Hapl-o-Mat - Getting Started Linux

Please also see the README.

Hapl-o-Mat

Hapl-o-Mat is software for HLA haplotype inference coded in C++. Besides estimating haplotype frequencies via an expectation-maximization algorithm, it is capable of processing HLA genotype population data. This includes translation of alleles between various typing resolutions and resolving allelic and genotypic ambiguities. Both common formats for recording HLA genotypes, multiple allele codes (MAC) and genotype list strings (GLS), are supported.

For more information refer to our publications on Hapl-o-Mat:

> Schäfer C, Schmidt AH, Sauter J. Hapl-o-Mat: open-source software for HLA haplotype frequency estimation from ambiguous and heterogeneous data. BMC Bioinformatics. 2017; 18(1):284. doi: 10.1186/s12859-017-1692-y.

> Sauter J, Schäfer C, Schmidt AH. HLA Haplotype Frequency Estimation from Real-Life Data with the Hapl-o-Mat Software. Methods Mol Biol. 2018; 1802:275-284. doi: 10.1007/978-1-4939-8546-3_19.

> Solloch UV, Schmidt AH, Sauter J. Graphical user interface for the haplotype frequency estimation software Hapl-o-Mat. Hum Immunol. 2022; 83(2):107-112. doi: 10.1016/j.humimm.2021.11.002.

If you use Hapl-o-Mat for your research, please cite preferably the journal articles.

Getting Started

This guide is an introduction on how to use Hapl-o-Mat under Linux. In order to follow this guide, you need a Linux system, a C++ compiler supporting C++11, and Python. In this tutorial, we use Ubuntu 14.04.4 LTS and GNU compiler collection (GCC) version 4.8.4. This Ubuntu version comes with Python. We process every step from the terminal. Of course, you can use some tool with a graphic interface to move files, create folders, and so on. If you are a seasoned Linux-User, feel free to refer to the shorter version of this guide, "Hapl-o-Mat/gettingStarted".

After successfully downloading Hapl-o-Mat, start a terminal and browse to the location where Hapl-o-Mat is stored. Enter the folder Hapl-o-Mat by typing "cd Hapl-o-Mat". Check what is inside by typing "ls". You should see the following files, where we mark important files for using Hapl-o-Mat as bold.

File name	Description
COPYING	The GNU General Public License
detailedGettingStartedLinux	Guide for using Hapl-o-Mat under Linux
detailedGettingStartedWindows	Guide for using Hapl-o-Mat under Windows
examplePopulations	Some genotype population data we are going to work with in the section Tutorials.
gettingStarted	A shorter form of this tutorial
include	A part of Hapl-o-Mat's source code; If you do not want to change code, do not touch it.

Makefile	Instructions for building Hapl-o-Mat; You might need to adapt it, if you use another compiler than GCC.
parametersGLS, parametersGLSC, parametersMAC, parametersREAD	Parameter files for Hapl-O-mat; We are going to discuss these in section Parameters.
prepareData	Here is everything to create the data required by Hapl-o-Mat.
README.md	Read me
src	Another part of Hapl-o-Mat's source code; If you do not want to change code, do not touch it.
systemTest	Run the system test after changing code to check, if you broke something. Refer to its README.
textsForGettingStarted	Raw files for the guides including this guide
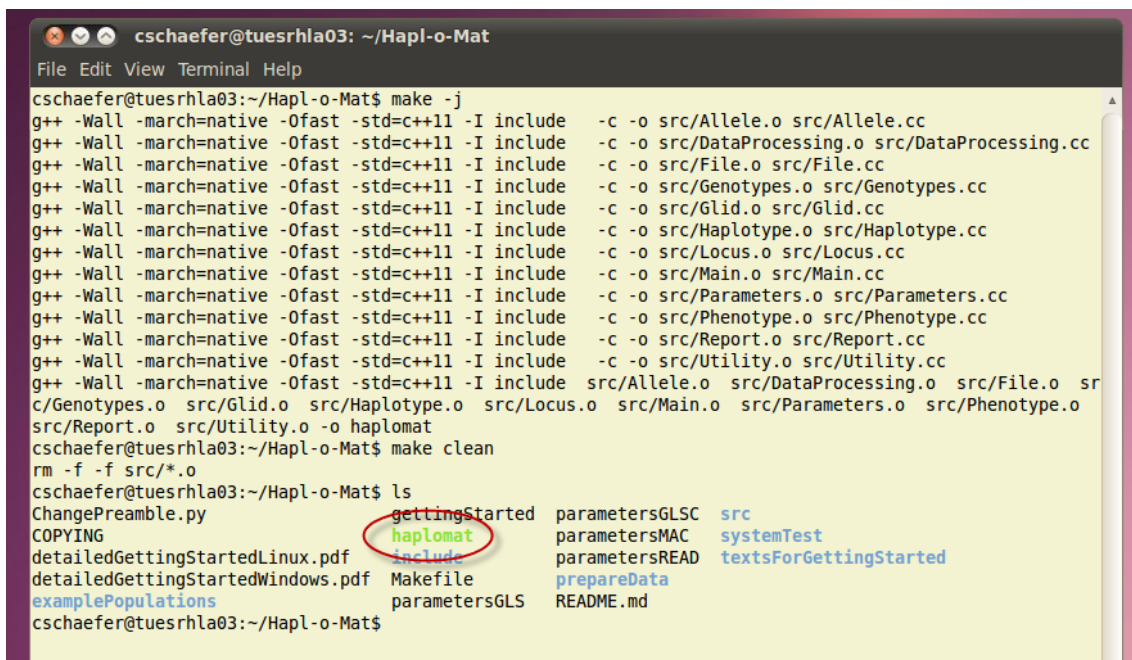
To estimate haplotype frequencies we only need to consider the folder prepareData and the files Makefile, parametersGLS, parametersGLSC, parametersMAC, and parametersREAD. To finish the tutorials we need the folder examplePopluations, too.

