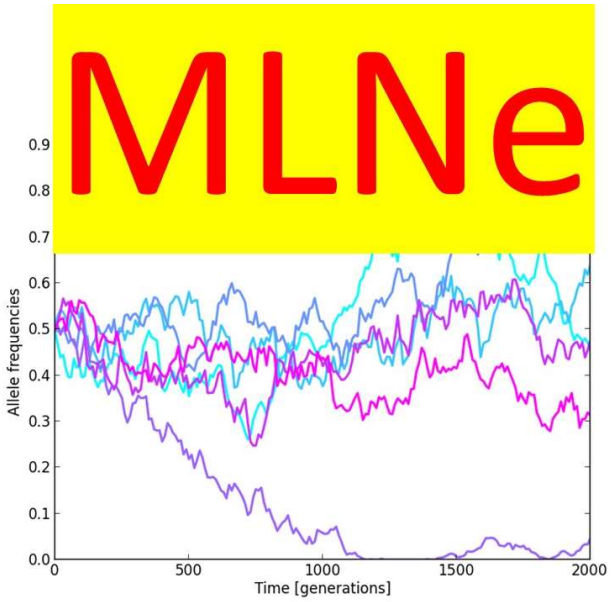
MLNe Software Manual

Version 2.1.0.0 (June 14, 2022)

Jinliang Wang

Institute of Zoology, Zoological Society of London, London NW1 4RY, UK



CONTENTS

1. Introduction

1.1 Overview

The frequency of an allele in a finite population is not constant. It changes over generations due to evolutionary forces acting on the population, such as genetic drift, selection, mutation and migration. For neutral alleles (i.e. no selection) over a short time scale (i.e. mutations negligible), allele frequency is expected to change due to genetic drift determined by the effective size, N_e, of the population, and due to immigration with rate m into the population. Using 2 or more temporal samples of individuals taken from the population, we can directly measure the extent of allele frequency changes at a number of neutral marker loci. From the observed (measured) allele frequency changes, we can deduce the average effective size, N_e, of and the average rate of immigration, m, into the population (Wang & Whitlock 2003).

The software package **MLNe** implements a moment estimator and a maximum likelihood estimator of N_e and m from temporal genotype data under two demographic models. The ***isolation*** model assumes a single isolated population (i.e. m=0). Therefore, any observed allele frequency changes at neutral loci in a short period of time are due to genetic drift only, and thus indicate the N_e of the population during this period. The more extensive changes of allele frequencies, the smaller the average N_e will be during the sampling period. While a population may not be closed (isolated) and may receive immigrants in the long term, it may be regarded as isolated or approximately so in a short period of time, say a couple of generations, during which samples are taken. For this isolation model, **MLNe** implements the maximum likelihood method developed by Wang (2001), and the moment estimator developed by Nei & Tajima (1981).

The ***migration*** model assumes a focal small population receiving immigrants from a large source population. In this model, allele frequencies of the focal population may change over generations because the population is small and thus suffers from genetic drift, and because the population receives immigrants from other (source) populations. With two or more temporal samples from the focal population and a sample from the source population, we can estimate both the average N_e of and the average rate of immigration, m, into the focal population. For this migration model, **MLNe** implements the moment and likelihood methods developed by Wang & Whitlock (2003) to estimate the effective population size (N_e) and immigration rate (m) jointly from temporal and spatial data on genetic markers. The methods assume an infinitely large source population providing immigrants into the focal population whose N_e and m are to be estimated. However, the methods are very robust to violations of the assumption, and can be applied approximately to a finite source population composing one or more small subpopulations (see Wang & Whitlock 2003 for details). Both point estimates and 95% confidence intervals of N_e and m are obtained from the moment and likelihood methods by the program.

Note that **MLNe** allows any number of temporal samples. For more than 2 samples, average N_e and m over the entire sampling period are estimated directly by the likelihood method, while N_es and ms for each sampling period are estimated by the moment estimator and their harmonic and arithmetic means are reported respectively.

The software package **MLNe** runs on Windows, Mac and linux computers. It includes the source code, executable, user's guide and example datasets. The computational part of **MLNe** program was written in Fortran 2003, and the source code can be compiled and run in different platforms (Windows, Linux and Mac). The Windows version of **MLNe** also includes a front end (Graphical User Interface, GUI) written in Visual Basic to help preparing input data and parameters and viewing analysis results in tables and graphs. The front end can also be used to simulate genotype data of temporal samples of individuals taken from a population with (user) given parameters of interest (N_e and m). The simulated data can then be analysed by **MLNe** to check the estimation accuracy and/or to investigate data sufficiency.

1.2 Features of the method and software

The current version of **MLNe** has the following features:

- Including a migration model and an isolation model;
- Simulating temporal genotype data with user defined sampling (e.g. sample sizes, sampling intervals, number of loci, marker polymorphism) and population (e.g. N_e and m) properties;
- Allowing any number (>1) of temporal samples in empirical data analysis;
- Capable of parallel runs using multiple cpus/cores by both MPI and openMP to speed up the analysis of large datasets;
- Windows GUI.

