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:

Journal article to come

C. Schäfer, A.H. Schmidt, J. Sauter: Hapl-O-mat: A Versatile Software for Haplotype Frequency Estimation. HLA (2016), 87, 236-320

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

General remark

If you intend to use Hapl-o-Mat on Windows operating systems, we recommend to download the Hapl-o-Mat Windows binary version (https://github.com/DKMS/Hapl-o-Mat_WinBin). This version contains both, Hapl-o-Mat and the GUI and does not require compiling.

To use Hapl-o-Mat from its C++ source on Windows, Hapl-o-Mat needs to be compiled. This process is described below.

Getting Started

This guide is an introduction on how to use Hapl-o-Mat from C++ source code version on Windows. In order to follow this guide, you need a Windows system, a C++ compiler supporting C++14, and Python (at least version 2.7.11). In this tutorial, we use Eclipse IDE for C/C++ Developers. You can get it here https://www.eclipse.org/downloads/. 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.

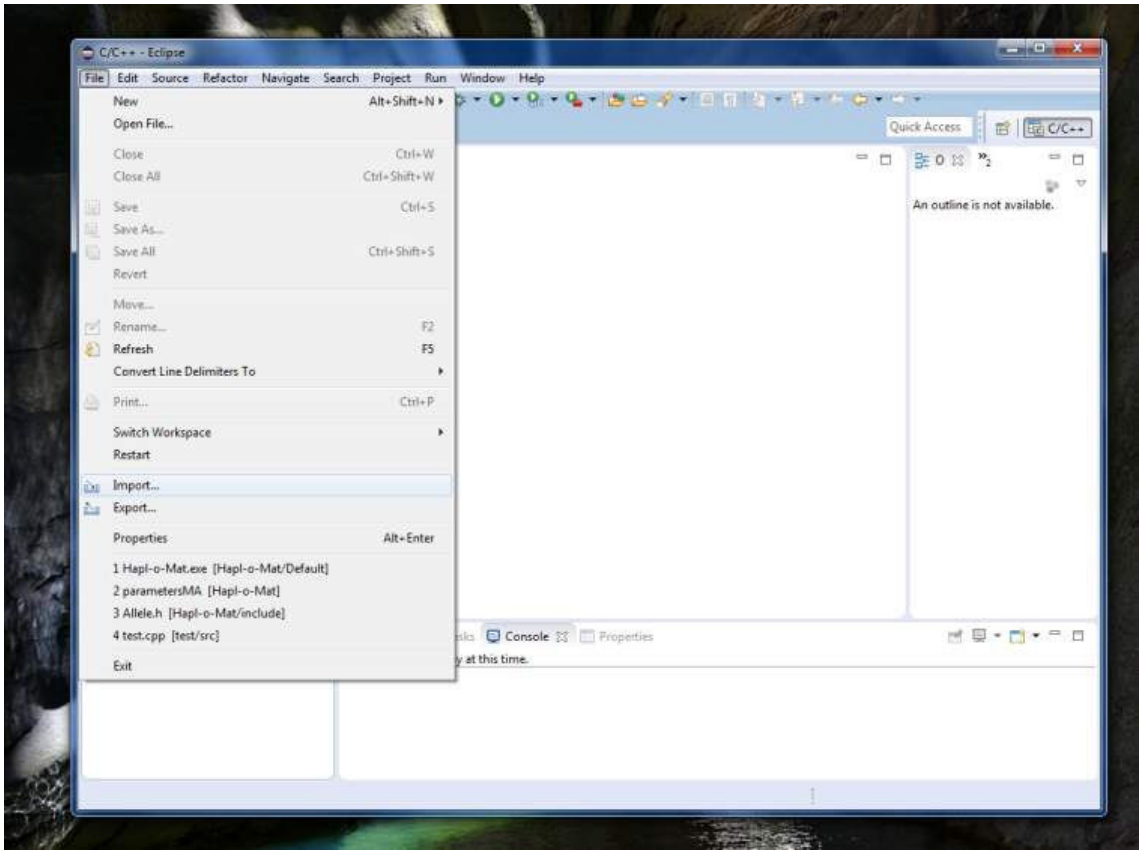
File name	**Description**
ChangePreamble.py	A python-script to adapt the preamble in all files; You are not going to use it.
COPYING	The GNU General Public License
detailedGettingStartedLinux	Guide for using Hapl-o-Mat under Linux
detailedGettingStartedWindows	Guide for using Hapl-o-Mat under Windows

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

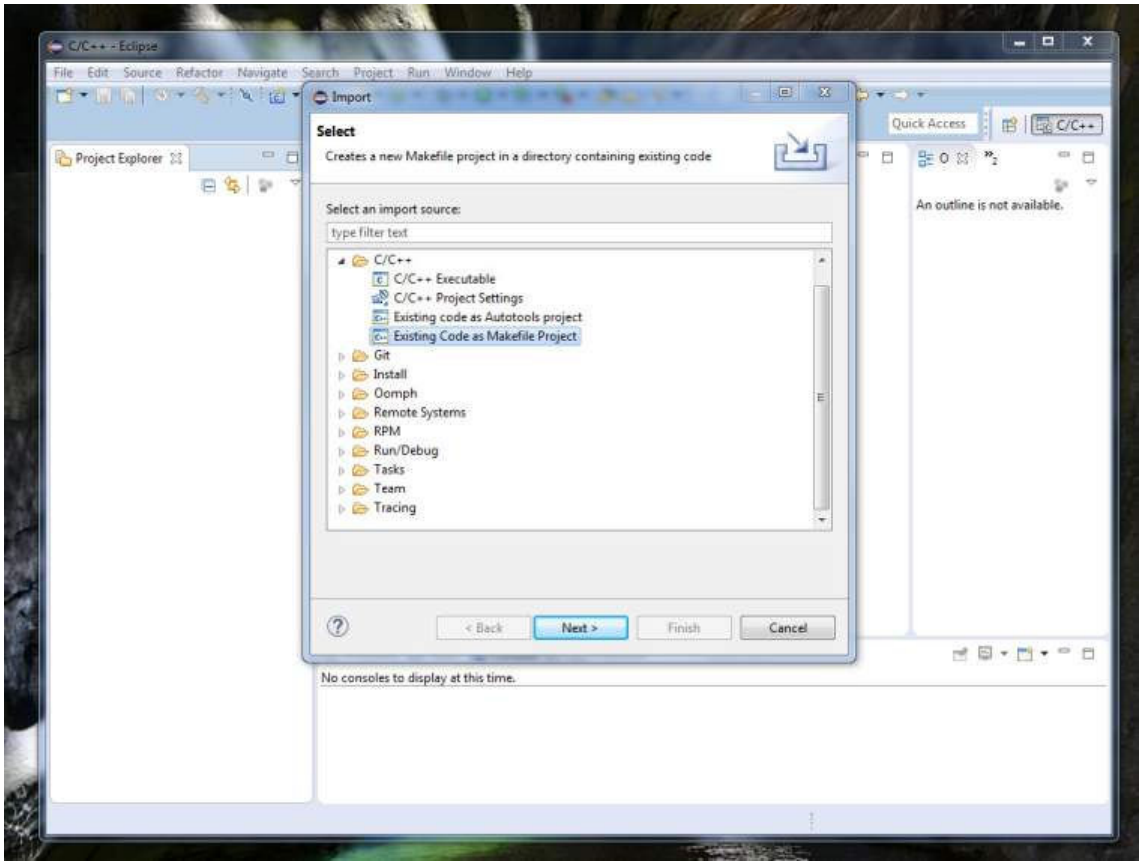
To estimate haplotype frequencies we only need to consider the folder prepareData and the files parametersGLS, parametersGLSC, parametersMAC, and parametersREAD. To finish this tutorial we need the folder examplePopluations, too.

Install Hapl-o-Mat

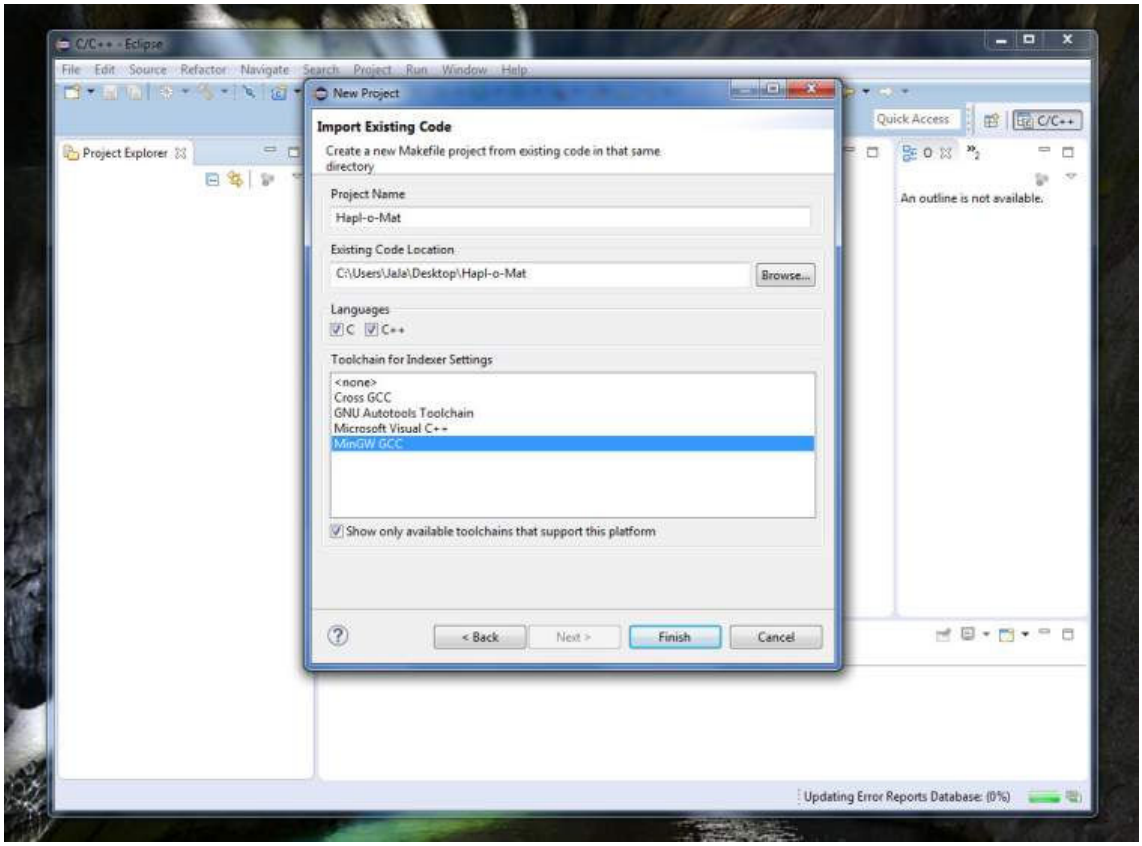
We compile Hapl-o-Mat using Eclipse. Start Eclipse and create a project by clicking on File -> Import.



