Hapl-o-Mat – Data Preparation

Please refer to gettingStarted, detailedGettingStartedLinux, or detailedGettingStartedWindows for information on how to use Hapl-o-Mat.

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 to update this information from time to time to allow new alleles to be handled by Hapl-o-Mat. Keep also in mind, that sometimes, rarely, alleles are also removed from the nomenclature (see also section "Invalid/Deprecated Alleles" below). 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. Enter the folder "prepareData". Everything is going to happen from here.

As the data-processing is a little bit tedious, we provide you with an automated script, see "Automated Way". If you prefer to do it all on your own, head to "Manual Way". The "Automated Way" relies on being able to download files from the internet. That can sometimes be hampered by firewall or proxy settings. If you can download files by different means and just want to skip fiddling with connectivity settings in python, we also provide a "Semi-Automated Way" that does everything for you except downloading files.

Automated Way

Just run the python script "BuildData.py", which does the whole job for you including creating the folder "Hapl-o-Mat/data" and moving the required files there.

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”.

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 refer to the section “Semi-Automated Way”.

If you want to analyse data in **GL-format with unresolved genotypes** (GL-id=0), you can prepare the file **AlleleList.txt** from the GL-id input file or optionally from AllAllelesExpanded.txt (all alleles known at the time of the data basis provided) by running BuildAlleleList.py (run “python BuildAlleleList.py” in folder “Hapl-o-Mat/prepareData”).

Semi-Automated Way

If you have tried the “Automated Way” and the script was not able to run properly to the end but issued an error message such as “unable to download file” or “connection timeout” then you might be able to still get your data prepared almost automated. Except for the download.

Presuming you have access to the internet and can download data, then to cope with the connection errors you can download the following four files manually: hla_nom_p.txt, hla_nom_g.txt, alpha.v3.zip, and hla_ambigs.xml.zip. You can get these files at the following locations:

1. hla_nom_p.txt
   Go to https://raw.githubusercontent.com/ANHIG/IMGTHLA/Latest/wmda/hla_nom_p.txt to get this file.
2. hla_nom_g.txt
   Go to https://raw.githubusercontent.com/ANHIG/IMGTHLA/Latest/wmda/hla_nom_g.txt to get this file.
3. alpha.v3.zip
   Go to https://hml.nmdp.org/mac/files/alpha.v3.zip to get this file.
4. hla_ambigs.xml.zip
   Go to https://raw.githubusercontent.com/ANHIG/IMGTHLA/Latest/xml/hla_ambigs.xml.zip to get this file.

These repositories are also denoted in the file “url_config.txt”.
If you need a more detailed description of what to do, please refer to “Manual Way”, section “Download Data”, steps 1, 2, 3, and 4a).

Once you have downloaded these files, place them in a separate folder of your liking for further reference. Then place a copy of these four files in the directory “Hapl-o-Mat/prepareData”. Please note, that the four files in “Hapl-o-Mat/prepareData” will be removed after data preparation so keeping a copy of them in separate folder of you liking is advised.

After copying the files to “Hapl-o-Mat/prepareData”, just run the script “BuildData.py” (e.g. via “python3 BuildData.py”). The script will realize that these files are already present, skip the download and proceed from there.

If you want to analyse data in **GL-format with unresolved genotypes** (GL-id=0), you can prepare the file **AlleleList.txt** from the GL-id input file or optionally from AllAllelesExpanded.txt (all alleles known at the time of the data basis provided) by running BuildAlleleList.py (run "python BuildAlleleList.py" in folder "Hapl-o-Mat/prepareData").

Source control for public internet repositories

The script "BuildData.py" 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.

The file "url_config.txt" needs to contain four lines; each denoting the source for the four input files hla_nom_p.txt, hla_nom_g.txt, alpha.v3.zip, and hla_ambigs.xml.zip. Each filename is to be followed by an equal sign and then the URL of the file in the repository, without any spaces. For example:

alpha.v3.zip=https://hml.nmdp.org/mac/files/alpha.v3.zip

You can adjust these URLs to your liking.

Troubleshooting

In rare cases, BuildData.py will not be able to succeed. This happens, if one or more of the files (hla_nom_p.txt, hla_nom_g.txt, alpha.v3.zip, hla_ambigs.xml.zip, g.txt, alpha.v3, or hla_ambigs.xml) is present in the directory "Hapl-o-Mat/prepareData" but is for any reason incomplete, broken, or corrupted in any way.
This situation can be redeemed by deleting these files, replacing them with newly downloaded copies and running BuildData.py again.