The methods implemented by the software are described in the following paper:

Wang J. 2001. A pseudo-likelihood method for estimating effective population size from temporally spaced samples. *Genetics Research* 78: 243-57.

Wang J, Whitlock MC. 2003. Estimating effective population size and migration rates from genetic samples over space and time. *Genetics* 163: 429-46.

Nei M, Tajima F. 1981 Genetic drift and estimation of effective reach population size. *Genetics* 98: 625–640.

Wang J. 2022. MLNe: Simulating and estimating effective size and migration rate from temporal changes in allele frequencies. Journal of Heredity.

Please cite these publications when using the **MLNe** software.

1.3 Bug report

 It is downloadable from the website http://www.zsl.org/science/research/software/. Any updates of the program will also be put in the same website. Every effort has been made to implement the methods correctly and efficiently, but there is no guarantee that the program is free of bugs. Reports of bugs are welcome, and should be sent to: mailto:jinliang.wang@ioz.ac.uk?subject=MLNe.

2. Installation

MLNe is written in Fortran 2003, and is compiled for Windows 10, Mac and Linux 64bit operating systems. For Windows, it also has a graphical user interface (GUI) written in Vb.net, which can be

used to help inputting data and analysis parameters, and viewing analysis results in graphs and tables. Mac users can also install a Windows simulator to run the Windows version, as the Mac version is x-terminal based and has no GUI. For large genomic data, it is better to use the linux version for MPI and openMP parallel computation using many cores of a linux cluster.

2.1 Windows version

For Windows users, please download and unzip the zipped file for Windows version of **MLNe** to obtain an installation file called "MLNeSetup.msi". Double click this file to start the installation. By default, it will be installed in "C:\ZSL\MLNe". However, you can change the directory where **MLNe** will be installed during the installation process. It is suggested that it NOT be installed in the "Windows" directory or the "Program Files" directory. Otherwise, due to windows security issues, subsequent input and output files of **MLNe** might be automatically moved to a folder in VirtualStore and, the simulation program may not run properly.

Upon installation, you will find **MLNe** executables, user's manual in PDF, libraries, and two example datasets in your **MLNe** program folder. A shortcut to **MLNe** program is also placed in your Windows' desktop. Double clicking the shortcut will start the program. If for any reason the shortcut is missing, you can click the executable MLNe.exe in your **MLNe** program folder to start the program. You can also make a shortcut of this executable manually on the desktop.

MLNe should run on a PC or server with Microsoft Windows operating system version 10, 64bit. It requires the .Net Framework 4.5.2 (or higher), which is probably already installed on your computer. You can check by clicking Start button (on the bottom left corner) on your Windows desktop, selecting Control Panel, and then double-clicking the Add or Remove Programs icon. When that window appears, scroll through the list of applications to check whether Microsoft .Net Framework 4.5.2 (or higher) is listed. If it is not installed (which is unlikely), you need first download it from the Microsoft website and install it before installing **MLNe**. Occasionally, .Net might be installed but not enabled on your computer. In such a case, you need to enable it for **MLNe**'s GUI front end to work.

This PDF file of the user's manual is included in the package. I suggest opening/printing and reading this document before running **MLNe**. The analysis on one's own dataset using the program is described below.

2.2 Mac and Linux versions

Mac and Linux users can download the corresponding package, unzip it, and then copy everything (including the folders, subfolders and files) to your desired location. Everything described above, except for the GUI and the simulation module, should be included in the package.

It is suggested to include the **MLNe** program path permanently in the automatic search paths of your shell so that the program can be launched conveniently without specifying the path of the program. If you use the Bash shell of linux, for example, you can add the line "export PATH=$PATH:prmpath" (where prmpath is the path for **MLNe** program folder where the binaries are found) to file .bashrc that will be read when your shell launches. To do so, simply open the file by typing "nano ~/.bashrc", append the line, and save the file. The next time you launch your shell, the shell knows where to find **MLNe** program (e.g. **`mNe5`**) to run an analysis, no matter where (in which folder) the command line

invoking **mNe5** is issued. If you use zsh in Mac, append "export PATH=$PATH:prmpath" to file .zshenv. Similarly, if you run **MLNe** in DOS, you need to add **MLNe** program folder to the environment variable by using "Control Panel>System>Advanced System Settings>Environment Variables".

2.3 Example datasets

Two example datasets are also installed in **MLNe** program folder. The example "isolation" subfolder has 2 files for an empirical data analysis under the isolation model, and the example "migration" subfolder has 2 files for an empirical data analysis under the migration model. In each subfolder, "*.par" is the parameter file and "*.dat" is the genotype data file.

3 Input Data Files: Empirical Analysis

For an **MLNe** analysis of an empirical dataset, two input files in specific formats need to be prepared. One file, called parameter file hereafter, contains analysis parameters. The other one, called data file hereafter, contains genotypes of sampled individuals. I describe how to prepare the two files for running **MLNe** with Windows GUI front end (on a PC running Windows 10) and without (on Windows DOS or on the x-terminal of Linux, Mac).

3.1 No Windows GUI front end

To keep input and output files well organized, it is advised to create a suitably named (a string with no space and no other illegal characters for file/folder names) new folder as your project folder, and save the data file and parameter file in this folder. The path of the project folder, called *project path* hereafter, will be frequently used in working with the project.

