HaploBlock Version 1.2
Instruction Manual

Gideon Greenspan
Computer Science Department
Technion – Israel Institute of Technology

gdg@cs.technion.ac.il

April 28, 2004

Contents

List of Tables

1 Introduction

HaploBlock (`http://bioinfo.cs.technion.ac.il/haploblock/`) is a software package for inferring statistical models of haplotype block variation and applying them for high density haplotype resolution and linkage disequilibrium mapping. HaploBlock is described in these papers:

- Model-based Inference of Haplotype Block Variation, *Proceedings of the Seventh Annual International Conference on Computational Molecular Biology (RECOMB 2003).* Also to appear in *Journal of Computational Biology*, Volume 11, Number 2-3.
- High Density Linkage Disequilibrium Mapping using Models of Haplotype Block Variation. Accepted for the *Twelfth International Conference on Intelligent Systems for Molecular Biology (ISMB 2004)* and to appear in *Bioinformatics*.

HaploBlock is written in ANSI C and available as a command-line executable for Linux, Mac OS X and Sun OS. The HaploBlock package comes with two executables, compiled to deal with two different encodings for SNP marker alleles. The base pair version (`haploblock_b`) reads and writes files where alleles are represented as their bases (i.e. `A`, `C`, `G`, `T`, $-$) and so can deal with up to five allelic variants at each site. The numerical version (`haploblock_n`) reads and writes files where alleles are represented numerically (i.e. `1`, `2`) and can only deal with biallelic data. If possible, `haploblock_n` should be used, since it runs considerably faster than `haploblock_b`. HaploBlock includes utility functions for converting between base pair and numerical encoding.

2 Quick start

HaploBlock is most commonly used for high density haplotype resolution or linkage disequilibrium mapping with biallelic marker data. This section explains the minimum required for these operations – HaploBlock has many additional parameters which should be understood before using it as part of a scientific study. Almost all of the processing time will be taken by the model inference stage (`-W`) which can be interrupted since models are output while the algorithm progresses. For reasonable results, 10 models should be sampled at the very least.

2.1 Haplotype resolution

- Arrange the genotypes in file *genofile*, formatted as per Section 3.1 with numerical encoding.
- Run `./haploblock_n -W -g` *genofile* `-m` *modelfile* to sample 100 models in file *modelfile*.
- Run `./haploblock_n -S -g` *genofile* `-m` *modelfile* `-h` *haplofile* to resolve the genotypes using the sampled models, placing the results in file *haplofile*.
- Interpret the haplotype pairs in file *haplofile* according to Section 3.1.

2.2 Linkage disequilibrium mapping

- Arrange the haplotype or genotype data with phenotype indicators in file *phenofile*, formatted as per Section 3.1 with numerical encoding.
- Arrange the physical SNP locations in file *mapfile*, formatted as per Section 3.4.
- If *phenofile* contains haplotypes, run `./haploblock_n -W -h` *phenofile* `-m` *modelfile* to sample 100 models in file *modelfile*. Otherwise, substitute `-g` for `-h`.
- If *phenofile* contains haplotypes, run `./haploblock_n -X -h` *phenofile* `-m` *modelfile* `-y` *mapfile*. Otherwise, substitute `-g` for `-h`.
- Read the posterior probabilities and densities output for each SNP interval.

3 File formats

On any platform, HaploBlock can read files with Unix (LF), PC (CR+LF) or Mac OS (CR) line endings. Files will be written with the native line endings of the platform on which the executable is running (currently Unix in all cases). In all file formats, each line can end with an optional comment, preceded by a `#` character. Blank lines and leading white space are ignored.

For data representing a possible SNP marker alleles, we require $a + 1$ symbols to represent each haplotype site, since we allow for unknown values. To encode each genotype allele pair, we require $(a + 1)(a + 2)/2$ symbols, to represent all possible unordered pairs of haplotype alleles, including unknowns. The symbols used for haplotype and genotype data in base pair format (for `haploblock_b`) are shown in Table 1, following IUPAC conventions where possible. The symbols for data in numerical format (for `haploblock_n`) are shown in Table 2. In each table, ? represents an unknown allele. Note that alleles in base pair encoding are case sensitive.

3.1 Marker data

SNP marker data is represented as a flat file, with each line encoding a single haplotype or genotype, with an optional phenotype indicator. Alleles are encoded by the symbols in Table 1 for `haploblock_b` and Table 2 for `haploblock_n`. Each haplotype or genotype in the file must have the same number of SNP markers. An example genotype file with base pair encoding containing 3 SNPs for 2 individuals without phenotypes is shown below:

```
RYP
PCZ
```

The genotypes defined by this file are $([A, G], [C, T], [G, -])$ and $([G, -], [C, C], [-, ?])$. HaploBlock can also read marker data in FASTA format, where each haplotype or genotype is preceded by a line which begins with the `>` character. In FASTA format, haplotypes or genotypes can be broken over multiple lines since `>` acts as a delimiter.

