LinkImpute - User Guide

Daniel Money

January 2020

Contents

1 Change log

Version 1.0 *August 2015* Initial release.

Version 1.1 *September 2015* Added support for Plink ped files.

Version 1.1.1 *November 2015* Bug fix (would mangle data if an allele was encoded as a 1 in ped format).

Version 1.1.2 *July 2016* Bug fixes (did not allow duplicate family ids, would output identical SNPs if using VCF format).

Version 1.1.3 *October 2017* Added option to change the number of genotypes masked during optimization.

Version 1.1.4 *July 2019* Bug fix (changed known genotypes for some snps when any other snp did not have enough genotypes for imputation).

Version 1.1.5 *January 2020* Change java version required to Java 8.

2 License

LinkImpute is licensed under GPL v3 (see gpl.txt for more information).

3 Citation

If you use LinkImpute please cite us. The advance access citation is:

D. Money, K. Gardner, Z. Migicovsky, H. Schwaninger, G-Y. Zhong, and S. Myles (2015) LinkImpute: Fast and Accurate Genotype Imputation for Non-Model Organisms *G3: Genes, Genomes, Genetics* g3.115.021667

4 Overview

LD-kNNi is a *k*-nearest neighbour genotype imputation method designed for unordered markers and implemented in LinkImpute. No physical or genetic maps are required and it is designed to work on unphased genotype data from heterozygous species. It exploits the fact that markers useful for imputation are often not physically close to the missing genotype but rather distributed throughout the genome. Performance, in terms of accuracy and speed, is comparable to Beagle but, unlike Beagle, LD-kNNi does not require ordered markers.

5 Installation

LinkImpute 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 LinkImpute zip file, uncompress and untar it. To do this on a linux-style command line (including Mac) use the command `tar -xvzf LinkImpute.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 7.

6 Input data

The input to LinkImpute is a genome file. This can be a Plink raw file (produced with the Plink command `--recodeA`), a Plink ped file, a vcf file or a simple array file. An example Plink raw file (`apple.raw`) is included with the LinkImpute distribution.

If using the Plink ped option LinkImpute only requires the .ped file and not the associated .map file. LinkImpute will only output a new .ped file. The .map file does not change as a result of imputation. LinkImpute requires the unknown allele character to be a zero.

The VCF option is very much experimental, poorly documented and currently unsupported. If it's convenient for you, give it a try but be aware that it may not work.

The array file is a white-space delimited file containing a line per sample. Each sample line should list the numerical genotype for each SNP for that sample, space separated. Unknown genotypes should be coded as `-1`.

7 Running LinkImpute (Simple)

LinkImpute is a command line tool and a simple command line is described below. All options other than `INFILE` and `OUTFILE` are optional. `java -jar LinkImpute.jar [-p | -q | -a | -v] INFILE OUTFILE`

Options are:

`-p`	Use plink raw file format (default)
`-q`	Use plink ped file format
`-a`	Use array file format
`-v`	Use VCF file format (experimental)

So the simplest command line, using Plink format files, would be:

```
java -jar LinkImpute.jar INFILE OUTFILE
```

To use VCF formatted files instead:

```
java -jar LinkImpute.jar -v INFILE OUTFILE
```

LinkImpute 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 LinkImpute 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. A complete example command line, using LD-kNNi and Plink files would be:

```
java -Xmx6G -jar LinkImpute.jar INFILE OUTFILE
```

8 Running LinkImpute (Advanced)

The full command line for LinkImpute is:

```
java -jar LinkImpute.jar [-p | -q | -a | -v]
[--knni | --mode] [--verbose]
[--fixedk=<arg>] [--fixedl=<arg>]
[--ldin=<arg>] [--ldnum=<arg>] [--ldout=<arg>] [--ldonly]
[--nummask=<arg]
INFILE OUTFILE
```

The extra options fall into five groups. The first allow use of mode or standard kNNi imputation instead of LD-kNNi. The second turns on verbose mode that displays extra information about optimizing parameters (k and l) and runtimes. The third groups allows the value of k and l to be fixed at given values rather than be optimized.

The fourth group of options concern creating or reading in a file containing LD information for each SNP. The file contains one line per SNP. The first field on each line is the SNP of interest and the following fields are the SNPs most in LD with that SNP starting at the SNP most in LD. SNPs are identified by their position in the input file. These options were used in the development of LinkImpute and are left here in case either a) users want to use some other method of generating relationships between the SNPs (LinkImpute uses the Pearson correlation as a measure of LD) or b) the information is of use in analysing the dataset and/or the performance of LinkImpute.

The final group controls parameters used during optimization.

Options are:

`-p`	Use plink raw file format (default)
`-p`	Use plink ped file format
`-a`	Use array file format
`-v`	Use VCF file format (experimental)
`--knni`	Use kNN imputation instead of LD-kNNi
`--mode`	Use mode imputation instead of LD-kNNi
`--verbose`	Display detailed run information
`--fixedk`	Fix the value of k to the given value
`--fixedl`	Fix the value of l to the given value
`--ldin=<arg>`	Read LD information from the given file rather than calculate it
`--ldout=<arg>`	Output the SNPs most in LD with each SNP to the given file
`--ldnum=<arg>`	Output the given number of SNPs most in LD. Defaults to 65
`--ldonly`	Do not perform the imputation. Use to obtain just the LD information
`--nummask=<arg>`	Number of genotypes to mask during optimization
`INFILE`	Input file name. Can be either relative or absolute.
`OUTFULE`	Output file name. Can be either relative or absolute.
`--help`	Display a help message

9 Outputs

Running LinkImpute will produce an output file, in the same format as the input file, with the unknown values changed to their imputed values. In all other respected it will be the same as the original file, except for the following minor differences.

PED format – The two alleles of a genotype may be swapped (e.g. input could have 'A G' and output could have 'G A').
Array format – All white space separators will be a single tab.
VCF format – The header lines may be in a different order.

10 Contact

LinkImpute is maintained by Daniel Money who can be contacted at `daniel@danielmoney.co.uk`.

11 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:

```
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 LinkImpute 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.