3.1.1 Parameter file

A parameter file with a suitable name (say, MyData.par) needs to be prepared and saved in the project folder. The extension name ".par" is not compulsory, but is recommended as the simulation module in Windows GUI will generate a parameter file with this fixed extension ".par". The file should have exactly 13 text lines in the right order (below), with each line listing one or more parameter values. In the example file shown below, each line is started with a parameter value, followed by the comment/note (the part following exclamation mark !). In the comment, the 1st, 2nd and 3rd parts (separated by commas) are the parameter type, name, and meaning (where "#" means "number of"), respectively. Note, a Boolean type variable takes an integer value 1 or 0 for TRUE or FALSE, respectively. Also note that for each line, only the parameter value is necessary, and the comment is just for clarity. More details about each parameter in each line are as follows.

```
1                 !Boolean,  M_ESTIMATE, 1/0 for migration/isolation model
1                 !Boolean,  I_Equilibrium, 1/0 for equilibrium=T/F
50000             !Integer,  MaximumNe, maximal Ne allowed
2                 !Integer,  Monitor, 0/1/2/3 for monitoring level
2                 !Integer,  NumThread, #openMP threads
100               !Integer,  NumLoci, #Loci in genotype data
MYDATA.dat        !String,   DataFile, genotype data file name
2                 !Integer,  DataForm, 0/1/2 for GenePop/0123/2Alleles_per genotype
4                 !Integer,  NumSmp, #temporal samples from focal pop
0 1 2 3           !Integers, NumGen, #generations of each temporal sample from focal pop
100 100 100 100   !Integer,  SmplSize1, #indivduals of each temporal sample from focal pop
```

```
100                !Integer,  SmplSize0, #indivduals of the sample from source pop
1                  !Integer,  INIT_POINT, #initial random points to be used
```

1. M_ESTIMATE: A Boolean, whether it is the migration model (1) or the isolation model (0). When **M_ESTIMATE** =1, a sample of individuals from the (migration) source population is needed as well as 2 or more temporal samples of individuals from the focal population. Both the N_e and the immigration rate (m) of the focal population will be estimated. When **M_ESTIMATE** =0, only temporal samples of individuals from the focal population are needed, and the N_e of the population will be estimated.

2. I_Equilibrium: A Boolean, whether the population is assumed at migration-drift equilibrium (1) or not (0) when samples are taken. This is required for the migration model. For isolation model, the input has no effect on the analysis, but is still necessary.

3. MaximumNe: An integer, the maximal N_e value allowed in the maximum likelihood estimation. It is suggested to set a value not too big, say 10000. Otherwise, the run can be very slow. The likelihood method uses transition matrix to calculate the likelihood function. The number of elements of the matrix is in the order of N_e^2, and may be so large that exhausts the RAM of your computer if the maximal N_e is set too large. However, the current program can cope with N_e~100000 when run on a decent computer (RAM > 32GB), although computational speed decreases rapidly with increasing N_e. The accepted range of **MaximumNe** is [2, 200000].

4. **Monitor:** An integer, indicates the level of details of the screen output of intermediate computational results for monitoring progress purpose. The output details increase with increased monitor values.

5. **NumThread:** An integer, sets the number of openMP parallel threads to be used in the analysis. The program allows parallel computation using multiple cores in a shared memory computer. The computation time is roughly linearly decreasing with the use of an increasing number of cores (threads) of your computer, when the number of cores is much smaller than the number loci in your data. There is no extra benefit if you use more cores than the number of loci. The number of threads here can be set to any natural number. However, to achieve maximum efficiency, it is suggested to set the value equal to the number of logical cores of your computer or the number of loci, whichever is smaller. If you do not know how many cores of your computer, set a value of zero, so that the program will detect and use all the logical cores available. If you compile the codes with your own Fortran compiler capable of using Openmp, it is suggested to run your executable with the example data using different numbers of threads. If you obtain the same results using different numbers of threads, then you can use the executable to do parallel computation. Otherwise, recompile the program without invoking Openmp. The executable does serial computation.

6. **NumLoci**: An integer, the number of loci genotyped for each sampled individual. All loci are assumed codominant. Monomorphic loci are allowed but will be identified and excluded by the program from the analysis. A locus can have a maximum of 127 alleles.

7. **DataFile:** A string, giving the path and name of the genotype file. The string can have a maximal number of 499 characters, and should not contain illegal characters for file/folder names (such as “%”, “#” or “&”).

