LASER: Locating Ancestry from SEquence Reads version 2.04

Chaolong Wang[1]
Computational and Systems Biology
Genome Institute of Singapore
A*STAR, Singapore 138672, Singapore

Xiaowei Zhan[2]
Department of Clinical Sciences
Southwestern Medical Center
University of Texas, TX 75235, USA

April 6, 2017

The *LASER* software[3] is available at
http://csg.sph.umich.edu/chaolong/LASER/

[1]Comments on the *LASER* software can be sent to chaolong@umich.edu.
[2]Comments on tools for preparing the input files can be sent to zhanxw@umich.edu.
[3]This software is licensed under the GNU General Public License, version 3.0.

Contents

1 Introduction **3**

2 Getting started **4**

2.1 Availability . . . 4
2.2 Installing *LASER* . . . 4
2.3 Running *LASER* . . . 4

3 Examples **5**

3.1 Basic usage . . . 5
3.2 Checking format . . . 7
3.3 Parallel jobs . . . 7
3.4 Repeated runs . . . 8
3.5 Checking coverage . . . 8
3.6 Excluding loci . . . 9
3.7 PCA mode . . . 9

4 Input files **10**

4.1 *parameterfile* (*_.conf*) . . . 10
4.2 *genotypefile* (*_.geno*) and *sitefile* (*_.site*) . . . 11
4.3 *sequencefile* (*_.seq*) . . . 11
4.4 *coordinatefile* (*_.coord*) . . . 14

5 Usage options **14**

5.1 Main parameters . . . 14
5.2 Advanced parameters . . . 15
5.3 Command line arguments . . . 18

6 Output files **20**

6.1 *_.log* and terminal outputs . . . 20
6.2 *_.RefPC.coord* and *_.RefPC.var* . . . 21
6.3 *_.RefPC.grm* . . . 21
6.4 *_.RefPC.load* . . . 21
6.5 *_.SeqPC.coord* . . . 22
6.6 *_.SeqPC.coord.sd* and *_.SeqPC.coord.reps* . . . 22
6.7 *_.ind.cov* and *_.loc.cov* . . . 23

7 Computational complexity **23**

8 Version changes **24**
8.1 Version 1.0 (Feb 1, 2013) . . . 24
8.2 Version 1.01 (Mar 11, 2013) . . . 24
8.3 Version 1.02 (Jun 19, 2013) . . . 24
8.4 Version 1.03 (Aug 8, 2013) . . . 24
8.5 Version 2.0 (May 19, 2014) . . . 24
8.6 Version 2.01 (Jun 5, 2014) . . . 25
8.7 Version 2.02 (Apr 6, 2015) . . . 25
8.8 Version 2.03 (Feb 25, 2016) . . . 26
8.9 Version 2.04 (Jan 11, 2017) . . . 26

9 Acknowledgements **26**

References **27**

1 Introduction

LASER is a software program for analyzing next generation sequencing data, and it can estimate individual ancestry by directly analyzing shortgun sequence reads without calling genotypes. The method relies on the availability of a set of reference individuals whose genome-wide SNP genotypes and ancestry information are both available. *LASER* first constructs a reference ancestry space by applying principal components analysis (PCA) to the genotype data of the reference individuals. Then, *LASER* analyzes genome-wide sequence reads for each sequence sample to place the sample into the reference PCA space. With an appropriate reference panel, the estimated coordinates of the sequence samples reflect their ancestral background and can be used to correct for population stratification in association studies.

To place each sample, we proceed as follows. First, we simulate sequence data for each reference individuals, exactly matching the coverage pattern of the sample being studied (in this way, each sequenced sample will have the same number of reads covering each locus as the study sample). Then, we build a PCA ancestry map based on these simulated sequence reads for the reference samples together with the real sequence reads for the study sample. Finally, using Procrustes analysis (Wang *et al.*, 2010) we project this new ancestry map into the reference PCA space. The transformation obtained from this analysis of the reference samples is then used to place the study sample in the reference PCA space. We repeat this procedure on every study sample until all samples are mapped to the reference PCA space. The first version of this method was described in (Wang *et al.*, 2014). In the second version, we introduce a projection Procrustes analysis technique (Gower and Dijksterhuis, 2004), which together with genotype imputation can substantially improve our ancestry estimation (Wang *et al.*, 2015). For details, please refer to our papers (Wang *et al.*, 2014, 2015).

Using both simulations and empirical data, we have shown that our method can accurately infer the worldwide continental ancestry or even the fine-scale ancestry within Europe with extremely low-coverage sequencing (Wang *et al.*, 2014). We expect this method to be particularly useful for studies based on targeted sequencing technologies, in which genetic variation within target regions often do not provide sufficient information for ancestry inference. In this setting, *LASER* can provide accurate estimation of individual ancestry by combining information from sequence reads that are mapped to genome-wide off-target regions. It is worth noting that one should be extremely careful in interpreting the results when the reference panel does not include ancestries in the study sample. For this reason, we recommend users to start with a worldwide reference panel and gradually focus on more regional reference panels.

In addition to estimation of individual ancestry using sequence reads, *LASER* also provides an option to perform standard PCA on genotype data. This option is implemented to prepare the reference PCA coordinate file as an input for *LASER*. It can also be used independently as a PCA tool for analyzing population structure based on SNP genotypes.

2 Getting started

2.1 Availability

A pre-compiled executable for *LASER* for Linux (64-bit) operation systems can be downloaded from the following webpage: http://csg.sph.umich.edu/chaolong/LASER/. This program is licensed under the GNU General Public License, version 3.0. A copy of the license is included in the package or can be found at http://www/gnu.org/licenses.

Source code written in C++ is provided in the package. The *LASER* program uses two external libraries: the Armadillo Linear Algebra Library (http://arma.sourceforge.net) (Sanderson and Curtin, 2016) and the GNU Scientific Library (http://www.gnu.org/software/gsl). The Armadillo library requires two additional libraries: LAPACK and OpenBLAS. Therefore, to compile from source code, you need to have these four libraries installed in your computer.

Please cite the following papers when you use *LASER*:

1. Wang *et. al.* (2014) Ancestry estimation and control of population stratification in sequence-based association studies. *Nature Genetics*, 46: 409-415.
2. Wang *et al.* (2015) Improved ancestry estimation for both genotyping and sequencing data using projection Procrustes analysis and genotype imputation. *AJHG*, 96: 926-937.

2.2 Installing *LASER*

Open a terminal in the same directory as the `.tar.gz` file. Extract the file by typing `tar -xzvf LASER-2.04.tar.gz` in the terminal. This will create a new directory called `LASER-2.04`. This direcotry contains executables for both *LASER* (version 2.04) and *TRACE* (version 1.03), as well as two companion tools, *vcf2geno* and *pileup2seq*.

