

Hapl-o-Mat - Getting Started Windows

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 on Windows OS. In order to follow this guide, you need a Windows system, a C++ compiler supporting C++11, and Python (at least version 2.7.11).

In this tutorial, we use Visual Studio Code for compiling Hapl-o-Mat. You can get it here

<https://code.visualstudio.com/>. Python for Windows can be found here

<https://www.python.org/downloads/windows/>.

After successfully downloading Hapl-o-Mat, look into its folder. 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

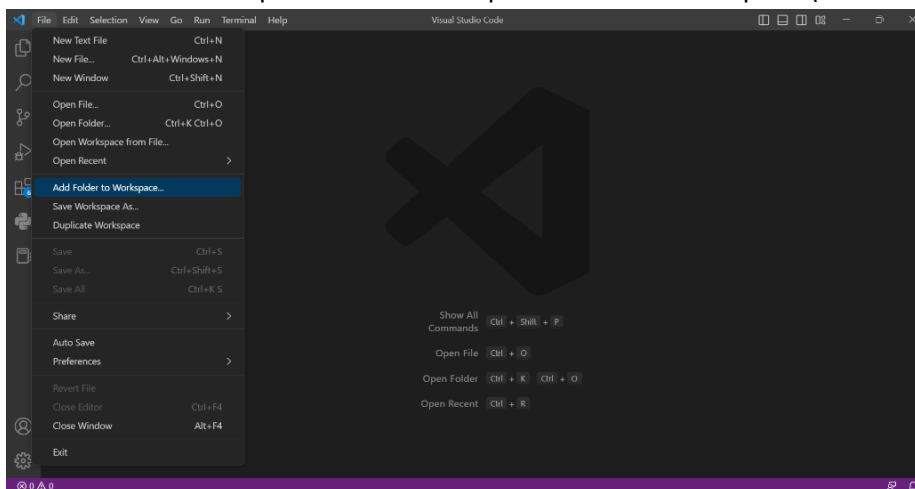
To estimate haplotype frequencies we only need to consider the folder prepareData and the files parametersGLS, parametersGLSC, parametersMAC, and parametersREAD. To finish this tutorial we need the folder examplePopulations, too.

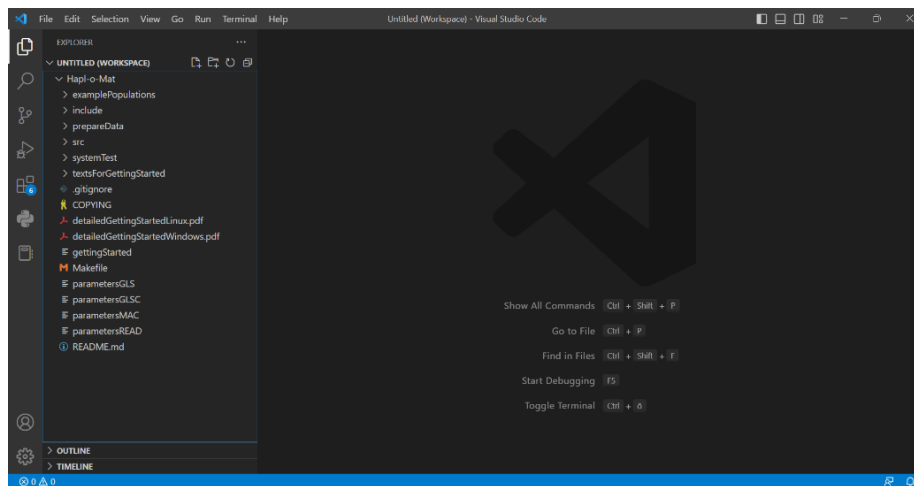
Install Hapl-o-Mat

We compile Hapl-o-Mat using VS Code.