8. **DataForm:** An integer, indicating the format of the genotype data. **DataForm**=0/1/2 is for GenePop format, 0123 format, and 2 allele/genotype format, respectively. Details of the 3 genotypes formats are described below.
9. **NumSmp**: An integer, the number of temporal samples taken from the focal population. It must be equal to or greater than 2.
10. **NumGen**: Integers, the generation numbers of each temporal sample from the focal population. Note, generation number is counted from the 1st (earliest) sample, which always has a generation number of 0. Following the 1st number, the *i*th number gives the sampling generation number of sample *i*, with *i*=2, 3, …, **NumSmp**. For example, if we have 3 samples taken every 4 generations, then the input is "0, 4, 8". Please also note that, in reality, the interval between samplings (say, between the 1st and the 2nd sample) may not be an integer (because of overlapping generations). However, MLNe accepts only integer number of generations. In such a case, we can still get an approximate estimate of N_e and m if we are satisfied with the diffusion approximation which usually works very well if N_e is not very small. We can round the sampling interval (say, T) to the nearest integer (say, t) and use it in the estimation. The estimates of N_e and m ($N_e^{'}, m'$) are then converted by $\widehat{N}_e = (\frac{T}{t})N_e'$ and $\widehat{m} = 1 - e^{(t/T)\log(1-m')}$, respectively. For example, the 1st sample has a fixed sampling generation of 0, and the 2nd sample has an actual sampling generation of T=2.8. The nearest integer of T is t=3, which should be used in MLNe analysis. Suppose we get estimates of $N_e' = 80$ and m'=0.05. The final estimate should be $\widehat{N}_e = \left(\frac{2.8}{3}\right) * 80 = 74.7$ and $\widehat{m} = 1 - e^{\left(\frac{2.8}{3}\right)\log(1-0.05)} = 0.0467$.
11. **SmplSize1**: Integers, the numbers of individuals in samples 1, 2, 3, …, **NumSmp** from the focal population.
12. **SmplSize0**: An integer, the number of individuals of the sample from the source population. This is required for migration model. However, for isolation model, an integer number should also be provided, although it has no effect on the analysis.
13. **INIT_POINT**: An integer, the number of random starting points to be used in maximum likelihood analysis. This is required for the migration model. You can choose to use either a single point of the moment estimates or many widely different starting points. From simulation studies, we find that typically the likelihood surface is smooth and has a single peak, which means a single starting point is enough. We suggest here setting this indicator to 1; otherwise, it is much more computationally intensive. For estimating N_e only of an isolated population, this part of data has no effect on the analysis, but an integer number should still be provided.

3.1.2 Data file

A data file named in the parameter file above should be prepared and saved in the project folder. It contains the ID and the genotypes at each locus of each sampled individual. All data for an individual are placed in a single row. Note the first column is reserved for individual ID, which is a string containing NO blank spaces (otherwise it will be taken as two or more columns and runtime error will occur) and which must be unique (i.e. no individuals are allowed to have the same ID). Genotype data per individual are listed from column 2 onwards in one of three possible formats, defined by **DataForm** in the parameter file described above.

When **DataForm** =2, the two observed alleles (represented by two integers, each of the range [0, 32767]) at a locus, separated by a single blank space, are listed in two consecutive columns. A missing allele at any locus is denoted by 0, and a missing diploid genotype is denoted by 0 0.
Data for two example individuals at 3 loci are as follows, where the 1st column gives individual name.

```
Bob 110 116 142 148 120 126
Peter 110 120 120 124 150 154
```

Individuals are ordered in the genotype data file as follows. First, list genotype data for all individuals from focal sample 1, and then those from focal sample 2, … At last, for migration model, list genotype data for all individuals from the source population. An example is as follows.

```
S1_1 2 4 7 7 4 8 2 2 1 4 9 9 4 6 6 9 1 3 9 8
S1_2 5 1 5 9 4 8 9 2 6 7 4 9 4 6 9 9 1 2 8 1
……
S1_50 4 1 5 1 5 8 4 4 4 4 4 9 6 8 5 1 2 8 8 8
S2_1 1 7 6 8 8 2 2 4 4 9 6 3 4 8 2 9 3 1 9 9
S2_2 7 4 7 6 3 4 4 8 3 2 2 3 4 4 4 4 1 3 5 9
……
S2_50 9 6 7 6 4 5 6 2 4 6 9 9 4 8 4 6 1 3 9 9
S0_1 4 3 8 7 5 3 3 6 4 7 1 4 4 4 1 4 1 1 8 9
S0_2 8 5 7 7 8 2 2 4 7 4 4 9 4 4 4 9 7 8 8 9
……
S0_50 1 1 6 7 8 4 4 3 5 7 6 4 8 4 9 9 7 8 9 3
```

In the above example, individuals are genotyped at 10 marker loci. The 1st focal sample contains 50 individuals (from S1_1 to S1_50), occupying rows 1 to 50 of the genotype data file. The 2nd focal sample also contains 50 individuals (from S2_1 to S2_50), occupying rows 51 to 100 in the genotype data file. 50 individuals (from S0_1 to S0_50) are also sampled from the source population, occupying rows 101 to 150 in the data file. For all of the 150 rows, the 1st column lists the individual name/ID, which as a string should not contain blank space.

DataForm =1 is used only when all markers are diallelic (such as SNPs). Each genotype is represented as either 0, 1 or 2, indicating the number of reference alleles in the genotype. A missing genotype is represented by 3. There is no blank space between genotypes. For example, the genotype data at 3 diallelic loci for an individual could be

```
Rob 123
```

which means individual Rob has 1 and 2 reference alleles at locus 1 and 2, respectively, and has missing genotype at locus 3. Individuals (or rows) are ordered in the genotype data file as follows. First, list genotype data for all individuals from focal sample 1, and then those from focal sample 2, … At last, for migration model, list genotype data for all individuals from the source population.

