -T1
```

To find blocks with minimal LD range using 0.8 as threshold, haplotype file is "sample_haplotypes_fasta.txt", and the locus informatio file is "locusInfo.txt", common SNPs with minor allele frequency higher than 15% are selected, LD matrix is generated, and htSNPs are identified.

Example 5:

```
[kzhang@cgi kzhang]$./hbfWrapper.pl -A4 -D0.5 -Isample_haplotypes_Merlin.txt -M1
```

To find blocks with minimal d^2 range using 0.5 as threshold, haplotype file is in a format generated by Merlin, and the locus informatio file is "locusInfo.txt", LD matrix is generated.

HOW TO READ REPORTS

The program will generate report in text files (tab delimited), and two graphic files:

(1) **haploblock blocks.txt**: report the features and statistics for each blocks.

Details for each column:

(a)Block ID: Internal block ID

(b)Start Locus: The first locus of a block

(c)End Locus: The last locus of a block

(d)#loci: The number of loci within the block

(e)Start Pos: The starting chromosomal position (bp) of the block, represented by the position of the first loci

(f)End Pos: The ending chromosomal position (bp) of the block, represented by the position of the last loci

(g)Length: Length of the block

(h)minimal D': The minimum pair-wise D' value between any pair of loci with the block

(i)minimal d^2 : The minimum pair-wise d^2 value between any pair of loci with the block

(j)minimal r^2 : The minimum pair-wise r^2 value between any pair of loci with the block

(k)minimal LLR: The minimum pair-wise LLR (log likelihood ratio) between any pair of loci with the block

(l)Ho/He: The ratio between observed heterozygosity and expected heterozygosity. This measurement can represent

how diversity is reduced within a haplotype block. Please note that in this application

heterozygosity is calculated based on haplotype frequency NOT the average of heterozygosity

of all loci within the block. Here is an example of the calculation:

Suppose we have a three-locus block with three observed haplotypes:

AAA 65

TTT 29

TAA 2

Then the haplotype frequencies are:

$p(\text{AAA}) = 65/96 = 0.677$

$p(\text{TTT}) = 29/96 = 0.302$

$p(\text{TAA}) = 2/96 = 0.021$

$H_o = 1 - p(\text{AAA})^2 - p(\text{TTT})^2 - p(\text{TAA})^2 = 0.5495$

The measurement H_e is calculated assuming the three loci is in complete equilibrium.

In this case,

$$p(\text{AAA}) = p_1(\text{A}) * p_2(\text{A}) * p_3(\text{A}) = 0.677 * 0.698 * 0.698 = 0.3298$$

$$p(\text{AAT}) = p_1(\text{A}) * p_2(\text{A}) * p_3(\text{T}) = 0.677 * 0.698 * 0.302 = 0.1427$$

$$p(\text{ATA}) = p_1(\text{A}) * p_2(\text{T}) * p_3(\text{A}) = 0.677 * 0.302 * 0.698 = 0.1427$$

$$p(\text{ATT}) = p_1(\text{A}) * p_2(\text{T}) * p_3(\text{T}) = 0.677 * 0.302 * 0.302 = 0.0617$$

$$p(\text{TAA}) = p_1(\text{T}) * p_2(\text{A}) * p_3(\text{A}) = 0.323 * 0.698 * 0.698 = 0.1574$$

$$p(\text{TAT}) = p_1(\text{T}) * p_2(\text{A}) * p_3(\text{T}) = 0.323 * 0.698 * 0.302 = 0.0681$$

$$p(\text{TTA}) = p_1(\text{T}) * p_2(\text{T}) * p_3(\text{A}) = 0.323 * 0.302 * 0.698 = 0.0681$$

$$p(\text{TTT}) = p_1(\text{T}) * p_2(\text{T}) * p_3(\text{T}) = 0.323 * 0.302 * 0.302 = 0.0295$$

$$H_e = 1 - p(\text{AAA})^2 - p(\text{AAT})^2 - p(\text{ATA})^2 - p(\text{ATT})^2 - p(\text{TAA})^2 - p(\text{TAT})^2 - p(\text{TTA})^2 - p(\text{TTT})^2 = 0.8118$$

Finally, we have,

$$H_o/H_e = 0.5495/0.8118 = 0.6769$$

(1) # HtSNPs: Minimal number of tagging SNPs that can distinguish each haplotypes a block. Please refer to the log at 11-02-2002 regarding the detailed information of htSNP selection!

