MicroDrop: a program for estimating and correcting for allelic dropout in nonreplicated microsatellite genotypes version 1.01

Chaolong Wang[1]
Department of Computational Medicine and Bioinformatics
University of Michigan, Ann Arbor, MI

Noah A. Rosenberg
Department of Biology
Stanford University, Stanford, CA

December 29, 2012

The *MicroDrop* software is available at
`http://rosenberglab.stanford.edu/software.html`

[1]Comments on *MicroDrop* can be sent to chaolong@umich.edu

Contents

1 Introduction

MicroDrop is a program to estimate and correct for allelic dropout in microsatellite genotype data from diploid organisms. The code is written in C++, and the executable program can work on reasonably recent versions of Linux and Windows.

Unlike methods that usually estimate allelic dropout on basis of replicate genotyping (e.g. Miller *et al.*, 2002; Johnson and Haydon, 2007), *MicroDrop* is specifically designed for population-genetic data with only a single set of genotypes. Instead of using replicate genotyping, *MicroDrop* requires genotypes of a sufficient large number of individuals on multiple independent loci such that population-genetic assumptions can be applied.

In this section, we provide a brief description of the framework and methods used in *MicroDrop*. For detailed information about the methods, including the model and specific implementation of the algorithm, please refer to our paper entitled "A maximum likelihood method to correct for allelic dropout in microsatellite data with no replicate genotypes" (Wang *et al.*, 2012).

1.1 Examination for allelic dropout

Allelic dropout is the most common genotying error for microsatellites, occurring when one or both of an individual's two allelic copies fail to be amplified by the polymerase chain reaction (PCR). Because allelic dropout can cause both missing genotypes (if both alleles drop out) and false homozyotes (if only one allele of a heterozygote drops out), we expect to observe a positive correlation between the amount of homozygotes and the amount of missing data across individuals and across loci. In particular, if sample-specific factors, such as low DNA concentration or poor sample quality, are the dominant factors for allelic dropout, the positive correlation will be more significant across individuals. If locus-specific factors, such as binding affinity between primers or polymerase and the target DNA, are significant, a positive correlation between homozygotes and missing data will be observed across loci as well. The first step of *MicroDrop* is to calculate the correlation between the amount of missing data and the amount of homozygotes across individuals and across loci respectively. Two permutation tests are used to assess the statistical significance of the correlations. Results from this step provide a simple examination of the existence of allelic dropout in a given microsatellite data set.

1.2 Model and parameter estimation

In the second step, *MicroDrop* jointly estimates the sample-specifc dropout rates, locus-specific dropout rates, and other parameters in the model. The model has five assumptions:

1. All distinct alleles are observed at least once in the data set;
2. All missing and incorrect genotypes are attributable to allelic dropout;
3. Both copies at a locus ℓ of an individual i have equal probability $\gamma_{i\ell}$ of dropping out. This probability is a function of a sample-specific dropout rate $\gamma_{i\cdot}$ and a locus-specific dropout rate $\gamma_{\cdot\ell}$:

$$\gamma_{i\ell} = \gamma_{i\cdot} + \gamma_{\cdot\ell} - \gamma_{i\cdot}\gamma_{\cdot\ell};$$

4. All individuals are unrelated and have the same inbreeding coefficient ρ, such that for any locus of any individual, the two allelic copies are identical by descent (IBD) with probability ρ;
5. Each pair of loci is independent (i.e. each pair of loci is at linkage equilibrium).

Suppose the number of individuals is N and the number of genotyped loci is L. There are three sets of parameters in this model. The first set is allele frequencies $\Phi = \{\phi_{\ell k} : \ell = 1, 2, ..., L; k = 1, 2, ..., K_\ell\}$, in which K_ℓ is the number of distinct alleles at locus ℓ. The second set is the dropout rates $\Gamma = \{\gamma_{i\cdot}, \gamma_{\cdot\ell} : i = 1, 2, ..., N; \ell = 1, 2, ..., L\}$, in which $\gamma_{i\cdot}$ is the probability of dropping out due to sample-specific factors for a randomly chosen allelic copy at a randomly chosen locus of individual i, and $\gamma_{\cdot\ell}$ is the probability of dropping out due to locus-specific factors for a randomly chosen allelic copy at locus ℓ of a randomly chosen individual. The last one is the inbreeding coefficient ρ, which is assumed to be the same for all loci in all individuals.

