LinkImputeR - User Guide

Daniel Money

January 2019

Contents

1 Change log

Version 0.9 *May 2016* Beta release.

Version 0.9.1 *August 2016* No longer crashes if no SNPs or samples pass filters - sensible messages given instead.

Version 0.9.2 *December 2016* Fixed bug if a read depth of zero is selected. Loosened allowed missing data representation in VCF files.

Version 1.0 *May 2017* Incorporate various changes in response to reviewers of the manuscript.

- Genotype sampling is now done without replacement.
- Seperate sets of masked genotypes are used for optimizing the imputation parameters, optimizing the combination parameter and reporting final accuracy.
- Option to use correlation instead of proportion correct.
- Correlation is now reported as well as proportion correct.
- Change to how Minor Allele Frequency is calculated.
- Option for a max read depth for genotypes above which the genotype is set to unknown.
- Two different genotypes sampling methods added for masking.

Version 1.1 *August 2017* Much improved error handling, handling gzipped input/output and other minor changes.

Version 1.1.1 *August 2017* Options to deal with different ways of specifying read counts (since this was only defined in later VCF standards).

Version 1.1.2 *August 2017* Minor bug fixes.

Version 1.1.3 *January 2019*

- Improved error handling if command line is specified incorrectly.
- Added option to output file containing true and imputed genotyped for every masked sample/position combination.
- Removed reliance on lib directory (now just the single jar is needed).

2 License

3 Citation

If you use LinkImputeR please cite us. The citation will be given here once the manuscript has been accepted.

4 Installation

LinkImputeR has been implemented in Java 8 and is distributed as an executable jar file. It is available from `http://www.cultivatingdiversity.org/software`.

To check if java is installed on your system type `java -version` at the command prompt. This will display the version of java installed. If an error message is displayed, or the version listed is less than 1.8 (Java 8 is called 1.8 when using the version command) you can download and install java (version >= 8) from the Oracle website (`http://www.java.com/en/download/index.jsp`). If you are having trouble getting versions of Java >= 8 running on a Mac please see our suggestions at the end of this document.

Once java is installed simply download the LinkImputeR zip file, uncompress and untar it. To do this on a linux-style command line (including Mac) use the command `tar -xvzf LinkImputeR.tar.gz`. Windows users will have to use an external program such as 7zip. Once uncompressed and and untarred follow the commands outlined in section 8.

5 Input Data

LinkImputeR takes a standard VCF file as input. Genotype fields should have, as a minimum, an AD field for read counts and a DP field for total read depth. In this version the GT field (for genotype) is also needed. LinkImputeR filters input data for biallelic SNPs before performing any other steps. The input file can be gzipped and this should be detected automatically.

6 Running LinkImputeR (Simple)

In a simple case running LinkImputeR will consist of four steps:

1. Create a control file in ini format. The control file includes information

on input and output files and the filters to be tested. Full details are in section 7.

2. Run LinkImputeR in accuracy mode. This produces data set size and accuracy statistics for each set of filters to be tested. Full details are in section 8

3. Look at the accuracy results and decide which filters you wish to use for the final imputation. Full details are in section 9.

4. Perform the final imputation. Full details are in section 8

7 Control File (Simple)

This is a standard ini file. Section headings are in square brackets. Within each section parameters and their values are separated by an equal sign. Each section heading or parameter / value pair should be on a separate line. Each section is described below. An example file is included with LinkImputeR and is called `accuracy.ini`. All file names can be relative or absolute. Optional parameters are marked with a *.

7.1 Input

This section is where the input file and an (optional) intermediate save file is given. (The intermediate save file is in this section as it is the input for the imputation stage of LinkImputeR).

filename The filename of the input.

save* If this parameter is included a vcf file will be outputted with this name after the filters in the InputFilters section have been applied. If the file name ends in .gz then the file will be gzipped.

maxdepth* Max read depth allowed for a genotype. Genotypes with a greater read depth are set to totally missing (i.e. no reads / no genotype). Defaults to 100.

readsformat* This paramater should be included if read information is included in the VCF in a non standard way. There are two options. If a single format is given then this is assumed to include comma separated read depths. If two formats are given comma separated then it is assumed the first format is for the reference allele read count, the second for the alternate allele read count. For example setting this to `RO,AO` would mean that reference allele counts are in RO and alternative allele counts are in

AO. If either option is used then both the save file vcf (see above) and any final imputed vcf will have had their read count information standardized to be in an AD field.

7.2 InputFilters

These filters are applied before any accuracy run is performed and hence are only performed once no matter how many other filter cases are being tested. In the case of filters also included under CaseFilters it is worthwhile including the least stringent of the cases here (in the case of positionmissing the least stringent of the missing cases in CaseFilters) as this will improve performance.

maf* Minor Allele Frequency filter. Value is the minimum MAF for the SNP to be included.

positionmissing* Position Missing filter. Value is the maximum missingness allowed per SNP.

hw* Hardy-Weinberg filter. Parameter is p-value below which SNPs are discarded.

7.3 CaseFilters

This section lists the different filters to be tested, for example you may wish to test multiple MAF filters with differing values for the allowed MAF. For each filter the different values to be tested for that filter are separated by commas (this is best illustrated by looking at the example ini file). The test cases then consist of every possible combination of the different filters.

maf* Minor Allele Frequency filter. Value is the minimum MAF for the SNP to be included.

missing* Missing filter. Value is the maximum missingness allowed per SNP and sample.

7.4 Global

Here are included two global parameters that can effect multiple filters.

depth* The minimum depth used to call a genotype for use in the filters.

error* The read error rate. Defaults to 0.01.