Install Hapl-o-Mat

We compile Hapl-o-Mat with GCC using a Makefile. Just enter the folder "Hapl-o-Mat" and type "make" to create the executable "haplomat" (Run "make -j" to use more than one core for compiling). Type "make clean" to clean up. Use the command "ls" to find a new file, haplomat.

Data Preparation

Hapl-o-Mat relies on information on the HLA nomenclature. This information is provided by data files, which we are going to create. As the HLA nomenclature evolves over time, e.g. by finding new alleles or adding new multiple allele codes, it is important to consider updating this information from time to time to allow new alleles to be handled by Hapl-o-Mat. In rare cases, alleles are also removed from the nomenclature (see section "Invalid/Deprecated Alleles" in "Hapl-o-Mat/prepareData/detailedExplanationPrepareData.pdf"), which results in input genotypes with such

alleles being removed from the haplotype frequency analysis by Hapl-o-Mat. Thus, rerunning older analyses can behave differently.

Hapl-o-Mat relies on the following files, which must be placed in the folder "Hapl-o-Mat/data" for Hapl-o-Mat to work:

File name	Description
AllAllelesExpanded.txt	A list of relevant existing HLA alleles with their enclosed more-digit typing resolutions
AlleleList.txt	If your input data in GLS format includes a missing single-locus genotype, this missing locus information can be treated as an ambiguity that can be resolved either by insertion of all alleles of the respective locus that are represented in your input file or by all known alleles of this locus. AlleleList.txt is only required if you are going to use this feature.
Ambiguity.txt	Data for the ambiguity filter
LargeG.txt	A list of G-groups with their enclosed alleles in 8-digit resolution
MultipleAlleleCodes.txt	A list of multiple allele codes and their translation to alleles in 4-digit resolution
P.txt	A list of P-groups with their enclosed alleles in 8-digit resolution
Smallg.txt	A list of g-groups with their enclosed alleles in 8-digit resolution

In the following we are going to create these data files. As the data-processing is a little bit tedious, we provide you with an automated script. If you want to build the data manually, follow the short instructions in "Hapl-o-Mat/prepareData/README" or the detailed version in "Hapl-o-Mat/prepareData/detailedExplanationPrepareData.pdf".

To build data automatically, enter the folder "Hapl-o-Mat/prepareData" und just run

```
python BuildData.py
```

to download all relevant data, process them, and move the created files to folder "Hapl-o-Mat/data".

The script accesses a default set of publicly available repositories. These repositories are denoted in the file "url_config.txt". You should be fine to work with the initial settings. However, if these repositories were to move, or you'd choose to use other sources, this can be adjusted here. For details please see the description in "detailedExplanationPrepareData.pdf".

If the script terminates due to a connection time out, a proxy or a firewall issue, you can still use the command "python BuildData.py" but you have to download certain files manually. Please see the document "detailedExplanationPrepareData.pdf" for a detailed description; refer there to the section "Semi-Automated Way".

Please see the document "detailedExplanationPrepareData.pdf" for a detailed description of the creation of AlleleList.txt.

Run Hapl-o-Mat

Now, you are able to run Hapl-o-Mat for the first time. Enter the folder "Hapl-o-Mat" and run

```
./haplomat MAC
```

We explain the output and the meaning of "MAC" in the following sections.

