TRACE:
fasT and Robust Ancestry Coordinate Estimation
version 1.0

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

May 19, 2014

The *TRACE* software[2] is available at
http://www.sph.umich.edu/csg/chaolong/LASER/

[1]Comments on the *TRACE* software can be sent to chaolong@umich.edu.

Contents

1 Introduction

TRACE is a software program that uses SNP genotypes to *trace* individual ancestry by comparing to a set of reference individuals. The basic idea is to construct a reference ancestry map by applying principal components analysis (PCA) on genotypes of the reference individuals, and then to place the study samples one by one into this reference ancestry map. With an appropriate reference panel, the estimated coordinates of the study samples are informative on their ancestry background. To place each sample, we use a two-step approach involving PCA and projection Procrustes analysis (Gower and Dijksterhuis, 2004; Wang *et al.*, 2010). The *TRACE* program follows the same framework as the *LASER* program, which we previously developed to estimate individual ancestry by directly analyzing shortgun reads from next generation sequencing without calling genotypes (Wang *et al.*, 2014a). Different from *LASER*, *TRACE* takes genotype data as the input, and thus can benefits studies when sequence read data are not available.

TRACE can identify population structure in large cohorts. Compared to standard PCA, *TRACE* has several advantages in terms of robustness and computational efficiency. First, *TRACE* is robust to missing data in the study samples. Standard PCA requires high quality genotypes with low missing data rate for the study samples. Commonly used PCA software programs such as *smartpca* (Patterson *et al.*, 2006) often impute missing data with mean values at the corresponding loci of the genotype matrix. Consequently, samples that have more missing data will shrink closer to the center of the PCA map. *TRACE*, in contrast, only requires low missingness in the reference individuals and can appropriately handle missing data in the study samples, which makes *TRACE* applicable to low quality data such as those obtained from ancient DNA samples (Skoglund *et al.*, 2012). Second, *TRACE* is robust to the presence of close genetic relatedness between study samples. If PCA is applied direclty on all study samples, closely related individuals often appear as outliers, and might distort the overall PCA pattern. *TRACE* avoids this problem by analyzing each sample independently with the reference panel, which only requires the reference individuals to be unrelated with each other and with the study samples. For the same reason, *TRACE* is also robust to uneven sampling scheme and existence of genetic outliers in the study samples, which are known to have large impacts on PCA results (McVean, 2009; Lee *et al.*, 2010). Last, the computational time of *TRACE* increases linearly with the number of study samples, so that *TRACE* can be much faster than standard PCA when the sample size is large.

Details of the method and examples illustrating the aforementioned advantages of *TRACE* can be found in our paper entitled " Improved ancestry estimation for both genotyping and sequencing data using projection Procrustes analysis and genotype imputation" (Wang *et al.*, 2014b).

2 Getting started

2.1 Availability

The *TRACE* program is distributed as part of the *LASER* software package (Wang *et al.*, 2014a), which can be downloaded from the following webpage: http://www.sph.umich.edu/csg/chaolong/LASER/. In the package, we provide a pre-compiled executable for *TRACE* for Linux (64-bit) operation systems. 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 *TRACE* 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 cite the following papers when you use *TRACE*:

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

2.2 Installing *TRACE*

Open a terminal in the same directory as the `.tar.gz` file. Extract the file by typing `tar -xzvf LASER-2.0.tar.gz` in the terminal. This will create a new directory called `LASER-2.0`. This direcotry contains executables for both *LASER* (version 2.0) and *TRACE* (version 1.0), and other resource files.

2.3 Running *TRACE*

Open a terminal and path to the directory that contains the executable *TRACE*. If you did not rename the directory after extracting the `.tar.gz` file, the directory will be `LASER-2.0`. Execute the program by typing `./trace -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, *TRACE* will search in the current directory for a *parameterfile* named "trace.conf", and execute the program with parameter values specified in "trace.conf". If this file does not exist, an empty template *parameterfile* named "trace.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 *TRACE* 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).

