LASER: Locating Ancestry from SEquencing Reads
version 1.0

Chaolong Wang[1]
Department of Biostatistics
School of Public Health
Harvard University

Xiaowei Zhan[2]
Department of Biostatistics
School of Public Health
University of Michigan

January 5, 2013

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

---

[1]Comments on the *LASER* software can be sent to cwang@hsph.harvard.edu.
[2]Comments on tools for preparing the input files can be sent to zhanxw@umich.edu.
[3]This software is distributed "as is" without warranty of any kind.

Contents

1 Introduction

LASER is a C++ software program for analyzing next generation sequencing data. *LASER* can estimate individual ancestry directly from genome-wide shortgun sequencing reads without calling genotypes. The *LASER* method relies on the availability of a set of reference individuals whose genome-wide SNP genotypes and ancestral information are known. We first construct a reference coordinate system by applying principal components analysis (PCA) to the genotype data of the reference individuals. Then, for each sequence sample, *LASER* uses the genome-wide sequencing reads to place the sample into the reference PCA space. With an appropriate reference panel, the estimated coordinates of the sequence samples identify their ancestral background and can be directly used to correct for population structure in association studies or to ensure adequate matching of cases and controls.

To place each sample, we proceed as follows. First, we simulate sequencing 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 studied sample). Then, we build a PCA ancestry map based on these simulated sequencing reads for the reference samples together with the real sequencing 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 studied sample until all samples are mapped to the reference PCA space. For detailed information about the method, please refer to our paper entitled "Estimating individual ancestry using next generation sequencing" (WANG *et al.*, 2013).

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.*, 2013). We expect this method to be particularly useful for studies based on targeted sequencing technologies, in which genetic variation within targeted 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 sequencing 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 always recommend starting with a worldwide reference panel and gradually focusing on more regional reference panels.

In addition to estimating individual ancestry from sequencing reads, *LASER* also provides an option to perform 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

Pre-compiled executables for *LASER* for Linux (64-bit) system can be downloaded from the following webpage: http://www.sph.umich.edu/csg/abecasis/software.html.

Source code written in C++ is available upon request. The *LASER* program uses two external C++ libraries: the Armadillo Linear Algebra Library (http://arma.sourceforge.net) (SANDERSON, 2010) and the GNU Scientific Library (http://www.gnu.org/software/gsl). The Armadillo library requires two additional libraries: LAPACK and BLAS. Therefore, to compile from source code, you need to have these four libraries installed in your computer.

Please use the following citation for *LASER*: **Wang et. al. (2013). Estimating individual ancestry using next generation sequencing. (in preparation)**

2.2 Installing *LASER*

Open a terminal in the same directory as the `.tar.gz` file. Extract the file by typing `tar -xzvf LASER-1.0.tar.gz` in the terminal. This will create a new directory called `LASER-1.0` containing the executable and other resource files.

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-1.0`. 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, and the program will then exit with an error message. 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 HapMap samples (THE INTERNATIONAL HAPMAP 3 CONSORTIUM, 2010), 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 number of PCs (*DIM*) to 10, the prefix of output file names (*OUT_PREFIX*) to "HapMap.example", and the seqeuncing error rate to 0.0075 per base, while the other parameters are defined by the *example.conf* file.

```
./laser -p ./example/example.conf -o HapMap.example -k 10 -e 0.0075
```

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 "#" if the parameter is undefined). Texts after the "#" in each line are comments, which will not be read by the program.

```
# This is a parameter file for LASER v1.0.
# 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
MIN_LOCI      100                                   # default 100
SEQ_ERR       0.01                                  # default 0.01
###----Advanced Parameters----###
```

```
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
###----Command line arguments----###
# -p   parameterfile (this file)
# -g   GENO_FILE
# -s   SEQ_FILE
# -c   COORD_FILE
# -o   OUT_PREFIX
# -k   DIM
# -l   MIN_LOCI
# -e   SEQ_ERR
# -x   FIRST_IND
# -y   LAST_IND
# -r   REPS
# -R   OUTPUT_REPS
# -fmt CHECK_FORMAT
# -cov CHECK_COVERAGE
# -pca PCA_MODE
###----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

We currently do not implement multi-thread option to run 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.*, 2013). 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:

6

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

Please refer to Section 6.4 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 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. 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 $\sim$25 minutes on a single node of a cluster (64-bit Linux platform).

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 11 main parameters and 7 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 *paramfile*, this parameter must be followed by a '#' character (even without comments) to avoid unexpected errors in assigning other parameter values. To generate an empty template *paramfile*, run *LASER* when the default *parameterfile* does not exist and without any command line arguments. Seven parameters do not have default values, among which five parameters (*GENO_FILE*, *SEQ_FILE*, *REF_INDS*, *SEQ_INDS*, *LOCI*) need to be explicitly defined the users when in use, either in the *parameterfile* or in the command line (see Section 5.3), and two parameters (*CO-ORD_FILE*, , and *NUM_PCS*)are optional. The other 11 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
...  ...      ...         ...  ...
```