Input Genotype Data

Hapl-o-Mat infers haplotypes from population genotype data. It supports different formats of recording genotype data. To use Hapl-o-Mat, your data should be in one of the following data formats:

Data format	Description
MAC	**M**ultiple **A**llele **C**odes: ambiguities are encoded by multiple allele codes (MAC). Except for the first line, input files hold an individual's identification number and genotype per line. Genotypes are saved allele by allele without locus name. Identification number and alleles are TAB-separated. The first line of the file is a header line indicating the name of the first column and the loci of the other columns. Same loci must be placed next to each other. For an example refer to "Hapl-o-Mat/examplePopulations/populationData_a.dat".
GLSC	**G**enotype **L**ist **S**trings **C**olumn-wise: genotypes with or without ambiguities are saved by genotype list strings (GLS). Input files hold an individual's identification number and genotype per line. Identification number and single-locus genotypes are TAB-separated. For an example refer to "Hapl-o-Mat/examplePopulations/populationData_b.dat".
GLS	**G**enotype **L**ist **S**trings: genotypes with or without ambiguities are saved by genotype list strings (GLS). Population data is saved in two files. The pull-file contains an individual's identification number and a list of integer numbers, GLS-ids, referring to its single-locus genotype. The GLS-ids are separated from the identification number via ";" and from each other via ":". The second file, the glid-file, contains a translation from GLS-ids starting with "1" to actual single-locus genotypes. GLS-id and genotype are separated via ";". A GLS-id of "0" is interpreted as a missing typing at the corresponding locus and does not require a translation in the glid-file. For an example refer to "Hapl-o-Mat/examplePopulations/populationData_c.pull" and "Hapl-o-Mat/examplePopulations/populationData_c.glid".
READ	**READ**: ambiguities are completely resolved and alleles are already translated to the wanted typing resolutions. The input data is of the format as Hapl-o-Mat records processed genotype data. This allows for easily repeating a run without the need to resolve genotype data again.

Transform input genotype data to GLS format

Only in GLS format, missing single-locus genotypes can be handled. It is assigned the GLS-id=0. The parameter RESOLVE_MISSING_GENOTYPES in the "parametersGLS" file offers the option (when set to "true") to replace a missing single-locus genotype by a combination of all alleles from AlleleList.txt at the locus.

If your input data in MAC or GLSC format contains missing single-locus genotypes, you have the possibility to transform it to GLS input format:

Transform GLSC to GLS input: Run the script "GLSC2GLS.py" in folder "manageInput/input2GLS". You need to add the input file as argument to the command line. A valid command would be e.g.

```
python3 GLSC2GLS.py ../../examplePopulations/populationData_b.glc
```
. A single missing typing will be treated as homozygous, i.e. the present typing will be duplicated.

Transform MAC to GLS input: Run the script "MAC2GLS.py" in folder "manageInput/input2GLS". During the transformation of the MAC input you have additional option to replace a single missing typing at a locus as follows:

'remove'   treat as empty locus (remove present allele)
'homozygous'   treat as homozygous (duplicate present allele)
'any'   treat as any allele of the locus (according to AlleleList.txt)

You need to add the input file AND the option for missing single typings as argument to the command line. A valid command would be e.g.

```
python3 MAC2GLS.py ../../examplePopulations/populationData_a.dat homozygous
```
.

The resulting input files in GLS format (one .glid and one .pull file) are stored in folder "manageInput/input2GLS/results".

Absent loci DRB3/4/5

Some HLA loci may be absent in a genotype, for example the loci DRB3, DRB4 and DRB5. For these three loci a missing locus has to be denoted as "NNNN" in the input file in order to be adequately processed by Hapl-o-Mat (DRB3*NNNN, DRB4*NNNN, DRB5*NNNN, only "NNNN" in the respective column in MAC input format!).

If your input data for these loci contains a different abbreviation for missing alleles, please change file "data/AllAllelesExpanded.txt" after data preparation by replacing the three lines (end of file)

DRB3*NNNN   DRB3*NNNN
DRB4*NNNN   DRB4*NNNN
DRB5*NNNN   DRB5*NNNN

the "NNNN" with a code of your liking. Please do not use Word for the file manipulation to avoid changing the EOL characters to Windows format. Please note that the code must not overlap with existing allele names or MAC.