Each haplotype or genotype can have an optional phenotype attached (excluding FASTA format). Phenotypes are indicated by a leading integer ≥ 0 separated by white space from the marker data for that haplotype or genotype. Haplotypes or genotypes without a phenotype indicator are treated as having unknown phenotype for the purposes of mapping. An example file with numerical encoding containing 48 SNPs for 5 haplotypes, 4 of which have phenotypes:

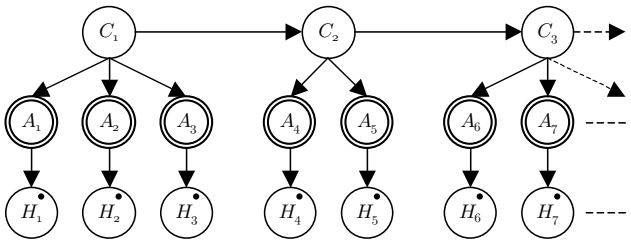
```
0 212212121121221211000111211121222211212201221122
1 221212010121121111222111212211222111221112122111
2 221111212222112121212122121211211222121120212121 12
0 112111122211121002112112212112112111122001111121
121212101121222121121211121220212121111211212112
```

For some of HaploBlock's functions, marker data is interpreted as pairs of haplotypes belonging to individuals. In this case, each sequential pair of haplotypes belongs to the same individual, so that individual i's haplotypes are in positions $2i - 1$ and $2i$ in the file.

In `haploblock_b` only, there is an option `-i` which should be used if marker data was generated by multiple alignment. With this option, HaploBlock will interpret a series of `-` symbols that reaches the start or end of a sequence as unknown alleles, since it is usually a consequence of sequences in the alignment containing different amounts of genetic data. For example, the sequence `---CT-G-A--` would be read under option `-i` as `???CT-G-A??`.

3.2 Statistical models

A HaploBlock statistical model defines a distribution over haplotypes. The Bayesian Network below depicts an example model, with a random variable C_k for each block $k = 1 \ldots b$ and two random variables A_j and H_j for each SNP $j = 1 \ldots l$. Each variable C_k defines the ancestor from which a haplotype is descended in block k. For each block k, variables $A_{s_k} \ldots A_{e_k}$ define the sequence of the ancestor indicated by the value of C_k, where the first and last SNPs of block k are numbered s_k and e_k respectively. Variables $H_1 \ldots H_l$ define the observed haplotype data over loci $1 \ldots l$.



A particular statistical model is defined by a block partition and parameter vectors θ, $\hat{a}$ and μ, which specify ancestor distributions, ancestor sequences and mutation rates respectively. For the first block, $Pr(C_1 = c) = \theta_{1,c}$ and for subsequent blocks, $Pr(C_k = c|C_{k-1} = c') = \theta_{k,c' \to c}$. For SNP j in block k, $Pr(A_j = a|C_k = c) = 1$ if $\hat{a}_{k,c,j} = a$ and 0 otherwise. For SNP j, $Pr(H_j = h|A_j = a) = \mu_{j,a \to h}$. For a fuller explanation of this model and its extension to represent genotypes, consult the papers cited in Section 1.

A statistical model file represents one or more such models, separated by a line which begins with a hyphen (`-`). For example, the following file contains two trivial models:

```
2
B 1 1.0 AG

- Here is the other

1 2
B 1 1.0 C
T 2 1.0 C T
```

The first line for each model contains a series of integers separated by white space, one for each block $k = 1 \ldots b$ in order. Each integer on this line contains the index e_k of the last SNP in block k, so that the last integer gives the total number of SNPs l represented by the model. Multiple models contained within a file must have the same value for l.

Every subsequent line of a model description begins with a symbol `B`, `T` or `M`. Lines beginning with `B` specify an ancestor sequence $\hat{a}$, those with `T` specify a Markov transition probability θ and those with `M` specify a mutation rate μ.

An ancestor sequence line is formatted as either '`B` k $\pi_{k,c}$ $\hat{a}_{k,c}$' or '`B` k $\pi_{k,c}$ $v_{k,c}$ $\hat{a}_{k,c}$', specifying an ancestor with sequence $\hat{a}_{k,c}$ and marginal probability $\pi_{k,c}$ for block k. In the second form, $v_{k,c}$ sets a label which may subsequently be used as a shortcut for $\hat{a}_{k,c}$ in the model. The index c is not specified and is assigned automatically. Sequences $\hat{a}_{k,c}$ must contain $e_k - s_k + 1$ characters and are specified using base pair or numerical encoding as appropriate. Labels $v_{k,c}$ must be unique within each block and must not begin with a character that encodes an allele (lower case letters are safe). To aid readability, model files generated by HaploBlock automatically list ancestors for each block k in descending order of their prior probability $\pi_{k,c}$ and use labels for the ancestors of blocks which contain more than a few SNPs.

A Markov transition line is formatted as '`T` k $\theta_{k,c' \to c}$ $\hat{a}_{k-1,c'}$ $\hat{a}_{k,c}$', specifying the conditional probability $\theta_{k,c' \to c}$ of the ancestor with sequence $\hat{a}_{k,c}$ for block k given that ancestor $\hat{a}_{k-1,c'}$ was

present for block $k-1$. As with `B` lines, the indices c' and c are assigned automatically. Sequences $\hat{a}_{k,c}$ and $\hat{a}_{k-1,c'}$ must contain $e_k - s_k + 1$ and $e_{k-1} - s_{k-1} + 1$ characters respectively. If previously set, a label can be used instead of sequence $\hat{a}_{k,c}$ and/or $\hat{a}_{k-1,c'}$. Omitted transitions are assumed to have zero probability. However, if no transitions are specified for block k, it is assumed to be independent of the previous block, with the marginal distribution specified by the block's `B` lines.