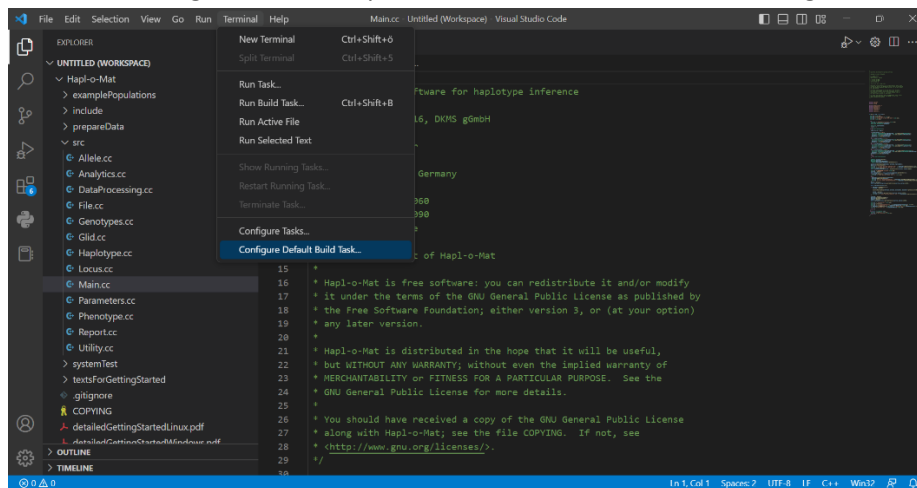
Install Visual Studio Code (<https://code.visualstudio.com/>), the C/C++ extension for VS Code and the latest version of Mingw-w64 via MSYS2 (<https://www.msys2.org/>). A detailed instruction to these installations and how to compile your code in VS Code is provided here: <https://code.visualstudio.com/docs/cpp/config-mingw>. Be sure to add the path to your Mingw-w64 bin folder to the Windows PATH environment variable according to the instructions.

Start VS Code and open the folder 'Hapl-o-Mat' as a workspace (File > Add folder to workspace).

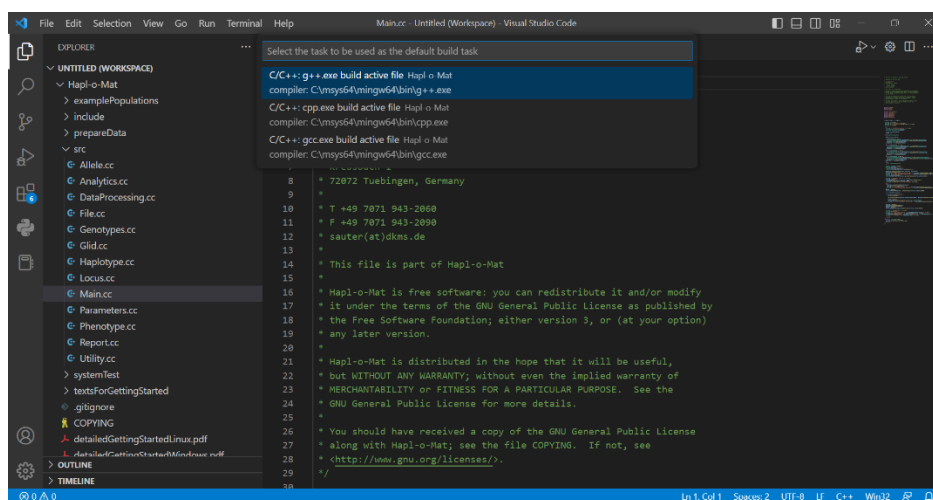




Klick on the file 'Main.cc' in the 'src' folder. The content of 'Main.cc' is shown as active file in the editor on the right side of the panel. Now choose Terminal > Configure Default Build Task.