DataForm =0 is in GenePop format. The difference is that the 1st column is a string that must NOT contain blank spaces, and there should be NO blank space between the string and the following comma, ",", when it is present. The GenePop formatted line for the above example is

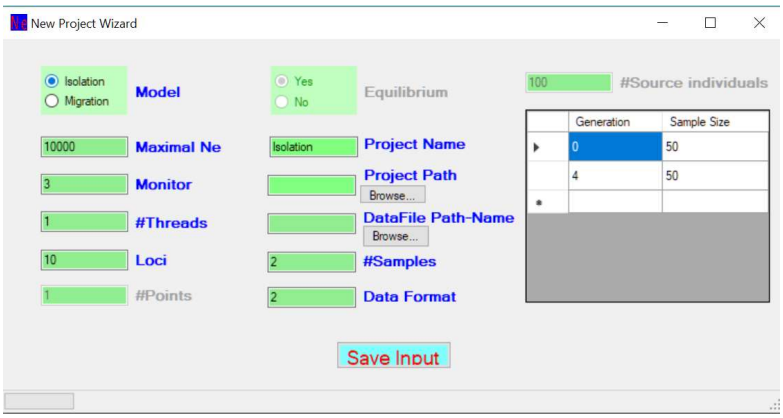
```
Rob, 0102 0101 0000
```

Individuals (or rows) are similarly ordered in the genotype data file.

3.2 Windows GUI front end

3.2.1 Parameter file

The same parameter file as described in "**3.1.1 Parameter file**" is prepared more conveniently by using **MLNe**'s GUI front end (Figure below). The new project wizard is started by clicking "Project" and then "New Project". The parameter values and settings are pre-set for the isolation example dataset (included in the package), except for "Project Path" and "DataFile Path-Name" which are left open to be filled by a user. For a dataset other than the isolation example, all parameters and settings are expected to be changed by a user.



1. *Model*: Two radio buttons are used to choose values for parameter **`M_ESTIMATE`**, whether to use the isolation (**`M_ESTIMATE`**=0) or the migration (**`M_ESTIMATE`** =1) model.

2. *Maximal* N_e: The text box accepts an integer giving the value of parameter **`MaximumNe`**, the maximal N_e value allowed in the maximum likelihood analysis. The range of **`MaximumNe`** is [2, 200000].

3. *Monitor*: An integer giving the value (0, 1, 2, 3) of parameter **`Monitor`**.

4. *#Threads*: An integer giving the value of parameter **`NumThread`**, number of openMP threads. Note, the number of MPI processes to be used in an analysis will be set at runtime.

5. *Loci*: An integer giving the value of parameter **`NumLoci`**, number of loci genotyped for each individual.

6. *#Points*: An integer giving the value of parameter **`INIT_POINT`**. The text box is automatically disabled when the model is set as Isolation.

7. *Equilibrium*: Radio buttons to choose values for parameter **`I_Equilibrium`**. The buttons are automatically disabled when the model is set as Isolation.

8. *Project Name*: The text box accepts a string that will be used as output file name and the project folder name. On completing parameter input using the wizard, a parameter file *.par and a data file *.dat will be written to the project folder, where * is Project Name provided in this text box. The analysis results will be written to the output file named *.mNe in the project folder.

9. *Project Path*: The text box accepts a string specifying the path of the project. A project folder with the above specified project name will be set up in the project path specified. The output file, on completing an **MLNe** analysis, will be written into this project folder. You can also use the Browse… button to locate the path where to make the project folder (path).

10. *Data File Path-Name*: A string specifying the path and name of the original data file. You can also use the Browse… button to locate the file. The file will be loaded by **MLNe**, scanned quickly for validity, and then saved in the project folder with name *.dat.

11. *#Samples*: An integer giving the value of parameter **`NumSmp`**, number of temporal samples from the focal population.

12. *Data Format*: An integer giving the value of parameter **`DataForm`**.

13. *#Source Individuals*: An integer giving the value of parameter **`SmplSize0`**. The text box is automatically disabled when the model is set as Isolation.

14. *DataGridView*: It is used to get inputs for sampling generations (column 1, for parameter **`NumGen`**) and number of individuals of each temporal sample (column 2, for parameter **`SmplSize1`**) from the focal population. There should be exactly **`NumSmp`** non-empty rows on completing the input. The sum of column 2 gives the total number of individuals sampled from the focal population. This number, plus **`SmplSize0`** if the Migration model is chosen, should be the number of non-empty rows of the genotype data file.

On completing the inputs, click the "Save Input" button for the program to check the inputs and save them in a parameter file *.par and in a data file *.dat in the project folder. Any data errors detected will be reported and you need to correct and re-enter the data.

4 Running MLNe

Once valid parameter file and data file have been prepared, you can conduct the analysis by **MLNe**.

4.1 No Windows GUI front end

1. Open a DOS window (for PC, using Windows command cmd.exe), or a Linux' or Mac's x-terminal. For the latter 2 cases, it is better to get the administrator's privileges (e.g. using "sudo -s" on Mac) to run the commands smoothly without permission problems.

