PopCluster 1.5 Software Manual

September 15, 2024

Jinliang Wang

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

CONTENTS

1 Introduction

PopCluster is a computer program implementing the methods described in Wang (2022) for population structure inferences from multilocus genotype data. Briefly, the methods assume the mixture model of Pritchard, Stephens and Donnelly (2000) to conduct a maximum likelihood clustering analysis by the simulated annealing algorithm (Kirkpatrick et al. 1983), classifying the sampled individuals into *K* clusters with each representing a population. Assuming their admixture model, the methods further infer the admixture proportions of (i.e. proportional contributions of the *K* source populations to) each sampled individual, using the clustering analysis results as the initial configuration in an expectation maximization (EM) algorithm modified from that of Tang et al. (2005). The fast algorithms adopted for both clustering and admixture analyses, coupled with the use of efficient data coding (using 2 bits to store a diallelic genotype) and the use of parallel computation (both openMP and Message Passing Interface, MPI), enable PopCluster to handle genomic data (with terabytes of genotypes) as well as small multiallelic marker data (such as microsatellites) at ease. The advanced algorithms and models (e.g. allele frequency model and likelihood scaling model) also lead to more accurate population structure inferences, especially in difficult situations such as low differentiation, hierarchical population structure, highly unbalanced sampling or small sample sizes, and many populations (Wang 2022). The methods were implemented in software PopCluster runnable on Windows, Mac and Linux machines. Additionally, the Windows version of PopCluster has an integrated GUI that helps in data input and analysis result visualization. It also has a built-in simulation module to simulate genotype data under the admixture model, the spatial admixture model, the hybridization model and the migration model. The simulated data can then be used to test the performance of PopCluster. With slight modifications, the data can also be used for analysis by other methods such as STRUCTURE.

Details of the methodology and its performance in comparison with other methods can be found in Wang (2022), which should be cited when using the PopCluster software. In this document, I will focus on how to use PopCluster to analyse an empirical dataset, how to conduct a simulation, and how to understand and interpret the analysis results.

2. Installation

PopCluster 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 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 and the large distributed memory (RAM) of a linux cluster.

2.1 Windows version

Windows users should download the zipped file for Windows version of PopCluster to obtain an installation file named “PopCluster1.msi”. Double click this file to start the installation. By default, it will be installed in “C:\ZSL\PopCluster1”. However, you can change the directory where PopCluster 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 PopCluster might be automatically moved to a folder in VirtualStore and, the simulation program may not run properly. It is also suggested that the installation path should not contain blank space, such as “my folder”.

Upon installation, you will find PopCluster executables, user’s manual in PDF, libraries, and a number of example datasets (each in a separate subfolder) in your PopCluster program folder. A shortcut to PopCluster program is also placed in your Windows’ desktop. Double clicking the shortcut will start the program.

PopCluster 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 this by clicking **Start** 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 PopCluster. Occasionally, .Net might be installed but not enabled on your computer. In such a case, you need to enable it for PopCluster’s GUI front end to work.

This PDF file of the user’s manual is included in the package. I suggest printing and reading this document carefully before running PopCluster. It is also highly recommended that the user first tests the program by running at least one of the example datasets provided. The analysis on one’s own dataset using the program is described below.

2.2 Mac and Linux versions

Mac and Linux users should download the corresponding package and unzip it to your desired location (more details can be found in file readme.pdf in the downloaded package). Everything described above, except for the GUI and the simulations, should be included in the package.

It is suggested to include the PopCluster 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=$PATH:prmpath`” (where `prmpath` is the path for PopCluster program folder where the binaries are found) to file .bashrc that will be read when your shell launches. To do so, simply type “nano ~/. bashrc”, append the line, and save the file. The next time you launch your shell, the shell knows where to find PopCluster program (e.g. `PopClusterLnx`) to run an analysis, no matter where (in which folder) the command line invoking `PopClusterLnx` is issued. If you use zsh in Mac, append line “`export PATH=$PATH:prmpath`” to file .zshenv. Similarly, if you run PopCluster in DOS, you need to add PopCluster program folder to the environment variable by using “**Control Panel>System>Advanced System Settings>Environment Variables**”.

2.3 Example datasets

Example datasets are also installed in PopCluster program folder. For Windows, the “Examples” folder has 6 subfolders. The “ant” subfolder has 3 files used for setting up (or running) this empirical data analysis of admixture. The “readme.txt” file briefly describes how to run or set up the project. The other subfolders are for simulating data, with each containing a data file and a readme file. For linux and Mac versions, only the ant dataset is provided.

3 Input Data Files: Empirical Analysis

For a PopCluster 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 PopCluster 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 (quite a few) well organized, it is advised to create a suitably named (a string with no blank 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.PcPjt) needs to be prepared and saved in the project folder. The extension name “.PcPjt” is not compulsory, but is recommended as in Windows GUI it has this fixed extension “.PcPjt”. The file should have exactly 24 text lines in the right order (below), with each line listing a single parameter value. 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 YES or NO, 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.

```
328           !Integer, NumIndiv,     #Individuals
15            !Integer, NumLoci,      #Loci
0             !Boolean, AllSNP,       All loci are SNP (1/0=Y/N)
0             !String,  MisValue,     Missing allele or PopData
111           !Integer, ISeed,        Seed of random number
test.dat      !String,  DataName,     Genotype file name
test          !String,  OutFile,      Output file name
0             !Integer, Scaling,      0/1/2/3/4=No/Weak/Medium/Strong/VeryStrong scaling
1             !Integer, K_Low,        Minimum K value
10            !Integer, K_High,       Maximum K value
5             !Integer, NumRun,       #Replicate runs per K value
0             !Integer, QFindMthd,    Search method (0/1=AssignPrb/Relatedness)
0             !Boolean, OutAlleleFre, Whether to output allele frequency or not (1/0=Y/N)
0             !Boolean, PopDataQ,     Whether PopData available or not (1/0=Y/N)
0             !Boolean, PopFlagQ,     Whether PopFlag available or not (1/0=Y/N)
2             !Integer, Model,        1/2/3/4=Mixture/Admixture/Hybridization/Migration model
0             !Boolean, LocusFst,     Whether to estimate locus F-statistics or not (1/0=Y/N)
0             !Boolean, UseKMeans,    Whether to use K-Means clustering method or not (1/0=Y/N)
0             !Boolean, IndivLocQ,    Whether IndivLoc data available or not (1/0=Y/N)
0             !Integer, DataForm,     0/1/2/3/4/5=Indiv data in 1-row/2-rows/1-col/bytes/.ped/.bed
0             !Boolean, LocusAdmixQ,  Whether to compute individual-locus admixture or not(1/0=Y/N)
0             !Boolean, Mthd_R,       0/1/2: Compute relatedness = No/Wang/LynchRitland
0             !Boolean, KinshipQ,     Whether to infer kinship or not (1/0=Y/N)
0             !Integer, FrePrior,     0/1/2=Undefined/equal/unequal prior allele freq
```

1. NumIndiv: An integer, the number of genotyped individuals to be analysed for population structure. An individual with no data or with missing genotype data at all loci has no information and should be excluded from an analysis. Individuals are assumed diploid at all marker loci. However, haploid individuals can be analysed by coding one allele at each locus as missing.

2. NumLoci: An integer, the number of loci genotyped for each of **NumIndiv** individuals. All loci are assumed codominant. Monomorphic loci are allowed but will be identified and excluded by the program in the analysis. A locus can have a maximum of 127 alleles. More than 127 alleles will be pooled to have 127 alleles.

3. AllSNP: Boolean, whether all markers are diallelic (i.e. SNPs, **AllSNP** =1) or not (**AllSNP** =0). Note, PopCluster is much more efficient in data storage and computation if **AllSNP** =1. With **AllSNP** =1, a diploid genotype is encoded by 0, 1 and 2, denoting the number of reference alleles. A missing genotype is encoded by 3. Therefore, a genotype can be stored in 2 bits, and a 4-byte integer can be used to store 16 genotypes. Also note that, with **AllSNP** =1, a partial missing genotype (with 1 missing allele) is not allowed because it cannot be denoted in 2 bits encoding system (above). It is advised to set **AllSNP** =1 if most loci are indeed diallelic. In such a case, you can set **AllSNP** =1 and use a suitable data format (**DataForm**=0 below, with two numbers to denote a genotype), and a 3-allele (or more alleles) locus will be converted automatically by PopCluster to a diallelic locus (SNP) by merging the rare alleles.

4. MisValue: A string of maximal length 3, denoting the missing allele across loci (when **AllSNP** =0), and the missing value for **PopData** (below). A haploid genotype can be denoted as a partially known diploid genotype containing a missing allele and the observed allele. Note when **AllSNP** =1, **MisValue** is ineffective for a missing allele or a missing genotype, which is always coded as 3. In such a case, **MisValue** signifies the missing value for **PopData** only, if available in data file. Note also that the maximal length of **MisValue** is 3. If you provide a string with more than 3 characters, only the first 3 characters will be used.

5. ISeed: An integer used to seed the random number generator.

6. DataName: A string, specifying the name of the genotype data file. The file must be present in the project folder. If there are no blank spaces within the string, then it can be provided without the enclosing quotation marks. Otherwise (which is NOT recommended), enclosing quotation marks are necessary. For example, the **DataName** could be “My New Test”. If **DataName** is written as My New Test, then the data name will be taken as My, causing errors. It is suggested not to include blank spaces in file or folder names.

7. OutFile: A string, specifying the generic file name for all outputs. On completion, PopCluster outputs a few files to the project folder, all files having this generic name with different extensions (below). Similar rules regarding the use of quotation marks apply to **OutFile** as a string.

8. Scaling: An integer, taking values 0, 1, 2, 3 and 4 to specify no, weak, medium, strong and very strong scaling to be applied in a clustering analysis. Scaling helps to infer structure and admixture accurately when sampling is highly unbalanced (many individuals from one population, but few from another population). Please see "*7. When should I use scaling?*" in frequently asked questions section.

9. K_Low: An integer, the minimum K value to be used in analysis. It must be ≥1 and ≤**NumIndiv**.

10. K_High: An integer, the maximum K value to be used in analysis. It must be ≥ **K_Low** and ≤**NumIndiv**.

11. NumRun: An integer, the number of replicate runs for each K value in the range [**K_Low, K_High**].

12. QFindMthd: An integer, the method for proposing a configuration in clustering analysis. The default is 0 for the assignment probability method, and the alternative is 1 for relatedness method. The former (default) is slightly more accurate than the latter, especially with unbalanced sampling. It is also faster when many individuals (say, hundreds of thousands) are sampled, but becomes slower than the latter with many populations (say, in hundreds or thousands) and many markers (say, in millions).

13. OutAlleleFre: Boolean, whether to output the allele frequencies estimated for each population and locus (=1) or not (=0). For a genomic dataset with many markers, allele frequency estimates can take a lot of space. In such a case, setting **OutAlleleFre** =0 is suggested, except when you do need allele frequency estimates.

14. PopDataQ: Boolean, whether information for a user-defined population-of-origin for each individual is present in data file or not. If **PopDataQ**=1, the data file must contain a user-defined population origin (**PopData,** below) for each individual, together with genotype data. For migration model (**Model**=4, below), **PopDataQ** must be set a value of 1, and no missing value for **PopData** is allowed for any individual.

15. PopFlagQ: Boolean, whether flag information for a user-defined population-of-origin for each individual (**PopData**) is present in data file or not. The flag (**PopFlag**, below) is a Boolean, indicating whether the **PopData** data for an individual is to be used in analysis or not. When **PopDataQ**=0, **PopFlagQ** must be set a value of 0. Otherwise, a runtime error occurs. When **PopFlagQ** =1, **PopFlag** for an individual takes value 1 or 0 to designate that the corresponding **PopData** is and is not used in structure analysis, respectively.

16. Model: Integer, taking values 1, 2, 3, and 4 for a clustering analysis, admixture analysis, hybridization analysis, and migration analysis, respectively. By default, **Model** =2.

17. LocusFst: Boolean, whether to calculate and output (to a file) the F_{ST}, F_{IS} and F_{IT} values for each locus of each inferred population (=1) or not (=0). It is advised to set **LocusFst** =0 for a genomic dataset, as the output can be huge.

18. UseKMeans: Boolean, whether to use the K-Means method for an additional clustering analysis of the genotype data or not. When **UseKMeans** =1, PopCluster will calculate the relatedness for each pair of individuals using their genotype data, yielding a square matrix $R(i, j)$, with i, j = 1, 2, …, **NumIndiv**. Matrix $R(i, j)$ is then used by the K-Means method for a clustering analysis, whose results together with $R(i, j)$ are shown in separate output files (below). When **UseKMeans** =0, PopCluster will NOT calculate $R(i, j)$ and will NOT conduct the clustering analysis by the K-Means method. Note the computation of $R(i, j)$ and the use of K-

Means method can be slow when many individuals (say, millions) are sampled. In such a case, it is advised to set **UseKMeans** =0. Note also that matrix $R(i,j)$ is still calculated even when **UseKMeans** =0, if **FrePrior**=0 (see below) or **Mthd_R** >0 (see below).

19. IndivLocQ: Boolean, whether the sampling location information (longitudinal and latitudinal) of each individual, **IndivLoc**, is present in the data file or not. **IndivLoc** is not used in structure inference. It is used solely for visualizing population structuring in relation to individual geographic locations in PopCluster's GUI.