2.3 Running *LASER*

Open a terminal and path to the directory that contains the executable *LASER*. If you did not rename the directory after extracting the `.tar.gz` file, the directory will be `LASER-2.04`. Execute the program by typing `./laser -p` *`parameterfile`*, in which `-p` is the command line flag specifying the parameter file and *parameterfile* is the name of the parameter file. If your *parameterfile* is not in the same directory, you must specify the whole path to the file. If the *parameterfile* is not specified, *LASER* will search in the current directory for a *parameterfile* named "laser.conf", and execute the program with parameter values specified in "laser.conf". If this file does not exist, an empty template *parameterfile* named "laser.conf" will be created in the current directory. For more command line arguments, see Section 5.3.

3 Examples

This section provides example usage of the *LASER* program based on data in the folder named "example" (included in the download package). If you have questions when reading this section, please refer to the next few sections for detailed information about the input files (Section 4), usage options (Section 5), and output files (Section 6).

In the "example" folder, the *HGDP_200_chr22.geno* file is a *genotypefile* that includes genotypes at 9,608 SNP loci on chromosome 22 for 200 individuals randomly selected from the Human Genome Diversity Panel (HGDP, Li *et al.*, 2008). The *HGDP_200_chr22.site* file is the corresponding *sitefile*. The *HGDP_200_chr22.RefPC.coord* file contains PCA coordinates for the top 8 PCs based on genotypes in the *HGDP_200_chr22.geno* file. The *HapMap_6_chr22.seq* file is the *sequencefile* for 6 test samples, whose sequence reads were piled up to the 9,608 SNP loci in the *HGDP_200_chr22.site* file. The folder also includes a *parameterfile* named "example.conf", which specifies parameters for running *LASER* on the example data.

3.1 Basic usage

After decompressing the download package, enter the folder that contains the executable *LASER* program. The following command will use parameter values provided in the example *parameterfile* (shown at the end of this section).

```
./laser -p ./example/example.conf
```

The following command will change the dimension of the reference space (*DIM*) to 10, the number of PCs used for projection (*DIM_HIGH*) to 20, the prefix of output files (*OUT_PREFIX*) to "HapMap.example", and use the other parameters as defined by the *example.conf* file.

```
./laser -p ./example/example.conf -o HapMap.example -k 10 -K 20
```

Results from *LASER* will be output to the current working directory.

The example *parameterfile* is similar to the one shown below. Each line specifies one parameter, followed by the parameter value (or followed by a "#" character if the parameter is undefined). Text after a "#" character in each line is treated as comments.

```
# This is a parameter file for LASER v2.03.
# The entire line after a '#' will be ignored.
###----Main Parameters----###
GENO_FILE     ./example/HGDP_200_chr22.geno          # no default value
SEQ_FILE      ./example/HapMap_6_chr22.seq           # no default value
COORD_FILE    ./example/HGDP_200_chr22.RefPC.coord   # no default value
OUT_PREFIX    test                                   # default "laser"
```

```
DIM           2                                          # default 2
DIM_HIGH      10                                         # default 20
MIN_LOCI      100                                        # default 100
###----Advanced Parameters----###
SEQ_ERR             # default -1 [use estimated values in the SEQ_FILE]
ALPHA               # default 0.1
THRESHOLD           # default 0.000001
FIRST_IND           # default 1
LAST_IND            # default [last sample in the SEQ_FILE]
REPS                # default 1
OUTPUT_REPS         # default 0
CHECK_FORMAT        # default 10
CHECK_COVERAGE      # default 0
PCA_MODE            # default 0
REF_SIZE            # default [sample size of the GENO_FILE]
TRIM_PROP           # default 0
EXCLUDE_LIST        # no default value
MIN_COVERAGE        # default 0
MAX_COVERAGE        # default -1 [do not exclude any site]
PROCRUSTES_SCALE    # default 0 [include scaling as a free parameter]
RANDOM_SEED         # default 0
###----Command line arguments----###
# -p    parameterfile (this file)
# -g    GENO_FILE
# -s    SEQ_FILE
# -c    COORD_FILE
# -o    OUT_PREFIX
# -k    DIM
# -K    DIM_HIGH
# -l    MIN_LOCI
# -e    SEQ_ERR
# -a    ALPHA
# -t    THRESHOLD
# -x    FIRST_IND
# -y    LAST_IND
# -r    REPS
# -R    OUTPUT_REPS
```

```
# -cov  CHECK_COVERAGE
# -fmt  CHECK_FORMAT
# -pca  PCA_MODE
# -N    REF_SIZE
# -M    TRIM_PROP
# -ex   EXCLUDE_LIST
# -minc MAX_COVERAGE
# -maxc MAX_COVERAGE
# -rho  PROCRUSTES_SCALE
# -seed RANDOM_SEED
###----End of file----###
```

3.2 Checking format

The parameter *CHECK_FORMAT* (command line flag `-fmt`) provides options to check the format of different input data files. By default, *LASER* will first check all input data files before starting the analysis. Sometimes it is unnecessary to check the format of all input files. For example, if the same *genotypefile* is used to analyze multiple sets of sequence samples, we only need to check the format of the *genotypefile* once. The following command will turn off check-format function and directly proceed to the main analysis:

```
./laser -p ./example/example.conf -fmt 0
```

And the following command will only check the format of the *sequencefile* and proceed to the main analysis:

```
./laser -p ./example/example.conf -fmt 30
```

Please refer to Section 5.2 for more options regarding the parameter *CHECK_FORMAT*.

3.3 Parallel jobs

Because each sequence sample is analyzed independently, users can easily parallel the analyses by running multiple jobs simultaneously. The `-x` and `-y` flags provide a convenient way to specify a subset of samples to analyze in each job. We recommend users first run *LASER* with `-fmt 1` to check the format of all input data files, and then run multiple parallel jobs with with `-fmt 0` to analyze different subsets of samples without repeatedly checking the data format. For example, first run the following command to check the format of input files *HGDP_200_chr22.geno*, *HGDP_200_chr22.RefPC.coord*, and *HapMap_6_chr22.seq*:

```
./laser -p ./example/example.conf -fmt 1
```

And then run the following commands to submit two jobs: the first job will analyze samples 1 to 3; and the second job will analyze samples 4 to 6 in the *HapMap_6_chr22.seq* file.

```
./laser -p ./example/example.conf -fmt 0 -x 1 -y 3 -o results.1-3 &
./laser -p ./example/example.conf -fmt 0 -x 4 -y 5 -o results.4-6 &
```

Outputs from these two jobs will have different file name prefixes *results.1-3* and *results.4-6* specified by the `-o` flag. We also recommend users to provide the *coordinatefile* when running multiple jobs using the same set of of reference individuals to save computational time by avoiding redundant calculation of the reference PCA coordinates in each job.