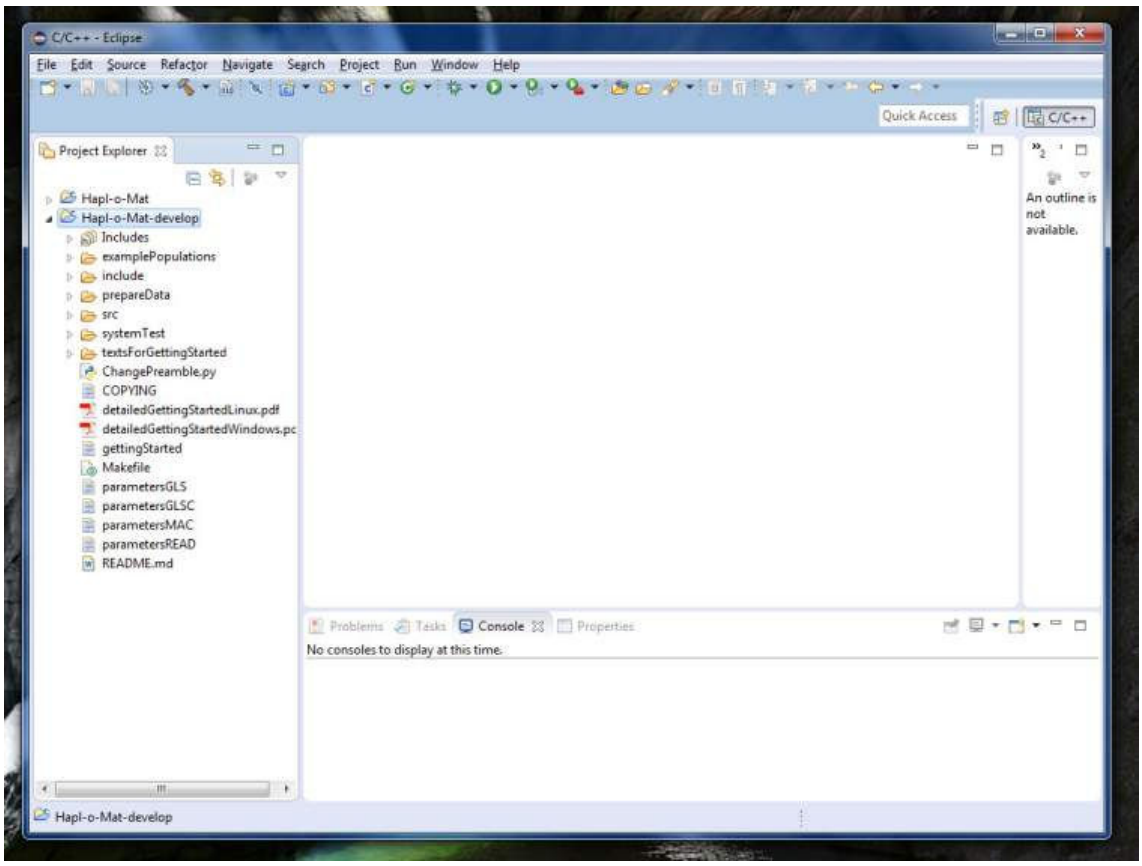
Choose Existing Code as Makefile Project in the pop-up window and then click Next.



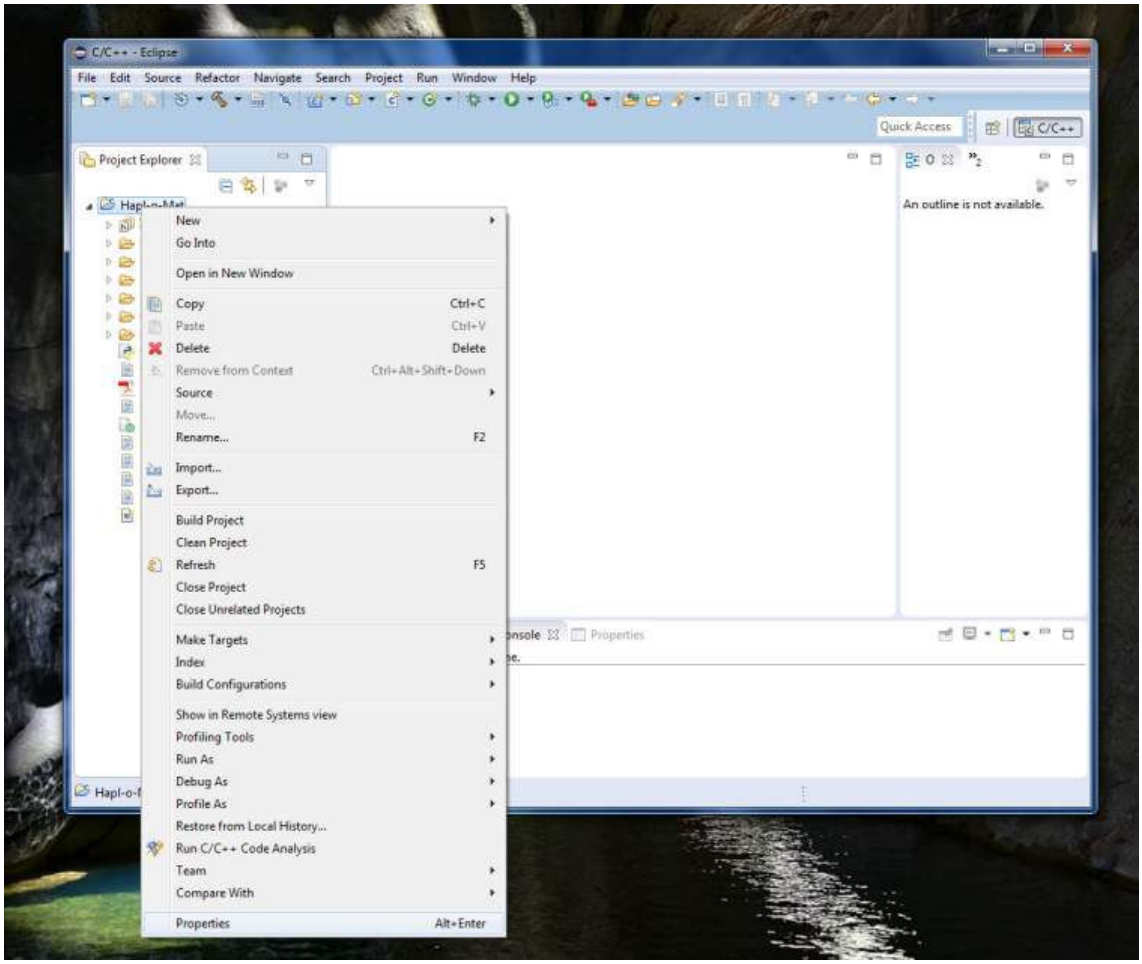
Enter the Project Name, e.g. Hapl-o-Mat, and browse to the location where you saved Hapl-o-Mat. Select MinGW GCC as Toolchain. Click Finish.



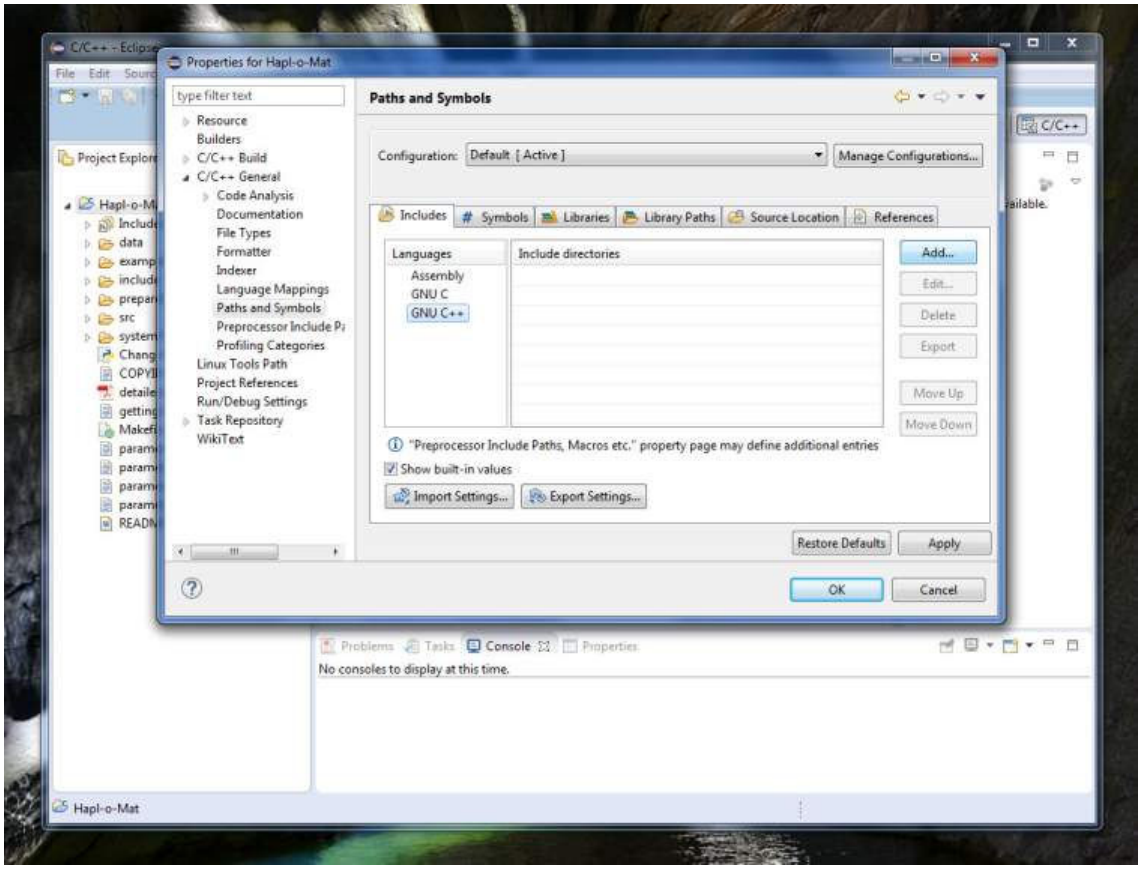
Now, you find all files and folders of Hapl-o-Mat in the Project Explorer. If your Eclipse does not show the Project Explorer, activate it via Window -> Show View -> Project Explorer.