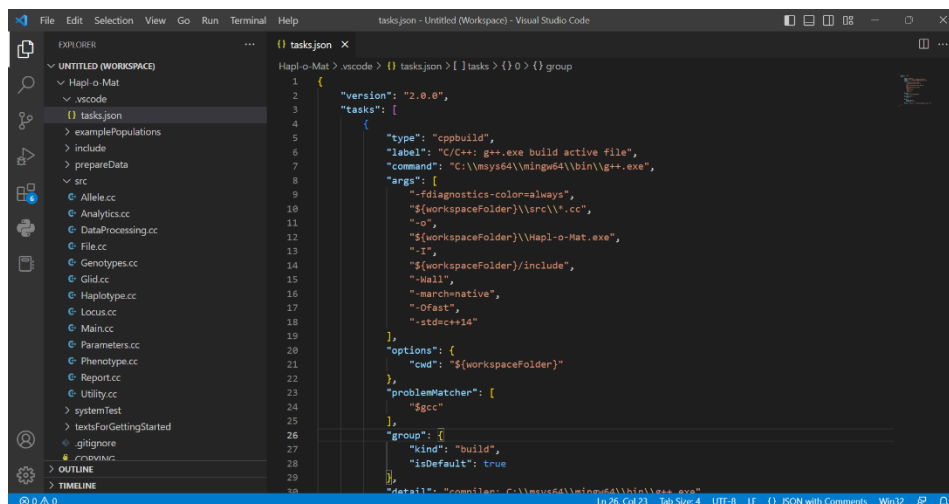
In the dropdown menu, which will list various predefined build tasks for C++ compilers, choose 'C/C++:g++.exe build active file'.



This will create a 'tasks.json' file in a '.vscode' folder in the Hapl-o-Mat folder and open it in the editor.

'tasks.json' has to be modified for Hapl-o-Mat compilation: Arguments must contain information on files to include and further individual build options. The working directory (cwd) and the destination of the resulting .exe-file must be specified. You can copy the following content and replace the content of the 'task.json' file:

```
{
  "version": "2.0.0",
  "tasks": [
    {
      "type": "cppbuild",
      "label": "C/C++: g++.exe build active file",
      "command": "C:\\msys64\\mingw64\\bin\\g++.exe",
      "args": [
        "-fdiagnostics-color=always",
        "${workspaceFolder}\\src\\*.cc",
        "-o",
        "${workspaceFolder}\\Hapl-o-Mat.exe",
        "-I",
        "${workspaceFolder}/include",
        "-Wall",
        "-march=native",
        "-Ofast",
        "-std=c++14"
      ],
      "options": {
        "cwd": "${workspaceFolder}"
      },
      "problemMatcher": [
        "$gcc"
      ],
      "group": {
        "kind": "build",
        "isDefault": true
      },
      "detail": "compiler: C:\\msys64\\mingw64\\bin\\g++.exe"
    }
  ]
}
```



Now create a `c_cpp_properties.json` file in folder `.vscode` with the following content, which defines further settings:

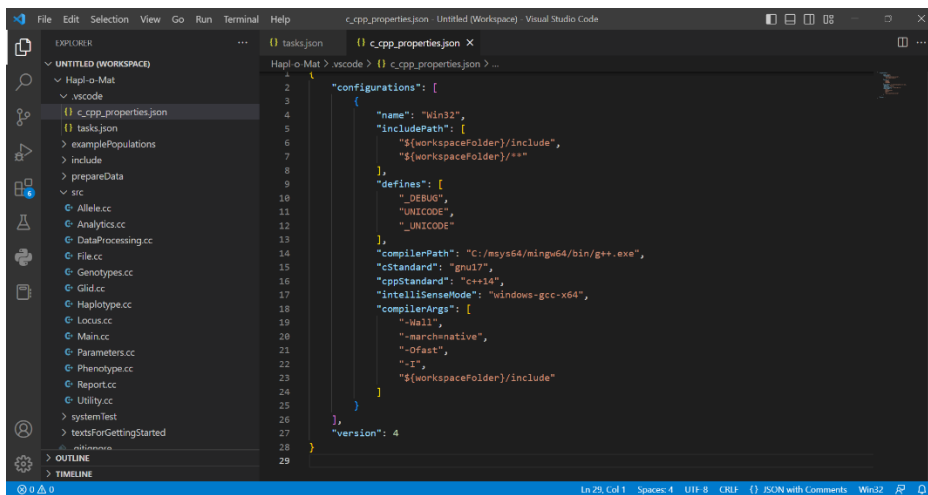
```
{
  "configurations": [
    {
      "name": "Win32",
      "includePath": [
        "${workspaceFolder}/include",