3.4 Repeated runs

Stochastic variation introduced by the simulation procedure of the *LASER* method can lead to slightly different results in placement of each sequence sample. We have shown that by running the *LASER* analysis on the same sample multiple times and then using the mean coordinates averaged across multiple repeated runs can provide a higher accuracy than using coordinates from a single run (Wang *et al.*, 2014). The parameter *REPS* (command line flag `-r`) provides this option and allows users to set different number of repeated runs. For example, the following command line will run the *LASER* analysis 3 times on each sequence sample and output the mean and standard deviation across results from all 3 repeated runs:

```
./laser -p ./example/example.conf -r 3
```

By default, results from each single run will not be output. If users are interested in seeing results from all repeated runs, they can set the parameter value of *OUTPUT_REPS* to 1 in the *parameterfile*, or use the following command line:

```
./laser -p ./example/example.conf -r 3 -R 1
```

Please refer to Section 6.6 for description of the output files. Note that the computational time will increase linearly with the number of repeated runs.

3.5 Checking coverage

The parameter *CHECK_COVERAGE* (command line flag `-cov`) provides options to summarize the sequencing depth per site and per sample based on data in the *sequencefile*. By default, *LASER* will not check the coverage. The following command will first calculate the coverage of samples in the *sequencefile* and then proceed to the main analysis:

```
./laser -p ./example/example.conf -cov 1
```

Using `-cov 2` will only check the coverage and then stop.

3.6 Excluding loci

LASER implement four parameters to provide options to exclude loci from the analysis. These parameters are useful to exclude problematic SNPs or to reduce the memory usage and to speed up the computation with potential cost of losing accuracy. The parameter *EXCLUDE_LIST* (command line flag `-ex`) specifies a given list of SNPs to exclude. The list of SNPs need to be provided in a file, in which each line is a SNP ID. For example,

```
rs17433377
rs2190742
rs5748623
...
```

The parameter *TRIM_PROP* (`-M`) specifies the proportion of (randomly selected) SNPs to exclude from the analysis. The parameters *MIN_COVERAGE* (`-minc`) and *MAX_COVERAGE* (`-maxc`) specifies range of the mean sequencing depth (averaged across samples being analyzed) for loci to be included in the analysis. For example, the following command will first exclude loci given by the file "./example/snps2exclude.txt" and loci that have mean depth <0.01x or >4x, and then randomly exclude ∼30% of the remaining loci from the analysis:

```
./laser -ex ./example/snps2exclude.txt -minc 0.01 -maxc 4 -M 0.3
```

When *FIRST_IND* (`-x`) and *LAST_IND* (`-y`) are specified, the mean sequencing depth per locus will be calculated based on the specified range of samples. Please refer to Section 5.2 for more information regarding these parameters.

3.7 PCA mode

LASER can perform PCA on genotype data given by the *genotypefile*. To turn on the PCA mode, use the following command to change the parameter value of *PCA_MODE* (command line flag `-pca`) to 1:

```
./laser -p ./example/example.conf -pca 1
```

In PCA mode, *LASER* will perform PCA on the genotype data and then stop. Both coordinates of the top k PCs, where k is defined by the parameter *DIM* (command line flag `-k`), and the proportion of variance explained by each PC will be output. When memory is limited, you can also perform PCA by setting *PCA_MODE* to 2, which uses about half of the memory but is slow. If you want to know about the mean and standard deviation of the genotypic values, and the PC loadings (weights) of each locus, you can set *PCA_MODE* to 3 and the information will be output. When *PCA_MODE* is set to 1 or 2, *LASER* performs PCA based

on eigen value decomposition on an $N \times N$ sample covariance matrix (for N individuals). When *PCA_MODE* is set to 3, *LASER* performs PCA based on singlular value decomposition on an $N \times L$ standardized genotypic matrix (for N individuals and L loci).

We initially implemented this option to prepare the input *coordinatefile* for the main analysis of *LASER*. This option can also be used independently as a PCA tool to explore population structure based on genotype data. In terms of computational speed, *LASER* can finish PCA on the HGDP dataset (Li *et al.*, 2008), which includes SNP genotypes of 938 individual across 632,958 loci, in ∼15 minutes on a single processor (2.8GHz CPU).

4 Input files

In this section, we describe four input files that are taken by *LASER* — the *parameterfile*, the *genotypefile*, the *coordinatefile*, and the *sequencefile*. We also describe one additional file, the *sitefile*, which is required for preparing the input files.

4.1 *parameterfile* (*_.conf*)

The *parameterfile* contains all parameters required for running *LASER*. The default *parameterfile* is "laser.conf", which does not need to be explicitly specified in the command line (i.e. `./laser` is equivalent to `./laser -p laser.conf`). There are 18 parameters in the *parameterfile* , including 7 main parameters and 11 advanced parameters. Each parameter is followed by its assigned value, separated by whitespaces. Text in the same line after a '#' character is treated as comment and will not be read. For example, the following parameter specifications are equivalent in setting the parameter *DIM* equal to 4:

```
DIM 4
DIM   4 # Number of PCs to compute
DIM 4   # Other comments
```

If the user does not assign a value to a parameter in the *parameterfile*, this parameter must be followed by a '#' character (even without comments) to avoid unexpected errors in assigning other parameter values. An example *parameterfile* is provided in Section 3.1. To generate an empty template *parameterfile*, run *LASER* when the default *parameterfile* does not exist and without any command line arguments. Three parameters do not have default values, among which two parameters (*GENO_FILE* and *SEQ_FILE*) need to be explicitly defined by the users when in use, either in the *parameterfile* or in the command line (see Section 5.3), and one parameter (*COORD_FILE*) is optional. The other 15 parameters do not need to be explicitly defined unless the user wants to use settings different from the default. Please refer to Section 5 for more information on these parameters.