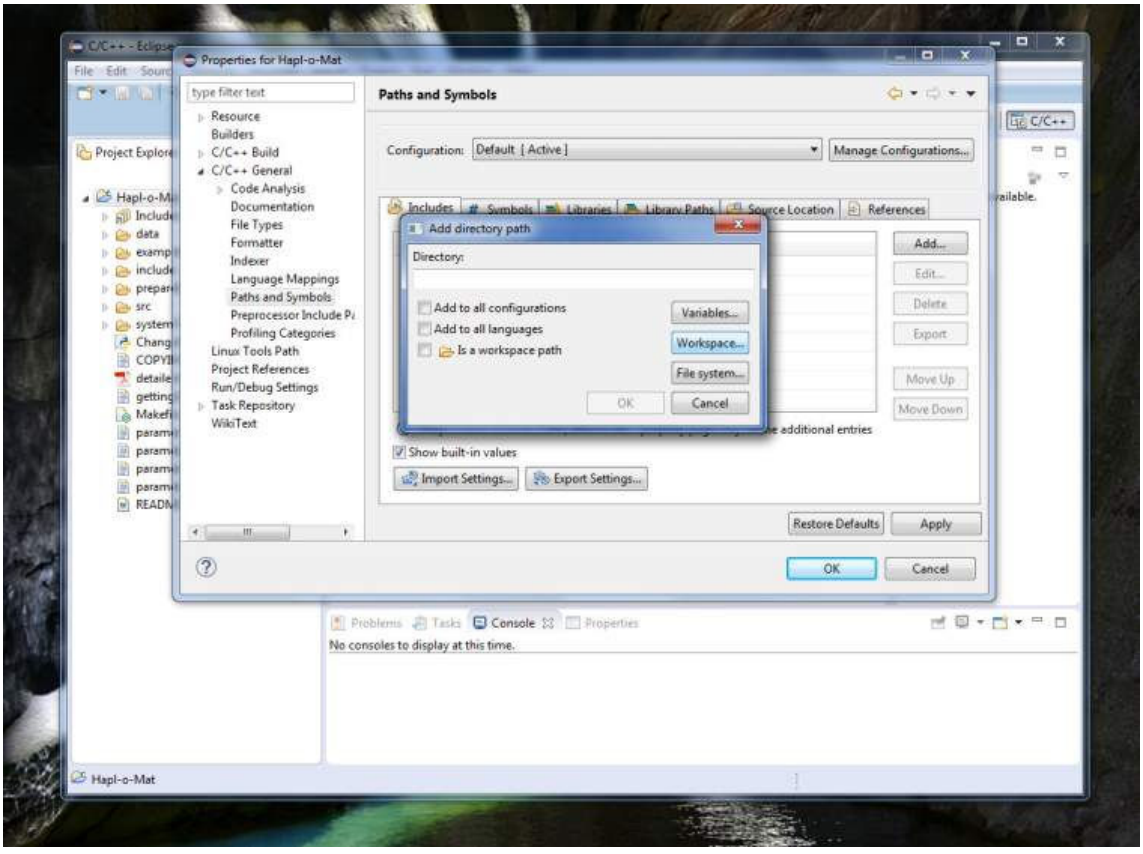
Next, we have to adapt some settings. Right-click into the Project Explorer and choose Properties.



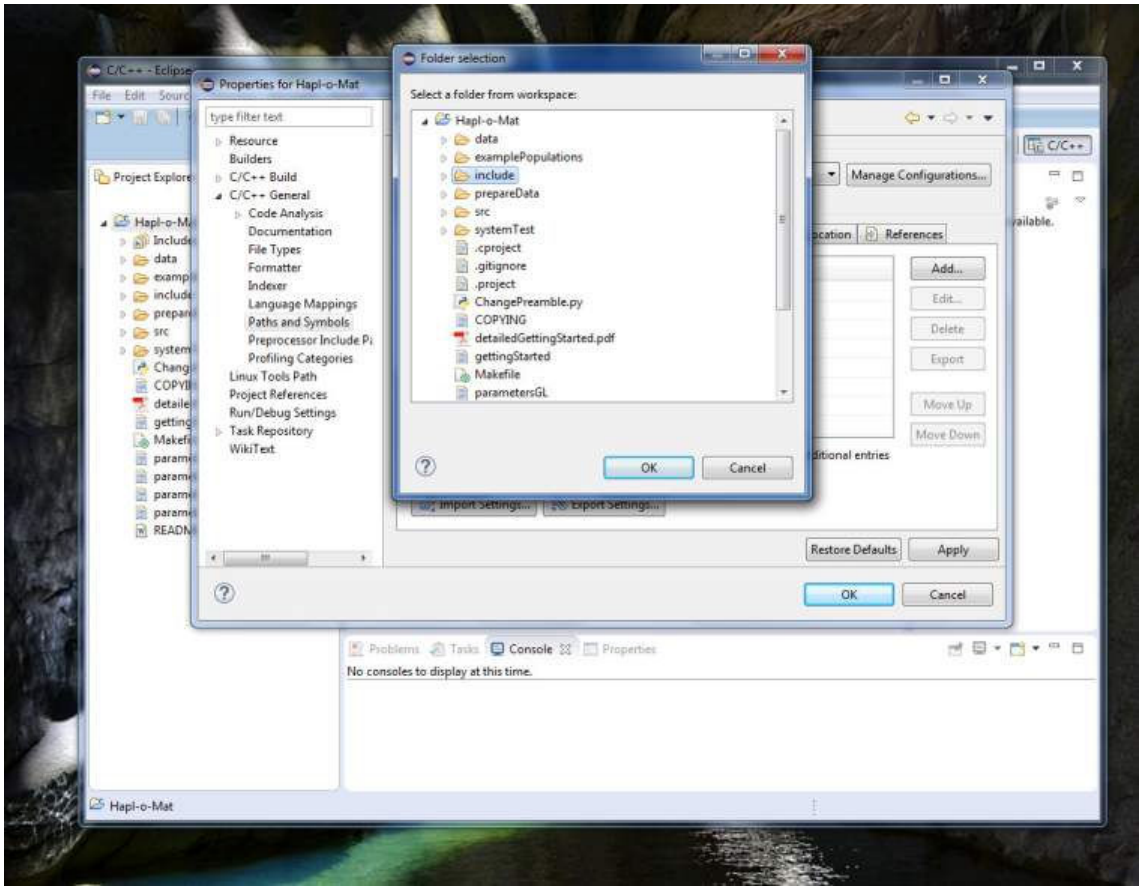
A new window pops up. In the left column choose C/C++ General -> Paths and Symbols. Select GNU C++ in the right column and then click Add.



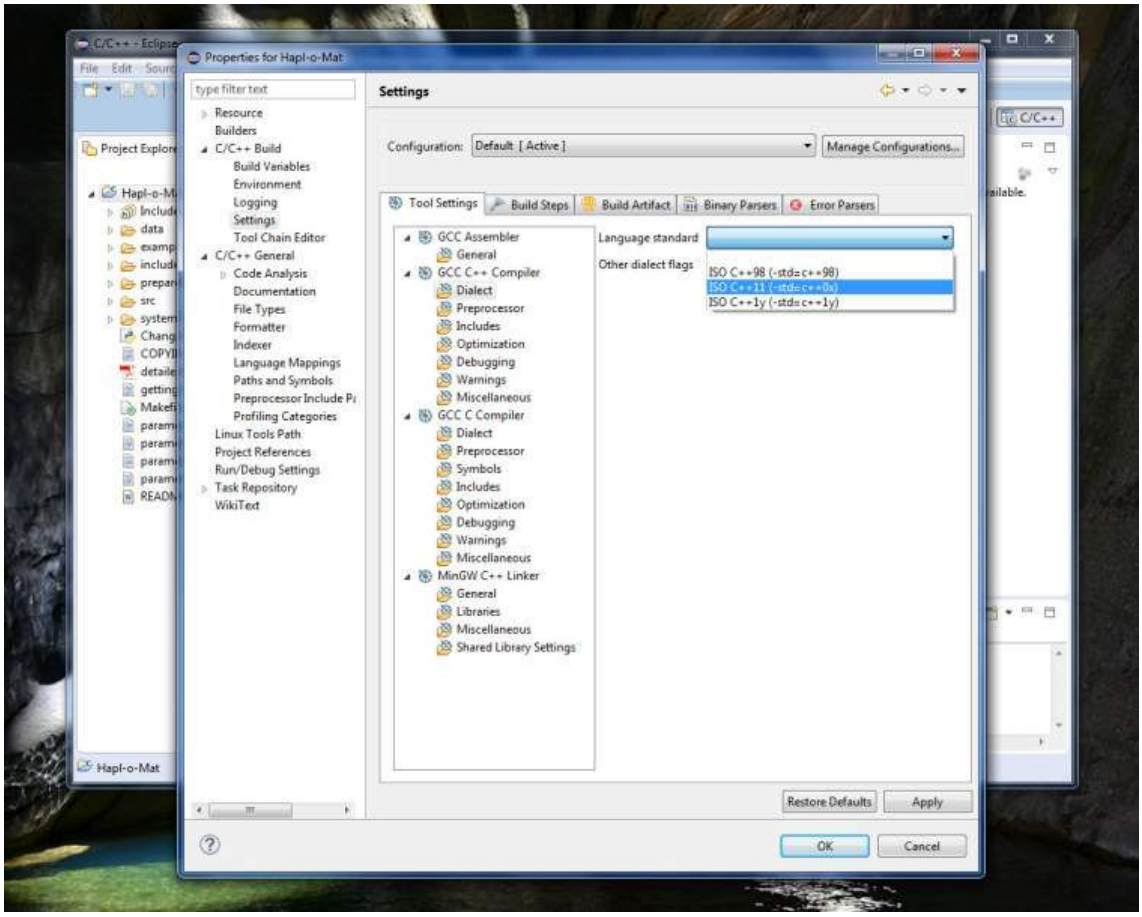
A new window pops up, where you click on Workspace….