```

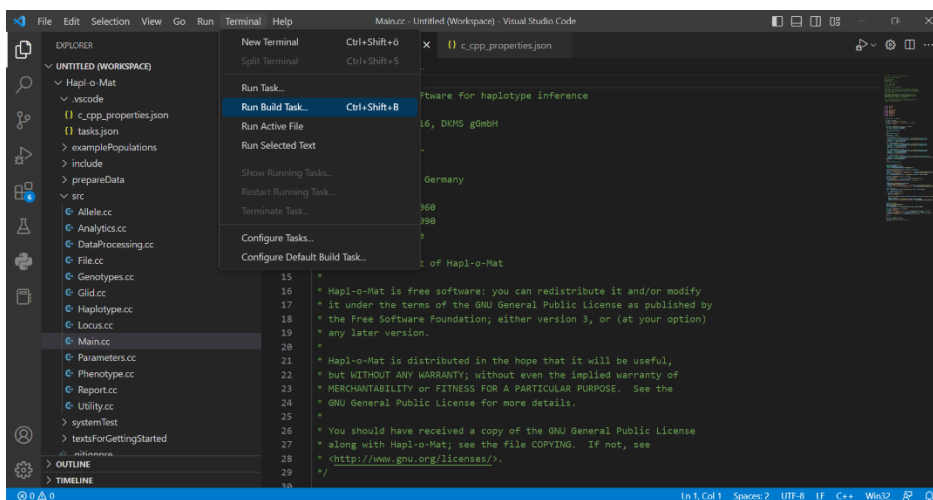
```

    "${workspaceFolder}/**"
  ],
  "defines": [
    "_DEBUG",
    "UNICODE",
    "_UNICODE"
  ],
  "compilerPath": "C:/msys64/mingw64/bin/g++.exe",
  "cStandard": "gnu17",
  "cppStandard": "c++14",
  "intelliSenseMode": "windows-gcc-x64",
  "compilerArgs": [
    "-Wall",
    "-march=native",
    "-Ofast",
    "-I",
    "${workspaceFolder}/include"
  ]
},
{
  "version": 4
}

```



Compile: Open main.cc as active window, then press Terminal > Run Build Task. A terminal window will appear under the editor and show the progress of the compilation.



The executable Hapl-o-Mat (in the described case named Hapl-o-Mat.exe) will appear in the HaploMat folder.

Congratulations, you compiled Hapl-o-Mat under Windows. But before we run Hapl-o-Mat, we have to prepare some data.

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 basis 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, open a terminal window in VS Code with Terminal > New Terminal, enter the folder “Hapl-o-Mat/prepareData” and run the python-script “BuildData.py” (e.g., “python3 BuildData.py”) to download all relevant data, process them, and move the created files to folder “Hapl-o-Mat/data” (you can easily get Python for Windows here:

<https://www.python.org/downloads/windows/>).