Parameters

Each input format for genotype data requires a different set of parameters. The parameters are saved in the corresponding files "parametersMAC", "parametersGLSC", "parametersGLS", and "parametersREAD". All input formats have the following parameters in common:

Parameter	Description
FILENAME_HAPLOTYPES	Name of the file which temporarily saves haplotype names
FILENAME_GENOTYPES	Name of the file which saves resolved genotypes
FILENAME_HAPLOTYPEFREQUENCIES	Name of the file which saves haplotypes and estimated haplotype frequencies
FILENAME_EPSILON_LOGL	Name of the file which saves stopping criterion and log-likelihood per iteration
INITIALIZATION_HAPLOTYPEFREQUENCIES	Initialization routine for haplotype frequencies. It takes the following values:

	• "equal": All haplotype frequencies are initialized with the same frequency, the inverse number of haplotypes.
	• "numberOccurrence": Haplotype frequencies are initialized according to the initial number of occurrence of haplotypes.
	• "random": Haplotype frequencies are initialized randomly.
	• "perturbation": Haplotype frequencies are initialized as in numberOccurrence and then randomly modified by a small (<10%) positive or negative offset.
EPSILON	Value for the stopping criterion, i.e. the maximal change between consecutive haplotype frequency estimations is smaller than the assigned value
CUT_HAPLOTYPEFREQUENCIES	Estimated haplotype frequencies smaller than this value are removed from the output.
RENORMALIZE_HAPLOTYPEFREQUENCIES	Takes values "true" and "false". If "true", normalize estimated haplotype frequencies to sum to one. Within machine precision, this becomes necessary, if estimated haplotypes are removed, e.g. via the option CUT_HAPLOTYPEFREQUENCIES.
SEED	Set the seed of the used pseudo random number generator. If set to "0", the seed is initialized by the system time.

Depending on the input format, additional parameters are:

Parameter	Input format	Description
FILENAME_INPUT	MAC, GLSC, READ	The file name of the input population data
FILENAME_PULL	GLS	The file name of the pull-file
FILENAME_GLID	GLS	The file name of the glid-file
LOCI_AND_ RESOLUTIONS	MAC, GLS, GLSC	Loci included into analysis and desired typing resolution per locus; The list is separated by "," and contains the locus name followed by ":" and the desired typing resolution, e.g. A:g,B:4d,C:g. Supported typing resolutions and their abbreviations are g-groups (g), P-groups (P), G-groups (G), first fields (1f), second fields (2f), three fields (3f), and four fields (4f). Alleles are not translated via the option asItIs (applying the ambiguity filter includes an intrinsic translation to G-groups).
LOCIORDER	GLS	Specify the order of loci the individual's GL-ids correspond to. Loci are separated via ",".
RESOLVE_MISSING_ GENOTYPES	GLS	Takes values "true" and "false". If set to true, a missing typing is replaced by a combination of all alleles from

		AlleleList.txt at the locus. Else, individuals with a missing typing are discarded from analysis.
MINIMAL_FREQUENCY_GENOTYPES	MAC, GLS, GLSC	Genotypes which split into more genotypes than the inverse of this number are discarded from analysis.
DO_AMBIGUITYFILTER	MAC, GLS, GLSC	Takes values "true" and "false". The option "true" activates the ambiguity filter.
EXPAND_LINES_AMBIGUITYFILTER	MAC, GLS, GLSC	Takes values "true" and "false". If set to "true", matching lines with additional genotype pairs in the ambiguity filter are considered.
WRITE_GENOTYPES	MAC, GLS, GLSC	Takes values "true" and "false". If "false", no file which saves resolved genotypes (FILENAME_GENOTYPES) will be written out.

Whenever specifying a file name including folders, you have to create the folders before running Hapl-o-Mat.

Quick Guide

The following overview gives you a small reminder on how to use Hapl-o-Mat:

1) Build the executable "haplomat" via make.

2) Update or build the data comprising information on the HLA nomenclature using the python-script "Hapl-o-Mat/prepareData/BuildData.py".

3) Prepare the genotype population data you want to study. Identify how genotyping ambiguities are recorded (MAC or GLS) and choose the input format accordingly. Adapt the format of your data, e.g. include the header line or make alleles TAB separated.

4) Set the parameters in the parameter file corresponding to your input format.

5) Copy the executable "haplomat", the folder "data", the parameter file, and your input population data into one folder. Create any folders you specified in the parameter file. You do not need all the other files to run Hapl-o-Mat. Run Hapl-o-Mat.

Tutorials

We have everything ready to use Hapl-o-Mat. In the following we estimate haplotype frequencies from some included genotype data recorded in different input formats. For all formats we are going to infer three locus (A, B, DRB1) haplotypes from this data. Alleles at loci A and B shall be translated to typing resolution g and alleles at locus DRB1 to 4-digits typing resolution.
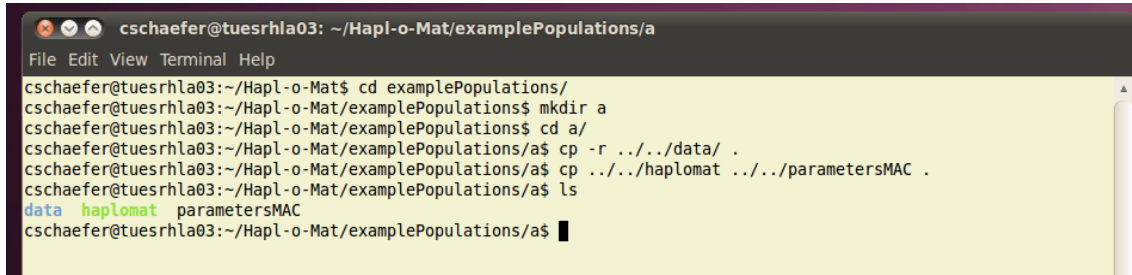
Input Format MAC

You find the relevant population data in "Hapl-o-Mat/examplePopulations/populationData_a.dat". As ambiguities are recorded as multiple allele codes, the input format is MAC.