A mutation rate line is formatted as '`M` j a h $\mu_{j,a\to h}$', specifying the mutation rate at site j from allele a to allele $h \neq a$. The alleles a and h are specified using base pair or numerical encoding as appropriate. Omitted mutations are assumed to have zero probability and the probability of each non-mutation is automatically set to $\mu_{j,a\to a} = 1 - \sum_{h\neq a} \mu_{j,a\to h}$.

Note that model files contain redundancy in order to make them more readable. Ancestor sequences for each block are picked up from both `B` lines and `T` lines. If there are any `T` lines for block k, all of the marginal probabilities specified by its `B` lines are ignored and calculated directly from the Markov chain. In practice, this means that `B` lines are only required for the first block.

An example model file with numerical encoding describing a distribution with 2 blocks over 8 SNPs, each with 2 ancestors, is shown below:

```
3 8

M 3 1 2 0.005
M 7 2 1 0.025

B 1 0.6 111
B 1 0.4 221

B 2 0.6 a 21121
B 2 0.4 b 12212

T 1 0.6 111
T 1 0.4 221

T 2 0.8 111 a
T 2 0.2 111 b

T 2 0.3 221 a
T 2 0.7 221 b
```

The parameters defined by this model file are: $b = 2$, $l = 8$, $s_1 = 1$, $e_1 = 3$, $s_2 = 4$, $e_2 = 8$, $q_1 = 2$, $q_2 = 2$, $\mu_{1,1\to1} = 1.0$, $\mu_{1,1\to2} = 0.0$, $\mu_{1,2\to1} = 0.0$, $\mu_{1,2\to2} = 1.0$, $\ldots$, $\mu_{3,1\to1} = 0.995$, $\mu_{3,1\to2} = 0.005$, $\mu_{3,2\to1} = 0.0$, $\mu_{3,2\to2} = 1.0$, $\ldots$, $\mu_{8,1\to1} = 1.0$, $\mu_{8,1\to2} = 0.0$, $\mu_{8,2\to1} = 0.025$, $\mu_{8,2\to2} = 0.975$, $\hat{a}_{1,1} = 111$, $\hat{a}_{1,2} = 221$, $\hat{a}_{2,1} = 21121$, $\hat{a}_{2,2} = 12212$, $v_{2,1} =$ `a`, $v_{2,2} =$ `b`, $\theta_{1,1} = 0.6$, $\theta_{1,2} = 0.4$, $\theta_{2,1\to1} = 0.8$, $\theta_{2,1\to2} = 0.2$, $\theta_{2,2\to1} = 0.3$, $\theta_{2,2\to2} = 0.7$.

3.3 Allele mapping

An allele mapping file represents a conversion between base pair and numerical allele encoding. Each line j in the file represents SNP j. The first symbol in each line contains the base pair for the major allele, encoded numerically as `1`. The second symbol contains the base pair for the minor allele, encoded numerically as `2`. An example allele mapping file for 3 SNPs is shown below:

```
AG
GC
TC
```

The file implies the haplotype mappings $AGT \leftrightarrow 111$, $GCC \leftrightarrow 222$, $GGC \leftrightarrow 212$.

3.4 Physical map

A physical map file describes the relative location of some SNP markers. Each line j in the file specifies the location of SNP j. Since a map file is only used to calculate the relative sizes of the intervals between adjacent SNPs, any starting point and unit of measurement can be used. An example physical map file for 4 SNPs is shown below:

```
148.191060
148.192022
148.193108
148.194345
```

4 Function reference

The first parameter to `haploblock_b` or `haploblock_n` specifies which function to perform. Each function takes a subset of the additional parameters listed in Table 3, as explained in the sections that follow. Each additional parameter is preceded by an identifying prefix, so these parameters can be specified in any order. Parameters which are omitted receive their default value from Table 3. The `-r` (reporting level) parameter is accepted by all functions, determining the level of detail with which progress is reported. Within the function descriptions below, the value specified for each parameter is indicated by that parameter's alphabetic symbol in `this typeface`.

If a function is not specified correctly, HaploBlock outputs a list of available function codes and exits. Similarly, if a function's parameters are not specified correctly, HaploBlock outputs a list of possible parameters for the function and exits. Otherwise, HaploBlock displays the function to be performed and the values taken for each parameter before proceeding. Once a function has successfully completed, HaploBlock displays its running time.

4.1 Generate data

4.1.1 Generate marker data or models by simulation (`-P`)

This function takes the following parameters from Table 3: `-c` (population capacity), `-d` (physical length), `-e` (simulation time), `-f` (number of founders), `-h` (haplotype file), `-j` (no Markov chain), `-m` (model file), `-n` (sample size), `-p` (hotspot density), `-s` (SNP density), `-u` (minimum mutation rate), `-v` (maximum mutation rate), `-w` (growth rate), `-x` (crossover density), `-y` (map file).