Under this model, *MicroDrop* uses an expectation-maximization (EM) algorithm to jointly obtain the maximum likelihood estimates (MLEs) of all parameters $\Psi = \{\Phi, \Gamma, \rho\}$. In addition, *MicroDrop* provides advanced options that allow users to replace the inbreeding assumption (model assumption 4) with an assumption of Hardy-Weinberg equilibrium (i.e. $\rho = 0$, HWE), as well as to assume only sample-specific factors for allelic dropout (i.e. $\gamma_{\cdot\ell} = 0$ for $\ell = 1, 2, ..., L$), or only locus-specific factors for allelic dropout (i.e. $\gamma_{i\cdot} = 0$ for $i = 1, 2, ..., N$). The options of different assumptions on allele frequencies (inbreeding or HWE) and dropout factors (only sample-specific factors, only locus-specific factors, or both) can be applied together, such that there are six (2×3) combinations in total.

1.3 Genotype imputation

In the last step, *MicroDrop* creates imputed data sets by drawing genotypes according to the posterior probability based on the model and the parameters estimated in the second step. The imputed data sets can then be used for various downstream analyses in replace of the original data set with a large amount of allelic dropout. It is worth noting that there are many other imputation strategies that can be used to create imputed data sets (e.g. Little and Rubin, 2002). It may be more appropriate to choose other imputation strategies depending

on specific questions. In such cases, users can implement their own imputation methods based on parameter values estimated in the second step of *MicroDrop*.

2 Getting started

2.1 Availability

Pre-compiled executables for *MicroDrop* for Linux (64-bit) and Windows (64-bit) can be downloaded from the Rosenberg Lab webpage:
`http://rosenberglab.stanford.edu/software.html`
For Windows users, the operating system is requested to be Windows 2000, XP, Vista, or 7. Please use the following citation for *MicroDrop*.

Chaolong Wang, Kari B. Schroeder, and Noah A. Rosenberg (2012). A maximum-likelihood method to correct for allelic dropout in microsatellite data with no replicate genotypes. *Genetics*, 192: 651-669.

2.2 Installing *MicroDrop*

2.2.1 Linux

Open a terminal in the same directory as the `.tar.gz` file. Extract the file by typing `tar -xzvf MicroDrop-1.01-x86_64.tar.gz` in the terminal. This will create a new directory called `MicroDrop-1.01-x86_64` containing the executable.

2.2.2 Windows

Extract the file `MicroDrop-1.01-win64.zip`. This will create a directory `MicroDrop-1.01-win64` containing the executable.

2.3 Running *MicroDrop*

2.3.1 Linux

Open a terminal and path to the directory that contains the executable *MicroDrop*. If you did not rename the directory after extracting the `.tar.gz` file, the directory will be `MicroDrop-1.01-x86_64`. Execute the program by typing `./microdrop -p` ***`paramfile`***, in which `-p` is the command line flag specifying the parameter file and *paramfile* is the name of the parameter file. If your *paramfile* is not in the same directory, you must specify the whole path to the file. If the *paramfile* is not specified, *MicroDrop* will search in the current directory for a *paramfile* named "paramfile.txt", and execute the program with parameter values

specified in "paramfile.txt". If this file does not exist, an empty template *paramfile* named "paramfile.txt" will be created in the current directory, and the program will then exit with an error message. For more command line arguments, see Section 4.3.

2.3.2 Windows

There are two ways to run *MicroDrop* in Windows. The first one to run it with an MS-DOS command prompt. Open a command prompt by clicking on RUN in the START menu and typing `cmd` or `command`. Specify the path to the directory where the *MicroDrop* executable locates. Execute the program by typing `microdrop -p` ***`paramfile`***, where `-p` is the command line flag specifying the parameter file and *paramfile* is the name of the parameter file. If your *paramfile* is located in a different directory, the whole path to the file must be specified. When running *MicroDrop* without specifying the *paramfile*, the program will search in the current directory for a *paramfile* named "paramfile.txt". If this file does not exist, *MicroDrop* will generate an empty template paramfile named "paramfile.txt" in the current directory and then exit with an error message. For more command line arguments, see Section 4.3.