4.2 *genotypefile* (*_.geno*) and *sitefile* (*_.site*)

The *genotypefile* contains genotype data of the reference individuals. Each line represents genotype data of one individual. The first two columns represent population IDs and individual IDs, respectively. Starting from the third column, each column represent a locus. We only consider bi-allelic SNP markers. To be consistent with the sequence data, genotypes should be given on the forward strand. Genotypes are coded by 0, 1, or 2, representing copies of the reference allele at a locus in one individual. Missing data are coded by -9. Columns in the *genotypefile* are tab-delimited. An example *genotypefile* is provided below:

```
POP_1   IND_1    2    0    1    ...
POP_1   IND_2    2    -9   2    ...
POP_2   IND_3    0    0    -9   ...
POP_3   IND_4    1    2    1    ...
...     ...      ...  ...  ...  ...
```

Information on each locus, including chromosome number, genomic position, SNP ID, reference allele, and alternative allele, is listed in a separate *sitefile*. The reference allele and the alternative allele should be given on the forward strand. The *sitefile* is not required as input for running *LASER*, but it is needed for preparing the *sequencefile* (Section 4.3). The first row of the *sitefile* is the header line. Starting from the second line, each line represents one locus. Columns in the *sitefile* are tab-delimited. An example *sitefile* is provided below:

```
CHR  POS      ID          REF  ALT
1    752566   rs3094315   G    A
1    768448   rs12562034  G    A
1    1005806  rs3934834   C    T
...  ...      ...         ...  ...
```

We have prepared a tool called *vcf2geno* to covert a VCF file to a *genotypefile* and a *sitefile*. *vcf2geno* is a pre-compiled C executable program, which is also included in the download package. The command line for running the basic usage option of *vcf2geno* is

```
./vcf2geno --inVcf filename.vcf --out output
```

which will generate a *genotypefile* named "output.geno" and a *sitefile* named "output.site".

4.3 *sequencefile* (*_.seq*)

The *sequencefile* contains information of sequence reads that are piled up to the loci listed in the *sitefile* for the study samples. Each line represents one individual. Similar to the

genotypefile, the first two columns are population IDs and individual IDs. Starting from the third column, each locus is represented by three consecutive integer numbers. The first number indicates the total number of sequence reads that are mapped to the locus (coverage), the second number indicates the number of reads that correspond to the reference allele (specified in the *sitefile*) and are mapped to the locus, and the third number indicates the average base quality score for all bases mapped to the locus (in Phred scale). For the same locus, these three numbers are space-delimited. Columns for different loci and the first two columns are tab-delimited. Missing data are represented by "0 0 0" if one individual is not covered at a locus. (Note: the *sequencefile* for *LASER* version 1.03 or earlier does not include the average base quality scores.)

An example *sequencefile* is provided below:

```
POP_1   IND_1   2 1 20   0 0 0   15 5 22   ...
POP_1   IND_2   4 3 21   0 0 0   10 0 27   ...
POP_2   IND_3   2 2 30   0 0 0   0 0 0     ...
POP_3   IND_4   2 2 35   0 0 0   11 7 33   ...
...     ...     ...      ...     ...        ...
```

From version 2.03, the *LASER* program requires each input *sequencefile* to have an associated *sitefile* to provide locus information of the *sequencefile*. The loci in the *sequencefile* do not need to be identical to those in the *genotypefile*. The *LASER* program will automatically identify the shared loci for ancestry analysis.

We provide python scripts (included in the download package) to generate the *sequencefile* and the associated *sitefile* from BAM files. The procedure involves two steps: (1) generating pileup data from BAM files; and (2) generating *sequencefile* and *sitefile* from the pileup data.

For the first step, users can use the `mpileup` command in *samtools* (version 0.1.19) (Li *et al.*, 2009) to generate the pileup data. The following files are needed: BAM files for all study samples, a faidx-indexed reference sequence file in the FASTA format (*e.g.* hs37d5.fa), and a BED file that contains a list of sites where pileup should be generated. The BED file can be easily generated based on the *sitefile* using an `awk` command:

```
awk '{if(NR>1){print $1, $2-1, $2, $3;}}' ref.site > ref.bed
```

A typical *samtools* command to generate pileup data for a sample A is:

```
samtools mpileup -q 30 -Q 20 -f hs37d5.fa -l ref.bed A.bam > A.pileup
```

This command will generate a file named "A.pileup", which contains the pileup data for sample A. We recommend using "-q 30" to remove sequence reads with mapping quality score less than 30 (Phred scale) and using "-Q 20" to remove bases with base quality score less than

20 (Phred scale). The reference file "hs37d5.fa" is an integrated human reference sequence in NCBI Build 37. Users should make sure that genomic positions in the BAM files and the *sitefile* (and the corresponding BED file) are consistent with the reference sequence file, and are indexed consistently (0-based or 1-based). Note that the BED format uses 0-based index as the starting position. Therefore, for example, if your reference genotype data include SNP rs3094315, which locates at chromosome 1 position 752566 (Build 37, 1-based), the BED file should have the following line:

```
1  752565  752566  rs3094315
```

Pileup data for other samples can be generated using similar *samtools* commands.

After generating the pileup data for all study samples, users can use our *pileup2seq.py* script to prepare the *sequencefile*. In this step, users can provide a file (in BED format) that defines the target regions if they prefer to exclude on-target reads from downstream analysis of *LASER*. An example *pileup2seq.py* command looks like:

```
python pileup2seq.py -f hs37d5.fa -m ref.site -b target.bed \
-i example.id -o output A.pileup B.pileup C.pileup
```

This command will generate a *sequencefile* named "output.seq" that contains samples *A*, *B*, and *C*, and a *sitefile* named "output.site" that is the same as the "ref.site" file specified by `-m`. In the command, `-f` defines the human reference genome, `-m` defines the *sitefile*, `-b` defines the BED file for the target regions, and `-i` defines a sample ID file. The script will automatically check if the reference alleles given in the *sitefile* is consistent with the human reference genome. Both the `-b` flag and the `-i` flag are optional. When the `-b` flag is used, sequence reads that were piled up to loci within the target regions will be removed by setting the corresponding entry in the *sequencefile* as "0 0 0" (missing data). Running *pileup2seq.py* without the `-b` flag will generate a *sequencefile* that includes sequence reads from both on-target and off-target regions. The `-i` flag provides an option to use an alternative set of IDs listed in the file named "example.id", which has a format as given below:

```
A    POP_1   IND_1
B    POP_1   IND_2
C    POP_2   IND_3
...  ...     ...
```

With this ID file, the first two columns in the *sequencefile* output.seq will use the newly defined population IDs (*POP_1*, *POP_1*, *POP_2*, ...) and sample IDs (*IND_1*, *IND_2*, *IND_3*, ...). If the `-i` flag is not used, both of the first two columns in the *sequencefile* will use the pileup file names (*A*, *B*, *C*, ...).