The population simulation begins with a bottleneck event and proceeds to form new generations based on exponential growth, random mating, no migration, neutral selection and recombination at hotspots only. Note that the simulation is based on many of the assumptions underlying the HaploBlock model, so it is clearly not a basis on which to assess its validity!

The simulation distributes SNPs and recombination hotspots within a chromosomal region of length `d` by a random Poisson process. SNPs are distributed with average density per nucleotide `s` and recombination hotspots are placed independently with density `p`. The crossover probability per generation remains fixed throughout the simulation and is selected randomly and independently for each recombination hotspot from the uniform distribution over $0 \ldots 2 \cdot \mathtt{x} \cdot \mathtt{d}/(t+1)$, where t is the number of hotspots placed.

The population is initiated with `f` founders whose gender is assigned randomly. The SNP alleles on these founders' `2f` haplotypes are assigned randomly, where each SNP is biallelic with equal probability for each allele. Over `e` generations, each individual in a new generation is descended from a random independent union between a male and female from the previous generation, and has gender assigned randomly. The new individual's haplotypes are obtained from those of its parents by simulated meiosis, in which crossover occurs randomly at each hotspot with the probability calculated above. If the population of generation i is p_i, the population of generation $i+1$ is given by $p_{i+1} = (1+\mathtt{w})^{(1-p_i/\mathtt{c})}$, representing an initial growth rate of `w` which tends to zero as the population reaches capacity.

The simulation's output is based on the final generation. SNPs with no variation in this generation are removed, to reflect what would be visible in real-world data. For each remaining SNP, the cumulative mutation rate (over all generations) from each allele to the other is set independently to $\mathtt{u} \cdot \exp(r \cdot \log(\mathtt{v}/\mathtt{u}))$ where r is a random variable distributed uniformly over $0 \ldots 1$. Note that these mutation distributions are only used to generate data once the simulation is complete.

Three types of data can be output from the simulation. If parameter `y` is specified, the SNP locations are output to physical map file `y`. If parameter `h` is specified, `n` haplotypes are sampled (with repeats) from the final generation and output with mutations to file `h`. If parameter `m` is specified, a statistical model is inferred from the final generation and output to file `m`.

When inferring a statistical model from the final generation, the number of ancestors q_k for each block k and their sequences $\hat{a}$ are determined after uniting any block ancestors with identical sequences. The parameters θ of the Markov chain are determined simply by counting the frequency with which ancestors in adjacent blocks appear together. However, if option `j` is specified, a model is inferred with independent blocks and no Markov chain. The mutation rates μ in the model are set according to those randomized at the end of the simulation process.

4.1.2 Generate haplotypes from models (`-H`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-k` (unknown rate), `-m` (model file), `-n` (sample size). It generates `n` haplotype samples independently using the model/s in file `m`. If file `m` contains more than one model then each haplotype is based on a model drawn uniformly and independently. To simulate failed measurements in a laboratory, each marker allele is converted to an unknown with probability `k`. The generated haplotype data is output to file `h`.

4.1.3 Generate genotypes from models (`-G`)

This function takes the following parameters from Table 3: `-g` (genotype file), `-k` (unknown rate), `-m` (model file), `-n` (sample size). It generates `n` genotype samples independently using the model/s in file `m`. If file `m` contains more than one model then each genotype is based on a model drawn uniformly and independently. Each marker allele is converted to an unknown with probability `k` before pairing. The generated genotype data is output to file `g`.

4.1.4 Generate quasi-phenotypes from marker data (`-Q`)

This function takes the following parameters from Table 3: `-d` (disease dominance), `-g` (genotype file), `-h` (haplotype file), `-o` (new haplotype file), `-p` (disease penetrance), `-s` (selected SNP), `-u` (new genotype file), `-y` (map file), `-z` (new map file). Multiple haplotype and/or genotype input files can be specified using multiple `-h` or `-g` parameters, but all must have the same number of SNPs. The function reads in the haplotypes in files `h` and/or the genotypes in files `g`, converts the alleles at a target SNP into phenotypes, and outputs the resulting haplotype and genotype marker files with phenotypes (and minus the target SNP) to `o` and `u` respectively.

If parameter `s` is included, it specifies the index (≥ 1) of the target SNP in the marker data, otherwise a target will be selected randomly. If a map file `y` is specified, this random selection takes account of physical distances between SNPs, giving each SNP probability in proportion to the distance between its neighboring SNPs. If no map file is specified, the random selection will be uniform over all but the first and last SNPs. If parameter `z` is specified, a new physical map file is written to file `z` with the target SNP removed, suitable for use with `o` and `u`.

For haplotype data, the phenotype for each haplotype is assigned based on its allele for the target SNP. The most common allele for the target is mapped to phenotype `0`. Each additional allele observed is mapped to a different phenotype, numbered from `1` upwards in descending order of allele frequency. To simulate penetrance, the phenotype is then set to `0` with probability $1 - \mathtt{p}$ independently of the allele at the target SNP. If a haplotype's allele for the target is unknown, that haplotype is always assigned the unknown phenotype.