20. DataForm: Integer, taking values 0, 1, 2, 3, 4 and 5. Values 0, 1 and 2 indicate that the genotypes for each individual are listed (as string) in 1 row, 2 rows, and 1 column in the data file. Value 3 indicates that the genotypes for each individual are listed, as bytes, in 1 row. Values 4 and 5 indicate that individual genotype data are in Plink's .ped file and .bed file format, respectively. If **AllSNP** =1, **DataForm** can take any of the 6 possible values. Otherwise, **DataForm** must be set a value of 0. For **AllSNP** =1, the most space-saving format is **DataForm**=3 or **DataForm**=5, where a single byte encodes 4 genotypes. Note however that, when **DataForm>2**, only genotype data and individual ID (when **DataForm** =4) are provided, and other information (such as **PopData**) is not allowed.

21. LocusAdmixQ: Boolean, whether to estimate and output the admixture proportions for each individual at each locus (=1) or not (=0). For a genomic dataset with many loci, it is advised to set **LocusAdmixQ** =0. Otherwise, the output is huge.

22. Mthd_R: Integer, taking values 0, 1, and 2 to indicate that relatedness is NOT calculated and is calculated by the estimator of Wang (2002) and Lynch and Ritland (1999), respectively. Note, it is better to avoid calculating pairwise relatedness when **NumIndiv** is large, say when **NumIndiv** >65000, because it becomes slow and takes a lot of memory with an increasing value of **NumIndiv**.

23. KinshipQ: Boolean, whether to infer kinship between all sampled individuals or not. The inference takes population structure (admixture) into account by using individual specific allele frequencies, calculated from individual ancestry estimates and population specific allele frequencies (both from admixture analysis). The kinship for full sibs and parent-offspring is expected to be 0.25, irrespective of the population structure. Note, the computational time of a kinship analysis increases quadratically with the number of individuals (N), as kinship for $N(N+1)/2$ dyads needs to be calculated.

24. FrePrior: Integer, taking values 0, 1 and 2 to indicate that allele frequency prior is to be determined by the program, is the Equal Frequency prior and Unequal Frequency prior (default), respectively. In a clustering analysis, the frequency of an allele *x* at a locus in a population is calculated by $(C_x + P_x)/\sum_{j=1}^{J}(C_j + P_j)$, where J is the number of alleles observed in the entire sample, C_x is the count of the *x* allele copies among individuals assigned to the population, and P_x is the prior frequency for allele *x*. With the Equal Frequency prior (**FrePrior**=1), we have $P_x = 1/J$ independent of allele *x*. With the Unequal Frequency prior (**FrePrior**=2), P_x is the frequency of allele *x* in the entire sample. It is found that the Equal Frequency prior leads to more accurate admixture inferences when the sample contains a substantial proportion of highly related individuals (e.g. full or half siblings). Otherwise, the

Unequal Frequency prior gives more accurate results. In the case of unknown (but suspected) family structure of your sample, you can set Allele Frequency Prior=0 such that the program will calculate the relatedness and to detect whether close relatives are prevalent in the sample or not. It will use the Equal Frequency Prior and Unequal Frequency Prior when close relatives are inferred to be frequent and rare, respectively.

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 genotypes at each locus of each sampled individual. Optionally, it could also contain other information, such as user-defined population-of-origin, and flag for user-defined population-of-origin. When **AllSNP** =1, any of the four data file formats described below is accepted. Otherwise, only the first data format (individual data in 1 row) is accepted.

1. Individual genotypes in 1 row (DataForm =0)

This data format is used for both multiallelic and diallelic marker data. It is the only format that allows for a partially missing genotype (i.e. 1 allele missing and 1 allele present). In the parameter file, parameter **AllSNP** can take value 1 or 0.

All data for an individual are placed in a single row, with columns separated by a single blank space. The possible columns are as follows.

IndivID: The 1st column is **IndivID,** which is compulsory. It is a string of a maximal length of 20 characters, specifying the individual ID. The string must NOT contain blank space and other illegal characters (such as /), and must be unique among all sampled individuals (i.e. NO duplications). Any string longer than 20 characters for individual ID will be truncated to have 20 characters.

PopData: The 2nd column is **PopData**, which is optional. It is a string of a maximal length of 20 characters, representing a user-defined population-of-origin. When **PopDataQ** is set a value of 1 and 0 in the parameter file, the column for **PopData** in data file must be present and absent, respectively. When present, **PopData** will be read by the program. It is and is not used in structure analysis when the corresponding **PopFlag**=1 and **PopFlag**=0, respectively. When **PopData**=**MisValue**, the individual's population-of-origin is undefined (unknown). When we are sure of the source populations of some sampled individuals, providing this information to PopCluster would help the clustering and admixture analysis of all individuals, including those with no **PopData** (denoted as **MisValue**) or those with **PopData** but with **PopFlag**=0. When 2 individuals have the same (different) user-defined population-of-origin (**PopData**) and the information is designated to be used in analysis (**PopFlag**=1 for these individuals), then they will always be placed in the same cluster (different clusters) in a clustering analysis, even when their genotype data may indicate the opposite. The use of **PopData** makes it possible for PopCluster to conduct a *supervised* clustering and admixture analysis, using both genotype data and partially known source population data. For a migration analysis with **Model**=4 in parameter file, **PopDataQ** in parameter file must take value 1, and **PopData** for all individuals must be provided without missing data.

PopFlag: The 3rd column, which is optional, lists **PopFlag** which is a Boolean (taking value 1 or 0) indicating whether the corresponding **PopData** is to be used in a structure analysis or not. When **PopFlagQ**=1 and **PopFlagQ**=0 in the parameter file, the column for **PopFlag** in data file must be present and absent, respectively.

IndivLoc: The 4th and 5th columns are optional, listing the sampling location, **IndivLoc**, which should be a pair (GPS coordinates, latitude and longitude) of real numbers. When **IndivLocQ**=1 and **IndivLocQ**=0 in the parameter file, these columns for **IndivLoc** in data file must be present and absent, respectively. Note, **IndivLoc** is not used in a structure analysis; but is used in presenting/visualizing the structure analysis results only in PopCluster's GUI.

Genotype: From the 6th column on, two consecutive columns list the two observed alleles (represented by two strings, each of a maximal length of 3 characters/digits) at a locus, separated by a single blank space. A missing allele at any locus is denoted by the string **MisValue**, and a missing diploid genotype is denoted by **MisValue MisValue**.

Data for two example individuals are as follows.

```
Bob ForestA 1 51.6 31.2 110 116 142 148 120 126
Peter ForestB 0 59.2 22.3 110 120 120 124 150 154
```

Any or all optional columns can be missing from the data file. When no optional data are available, for example, there are only 1+**2NumLoci** columns for an individual, the 1st being **IndivID** and the rest being genotypes (alleles). An example for the above two individuals without any optional columns is as follows.

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

2. Individual genotypes in 2 rows (DataForm =1)

This data format is for **AllSNP**=1 only. It is much more succinct than the 1-row format. The format makes the data storage more efficient, and the reading of data and computation faster.

Two rows are used to list the data for an individual. The 1st row lists data for **IndivID**, **PopData, PopFlag** and **IndivLoc**, with the 1st column compulsory and the rest of the columns optional as described above. The 2nd row lists the genotypes of the individual. The genotype at each locus is encoded by a number 0, 1 or 2, which signifies the number of copies of the (arbitrary) reference allele. A missing genotype is encoded by 3. There is NO separator between genotypes. In the parameter file, therefore, the parameter **MisValue** is irrelevant for missing alleles when **AllSNP**=1; it is used for other missing information (such as **PopData**) only.
Data for two example individuals (5 SNPs, with **AllSNP**=1) are as follows.

```
Bob ForestA 1 51.6 31.2
11013
Peter ForestB 0 59.2 22.3
20101
```

Note, a genotype row has no separators between genotypes. It should have exactly **NumLoci** columns, each being an integer of value 0, 1, 2, or 3.

Individual Bob, for example, is known sampled from a user defined source population (**PopData**=ForestA). This information is to be used in analysis (**PopFlag**=1). The sampling location is (51.6, 31.2). The first 4 SNP genotypes have 1, 1, 0 and 1 reference alleles, and the 5th genotype is missing.

The same data with no optional columns are

```
Bob
11013
Peter
20101
```

3. Individual genotypes in 1 column (DataForm =2)

This data format suits **AllSNP**=1 only. It is much more succinct than the 1-row format.

The 1st row is compulsory, listing the ID, **IndivID**, for individuals [1,**NumIndiv**]. The 2nd row is optional. It is present and absent when **PopDataQ**=1 and 0, respectively. When present, it gives the data of **PopData** for individuals [1,**NumIndiv**]. The 3rd row is optional. It is present and absent when **PopFlagQ**=1 and 0, respectively. When present, it gives the data of **PopFlag** for individuals [1,**NumIndiv**]. The 4th row is optional. It is present and absent when **IndivLocQ** =1 and 0, respectively. When present, it gives the data of **IndivLoc** (a pair of real numbers for GPS coordinates) for individuals [1,**NumIndiv**]. From the 5th row on, each row (compulsory) lists the genotype of each individual at a diallelic locus. The genotype is encoded by 0, 1 and 2, the number of reference alleles in the genotype. The missing genotype is encoded by 3 (i.e. the parameter **MisValue** is irrelevant for a missing allele when **AllSNP**=1). The above example dataset converted to this format is

```
Bob Peter
ForestA ForestB
1 0
51.6 31.2 59.2 22.3
12
10
01
10
31
```

Note, in a genotype row, there are no separators between columns (genotypes). When no optional data are available, the data become

```
Bob Peter
12
10
01
10
31
```

4. Individual genotypes in 1 row of bytes (DataForm =3)

This data format suits **AllSNP**=1 only. Being the most succinct, it is recommended for extremely large genotype dataset, say 10^6 individuals at 10^6 SNP loci, to save disk space and to speed up data reading. This format allows genotype data only, other information such as **IndivID**, **PopData, PopFlag** and **IndivLoc** is not allowed.

An individual's genotypes at L loci are listed in a single row of $L/4$ bytes. Each byte encodes genotypes at 4 loci. The 1st two bits of the 1st byte store the genotype at the 1st locus. The next two bits store the genotype at the 2nd locus, and so on for the genotypes at the 3rd and 4th locus. The second byte stores genotypes at the 5th-8th loci, the third byte stores genotypes at the 9th-12th loci, etc.

The two-bit genotype codes have the following meanings:

00	Homozygous for alternative allele
01	Heterozygous
10	Homozygous for reference allele
11	Missing genotype

Typically, a genotype data file in **DataForm**=3 takes about 1/6 the disk space of a data file in other formats (other than **DataForm**=5).

5. Individual genotypes in Plink's .ped format (DataForm =4)

This data format assumes **AllSNP**=1, and allows genotype and **IndivID** data only. Other information such as **PopData, PopFlag** and **IndivLoc** is not allowed.

A data file in .ped format is generated by Plink. An individual takes one row with 2**L**+6 fields where **L** is the number of loci (SNPs). The 2nd field lists individual ID, the 7th and 8th fields are allele calls for the first SNP ('0' = no call/missing), the 9th and 10th are allele calls for the second SNP, and so on. Field 1 and fields 3-6 are not used by PopCluster.

Data for two example individuals are as follows.

```
1   Bob 0 0 1 1 2 2 2 2 1 2 1 1 1 2
1 Peter 0 0 1 1 1 2 1 1 0 0 1 2 2 2
```

6. Individual genotypes in Plink's .bed format (DataForm =5)

This data format assumes **AllSNP**=1, and allows genotype data only. Other information such as **IndivID, PopData, PopFlag** and **IndivLoc** is not allowed.

A data file in .bed format is generated by Plink. Each genotype is coded by 2 bits in a byte. The first three bytes should be 0x6c, 0x1b, and 0x01 in that order, which are used to specify that the format is SNP-major format. The rest of the file is a sequence of **L** blocks of **N**/4 (rounded up) bytes each, where **L** is the number of SNPs and **N** is the number of individuals. The 1st block corresponds to the 1st SNP, the 2nd block corresponds to the 2ed SNP, etc.

The low-order two bits of a block's first byte store the first individual's genotype. The next two bits store the second individual's genotype, and so on for the 3rd and 4th individuals. The second byte stores genotypes for the 5th-8th individuals, the third byte stores codes for the 9th-12th, etc.

The two-bit genotype codes have the following meanings:

00 Homozygous for first allele
01 Missing genotype
10 Heterozygous
11 Homozygous for second allele in .bim file

If **N** is not divisible by four, the extra high-order bits in the last byte of each block are always zero.

NOTES for **DataForm** >2

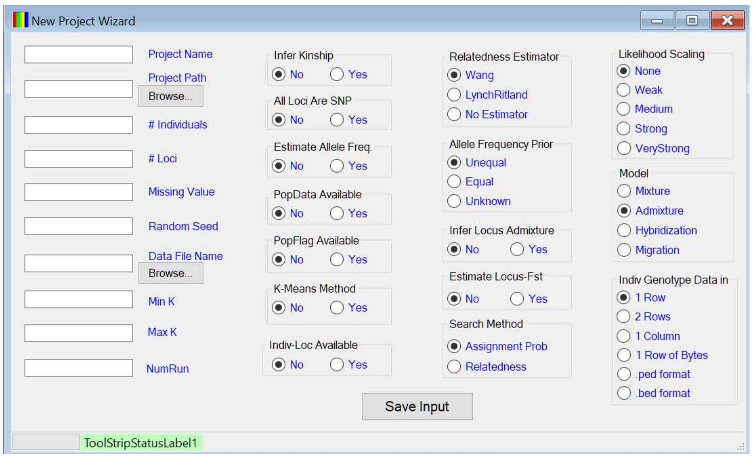
(1) These formats admit genotype data, some formats also admit individual ID. No other information is allowed. Therefore, data in these formats do not suit migration analysis.