4.4 *coordinatefile* (*_.coord*)

The *coordinatefile* contains PCA coordinates of the reference individuals. The first line is the header line. Starting from the second line, each line represent one individual. The first two columns correspond to population IDs and individual IDs respectively, and the following K columns represent the top K principal components (PCs). The order of the reference individuals must be the same as in the *genotypefile*. The *coordinatefile* is required to be tab-delimited. Below is an illustration of the format of the *coordinatefile*:

```
popID   indivID   PC1    PC2    PC3    ...
POP_1   IND_1     -3.5   0.2    0.7    ...
POP_1   IND_2     -2.2   4.5    0.8    ...
POP_2   IND_3     7.8    -0.8   -1.0   ...
POP_3   IND_4     1.6    -3.8   -0.4   ...
...     ...       ...    ...    ...    ...
```

Users can generate the *coordinatefile* by performing PCA on the *genotypefile* using the PCA mode of *LASER* (see Section 3.7). If the *coordinatefile* is not provided, *LASER* will automatically compute the reference coordinates based on the *genotypefile*. We recommend users to prepare a *coordinatefile* as input for *LASER* when submitting multiple jobs using the the same reference panel (so that the same computation will not be repeated for every job).

5 Usage options

LASER has 26 parameters that users can set in the *parameterfile*, including 7 main parameters that are required for running *LASER* and 20 advanced parameters for some special options. Among the 7 main parameters, 3 are parameters regarding the input data files and need to be explicitly defined when in use. The other 4 main parameters and the 20 advanced parameters have default values. In addition, *LASER* takes 28 command line arguments, which are described in Section 5.3.

5.1 Main parameters

GENO_FILE (string) The name of the *genotypefile*. If the file is not in the same directory as *LASER*, the whole path must be specified. This parameter must be explicitly defined.

SEQ_FILE (string) The name of the *sequencefile*. If the file is not in the same directory as *LASER*, the whole path must be specified. This parameter must be explicitly defined unless the parameter *PCA_MODE* is set to 1.

COORD_FILE (string) The name of the *coordinatefile*. If the file is not in the same directory as *LASER*, the whole path must be specified. This parameter is optional but recommended to be explicitly defined. If undefined, *LASER* will automatically compute the reference coordinates based on the *genotypefile*.

OUT_PREFIX (string) The prefix that will be added to the file names of outputting results. A path can be specified to output results to a different directory. The default value is "*laser*".

DIM (int) The number of PCs to compute (must be a positive integer). This number must be smaller than the number of individuals and the number of loci in the *genotypefile*, and cannot be greater than the number of PCs in the *coordinatefile* if a *coordinatefile* is provided. The default value is 2.

DIM_HIGH (int) Dimension of the sample-specific PCA map to project from (must be a positive integer). This number must be smaller than the number of individuals and the number of loci in the *genotypefile*, and cannot be smaller than *DIM*. *LASER* will project each study sample from a *DIM_HIGH* dimensional PC space to the *DIM* dimensional reference ancestry map. If set to 0, the program will use the number of significant PCs based on Tracy-Widom tests for each sample. The default value is 20.

MIN_LOCI (int) The minimum number of covered loci required for a sequence sample to be analyzed (must be a positive integer). If the number of covered loci in a sequence sample is smaller than *MIN_LOCI*, the sample will not be analyzed and results for this sample are output as "*NA*". The default value is 100.

5.2 Advanced parameters

MAX_LOCI (int) The maximum number of covered loci used for a sequence sample to be analyzed (must be a positive integer). If the number of covered loci in a sequence sample is larger than *MAX_LOCI*, the program will randomly select *MAX_LOCI* loci for the analysis. The default value is 1,000,000.

SEQ_ERR (double) The sequencing error rate per base (must be a number between 0 and 1, or -1). If set to -1, the program will use the estimated locus-specific error rates in the *sequencefile*. Otherwise, the program will use the specified error rate uniformly for all loci and all samples. The default value is -1.

ALPHA (double) Significance level in Tracy-Widom tests to determine the number of informative PCs, or *DIM_HIGH*, in the sample-specific PCA (must be a number between 0 and 1). This parameter is effective only if *DIM_HIGH* is undefined or set to 0. The default value is 0.1.

THRESHOLD (double) Convergence criterion of the projection Procrustes analysis (must be a positive number). The default value is 0.000001.

FIRST_IND (int) The index of the first sequence sample to analyze (must be a positive integer). This number cannot be greater than the number of individuals in the *sequencefile*. Samples that have indices smaller than *FIRST_IND* will be skipped. The default value is 1.

LAST_IND (int) The index of the last sequence sample to analyze (must be a positive integer). This number cannot be greater than the number of samples in the *sequencefile* or smaller than *FIRST_IND*. Samples that have indices greater than *LAST_IND* will be skipped. The default value is the number of samples in the *sequencefile*.

REPS (int) The number of repeated runs in analyzing each sequence sample (must be a positive integer). The default value is 1.

OUTPUT_REPS (int) This parameter specifies whether to output results from each repeated run when REPS is greater than 1 (must be 0 or 1). A value of 0 will only output mean and standard deviation of the estimated coordinates for each sample across all repeated runs. A value of 1 will also output the estimated coordinates obtained from each repeated run. The default value is 0.

CHECK_COVERAGE (int) This parameter specifies whether to check the sequencing coverage across samples and across loci in the *sequencefile* (must be 0, 1, or 2). A value of 0 will not check the coverage, and proceed to the major computation. A value of 1 will check the coverage, and proceed to the major computation. A value of 2 will check the coverage, and stop. The default value is 0.

CHECK_FORMAT (int) This parameter specifies whether to check the format of the input files (must be 0, 1, 2, 3, 4, 10, 20, 30, or 40). A value of 0 will not check format of the input files, and proceed to the major computation. A value of 1 will check the format of all input files, and stop. A value of 2 will check the format of the *genotypefile*, and stop. A value of 3

will check the format of the *sequencefile*, and stop. A value of 4 will check the format of the *coordinatefile*, and stop. A value of 10, 20, 30, or 40 will check the corresponding file(s) as a value of 1, 2, 3, or 4 does, and proceed to the major computation. The default value is 10.

PCA_MODE (int) This parameter specifies whether to turn on the PCA mode (must be 0, 1, 2, or 3). A value of 0 will not turn on the PCA mode and the program will perform the full function to estimate individual ancestry from sequence data. A value of 1, 2, or 3 will switch to the PCA mode and the program will only perform PCA on the reference genotype data. A value of 1 or 2 will perform PCA based on eigen value decomposition on the sample covariance matrix, and will output the PC coordinates and the proportion of variance explained by each PC. The program uses less memory when using option 2 but the computational time is much longer. A value of 3 will perform PCA based on singular value decomposition on the standardized genotypic matrix, and will output the mean, standard deviation, and PC loadings of each locus (in addtion to the PC coordinates). When this parameter is set to 1, 2, or 3, *COORD_FILE* and *SEQ_FILE* do not need to be defined. The default value is 0.