For genotype data, the two alleles at the target SNP are converted separately to phenotypes, which are then combined under the dominance model specified by `d`. If parameter `d` is `0` (recessive), the numerically lower number is set as the overall phenotype. If `d` is `1` (codominant), every different (unordered) pair of phenotypes is mapped to a different overall phenotype. If `d` is `2` (dominant), the numerically higher number is set as the overall phenotype. If one of the two phenotypes is unknown, the overall phenotype is set to unknown unless the missing phenotype makes no difference under the dominance model specified. These rules generalize the standard dominance model for more than 2 alleles, and will produce the expected results for biallelic data.

4.2 Infer models from data

4.2.1 Infer a single model from marker data (`-F`)

This function takes the following parameters from Table 3: `-b` (maximum blocks), `-g` (genotype file), `-h` (haplotype file), `-i` (detect alignments), `-j` (no Markov chain), `-l` (initial block length), `-m` (model file), `-q` (maximum ancestors), `-u` (minimum mutation rate), `-v` (maximum mutation rate), `-z` (resume model search). Multiple haplotype and/or genotype input files can be specified using multiple `-h` or `-g` parameters, but all must have the same number of SNPs. HaploBlock will search for a single model (see RECOMB 2003 paper) for the marker data in `h` and/or `g`, constraining mutation rates by $\mathtt{u} \leq \mu_{j,a \to h} \leq \mathtt{v}$, the number of blocks by $b \leq \mathtt{b}$ and the number of ancestors by $q_k \leq \mathtt{q}$. If option `j` is specified, a model will be inferred with independent blocks and no Markov chain. If option `z` is specified, the search begins from the last model in file `m`, otherwise it begins from an initial model which contains evenly-spaced hotspots to ensure that the length of each block is no more than `l`. At the end of each full search round, the best model seen replaces that in file `m`, so this function can be interrupted and later resumed using parameter `z`.

4.2.2 Infer a model ensemble from marker data (`-W`)

This function takes the same parameters as function `-F` above, with the addition of `-n` (sample size). It infers an ensemble of models (see journal version of RECOMB 2003 paper or ISMB 2004 paper) for the marker data in `h` and/or `g`. After each round of the sampling algorithm, a model is appended to file `m`, so this function can be interrupted and later resumed using parameter `z`.

4.3 Haplotype resolution

4.3.1 Join haplotype pairs to form genotypes (`-J`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-g` (genotype file), `-i` (detect alignments). It combines the alleles at each SNP for each haplotype pair in file `h`, writing the resulting genotypes to file `g`.

4.3.2 Haplotype resolution by models (`-S`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-g` (genotype file), `-i` (detect alignments), `-m` (model file). It resolves the genotypes in file `g` by applying the models in file `m` (see RECOMB 2003 paper). If file `m` contains more than one model then the resolution is performed separately for each model and each site in the final haplotype pair is assigned to the allele pair which was inferred most often. For heterozygous sites, the pair is oriented relative to the previous heterozygous site so as to be compatible with the maximum number of the individual model-based resolutions. The inferred haplotype pairs are written to file `h`.

4.3.3 Haplotype resolution by Clark's algorithm (`-L`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-g` (genotype file), `-i` (detect alignments). It splits the genotypes in file `g` using the Clark algorithm (modified slightly to work with unknowns) as described in *Inference of haplotypes from PCR-amplified samples of diploid*

populations (Clark A.G., 1990, Mol Biol Evol. 7:111). Any genotypes which remain unresolved are divided arbitrarily and the inferred haplotype pairs are written to file `h`.

4.3.4 Haplotype resolution by Local EM (`-I`)

This function takes the following parameters from Table 3: `-e` (number of repeats), `-h` (haplotype file), `-g` (genotype file), `-i` (detect alignments). It splits the genotypes in file `g` using a modification of the standard EM haplotype resolution algorithm which overcomes its exponential complexity using a divide-and-conquer approach. This approach is similar to that described in *Bayesian Haplotype Inference for Multiple Linked Single-Nucleotide Polymorphisms* (Niu et al., 2002, Am J. Hum. Genet. 70:157). The inference is performed independently `e` times, after which the set of inferred haplotype pairs with highest likelihood is written to file `h`.

4.3.5 Evaluate haplotype resolution (`-E`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-i` (detect alignments), `-t` (test file). Files `h` and `t` should both contain $2n$ haplotypes, where n is the number of individuals, determined automatically from the files. Let $h_{i,1,j}$ and $h_{i,2,j}$ be the respective alleles of the first and second haplotypes of the true pair (from file `h`) for individual $i = 1 \dots n$ at site $j = 1 \dots l$. Similarly, let $h'_{i,1,j}$ and $h'_{i,2,j}$ be the respective alleles of the first and second haplotypes of the inferred pair (from file `t`) for individual i at site j.