(2) In PopCluster's Windows GUI, data in these formats are not checked; they are copied to the project folder without checking the validity. The genotype data cannot by viewed in the GUI.

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 PopCluster's GUI front end (figure below).


New Project Wizard

Project Name
Project Path
Browse...
# Individuals
# Loci
Missing Value
Random Seed
Data File Name
Browse...
Min K
Max K
NumRun

Infer Kinship: [✓ No] Yes
All Loci Are SNP: [✓ No] Yes
Estimate Allele Freq.: [✓ No] Yes
PopData Available: [✓ No] Yes
PopFlag Available: [✓ No] Yes
K-Means Method: [✓ No] Yes
Indiv-Loc Available: [✓ No] Yes

Relatedness Estimator: [✓ Wang] LynchRitland No Estimator
Allele Frequency Prior: [✓ Unequal] Equal Unknown
Infer Locus Admixture: [✓ No] Yes
Estimate Locus-Fst: [✓ No] Yes
Search Method: [✓ Assignment Prob] Relatedness

Likelihood Scaling: [✓ None] Weak Medium Strong VeryStrong
Model: Mixture [✓ Admixture] Hybridization Migration
Indiv Genotype Data in: [✓ 1 Row] 2 Rows 1 Column 1 Row of Bytes .ped format .bed format

Save Input

ToolStripStatusLabel1

1. *Project Name*: The text box accepts a string that will be used as output file names and the project folder name as well.

2. *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. All output files, on completing a PopCluster analysis, will be written into this project folder. You can also use the Browse… button to locate the path where you wish to make the project folder (path).

3. *# Individuals*: An integer giving the value of **NumIndiv**, number of sampled individuals.

4. *# Loci*: An integer giving the value of **NumLoci**, number of sampled loci.

5. *Missing Value*: The text box accepts a string (**MisValue**) specifying the missing allele (when **AllSNP**=0) and the missing value for **PopData** (when **PopDataQ**=1). The maximal length is 3 characters/digits.

6. *Random Seed*: An integer giving the value of **ISeed**, seed for random number generator.

7. *Data File Name*: A string specifying the path and name of the data file. You can also use the Browse… button to locate the file.

8. *Min K*: An integer giving the value of **K_Low**, the minimum K value assumed.

9. *Max K*: An integer giving the value of **K_High**, the maximum K value assumed.

10. *NumRun*: An integer giving the value of **NumRun**, number of replicate runs per K value.

11. *Infer Kinship*: A Boolean (**KinshipQ**) specifying whether to infer kinship by accounting for population structure and admixture or not.

12. *All Loci Are SNP*: A Boolean (**AllSNP**) specifying whether all markers are diallelic (SNPs) or not.

13. *Estimate Allele Freq.*: A Boolean (**OutAlleleFre**) specifying whether to estimate and output allele frequencies at each locus in each inferred population or not.

14. *PopData Available*: A Boolean (**PopDataQ**) specifying whether information for a user-defined population-of-origin for each individual is present in data file or not.

15. *PopFlag Available*: A Boolean (**PopFlagQ**) specifying whether flag information for a user-defined population-of-origin for each individual is present in data file or not.

16. *K-Means Method*: A Boolean (**UseKMeans**) specifying whether to use the K-means method for an additional clustering analysis or not.

17. *Indiv-Loc Available*: A Boolean (**IndivLocQ**) specifying whether individual location information is present in data file or not.

18. *Infer Locus Admixture*: A Boolean (**LocusAdmixQ**) specifying whether to infer admixture at each locus for each individual or not.

19. *Model*: This is the value of parameter **Model**, =1,2,3,4 for mixture, admixture, hybridization and migration analysis.

20. *Likelihood Scaling*: This is the value of parameter **Scaling** for applying scaling (=0,1,2,3,4 for no scaling, weak, medium, strong and very strong scaling) to structure analysis.

21. *Estimate Locus-Fst*: A Boolean (**LocusFst**) specifying whether to estimate and output *F* statistics calculated for each locus of each inferred population or not.

22. *Indiv Genotype Data*: This is the value of parameter **DataForm**, specifying options for six different data formats.

23. *Search Method*: This is the value of parameter **QFindMthd**, whether to use Assignment Probability (default) or Relatedness in proposing configurations of a clustering analysis.

24. *Relatedness Estimator*: The value of parameter **Mthd_R**. This section is enabled for selecting relatedness estimator when **QFindMthd**=Assignment Prob. or K-Means Method=Yes or Allele Frequency Prior=Unknown. Otherwise, it is disabled because relatedness will not be calculated.

25. *Allele Frequency Prior*: The value of parameter **FrePrior**.

3.2.2 Data file

A data file with format specified by **DataForm** should be prepared with name specified in "7. *Data File Name*". PopCluster will load this file and save it to the project folder. When **DataForm**<3, some rather limited check of the format and validity of the data are conducted before saving. When **DataForm**>2, however, data file is not checked and is simply copied to the project folder. There is no guarantee that runtime errors will not occur due to data errors.

3.3 VCF file to genotype input file conversion

PopCluster can convert a VCF file to a given format (individual genotypes in 1 row, 2 rows or one column) and save the reformatted data in a file in the same folder and by the same name (except for the extension .vcf being replaced by .dat) as the original VCF file. A file containing the numbers of loci and individuals of the data is also saved in the same folder and by the same name (except for the extension .vcf replaced by .txt) as the original VCF file. The reformatted .dat file can be used by PopCluster for admixture analysis.

3.4 .Ped file to genotype input file conversion

PopCluster can convert a .Ped file (produced by Plink) to a genotype (individual genotypes in 1 row) data file and save the reformatted data in a file in the same folder and by the same name (except for the extension .dat) as the original .Ped file. Note the genotypes in .Ped file must be coded by integers (i.e. genotypes coded by ATCG are not accepted). Therefore, the option --allele1234 of Plink needs to be invoked in generating the .Ped file.

4 Running PopCluster

Once valid parameter file and data file have been prepared, you can call PopCluster to make a population structure analysis.

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. Enter the PopCluster program folder by using command "cd". This folder is where the PopCluster executable is found. In Windows, it is the folder called PopCluster1. In Mac or Linux, it is PopCluster/Bin.

3. Start a serial run by the following command line

```
PopClusterWin  INP:mypath\MyData.PcPjt                          (for DOS)
./PopClusterMac  INP:mypath/MyData.PcPjt                        (for Mac)
./PopClusterLnx  INP:mypath/MyData.PcPjt                      (for Linux)
```

where "`mypath`" is the path and "`MyData.PcPjt`" is the name of the parameter file (which should be in the project folder, so "`mypath`" should be the project path), "`INP:`" is the command line flag for input (`INP`) file.

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

```
PopClusterWin INP:MyData.PcPjt                                   (for DOS)
PopClusterMac INP:MyData.PcPjt                                   (for Mac)
PopClusterLnx INP:MyData.PcPjt                                   (for Linux)
```

4. Alternatively start an MPI run with M parallel processes by the following command line (started in PopCluster's program folder)

```
mpiexec -n M PopClusterWin INP:mypath\MyData.PcPjt MPI:1              (for DOS)
mpirun -n M ./PopClusterMac_mpi INP:mypath/MyData.PcPjt MPI:1         (for Mac)
mpirun -n M ./PopClusterLnx_impi INP:mypath/MyData.PcPjt MPI:1        (for Linux)
```

Note, parallelization is at the locus level in calculating likelihood and locus specific quantities (e.g. allele frequencies). Therefore, there is no extra gain in using more processes than **NumLoci** or the number of cores of your computer, whichever is smaller. To exploit hyperthreading (two logical cores per physical core) of some CPUs, you can add the run option OMP:2 such that two parallel threads will be generated and used per MPI process in computation. Herein flag "OMP:" sets the number of openMP threads per MPI process.

If the PopCluster program folder is in your computer's environment variable, you can enter your project folder and launch an MPI run with a shorter command line

```
mpiexec -n M PopClusterWin INP:MyData.PcPjt MPI:1                     (for DOS)
mpirun -n M PopClusterMac_mpi INP:MyData.PcPjt MPI:1                  (for Mac)
mpirun -n M PopClusterLnx_impi INP:MyData.PcPjt MPI:1                 (for Linux)
```

Here are the command line flags which can be used in running PopCluster to override the corresponding parameter values set in a parameter file.

(1) "IND:[Number of individuals]": The number of individuals, **NumIndiv**. It must be equal to or smaller than the number of individuals with genotypes in data file. In the latter case, only data for the first **NumIndiv** individuals in the data file will be analysed.

(2) "LOC:[Number of loci]": The number of loci, **NumLoci**. It must be equal to or smaller than the number of markers genotyped for an individual in data file. In the latter case, only genotype data for the first **NumLoci** loci in the data file will be analysed.

(3) "K_1:[Minimum K]": The minimum K value, **K_Low**.

(4) "K_2:[Maximum K]": The maximum K value, **K_High**.

(5) "RAN:[Random number seed]": The seed for random number generator, **ISeed**.

(6) "INP:[Input parameter file name]": The path and name for the parameter file.

(7) "RUN:[Number of replicate runs per K value]": The value of **NumRun**.

(8) "OUT:[Output file name]": The path and name of output files, the value of **OutFile**.

(9) "GEN:[Genotype file name]": The path and name of genotype file, the value of **DataName**.

(10) "SCA:[Scaling (0/1/2/3/4)]": The scaling strength, the value of **Scaling**.

(11) "SKI:[Skip level (0/1/2)]": "SKI:2" is equivalent to this flag missing from the command line. With this option, the program will skip the run if both clustering and admixture analyses are already completed, and will skip clustering analysis and initiate the admixture analysis only if the former is completed only. With option "SKI:1", the program will skip clustering analysis if it is already completed. It will initiate admixture analysis even when it is already completed in a previous run. With option "SKI:0", the program will initiate clustering analysis and then admixture analysis even when both have completed in a previous run.

(12) "MPI:[1/0]": Whether an MPI run (1) or a serial run (0) is to be launched. For an MPI run (launched by mpiexec or mpirun) and a serial run, always use "MPI:1" and "MPI:0" (or no MPI flag), respectively.

(13) "OMP:[#openMP threads]": The number of openMP threads to be used per MPI process.

(14) "KIN:[Estimate kinship (0/1)]": Whether to estimate kinship (1) or not (0, the default).

(15) "RAL:[Relatedness estimator (0/1/2)]": Whether not to calculate relatedness (0), or to calculate relatedness by Wang (2002) estimator (1) or by Lynch and Ritland (1999) estimator (2).

(16) "FRE:[Frequency prior (0/1/2)]": Whether to use the Equal Frequency (1), Unequal Frequency (2) prior, or an undefined prior (0) to be determined by the program using inferred relatedness distribution.

(17) "COP:[1/0]": Whether to output pairwise co-assignment probability for each pair of individuals (1, the default) or not (0).

An example command line in DOS is

```
mpiexec -n 4 PopClusterWin INP:mypath\MyData.PcPjt IND:99 LOC:9 K_1:5 K_2:8
RAN:222 RUN:5 OUT:C:\temp\test\mytest1 GEN:C:\temp\test\myGtype.dat SCA:0 SKI:0
MPI:1 OMP:2
```

Note these flags can appear in any order AFTER the PopCluster program name and can be used in any combinations. Also note these flags have exactly 3 capital letters followed by colon ":" (there is no blank space before or after :).

To show these flags, you can type "`PopClusterWin -help`" in DOS, "`./PopClusterMac -help`" in Mac's X terminal, and "`./PopClusterLnx -help`" in linux's X-terminal.

4.2 Windows GUI front end

The GUI asks you to provide the number of MPI parallel processes to be used in analysing the data. The minimum value is 1, which specifies an MPI serial run. The maximum value is suggested by PopCluster, which is the number of loci (**NumLoci**) or the number of logic cores of your computer, whichever is smaller. Selecting a value less than 1 is illegal, and a value larger than the suggested maximum value has no gains in computational efficiency over that at the suggested maximum value.

Suppose the number of MPI processes you have chosen is n, and the minimum value between the number of loci (**NumLoci**) and the number of logic cores of your computer is m. If $2n \leq m$, then the GUI further asks you to provide the number of openMP parallel threads, t, to be used per MPI process. The suggested range of t values is $[1, m/n]$. Selecting a value less than 1 is illegal, and a value larger than the suggested maximum value m/n has no gains in computational efficiency over that at the suggested maximum value.

All other runtime parameters that could be set without Windows GUI front end as described above cannot be changed at runtime in GUI; they are specified in the parameter file and will be used as such in an analysis.

5 Output Files

Outputs from PopCluster are organized in several files, all in the same project folder you have chosen in setting up the project. In the following, "*" represents the project (or output file) name.

5.1 *_K_x_R_y

These are the (**K_High – K_Low** + 1)×**NumRun** main output files from PopCluster. They have names *_K_x_R_y, where * is the project name, *x* is the assumed K value (K_Low $\le x \le$ K_High) and *y* is the replicate run ($1 \le y \le$ **NumRun**). A file *_K_x_R_y contains the main analysis results obtained in the *y*th replicate run at an assumed *K* value of *x*. It has the following information.

5.1.1 Running parameters and options

This section lists the main parameter values and running options used in conducting the analysis.

5.1.2 Summary of analyses conducted

This section lists the analyses conducted in replicate run *y* with assumed *K* value *x*. These include clustering analysis, admixture analysis, and optionally other analyses such as K-means clustering analysis, relatedness analysis, hybridization analysis, migration analysis and kinship analysis. For each analysis, it lists summary information such as number of iterations, maximum likelihood, and time taken. It also shows the starting and finishing date and time of replicate run *y* with assumed *K* value of *x*.