Note that in order to calculate the average block size, just load this file into a spreadsheet program,

add a new column which is defined as $\text{column}(f) - \text{column}(e) + 1$, and take an average of the new column.

Locus information file is required in this case so that the program knows chromosomal position for each locus.

(2) **haploblock_haplotypes.txt**: report all haplotype for each block. The haplotypes are presented in two format: (i) haplotype with original coding; (ii) haplotypes with internal numerical coding, where 0 represents ambiguous call, 1 represents the allele with higher frequency, and 2 represents the rare allele. The number of chromosomes represents by each haplotype is also reported.

(3) **haploblock_map.txt**: report the position of blocks.

(4) **haploblock_chromosomes.txt**: A report summarizes all haplotypes in the input file, and show these haplotypes in block, as well as using HtSNPs only.

(5) **blocks.png**: a graphical diagram of haplotype blocks. Up to four common haplotypes are shown, and the colors represent their frequencies..

(6) **LD_matrix.png**: a graphical diagram to compare LD pattern with haplotype blocks. Please note that if the position of SNPs are not submitted to the haploBlockFinder program, this Perl script assumes all SNPs are evenly spaced.

In this diagram, the set of haplotype blocks identified by haploBlockFinder are shown on the top side and the left side. Short little white lines on the top and left represents the position of SNPs. The top right part of the matrix represent the Log Likelihood Ratio (LLR) between observed haplotype frequencies and expected haplotype frequencies under the assumption of equilibrium. The bottom left part of the matrix represent the pair-wise $|D'|$ values.

LLR indicates the significant level of association between two loci. It is calculated as below:

$$LLR = n_{AB} \cdot \log_{10}(P_{AB}/P_A \cdot P_B) + n_{Ab} \cdot \log_{10}(P_{Ab}/P_A \cdot P_b) + n_{aB} \cdot \log_{10}(P_{aB}/P_a \cdot P_B) + n_{ab} \cdot \log_{10}(P_{ab}/P_a \cdot P_b)$$

(7)**log.txt**: If you have problem in using the program on our website, read this file first!

History:

04-22-2004 The version 0.7 was released. There are two major changes in V0.7 compared with V0.6c:

- (1) One additional LD statistics r^2 was included for block searching.
- (2) The showLDMatrix.pl program has been rewritten, such that any combination of two out of the four LD statistics ($|D'|$, d^2 , r^2 , LLR) can be displayed in a single diagram.\

Besides, the report file haploblock_blocks.txt has one additional column for r^2 values.

04-16-2004 A bug that leads to incorrect color coding in the LD diagram for all LLR values was reported by Dr. Roger Vallejo (rvallejo@psu.edu). It has been fixed.

11-24-2003 A bug that leads to "segmentation fault" with the -T option is not used was identified and fixed.

05-19-2003 The version 0.6 was released, with a few changes:

(1) Two measurements for linkage disequilibrium (d^2 and LLR) were included for block searching, because there has been quite a number of reports showing that $|D'|$ might not be a good measurement for association study. LLR and d^2 were incorporated into this program so that it is very convenient to compare blocks defined by different LD measurements. However, I have not seen any publication using them for block definitions. So use them cautiously.

(2) Inferred haplotypes generated by the PHASE program is accepted by haploBlockFinder in this version (suggested by Dr. Cyrus Zabetian <zabetian@u.washington.edu>). Files generated by other haplotype inference program will be supported later (if I receive any request).

(3) The V0.6 program generated another report file (haploblock_chromosomes.txt) so that interpretation of results could be a little bit easier. (suggested by Dr. Kostya Krutovskii <kkrutovs@dendrome.ucdavis.edu>)

12-16-2002 A bug was identified, which could removed the last singleton block (a block

with one SNP only). The problem was fixed.