Let the function $\delta(x, y)$ return 1 if alleles $x \neq y$ and 0 otherwise. The value $\lambda_{i,j}$ indicates whether the haplotypes in the true and inferred pairs for individual i, oriented as they appear in the files, are incompatible at site j, where $\lambda_{i,j} = \delta(h_{i,1,j}, h'_{i,1,j}) \vee \delta(h_{i,2,j}, h'_{i,2,j})$. Similarly, the value $\lambda'_{i,j}$ indicates whether the haplotypes in the true and inferred pairs for individual i, oriented in reverse, are incompatible at site j, where $\lambda'_{i,j} = \delta(h_{i,1,j}, h'_{i,2,j}) \vee \delta(h_{i,2,j}, h'_{i,1,j})$. Let $\alpha_{i,j}$ denote whether site j of individual i is heterozygous, where $\alpha_{i,j} = \delta(h_{i,1,j}, h_{i,2,j})$. Clearly, for all individuals i and sites j, if $\alpha_{i,j} = 1$, exactly one of $\lambda_{i,j}$ or $\lambda'_{i,j}$ is 1, otherwise both $\lambda_{i,j}$ and $\lambda'_{i,j}$ are 0. Let β_i be the ordered list of heterozygous sites in individual i, so that for all $j = 1 \dots l$, $\alpha_{i,j} = 1 \Leftrightarrow j \in \beta_i$ and for all $t = 1 \dots |\beta_i| - 1$, $\beta_{i,t} < \beta_{i,t+1}$.

HaploBlock provides four metrics for measuring phasing errors, calculated from these values. The first metric Δ_I is the number of individuals who were not phased perfectly, where $\Delta_I = \sum_i \left(\bigvee_j \lambda_{i,j}\right) \wedge \left(\bigvee_j \lambda'_{i,j}\right)$. This measure is common in the literature but provides little information on the degree of correctness of the inferred haplotype pairs. The second metric Δ_S is the number of sites which were phased incorrectly, taking the better orientation for each individual, where $\Delta_S = \sum_i \min\left(\sum_j \lambda_{i,j}, \sum_j \lambda'_{i,j}\right)$. This provides a good overall measure of the degree of correctness of the phased haplotype pairs if marker order is unimportant.

The third metric Δ_A is the number of adjacent pairs of sites which were phased incorrectly relative to each other, given by $\Delta_A = \sum_i \sum_{j=1}^{l-1} (\lambda_{i,j} \vee \lambda_{i,j+1}) \wedge \left(\lambda'_{i,j} \vee \lambda'_{i,j+1}\right)$. This measures the local correctness of the phased haplotype pairs and is particularly relevant if the inferred haplotypes are to be used for disease mapping. The fourth metric Δ_H (sometimes called 'switch rate') is the number of pairs of heterozygous sites (in order but not necessarily adjacent) which were phased incorrectly relative to each other, given by $\Delta_H = \sum_i \sum_{t=1}^{|\beta_i|-1} \left(\lambda_{i,\beta_{i,t}} \vee \lambda_{i,\beta_{i,t+1}}\right) \wedge \left(\lambda'_{i,\beta_{i,t}} \vee \lambda'_{i,\beta_{i,t+1}}\right)$. Note that $\Delta_I = 0 \Leftrightarrow \Delta_S = 0 \Leftrightarrow \Delta_H = 0 \Rightarrow \Delta_A = 0$, for example if true haplotypes (ACA, TCT) were inferred as (ACT, TCA), we would obtain the statistics $\Delta_I = 1, \Delta_S = 1, \Delta_A = 0, \Delta_H = 1$.

When calculating error rates, unknowns in the true haplotypes (file `h`) and inferred haplotypes (file `t`) are dealt with differently. If site j on either true haplotype for individual i is unknown, we automatically exclude that site from consideration, setting $\lambda_{i,j} = \lambda'_{i,j} = \alpha_{i,j} = 0$. However, if the inferred haplotype pair contains an unknown which 'hedges its bets' against a heterozygous site in the true haplotype pair, we set $\lambda_{i,j} = \lambda'_{i,j} = \frac{1}{2}$, rounding each metric as appropriate.

4.4 Linkage disequilibrium mapping

4.4.1 LD mapping by models (-X)

This function takes the following parameters from Table 3: `-g` (genotype file), `-h` (haplotype file), `-m` (model file), `-y` (map file). Multiple haplotype and/or genotype input files can be specified using multiple `-h` or `-g` parameters, but all must have the same number of SNPs. The function performs linkage disequilibrium mapping on the phenotyped haplotypes in file `h` and/or the phenotyped genotypes in file `g` by applying the models in file `m` (see ISMB 2004 paper). The file `y` specifies the physical location of the SNPs in the haplotype and genotype files – if no map file is specified, the SNPs are assumed to be uniformly spaced. For each interval between adjacent SNPs, the function outputs the posterior probability that the interval contains the phenotype locus, as well as its posterior density, with standard deviations over the models in `m`.

4.4.2 LD mapping by individual SNPs (-Y)

This function takes the same parameters as function `-X` above, with the exception of `-m` (model file). The function performs linkage disequilibrium mapping on the phenotyped haplotypes in file `h` and/or the phenotyped genotypes in file `g` by assuming that the alleles at each SNP are independent (see ISMB 2004 paper), producing a similar output to function `-X` above.

