HAPI-UR v 1.01 Software Manual

Amy L. Williams Nick Patterson David Reich

September 27, 2012

Contents

1 Introduction: Mailing List and File Formats

HAPI-UR is a method for inferring haplotype phase of large genotype datasets with at least 1,000 samples or more. HAPI-UR supports phasing of unrelated samples as well as trio and duo families, in any combination.

Register for Announcements!

We strongly encourage registration for our moderated, low-traffic announcements email list. We use this for important announcements related to HAPI-UR, including new releases and bug fixes. Register at:
`http://receptor.med.harvard.edu/mailman/listinfo/hapi-ur-announce`

File Formats

HAPI-UR accept three file formats: Eigenstrat, packed Ancestrymap, and PLINK BED. (A description of the Eigenstrat file format appears in Section 4.) To convert PLINK PED files to BED format, use the following command:

```
plink --file <ped_prefix> --out <out_prefix> --recode --make-bed
```

The EIGENSOFT package also has a program called `convertf` that converts between PLINK PED, PLINK BED, Eigenstrat, and packed Ancestrymap formats. EIGENSOFT is available from
`http://genetics.med.harvard.edu/reich/Reich_Lab/Software.html`

2 Running HAPI-UR

HAPI-UR is intended to run on large datasets with at least 1,000 samples or more and may not work well for smaller datasets. Running HAPI-UR without any arguments prints a usage message with brief documentation of the required and optional arguments. This document describes usage of HAPI-UR in greater detail.

2.1 Required Arguments

HAPI-UR requires five arguments: a genotype file, SNP/marker file, individual file, output file prefix, and maximum window size. If the genotype, SNP, and individual files all have the same prefix, you can use one shortcut argument to specify these files.

The examples below describe running HAPI-UR for either Eigenstrat or packed Ancestrymap format data files and for PLINK BED files.

Eigenstrat and Packed Ancestrymap Input Data

To run HAPI-UR on Eigenstrat or packed Ancestrymap format data files named `data.geno`, `data.snp`, and `data.ind`, with a maximum window size of 73 markers, and with the output prefix `phase`, use the command:

```
hapi-ur -g data.geno -s data.snp -i data.ind -w 73 -o phase
```

Alternatively, because these genotype files all have the same prefix `data`, you can use the `-b` option to load a given set of `.geno`, `.snp`, and `.ind` files. The following is equivalent to the previous command:

```
hapi-ur -b data -w 73 -o phase
```

PLINK BED Input Data

To run HAPI-UR on PLINK BED format data files named `plink.bed`, `plink.bim`, and `plink.fam`, with a maximum window size of 73 markers, and with the output prefix of `phase`, use the command:

```
hapi-ur -g plink.bed -s plink.bim -i plink.fam -w 73 -o phase
```

For a set of PLINK BED files with the same prefix, use the `-p` option to load a given set of `.bed`, `.bim`, and `.fam` files. The following is equivalent to the previous command:

```
hapi-ur -p plink -w 73 -o phase
```

2.2 Determining the Maximum Window Size to Use

As a rough guide for setting the maximum window size (given by the `-w` option), we suggest linear increase relative to the window sizes evaluated in the HAPI-UR paper. Thus, if s is the number of SNPs/markers in a given dataset, the following formula should give a reasonable maximum window size value:

$$(85 - 64)/(755008 - 386353) \times (s - 386353) + 64$$

Note that this assumes that s is a genome-wide count of markers, not the number of markers in an individual chromosome.

For very large datasets, it can be beneficial to increase the maximum window size somewhat above this guideline.

2.3 Genetic Maps

For best phasing results, it is necessary to include a genetic map in the input SNP/marker file. All three file formats HAPI-UR accepts contain a genetic map column in the input SNP/marker file. HAPI-UR will infer phase if this column contains all zeros by assuming a flat genetic map with recombination rate of 1 cM per Mb. *This is not recommended.*

The HAPI-UR distribution includes a script called `insert-map.pl` that will genetic positions from a HapMap formatted genetic map file (also called recombination rate) into a SNP/marker file. A genetic map for human genome builds 36 and 37 are available from `http://hapmap.ncbi.nlm.nih.gov/downloads/recombination/`; this is a commonly used map and is often referred to as the "Oxford Map."