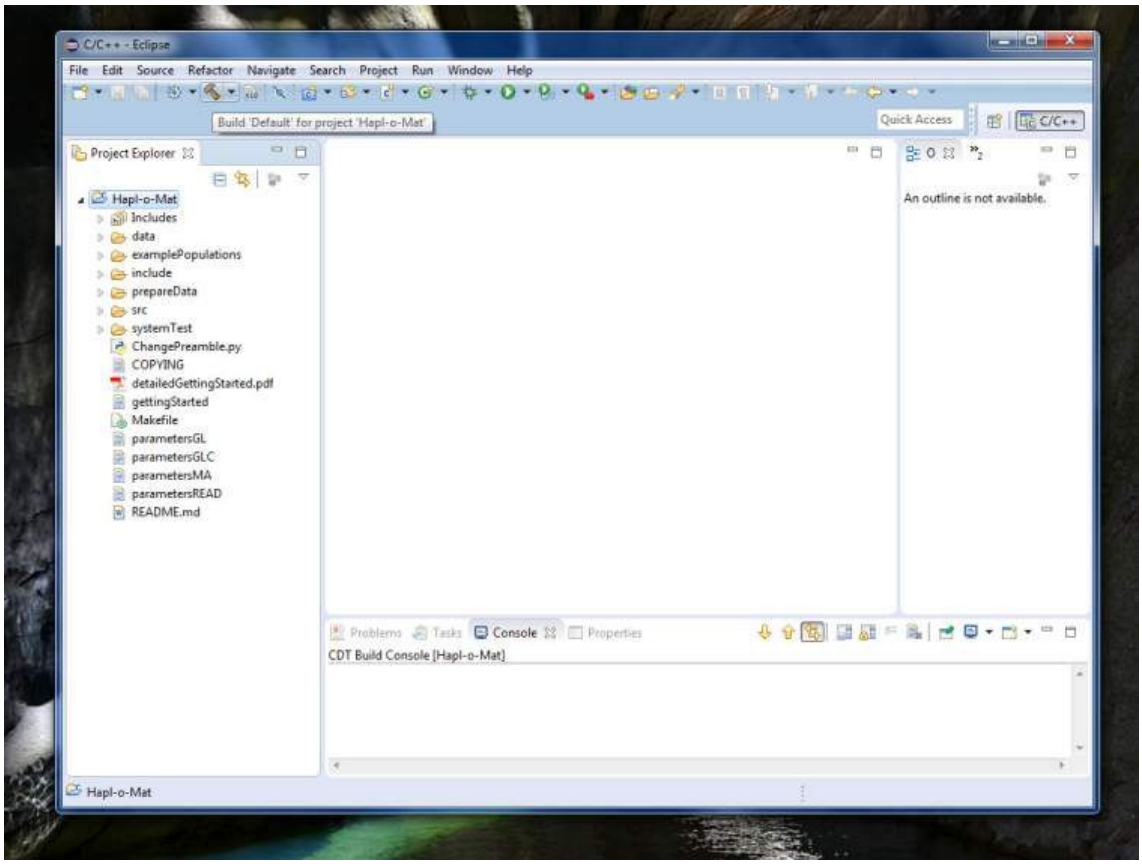
In the following window choose folder include.



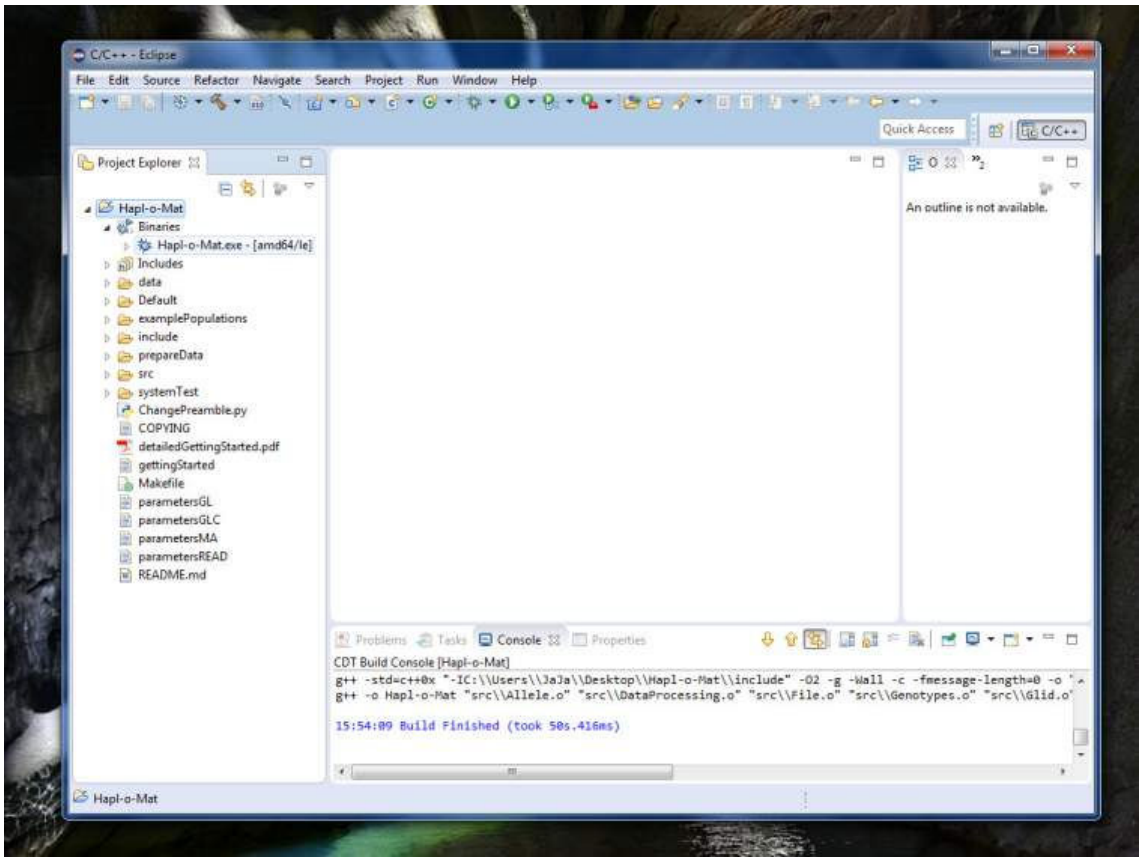
Click OK and then Apply. Next, choose in the left column C/C++ Build -> Settings. In the right column choose GCC C++ Compiler -> Dialect and then choose ISO C++14(-std=c++0x) as Language standard. Click OK and then Apply.



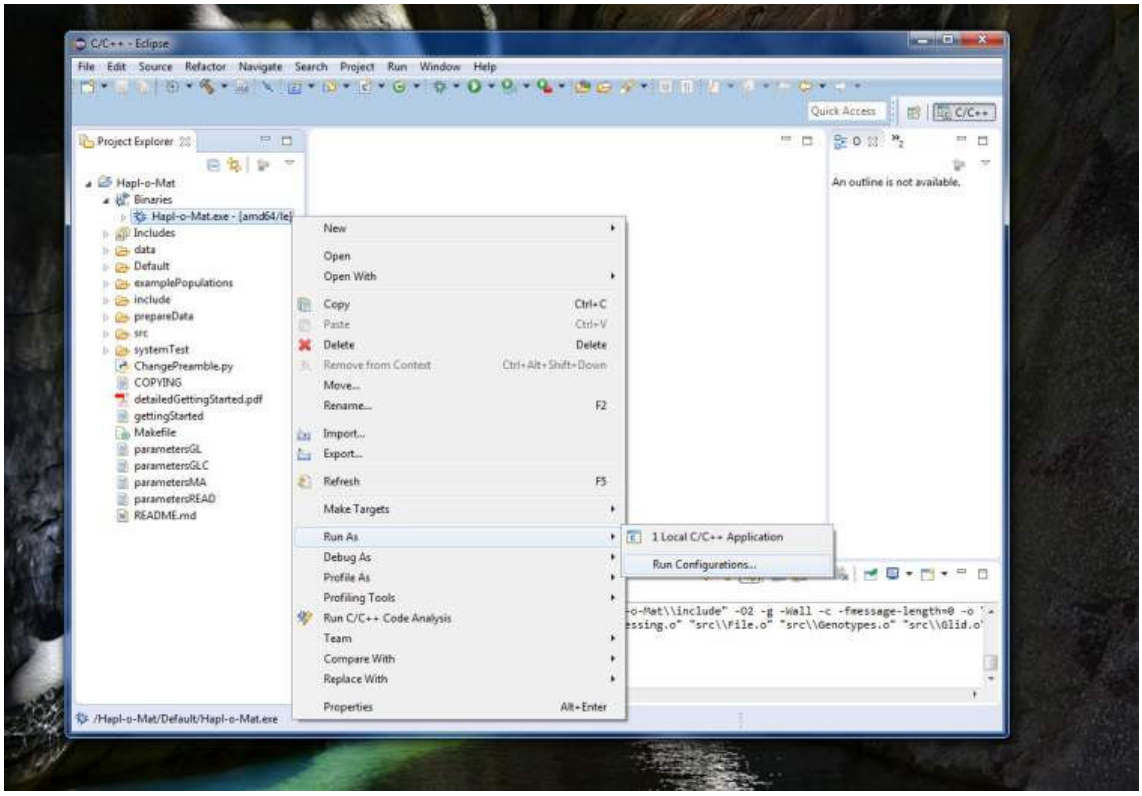
Finally, we compile Hapl-o-Mat by clicking the hammer-symbol in the upper navigation bar.