4.5 Analyze models

4.5.1 Summarize models (-M)

This function takes the following parameters from Table 3: `-m` (model file), `-n` (sample size). Given an input set $\{M^1, \ldots, M^z\}$ of models in file `m`, the function outputs the description length $DL(M^i)$ of each model M^i, with parameter accuracy in the model descriptions based on a sample of `n` haplotypes (see RECOMB 2003 paper). It also outputs the proportion of models with a hotspot between SNPs $j-1$ and j for each $j = 2 \ldots l$, given by $\frac{1}{z}\sum_{i=1}^{z} |\{k|s_k^i = j\}|$ where $\square^i$ refers to parameter $\square$ for model M^i. Similarly, it outputs the average transition conditional entropy between SNPs $j-1$ and j, given by $\frac{1}{z}\sum_{i=1}^{z}\sum_{k|s_k^i=j} \xi^i_{(k-1)\to k}$. Let $k^i(j)$ be the block in which site j falls in model M^i, so that $s^i_{k^i(j)} \le j \le e^i_{k^i(j)}$. HaploBlock outputs the average number of ancestors for each SNP $j = 1 \ldots l$, given by $\frac{1}{z}\sum_{i=1}^{z} q^i_{k^i(j)}$. Lastly, it outputs the average overall site mutation rate for each SNP j, given by $\frac{1}{z}\sum_{i=1}^{z} \rho^i_j$, where $\rho^i_j = \sum_{a\in B}(\sum_{h\neq a} \mu^i_{j,a\to h} \cdot \sum_{c|a=\hat{a}^i_{k^i(j),c,j}} \pi^i_{k^i(j),c})$. Standard deviations over the models for these last two statistics are also displayed.

4.5.2 Evaluate data under models (-D)

This function takes the following parameters from Table 3: `-g` (genotype file), `-h` (haplotype file), `-i` (detect alignments), `-m` (model file). Multiple haplotype and/or genotype input files can be specified using multiple `-h` or `-g` parameters, but all must have the same number of SNPs as the model file. The function outputs the data probability $Pr(H, G|M^i)$ of the haplotypes and/or genotypes in files `h` and/or `g` under each model M^i in file `m`, as well as the total description length $DL(H, G, M^i)$ (see RECOMB 2003 paper).

4.5.3 Compare models (-V)

This function takes the following parameters from Table 3: `-m` (model file), `-n` (sample size), `-t` (test file). The K-L divergence between the haplotype distributions of the true model M (in file `m`) and test model M' (in file `t`) is defined as $\sum_h Pr(h|M)\log\left(Pr(h|M)/Pr(h|M')\right)$. Since an exact calculation of this is infeasible (except in the extreme case where all mutation rates are zero), we generate an unbiased estimate. A list H of `n` haplotypes is drawn randomly and independently from the distribution defined by the true model. The K-L divergence is estimated

as $\frac{1}{|H|}\sum_{h\in H}\log\left(Pr(h|M)/Pr(h|M')\right)$. Note that HaploBlock's default value of `n` is inappropriate for this function – for a good estimate, it should be set to at least 10^5.

4.6 Miscellaneous haplotype operations

4.6.1 Randomly reorder haplotypes (`-R`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-i` (detect alignments), `-o` (new haplotype file). The function randomly reorders the haplotypes in file `h`, writing the resulting haplotype list to file `o`.

4.6.2 Randomly resample haplotypes (`-C`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-i` (detect alignments), `-n` (sample size), `-o` (new haplotype file). The function randomly and independently samples `n` haplotypes (with repeats) from the uniform distribution over the haplotypes in file `h`, writing the resulting haplotype list to file `o`.

4.6.3 Complete haplotypes by models (`-A`)

This function takes the following parameters from Table 3: `-h` (haplotype file), `-m` (model file), `-o` (new haplotype file). The function infers the value of any unknown sites in the haplotypes in file `h` using the models in file `m`. This is performed similarly to model-based haplotype resolution by finding the most likely a posteriori assignment of the unknown variables in the model, given the value of the known variables (see RECOMB 2003 paper). If file `m` contains more than one model then the completion is performed separately for each model and each site in the final haplotypes is assigned to the allele which was inferred most often. The inferred haplotypes are written to file `o`.

4.7 Format conversions

4.7.1 Convert marker data to numerical encoding (`-N`)

This function (available in `haploblock_b only`) takes the following parameters from Table 3: `-a` (allele file), `-h` (haplotype file), `-g` (genotype file), `-i` (detect alignments), `-o` (new haplotype file), `-u` (new genotype file). Multiple haplotype and/or genotype files can be specified using multiple `-h` or `-g` parameters, but all must have the same number of SNPs. The function converts the haplotypes in files `h` and/or the genotypes in files `-g` into numerical encoding by assigning the major observed allele to numerical allele `1` and the minor allele to `2`. If more than two alleles are observed for any site, the conversion will fail. The allele mapping obtained is output to file `a` and the numerically encoded haplotypes and genotypes are output to files `o` and/or `u` respectively.

4.7.2 Convert models to base pair encoding (`-B`)

This function (available in `haploblock_b only`) takes the following parameters from Table 3: `-a` (allele file), `-m` (model file), `-t` (new model file). It converts the statistical models in file `m` to base pair encoding using the allele mapping in file `a`, and outputs the resulting models in file `t`.