To insert a genetic map into a SNP/marker file (either in Eigenstrat, packed Ancestrymap, or PLINK `.bim` format), first download the appropriate genetic maps for the genome build that you have physical positions for. For physical positions from build 36, download the build 36 genetic map from `http://hapmap.ncbi.nlm.nih.gov/downloads/recombination/2008-03_rel22_B36/rates/`. Build 37 is available from `http://hapmap.ncbi.nlm.nih.gov/downloads/recombination/2011-01_phaseII_B37/genetic_map_HapMapII_GRCh37.tar.gz`. Build 37 files have a slightly different format, with an extra column in each file; to work with `insert-map.pl`, use `awk` to remove this column (for example, on to remove this column from the chromosome 1 file, execute:

```
awk '{print $2,$3,$4}' genetic_map_GRCh37_chr1.txt > new_map_b37_chr1.txt
```

To insert the genetic positions for chromosome 1 into a SNP/marker file named data-chr1.snp, with the new file called data-genet-chr1.snp, run:

```
./insert-map.pl data-chr1.snp genetic_map_chr1_b36.txt > \
    data-genet-chr1.snp
```

This same syntax also works for PLINK .bim formatted data.

The genetic map files are separated by chromosome, and the insert-map.pl script assumes that the SNP/marker file contains information for only one chromosome. The UNIX awk utility can easily separate the SNP/marker file by chromosome. For example, the following command generates 22 files called data-chrX.snp with markers from each of the 22 autosomes from an (original) Eigenstrat format file called data.snp:

```
for c in {1..22}; do awk "\$2 == $c" data.snp > data-chr$c.snp; done
```

and the following generates files called data-chrX.bim from an (original) PLINK .bim format file called data.bim:

```
for c in {1..22}; do awk "\$1 == $c" data.bim > data-chr$c.bim; done
```

2.4 HAPI-UR Output

HAPI-UR produces output phased results in either Eigenstrat (the default) or IMPUTE2 format as described below.

Phased Eigenstrat Output

By default, HAPI-UR prints phase results in Eigenstrat format and generates four output files. The -o option gives the filename prefix for HAPI-UR to print results to, and it generates four files with different extensions that contain these results. Output files with the extensions .phgeno, .phind, and .phsnp contain the phase results in Eigenstrat format, and the file with extension .log contains the information printed to stdout during program execution (including the random seed used).

The .phgeno file contains Eigenstrat formatted phased haplotypes, with one line per SNP/marker and one character on each line corresponding to the allele at that marker in a single haplotype. Thus, each column of characters in the .phgeno file contains one haplotype. The .phind file specifies which haplotype occurs in a given column, with the order of entries in the .phind file the same as the order of the columns in the .phgeno file. The .phind file contains double the number of entries as in the input individual file (i.e., two haplotypes per individual). The ids for each haplotype in the .phind file are the original id from the input individual file with "_A" and "_B" concatenated for the two haplotypes of the individual. The order of individuals is the same as in the original individual file, with the two haplotypes for an individual appearing in succession. The .phsnp file specifies the SNP/marker information contained in the .phgeno file and is equivalent to the input SNP/marker file.

PLINK BED files provide a family and person id for each individual, but the Eigenstrat format individual file contains only id field. To produce the single id printed in the .phind output file when PLINK BED is used as input, HAPI-UR concatenates the two ids, separating them with a colon; i.e., “Family_id:Person_id”. In contrast, the IMPUTE2 output sample file has two id columns and prints the family and person id of a PLINK BED input separately in these two columns.

IMPUTE2 Format Output

When the `--impute2` argument is included in a run, HAPI-UR produces output in the file format that IMPUTE2 expects for pre-phased data. This enables ready imputation via IMPUTE2 of variants into data pre-phased using HAPI-UR. When `--impute2` is specified, HAPI-UR produces three output files with the given output filename prefix and the extensions `.haps`, `.sample`, and `.log`. The `.haps` file contains the haplotypes, with SNP/marker information contained in the first five fields of each line and allelic values in the subsequent fields on the line.