The second way to execute the *MicroDrop* in Windows is by double-clicking on the *MicroDrop* icon or the *microdrop.exe* file. However, there must be a valid *paramfile* named "paramfile.txt" in the same directory. If not, *MicroDrop* will generate an empty template paramfile named "paramfile.txt" in the current directory and then exit with an error message. No command line arguments can be passed when *MicroDrop* is run in this way.

3 Input files

In this section, we describe two input files that are taken by *MicroDrop* — the *paramfile* and the *datafile*.

3.1 *paramfile*

The *paramfile* contains all parameters required for running *MicroDrop*. The default *paramfile* is "paramfile.txt", which does not need to be explicitly specified in the command line (i.e. `./microdrop` is equivalent to `./microdrop -p paramfile.txt`). There are 14 parameters in the *paramfile* , including 12 main parameters and two 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 *MAX_ITER* equal to 1000:

```
MAX_ITER 1000
MAX_ITER   1000 # Number of EM iterations
```

```
MAX_ITER 1000   # 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 *MicroDrop* when the default *paramfile* does not exist and without any command line arguments. Five parameters — *DATA_FILE*, *INDS*, *LOCI*, *NON_DATA_ROWS*, and *NON_DATA_COLS* — must be explicitly defined by the user, either in the *paramfile* or in the command line (see Section 4.3). The other nine parameters do not need to be explicitly defined unless the user wants to use settings different from the default. Please refer to Section 4 for more information on these parameters.

3.2 *datafile*

The *datafile* follows the format used by *structure* (Figure 1). The first line must list all locus names, and may optionally be followed by a specified number of non-data lines. We only consider diploid organisms. Genotypes of each individual are printed on two consecutive lines. Each line of an individual starts with a specified number of non-data columns followed by the allelic values at L loci. It is required that the first non-data column must be the individual IDs. The example shown in Figure 1 contains one non-data line that lists the locus names, and two non-data columns that specify individual IDs and population names, respectively. The *datafile* is required to be whitespace-delimited. Missing data are represented by -9, and nonmissing alleles are represented by positive integers. In addition, when a missing allele is detected, the program assumes that the other allele at the same locus of the same individual is also missing. Otherwise, the program will exit with an error message.

		Loc_1	Loc_2	Loc_3	Loc_4	Loc_5	Loc_6	Loc_7
Indiv_1	Pop_1	180	205	-9	274	186	218	192
Indiv_1	Pop_1	170	205	-9	271	183	215	188
Indiv_2	Pop_1	180	213	219	274	192	215	192
Indiv_2	Pop_1	180	205	219	271	183	215	192
Indiv_3	Pop_2	190	201	219	-9	186	-9	192
Indiv_3	Pop_2	180	201	219	-9	177	-9	176
Indiv_4	Pop_2	185	201	222	277	189	221	188
Indiv_4	Pop_2	180	201	219	274	183	212	164
Indiv_5	Pop_2	190	209	219	274	186	221	192
Indiv_5	Pop_2	180	205	216	271	180	206	176

Figure 1: A small *datafile* to illustrate the data format.

4 Usage options

MicroDrop has 14 parameters that users can set in the *paramfile*, including 12 main parameters that are required for running *MicroDrop* and two advanced parameters that allow users to change the model assumptions. Among the 12 main parameters, five are parameters regarding the data and need to be explicitly defined for *MicroDrop* to function properly. The other seven main parameters and the two advanced parameters have default values. In addition, *MicroDrop* takes 15 command line arguments, which are described in Section 4.3.

4.1 Main parameters

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

INDS (int) The number of individuals in the data set (must be a positive integer). This value is used to error-check against the *datafile*. This parameter must be explicitly defined.

LOCI (int) The number of loci in the data set (must be a positive integer). This value is used to error-check against the *datafile*. This parameter must be explicitly defined.

NON_DATA_ROWS (int) The number of non-data rows in the *datafile* (must be a positive integer). This parameter is always at least 1, indicating a row for locus names. Locus names are always assumed to be on the first row, and the other non-data rows will be ignored. This value is used to error-check against the *datafile*. This parameter must be explicitly defined.

NON_DATA_COLS (int) The number of non-data columns in the *datafile* (must be a positive integer). This parameter is always at least 1, indicating a column for individual IDs. Individual IDs are always assumed to be on the first column, and the other non-data columns will be ignored. This value is used to error-check against the *datafile*. This parameter must be explicitly defined.

OUT_FOLDER (string) The name of the folder to be created for outputting results. If a path is not specified, the folder will be created in the current directory. If the folder already exists, it will be completely overwritten. The default value is "*MicroDrop_Outputs*".

TOLERANCE (double) The tolerance of the log-likelihood function for assessing the convergence of the EM (must be a nonnegative number). An EM will be considered as converged

if the increase of the log-likelihood function ($\log_{10}$) in one iteration is less than the tolerance value. The iteration process will then be stopped. The default value is 10^{-4}.

MAX_ITER (int) The maximum number of iterations for each EM (must be a positive integer). When the number of iterations reaches this value, the EM will stop even if it has not met the convergent criterion set by *TOLERANCE*. The default value is 10,000.

MAX_ALLELES (int) The maximum number of distinct alleles allowed at any locus (must be a positive integer). This number must be no less than the actual number of distinct alleles at any locus. The default value is 100.

N_PERMU (int) The number of permutations used to evaluate the significance of the correlation between missing genotypes and homozygous genotypes (must be a positive integer). The default value is 10,000.

N_REPS (int) The number of replicates for running the EM algorithm (must be a positive integer). Each replicate starts with an independent set of randomly generated initial values for parameters in the model, and runs until the *TOLERANCE* criterion is reached or until *MAX_ITER* iterations have been run. Only the results from the replicate that converges to the highest likelihood will be chosen as the MLEs and output by *MicroDrop*. Running multiple replicates can lower the chance of being stuck at a local maximum. The default value is 100.

N_IMPT (int) The number of imputed data sets to be created (must be a nonnegative integer). If set to 0, *MicroDrop* will skip the imputation step without creating any imputed data set. The default value is 2.

4.2 Advanced parameters

ASMPT (int) This parameter specifies the population-genetic assumption in the model (must be 0 or 1). A value of 0 will use the inbreeding assumption as described in Section 1 (model assumption 4). A value of 1 will replace the inbreeding assumption with an assumption of Hardy-Weinberg equilibrium, which is equivalent to assuming the inbreeding coefficient ρ is equal to 0. The default value is 0.

FCTR (int) This parameter specifies the dropout factor assumption in the model (must be 0, 1 or 2). A value of 0 will use the dropout factor assumption as described in Section 1 (model assumption 3), which considers both sample-specific and locus-specific factors for allelic dropout. A value of 1 will consider only sample-specific factors, which is equivalent to

assuming all locus-specific dropout rates are equal to 0. A value of 2 will consider only locus-specific factors, which is equivalent to assuming all sample-specific dropout rates are equal to 0. The default value is 0.

4.3 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 *paramfile*. If a parameter is specified with an invalid value in the *paramfile* 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, on a Linux platform, the following command lines are equivalent and will change the value of the parameter *MAX_ITER* to be 200 while using the other parameters defined in the *paramfile* named "my_paramfile".