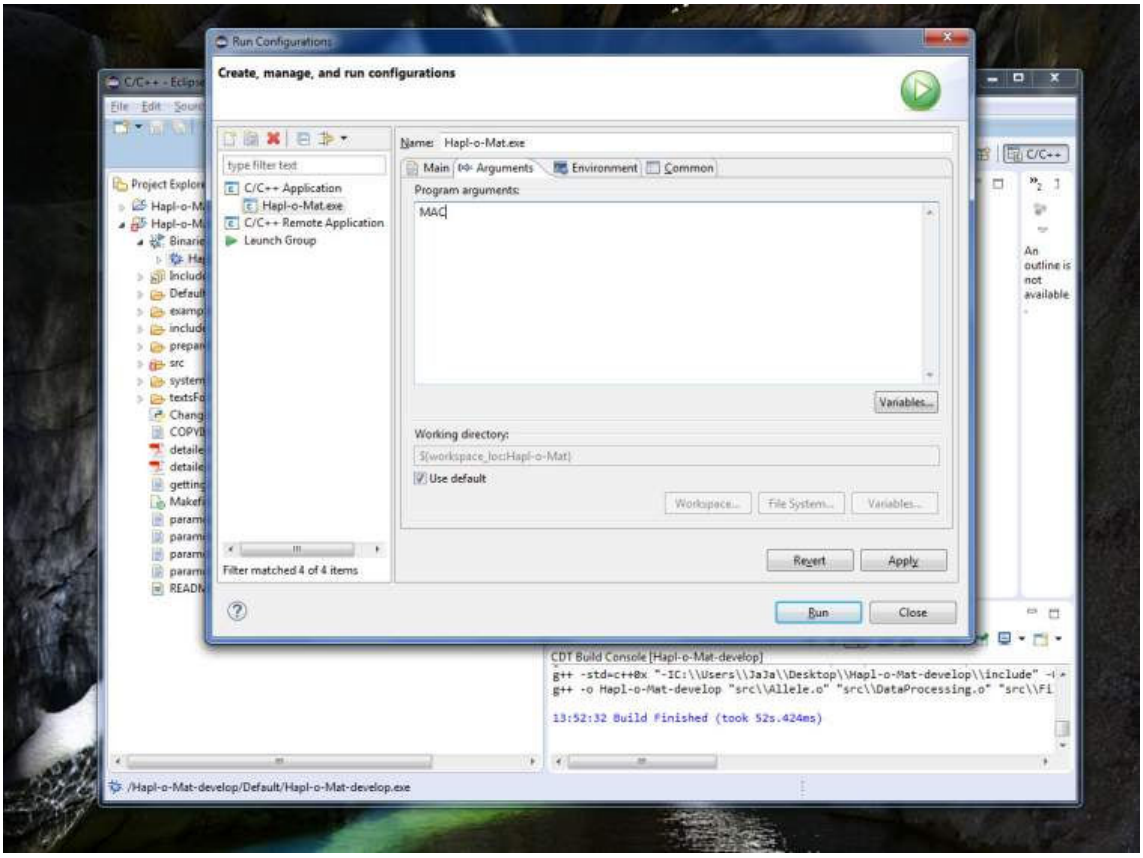
The executable Hapl-o-Mat.exe appeared under Binaries.



Right-click on it and choose Run As -> Run Configurations….



Click on the tab Arguments and enter MAC into the field Program arguments.



Then click Close. Congratulations, you compiled Hapl-o-Mat under Windows. 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 update data from time to time. 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, it can be replaced by combining all alleles of the same locus from this file. You only must create it in this case.
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/detailedExplanationManuallyPrepareData.pdf”.

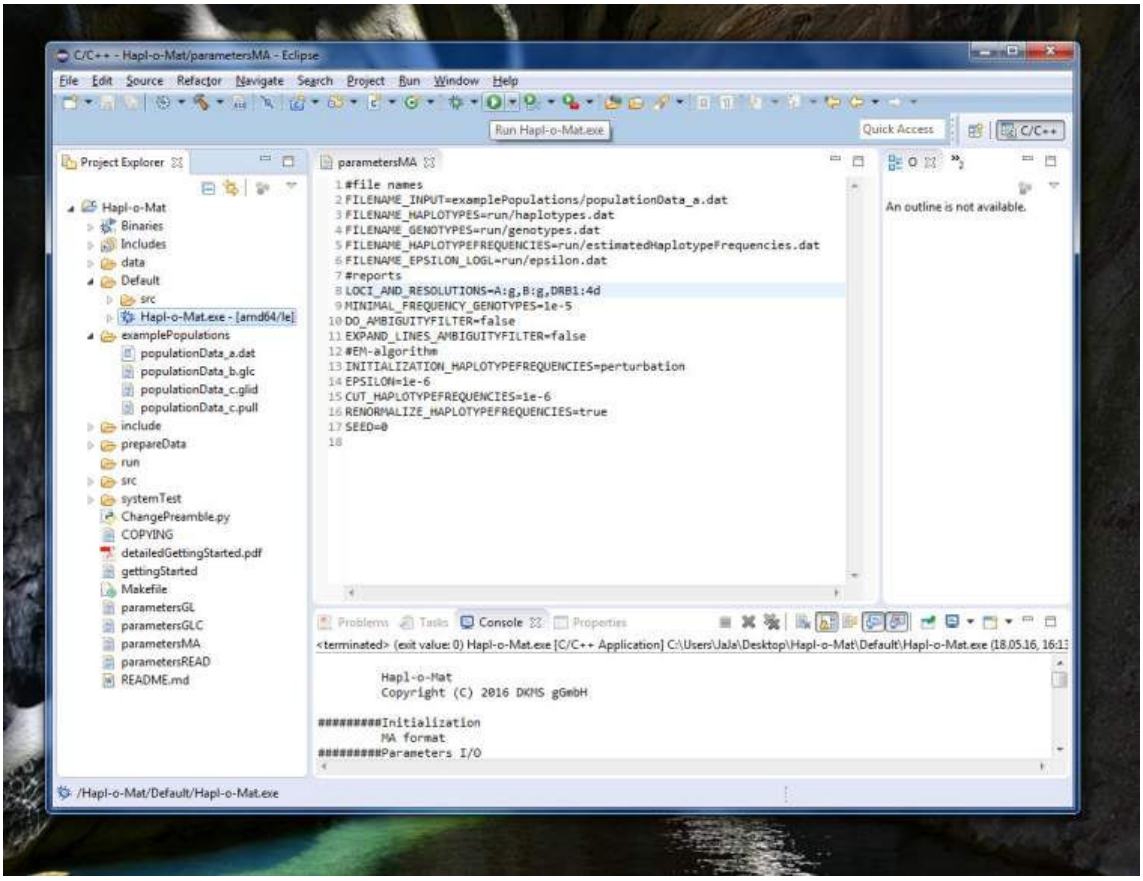
To build data automatically, enter the folder “Hapl-o-Mat/prepareData” und run the python-script “BuildData.py” to download all relevant data, process them, and move the created files to folder “Hapl-o-Mat/data” (you can easily get Python for Windos here: https://www.python.org/downloads/windows/).

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.doc” for a detailed description; refer there to the section “Semi-Automated Way”.

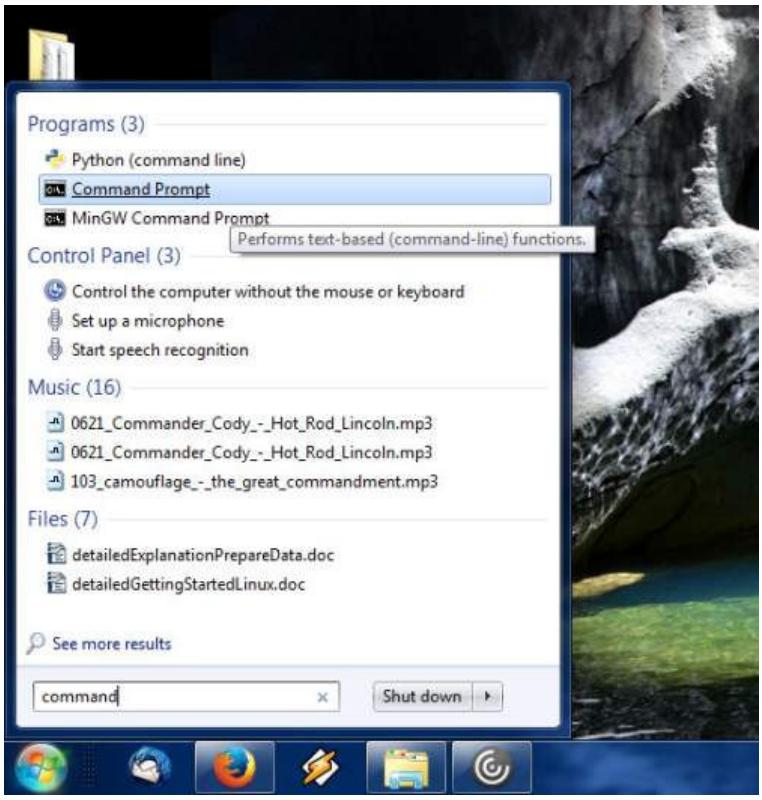
If you want to use Hapl-o-MatGUI in the source code version (https://github.com/DKMS/Hapl-o-Mat_GUI) with Hapl-o-Mat on Windows, you must first create a binary file from the prepareData package. To do this, enter the folder "Hapl-o-Mat/prepareData" and run "pyinstaller BuildData.py --onefile". A new folder "dist" will be created, which contains the executable file "BuildData.exe". Please move "BuildData.exe" to the "Hapl-o-Mat/prepareData". From here it will be triggered by the GUI. However, we recommend that you download the Hapl-o-Mat Windows binary (https://github.com/DKMS/Hapl-o-Mat_WinBin) if you intend to use Hapl-o-Mat with Hapl-o-MatGUI on Windows.

Run Hapl-o-Mat

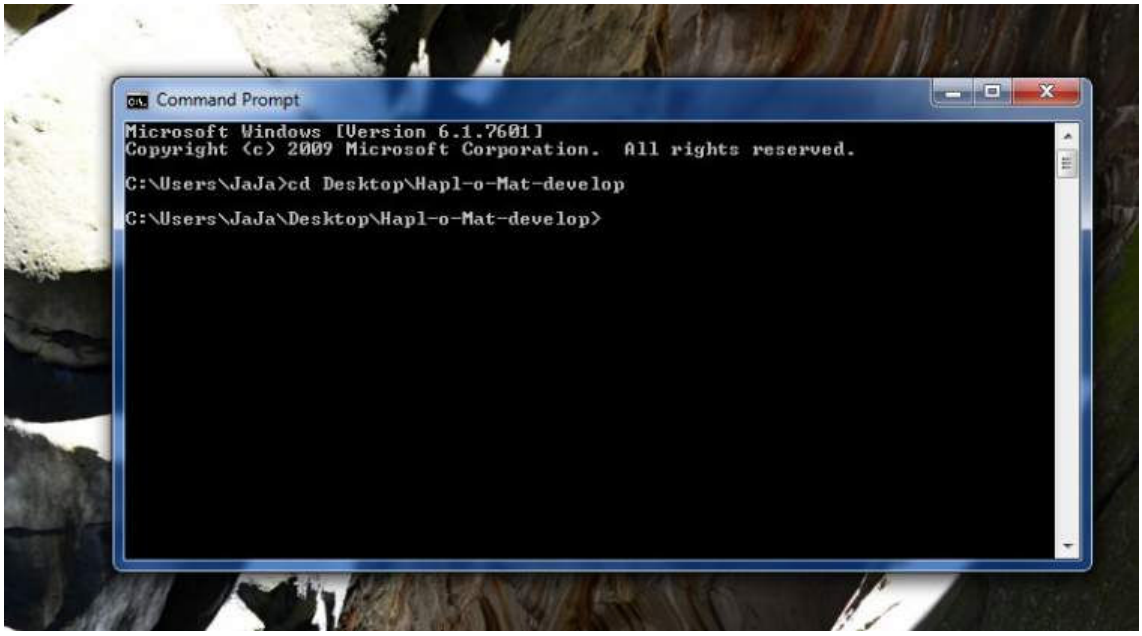
Now, you are able to run Hapl-o-Mat for the first time. Click on the green Play button in the upper navigation bar. You see the output of Hapl-o-Mat in the window at the bottom. Furthermore, some result-files ending with “*.dat” should have appeared in your folder, including the estimated haplotype frequencies in “hfs.dat”.