Manual Way

Here, we perform the data preparation step by step.

Download Data

First, we need some input data from the internet. Download the following files (The repositories are also denoted in the file "url_config.txt".):

1) Go to the website https://raw.githubusercontent.com/ANHIG/IMGTHLA/Latest/wmda/hla_nom_p.txt and save the file hla_nom_p.txt by right-clicking on the text and choosing "Save as…". Move the file "hla_nom_p.txt" to the folder "Hapl-o-Mat/prepareData".

2) Go to the website https://raw.githubusercontent.com/ANHIG/IMGTHLA/Latest/wmda/hla_nom_g.txt and save the file hla_nom_g.txt (same as in 1)). Move the file "hla_nom_g.txt" to the folder "Hapl-o-Mat/prepareData".

3) Download the file https://hml.nmdp.org/mac/files/alpha.v3.zip.

Extract the archive alpha.v3.zip. This should be straightforward in Windows. Using a Terminal under Linux you can use the command “unzip alpha.v3.zip”. You can remove the archive “alpha.v3.zip” afterwards. We only need the file “alpha.v3.txt”. Move it to the folder “Hapl-o-Mat/prepareData”.

4) Download the file https://raw.githubusercontent.com/ANHIG/IMGTHLA/Latest/xml/hla_ambigs.xml.zip . Extract it as in 3) including removing the archive. Move the file hla_ambigs.xml to folder “Hapl-o-Mat/prepareData”.

Build Data for Hapl-o-Mat

Now your are ready to build the data. Enter the folder “Hapl-o-Mat/prepareData” and run the following python scripts in the given order:

1) python BuildAllAllelesFrom_hla_nom_g.py
2) python BuildAllAllelesExpanded.py
3) python BuildP.py
4) python BuildLargeG.py
5) python BuildSmallg.py
6) python AddAllelesMissingIngCode.py
7) python TransferAlphaToMultipleAlleleCodes.py
8) python BuildAmbiguityFromXML.py
9) python AddGToAmbiguity.py
10) python TestAlleleVersions.py

Next, create the folder “Hapl-o-Mat/data” and move the freshly created files LargeG.txt, P.txt, Smallg.txt, Ambiguity.txt, MultipleAlleleCodes.txt, and AllAllelesExpanded.txt there. You can remove the files alpha.v3.txt, hla_ambigs.xml, hla_nom_g.txt, hla_nom_p.txt, allAlleles.txt, and OneElementG.txt.

If you want to analyse data in **GL-format with unresolved genotypes** (GL-id=0), you can prepare the file **AlleleList.txt** from the GL-id input file or optionally from AllAllelesExpanded.txt (all alleles known at the time of the data basis provided) by running BuildAlleleList.py (run “python BuildAlleleList.py” in folder “Hapl-o-Mat/prepareData”).

Invalid/Deprecated Alleles

The HLA nomenclature evolves over time, e.g. new alleles or new multiple allele codes are added, renamed and rarely removed from the nomenclature. For more details on invalid allele names see https://hla.alleles.org/alleles/deleted.html.

Multiple Allele Codes (MACs) that resolve to deprecated or invalid allele are not removed from alpha.v3 in subsequent versions, but complemented with new, valid MACs. If an input file contains Ids with such invalid alleles or MACs, the Ids will be removed before haplotype frequency estimations by Hapl-o-Mat. This would be indicated in detail in the terminal output of Hapl-o-Mat.

A list of all alleles (and the MACs that point to those alleles) that are contained in alpha.v3.txt but not existent in AllAllelesExpanded.txt (built from hla_nom_g.txt) in any locus combination is generated automatically during Data Preparation as “DeprecatedMultiAlleleCodes.txt” and can be found in the folder “manageInput/checkInputDeprecatedAlleles”.

This folder also contains the script “CheckInputDeprecatedAlleles.py” that will test your input for the abundance of alleles listed in “DeprecatedMultiAlleleCodes.txt”.

You may run the script

- either via command line with arguments as 'python3 CheckInputDeprecatedAlleles.py inputFormat inputPath', e.g.
  ```
  python3 CheckInputDeprecatedAlleles.py MAC ../../examplePopulations/populationData_a.dat
  ```
- or via command line with prompts asking for input format and input path (‘python3 CheckInputDeprecatedAlleles.py’) and follow the instructions.

The output (name of the input file with suffix “_CheckResults.txt”) will be saved in “manageInput/checkInputDeprecatedAlleles /results”.