Preparations

Enter the folder "Hapl-o-Mat/examplePopulations" by typing "cd examplePopulations", create a folder named "a" by typing "mkdir a", and enter the folder by typing "cd a". Then provide the data

required by Hapl-o-Mat by copying the folder "Hapl-o-Mat/data" to "a", "cp -r ../../data .".
Additionally, copy the executable "haplomat" and the file "parametersMAC" to folder "a", "cp
../../haplomat ../../parametersMAC .". Check that everything is there by typing "ls".

Additionally, create the folder "run" in folder "a" by typing "mkdir run".

Parameters

According to the format of the input genotype data we use the parameter file "parametersMAC". Open
it in a text editor of your choice and set the following values:

```
#file names
FILENAME_INPUT=../populationData_a.dat
FILENAME_HAPLOTYPES=run/haplotypes.dat
FILENAME_GENOTYPES=run/genotypes.dat
FILENAME_HAPLOTYPEFREQUENCIES=run/hfs.dat
FILENAME_EPSILON_LOGL=run/epsilon.dat
FILENAME_ANALYTICS=run/analytics.dat
#reports
LOCI_AND_RESOLUTIONS=A:g,B:g,DRB1:1f
MINIMAL_FREQUENCY_GENOTYPES=1e-5
DO_AMBIGUITYFILTER=false
EXPAND_LINES_AMBIGUITYFILTER=false
WRITE_GENOTYPES=true
DO_ANALYTICS=false
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=1000
```

Run Hapl-o-Mat

Compute haplotype frequencies from the genotype input data by running Hapl-o-Mat. If you are not
already there, go to folder "a" and run Hapl-o-Mat via

```
./haplomat MAC
```

It produces some output on the screen including your chosen parameters, statistics on the resolved
genotype data and the expectation-maximization algorithm, and the run time. You can easily write this
output to an extra file by starting Hapl-o-Mat with

./haplomat MAC > Log.dat

```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/a
File  Edit  View  Terminal  Help


cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/a$ mkdir run
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/a$ ls
data  haplomat  parametersMAC  run
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/a$ ./haplomat MAC

        Hapl-o-Mat
        Copyright (C) 2016 DKMS gGmbH

#########Initialization
        MA format
#########Parameters I/O
        Input: ../populationData_a.dat
        Output haplotypes: run/haplotypes.dat
        Output genotypes: run/genotypes.dat
        Output estimated haplotype frequencies: run/hfs.dat
        Output epsilon and log(L): run/epsilon.dat
#########Parameters resolving genotypes
        Minimal frequency of genotypes: 1e-05
        Loci with target allele resolutions:
        A : g
        B : g
        DRB1 : 4d
        Apply ambiguity filter: no
#########Parameters EM algorithm
        Haplotype frequency initialization: perturbation
        Epsilon: 1e-06
        Cut haplotype frequencies: 1e-06
        Renormalize haplotype frequencies
        Zero: 1e-14
```
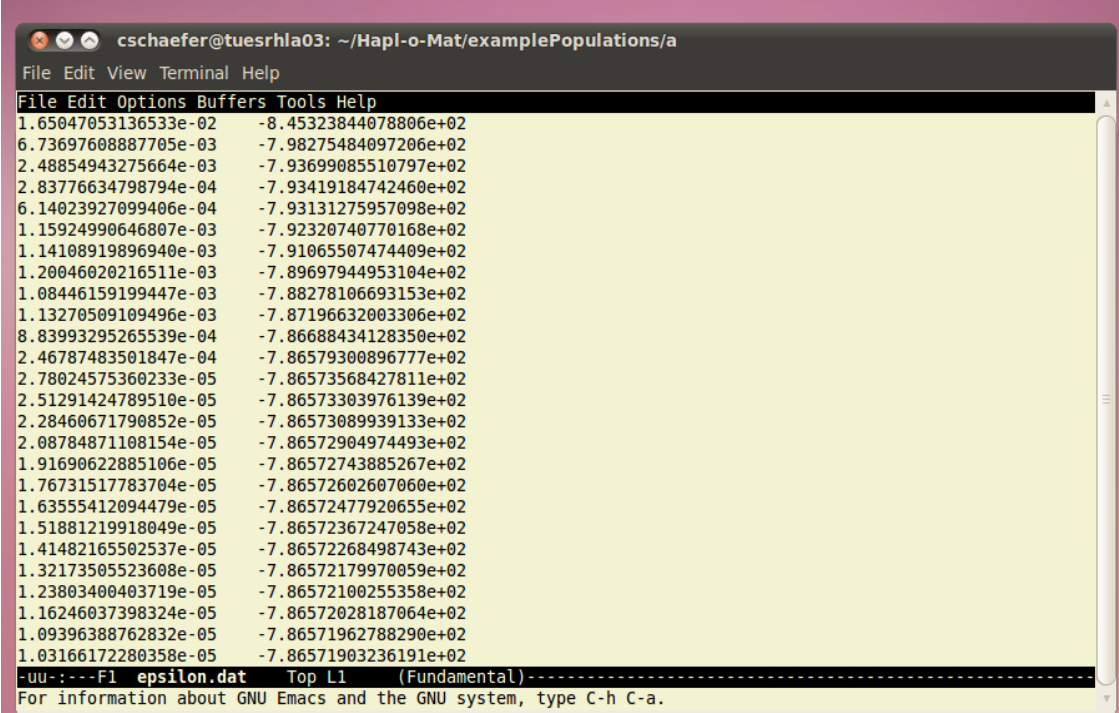
Results