The "example" folder contains genotype data across 9,608 SNPs on chromosome 22 for 938 individuals from the Human Genome Diversity Panel (HGDP, Li *et al.*, 2008). The HGDP data are splited into two files named *HGDP_700_chr22.geno* and *HGDP_238_chr22.geno*, containing 700 and 238 individuals, respectively. The *HGDP_chr22.site* file provides detailed information of the 9,608 SNP markers. The *HGDP_238_chr22.RefPC.coord* file contains PCA coordinates for the top 8 PCs based on genotypes in the *HGDP_238_chr22.geno* file. The folder also includes a *parameterfile* named "example.conf", which specifies parameters for running *TRACE* on the example data. In the following examples, we will use the subset of 238 individuals to construct the reference PCA space and use *TRACE* to place the remaining 700 individuals into the reference PCA space.

3.1 Basic usage

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

```
./trace -p ./example/example.conf
```

The following command will change the number of PCs (*DIM*) to 6 and the prefix of output file names (*OUT_PREFIX*) to "HGDP", while the other parameters are defined by the *example.conf* file.

```
./trace -p ./example/example.conf -o HGDP -k 6
```

Results from *TRACE* 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 comment and will not be read by the program.

```
# This is a parameter file for TRACE v1.0.
# The entire line after a '#' will be ignored.
###----Main Parameters----###
STUDY_FILE    ./example/HGDP_700_chr22.geno          # no default value
```

```
GENO_FILE     ./example/HGDP_238_chr22.geno          # no default value
COORD_FILE    ./example/HGDP_238_chr22.RefPC.coord   # no default value
OUT_PREFIX    test                                   # default "trace"
DIM           2                                      # default 2
DIM_HIGH      20                                     # default [20]
MIN_LOCI      100                                    # default 100
###----Advanced Parameters----###
ALPHA               # default 0.1
THRESHOLD           # default 0.000001
REF_SIZE            # default 200
FIRST_IND           # default 1
LAST_IND            # default [last sample in the STUDY_FILE]
TRIM_PROP           # default 0
MASK_PROP           # default 0
EXCLUDE_LIST        # no default value
###----Command line arguments----###
# -p   parameterfile (this file)
# -s   STUDY_FILE
# -g   GENO_FILE
# -c   COORD_FILE
# -o   OUT_PREFIX
# -k   DIM
# -K   DIM_HIGH
# -l   MIN_LOCI
# -a   ALPHA
# -t   THRESHOLD
# -N   REF_SIZE
# -x   FIRST_IND
# -y   LAST_IND
# -M   TRIM_PROP
# -m   MASK_PROP
# -ex  EXCLUDE_LIST
###----End of file----###
```

3.2 Parallel jobs

We currently do not implement multi-thread option to run parallel jobs. Because each study 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. For example, running the following commands will submit two jobs: the first job will analyze samples 1 to 350; and the second job will analyze samples 351 to 700 in the *HGDP_700_chr22.seq* file.

```
./trace -p ./example/example.conf -x 1 -y 350 -o results.1-350 &
./trace -p ./example/example.conf -x 351 -y 700 -o results.351-700 &
```

Outputs from these two jobs will have different file name prefixes *results.1-350* and *results.351-700* 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.

3.3 Internal reference

In cases when external references are not available or when users want to focus on the study sample, *TRACE* provides an option to use a subset of individuals randomly from the study sample as an internal reference panel. The movitation here is to reduce computational cost when the sample size is too large for using standard PCA approaches. The size of the internal reference panel is defined by the parameter *REF_SIZE* (`-N`). This option is only in effect when the parameter *GENO_FILE* (`-g`) is undefined. For example, after commenting out the line for *GENO_FILE* in the *example.conf* file, the following command will randomly select 300 individuals from the *STUDY_FILE* as the reference panel, and run the *TRACE* analysis on the remaining 400 individuals.

```
./trace -p ./example/example.conf -N 300
```

Note that there is trade-off between computational efficiency and accuracy when using this internal reference option. Setting a smaller value for *REF_SIZE* can reduce the computational time, but might lead to less accurate results compared to results based on standard PCA. In addition, the option for running multiple jobs in parallel using parameters *FIRST_IND* (`-x`) and *LAST_IND* (`-y`) is not available when using the internal reference option.

4 Input files

In this section, we describe four input files that are taken by *TRACE* — the *parameterfile*, the *studyfile*, the *genotypefile*, and the *coordinatefile*. The *genotypefile* and the *studyfile* have the same format. We also describe one additional file, the *sitefile*, which is associated with the the *genotypefile* and the *studyfile*. Note that the formats of the *genotypefile*, the *sitefile*, and the *coordinatefile* are the same as used by the *LASER* software (Wang *et al.*, 2014a).