4.7.3 Reformat models (`-O`)

This function takes the following parameters from Table 3: `-m` (model file), `-t` (new model file). It reads in the statistical models in file `m` and outputs them in file `t`, using default ordering and formatting. This function is useful for making a model file more human readable.

5 Version History

The following versions of HaploBlock have been released publicly:

- **Version 1.2, April 2004**

  Added `-X` function for LD mapping based on block models.
  Added `-Y` function for LD mapping based on individual SNPs.
  Added `-Q` function to generate quasi-phenotypes from marker data.
  Added `-I` function to perform haplotype resolution by Local EM.
  Added `-B` function to convert models to base pair encoding.
  Added `-O` function to reformat models for readability.
  Added physical map file format for use with mapping and other functions.
  Extended model file format to allow a partial Markov chain.
  Extended marker data file format for optional phenotypes.
  Added `-j` option to `-F`, `-W` and `-P` functions to generate models with no Markov chain.
  Added output of haplotypes and physical map from `-P` simulation function.
  Extended `-N` function to convert genotype data.
  Optimized model inference from unphased genotype data.
  Improved convergence testing to reduce EM iterations.

- **Version 1.1, March 2003**

  Added `-W` function to infer an ensemble of models.
  Added `-A` function to complete haplotypes using models.
  Extended model file format to express multiple models in an ensemble.
  Extended all functions to work with multiple models.
  Extended model file format to allow ancestor labeling.
  Added `-z` option to resume model search from point of interruption.
  Added `-b` and `-q` options to specify maximum blocks and ancestors for search.

- **Version 1.0, December 2002**

  Added caching of many model calculations to drastically reduce search time.
  Fixed underflow issue by extracting common log factors where appropriate.
  Added `-l` option to specify initial block length for search.
  Removed prior factor when calculating T_k parameter description length.
  Added `-i` option to detect partial sequences from alignments.
  Added new pairwise measure of haplotype resolution accuracy.

- **Version 1.0 Beta, October 2002**

  Initial release

Table 1: Base pair allele encoding

Symbol	Haplotype	Genotype
A	A	$[A, A]$
M		$[A, C]$
R		$[A, G]$
W		$[A, T]$
L		$[A, -]$
E		$[A, ?]$
C	C	$[C, C]$
S		$[C, G]$
Y		$[C, T]$
O		$[C, -]$
F		$[C, ?]$
G	G	$[G, G]$
K		$[G, T]$
P		$[G, -]$
I		$[G, ?]$
T	T	$[T, T]$
Q		$[T, -]$
J		$[T, ?]$
-	$-$	$[-, -]$
Z		$[-, ?]$
N	$?$	$[?, ?]$

Table 2: Numerical allele encoding

Symbol	Haplotype	Genotype
1	A	$[A, A]$
0	$?$	$[A, a]$
4		$[A, ?]$
2	a	$[a, a]$
5		$[a, ?]$
3		$[?, ?]$

Table 3: Parameters

Code	Parameter	Default value	Range	Units
`-a`	Allele mapping file	–	–	–
`-b`	Maximum blocks	none	≥ 1	blocks (0=no maximum)
`-c`	Population capacity	10,000	≥ 1	individuals
`-d`	Dominance model	0	0|1|2	recessive|codominant|dominant
`-d`	Physical length	10,000	≥ 1	nucleotides
`-e`	Number of repeats	20	≥ 1	iterations
`-e`	Simulation time	500	≥ 0	generations
`-f`	Bottleneck founders	20	≥ 1	individuals
`-g`	Genotype file	–	–	–
`-h`	Haplotype file	–	–	–
`-i`	Detect alignments	off	–	–
`-j`	No Markov chain	off	–	–
`-k`	Unknown rate	0.0	$0.0 \ldots 1.0$	unknowns/SNP
`-l`	Initial block length	100	≥ 0	SNPs (0=no blocks)
`-m`	Statistical models file	–	–	–
`-n`	Sample size	100	≥ 1	samples
`-o`	New haplotype file	–	–	–
`-p`	Disease penetrance	1.0	$0.0 \ldots 1.0$	probability
`-p`	Hotspot density	10^{-4}	≥ 0.0	hotspots/nucleotide
`-q`	Maximum ancestors	none	≥ 1	ancestors (0=no maximum)
`-r`	Reporting level	3	$0 \ldots 4$	–
`-s`	Selected SNP	none	–	–
`-s`	SNP density	10^{-3}	≥ 0.0	SNPs/nucleotide
`-t`	New model file	–	–	–
`-t`	Test file	–	–	–
`-u`	Minimum mutation rate	10^{-6}	$0.0 \ldots 1.0$	mutations/SNP
`-u`	New genotype file	–	–	–
`-v`	Maximum mutation rate	10^{-3}	$0.0 \ldots 1.0$	mutations/SNP
`-w`	Growth rate	0.05	≥ 0.0	rate/generation
`-x`	Crossover density	10^{-8}	$0.0 \ldots 1.0$	crossovers/nucleotide/generation
`-y`	Physical map file	–	–	–
`-z`	New physical map file	–	–	–
`-z`	Resume model search	off	–	–