7.5 Accuracy

This section defines how the accuracy results should be calculated.

maskmethod* The method used to mask. Options are `all`, `bysnp`, `bysample`. `all` picks genotypes to mask at random across all snps with a depth of greater or equal to 30 reads. `bysnp` first picks a snp at random then picks a genotype with sufficient depth from that snp. `bysample` is similar except it operates on samples rather than snps. Default is `all`.

accuracymethod* The method used to calculate accuracy. Options are `correct` and `correlation`. `correct` uses the propotion of imputed genotypes that are the same as the true genotype while `correlation` uses the correlation between imputed and true genotypes. Default is `correct`.

numbermasked* The number of genotypes to mask. Default is 10 000.

mindepth* The minimum read depth required for a genotype to be considered for masking. Default is 30.

7.6 Stats

This section defines how the accuracy results should be outputted.

root The root directory to which accuracy statistics should be saved.

level How much output should be written. Options are `sum`, `pretty` and `table`. `sum` will just produce a summary file, `pretty` will output extra files including information in a easy human readable way and `table` will also output the same information in a tab-delimited format.

eachmasked* Whether to output a file containing the true and imputed genotype for each sample / position masked. Valid values are `yes` and `no`. Default is no.

7.7 Output

Where should the output be written

control The file name of the output control file which will be used in the imputation stage of LinkImputeR.

7.8 Log

This section controls logging

file* The file name of the log file.

level* How much should be logged. Options are, in order of increasing output, `critical` (default), `brief`, `detail` and `debug`.

8 Command Line (Simple)

LinkImputeR is a command line tool. In the simple run case you will run two commands, one to produce accuracy statistics and one to do the actual imputation. To perform the accuracy calculation the command is:

```
java -jar LinkImputeR.jar -s INIFILE
```

where `INIFILE` is the name of the ini control file. An example is:

```
java -jar LinkImputeR.jar -s accuracy.ini
```

And then to perform the actual imputation: `java -jar LinkImputeR.jar CONTROL CASE OUTPUT`

where `CONTROL` is the name of the control file you asked to be created in the ini file, `CASE` is the name of the case you want to do imputation for and `OUTPUT` is the name of the imputed vcf file. `CASE` will have to be in quotes since the default case names include spaces. If `OUTPUT` ends in .gz then the file will be gzipped. An example command line is:

```
java -jar LinkImputeR.jar impute.xml 'Case 2' output.vcf
```

LinkImputeR can use a significant amount of memory on a large dataset. If you encounter an out of memory error consider increasing the amount of memory available to Java by using the `-Xmx` java option. This is a java option not a LinkImputeR option so needs to be included between `java` and `-jar`. An example usage would be `-Xmx6G` where 6G means make 6GB of memory available. There should be no space between the 6 and the G. For example (where `...` is the rest of the command as above):

```
java -Xmx6G -jar LinkImputeR.jar ...
```

9 Accuracy Results

The following files are produced in the stats directory.

`sum.dat` includes information on each case run, the number of SNPs, the number of samples and accuracy.

`table.dat` contains information on the case name, number of snps, number of samples, accuracy (in that order) in tab-delimited format.

The following files are produced for every case. The relevant case number can be found in `sum.dat`.

`pretty_`*`casenumber`*`.dat` is a summary of accuracy by depth, by "real" genotype and by total read depth & "real" genotype in human readable format.

`geno_`*`casenumber`*`.dat` contains information on the accuracy by "real" genotype in tab deliminted format.

`depth_`*`casenumber`*`.dat` contains information on the accuracy by total read depth in tab deliminted format.

`dg_`*`casenumber`*`.dat` contains information on the accuracy by "real" genotype and total read depth in tab deliminted format.

`each_`*`casenumber`*`.dat` contains the true and imputed genotyped for each sample / position masked.

10 Outputs

Running LinkImputeR in imputation mode will produce an output vcf file. This file will be similar to the input file except the GT field will now contain the imputed genotypes. The unimputed genotypes (i.e. those previously in the GT field) will now be in a new UG field. Finally imputation genotype probabilities will be in a new IP field. Appropriate changes are also made to the VCF meta data.

11 Running LinkImputeR (Advanced)

LinkImputeR actually uses XML files as input - the simple input described above is converted into XML before further processing. To run LimkImputeR with an XML file run LinkImputeR with just the XML file name without any

flags, e.g.:

```
java -Xmx6G -jar LinkImputeR.jar filename
```

To produce the XML file that is the equivalent to a simple input file (as described above) run (no calculations will be performed):

```
java -Xmx6G -jar LinkImputeR.jar -c ini_filename xml_filename
```

Looking at the XML files generated by LinkImputeR should provide a lot of information on these files. If further information is required please contact the author.

12 Contact

LinkImputeR is maintained by Daniel Money who can be contacted at `info@cultivatingdiversity.org`.

13 Java and Macs

Getting a version of java greater than 6 to run properly on some older Macs is known to be problematic. We found that the easiest way to do so is to start by downloading and installing the JDK (not the JRE). Next you need to edit your `.profile` which should be in your home directory and which you should create if it doesn't exist. For example to edit the file in `vim` you could type:

```
vim ~/.profile
```

You should then add the following line to to file (replacing 1.8 with 1.7 depending on which version you have installed):

```
export JAVA_HOME=`/usr/libexec/java_home -v 1.8`
```

Your `.profile` is processed automatically whenever you start a terminal session so this should allow you to run the new version of java whenever you start a new session. To run the correct version of Java in the current session without restarting it you need to process your `.profile` manually by typing:

```
source ~/profile
```

This problem is with Java on Macs rather than LinkImputeR itself. This means we will probably be unable to offer support for your specific setup if the above does not work for you. Instead we suggest that you browse the extensive discussion of this problem on the internet.