4.1 *parameterfile* (*_.conf*)

The *parameterfile* contains all parameters required for running *TRACE*. The default *parameterfile* is "trace.conf", which does not need to be explicitly specified in the command line (i.e. `./trace` is equivalent to `./trace -p trace.conf`). There are nine parameters in the *parameterfile* , including six main parameters and three 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 *TRACE* 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 *STUDY_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 six 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 *studyfile* (*_.geno*), *genotypefile* (*_.geno*), and *sitefile* (*_.site*)

The *studyfile* and the *genotypefile* contain genotype data for the study sample and the reference panel, respectively. *TRACE* does not require the *genotypefile* and the *studyfile* to contain same set of loci in the same order. Detail information of the loci should be provided in the *sitefile*'s. *TRACE* will automatically extract loci shared by the *studyfile* and the *genotypefile* for downstream analyses.

Each line in the *genotypefile*/*studyfile* 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 diploid individual. Missing data are coded by -9. *TRACE* can also be applied to multi-ploidy organisms. In general, genotypes should be coded by 0, 1, ..., K for K-ploidy organisms. Columns in the *genotypefile*/*studyfile* are tab-delimited. An example 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 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
...  ...      ...         ...  ...
```

To run *TRACE*, a ".site" file is required for each ".geno" file. Users can convert their genotype data from VCF format to our ".geno" and ".site" format using a program called *vcf2geno*. This program is available from the *LASER* software package (http://www.sph.umich.edu/csg/chaolong/LASER/). The command line for running *vcf2geno* is

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

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

4.3 *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   ...
...     ...       ...    ...    ...    ...
```

If the *coordinatefile* is not provided, *TRACE* will automatically compute the reference coordinates based on the *genotypefile*, and the results will be output. We recommend users to prepare a *coordinatefile* as input for *TRACE* 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

TRACE has 15 parameters that users can set in the *parameterfile*, including 7 main parameters that are required for running *TRACE* and 8 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 8 advanced parameters have default values. In addition, *TRACE* takes 16 command line arguments, which are described in Section 5.3.

5.1 Main parameters

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

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

COORD_FILE (string) The name of the *coordinatefile*. If the file is not in the same directory as *TRACE*, the whole path must be specified. This parameter is optional. If undefined, *TRACE* 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 "*TRACE*".

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*. *TRACE* 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 non-missing loci required for an individual in the study sample to be analyzed (must be a positive integer). If the number of non-missing loci is smaller than *MIN_LOCI*, the individual will not be analyzed and results for this individual are output as "*NA*". The default value is 100.

5.2 Advanced parameters

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.

REF_SIZE (int) The number of individuals to be randomly selected from the study sample as an internal reference panel (must be a positive integer). This number cannot be greater than the number of individuals in the *studyfile*. This parameter will be in effect only if *GENO_FILE* is undefined (*i.e.*, an external reference panel is not provided). The default value is 200.

FIRST_IND (int) The index of the first individual in the study sample to analyze (must be a positive integer). This number cannot be greater than the number of individuals in the *studyfile*. Individuals that have indices smaller than *FIRST_IND* will be skipped. This parameter will be in effect only if *GENO_FILE* is defined (*i.e.*, an external reference panel is provided). The default value is 1.

LAST_IND (int) The index of the last individual in the study sample to analyze (must be a positive integer). This number cannot be greater than the number of samples in the *studyfile* or smaller than *FIRST_IND*. Individuals that have indices greater than *LAST_IND* will be skipped. This parameter will be in effect only if *GENO_FILE* is defined (*i.e.*, an external reference panel is provided). The default value is the number of samples in the *studyfile*.

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.

MASK_PROP (double) Proportion of loci in a study sample that will be randomly set to missing (must be a number between 0 and 1). This option is useful when testing the robustness to missing data or examing the minimal number of markers requested in a sample in order to have satisfying performance given a reference panel. 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 *TRACE*, the whole path must be specified. This parameter do not have default value.