```
./microdrop -p my_paramfile -t 200
./microdrop -t 200 -p my_paramfile
./microdrop -t 300 -p my_paramfile -t 200
```

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

-p (*paramfile*) This flag defines the *paramfile*. If the *paramfile* 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 *paramfile* named "paramfile.txt" in the current directory. If this file does not exist, an empty template *paramfile* named "paramfile.txt" will be created in the current directory, and the program will then exit with an error message.

-d (DATA_FILE) Change the source *datafile* .

-i (INDS) Change the number of individuals (useful when using -d for a new *datafile*).

-l (LOCI) Change the number of loci (useful when using -d for a new *datafile*).

-u (NON_DATA_ROWS) Change the number of non-data rows (useful when using -d for a new *datafile*).

-v (NON_DATA_COLS) Change the number of non-data columns (useful when using -d for a new *datafile*).

-o (OUT_FOLDER) Change the name of the folder to be created for outputting results.

-s (TOLERANCE) Change the tolerance for assessing the convergence of the EM.

-t (MAX_ITER) Change the maximum number of iterations for each EM.

-k (MAX_ALLELES) Change the maximum number of distinct alleles allowed at any locus.

-x (N_PERMU) Change the maximum number of permutations for evaluating the significance of the correlation between missing data and homozygotes.

-r (N_REPS) Change the number of EM replicates.

-m (N_IMPT) Change the number of imputed data sets to be created.

-a (ASMPT) Change the population-genetic assumption in the model.

-f (FCTR) Change the dropout assumption in the model.

5 Output files

All output files will be saved in a folder named by *OUT_FOLDER*, which will be automatically created by *MicroDrop*. These files are described below.

5.1 *microdrop.log* and terminal outputs

The terminal outputs are used to monitor and record the progress when running *MicroDrop*. For Windows users who run *MicroDrop* without a terminal, the *microdrop.log* file will provide identical information to the terminal outputs. Please see Figure 2 for an example of the terminal outputs and the *microdrop.log* file. It starts with all parameter values used in the execution of *MicroDrop*, and reports the progress of the program step by step.

Importantly, *microdrop.log* contains a table of information on the convergence of all EM replicates (as shown in Step 2 in Figure 2). The first column of this table is the index of each EM replicate. The second column is the number of iterations run by each EM replicate.

The third and the fourth columns report the estimated values of the inbreeding coefficient ρ in the last two iterations (e.g. columns *rho(t-1)* and *rho(t)* in the table in Figure 2). If the assumption of Hardy-Weinberg equilibrium is applied ($ASMPT = 1$), these two columns are zeros ($\rho = 0$). The fourth and the fifth columns report the log-likelihood ($\log_{10}$) in the last two iterations (e.g. columns *logL(t-1)* and *logL(t)* in the table in Figure 2). This table provides useful information for users to monitor the convergence of the EM. If there is a big difference between estimated values of ρ or the log-likelihood in the last two iterations, it means the EM has not converged yet. Therefore, users can adjust the values of *TOLERANCE* and *MAX_ITER* accordingly. Below the table, the program will report the index of the EM replication whose results are selected as the MLEs.