2. Navigate to your project folder (where the parameter file *.par and data file *.dat are found) by using command "cd".

3. Start a run without MPI by the following command line

```
mypath\mNe5 MyData.par                    (for DOS)
mypath/mNe5_nompi MyData.par              (for Mac)
```

```
mypath/mNe5_nompi MyData.par                    (for Linux)
```

where "mypath" is the path of the binary `mNe5` or `mNe5_nompi` and "MyData.par" is the name of the parameter file (which should be in the project folder).

If you have already set the **MLNe** program folder in the environment variable of your computer after installation of the program, you can navigate to (using command cd) your **MLNe** project folder and start a serial run by command line

```
mNe5 MyData.par                                 (for DOS)
mNe5_nompi MyData.par                           (for Mac)
mNe5_nompi MyData.par                           (for Linux)
```

Please note that, without MPI, you can still use openMP to make a parallel run (using multiple cores with shared memory) with multiple threads if the parameter **`NumThread`** is set a value larger than 1 in the parameter file.

4. Start an MPI run with M parallel processes by the following command line

```
mpiexec -np M mypath\mNe5 MyData.par 1           (for DOS)
mpirun -np M mypath/mNe5_ompi MyData.par 1       (for Mac)
mpirun -np M mypath/mNe5_impi MyData.par 1       (for Linux)
```

where "mypath" is the path of the binary, "MyData.par" is the name of the parameter file (which should be in the project folder), and "1" after the parameter file means MPI parallel processes will be used. Note, option "1" should always be used for MPI run (with command mpiexec or mpirun), and the options "1", "0" or "" (nothing) have no effect for non-MPI run (without command mpiexec or mpirun). `mNe5` and `mNe5_impi` are compiled by Intel Fortran compiler and linked to Intel's MPI Library for Windows and linux respectively, and `mNe5_ompi` is compiled by GFortran compiler and linked to openMPI for Mac. Similarly, if you have openMPI 3.1.4 installed on your Linux machine, you can use the binary mNe5_ompi, which is also included in the downloaded package for linux.

The MPI version of MLNe may not work on Linux and Mac machine when no mpi is installed on the machine, or when the installed MPI differs from that used in the generating the binary. For this reason, I have included in the downloaded package the Fortran code so that you can compile and link to your installed MPI library.

If you have already set the **MLNe** program folder in the environment variable of your computer after installation of the program, you can navigate to (using command cd) your **MLNe** project folder and start an MPI parallel run by command line

```
mpiexec -np M mNe5 MyData.par 1                  (for DOS)
mpirun -np M mNe5_ompi MyData.par 1              (for Mac)
mpirun -np M mNe5_impi MyData.par 1              (for Linux)
```

Please note that you might be actually using both MPI and openMP for parallel computation. In the above example, you use M parallel MPI processes and **NumThread** openMP parallel threads per

process. The total number of parallel threads is thus M×**NumThread**. Between openMP and MPI, it is more efficient (with less cost) to use the former. If your computer has multiple cpus/cores with shared memory, it is better to use openMP rather than MPI. However, if you have a cluster with nodes of distributed memory, you may have to use MPI for parallelization. In such a case, suppose a node has a maximal number of X logical cores (with shared memory), you can set **NumThread=X** in parameter file, and start a MPI run with command line **`mpirun -np M mNe5_impi MyData.par 1`**. The total number of parallel threads you are using is XM. Please also note that both openMP and MPI parallelization is over loci in calculation the likelihood. Therefore, for maximal efficiency, the total number of parallel threads, XM, should not be larger than the number of loci.

4.2 Windows GUI front end

Click the "Run", "Start Running" buttons to start an analysis. You can stop the run by clicking "Run", "Stop Running" buttons. If the number of logical cores of your computer, *N*, is not smaller than 2 ×**NumThread** (where **NumThread** is the number of openMP parallel threads set in your parameter file), then you will be asked to input the number of MPI processes to be used in the analysis. The number should be in the range [1, *N*/**NumThread**].

5 Output File

A single output file, named as *.mNe where * is the project name, will be generated in the project folder on the completion of an analysis. It has the following information.

-Point estimates and 95% confidence intervals of N_e and m by the maximum likelihood method. Note that if the likelihood point estimate of N_e is very close to the maximal N_e value set in your data (or by the program), then the estimate is obviously not a point estimate. Rather it indicates that N_e estimate is at least the maximal value.

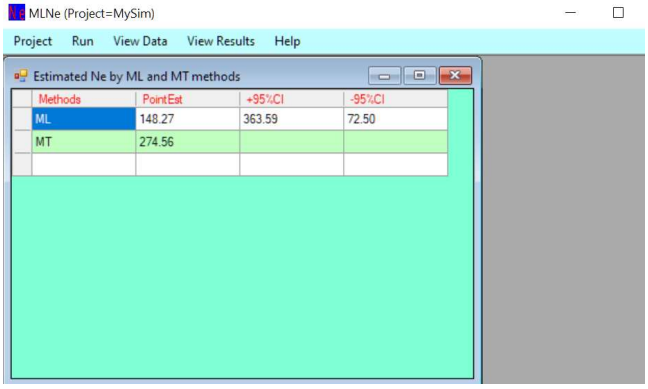
-Relative profile log-likelihood as a function of N_e.
-Relative profile log-likelihood as a function of m.
-Point estimates of N_e and m by the moment method.
-Estimates of allele frequencies at each locus of each sample.
-Date and time when the analysis is started and finished.