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".

```
./trace -p my_parameterfile -k 4
./trace -k 4 -p my_parameterfile
./trace -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 "trace.conf" in the current directory. If this file does not exist, an empty template *parameterfile* named "trace.conf" will be created in the current directory, and the program will then exit with an error message.

-s Change the parameter value of *STUDY_FILE*.

-g Change the parameter value of *GENO_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*.

-a Change the parameter value of *ALPHA*.

-t Change the parameter value of *THRESHOLD*.

-N Change the parameter value of *REF_SIZE*.

-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).

-M Change the parameter value of *TRIM_PROP* (useful when memory is limited).

-m Change the parameter value of *MASK_PROP*.

-ex Change the parameter value of *EXCLUDE_LIST*.

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 *TRACE*. It starts with all parameter values used in the execution of *TRACE*, 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, *TRACE* 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 4.3), so that this file can be directly used as the input file for *TRACE*.

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 _*.ProPC.coord*

This file contains the estimated PCA (or "Procrustean PCA") coordinates of the study sample. 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 reports the number of nonmissing loci used in the analysis. The fourth column reports values of *DIM_HIGH*, dimension of the sample-specific PCA map used in projection Procrustes analysis. The fifth column reports the Procrustes similarity score between each sample-specific PCA map (after being projected to the *DIM* dimensional space) and the original *DIM* dimensional reference PCA map. Starting from the sixth 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 using the internal reference option, coordinates for the individuals that are selected as the internal reference will also be included in this file. The values of *DIM_HIGH* (the fourth column) and the Procrustes similarity scores (the fifth column) for these individuals will be listed as "*NA*".

7 Computational complexity

TRACE examines one study individual at a time. Therefore, the computational costs increase linearly with the number of individuals to be analyzed. We can easily run the analyses in parallel by submitting multiple jobs to analyze different subsets of the study sample (using command line flags `-x` and `-y`).

The cost for analyzing each individual depends on the number of individuals, N, and the number of loci, L, in the reference panel, and the proportion of missing data, m, in the study individual. We first calculate the $N \times N$ pairwise similarity matrix of the reference panel, for which the computational cost is $O(N^2L)$. This computation is only performed once and will be repeatedly used in analyzing each individual. Roughly, we expect computational cost of PCA for each study individual to be $O(NL + N^2Lm + N^3)$, in which NL is the time required for comuputing the extra row (and column) for the study individual in the similarity matrix, N^2Lm is for adjustment of the similarity matrix to account for missing data in the study individual, and N^3 is for eigen decomposition on the similarity matrix. The computational cost of projection Procrustes analysis is approximately $O[q(NK + K^3)]$, where q is the number of iterations required for projection Procrustes analysis to converge and K is the value of *DIM_HIGH*. When K is not big, the analysis often converges within a few iterations, and the cost for projection Procrustes analysis is negligible compared to the cost for PCA. Overall, the computational cost for *TRACE* on a study sample of n individuals is approximately $O[N^2L + n(NL + N^2L\bar{m} + N^3 + qNK + qK^3)]$. If we ignore the cost for projection Procrustes analysis and suppose the missing data rate is small, the computational cost for *TRACE* can be approximated as $O[N^2L + n(NL + N^3)]$. In comparison, the cost for a standard PCA on n individuals is approximately $O(n^2L + n^3)$.

8 Version changes

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

8.1 Version 1.0 (May 19, 2014)

- Initial release of the *TRACE* software.

9 Acknowledgements

I would like to thank Conrad Sanderson at the National ICT Australia for his help on using the Armadillo library.

References

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

Lee, A. B., D. Luca, L. Klei, B. Devlin and K. Roeder, 2010 Discovering genetic ancestry using spectral graph theory. Genet. Epidemiol. **34:** 51–59.

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.

McVean, G., 2009 A genealogical interpretation of principal components analysis. PLoS Genet. **5:** e1000686.

Patterson, N., A. L. Price and D. Reich, 2006 Population structure and eigenanalysis. PLoS Genet. **2:** 2074–2093.

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

Skoglund, P., H. Malmström, M. Raghavan, J. Storå, P. Hall, E. Willerslev, M. T. Gilbert, A. Götherström and M. Jakobsson, 2012 Origins and genetic legacy of Neolithic farmers and hunter-gatherers in Europe. Science **336:** 466–469.

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, 2014a Ancestry estimation and control of population stratification for sequence-based association studies. Nature Genetics **46:** 409–415.

Wang, C., X. Zhan, J. Z. Li, N. A. Rosenberg, L. Liang, G. R. Abecasis and X. Lin, 2014b Improved ancestry estimation for both genotyping and sequencing data using projection procrustes analysis and genotype imputation. (in preparation).