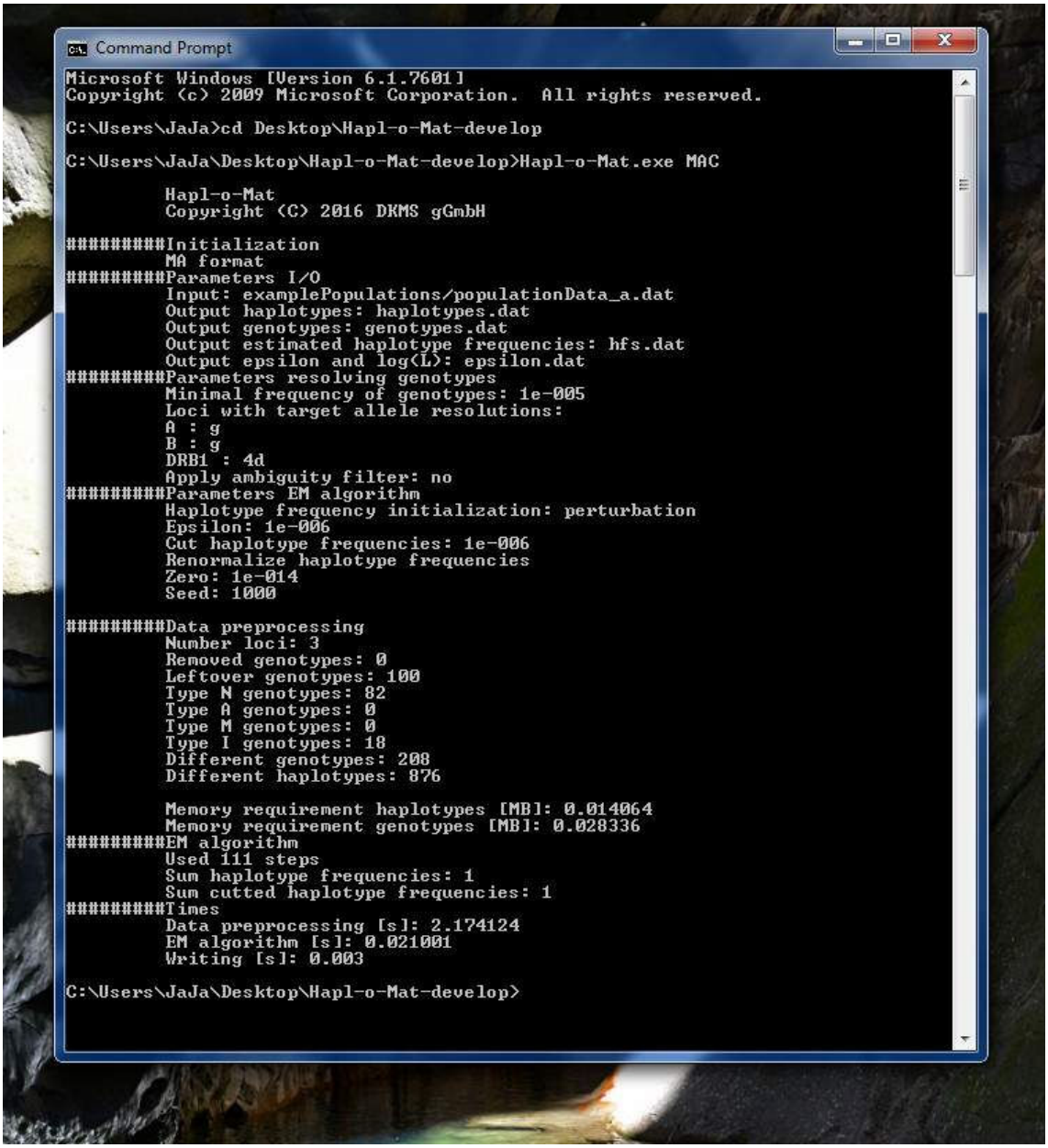
Alternatively, you can run Hapl-o-Mat from the command prompt. Eclipse saved an executable of Hapl-o-Mat in the folder “Default”. Copy it to the folder Hapl-o-Mat. Then click on the Windows symbol and type Command Prompt to open the command prompt.



Next, navigate to the folder, where you saved Hapl-o-Mat (we saved it at “Desktop\Hapl-o-Mat-develop”)



and type “Hapl-o-Mat.exe MAC” to run Hapl-o-Mat.



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.

When compiling Hapl-o-Mat we assigned MAC as argument to the executable. If you want to run another format from Eclipse, you have to change the argument to the appropriate format abbreviation. Right-click in the Project Explorer and select Run As -> Run Configurations…. Then, click on the tab Arguments and change MAC to the corresponding abbreviation from above. If you start Hapl-o-Mat from the command prompt, just change the command MAC to the appropriate format.

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), 2-digit fields (2d), 4-digit fields (4d), 6-digit fields (6d), and 8-digit fields (8d). 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” as described in section Install Haplomat.
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.

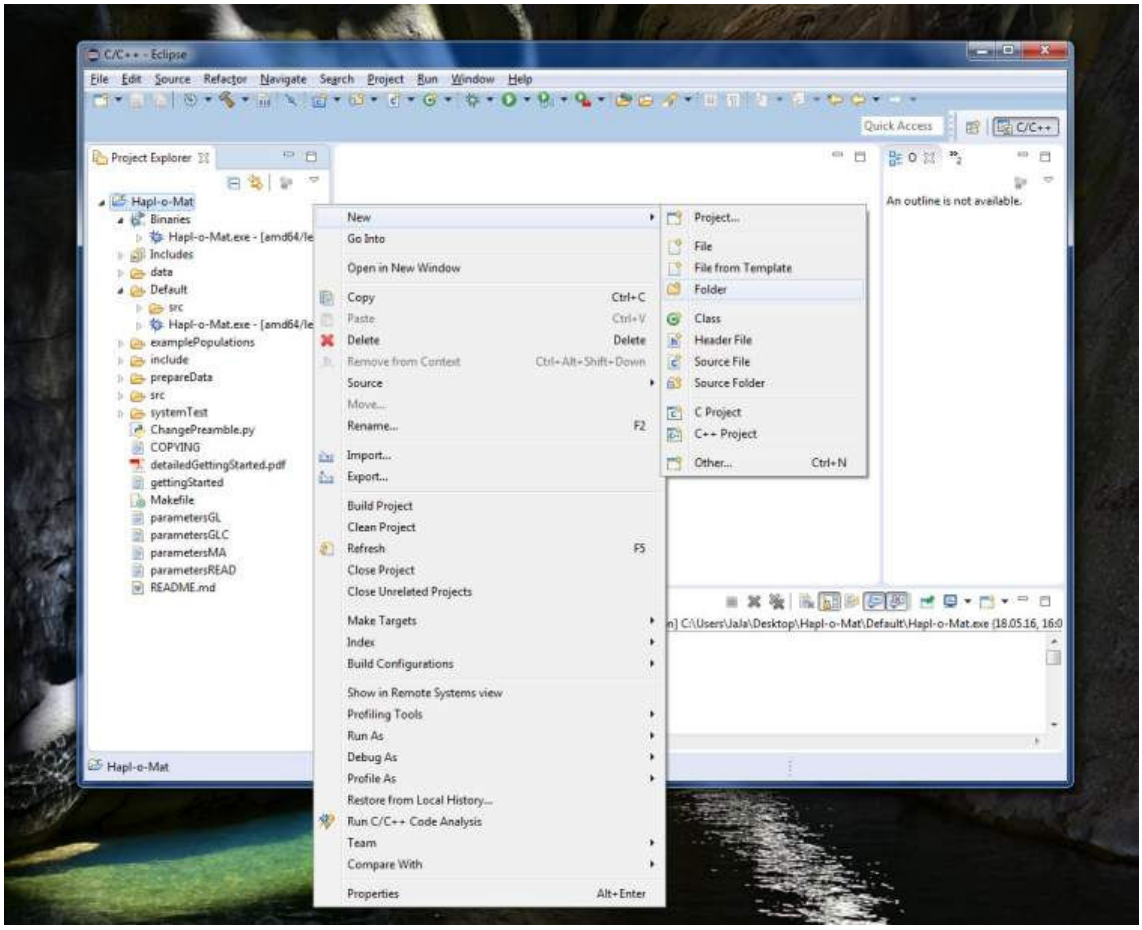
Tutorials

We have everything ready to use Hapl-o-Mat. In the following, we estimate haplotype frequencies from some included genotype data recorded in the input format MAC.