6 Output in GUI front end

The output can be viewed in tables and graphs as well as texts in **MLNe's** GUI front end. The graphs can be sized, saved to a file, and copied to clipboard for pasting (Ctrl V) to a document.

6.1 Estimated N_e by ML and MT methods

An example is shown below.

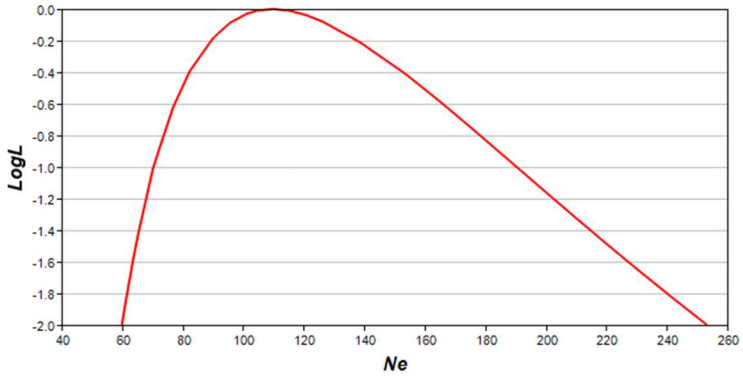


6.2 Estimated *m* by ML and MT methods

For the migration model, estimates for *m* can be viewed similarly to those for N_e as shown above.

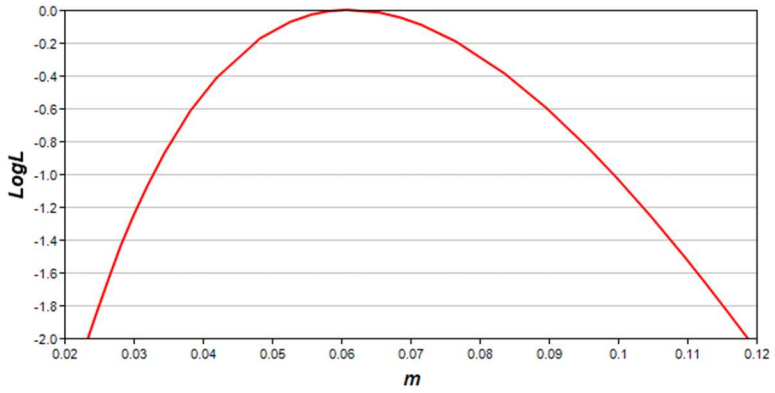
6.3 Plot of N_e profile log likelihood

An example is shown below. It shows the estimated 95% confidence interval for N_e is [60, 252].



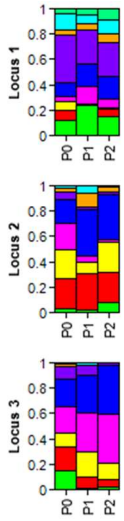
6.4 Plot of *m* profile log likelihood

For the migration model, the profile log likelihood is also estimated *m*. An example is shown below.



6.5 Plot of allele frequencies

Allele frequencies at each polymorphic locus of each sample are estimated by **MLNe**, and can be viewed in a table or in a stacked bar chart. An example is shown below, where P0, P1 and P2 signify allele frequency in the source sample, focal sample 1 and focal sample 2.

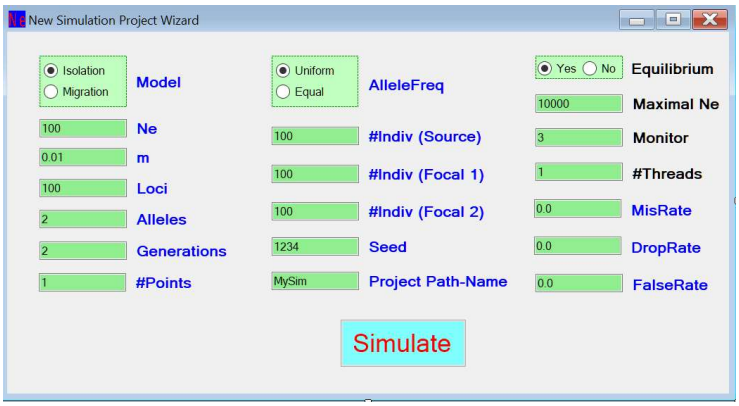


7 Simulations in GUI front end

MLNe's GUI front end for Windows can also be used to simulate temporal genotype data in the isolation or migration model. It simulates individual (unlinked) multilocus genotypes generation by generation, following the Wright-Fisher model with or without migration. The simulated data can then be analysed by **MLNe** to see if the simulated parameters, N_e and m, are recovered or not. They can also be used in understanding the power and accuracy of **MLNe,** in evaluating data (sampling scheme) sufficiency, and in optimising sampling design (such as sample sizes, sampling intervals and number of markers). In the planning stage of a study of population demography when one has no genotype data, one can still use the simulation to get a feel of what will be expected on completing the study.