REF_SIZE (int) Number of individuals randomly selected from *GENO_FILE* as the reference panel (must be a positive integer). This number cannot be greater than the number of samples in the *GENO_FILE*. This option is useful when the data size is too big such that memory limit becomes an issue. The default value is the sample size in the *GENO_FILE* (i.e., using all reference individuals).

TRIM_PROP (double) Proportion of randomly selected loci to exclude from the analysis for all samples (must be a number between 0 and 1). This option is useful when the data size is too big such that memory limit becomes an issue. The default value is 0.

EXCLUDE_LIST (string) This parameter specifies the file name of a list of SNPs to exclude from the analysis. Each line of the file is a SNP ID and no header is required. If the file is not in the same directory as *LASER*, the whole path must be specified. This parameter do not have default value.

MIN_COVERAGE (double) This parameter specifies the minimum mean depth (averaged across samples from *FIRST_IND* to *LAST_IND*) for sites to be included in the analysis (must be a non-negative number). This parameter is useful to exclude sites that have little data to save the memory usage (e.g., a value of 0.01 means loci that have ¡0.01x depth will be excluded). The default value is 0.

MAX_COVERAGE (double) This parameter specifies the maximum mean depth (averaged across samples from *FIRST_IND* to *LAST_IND*) for sites to be included in the analysis (must be a positive number or -1). This parameter is useful to exclude targeted regions in targeted/exome sequencing data (e.g., a value of 4 means loci that have ¿4x depth will be excluded). When set to -1, all sites will be included in the analysis. The default value is -1.

PROCRUSTES_SCALE (int) This parameter specifies how to scale coordinates in the Procrustes analysis (must be 0 or 1). When set to 0, the Procrustes analysis will include the scaling factor as a free parameter and estimates its value to minimize the sum of squared Euclidean distances between two sets of coordinates in the analysis. When set to 1, the Procrustes analysis will fix the scaling factor so that two sets of coordinates will have the same total variance (i.e. Procrustes analysis only search for rotation, reflection and translation to minimize the sum of squared Euclidean distances). The default value is 0.

KNN_ZSCORE (int) This parameter specifies the number of nearest neighbors used to calculate the Z score for each study individual (must an integer > 2). The default value is 10.

RANDOM_SEED (int) This parameter specifies the seed for the random number generator used the program (must a non-negative integer). The default value is 0.

NUM_THREADS (int) This parameter specifies the number of CPU cores for multi-threading parallel analysis (must a positive integer). The default value is 8.

5.3 Command line arguments

The command line flags provide the user an option to enter information from the command line. All command line arguments will overwrite values specified in the *parameterfile*. If a parameter is specified with an invalid value in the *parameterfile* but a valid value in the command line, the program will return a warning message and still execute correctly by taking the value from the command line. However, if a parameter value in the command line is not valid, the program will exit with an error message. If a command line flag is specified, it must be followed by a space and then the parameter value. Different command line flags can appear in any order. If the same command line flag is defined more than once, only the last value will be taken. For example, the following command lines are equivalent and will change the value of the parameter *DIM*, for which the command line flag is `-k`, to be 4 while using the other parameters defined in the *parameterfile* named "my_parameterfile".

```
./laser -p my_parameterfile -k 4
./laser -k 4 -p my_parameterfile
```

```
./laser -k 3 -p my_parameterfile -k 4
```

Most command line arguments are optional except for the *parameterfile*, for which the command line flag is `-p`. A list of all command line flags is provided below.

-p This flag defines the *parameterfile*. If the *parameterfile* is not in the current directory, a whole path to the file must be specified. This parameter can only be defined using the command line. If undefined, the program will use the default *parameterfile* named "laser.conf" in the current directory. If this file does not exist, an empty template *parameterfile* named "laser.conf" will be created in the current directory, and the program will then exit with an error message.

-g Change the parameter value of *GENO_FILE*.

-s Change the parameter value of *SEQ_FILE*.

-c Change the parameter value of *COORD_FILE*.

-o Change the parameter value of *OUT_PREFIX* (useful when running parallel jobs).

-k Change the parameter value of *DIM*.

-K Change the parameter value of *DIM_HIGH*.

-l Change the parameter value of *MIN_LOCI*.

-L Change the parameter value of *MAX_LOCI*.

-e Change the parameter value of *SEQ_ERR*.

-a Change the parameter value of *ALPHA*.

-t Change the parameter value of *THRESHOLD*.

-x Change the parameter value of *FIRST_IND* (useful when running parallel jobs).

-y Change the parameter value of *LAST_IND* (useful when running parallel jobs).

-r Change the parameter value of *REPS*.

-R Change the parameter value of *OUTPUT_REPS*.

-cov Change the parameter value of *CHECK_COVERAGE*.

-fmt Change the parameter value of *CHECK_FORMAT*.

-pca Change the parameter value of *PCA_MODE*.

-N Change the parameter value of *REF_SIZE*.

-M Change the parameter value of *TRIM_PROP*.

-ex Change the parameter value of *EXCLUDE_LIST*.

-minc Change the parameter value of *MIN_COVERAGE*.

-maxc Change the parameter value of *MAX_COVERAGE*.

-rho Change the parameter value of *PROCRUSTES_SCALE*.

-knn Change the parameter value of *KNN_ZSCORE*.

-seed Change the parameter value of *RANDOM_SEED*.

-nt Change the parameter value of *NUM_THREADS*.

6 Output files

All output files will be saved in the current directory unless the path to a different directory is given in the parameter value of *OUT_PREFIX*. All output file names will start with the parameter value of *OUT_PREFIX*. These files are described below.

6.1 _*.log* and terminal outputs

The terminal outputs are used to monitor and record the progress when running *LASER*. It starts with all parameter values used in the execution of *LASER*, and reports the progress of

the program step by step. The *log* file is identical to the terminal outputs.

6.2 *_.RefPC.coord* and *_.RefPC.var*

When *COORD_FILE* is not defined or the PCA mode is on (*PCA_MODE*=1, 2, or 3), *LASER* will perform PCA on the reference genotype data given by the *genotypefile*. Results of the top k PCs, where k is defined by the parameter *DIM*, are output to two files named *OUT_PREFIX.RefPC.coord* and *OUT_PREFIX.RefPC.var*.

The *RefPC.coord* file records the PCA coordinates of the reference individuals. The first line in this file is a header line. Starting from the second line, each line represents one individual. The first two columns are population ID and individual ID, respectively. The remaining columns correspond to the top k PCs. This file is tab-delimited. The format of this file is exactly the same as the *coordinatefile* (Section 4.4), so that this file can be directly used as the input file for *LASER*.