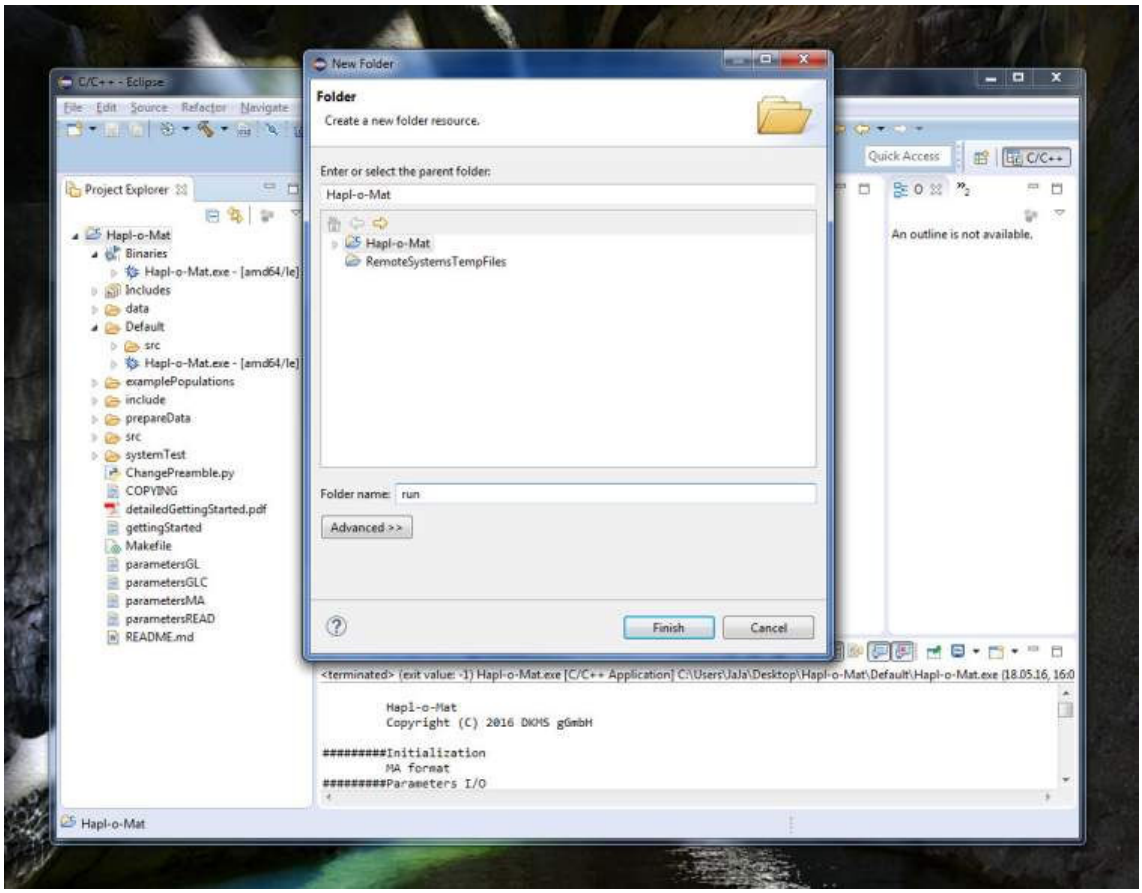
Input Format MAC

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. 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.

Create a folder by right-clicking in the Project Explorer and choosing New -> Folder.

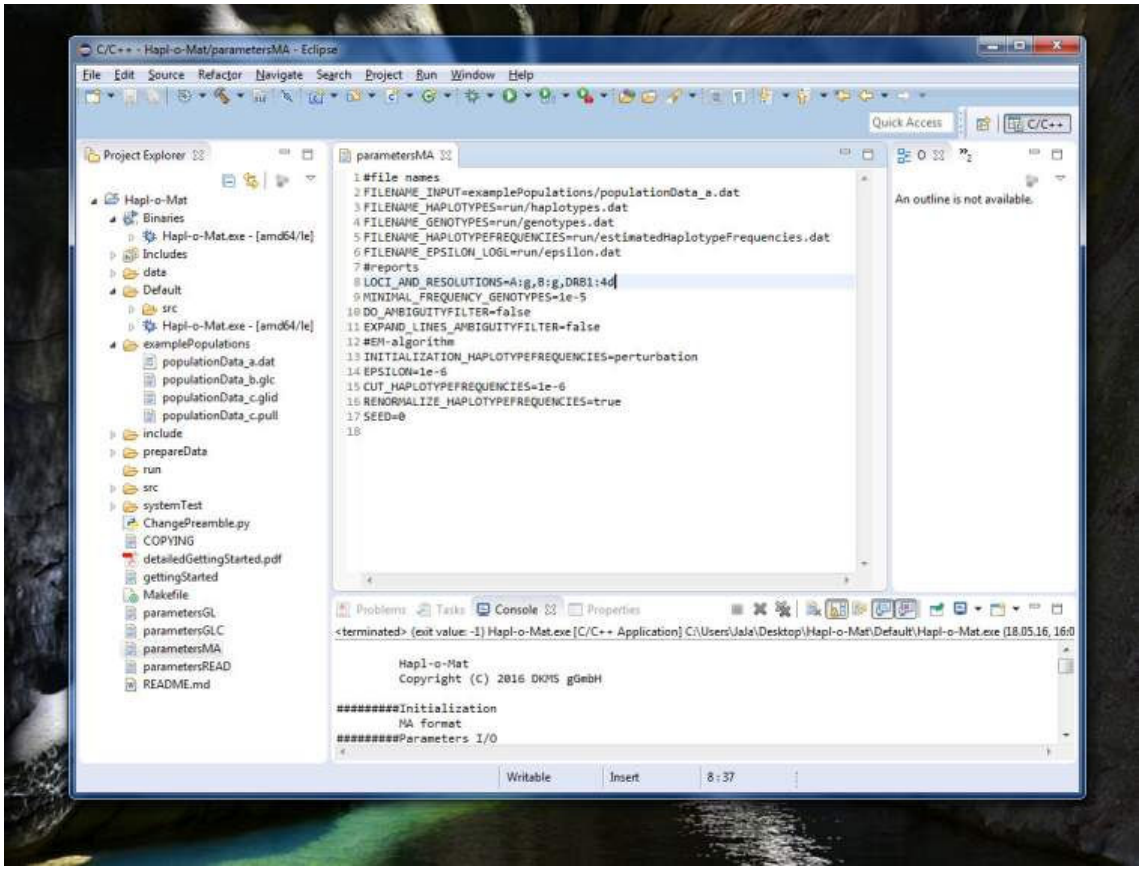


Name the folder “run” and then click Finish.

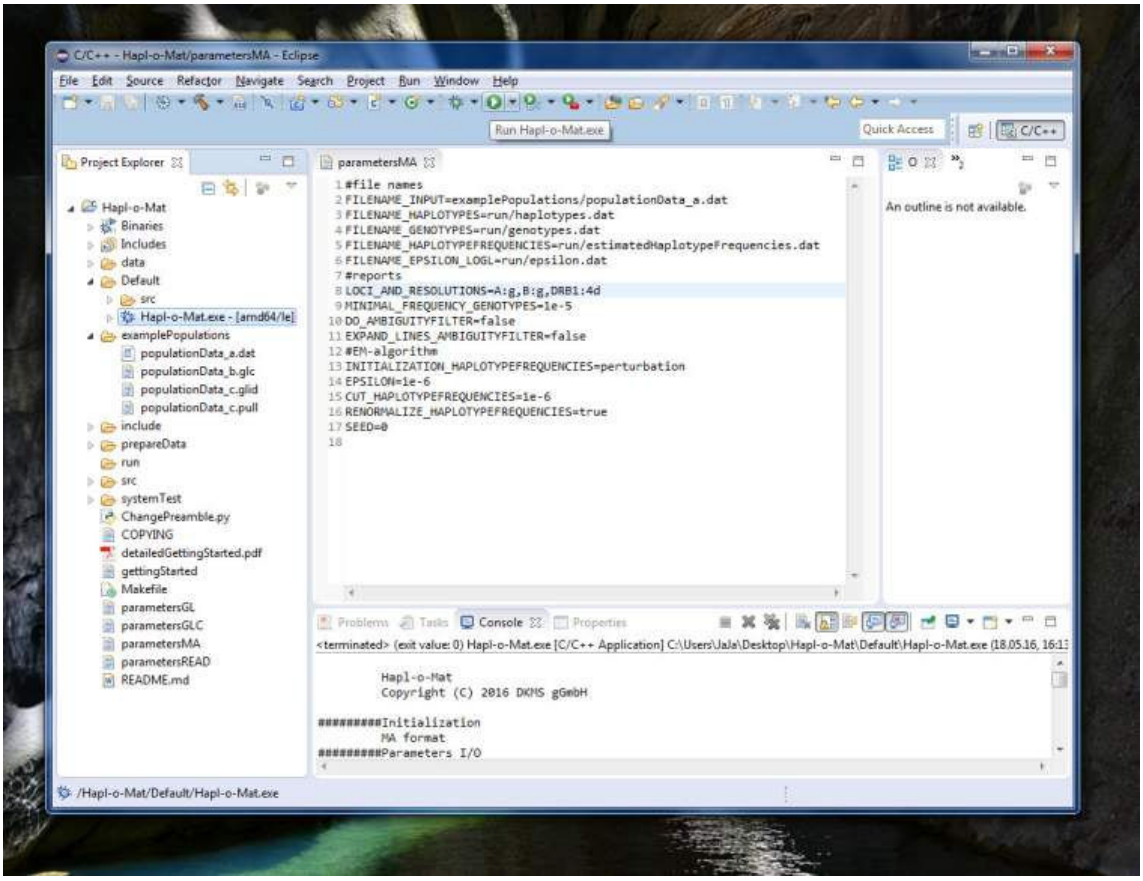


Next, open the parameter file parametersMAC by double-clicking on it. Change its parameters according to:

```
#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
#reports
LOCI_AND_RESOLUTIONS=A:g,B:g,DRB1:4d
MINIMAL_FREQUENCY_GENOTYPES=1e-5
DO_AMBIGUITYFILTER=false
EXPAND_LINES_AMBIGUITYFILTER=false
WRITE_GENOTYPES=true
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=1000
```