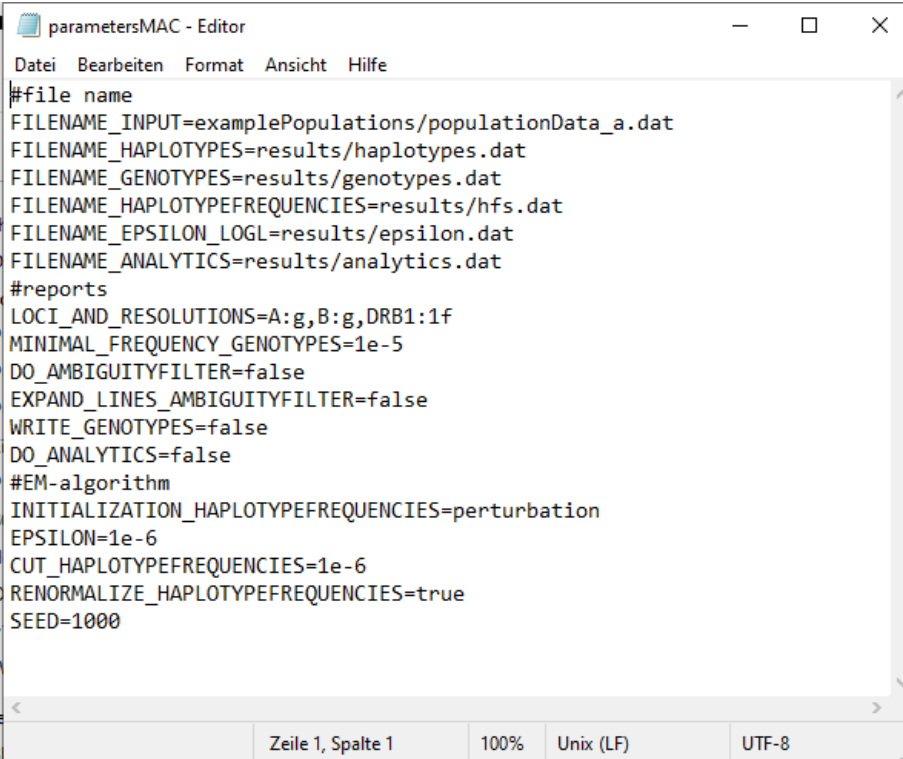
The script accesses a default set of publicly available repositories. These repositories are denoted in the file “url_config.txt”. You should be fine to work with the initial settings. However, if these repositories were to move, or you’d choose to use other sources, this can be adjusted here. For details 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. Open a terminal window in VS Code with Terminal > New Terminal and navigate to the Hapl-o-Mat folder where Hapl-o-Mat.exe is now located (This also works with Windows PowerShell or Command Prompt). Parameters for this run are saved in the default parametersMAC file:



```
parametersMAC - Editor
Datei Bearbeiten Format Ansicht Hilfe
#file name
FILENAME_INPUT=examplePopulations/populationData_a.dat
FILENAME_HAPLOTYPES=results/haplotypes.dat
FILENAME_GENOTYPES=results/genotypes.dat
FILENAME_HAPLOTYPEFREQUENCIES=results/hfs.dat
FILENAME_EPSILON_LOGL=results/epsilon.dat
FILENAME_ANALYTICS=results/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=false
DO_ANALYTICS=false
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=1000
Zeile 1, Spalte 1 100% Unix (LF) UTF-8
```

The parameter file in its default form requires a “results” folder to exist in the Hapl-o-Mat folder. Please create a “results” folder or change the destination of the results files in the parameter file.

Now type “Hapl-o-Mat.exe MAC” in the terminal window and hit return. This will execute Hapl-o-Mat with an example population (folder examplePopulations) in MAC input format. You will see the output of Hapl-o-Mat in the terminal window. Furthermore, some result files ending with “*.dat” should have appeared in your results folder, including the estimated haplotype frequencies in “hfs.dat”.

```

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Jada>cd Desktop\Hapl-o-Mat-develop
C:\Users\Jada\Desktop\Hapl-o-Mat-develop>Hapl-o-Mat.exe MAC

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

#####Initialization
MA format
#####Parameters I/O
Input: examplePopulations/populationData_a.dat
Output haplotypes: haplotypes.dat
Output genotypes: genotypes.dat
Output estimated haplotype frequencies: hfs.dat
Output epsilon and log(L): epsilon.dat
#####Parameters resolving genotypes
Minimal frequency of genotypes: 1e-005
Loci with target allele resolutions:
A : g
B : g
DRB1 : 4d
Apply ambiguity filter: no
#####Parameters EM algorithm
Haplotype frequency initialization: perturbation
Epsilon: 1e-006
Cut haplotype frequencies: 1e-006
Renormalize haplotype frequencies
Zero: 1e-014
Seed: 1000

#####Data preprocessing
Number loci: 3
Removed genotypes: 0
Leftover genotypes: 100
Type N genotypes: 02
Type A genotypes: 0
Type M genotypes: 0
Type I genotypes: 18
Different genotypes: 208
Different haplotypes: 876

Memory requirement haplotypes [MB]: 0.014064
Memory requirement genotypes [MB]: 0.028336
#####EM algorithm
Used 111 steps
Sum haplotype frequencies: 1
Sum cutted haplotype frequencies: 1
#####Times
Data preprocessing [s]: 2.174124
EM algorithm [s]: 0.021001
Writing [s]: 0.003

C:\Users\Jada\Desktop\Hapl-o-Mat-develop>

```

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	Multiple Allele Codes: 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	Genotype List Strings Column-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	Genotype List Strings: 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,

READ

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: 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: <ul style="list-style-type: none"> • "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 asltIs (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 "Hapl-o-Mat.exe" as described in section Install Haplo-mat.
- 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. Create any folders you specified in the parameter file.
- 5) 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” and create a folder named “a”. include a copy of the folder “data”, the executable “Hapl-o-Mat.exe”, the parameter file (here “parametersMAC”) to folder “a”. Additionally, create a folder named “run” in your folder “a”.