5.1.3 Clustering analysis results

The 1st, 2nd, 3rd, and 4th columns list the label (**IndivID**), the number of homozygotes, the number of heterozygotes, and the number of missing genotypes of an individual. The 5th column lists the estimated inbreeding coefficient (*F*), the 6th and the 7th (when **UseKMeans** =1) list the cluster index by the maximum likelihood mixture model and the cluster index by K-means method of an individual. When **UseKMeans** =0, the 7th column is absent. Note, the cluster indexes are arbitrary. They signify whether individuals belong to the same cluster (when they have the same cluster index) or not (when they have different cluster indexes). Please note that *F* is calculated assuming the absence of population structure. Therefore, it also includes F_{ST} in the presence of population structure (differentiation). Genotypes at monomorphic loci or with partially missing alleles are also counted as missing. Part of an example is as follows.

```
Label #Hom #Het #Mis        F ML_C
 P1_1  695  292    0  0.0906    1
 P1_2  725  262    0  0.1531    1
 P1_3  699  288    0  0.0841    1
```

5.1.4 *F*-Statistics analysis

The 1st part of this section lists the estimated *F* statistics of each inferred cluster in the final maximum likelihood clustering configuration. An example is as follows.

```
Cluster  #Members        F_ST        F_IS        F_IT
```

```
        1        33    0.0500    0.0197    0.0688
        2        32    0.0632    0.0226    0.0844
        3        27    0.0873   -0.0267    0.0630
        4        28    0.0617   -0.0137    0.0488
```

The 1st column is the cluster index (arbitrary, but consistent across all parts of an output file). The 2nd column is the number of individuals who are inferred to belong to the corresponding cluster. The 3rd, 4th, and 5th columns are the inferred F_{ST}, F_{IS} and F_{IT} values of the corresponding cluster.

The 2nd part of this section lists the F_{ST} values estimated for each inferred population in an admixture analysis. It is the differentiation of a population from the entire set of populations, as shown in the diagonal of a square matrix. It also lists the F_{ST} values between each pair of populations, shown on the non-diagonals of the same matrix. An example is shown below.

```
--------------------------------
Inferred FST between populations
--------------------------------
Pop#    Pop1    Pop2    Pop3    Pop4    Pop5
Pop1  0.0501
Pop2  0.0251  0.0315
Pop3  0.0274  0.0194  0.0409
Pop4  0.0287  0.0185  0.0210  0.0347
Pop5  0.0375  0.0261  0.0301  0.0263  0.0478
```

5.1.5 Admixture proportions

The admixture estimates are in **NumIndiv** rows (one for each individual) and 6+**NumPop** columns. The 1st column is the individual index. The 2nd column is the order of the individual computed from its admixture estimates, which is used by GUI in displaying the ordered admixture estimates in a stacked bar chart. The 3rd column is the label (**IndivID**) of the individual. The 4th column is the genotype data missing proportion. The 5th column is the cluster (index) to which the individual is assigned (the cluster contributing the highest proportion of an individual's genome). The 6th column is a colon. Columns 7 to 6+**NumPop** list the admixture proportions of the individual in the inferred populations (clusters) 1 to **NumPop**. An example is as follows.

```
 Index Order Label %Miss Cluster :  Inferred ancestry in clusters 1~4
     1    11  P1_1   0.0       1 :  0.957 0.005 0.002 0.036
     2    32  P1_2   0.0       1 :  0.354 0.353 0.142 0.151
     3     4  P1_3   0.0       1 :  0.989 0.002 0.008 0.001
     4    25  P1_4   0.0       1 :  0.730 0.090 0.158 0.022
     5    19  P1_5   0.0       1 :  0.864 0.002 0.003 0.131
     6     3  P1_6   0.0       1 :  0.990 0.003 0.002 0.005
```

Note Labels of the 3rd column are replaced by Indexes when there are many individuals, say **NumIndiv** in the millions, to save space.

5.1.6 Coassignment probabilities

This is a **NumIndiv** ×(**NumIndiv** + 1) matrix, listing the coassignment probability (calculated from admixture estimates above) for each pair of individuals. For each individual *i* (=1- **NumIndiv**), a row lists the individual ID (1st column) and the individual's coassignment probability with individual *j* (=1- **NumIndiv**) on columns 2- **NumIndiv**+1. To save space, *n* consecutive items in a row of the same value *x* is denoted as *n*x*, and *x* is converted to an integer. To retrieve the original values, the **NumIndiv** values on a row should be normalized to sum to 1. Apparently this matrix becomes too big and takes too much storage when **NumIndiv** is large. In such a case, the matrix is omitted from the output.

5.1.7 Hybrid classes

The output is conditional on **Model**. It is present only when **Model**>2 (i.e. hybridization analysis or migration analysis). For a hybridization analysis involving 4 or more source populations (i.e. **NumPop** ≥4), I assume an individual's genome comes from a maximum of 4 populations, A, B, C and D, due to hybridization in the 2 previous generations. A hybrid class is thus denoted by the 4 grandparent populations. For example, a hybrid class ABCD means the paternal grandfather, grandmother and maternal grandfather, grandmother (the 1st to the 4th population from left to right) are from populations A, B, C, D respectively. A hybrid class BBBB means all 4 grand parent populations are B, and so an individual of this class is a purebred with 100% of its genome coming from population B. A hybrid class ABCC means the paternal grandparent populations are A and B, and the maternal grandparent populations are C and C, and so an individual of this class has 25%, 25% and 50% of its genome coming from A, B and C, respectively. With 4 possible grandparent populations, there are 55 hybrid classes. This section lists the probabilities that an individual belongs to each of the 55 classes.

5.1.8 Migration rates

The output is conditional on **Model**. It is present only when **Model**=4 for a migration analysis. Migration rates among **NumPop** user-defined populations are calculated using admixture estimates, 0-generation-migrant (F0) estimates, 1-generation-migrant (F1) estimates, 2-generation-migrant (F2) estimates, or a combination (using F0, F1 and F2, or using F1 and F2). An example estimate is as follows.

```
PopIdx     1     2     3     4     5
     1 0.939 0.011 0.003 0.043 0.003
     2 0.075 0.685 0.009 0.225 0.006
     3 0.154 0.080 0.632 0.125 0.008
     4 0.059 0.004 0.008 0.919 0.011
     5 0.040 0.003 0.094 0.192 0.672
```

In the migration rate estimate matrix, element $m(i,j)$ means the rate of migration from population *j* (column, =1-**NumPop**) to population *i* (row, =1-**NumPop**). Apparently, elements on a row sum to 1, as an individual must come from one of the **NumPop** populations. The sum of elements on a column might be >1 or <1, indicating the population corresponding to the column is a migration source population and migration sink population, respectively.

5.1.9 Current effective population sizes

The output is conditional on **Model**. It is present only when **Model**=4 for a migration analysis. Assuming all populations in a meta-population are sampled and in equilibrium in Wright's island migration model, the current effective size of population *i* can be calculated from $F_{ST(i)} = 1/(1+4m_{(i)}N_{e(i)})$, where $F_{ST(i)}$ is the estimated F_{ST} and $m_{(i)} = 1 - m_{(i,i)}$ is the estimated total immigration rate of population *i*. Apparently, $N_{e(i)}$ estimates can be negative when $F_{ST(i)} < 0$. In such a case, $N_{e(i)}$ is set an arbitrarily large value, 10^{10}. Estimates of $m_{(i)}$ can be made from F0-, F1- and F2-migrant estimates, or from F1- and F2-migrant estimates when F0 estimates are deemed unreliable (because too few F0 migrants, or uncertain about whether these migrants are likely to be included in the sample or not). An example is as follows.

```
PopID     Fst     Ne(0-2)     Ne(1-2)
Pop1   0.0191  2.1155E+02  1.4103E+02
Pop2   0.0233  3.3296E+01  2.4342E+01
Pop3   0.0135  4.9732E+01  4.3713E+01
Pop4  -0.0014  1.0000E+10  1.0000E+10
Pop5   0.0267  2.7740E+01  2.0038E+01
```

5.1.10 Locus specific *F*-statistics

The output is conditional on parameter **LocusFst**. It is present in the output file only when **LocusFst** =1. This section lists the estimates of F_{IS} and F_{ST} for each locus in each cluster of the final best (maximum likelihood) clustering configuration. An example for F_{ST} is as follows.

```
Locus  #A    Fst1    Fst2    Fst3    Fst4    Fst5
    1  10  0.0120  0.0169 -0.0112 -0.0053  0.0552
    2  10  0.0135  0.0534  0.0216  0.0060  0.0664
    3  10  0.0129  0.0133  0.0450  0.0015  0.0041
    4  10  0.0039  0.1200  0.0131 -0.0074  0.0385
```

The 1st and 2nd columns list the index and number of alleles for a locus, and columns 3 to 2+**NumPop** list the F_{ST} estimates for populations 1 to **NumPop**. Each F_{ST} estimate is the differentiation of a population from the entire set of populations represented by the sampled individuals, including the population itself.

5.1.11 Allele frequency estimates

The output is conditional on parameter **OutAlleleFre**. It is present in the output file only when **OutAlleleFre** =1. This part of output gives the allele frequency estimates for each locus in each inferred (when **Model**<4) or predefined (when **Model**=4) population, and in the ancestral population (the mean frequency across populations). An example is as follows.

```
Locus 1: 10 alleles, 0.000% missing data
 2 0.067 0.059 0.030 0.104 0.011 0.054
 9 0.073 0.126 0.165 0.147 0.101 0.122
 1 0.149 0.047 0.037 0.123 0.013 0.074
10 0.111 0.208 0.160 0.172 0.056 0.141
 7 0.115 0.015 0.168 0.099 0.331 0.146
 4 0.194 0.068 0.097 0.071 0.086 0.103
```

The 1st column lists allele ID (as used in the original genotype data file when **AllSNP**=0, and always 1 or 2 when **AllSNP**=1). Columns 2 to **NumPop**+1 list the allele frequency estimates for populations 1 to **NumPop** (=5 in this example). The last column lists the allele frequency estimates for the ancestral population.

5.2 *_K_x_R_y.IndLocAdmix

These files are conditional on parameter **LocusAdmixQ**. They are available in the project folder only when **LocusAdmixQ** =1. A file *_K_x_R_y.IndLocAdmix gives the admixture estimates for each locus of each individual obtained in replicate run *y* with an assumed *K* value of *x*. Each file lists the admixture matrix, which has **NumIndiv**×**NumLoci** rows and 2+ **NumPop** columns. In each row, the 1st and 2nd columns list the individual index and locus index respectively, all starting from 1. Columns 3 to 2+ **NumPop** give the estimated admixture proportions in populations 1 to **NumPop**. To save space, when *n* consecutive values in a row are equal to *v*, they are displayed as *n***v*, and *v* is converted to an integer. The **NumPop** admixture estimates for an individual should be scaled to sum to 1. Part of an example output is as follows.

```
1 1 990 10 8*0
1 2 997 0 2 0 1 5*0
1 3 1000 9*0
```

In this example, the admixture proportions of individual 1 (1st column) in population 1 (3rd column) are estimated to be 0.990, 0.997 and 1.000 at locus 1-3 (2nd column) respectively. The assumed number of source populations is 10.

5.3 *_K_x_R_y.Log

This file gives the log of information such as moving (acceptance) rate, log likelihood and F_{ST} during the simulated annealing process of a clustering analysis conducted in replicate y with an assumed K value of x. Columns 1-8 list the K value (x), replicate run (y), annealing temperature, iterates, moving (acceptance) rate, time (ss:mm:hh), current log likelihood, and best log likelihood. Columns 9-(8+**NumPop**) and columns (9+ **NumPop**)-(9+2×**NumPop**) list the F_{ST} values and F_{IS} values for clusters 1- **NumPop**, respectively, estimated at the corresponding iterate.

5.4 *_K_x_R_y.Admix.Log

This file gives the log of information such as tolerance, mean admixture (across individuals) and F_{ST} during the EM process of an admixture analysis conducted in replicate y with an assumed K value of x. Columns 1-4 list the iterates, tolerance, mean admixture (among individuals) and standard deviation of admixture (among individuals). Columns 5~(4+**NumPop**) and columns (5+ **NumPop**)~(5+2×**NumPop**) list the F_{ST} values and QPop (total contribution of a population to the sample) values of each inferred population, respectively, at the corresponding iterate.

5.5 *_K_x_R_y.MidResult and *_K_x_R_y.MidResult1

This pair of files store information of the current clustering configuration produced and saved (every a few minutes) during the simulated annealing (SA) process of a clustering analysis. For a very large dataset with many individuals, many populations and many loci, a clustering analysis by SA could take hours or even days, depending on factors such as the number of MPI parallel processes and openMP threads per process used in the analysis. By saving the intermediate results, one can stop the run and then, after some time, launch the run again. As long as the data and parameters are not changed and the same running command (with options) is used, PopCluster will read the two files and continue the analysis from the breaking point.

5.6 *_K_x.KMeans

This file exists only when parameter **UseKMeans** = 1. It gives the best clustering analysis results of up to 10 replicate analyses by K-means method when *K* is assumed to be *x*. The 1st row lists the values for **NumIndiv**, **NumLoci**, and **NumPop**. From the 2nd row on, it lists the inferred cluster index of individual 1-**NumIndiv**. The sum of squared errors of each replicate run are listed in the main output file, *_K_x_R_y.