The *RefPC.var* file records the proportion of variance explained by each PC. The first line in this file is a header line. Starting from the second line, each line represents one PC. The first column is the PC index and the second column is the percentage of variance explained by each PC. Only results for the top k PCs are output. This file is tab-delimited.

6.3 *_.RefPC.grm*

When *PCA_MODE* is set to 1, *LASER* will perform PCA on the reference genotypes by applying eigen value decomposition on the sample covariance matrix, which is defined as $M = XX^T$, where X is an $N \times L$ standardized genotype matrix for N individuals and L loci. Besides the coordinates and proportion of variances of each PC, *LASER* also outputs an $N \times N$ genetic relationship matrix a file named *OUT_PREFIX.RefPC.grm*. The genetic relationship matrix, defined as $M' = M/L$, is often used in linear mixed models to model the correlation structure of phenotypes in genetic association tests and therefore to control for population stratification.

The *RefPC.grm* file does not have header lines. This file records a $N \times N$ matrix in N lines and each line is tab-delimited. N individuals are ordered as the input genotype file. This file follows the same format as that used in *GEMMA*, which is a program for genetic association tests based on linear mixed models (Zhou and Stephens, 2012).

6.4 *_.RefPC.load*

When *PCA_MODE* is set to 3, *LASER* will perform PCA on the reference genotypes based on singular value decomposition. The PC loadings, which are sometimes called SNP weights in genetic literature, will be output to the *_.RefPC.load* file.

The first line of the *RefPC.load* file is a header line. Starting from the second line, each line represents one locus. The first column is the locus ID. The second and the third column reports the mean and standard deviation of the genotypic values across all individuals in the *genotypefile*. These two values are used to standardize each locus in the PCA. Starting from the fourth column, each column reports the loadings of one PC. The total number of PCs are specified by the parameter *DIM* (`-k`). This file is tab-delimited.

6.5 *_.SeqPC.coord*

This file contains the estimated PCA coordinates of the sequence samples. The first line is a header line. Starting from the second line, each line represents one study sample. The first column is population ID, and the second column is individual ID. The third column records the number of covered loci (*i.e.* loci that have at least one read) in each individual. The fourth column is the mean coverage in each individual. The fifth column reports values of *DIM_HIGH*, dimension of the sample-specific PCA map used in the projection Procrustes analysis. The sixth column is the Procrustes similarity score between the PCA map constructed from the simulated sequence reads of the sample-specific reference panel (after projection) to the original reference PCA map based on SNP genotypes. Description of this statistic can be found in Wang *et al.* (2014). Briefly, this statistic ranges from 0 and 1, with higher values indicating higher confidence of the estimated coordinates the study sample. The seventh column reports the Z score for each individual, which reflects if the individual's ancestry is reprented in the reference panel. If Z score is much different from 0 (e.g. $Z > 4$), it indicates that the individual's ancestry might not be represented in the reference panel. The definition of Z score can be found in (Taliun *et al.*, 2017). Starting from the eighth column, each column represents coordinates of one PC (up to the kth PC, where k is defined by *DIM*). Columns in this file are tab-delimited.

When the parameter *REPS* is greater than 1, multiple runs of *LASER* are applied to analyze each study sample. The Procrustes similarity scores and the PC coordinates reported in the *SeqPC.coord* file are the mean values across results from all repeated runs.

6.6 *_.SeqPC.coord.sd* and *_.SeqPC.coord.reps*

These two files are generated when the parameter value of *REPS* is greater than 1. To generate the *SeqPC.coord.reps* file, the parameter *OUTPUT_REPS* should be set to 1.

The *SeqPC.coord.sd* file contains the standard deviation of results across all repeated runs of *LASER* on the same set of samples. The first line is a header line. Starting from the second line, each line represents one study sample. The first two columns are population ID and individual ID, respectively. The third column is the standard deviation of the Procrustes

similarity score. The fourth column is the standard deviation of the Z score. Starting from the fifth column, each column contains the standard deviation of one corresponding PC (up to the kth PC, where k is defined by *DIM*). Columns in this file are tab-delimited.

The *SeqPC.coord.reps* file contains results from all repeated runs. The format of this file is the same as the *SeqPC.coord* file, except that each study sample is now represented by *REPS* consecutive lines. Each line records the results from one repeated run of *LASER* on the sample.

6.7 *_.ind.cov* and *_.loc.cov*

These two files contain the estimated values of sequencing coverage across samples and across loci, respectively. These two files are generated when the parameter *CHECK_COVERAGE* is set to 1 or 2, or when *MAX_COVERAGE* is specified to a positive number.

For the *ind.cov* file, the first line is the header line and there are four tab-delimited columns. The first column is population ID; the second column is individual ID; the third column provides the number of covered loci (*i.e.* loci that have at least one read) in each individual; and the fourth column is the mean coverage in each individual.

For the *loc.cov* file, the first line is the header line and there are three tab-delimited columns. The first column is the locus ID; the second column provides the number of covered individuals (*i.e.* individuals that have at least one read) at each locus; and the third column is the mean coverage at each locus. Loci in this file are listed in the same order as the *sitefile*, where position and allele information of each locus can be found.

7 Computational complexity

LASER examines one sequence sample at a time. Therefore, the computational costs increase linearly with the number of samples to be analyzed. We can easily run the analyses in parallel by submitting multiple jobs to analyze different subsets of samples (using command line flags `-x` and `-y`). The cost for analysis of each sample depends on the number of individuals, N, and markers, L, in the reference panel and the fraction of loci with nonzero coverage, λ, in the sequence sample. Roughly, we expect computational cost for each sample to be $O(N^2L\lambda+N^3)$, which is the time required to compute the $N \times N$ pairwise similarity matrix of the sample specific reference panel and the corresponding eigen value decomposition. In our simulations (Wang *et al.*, 2014), analysis typically required no more than a few minutes per sample (*e.g.*, $\sim$1.3 minutes when $N = 1000$, $L \approx 319K$, and $\lambda \approx 0.2$). The evaluation was based on version 1.03 of the LASER program, which was linked with the Armadillo library version 3.909.1 and standard LAPACK and BLAS libraries. When using more advanced C libraries, such as the OpenBLAS, the program is expected to run faster.

In cases where computational resource is limited, users can speed up LASER by using a reference panel of fewer individuals (smaller N). We do not recommend reducing the number of loci L, because estimates of ancestry might not be accurate if the number of sequencing reads used by *LASER* is small, especially in the setting of extremely low coverage sequencing.

8 Version changes

Changes from previous versions of the *LASER* software are noted here.

8.1 Version 1.0 (Feb 1, 2013)

- Initial release of the *LASER* software.