Now let's examine the results produced by Hapl-o-Mat. We first look into the file with the resolved genotypes, "run/genotypes.dat".

```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/a
File  Edit  View  Terminal  Help
File Edit Options Buffers Tools Help
1       NNN      1         A*26:04+A*29:67^B*07:218+B*54:16^DRB1*11:182+DRB1*13:14
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:01^DRB1*12:01+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:01^DRB1*12:01+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:14^DRB1*12:01+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:14^DRB1*12:01+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:19^DRB1*12:01+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:19^DRB1*12:01+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:01^DRB1*12:06+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:01^DRB1*12:06+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:14^DRB1*12:06+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:14^DRB1*12:06+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:19^DRB1*12:06+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:19^DRB1*12:06+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:01^DRB1*12:10+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:01^DRB1*12:10+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:14^DRB1*12:10+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:14^DRB1*12:10+DRB1*14:23
2       III      0.098765432098765        A*02:570+A*24:02g^B*13:07N+B*14:19^DRB1*12:10+DRB1*14:23
2       III      0.012345679012346        A*02:570+A*24:50^B*13:07N+B*14:19^DRB1*12:10+DRB1*14:23
3       NNN      1         A*02:570+A*29:67^B*13:07N+B*35:54^DRB1*13:121+DRB1*14:23
4       NIN      0.33333333333333         A*02:77+A*66:10^B*14:01+B*15:154^DRB1*16:30+DRB1*16:30
4       NIN      0.33333333333333         A*02:77+A*66:10^B*14:14+B*15:154^DRB1*16:30+DRB1*16:30
4       NIN      0.33333333333333         A*02:77+A*66:10^B*14:19+B*15:154^DRB1*16:30+DRB1*16:30
5       NNN      1         A*03:217+A*29:36^B*15:01g+B*35:54^DRB1*11:95+DRB1*14:23
6       NNN      1         A*02:454+A*03:93^B*15:316+B*35:54^DRB1*04:52+DRB1*08:18
7       NNN      1         A*24:02g+A*30:13^B*18:19+B*39:27^DRB1*13:116+DRB1*14:23
-uu-:---F1  genotypes.dat   Top L1     (Fundamental)-----------------------------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```

The first column corresponds to the individual's identification number. The second column indicates how ambiguities per single-locus genotypes have been resolved. If no ambiguity occurred or no additional genotypes are formed, the type is N. If an ambiguity occurred and was resolved via building

all possible allele combinations, the type is I. Activating the ambiguity filter gives additional types: A, if one matching line in the ambiguity file was found, and M if multiple matching lines were found. The third column gives the frequency of the genotype and the fourth column the genotype itself. The genotype is saved in the GLS format. If an individual's genotype splits into a set of genotypes, each genotype is written to one line starting with the same identification number. The corresponding frequencies become non-integer and sum to one.

The evolution of the stopping criterion and log-likelihood while iterating expectation and maximization steps is written to "run/epsilon.dat". The first column is the stopping criterion and the second one the not normalized log-likelihood.