5.7 *.Relatedness

This file exists only when parameters **UseKMeans** = 1, or **QFindMthd** =1, or **FrePrior** = 0. It gives the pairwise relatedness calculated by Wang (2002) estimator or Lynch and Ritland (1999) estimator. These estimates are used in the clustering analysis by K-means method when **UseKMeans** = 1, in maximum likelihood clustering analysis when **QFindMthd** =1, and in determining the allele frequency prior when **FrePrior** = 0. Note, this file just gives the upper triangle of the square pairwise relatedness to save space. The elements are $R(i,j)$, where row i=1, 2, …, **NumIndiv** and column j=i, i+1, …, **NumIndiv**.

5.8 *.K

This file gives the estimates of K. The 1st part has 8 columns. Column 1 lists the assumed *K* value. Columns 2,3,4,5 list the output file that has the maximum likelihood, the mean log likelihood, the minimum log likelihood, and the maximum log likelihood, among replicated runs with the same K value. Columns 6,7 and 8 list the first order rate of change of the estimated log-likelihood, the second order rate of change (D_{LK2}), and mean F_{ST}/F_{IS}.

The 2nd part gives the best K estimated from D_{LK2} estimator and F_{STIS} estimator. For determining the best K, it is suggested to make 10 (or more) replicate runs for each possible K value.

The 3rd part lists summary information of each replicate run of each assumed K value, including output file name, time taken for the run, number of iterates completed and maximum likelihood of the clustering analysis. It also includes the time taken, iterates and tolerance of the admixture analysis.

5.9 *_K_x_R_y.Kinship

This file exists only when parameters **KinshipQ** = 1. It gives the kinship estimate calculated for a dyad of individual *i* (=1,2,…,**NumIndiv**) in a row and individual *j* (=*i*, *i* +1,…, **NumIndiv**) in a column, using individual specific allele frequencies calculated from population allele frequency and individual ancestry estimates of the *y*th replicate admixture analysis at *K*=*x*. The estimator is described by Thornton et al (2012). Note, this file just gives the upper triangle of the square pairwise kinship to save space.

5.10 *_K_x_R_y.Q

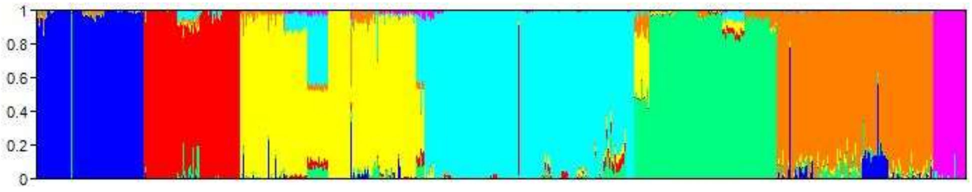
For convenience of importing the admixture estimates to another software for viewing or plotting, the individual admixture estimates (Q) obtained in replicate run *y* with an assumed *K* value of *x* are saved to a csv file named *_K_x_R_y.Q. The file has **NumIndiv** lines. Line *j* lists the ID (0 column) and the ancestry estimates in source populations *k* (=1, 2, …, *K*) of individual *j* (=1, 2, …, **NumIndiv**).

6 Output in GUI front end

The output can be viewed in tables and graphs as well as texts in PopCluster's GUI front end. The graphs can be formatted (in colour scheme), sized, and saved to files or copied to clipboard for pasting (Ctrl V) to a document. For the interpretations of the outputs, please see section 5 above.

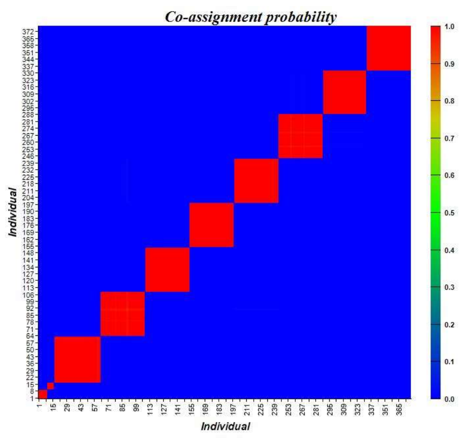
6.1 Admixture

Individual admixture estimates in output file *_K_x_R_y (obtained in the *y*th replicate run assuming K=*x*) can be viewed in a table, a stacked bar chart (sorted or unsorted), and a heatmap. For a bar chart, the user can define (and save) the colour scheme, re-order (and save) population index/label, view the chart in unsorted (individuals on *x* axis in the original order in the genotype data file) or sorted (individuals on *x* axis reordered according to admixture estimates). When **IndivLocQ** =1 (i.e. individual location information available), the admixture estimates can also be plotted in an X-Y pie chart to visualize the relationships between individual locations and clustering structure.



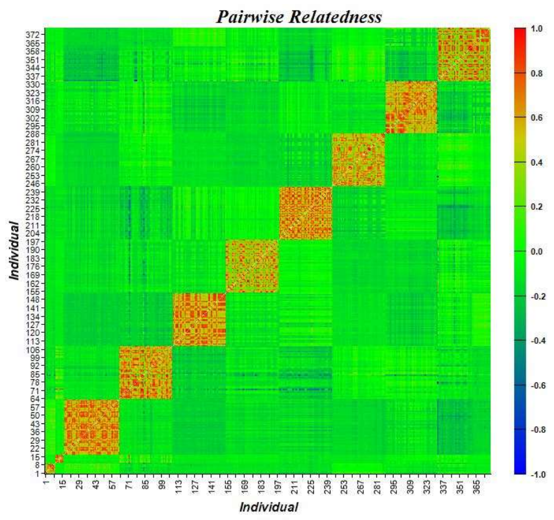
6.2 Coassignment probabilities

These results in output file *_K_x_R_y can be viewed in a table and a heatmap. The ant example (377 individual, 6 loci) is shown in the heatmap below.



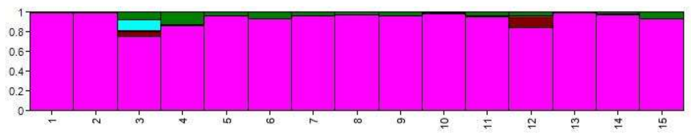
6.3 Pairwise relatedness

The results in output file *.relatedness can be viewed in a table and a heatmap. The ant example dataset shown above for coassignment probabilities is shown in a heatmap for relatedness estimates below.



6.4 Locus admixture

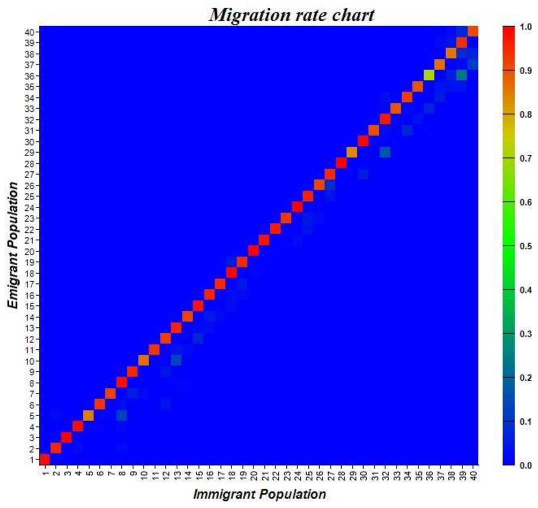
When **LocusAdmixQ** =1, a file *_K_x_R_y.IndLocAdmix exists in the project folder. You can view the results in this file in a table and a stacked bar chart. For the latter, the admixture estimated for each locus is displayed for a single chosen individual. An example is shown below.



In this example, the individual has 15 genotyped loci (on x axis). For each locus, the admixture proportions for K=6 populations (in different colours) are displayed in a stacked bar.

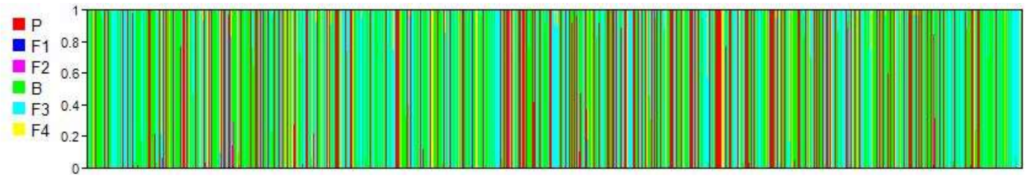
6.5 Migration rates

The results are in file *_K_x_R_y. They can be displayed in a table or a heatmap. An example heatmap for migration rates is below.



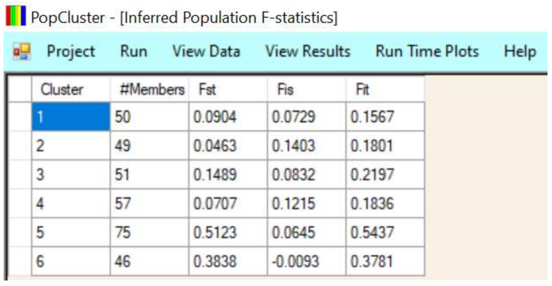
6.6 Hybrid class

The results are in file *_K_x_R_y. They can be displayed in a table or a stacked bar chart. For a clear chart, the many hybrid classes are pooled into 6 categories, P (purebreds), F1, F2, B (any back crosses), F3 (hybrids involving 3 populations, with ancestral contributions 25%, 25%, and 50%), and F4 (hybrids involving 4 populations, each contributing 25%). An example chart is shown below.



6.7 *F* statistics

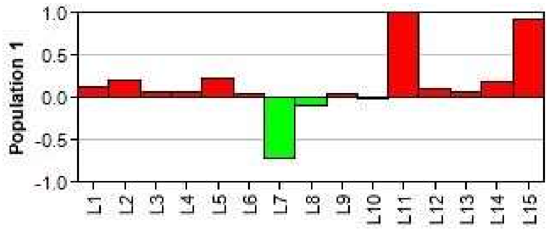
The results are in file *_K_x_R_y. They are displayed in a table. An example is below.



Cluster	#Members	Fst	Fis	Fit
1	50	0.0904	0.0729	0.1567
2	49	0.0463	0.1403	0.1801
3	51	0.1489	0.0832	0.2197
4	57	0.0707	0.1215	0.1836
5	75	0.5123	0.0645	0.5437
6	46	0.3838	-0.0093	0.3781

6.8 Locus F_{ST}

Locus specific F_{ST} estimates for each cluster are listed in file *_K_x_R_y. They can be viewed in a table, a heatmap or a stacked bar chart, as shown below for population 1 and 15 loci (on *x* axis).

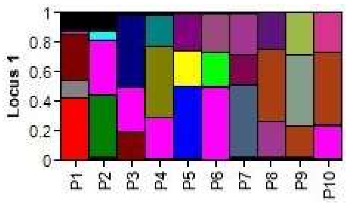


6.9 Locus F_{IS}

Locus specific F_{IS} estimates for each cluster are listed in file *_K_x_R_y. They can be viewed similarly to locus specific F_{ST} estimates.

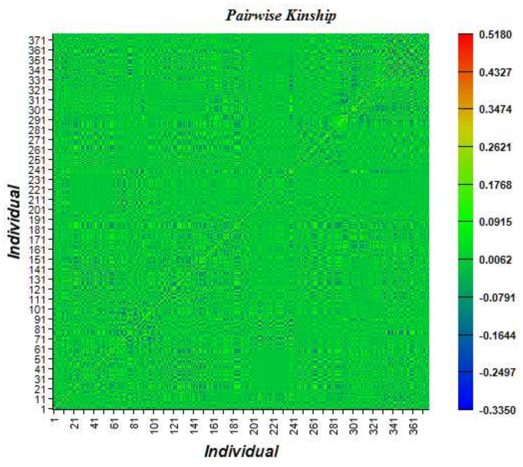
6.10 Allele frequency

Allele frequencies estimated for each locus of each population are in an output file, *_K_x_R_y, when **OutAlleleFre** =1. These estimates can be viewed in a table or a stacked bar chart. An example for the latter is shown below, for one locus and 10 populations (denoted by P1-P10 on *x* axis). Frequencies for different alleles are in different colours.



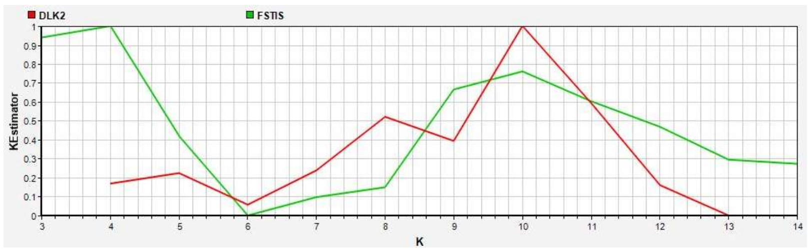
6.11 Pairwise Kinship

The results in output file *_K_x_R_y.Kinship can be viewed in a table and a heatmap. The kinship estimates of the ant example dataset analysed at *K*=10 is shown in a heatmap below.



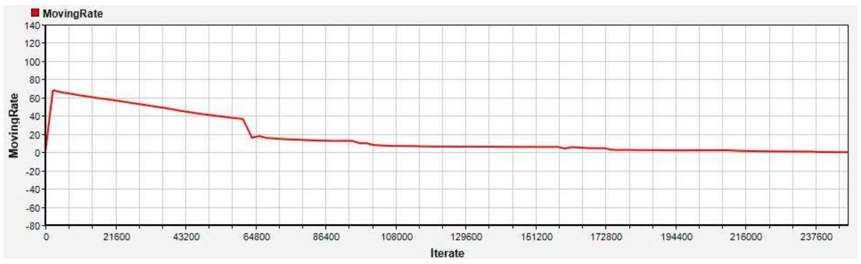
6.12 *K* Estimates

The file, *.K, gives the estimates of K, as described in 5.7 above. The results can be viewed in both a table and a graph. The graph plotting the D_{LK2} and F_{STIS} estimators for the ant dataset is shown below.