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 name
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. To this end, open a terminal window in VS Code with Terminal > New Terminal and navigate to the folder “a” where Hapl-o-Mat.exe is located (This also works with Windows PowerShell or Command Prompt).

Now type “Hapl-o-Mat.exe MAC” in the terminal window and hit return. This will execute Hapl-o-Mat with the example population (folder examplePopulations) in MAC input format. You will see the output of Hapl-o-Mat in the terminal window, including your chosen parameters, statistics on the resolved genotype data, the expectation-maximization algorithm, and the run time.



```
Microsoft Windows [Version 6.1.76011]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Jada>cd Desktop\Hapl-o-Mat-develop
C:\Users\Jada\Desktop\Hapl-o-Mat-develop>Hapl-o-Mat.exe MAC

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

#####Initialization
MA format
#####Parameters I/O
Input: examplePopulations/populationData_a.dat
Output haplotypes: haplotypes.dat
Output genotypes: genotypes.dat
Output estimated haplotype frequencies: hfs.dat
Output epsilon and log(L): epsilon.dat
#####Parameters resolving genotypes
Minimal frequency of genotypes: 1e-005
Loci with target allele resolutions:
A : 9
B : 9
DRB1 : 4d
Apply ambiguity filter: no
#####Parameters EM algorithm
Haplotype frequency initialization: perturbation
Epsilon: 1e-006
Cut haplotype frequencies: 1e-006
Renormalize haplotype frequencies
Zero: 1e-014
Seed: 1000

#####Data preprocessing
Number loci: 3
Removed genotypes: 0
Leftover genotypes: 100
Type M genotypes: 82
Type A genotypes: 0
Type M genotypes: 0
Type I genotypes: 18
Different genotypes: 208
Different haplotypes: 876

Memory requirement haplotypes [MB]: 0.014064
Memory requirement genotypes [MB]: 0.028336
#####EM algorithm
Used 111 steps
Sum haplotype frequencies: 1
Sum cutted haplotype frequencies: 1
#####Times
Data preprocessing [s]: 2.174124
EM algorithm [s]: 0.021001
Writing [s]: 0.003

C:\Users\Jada\Desktop\Hapl-o-Mat-develop>
```