```
========================================================================
======                      MicroDrop version 1.0                   =====
========================================================================
Started at: Fri Aug  3 16:43:20 2012

Parameter values used in execution:

DATA_FILE   test_data    # Data file name (-d)
INDS    200  # Number of individuals (-i)
LOCI    300  # Number of loci (-l)
NON_DATA_ROWS   1    # Number of non-data rows (-u)
NON_DATA_COLS   1    # Number of non-data columns (-v)
OUT_FOLDER  test_results     # Output folder name (-o)
TOLERANCE   0.001    # Tolerance for assessing the convergence (-s)
MAX_ITER    10000    # Maximum number of EM iterations (-t)
MAX_ALLELES 50   # Maximum number of distinct alleles (-k)
N_PERMU 100  # Number of permutations (-x)
N_REPS  3    # Number of EM replications (-r)
N_IMPT  4    # Number of imputed datasets (-m)
ASMPT   0    # Population genetic assumption (-a)
FCTR    0    # Dropout factor assumption (-f)

Step 1: checking correlation bewteen missing data and homozygotes

Pearson correlation across individuals:
     R=0.862953  P-value<0.01
Pearson correlation across loci:
     R=0.70449   P-value<0.01

Step 2: estimating parameters using EM algorithm

EM  t   rho(t-1)    rho(t)      logL(t-1)   logL(t)
rep-1   142 0.097114    0.097120    -63787.32   -63787.32
rep-2   120 0.097608    0.097601    -63787.32   -63787.32
rep-3   132 0.097486    0.097481    -63787.32   -63787.32

Results from rep-2 have been picked as MLEs.
MLEs have been output to files in folder 'test_results'.

Step 3: generating imputed datasets based on MLEs from step 2

4 imputed datasets have been output to folder 'test_results/Imputed_Data'.