```
   cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/a

 File  Edit  View  Terminal  Help
File Edit Options Buffers Tools Help
1.65047053136533e-02      -8.45323844078806e+02
6.73697608887705e-03      -7.98275484097206e+02
2.48854943275664e-03      -7.93699085510797e+02
2.83776634798794e-04      -7.93419184742460e+02
6.14023927099406e-04      -7.93131275957098e+02
1.15924990646807e-03      -7.92320740770168e+02
1.14108919896940e-03      -7.91065507474409e+02
1.20046020216511e-03      -7.89697944953104e+02
1.08446159199447e-03      -7.88278106693153e+02
1.13270509109496e-03      -7.87196632003306e+02
8.83993295265539e-04      -7.86688434128350e+02
2.46787483501847e-04      -7.86579300896777e+02
2.78024575360233e-05      -7.86573568427811e+02
2.51291424789510e-05      -7.86573303976139e+02
2.28460671790852e-05      -7.86573089939133e+02
2.08784871108154e-05      -7.86572904974493e+02
1.91690622885106e-05      -7.86572743885267e+02
1.76731517783704e-05      -7.86572602607060e+02
1.63555412094479e-05      -7.86572477920655e+02
1.51881219918049e-05      -7.86572367247058e+02
1.41482165502537e-05      -7.86572268498743e+02
1.32173505523608e-05      -7.86572179970059e+02
1.23803400403719e-05      -7.86572100255358e+02
1.16246037398324e-05      -7.86572028187064e+02
1.09396388762832e-05      -7.86571962788290e+02
1.03166172280358e-05      -7.86571903236191e+02
-uu-:---F1  epsilon.dat     Top L1     (Fundamental)-------------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```

The inferred haplotypes including estimated frequencies are listed in "run/hfs.dat". Haplotypes are saved in the GLS format. This is the file you were aiming at. It is sorted by descending frequency and already normalized if you activated the corresponding option (we did in this tutorial).

```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/a
File  Edit  View  Terminal  Help
File Edit Options Buffers Tools Help
A*29:25~B*15:101~DRB1*16:30       0.06000000000000
A*11:01g~B*40:01g~DRB1*07:56      0.04000000000000
A*24:02g~B*15:27~DRB1*14:85       0.03000000000000
A*68:70~B*37:19~DRB1*13:52        0.03000000000000
A*11:01g~B*15:316~DRB1*11:182     0.03000000000000
A*30:13~B*18:12~DRB1*11:147       0.02500000000000
A*32:33~B*54:16~DRB1*11:182       0.02500000000000
A*03:02g~B*35:32~DRB1*09:02       0.02500000000000
A*30:73N~B*14:08~DRB1*14:01       0.02013379227088
A*26:04~B*54:16~DRB1*11:182       0.02000000000000
A*03:217~B*35:51~DRB1*14:85       0.02000000000000
A*03:93~B*35:54~DRB1*04:52        0.02000000000000
A*02:454~B*15:316~DRB1*08:18      0.02000000000000
A*03:217~B*15:01g~DRB1*11:95      0.02000000000000
A*02:570~B*13:07N~DRB1*14:23      0.02000000000000
A*66:10~B*15:154~DRB1*16:30       0.01500000000000
A*68:70~B*35:32~DRB1*13:66        0.01500000000000
A*03:217~B*52:49N~DRB1*08:30      0.01500000000000
A*32:73~B*51:114~DRB1*07:56       0.01500000000000
A*29:67~B*35:54~DRB1*13:121       0.01500000000000
A*03:239~B*27:98~DRB1*03:103      0.01500000000000
A*02:258~B*42:14~DRB1*13:121      0.01500000000000
A*68:94N~B*39:27~DRB1*03:103      0.01500000000000
A*66:10~B*51:01g~DRB1*04:52       0.01500000000000
A*02:163~B*37:28~DRB1*16:30       0.01500000000000
A*23:06~B*08:145~DRB1*16:30       0.01500000000000
-uu-:---F1  estimatedHaplotypeFrequencies.dat  Top L1    (Fundamental)-------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```

Input Format GLSC

This time ambiguities in the genotypic population data are recorded via genotype list strings. The file with the population data is called "populationData_b.glc". As all the information is in one file, the input format is GLSC. Running Hapl-o-Mat works exactly as in the first tutorial. You just use the parameter file "parametersGLSC" instead of "parametersMAC" and make the appropriate changes. Run Hapl-o-Mat in folder "b" with

> ./haplomat GLSC


```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/b
File  Edit  View  Terminal  Help
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations$ ls
a  populationData_a.dat  populationData_b.glc  populationData_c.glid  populationData_c.pull
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations$ mkdir b
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations$ cd b/
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/b$ cp -r ../../data/ ../../haplomat ../../parametersGLSC .
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/b$ e parametersGLSC
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/b$ mkdir run
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/b$ ls
data  haplomat  parametersGLSC  parametersGLSC~  run
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/b$ ./haplomat GLSC