Results

Now let's examine the results produced by Hapl-o-Mat. The results are saved in the folder ‘run’ (press F5 to update, if they do not appear). We first look into the file with the resolved genotypes, “run/genotypes.dat”.

```

1 1 NNN 1 A*26:04+A*29:67^B*07:218+B*54:16^DRB1*11+DRB1*13
2 2 III 0.2962962962963 A*02:570+A*24:02g^B*13:07N+B*14:01g^DRB1*12+DRB1*14
3 2 III 0.037037037037037 A*02:570+A*24:50^B*13:07N+B*14:01g^DRB1*12+DRB1*14
4 2 III 0.2962962962963 A*02:570+A*24:02g^B*13:07N+B*14:14^DRB1*12+DRB1*14
5 2 III 0.037037037037037 A*02:570+A*24:50^B*13:07N+B*14:14^DRB1*12+DRB1*14
6 2 III 0.2962962962963 A*02:570+A*24:02g^B*13:07N+B*14:19^DRB1*12+DRB1*14
7 2 III 0.037037037037037 A*02:570+A*24:50^B*13:07N+B*14:19^DRB1*12+DRB1*14
8 3 NNN 1 A*02:570+A*29:67^B*13:07N+B*35:54^DRB1*13+DRB1*14
9 4 NIN 0.333333333333333 A*02:77+A*66:10^B*14:01g+B*15:154^DRB1*16+DRB1*16
10 4 NIN 0.333333333333333 A*02:77+A*66:10^B*14:14+B*15:154^DRB1*16+DRB1*16
11 4 NIN 0.333333333333333 A*02:77+A*66:10^B*14:19+B*15:154^DRB1*16+DRB1*16
12 5 NNN 1 A*03:217+A*29:36^B*15:01g+B*35:54^DRB1*11+DRB1*14
13 6 NNN 1 A*02:454+A*03:93^B*15:316+B*35:54^DRB1*04+DRB1*08
14 7 NNN 1 A*24:02g+A*30:13^B*18:19+B*39:27^DRB1*13+DRB1*14
15 8 NNI 1 A*03:93+A*30:13^B*18:12+B*18:50^DRB1*11+DRB1*15
16 9 NNN 1 A*02:259+A*68:94N^B*15:154+B*57:21^DRB1*13+DRB1*14
17 10 NNN 1 A*02:454+A*32:73^B*15:316+B*51:114^DRB1*07+DRB1*08
18 11 NNN 1 A*24:05g+A*32:33^B*18:50+B*54:16^DRB1*11+DRB1*11
19 12 NNN 0.983333333333333 A*02:163+A*02:454^B*13:07N+B*42:14^DRB1*03+DRB1*11
20 12 NNN 0.016666666666667 A*02:163+A*02:454^B*13:07N+B*42:14^DRB1*03N+DRB1*11
21 13 NNN 1 A*11:01g+A*24:02g^B*15:27+B*40:01g^DRB1*07+DRB1*14
22 14 NNN 1 A*03:239+A*11:01g^B*27:98+B*40:01g^DRB1*03+DRB1*07
23 15 NNN 1 A*02:259+A*03:02g^B*41:15+B*57:21^DRB1*11+DRB1*14
24 16 NNN 1 A*02:570+A*30:13^B*13:07N+B*18:12^DRB1*11+DRB1*14
25 17 NNN 1 A*11:109N+A*29:36^B*15:154+B*55:40^DRB1*04+DRB1*13
26 18 NNN 1 A*23:06+A*33:27^B*35:54+B*52:49N^DRB1*07+DRB1*14N
27 19 NNN 1 A*02:163+A*66:10^B*37:28+B*51:01g^DRB1*04+DRB1*16
28 20 NNN 1 A*23:03+A*29:25^B*15:101+B*39:27^DRB1*13+DRB1*16
29 21 NNN 1 A*02:454+A*29:25^B*13:07N+B*15:101^DRB1*11+DRB1*16
30 22 NNN 1 A*11:01g+A*66:10^B*15:154+B*15:316^DRB1*11+DRB1*16
31 23 NNN 0.983333333333333 A*29:01g+A*32:33^B*52:42+B*54:16^DRB1*03+DRB1*11

```

Normal text file length: 9423 lines: 145 Ln: 1 Col: 1 Pos: 1 Unix (LF) UTF-8 INS

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.

```

1 1.69903015748793e-02 -8.24277395321455e+02
2 6.42669480453906e-03 -7.77735808640654e+02
3 2.66852001127576e-03 -7.72008405634169e+02
4 1.08181824968119e-03 -7.71304126872460e+02
5 4.49439953600574e-04 -7.70942598793904e+02
6 9.44682627712547e-04 -7.69937250343107e+02
7 1.34635642305227e-03 -7.67977031411123e+02
8 1.31391738723141e-03 -7.66208310497808e+02
9 9.74201115758931e-04 -7.65286736455404e+02
10 5.56283662206575e-04 -7.64995271440771e+02
11 2.22213417517972e-04 -7.64939382226830e+02
12 1.62841391961374e-04 -7.64925571323633e+02
13 1.12097202145062e-04 -7.64916576944365e+02
14 7.36281403432208e-05 -7.64910670646466e+02
15 4.68016436203613e-05 -7.64906918888450e+02
16 2.91105189376032e-05 -7.64904586830570e+02
17 1.78572545549132e-05 -7.64903156979828e+02
18 1.08597765420565e-05 -7.64902287714532e+02
19 6.56927174751741e-06 -7.64901761993058e+02
20 3.96102226817474e-06 -7.64901445045292e+02
21 2.38366978359663e-06 -7.64901254327676e+02
22 1.43275331939745e-06 -7.64901139698697e+02
23 8.60572931891340e-07 -7.64901070849712e+02
24

```

Normal text file length: 989 lines: 24 Ln: 1 Col: 1 Pos: 1 Unix (LF) UTF-8 INS

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).