Finished at: Fri Aug  3 16:43:23 2012
========================================================================
```

Figure 2: An example of the *microdrop.log* file, which is identical to the terminal outputs.

5.2 *MisHom_ind* and *MisHom_loc*

These two files contain information on missing genotypes and homozygotes across individuals and across loci, respectively. For the *MisHom_ind* file, the first line of the file reports the Pearson correlation between the fraction of missing data among all loci and the fraction of homozygotes among all nonmissing loci across individuals. A p-value obtained from *N_PERMU* permutations is also reported in the first line. A three-column table starts from the second line of the file. The first column is the individual ID, and the second and the third columns are the fraction of missing data among all loci and the fraction of homozygotes among all nonmissing loci in each individual.

The the *MisHom_loc* file has a similar format. The first line of the file reports the Pearson correlation and the corresponding p-value between the fraction of missing data among all individuals and the fraction of homozygotes among all nonmissing individuals across loci. A table starts from the second line of the file, with three columns corresponding to the locus ID (first column), the fraction of missing data among all individuals at each locus (second column), and the fraction of homozygotes among all nonmissing individuals at each locus (third column). For both files, columns in the table are tab-delimited and the first row of the table contains the column names.

5.3 *EM_Loglikelihood*

This file reports the log-likelihood ($\log_{10}$) for every iteration of the EM replicate whose results are picked as MLEs. The file contains two tab-delimited columns. The first column is the index of each EM iteration and the second one is the value of the log-likelihood function. The first row of each column is the column name. The iteration index starts from 0, corresponding to the initial point of the EM algorithm.

5.4 *Allele_List* and *MLE_Allele_Freq*

These two files contain information about all observed alleles in the *datafile*. There are three tab-delimited columns in either of these two files. For both files, the first row provides column names, and the first two columns are the locus ID and the number of distinct alleles at each locus. The third column in the *Allele_List* file lists all distinct alleles for each locus in a space-delimited format. The third column of the *MLE_Allele_Freq* file is the estimated allele frequencies corresponding to the alleles listed in the *Allele_List* file, in the same order. The allele frequencies at a locus are also space-delimited.

5.5 *MLE_Rates_ind* and *MLE_Rates_loc*

These two files contain the estimated values of sample-specific dropout rates and locus-specific dropout rates, respectively. There are two tab-delimited columns in each file. The first row of both files provides column names. For the *MLE_Rates_ind* file, the first column is the individual IDs, and the second column is the estimated values of the corresponding sample-specific dropout rates. The *MLE_Rates_loc* file has a similar format. The first column is the locus IDs, and the second column is the estimated values of the corresponding locus-specific dropout rates. If $FCTR = 1$ or 2, both *MLE_Rates_ind* and *MLE_Rates_loc* will still be generated, and either *MLE_Rates_loc* (when $FCTR = 1$) or *MLE_Rates_ind* (when $FCTR = 2$) will contain zero values for all estimated dropout rates.

5.6 *MLE_Inbreeding_Coef*

This file contains the estimated value of the inbreeding coefficient. Similar to other files, the first row is the column name and the estimated value is in the second row (one value only). If $ASMPT = 1$, this file will still be generated and the estimated value is zero.

5.7 Imputed data sets

If *N_REPS* is greater than 0, *MicroDrop* will create imputed data sets by drawing genotypes according to the posterior probabilities based on the MLEs of parameters. *MicroDrop* will then automatically create a folder named "*Imputed_Data*" inside the *OUT_FOLDER* for saving the imputed data sets. Each imputed data set is named by "*ImptD_k*", in which k is an index from 1 to *N_REPS*. The format for these files is the same as in the *datafile*.

6 Version changes

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

6.1 Version 1.0 (Aug 15, 2012)

- Initial release of the *MicroDrop* software.

6.2 Version 1.01 (Dec 29, 2012)

- Seg fault bug fixed. In version 1.0, when running the program in termial, the program could potentially crashed if a command line flag is given but the corresponding parameter value is not provided. Version 1.01 corrects this bug.

- Bug in reading the *paramfile* fixed. In version 1.0, if comments in the *paramfile* after the "#" contain a parameter, texts following the parameter could be mistakenly taken as the paramter value. Version 1.01 corrects this bug.

7 Acknowledgements

We are grateful to Zachary Szpiech for his generous help on implementing and testing the software, and on this manual.

References

Johnson, P. C. D., and D. T. Haydon, 2007 Maximum-likelihood estimation of allelic dropout and false allele error rates from microsatellite genotypes in the absence of reference data. Genetics **175:** 827–842.

Little, R. J. A., and D. B. Rubin, 2002 *Statistical Analysis with Missing Data*. John Wiley & Sons, Hoboken, NJ 2nd edition.

Miller, C. R., P. Joyce and L. P. Waits, 2002 Assessing allelic dropout and genotype reliability using maximum likelihood. Genetics **160:** 357–366.

Wang, C., K. B. Schroeder and N. A. Rosenberg, 2012 A maximum-likelihood method to correct for allelic dropout in microsatellite data with no replicate genotypes. Genetics **192:** 651–669.