The `.sample` file contains the individual ids for the phased samples, with one entry for each individual contained in the input files (not two entries as in Eigenstrat format). When HAPI-UR is run with a PLINK BED file as input, the first column in the `.sample` file contains the family id and the second column contains the person id. For Eigenstrat and packed Ancestrymap input, the first and second columns both contain the single id that is given in the input individual file.

2.5 Trio and Duo Phasing

The PLINK `.fam` file can specify family relationships between individuals, and HAPI-UR will perform trio and duo phasing according to the relationships specified in this file. (Because Eigenstrat and packed Ancestrymap formats have no way to specify relationships, trio and duo phasing can only be done with PLINK BED format input files.) HAPI-UR only handles trios and duos, and HAPI-UR will not proceed with phasing if an individual is specified as the parent of more than one child or if an individual is both a parent and a child of some other individuals in the dataset.

Each line in a PLINK `.fam` file specifies the relationship of an individual to one or more parents. If either the father or the mother do not exist in the dataset, HAPI-UR treats the individual as an unrelated sample—that is, it acts as if no relationship was intended to be specified for that individual.

Before beginning phasing, HAPI-UR identifies any non-Mendelian errors in the trio or duo individuals and, at those sites, it sets the genotype values for all members of the trio or duo to missing. If you prefer a different style of handling non-Mendelian errors, be sure to perform that step before running HAPI-UR on the data.

2.6 Other Optional Arguments

Chromosome Number

HAPI-UR will only phase one chromosome per run. If your dataset contains more than one chromosome, specify the chromosome number for phasing with the `--chr` or `-c` option. For example, to phase chromosome 1 for the 88 HapMap CEU samples (available for download from the HAPI-UR website as hapmap-eig.tar.gz), do:

```
hapi-ur -b hapmap-eig/hapmap3_CEU -w 73 -o hapmap-chr1-phased -c 1
```

This produces output files named `hapmap-chr1-phased.phgeno`, `hapmap-chr1-phased.phind`, `hapmap-chr1-phased.phsnp` and `hapmap-chr1-phased.log`.

Effective Population Size

HAPI-UR defaults to using an effective population size of 10,000. Use the `--Ne` option to change this value. In general, modifying the effective population size does not change the phasing results substantially.

Random Number Seed

By default, HAPI-UR uses the current time when the program begins to generate a seed for the random number generator. To specify a specific random seed (and thus obtain identical results for multiple runs), use the `--seed` option. Note that for consensus phasing, you should not run HAPI-UR multiple times with the same `--seed` value, but must use different seeds.

Start and End Physical Positions

To run HAPI-UR on a section of a chromosome rather than the entire chromosome length, use the `--start` and `--end` options to specify starting and ending physical positions where phasing should occur. For example, to phase the 88 HapMap CEU samples on chromosome 1 from physical position 6,000,000 to 11,000,000, do:

```
hapi-ur -b hapmap-eig/hapmap3_CEU -w 73 -o hapmap-chr1-phased -c 1 \
        --start 6000000 --end 11000000
```

3 Consensus Phasing from Multiple Runs

The simple program `vote-phase` included with the HAPI-UR package infers consensus phase based on 3 or more phase output results. To run it, pass the phased files as arguments and redirect the output to a new file. For example, to perform consensus phasing based on files `phase1.phgeno`, `phase2.phgeno` and `phase3.phgeno`, and to save results to `out.phgeno`, do:

```
./vote-phase phase1.phgeno phase2.phgeno phase3.phgeno > out.phgeno
```

The `vote-phase` program also supports IMPUTE2 format files using the `-i` option. To perform consensus phasing based on the IMPUTE2 formatted files `phase1.haps`, `phase2.haps`, and `phase3.haps`, and to save the results to `out.haps`, do:

```
./vote-phase -i phase1.haps phase2.haps phase3.haps > out.haps
```

Sites that were missing data in the original phase file can produce different genotypes across separate phasing runs. For example, one run may infer that the two allelic values at some site that is missing data are `0,0`, while a different run may infer that the two allelic values are `0,1`. When the genotypes of the runs are different from each other, the `vote-phase` program prints the allelic values that are present in the first phased input file. Superior methods for handling this case are feasible, but this is all that is implemented at this time.