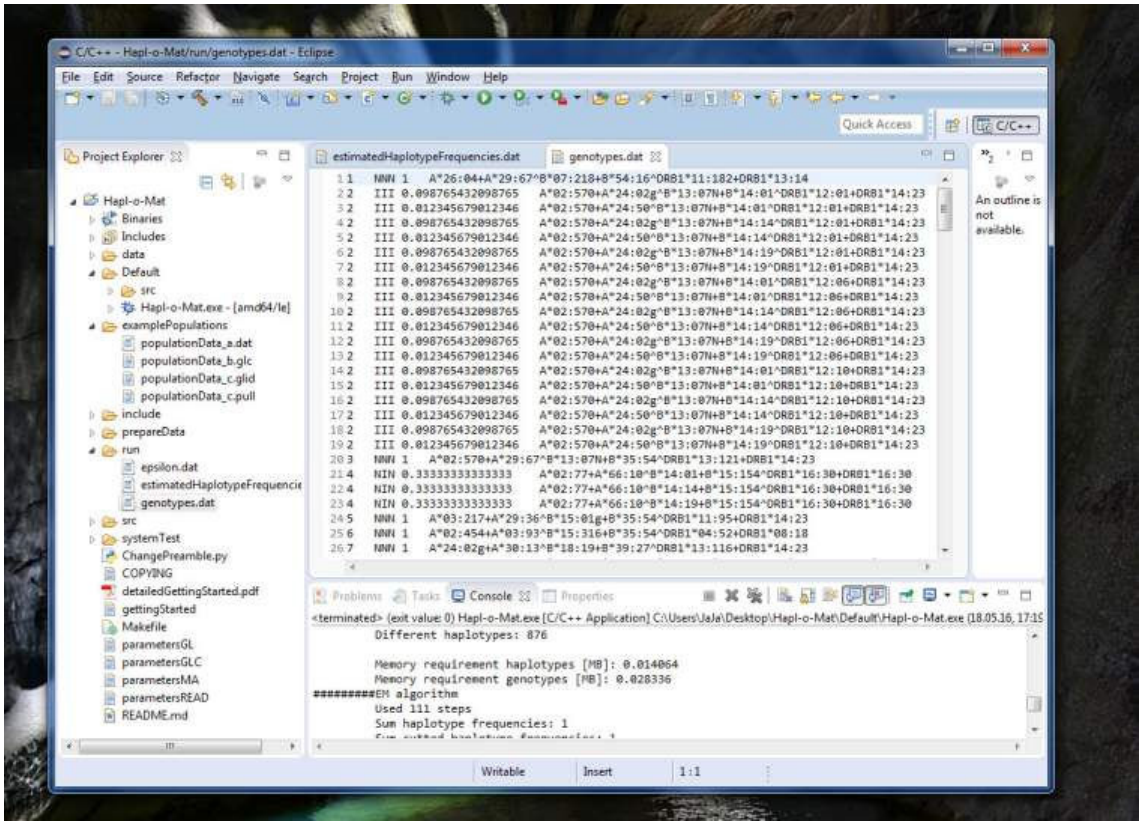


Compute haplotype frequencies from the genotype input data by running Hapl-o-Mat. To this end, click on the green Play button in the upper navigation bar.



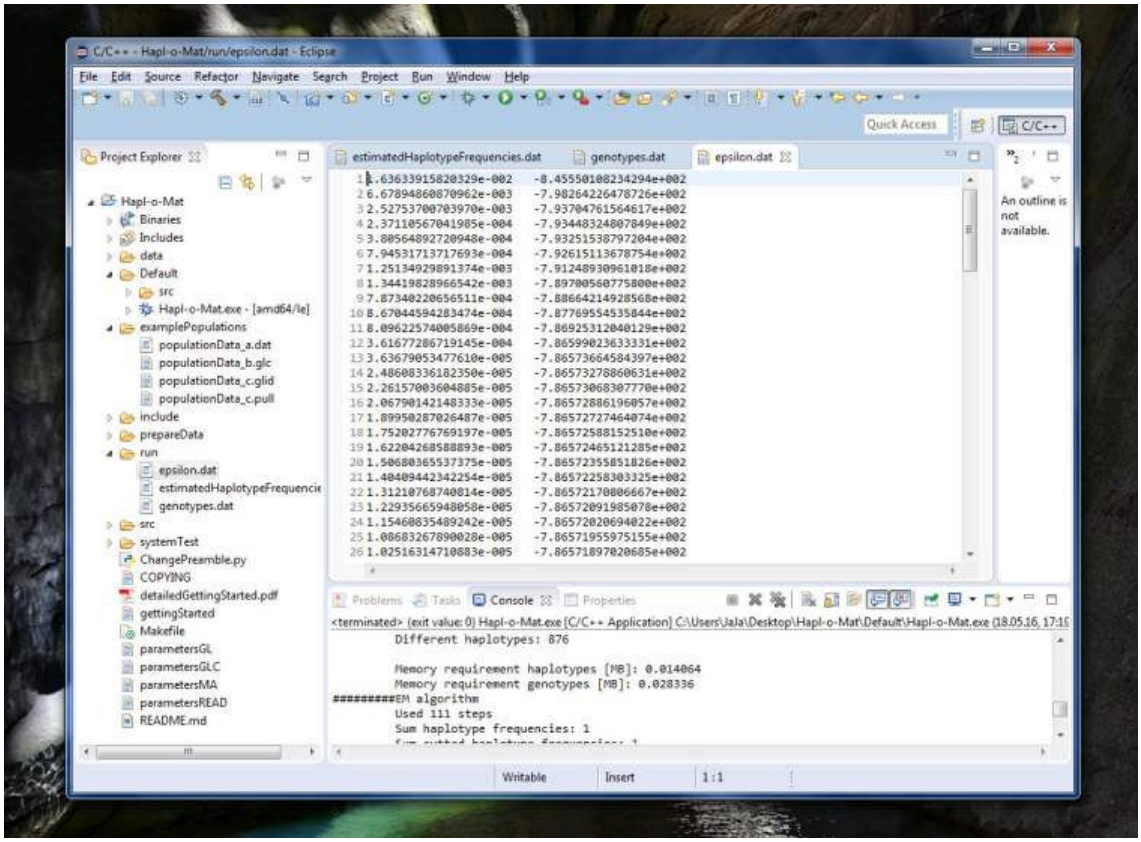
It produces some output on the screen including your chosen parameters, statistics on the resolved genotype data and the expectation-maximization algorithm, and the run time.

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”.

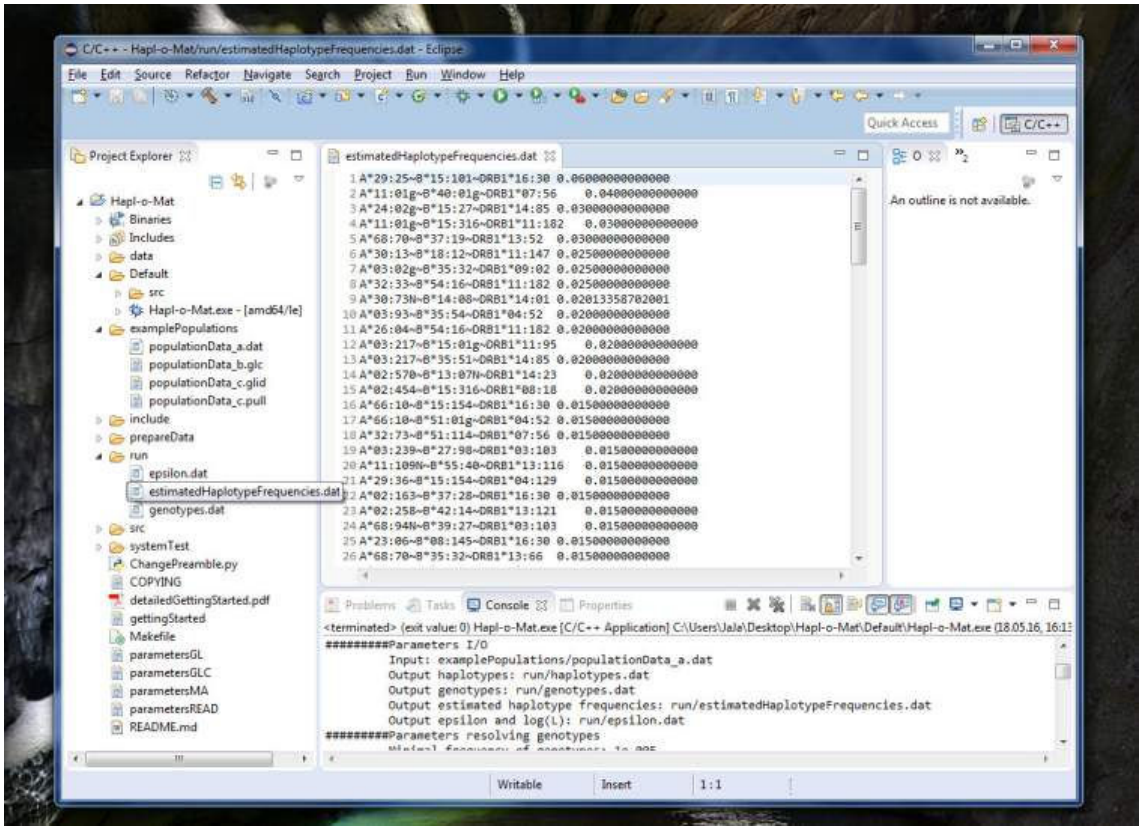


The first column corresponds to the individual's identification number. The second column indicates how ambiguities per single-locus genotypes have been resolved. If no ambiguity occurred or no additional genotypes are formed, the type is N. If an ambiguity occurred and was resolved via building all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



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).



You can also run Hapl-o-Mat from the command prompt, see the next section.

Input Format GLS

This time we run Hapl-o-Mat from the command prompt and use GL strings as input data. 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. The data files for the population data are "examplePopulations/populationData_c.pull" and "examplePopulations/populationData_c.glid".

Using for instance the Windows Explorer, create somewhere a new folder. Copy the folder "data", the executable "Hapl-o-Mat.exe", the parameter file "parametersGLS", and the input files "examplePopulations/populationData_c.pull" and "examplePopulations/populationData_c.glid" to your created folder. Then open "parametersGLS" in a text editor and adapt the parameters. 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". Now, your parameter file should look similar to that:

```
#file names
FILENAME_PULL=populationData_c.pull
FILENAME_GLID=populationData_c.glid
FILENAME_HAPLOTYPES=haplotypes.dat
FILENAME_GENOTYPES=genotypes.dat
FILENAME_HAPLOTYPEFREQUENCIES=estimatedHaplotypeFrequencies.dat
FILENAME_EPSILON_LOGL=epsilon.dat
#reports
LOCIORDER=B,A,DPB1,DRB1,C,DQB1
LOCI_AND_RESOLUTIONS=A:g,B:g,DRB1:4d
MINIMAL_FREQUENCY_GENOTYPES=1e-5
DO_AMBIGUITYFILTER=false
EXPAND_LINES_AMBIGUITYFILTER=false
RESOLVE_MISSING_GENOTYPES=false
WRITE_GENOTYPES=true
#EM-algorithm
INITIALIZATION_HAPLOTYPEFREQUENCIES=perturbation
EPSILON=1e-6
CUT_HAPLOTYPEFREQUENCIES=1e-6
RENORMALIZE_HAPLOTYPEFREQUENCIES=true
SEED=0
```

Open a command prompt (Windows symbol -> type Command Prompt), navigate to your created folder and run Hapl-o-Mat via

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

Afterwards, the result files should appear in your folder.