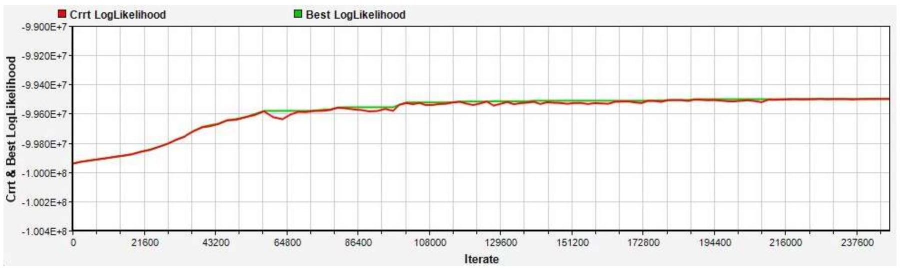
6.13 Moving rate

The moving (configuration proposal acceptance) rate during the simulated annealing process of a clustering analysis are in the set of files *_K_x_R_y.Log. You can choose one of the files to plot this rate as a function of iterates in linear or log scale. One example is shown below.



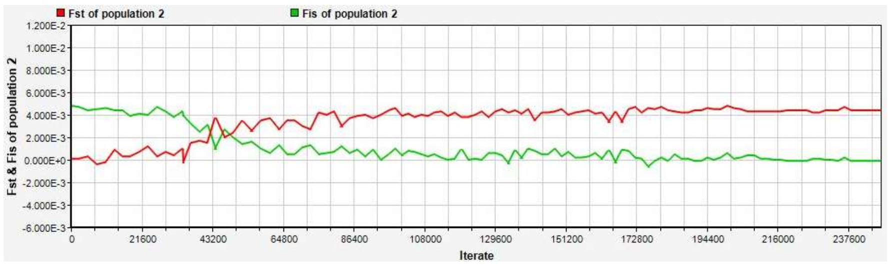
6.14 Log likelihood

The log likelihood values during the simulated annealing process of a clustering analysis are in the set of files *_K_x_R_y.Log. You can choose one of the files to plot these as a function of iterates. One example below shows the current and best log likelihood values.



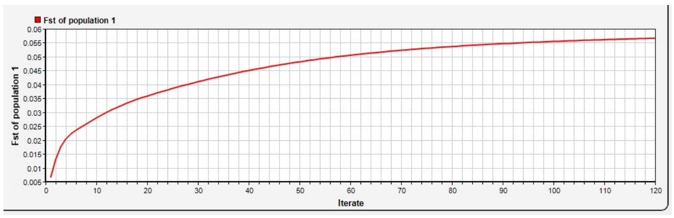
6.15 *F* statistics

The F_{ST} and F_{IS} values of each inferred source population during the simulated annealing process of clustering analysis are in the set of files *_K_x_R_y.Log. You can choose one of the files to plot these as a function of iterates. One example below shows the F_{ST} and F_{IS} values of population 2.



6.16 Runtime statistics in admixture analysis

These statistics calculated during an admixture analysis are in the set of files *_K_x_R_y.Admix.Log. You can choose one of the files to plot these statistics as a function of EM iterates. One example below shows the *FST* of population 1 as a function of the iterations during the EM (expectation-maximization) admixture analysis.



The statistics in *_K_x_R_y.Admix.Log are as follows. (1) Tolerance. It is the error tolerance for admixture proportions, calculated as $\tau_t = \frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}(q_{ik}^{t} - q_{ik}^{t-1})^2$, where q_{ik}^{t} is individual *i*'s ancestry contributed by population *k* estimated in iteration *t* and q_{ik}^{t-1} is the corresponding value in iteration *t*-1. (2) IndivAdmixMean. It is the mean admixture across individuals, calculated by $IAM_t = \frac{1}{N}\sum_{i=1}^{N}(1 - \sum_{k=1}^{K}(q_{ik}^{t})^2)$ at iteration *t*. It measures the average extent of individual admixture. (3)IndivAdmixSD. It is the standard deviation of individual admixture, calculated by $IASD_t = \left(\frac{1}{N}\sum_{i=1}^{N}(1 - \sum_{k=1}^{K}(q_{ik}^{t})^2)^2 - IAM_t^2\right)^{0.5}$. It measures the variation in individual admixture extent. (4)Fst. The differentiation estimated for a population over iterations. (5)PopQ. The proportional genetic contribution to the sample of individuals by a population *k* at iteration *t*, calculated by $PopQ_{k,t} = (\sum_{i=1}^{N} q_{ik})/(\sum_{i=1}^{N}\sum_{j=1}^{K} q_{ij})$.

6.17 Stacked bar charts of admixture from other programs

PopCluster's Windows GUI can also be used to plot stacked bar charts of admixture analysis results from programs STRUCTURE, ADMIXTURE and sNMF. The names of relevant output files from STRUCTURE, ADMIXTURE and sNMF must end with strings "_f", ".Q" and ".Q", respectively. For example, the file could be "testdata_run_1_f" for STRUCTURE and testdata.4.Q for ADMIXTURE or sNMF. The file browser will only search for files with names ending with "_f" for STRUCTURE, and ".Q" for ADMIXTURE and sNMF. To plot these admixture estimates, you need to click ToolStripMenuItem "View Results", "Admixture", and "STRUCTURE" (or "ADMIXTURE" or "sNMF").

7 Simulations in GUI front end

PopCluster's GUI front end can also be used to simulate genotypes and other data under admixture, hybridization and migration models. The simulated data can then be analysed by PopCluster to see if the simulated population structure (e.g. simulated individual admixture, number of populations, F1 or F2 hybrids, migration rates) is reconstructed correctly or not. They can also be used in understanding the power and accuracy of PopCluster and in evaluating data and data sampling scheme sufficiency, etc. With slight modification, the simulated data can also be analysed by other methods, such as STRUCTURE, ADMIXTURE, and sNMF. These methods can then be evaluated and compared using the same simulated data with known structures.

7.1 Admixture model: non-spatial

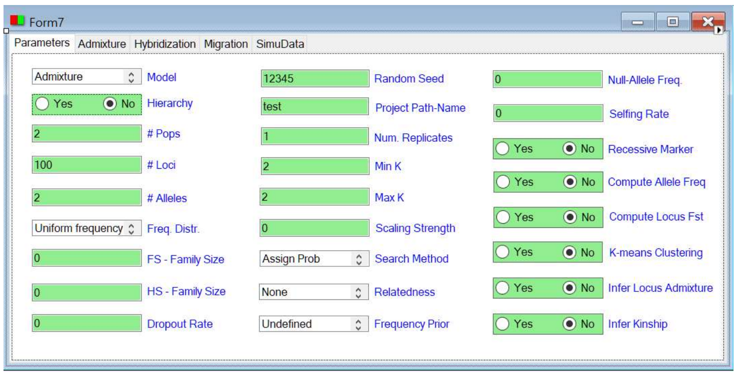
Under this model, we simulate a set of discrete populations differentiated to different extents (measured by F_{ST}) from their common ancestral population. We also simulate a 2-level hierarchical structure, where a set of populations are first subdivided (differentiated) into archipelagos, and then into islands within an archipelago. To keep the hierarchical structure as simple as possible, we assume the number of archipelagos, K_A, is equal to the number of islands, K_I, within each archipelago. Therefore, the number of simulated (island) populations, K, must be a square number such that $K_A = K_I = \sqrt{K}$.

The admixture frequency (the fraction of admixed individuals who have 2 or more recent ancestors coming from different populations) of a sample and the expected admixture extent (the probability that the paternal and maternal alleles at a locus come from different populations) of an admixed individual are moderated by the admixture parameter, as explained in detail in Wang (2022). Other simulation details can also be found there.

By selecting the "New Simulation" MenuItem, you start a new simulation project by first inputing parameters and data, and then running simulations to generate genotype and other data.

7.1.1 Basic parameters

These parameters are, as shown in the figure below, listed in the first tabpage. All parameters are provided with default values, but some are expected to be modified.



Admixture	Model	12345	Random Seed	0	Null-Allele Freq.
Yes [✓ No]	Hierarchy	test	Project Path-Name	0	Selfing Rate
2	# Pops	1	Num. Replicates	Yes [✓ No]	Recessive Marker
100	# Loci	2	Min K	Yes [✓ No]	Compute Allele Freq
2	# Alleles	2	Max K	Yes [✓ No]	Compute Locus Fst
Uniform frequency	Freq. Distr.	0	Scaling Strength	Yes [✓ No]	K-means Clustering
0	FS - Family Size	Assign Prob	Search Method	Yes [✓ No]	Infer Locus Admixture
0	HS - Family Size	None	Relatedness	Yes [✓ No]	Infer Kinship
0	Dropout Rate	Undefined	Frequency Prior		

1. Model: The listbox lists 4 models to select, which are Admixture, Hybridization, Migration and Admixture(Spatial). The first (nonspatial) Admixture model is the default. You need to click the optional model so that the background becomes blue to make it selected.

2. Hierarchy: The radio button allows you to choose the simple island model (default, the "No" option for hierarchy) or the two-level hierarchical structure model (alternative, the "Yes" option for hierarchy). If the latter is chosen, a structure with K_a archipeligos and K_i islands within each archipeligo will be simulated, with a total number of $K = K_a K_i$ islands, where $K_a \equiv K_i$ as explained above. In such a case, the sampled number of populations, **#Pops** below, must be a square number (e.g. 4, 9, 16, 25, …).

3. #Pops: Number of sampled populations.

4. #Loci: Number of sampled loci.

5. #Alleles: Number of alleles per sampled locus.

4. Freq. Distr.: Allele frequency distribution type assumed for simulating the ancestral population allele frequencies at each locus. Five options are provided, which are Uniform, Equal, Triangular, Skewed, and Highly Skewed frequency distributions. For Uniform, I assume $D[1, 1, \ldots, 1]$ where D means Dirichlet distribution. For Equal, I assume all alleles at a locus have an equal frequency. For Triangular, Skewed and Highly Skewed, I assume the frequency of allele i (i=1,2,…,n) at a n-allele locus is proportional to $1/i$, $1/i^2$ and $1/i^4$, respectively, where allele frequencies at a locus are normalized such that they sum to 1.

5. FS-Family Size: The number of siblings in a full-sib family is assumed to follow a Poisson distribution with parameter **FS-Family Size**, a real number ≥0. When **FS-Family Size** = 0, no full siblings will be generated.

6. HS-Family Size: The same as **FS-Family Size**, except for half siblings. NOTE, at most only one of the two types of siblings is allowed to be simulated, which means either **FS-Family Size, HS-Family Size** or both should be equal to 0.

7. Dropout Rate: The rate at which one (chosen at random) of the two homologous alleles at a locus of an individual drops out in the genotyping process such that a heterozygote could be mistyped as a homozygote. This parameter is used in simulating genotype data only.

8. Random Seed: The seed is used in both simulating genotype data and in the PopCluster input parameter file for analysing the data.

9. Project Path-Name: This textbox accepts a string specifying the path and name of the project. An example is "c:\test\mytest", where the path and name of the simulation project are "c:\test" and "mytest", respectively. Note (1) the path, if present, must exist at the time when setting up the project and (2) PopCluster program path would be used by default if the path is absent from the input string. The program will create a project folder with the given name in the given path, and save all outputs from the simulation (e.g. genotype data file) and from PopCluster analysis of the simulated data to this project folder.

10. Num. Replicates: The number of replicate runs per K value to be conducted in PopCluster analysis of the simulated data.

11. Min K: The minimum K value in PopCluster analysis of the simulated data. Note, when **PopData** are available, **Min K** must NOT be smaller than the number of known populations defined in **PopData**.

12. Max K: The maximum K value in PopCluster analysis of the simulated data. Note, **Max K** must be NOT smaller than **Min K**, and must be NOT larger than the number of individuals.

13. Scaling Strength: The parameter value of **Scaling** to be used in PopCluster data analysis.

14. Search Method: The parameter value of **QFindMthd** to be used in PopCluster data analysis. The default is the "Assign Prob" method. When the "Relatedness" method is chosen, the relatedness between pairs of individuals will be calculated and output to a file.

15. Null-Allele Freq.: The frequency of null alleles at each locus to be simulated in generating genotype data. At a locus of any number of alleles, the first allele is designated as the null allele if **Null-Allele Freq**>0. Note for a locus with 2 alleles, **Null-Allele Freq**>0 will lead to monomorphic genotype data. Therefore, null alleles should not be simulated for a diallelic locus.

16. Selfing Rate: The frequency of self-reproduction used in simulating genotype data.

17. Recessive Marker: When **Recessive Marker** is chosen, the first allele at a locus with any number of alleles is always recessive. Note, allelic dropouts, null alleles and recessive markers are not allowed to occur simultaneously. At most, only one of them is allowed to occur (i.e. **Recessive Marker**=True, or **Null-Allele Freq** >0, or **Dropout Rate**>0).

18. Compute Allele Freq: The parameter value of **OutAlleleFre** to be used in PopCluster data analysis. When chosen (Yes option), allele frequencies at each locus in each inferred population will be estimated and saved to a file.

19. Compute Locus F_{ST}: The parameter value of **LocusFst** to be used in PopCluster data analysis. When chosen (Yes option), F_{ST} and F_{IS} at each locus in each inferred cluster will be estimated and saved to a file.