7.1 Parameter input

By clicking "Project" and "New Simulation", a new simulation project wizard window shows up, as shown below. In the window, the wizard asks some parameters which are then written to a parameter file named "`SimPara.txt`". The file is saved in the project folder, and is used by the simulation program "SimNe.exe" to generate simulated genotype data.



The information required by the new simulation project wizard is as follows.

(1) *Model*: This is the same as that in ***3.2.1 Parameter file***.

(2) N_e: The text box accepts an integer in the range [2, 100000], specifying the effective population size, N_e, to be simulated.

(3) *m*: The text box accepts a real number in the range [0, 1], specifying the immigration rate into the focal population every generation. When the model is specified as Isolation, this text box will be disabled automatically.

(4) *Loci*: An integer specifying the number of loci to be simulated, in the range of [1, 100000000].

(5) *Alleles*: An integer specifying the number of alleles, in the range of [2, 127], per locus to be simulated.

(6) *Generations*: An integer specifying the number of generations (in the range [1, 10000]) for the interval between the 1st and 2nd samples taken from the focal population. The simulation module assumes only 2 temporal samples are taken from the focal population.

(7) *#Indiv (Source)*: An integer specifying the number of individuals (>0) sampled from the source population. This text box is automatically disabled when the model is specified as Isolation.

(8) *#Indiv (Focal 1)*: An integer specifying the number of individuals (>0) in the 1st sample from the focal population.

(9) *#Indiv (Focal 2)*: An integer specifying the number of individuals (>0) in the 2ed sample from the focal population.

(10) *Seed*: An integer giving the seed for random number generator.

(11) *Project Path-Name*: A string (of maximal length 999) specifying the project path and name.

(12) *#Points*: An integer giving the value of parameter **INIT_POINT** as in ***3.2.1 Parameter file***, to be used in analysing the simulated data by MLNe. This text box is automatically disabled when the model is specified as Isolation.

(13) *AlleleFreq*: Radio button options for initial allele frequency distribution, Uniform or Equal. With Uniform, initial allele frequencies are drawn from a uniform distribution. With Equal, initial allele frequencies are equal to $1/k$, where k is the number of alleles.

(14) *Equilibrium*: The same as in ***3.2.1 Parameter file***, to be used in analysing the simulated data by MLNe. This is automatically disabled when the model is specified as Isolation.

(15) *Maximal* N_e: The same as in ***3.2.1 Parameter file***, to be used in analysing the simulated data by MLNe.

(16) *Monitor*: The same as in ***3.2.1 Parameter file***, to be used in analysing the simulated data by MLNe.

(17) *#Threads*: The same as in ***3.2.1 Parameter file***, to be used in analysing the simulated data by MLNe.

(18) *MisRate*: A real number in the range of [0, 0.99] specifying the rate of missing genotypes. For a simulated genotype at a locus of a sampled individual, a random number R uniformed distributed in the

range [0,1] is generated and compared with MisRate. The genotype is taken as non-missing when R>MisRate, and is taken as missing when R<=MisRate. In the former and latter cases, the genotype is unchanged and set as {0, 0} respectively.

(19) *DropRate*: A real number in the range of [0, 0.99] specifying the rate of allelic dropouts. I assume at most only one of the two alleles at a locus drops out during the genotyping process (e.g. PCR, or NGS). Therefore, homozygous genotypes are not affected by dropouts. For a simulated heterozygous genotype at a locus of a sampled individual, a random number R uniformed distributed in the range [0,1] is generated and compared with DropRate. The dropout is taken to be absent when R> DropRate, and to be present when R<= DropRate. In the former case, no dropout occurs and the genotype is unchanged. In the latter case, one of the two alleles is selected at random to drop out and the final observed genotype is a homozygote of the allele that has not dropped out.

(20) *FalseRate*: A real number in the range of [0, 0.99] specifying the rate of false alleles. For each allele in a genotype at a locus of an individual, a random number R uniformed distributed in the range [0,1] is generated and compared with FalseRate. The allele is determined to be not affected by false-allele genotyping errors when R>FalseRate. Otherwise, the allele is set to a randomly selected allele, with all possible alleles at the locus are equally likely selected.

7.2 Running simulation

On completing the input, you can click “Simulate” button to (1) check the inputs, (2) save the inputs to a file named “SimPara.txt” in the project folder, and (3) simulate genotype data. If any errors were found in checking the input, a message will appear. In such a case, you need to correct the inputs before proceeding. If no errors were met, the simulated genotype data will be saved to a data file *.dat, the parameter values for running **MLNe** will be saved to a parameter file *.par, where “*” is the project name.

8 Literature

Wang J. 2001. A pseudo-likelihood method for estimating effective population size from temporally spaced samples. *Genetics Research* 78: 243-57.

Wang J, Whitlock MC. 2003. Estimating effective population size and migration rates from genetic samples over space and time. *Genetics* 163: 429-46.

Nei M, Tajima F. 1981 Genetic drift and estimation of effective reach population size. *Genetics* 98: 625–640.

Wang J. 2022. MLNe: Simulating and estimating effective size and migration rate from temporal changes in allele frequencies. *Journal of Heredity*.