8.2 Version 1.01 (Mar 11, 2013)

- Fixed a bug that might cause unexpected errors when running many jobs simultaneously.
- Minor changes to the log file; outputting computational time when the program exits.

8.3 Version 1.02 (Jun 19, 2013)

- Changed the program to be more memory efficient.
- Modified the *vcf2geno* tool to allow using alternative population and individual IDs.

8.4 Version 1.03 (Aug 8, 2013)

- Upgraded the program to use version 3.909.1 of the Armadillo Linear Algebra Library (faster and more memory efficient).
- Fixed a major bug in the previous version of the *vcf2geno* tool.
- Minor changes to the *pileup2seq.py* script.

8.5 Version 2.0 (May 19, 2014)

- Upgraded *LASER* to use projection Procrustes analysis to improve ancestry estimation (added new parameters *DIM_HIGH*, *ALPHA* and *THRESHOLD*; minor changes to the format of the output coordinate file *_.SeqPC.coord*).
- Upgraded *LASER* to use locus-specific base quality scores specified in the *sequencefile* (made changes to the parameter *SEQ_ERR* and the format of the *sequencefile*).

- Added new parameters *EXCLUDE_LIST* and *TRIM_PROP* to allow users to exclude a given list of SNPs or a fraction of randomly selected SNPs from the analysis.

- Added new parameters *MIN_COVERAGE* and *MAX_COVERAGE* to allow users to exclude loci that have mean depth smaller or greater than a certain level.

- Changed the *LASER* program to be memory efficient (version 2.0 uses much less memory than previous versions).

- Upgraded the *pileup2seq.py* script to be compatible with *samtools* version 0.1.19, and to output locus-specific error rates in the *sequencefile*.

- Included a new software program *TRACE* in the package. *TRACE* can estimate individual ancestry using genotype data under the same framework as *LASER*. Please see the document "TRACE_Manual" for details.

- Added example R codes to generate figures for results that use the HGDP data as the reference panel (including a list of colors and symbols for 53 HGDP populations).

8.6 Version 2.01 (Jun 5, 2014)

- Added an option to perform PCA based on singular value decomposition and output the PC loadings (*i.e.*, SNP weights).

- Made a minor change to the output coverage file *_.loc.cov* to report SNP IDs.

8.7 Version 2.02 (Apr 6, 2015)

- Added an option to output the genetic relationship matrix of the reference individuals (which is the sample covariance matrix used in PCA scaled by the number of loci).

- Added a new parameter *PROCRUSTES_SCALE* to allow scaling two sets of coordinates to have the same variance in the Procrustes analysis.

- Fixed a minor bug to avoid the possibility of using the same seed in the random number generator for repeated runs.

- Fixed a minor bug to in the projection Procrustes anlaysis (convergence issue, no impact on the results).

8.8 Version 2.03 (Feb 25, 2016)

- Added a requirement that the input *SEQ_FILE* should have an associated *sitefile* to specify the locus information. The *LASER* program will automatically identify the shared loci between *SEQ_FILE* and *GENO_FILE* for downstream analyses.

- Added a new parameter *NUM_REF* to allow using a random subset of individuals in the *GENO_FILE* as the reference panel. (use less memory and run faster)

- Added a new parameter *RANDOM_SEED* to allow users to specify the random seed.

- Changed the *pileup2seq.py* script to check the strand of the input *sitefile* by comparing with the reference genome. (require the reference genome as an input for *pileup2seq.py*)

- Changed the *pileup2seq.py* script to generate an associated *sitefile* for the *sequencefile*.

8.9 Version 2.04 (Jan 11, 2017)

- Implemented a new statistic Z to indicate if a study individual's ancestry is represented in the reference panel.

- Added a new parameter *KNN_ZSCORE* to specify the number of nearest neighbors used to calculate the Z score.

- Added a new parameter *MAX_LOCI* to specify the maximum number of sequenced loci used for LASER analysis.

- Linked the program with the OpenBLAS library instead of BLAS and LAPACK.

- Added a new parameter *NUM_THREADS* to specify the number of CPU cores for multithreaded matrix computation (a feature in OpenBLAS).

9 Acknowledgements

We would like to thank Conrad Sanderson at the National ICT Australia for his help on using the Armadillo library. Author contributions: Chaolong Wang wrote the ancestry estimation programs (*LASER* and *TRACE*); Xiaowei Zhan wrote the companion tools for preparing input files (*vcf2geno* and *pileup2seq*).

References

Gower, J. C., and G. B. Dijksterhuis, 2004 *Procrustes Problems*. Oxford University Press.

Li, H., B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin and 1000 Genome Project Data Processing Subgroup, 2009 The Sequence Alignment/Map format and SAMtools. Bioinformatics **25:** 2078–2079.

Li, J. Z., D. M. Absher, H. Tang, A. M. Southwick, A. M. Casto, S. Ramachandran, H. M. Cann, G. S. Barsh, M. Feldman, L. L. Cavalli-Sforza and R. M. Myers, 2008 Worldwide human relationships inferred from genome-wide patterns of variation. Science **319:** 1100–1104.

Sanderson, C., and R. Curtin, 2016 Armadillo: a template-based C++ library for linear algebra. Journal of Open Source Software **1:** 26.

Taliun, D., S. Chothani, S. Schönherr, L. Forer, M. Boehnke, G. R. Abecasis and C. Wang, 2017 LASER server: ancestry tracing with genotypes or sequence reads. Bioinformatics **doi:** 10.1093/bioinformatics/btx075.

Wang, C., Z. A. Szpiech, J. H. Degnan, M. Jakobsson, T. J. Pemberton, J. A. Hardy, A. B. Singleton and N. A. Rosenberg, 2010 Comparing spatial maps of human population-genetic variation using Procrustes analysis. Stat. Appl. Genet. Mol. Biol. **9:** Article 13.

Wang, C., X. Zhan, J. Bragg-Gresham, D. Stambolian, E. Chew, K. Branham, J. Heckenlively, R. S. Fulton, R. K. Wilson, E. R. Mardis, X. Lin, A. Swaroop, S. Zöllner and G. R. Abecasis, 2014 Ancestry estimation and control of population stratification for sequence-based association studies. Nature Genetics **46:** 409–415.

Wang, C., X. Zhan, L. Liang, G. R. Abecasis and X. Lin, 2015 Improved ancestry estimation for both genotyping and sequencing data using projection Procrustes analysis and genotype imputation. American Journal of Human Genetics **96:** 926–937.

Zhou, X., and M. Stephens, 2012 Genome-wide efficient mixed-model analysis for association studies. Nature Genetics **44:** 821–824.