4 Eigenstrat Format

The Eigenstrat format consists of three files: an individual file, a SNP file, and a genotype file. Eigenstrat formatted genotypes for 88 HapMap CEU samples are available on the HAPI-UR webpage in the file hapmap-eig.tar.gz

The individual file contains one sample per line with three columns: an alpha-numeric sample id, a character indicating the sample gender (M for male, F for female, U for unknown), and an alpha-numeric population label. Here are three lines from an example individual file:

```
NA12749 F        CEU
NA12748 M        CEU
NA12890 F        CEU
```

The SNP file contains one line per SNP and consists of six columns: an alpha-numeric SNP id, the chromosome number, the genetic position in Morgans (HAPI-UR uses this field to determine recombination probabilities), the physical position, the reference allele, and the alternate allele. Here are three lines from an example SNP file:

```
rs4040617    1        0.000788          769185 A G
rs10907175   1        0.007094         1120590 A C
rs3766180    1        0.010306         1468016 C T
```

Note: it is important to include the genetic distance values in Morgans in order for HAPI-UR to infer accurate haplotypes through the use of appropriate transition probabilities between each state in its model. See Section 2.3 for information on inserting a genetic map into a Eigenstrat SNP file.

The genotype file contains one line per SNP, with the genotypes on each line being data corresponding to the SNP at the same line number in the SNP file. Each character encodes the genotype of one individual at the given SNP. Values at a given column number correspond to data for the individual at the same line number in the individual file.

Each genotype value is a count of the number of reference alleles an individual carries, with 2 encoding homozygous reference, 1 encoding heterozygous, and 0 encoding homozygous alternate. A value of 9 indicates the individual is missing data at the given locus. Here are example genotype values for the individuals and the SNP positions shown above:

```
211
222
121
```

Here, the first line indicates that at rs4040617, NA12749 is homozygous reference and thus has genotype A/A, and NA12748 and NA12890 are both heterozygous with genotype A/C. The second line indicates that at rs10907175, NA12749, NA12748, and NA12890 are all homozygous reference or A/A. The third line indicates that at rs3766180, NA12749 is heterozygous with genotype C/T, NA12748 is homozygous reference with genotype C/C, and NA12890 is heterozygous with genotype C/T.

5 Bug Reports

Please submit bug reports using the interface on the HAPI-UR homepage http://hapi-ur.googlecode.com/ under the issues tab and feel free to send an email to Amy Williams (`alw@genetics.med.harvard.edu`).

6 Citing HAPI-UR

If you use HAPI-UR, please cite it as follows:

Williams AL, Patterson N, Glessner J, Hakonarson H, and Reich D (2012). Phasing of many thousands of genotyped samples. American Journal of Human Genetics 91:238-251.

`http://www.cell.com/AJHG/abstract/S0002-9297%2812%2900322-9`

7 Compling HAPI-UR

HAPI-UR requires the boost library (for the dynamic_bitset implementation; see `http://www.boost.org/`) and Google's dense hash set and dense hash map (included in the sparsehash project; see `http://sparsehash.googlecode.com/`). We made minor modifications to Google's dense hash map to improve efficiency when storing NULL values. The file `sparsehash-mod/densehashtable.h` included in the HAPI-UR release should replace Google's `sparsehash/internal/densehashtable.h`. Without this change, HAPI-UR will crash by attempting to dereference a NULL pointer.

This distribution includes Makefiles for use in a standard Linux environment. To build HAPI-UR, do the following:

1. Make the genetio library:

```
cd lib/genetio
make clean
make
cd ../..
```

2. Make HAPI-UR:

```
cd src
make clean
make
cd ..
```

To make the vote-phase program, make the genetio library as above and do:

```
cd src
make vote-phase
cd ..
```