Users can generate the *genotypefile* and the *sitefile* from a VCF file using a small tool called *vcf2laser*, which is also included in the dowload package. The command for running *vcf2laser* is `./vcf2laser --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 sequencing 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 two consecutive integer numbers. The first number indicates the total number of sequencing reads that are mapped to the locus (coverage), and the second number indicates the number of reads that correspond to the reference allele (specified in the *sitefile*) and are mapped to the locus. For the same locus, these two numbers are space-delimited. Columns for different loci and the first two columns are tab-delimited. Missing data are represented by "0 0" if one individual is not covered at a locus. An example *sequencefile* is provided below:

```
POP_1  IND_1  2 1  0 0  15 5  ...
POP_1  IND_2  4 3  0 0  10 0  ...
POP_2  IND_3  2 2  0 0  0 0   ...
POP_3  IND_4  2 2  0 0  11 7  ...
...    ...    ...  ...  ...   ...
```

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

For the first step, users can use the `mpileup` command in *samtools* (version 0.1.18) (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 '{print $1, $2-1, $2, $3;}' reference.site > reference.bed
```

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

```
samtools mpileup -q 30 -Q 20 \
-f hs37d5.fa -l reference.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 all mapped read 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' 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 on 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 *pileup2laser.py* script to prepare the *sequencefile*. In this step, users need to provide a file (in BED format) that defines the target regions of the seqeuncing data if downstream analysis of *LASER* will only focus on off-target sequencing reads. An example *pileup2laser.py* command looks like:

```
python pileup2laser.py -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*. In the command, `-m` defines the *sitefile*, `-b` defines the BED file for the target regions, and `-i` defines a sample ID file. 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" (missing data). Running *pileup2laser.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 defined by the ID 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 space-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 3.6). 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 14 parameters that users can set in the *paramfile*, including seven main parameters that are required for running *LASER* and seven advanced parameters for some special

options. Among the seven main parameters, three are parameters regarding the input data files and need to be explicitly defined when in use. The other four main parameters and the seven advanced parameters have default values. In addition, *LASER* takes 15 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.

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.

SEQ_ERR (double) The sequencing error rate per base (must be a number between 0 and 1). The default value is 0.01.

5.2 Advanced parameters

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 index 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 index 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_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.

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.

PCA_MODE (int) This parameter specifies whether to turn on the PCA mode (must be 0 or 1). A value of 0 will not turn on the PCA mode and the program will perform the full function to estimate individual ancestry from sequencing data. A value of 1 will switch to the PCA mode and the program will only perform PCA on the reference genotype data. When

this parameter is set to 1, *COORD_FILE* and *SEQ_FILE* do not need to be defined. The default value is 0.

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*.

-l Change the parameter value of *MIN_LOCI*.

-e Change the parameter value of *SEQ_ERR*.

-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*.

-fmt Change the parameter value of *CHECK_FORMAT*.

-cov Change the parameter value of *CHECK_COVERAGE*.

-pca Change the parameter value of *PCA_MODE* (useful when preparing the *coordinatefile*).

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), *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. 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 _.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 is the Procrustes similarity score between the PCA map constructed from the simulated sequencing reads of the sample-specific reference panel to the original reference PCA map based on SNP genotypes. Description of this statistic can be found in WANG *et al.* (2013). Briefly, this statistic ranges from 0 and 1, with higher values indicating higher confidence of the estimated coordinates the study sample. Starting from the sixth column, each column represent coordinates of one PC (up to the *k*th 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.

6.4 _.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. Starting from the fourth column, each column contains the standard deviation of one corresponding PC (up to the *k*th 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.

6.5 _.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.

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 two tab-delimited columns. The first column provides the number of covered individuals (*i.e.* individuals that have at least one read) at each locus; and the second column is the mean coverage at each locus. Loci in this file are listed in the same order as the *sitefile*, where locus IDs 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 covariance matrix of the sample specific reference panel and the corresponding eigen decomposition. In our simulations (WANG *et al.*, 2013), analysis typically required no more than a few minutes per sample (e.g., ∼1.7 minutes when $N = 1000$, $L \approx 319K$, and $\lambda \approx 0.2$).

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 estimate 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   Acknowledgements

References

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.

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., 2010   Armadillo: an open source C++ linear algebra library for fast protoyping and computationally intensive experiments. Technical Report, NICTA.

The International HapMap 3 Consortium, 2010   Integrating common and rare genetic variation in diverse human populations. Nature **467:** 52–58.

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. Y. Chew, R. Fulton, R. K. Wilson, E. R. Elaine, A. Swaroop, S. Zöllner and G. R. Abecasis, 2013   Estimating individual ancestry using next generation sequencing. (In preparation).