20. K-means Clustering: The parameter value of **UseKMeans** to be used in PopCluster data analysis. When chosen (Yes option), clustering by the K-means method will also be conducted and relatedness between pairs of individuals is calculated, and these results will be saved to a file.

21. Infer Locus Admixture: The parameter value of **LocusAdmixQ** to be used in PopCluster data analysis. When chosen (Yes option), admixture estimates at each locus of each individual will be estimated and saved to a file.

22. Infer Kinship: The parameter value of **KinshipQ** to be used in PopCluster data analysis. With Yes option, pairwise kinship estimates are estimated and saved to a file.

23. Relatedness: There are 3 options, None, Wang and LynchRitland, to choose in PopCluster data analysis, as explained in 3.2.1 Parameter file.

24. Frequency Prior: There are 3 options, Undefined, Equal and Unequal, to choose in PopCluster data analysis, as explained in 3.2.1 Parameter file.

7.1.2 Additional parameters

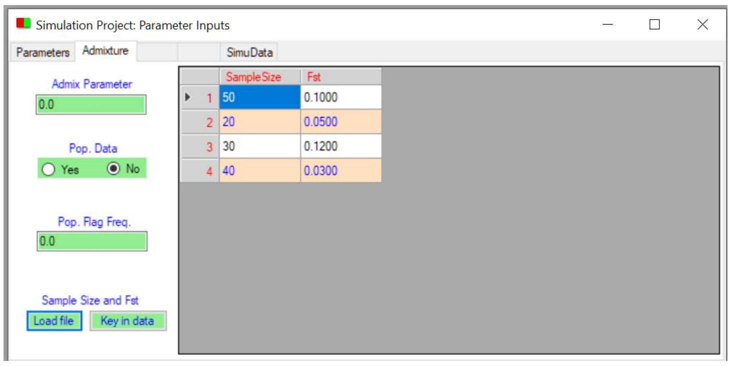
These parameters are listed in tabpage 2, as shown below.

1. Admixture Parameter: It is a real number in the range [0, 1], which determines both the admixture frequency and admixture extent as explained briefly above and in Wang (2022).

2. Pop. Data: Whether to generate **PopData** for each individual or not. If yes, **PopData** will be generated for each individual and included in the genotype data file, and the parameter **PopDataQ** in the parameter input file for PopCluster will be set a value 1.

3. Pop. Flag Freq: A real number in the range of [0, 1]. When **Pop. Data** = Yes, this parameter sets the frequency at which an individual's **PopFlag**=1 (i.e. its **PopData** is to be used in structure analysis). When **PopDataQ** =0, **Pop. Flag Freq** must have a value of 0.

4. Sample Size and *F_{ST}*: Clicking the "Load file" button will load a file into the DataGridView on the right. The file must have **#Pops** (number of sampled populations) rows and 2 columns. The 1st column is sample size (#individuals) and the 2nd is F_{ST} value for each (row) population. Sample size should be a positive integer, while F_{ST} is a real number >0 and < 1. Clicking the "Key in data" button, you will see a table filled with default values of sample sizes and F_{ST}. You are expected to change these values.



Admix Parameter: 0.0

Pop. Data: Yes [✓ No]

Pop. Flag Freq.: 0.0

Sample Size and Fst

	SampleSize	Fst
1	50	0.1000
2	20	0.0500
3	30	0.1200
4	40	0.0300

7.1.3 Simulation

Two buttons are in tabpage 3. The upper one (with text "Check and Save") should be clicked first to check and save the input parameters. If any errors are found in checking the input, a message will appear, and you need to go back to previous pages to correct the inputs. Otherwise, the input will be saved in the project folder with name *.par, where * is the project name. Then you can click the lower button to start simulating data. If no errors are met, the simulation results will be saved to a data file (*.dat), a true (simulated) admixture file (*.truth, listing the simulated admixture proportions of each individual), a pedigree file (*.ped, listing the simulated parent IDs of each individual), and a parameter file (*.PcPjt) in the project folder for use by PopCluster program. To save disk space, the genotype data will be in the two-row format (**DataForm** =1) when the number of alleles per locus is set as 2. Otherwise, the data are in one row format.

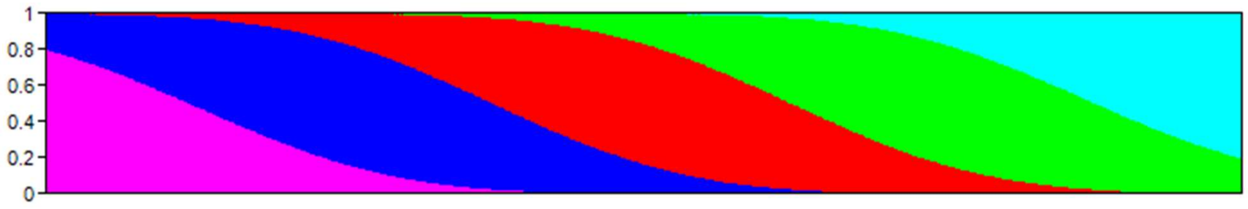
7.2 Admixture model: spatial

The spatial admixture model mimics isolation by distance where population structure changes gradually as a function of geographic location. Under this model, populations are not discrete as assumed by admixture models and have no recognizable boundaries, although clumped sampling (where individuals are sampled from well separated discrete locations) could still incur an artificial discrete population structure represented by the sample. Under the spatial admixture model, ancestral populations (k, =1, 2, …, K) are equally spaced along a line (say, a river in reality). Sampled individuals i (= 1, 2, …, N where N is sample size) are also equally spaced on the same line (i.e. no clumped sampling). The admixture proportions of individual i, $\boldsymbol{Q}_i=\{Q_{i1}, Q_{i2},\ldots, Q_{iK}\}$, are the proportional genetic contributions to i from ancestral populations k, and are a function of the individual's proximity to each population's location. Formally, we have

$$Q^*_{ik}=\left[1-\left(\frac{i-1}{N-1}-\frac{k-1}{K-1}\right)^2\right]^S,$$

$$Q_{ik}=\frac{Q^*_{ik}}{\sum_{k=1}^K Q^*_{ik}},$$

where parameter S is used to determine the admixture extent of the N sampled individuals. Under this spatial admixture model, an individual i's admixture ($\boldsymbol{Q}_i$) is determined by its location, or the relative distances from the K ancestral populations. The 1st and the last sampled individuals (i=1, N) always have the least admixture, measured by $1-\sum_{k=1}^K Q^2_{ik}$. The Q_{11} (=Q_{NK}) is always the largest among Q_{ik} values for i=1~N and k=1~K. Given a desired value of Q_{11} and K, the scaler parameter S can be solved from the above equations. Given K, N and S, the admixture proportions of an individual i (=1~N) can then be calculated from the above equations. As an example, the admixture of a sample of N=500 individuals in the spatial admixture model with K=5 and $Q_{11}=0.8$ is shown below.



To simulate data in the spatial admixture model, the option "Admixture(Spatial)" in **Model** (in **7.1.1 Basic parameters**) must be selected. When this model is selected, the "**Hierarchy**", "**FS-Family Size**" and "**HS-Family Size**" controls are automatically disabled. Furthermore, the sample sizes in TabPage 2's DataGridView must be equal (Note, with this spatial model what matters is the total sample size; populations are not discrete and there is no such a thing as an individual sampled from a population). Also under this spatial admixture model, "Admix Parameter" in TabPage 2 is the value of Q_{11}, which is used in determining parameter S and thus the admixture proportions of each sample individual.

7.3 Hybridization model

Under this model, populations with a given structure and differentiation are simulated as in the non-spatial admixture model. Instead of using the admixture parameter to simulate different admixture frequencies and extents, I simulate individuals of different hybrid classes, given their occurrence frequencies.

7.3.1 Basic parameters

These parameters are in tabpage 1, the same as those for admixture model shown above. You need to click the hybridization option so that the background becomes blue to make it selected.

7.3.2 Additional parameters

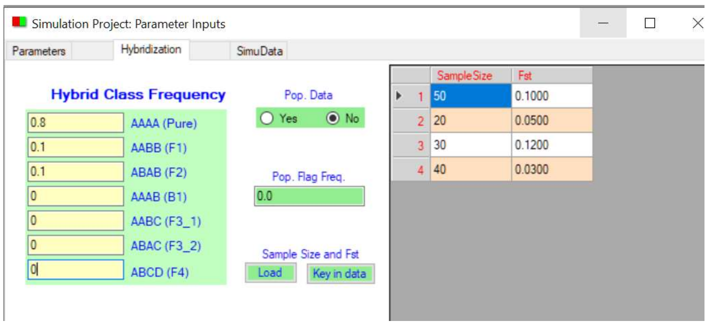
The parameters are listed in tabpage 2, as shown below.

1. Hybrid Class Frequency: The frequency of each of 7 hybrid classes (Purebred, such as AAAA; F1, such as AABB; F2, such as ABAB; B1, such as AAAB; F3_1, such as AABC; F3_2, such as ABAC; F4, such as ABCD), which should be a real number in the range [0, 1]. The 7 frequencies should sum to 1. Hybrid class ABCD means the paternal grandpa and grandma populations are A and B, the maternal grandpa and grandma populations are C and D.

2. Pop. Data: As in 7.1.2.

3. Pop. Flag Freq: As in 7.1.2.

4. Sample Size and Fst: As in 7.1.2.



Hybrid Class Frequency

0.8	AAAA (Pure)
0.1	AABB (F1)
0.1	ABAB (F2)
0	AAAB (B1)
0	AABC (F3_1)
0	ABAC (F3_2)
0	ABCD (F4)

Pop. Data: Yes [✓ No]

Pop. Flag Freq. 0.0

Sample Size and Fst

	SampleSize	Fst
1	50	0.1000
2	20	0.0500
3	30	0.1200
4	40	0.0300

7.3.3 Simulation

As in 7.1.3.

7.4 Migration model

The simulation procedure for migration model is different from that for the other three models as described above. For *n* simulated populations, it requires an immigration matrix ($n \times n$) and a vector (*n* elements) of N_e to conduct an individual-based forward simulation. When reaching the equilibrium between drift and migration approximately, individuals are sampled for structure analysis and immigration rate estimation. The number of sampled populations can be smaller than *n*, the number of simulated populations.

7.4.1 Basic parameters

These parameters are in tabpage 1, the same as those for admixture model shown above. Note, here **#Pops** is the number of sampled populations. It can be equal to or smaller than the number of simulated populations (below). The Hierarchy radio buttons are disabled, as hierarchical structure is not allowed in the migration model. The Min *K* and Max *K* textboxes are also disabled, as a single *K* value equal to the number of sampled populations (**#Pops**) will be used in structure analysis.

7.4.2 Additional parameters

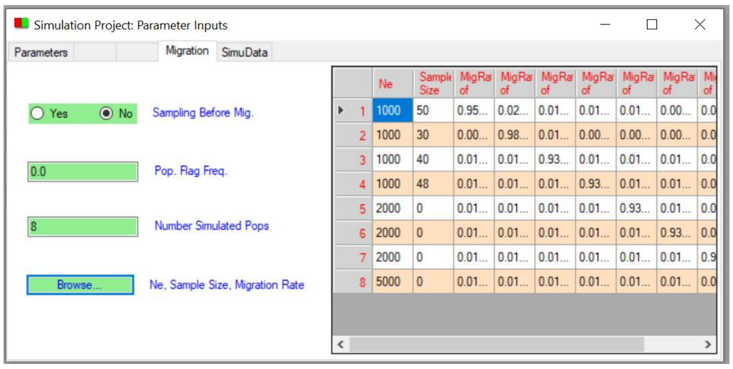
The parameters are listed in tabpage 2, as shown below.

1. Sampling Before Mig: Whether individuals are sampled before migration events or not.

2. Pop. Flag Freq: As in 7.1.2. Note, PopData is always simulated and included in the data file for migration model.

3. Number Simulated Pops: This is the total number of simulated populations that are directly or indirectly connected by migration. The migration rates and N_e of these populations should be provided, which will be used in simulations.

4. *N_e*, Sample Size, Migration Rate: Clicking this button will load a file with the number of rows equal to **Number Simulated Pops**. On each row *i*, the 1st and 2nd columns give the N_e and sample size of population *i*, and column *j*+2 gives the immigration rate into population *i* from population *j* where *j*=1, 2,…,**Number Simulated Pops**. Note, the sampled populations (say, N_{sam}) can be fewer than simulated (say, N_{sim}) populations. In such a case, the first N_{sam} rows give parameters for the sampled populations, and the corresponding sample sizes must be greater than 0. In the example shown in the graph below, N_{sim}=8 and N_{sam}=4, and therefore the first 4 rows must have non-zero sample sizes and the sample sizes on the last 4 rows take no effects (but they must be present).


Simulation Project: Parameter Inputs

Parameters | Migration | SimuData

Yes [✓ No] Sampling Before Mig.

0.0 Pop. Flag Freq.

8 Number Simulated Pops

Browse... Ne, Sample Size, Migration Rate

	Ne	Sample Size	MigRa of	MigRa of	MigRa of	MigRa of	MigRa of	MigRa of	Mi of
1	1000	50	0.95...	0.02...	0.01...	0.01...	0.01...	0.00...	0.0
2	1000	30	0.00...	0.98...	0.01...	0.00...	0.00...	0.00...	0.0
3	1000	40	0.01...	0.01...	0.93...	0.01...	0.01...	0.01...	0.0
4	1000	48	0.01...	0.01...	0.01...	0.93...	0.01...	0.01...	0.0
5	2000	0	0.01...	0.01...	0.01...	0.01...	0.93...	0.01...	0.0
6	2000	0	0.01...	0.01...	0.01...	0.01...	0.01...	0.93...	0.0
7	2000	0	0.01...	0.01...	0.01...	0.01...	0.01...	0.01...	0.9
8	5000	0	0.01...	0.01...	0.01...	0.01...	0.01...	0.01...	0.0

7.4.3 Simulation

As in 7.1.3.