11-02-2002 (1)The major difference between version 0.4 and version 0.5 is the selection of htSNPs. The goal of selecting htSNPs is to find a minimal set of SNPs that can uniquely distinguish common haplotypes. In most cases, there are more than one sets of htSNPs that meet the requirement. In version 0.4, the program simple report the first one it gets. In version 0.5, two considerations were taken into account in order to select the best one out of all candidate sets. Firstly, although several sets of htSNPs may meet the same criterion (distinguishing common haplotypes), the power for association study can be different among these sets. To find the one with highest power, candidate sets are evaluated based on r^2 , which represents absolute level of linkage disequilibrium. For each candidate set, the minimal pair-wise r^2 between all pair of tagged SNPs and untagged SNP are calculated, then we choose the set with highest $\min(r^2)$. However, even with this criteria, chances are that there are still more than one good candidates. Therefore, we introduced a third criteria, referred to a position score. The idea behind this is pretty straight forward. HtSNPs will be used to map variant sites associated with diseases. In the process of identifying htSNPs, we only studied the SNP markers we typed. Is there any chance of losing the power for those untyped markers? That is possible! An intuitive solution to reduce such chances (no theoretical proof yet) would be to select htSNPs that evenly spaced in the genome. The position score proposed here is a composite index to measure the distribution of htSNPs. In the ideal case, all htSNPs are evenly spaced across the whole block. Therefore, the average distance between adjacent markers would be $L/(n-1)$, where L refers to the length of block, and n refers to the number of htSNPs. We also denote the distance between adjacent markers as X, then the positional score is $|\text{Mean}(X) - L/(n-1)| + \text{STD}(X)$. A lower score indicates a distribution closer to the ideal case. In summary, we proposed three criteria for htSNPs (1)minimum number of SNP; (2) $\max(\min(r^2))$; (3) $\min(\text{position score})$. The three criteria are ranked as their order. Criterion with lower rank will be used only when there are multiple candidates that meet the criteria with higher rank.

(2)A bug that could cause problem in handling ambiguous base call was fixed.

10-21-2002 A few changes were made on the subroutine for the htSNP identification, so that it can not only find htSNPs that can distinguish all haplotypes, but can also identify those distinguishing a certain percentage of haplotypes.

10-18-2002 hbfWrapper.pl is released in the V0.4. It is basically a perl script to convert haplotype files generated by Merlin to the fasta format required by the program haploBlockFinder. Although the fasta format we initially supported is very efficient to handle, the other format popular in the genetic community, such as the one generated by Merlin, has a major advantage in that it can handle in/del loci with more than one bases. With this perl script, format conversion can be done automatically.

10-12-2002 Identification of tagging SNPs for haplotype blocks has been implemented. Please note that there is no clear conclusion regarding what htSNPs should be chosen so far! Therefore, this program identifies a minimum set of SNP that can distinguish ALL haplotypes in a block.

10-10-2002 Two type of variant sites, insertion and deletion are supported.

10-06-2002 The 0.3 version is released. There are two major changes:

(1) An allele frequency filter is implemented, so that analysis can be focused on SNPs in some range of the allele frequency spectrum.

(2) showLDmatrix.pl is released. It can draw a matrix-like diagram in order to compare the haplotype blocks by this program and the pair-wise LD patterns. Two pair-wise LD measurements are shown, D' and LLR. LLR is the log likelihood ratio between observed haplotypes and expected haplotypes assuming linkage equilibrium.

09-21-2002 A bug is found in drawBlocks.pl, which results in miscalculation in canvas height in compact mode. The problem is fixed.

09-11-2002 Another bug (when there are many loci, the color allocation may run out of space and all haplotypes in the rear part have the same color) in drawBlocks.pl is found and fixed.

08-30-2002 Two bugs (one related to the HSV color scheme and the other related to the compact display mode) were fixed. A new option is added to display more than 4 haplotypes.

06-25-2002 drawBlocks.pl is released.

06-14-2002 The first version of this program is released.

PROBLEMS AND COMMENTS

Please address any comment or problem to:
kzhang@genetics.med.harvard.edu