1	A*29:25~B*15:101~DRB1*16	0.0600000000000000
2	A*11:01g~B*40:01g~DRB1*07	0.0400000000000000
3	A*11:01g~B*15:316~DRB1*11	0.0300000000000000
4	A*24:02g~B*15:27~DRB1*14	0.0300000000000000
5	A*68:70~B*37:19~DRB1*13	0.0300000000000000
6	A*32:33~B*54:16~DRB1*11	0.0250000000000000
7	A*30:13~B*18:12~DRB1*11	0.0250000000000000
8	A*03:02g~B*35:32~DRB1*09	0.0250000000000000
9	A*30:73N~B*14:08~DRB1*14	0.02499870784292
10	A*26:04~B*54:16~DRB1*11	0.0200000000000000
11	A*02:570~B*13:07N~DRB1*14	0.0200000000000000
12	A*03:217~B*15:01g~DRB1*11	0.0200000000000000
13	A*02:454~B*15:316~DRB1*08	0.0200000000000000
14	A*03:93~B*35:54~DRB1*04	0.0200000000000000
15	A*03:217~B*35:51~DRB1*14	0.0200000000000000
16	A*02:77~B*56:19N~DRB1*14	0.01500129215708
17	A*66:10~B*15:154~DRB1*16	0.0150000000000000
18	A*02:163~B*37:28~DRB1*16	0.0150000000000000
19	A*23:06~B*08:145~DRB1*16	0.0150000000000000
20	A*32:73~B*51:114~DRB1*07	0.0150000000000000
21	A*03:239~B*27:98~DRB1*03	0.0150000000000000
22	A*23:01g~B*15:01g~DRB1*13	0.0150000000000000
23	A*02:258~B*42:14~DRB1*13	0.0150000000000000
24	A*11:109N~B*55:40~DRB1*13	0.0150000000000000
25	A*29:36~B*15:154~DRB1*04	0.0150000000000000
26	A*68:94N~B*39:27~DRB1*03	0.0150000000000000
27	A*68:70~B*35:32~DRB1*13	0.0150000000000000
28	A*29:67~B*35:54~DRB1*13	0.0150000000000000
29	A*66:10~B*51:01g~DRB1*04	0.0150000000000000
30	A*03:217~B*52:49N~DRB1*08	0.0150000000000000
31	A*24:02g~B*51:114~DRB1*03	0.0147500000000000

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

```
Hapl-o-Mat.exe GLSC
```

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 data files for the population data are “populationData_c.pull” and “populationData_c.glid”.

Open “parametersGLS” in a text editor and adapt the parameters. 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”. Now, your parameter file should look similar to that:

```
#file names
FILENAME_PULL=../populationData_c.pull
FILENAME_GLID=../populationData_c.glid
FILENAME_HAPLOTYPES=run/haplotypes.dat
FILENAME_GENOTYPES= run/genotypes.dat
FILENAME_HAPLOTYPEFREQUENCIES= run/estimatedHaplotypeFrequencies.dat
```

```
FILENAME_EPSILON_LOGL= run/epsilon.dat
FILENAME_ANALYTICS= run/analytcs.dat
#reports
LOCIORDER=B,A,DPB1,DRB1,C,DQB1
LOCI_AND_RESOLUTIONS=A:g,B:g,DRB1:g
MINIMAL_FREQUENCY_GENOTYPES=1e-5
DO_AMBIGUITYFILTER=false
EXPAND_LINES_AMBIGUITYFILTER=false
RESOLVE_MISSING_GENOTYPES=true
WRITE_GENOTYPES=true
DO_ANALYTICS=false
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=1000
```

Open a terminal window in VS Code with Terminal > New Terminal and navigate to folder “c” where Hapl-o-Mat.exe is located (This also works with Windows PowerShell or Command Prompt.), and run Hapl-o-Mat via

```
Hapl-o-Mat.exe GLS
```

Afterwards, the result files should appear in your folder.

Input Format READ

Finally, we test the input format READ. Create a folder “d” and copy one file with resolved genotypes (e.g. from “a/run/genotypes.dat”) there. Add “haplomat” and “parametersREAD” and an empty folder “run” 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
```