        Hapl-o-Mat
        Copyright (C) 2016 DKMS gGmbH

#########Initialization
        GLSC format
#########Parameters I/O
        Input: ../populationData_b.dat
        Output haplotypes: run/haplotypes.dat
        Output genotypes: run/genotypes.dat
        Output estimated haplotype frequencies: run/hfs.dat
        Output epsilon and log(L): run/epsilon.dat
#########Parameters resolving genotypes
        Minimal frequency of genotypes: 1e-05
        Loci with target allele resolutions:
        A : g
        B : g
        DRB1 : 4d
        Apply ambiguity filter: no
#########Parameters EM algorithm
        Haplotype frequency initialization: perturbation
```

```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/b
File Edit View Terminal Help
File Edit Options Buffers Tools Help
#file names
FILENAME_INPUT=../populationData_b.glc
FILENAME_HAPLOTYPES=run/haplotypes.dat
FILENAME_GENOTYPES=run/genotypes.dat
FILENAME_HAPLOTYPEFREQUENCIES=run/estimatedHaplotypeFrequencies.dat
FILENAME_EPSILON_LOGL=run/epsilon.dat
#reports
LOCI_AND_RESOLUTIONS=A:g,B:g,DRB1:4d
MINIMAL_FREQUENCY_GENOTYPES=1e-5
DO_AMBIGUITYFILTER=false
EXPAND_LINES_AMBIGUITYFILTER=false
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=0




-uu-:---F1  parametersGLC   All L1    (Fundamental)------------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```

Input Format GLS

Again, ambiguities in the genotypic population data are recorded via genotype list strings. Since the data is saved in two different files, the input format is GLS. Follow the steps from tutorial a), but use the parameter file "parametersGLS" and name the created folder "c". The file names for the population data are "populationData_c.pull" and "populationData_c.glid". I guess, you can figure out the matching positions in the parameter file. GLS input format requires the order of loci as input, which can be obtained by looking in the pull- and glid-file. The first individual from "populationData_c.pull" has GLS-ids 1, 2, 3, 4, 5, and 6. We know from "populationData_c.pull" that they correspond to loci B, A, DPB1, DRB1, C, and DQB1, respectively. Because of that we set "LOCIORDER=B,A,DPB1,DRB1,C,DQB1". Finally, set the additional option RESOLVE_MISSING_GENOTYPE to "false". Run Hapl-o-Mat in folder "c" with

    ./haplomat GLS

Input Format READ

Finally, we test the input format READ. Create a folder "d" and copy one file with resolved genotypes, say "a/run/genotypes.dat" there. Add "haplomat" and "parametersREAD" to this folder. Using the input format READ, Hapl-o-Mat does not resolve ambiguities or translates alleles, but reads in already resolved genotype data. Because of that the folder "data" is not required and the parameter file "parameterREAD" misses some options. Just adjust the file names and set parameters for the haplotype frequency estimation. Run Hapl-o-Mat in folder "d" via

```
./haplomat READ
```

```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/d

File  Edit  View  Terminal  Help
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations$ ls
a  b  c  populationData_a.dat  populationData_b.glc  populationData_c.glid  populationData_c.pull
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations$ mkdir d
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations$ cd d
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ cp ../a/run/genotypes.dat .
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ cp ../../parametersREAD .
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ cp ../../haplomat .
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ e parametersREAD
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ mkdir run
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ ls
genotypes.dat  haplomat  parametersREAD  parametersREAD~  run
cschaefer@tuesrhla03:~/Hapl-o-Mat/examplePopulations/d$ ./haplomat READ

        Hapl-o-Mat
        Copyright (C) 2016 DKMS gGmbH


#########Initialization
        Readin format
#########Parameters I/O
        Input: genotypes.dat
        Output haplotypes: run/haplotypes.dat
        Output estimated haplotype frequencies: run/estimatedHaplotypeFrequencies.dat
        Output epsilon and log(L): run/epsilon.dat
#########Parameters EM algorithm
        Haplotype frequency initialization: perturbation
        Epsilon: 1e-06
        Cut haplotype frequencies: 1e-06
        Renormalize haplotype frequencies
        Zero: 1e-14
        Seed: 1464859288806011387


#########Data preprocessing
```



```
cschaefer@tuesrhla03: ~/Hapl-o-Mat/examplePopulations/d

File  Edit  View  Terminal  Help
File Edit Options Buffers Tools Help
#file names
FILENAME_INPUT=genotypes.dat
FILENAME_HAPLOTYPES=run/haplotypes.dat
FILENAME_HAPLOTYPEFREQUENCIES=run/estimatedHaplotypeFrequencies.dat
FILENAME_EPSILON_LOGL=run/epsilon.dat
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=0








-uu-:---F1  parametersREAD    All L1     (Fundamental)------------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```