8 Frequently asked questions

8.1 How to determine K?

To determine the most likely number of populations (*K*) explaining the genetic structure of a sample of individuals, you need to make a number (say ≥10) of replicate runs of PopCluster for a *K* value in the range [K_{min}, K_{max}] which is likely to include the true *K*. PopCluster has two built-in *K* estimators. One is based on *F* statistics and the other is based on the rate of change in log-likelihood values. Both could use the analysis results to yield estimates of *K*. It is possible that several different *K* values could explain the data almost equally well. At different *K* values, PopCluster may reveal population structure at different spatial/temporal scales. It is recommended to examine visually the admixture plots obtained under different *K* values.

8.2 Which replicate run should I use?

For any given K value, multiple replicate runs can be conducted. Most often these replicates give very similar results. Occasionally (especially when K is not at the optimum), however, different replicate runs give different results. In such a case, the replicate run that has the maximal likelihood value among the replicate runs of a given K should be chosen to use. The output file *.K lists summary information, including likelihood value, of each replicate run at each K value. Please use this file in choosing the best replicate run.

8.3 Which method (Assign Prob, or Relatedness) to use in a clustering analysis?

Two methods are implemented in PopCluster in searching the best clustering (cluster configuration) by the simulated annealing algorithm. The default method is assignment probability (Assign Prob), and the alternative is pairwise relatedness. The former is slightly more accurate than the latter (especially with tricky data). As to computational efficiency, the alternative method can be more efficient when the assumed K is large (say, >99), the number of markers (M) is large (say, in millions), and the number of individuals (N) is small (say, $N \leq 10K$). Otherwise, the default method runs faster. Never use the alternative method when N is large, say $N > 65521$. This is because a $N*N$ square matrix of pairwise relatedness needs to be calculated, stored and used in making clustering proposals. When N is large, the calculation of the matrix takes a lot of time, and your computer's memory might be too small to store the matrix. When $N = 65521$, for example, the number of elements of the (upper) relatedness matrix is $N(N\text{-}1)/2= 2146467960$, which requires 8.5GB RAM to store them.

8.4 Can I run PopCluster in Linux (Mac) and show the results in Windows?

The Windows version of PopCluster has a built-in GUI that can be used conveniently for setting up the project and for viewing the results in graphs and tables. It can also be used to generate simulated data, and to make serial or parallel (using MPI and openMP) runs of simulated or empirical data. However, the number of cores and the memory (RAM) on a typical PC are rather limited, usually ≤ 8 and ≤64GB respectively. To analyse large genomic data in many GBs, it is better to use the Linux version of PopCluster in a big Linux cluster. You can employ PopCluster's MPI parallel run capability to use as many cores (within and between nodes) and as much distributed RAM as your cluster has. If you wish to use Windows GUI for viewing the analysis results obtained from Linux or Mac, you can first setup your project in Windows. Then copy the data file (*.dat) and project parameter file (*.PcPjt) to your linux (Mac) machine. With slight modification (such as project path), you can run the data in Linux (Mac). On completing the analysis, you can copy all output files to your Windows machine (into the original project folder) for visualization.

8.5 How much RAM is needed?

Determining the amount of memory (RAM) needed to analyse a particular dataset is especially important for a batch run of PopCluster in a Linux cluster. In such a case, you need to allocate an appropriate amount of RAM (as well as the number of parallel threads/cores) in your submitted job. If you allocate too little RAM, the job may fail to run, or may run slowly. If you allocate too much RAM, you may waste the resources (not only RAM but also possibly cores).

The main memories (in bytes) in RAM required by PopCluster are as follows.

Genotype data matrix: $2NM$ and $NM/4$ bytes for N individuals genotyped at a number of M non-SNP (i.e. AllSNP = 0) and SNP (i.e. AllSNP =1) markers, respectively. For non-SNPs and SNPs, 16bits (2 bytes) and 2 bits are used to store 1 genotype, respectively. For example, the SNP data of 10000 individuals at 100000 loci requires 250MB RAM.

Allele counts matrix: 4*KA* bytes for *K* assumed populations and a total number of *A* alleles across loci. For SNPs, *A*=2*M*. Each element of the matrix is a 4-byte integer. For example, 100000 SNPs and *K*=10 populations require 4×10× (2×100000)=8MB RAM.

Allele frequency matrix: 8*KA* bytes for *K* assumed populations and a total number of *A* alleles across loci. For SNPs, *A*=2*M*. Each element of the matrix is an 8-byte double precision real number representing the logarithm of an allele frequency in a population. For example, 100000 SNPs and *K*=10 populations require 8×10× (2×100000)=16MB RAM.

Relatedness matrix: When the search method is relatedness or the K-means method is selected for additional clustering analysis, a relatedness matrix of $2N^2$ bytes will be calculated, stored and used. For example, *N*=10000 individuals require 400MB RAM.

Admixture matrix: 4*NK* bytes for *K* assumed populations and *N* sampled individuals. For example, *K*=10 populations and *N*=10000 individuals require 0.4MB RAM.

Other working space: About 8*KA* + 8*N* bytes.

The total RAM required is the sum of the above 6 parts. For MPI parallel run with *n* processes, the memory required by each process is roughly 1/*n* of the total RAM required.

8.6 Why parallel run with multiple threads/cores is slower than a serial run?

For a computer with a single node (multiple cores with shared RAM), an MPI parallel run with multiple processes or an openMP parallel run with multiple threads could be slower than a serial (single-process and single thread) run. This is because PopCluster has implemented both fine-grained and coarse-grained parallelisms. The former is mainly realized by vectorization using SIMD (Single Instruction Multiple Data). It works no matter the run is serial (single threaded), MPI parallel or openMP parallel. In an MPI or openMP parallel run, marker and other data are partitioned into *n* equally-sized parts with each part being stored and processed by one parallel process or thread. Fine-grained parallelism is efficient (little communication cost), but can only use cores with shared memory (in the same node). MPI coarse-grained parallelism can use an unlimited (limited only by your computer) number of cores in multiple nodes and with shared or distributed RAM. However, there is a cost in communication among processes (MPI) and threads (openMP). Therefore, for a computer with a single node/CPU, a serial run could be already using all of the resources (cores and RAM) efficiently by the fine-grained parallelism. MPI parallel run with multiple processes adds more communication cost and could lead to a slower analysis. For a computer with multiple nodes, however, an MPI run using more processes than the number of cores in a single node should be faster than a serial run, except when the dataset is very small (few loci or/and few individuals). For large genomic data with many GB genotype data, MPI parallel runs in a cluster are much faster than serial runs, and are usually the only feasible approach simply because a node has insufficient RAM for the data.

8.7 What is the capacity of PopCluster?

Theoretically, PopCluster can handle data with a maximum number of 2147483647 sampled individuals and a maximum number of 2147483647 sampled loci. In reality, however, it is constrained by a computer's RAM. For a cluster with many nodes, memory should not be a constraint when MPI parallel run is conducted, with each of *n* parallel processes requiring just 1/*n* of the total memory requested by PopCluster. On a PC with a quadcore and 32GB RAM running Windows 10, I have successfully completed analyses of a simulated dataset with 10^8 individuals (100 SNPs, and *K*=100 populations), of the human 1000 genome phase I data with 38 million SNPs (1092 individuals, *K*=9 populations), and of a simulated dataset with K=10000 populations (10000 SNPs, 100000 individuals). The Windows GUI has much limited ability to handle large datasets. For a large dataset with genotype data in GBs or even TBs, it is better to run the analysis on a cluster using many MPI processes.

When relatedness matrix is calculated and used in the analysis (when **UseKMeans**=1, or **QFindMthd**=1**, or FrePrior** = 0), a *N*×*N* square matrix of pairwise relatedness needs to be calculated and used. When the number of individuals *N* is large, the matrix can be huge, taking a lot of time to calculate and a lot of memory to store it. It is suggested NOT to calculate and use the matrix (by setting **UseKMeans**=0, **QFindMthd**=0 and **FrePrior** >0) when *N*>65521.

8.8 When should I use scaling?

Scaling should be applied to analysing data with highly unbalanced sampling. When few individuals are sampled from one population and many individuals are sampled from another population, the default assumption that an individual comes from each of the *K* assumed populations with an equal *a prior* probability is severely violated. In such a situation, large populations tend to be split while small populations tend to be merged in reconstruction even when markers are highly informative. Applying an appropriate level of scaling can improve the structure inference substantially for the case of unbalanced sampling. The most appropriate scaling level (1, 2, 3 or 4) depends on how unbalanced the sampling is, how much differentiated the populations are, and how much informative the markers are. For example, a low scaling level (1) is appropriate when many markers are genotyped for a set of lowly differentiated (low F_{ST}) populations. Usually, we do not know these factors in analysing the data. Therefore, when the data are suspected to be unbalanced in sampling among populations, they are better analysed with different levels of scaling (0,1,2,3,4). When the applied level of scaling is too low, large populations tend to be split and small populations tend to be merged. When the applied level of scaling is too high, small populations tend to be merged among themselves or with a large population. With the help of some external information (such as sampling locations), the appropriate scaling level can be determined from the admixture assignment analysis.

8.9 What is the difference between relatedness and kinship?

Relatedness between two individuals is estimated by assuming the absence of subpopulation structure. Allele frequencies are first estimated from the entire sample of individuals, assuming they are all not admixed and sampled from a single non-inbred population. The estimated frequencies are then used in estimating pairwise relatedness. For full sibs and half sibs, the expected relatedness values are 0.25 and 0.125, respectively. In the presence of subpopulation structure, however, full sibs and half sibs from non-admixed parents would be expected to have values higher than 0.25 and 0.125, respectively. Kinship between two individuals is estimated by using individual specific allele frequencies calculated from the ancestry and population allele frequency estimates in an admixture analysis. For full sibs and half sibs, the expected kinship is 0.25 and 0.125, respectively, irrespective of the population structure.

8.10 How does PopCluster deal with label switching?

PopCluster handles label switching such that replicate runs with the same *K* value or different *K* values have consistent population labels (1, 2, 3, …, *K*). Suppose the estimated admixture matrix of the current run at an assumed number of *k* populations is $\boldsymbol{Q}_k=\{\mathbf{q}_1, \mathbf{q}_2,\ldots, \mathbf{q}_N\}$ where $\mathbf{q}_i = \{q_{i1}, q_{i2}, \ldots, q_{ik}\}$ for individual *i* (*i*=1,2,…, *N*). The labelling of populations in $\boldsymbol{Q}_k$ is arbitrary and has no biological significance. PopCluster standardizes the labelling of $\boldsymbol{Q}_k$ by maximizing the similarity between $\boldsymbol{Q}_k$ and $\boldsymbol{Q}^*_{k'}$, where $\boldsymbol{Q}^*_{k'}$ is the standardized (indicated by the star in superscript) admixture matrix obtained in a previous run with an assumed number of *k'* (≤*k*) populations. A simulated annealing algorithm is used to find the labelling configuration in $\boldsymbol{Q}_k$ that yields the maximum similarity between $\boldsymbol{Q}_k$ and $\boldsymbol{Q}^*_{k'}$. This $\boldsymbol{Q}_k$ with maximal similarity with $\boldsymbol{Q}^*_{k'}$, denoted by $\boldsymbol{Q}^*_{k}$, is reported. For any range of *K* values (i.e. **K_Low** and **K_High**) and any number of replicates (i.e. **NumRun**) in a PopCluster analysis, the admixture matrix of the first run at *K*= **K_Low** is not standardized (because it has no reference). The admixture matrixes of the subsequent runs are sequentially standardized as described above.

8.11 How to draw stacked bar charts of admixture proportions consistently across replicate runs and K values?

Benefiting from the standardized labelling in PopCluster's admixture analysis, PopCluster can draw stacked bar charts of admixture across runs at the same or different *K* values consistently using a given colour scheme (such that a population represented by a given set of sampled individuals tends to show the same colour across runs and *K* values). To this end, we first choose a colour scheme (a unique colour for each population) for the maximal *K* value (=K_High), and save it (named as, say, colour12 when K_High =12) in the same folder as the admixture output files. Then you can draw bar charts of admixture in any runs with any *K* values by loading and using this colour scheme. When *K*< K_High, only the first *K* colours are loaded and used in the charts.

9 Literature

Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science* **220**: 671-680.

Lynch, M. & Ritland, K. (1999). Estimation of pairwise relatedness with molecular markers. *Genetics* **152**: 1753-66.

Pritchard, J. K., Stephens, M. & Donnelly, P. (2000). Inference of population structure using multilocus genotype data. *Genetics* **155**: 945-959.

Tang, H., Peng, J., Wang, P. & Risch, N. J. (2005). Estimation of individual admixture: analytical and study design considerations. *Genetic Epidemiology* **28**: 289-301.

Thornton T, Tang H, Hoffmann TJ, Ochs-Balcom HM, Caan BJ, Risch N. 2012. Estimating kinship in admixed populations. *The American Journal of Human Genetics* **91**: 122-138.

Wang J. 2002. An estimator for pairwise relatedness using molecular markers. *Genetics* **160**: 1203-1215.

Wang, J. 2022. Fast and accurate population admixture inference from genotype data from a few microsatellites to millions of SNPs. *Heredity* **129**: 79-92.