

Documentation for *structure* software

Jonathan K. Pritchard

Department of Statistics
University of Oxford
1 South Parks Rd
Oxford, OX1-3TG, UK

`www.stats.ox.ac.uk/~pritch/home.html`

May 26, 2000

Contents

1	Introduction	2
1.1	Registering with me	2
2	Getting started	3
3	Format for the data file	3
3.1	Components of the data file:	3
4	Parameters set by User	5
4.1	Parameters in file <i>mainparams</i>	6
5	Interpreting the output	8
5.1	Output to screen during run	8
5.2	Printout of Q	8
5.3	Printout of Q when using prior population information	9
5.4	Printout of estimated allele frequencies (P)	9
6	How long to run the program	10
7	Estimation of K (the number of populations)	10
7.1	Steps in estimating K	11
8	Missing data, null alleles... and other such problems	12
8.1	Dominant loci	13
9	Missing-data bug in original release	13
10	Appendix: Parameters in file <i>extraparams</i>.	15
10.1	Program Options	15
10.2	Using prior population information	16
10.3	Output options	17
10.4	Priors	19
10.5	Further miscellaneous options	19

1 Introduction

The program *structure* implements a model-based clustering method for inferring population structure using genotype data consisting of unlinked markers. The methods used are introduced in a paper by Pritchard, Stephens and Donnelly (Genetics, June 2000), which is the appropriate citation for this program.¹ Applications of our method include demonstrating the presence of population structure, assigning individuals to populations, and identifying migrants and admixed individuals.

Briefly, we assume a model in which there are K populations (where K may be unknown), each of which is characterized by a set of allele frequencies at each locus. Individuals in the sample are assigned (probabilistically) to populations, or jointly to two or more populations if their genotypes indicate that they are admixed. It is assumed that within populations, the loci are at Hardy-Weinberg equilibrium, and linkage equilibrium. Loosely speaking, individuals are assigned to populations in such a way as to achieve this.

Our model does not assume a particular mutation process, and it can be applied to most of the commonly used genetic markers including microsatellites, SNPs and RFLPs, provided that they are unlinked (or at least, not so tightly linked that they are in linkage-disequilibrium).

While the computational approaches implemented here are fairly powerful, some care is needed in running the program in order to ensure sensible answers. For example, it is not possible to determine suitable run-lengths theoretically, and this requires some experimentation on the part of the user. The user should consult the accompanying paper (Pritchard *et al.*, 2000) for a description of the uses and limitations of this method.

This document describes the use of this software, and supplements the Pritchard *et al.* paper.

1.1 Registering with me

I strongly encourage users to send me a brief email at pritch@stats.ox.ac.uk, saying that they have downloaded the program. This will help me to assess the amount of interest in *structure*, and how much effort I should put into maintaining and improving it. I would also welcome a sentence or two saying what organism, and what sort of questions the program is being applied to.

Questions, comments, and bug reports should also be directed to me. I would welcome suggestions about the program and documentation.

¹The software and the manuscript are available from www.stats.ox.ac.uk/~pritch/home.html. (Click on “software”.)

2 Getting started

You will need an executable file: *structure*, two files of parameters: *mainparams* and *extraparams*, and a data file. All of these should be in the same directory (except possibly the data file, provided that you set the appropriate path in *mainparams*). There is a sample data file *testdata*, which you can run the program on for fun, just to check that things are working. There are some platform-specific comments on my webpage about how to begin.

To analyse your own data, you need to prepare a data file in the appropriate format. This is described in Section 3. You will also need to set the values in the file *mainparams* (described in Section 4.1).

A number of program options are provided in the file *extraparams*; these are described in the Appendix. These options can provide considerable flexibility in analysing data. However, the default values in *extraparams* will probably be ok initially.

3 Format for the data file

The format for the genotype data is shown in Table 1 (and Table 2 shows an example). Essentially, the data set is arranged as a matrix in which the data for each individual are stored as two consecutive rows (since many organisms are diploid). The first few columns are for storing several kinds of non-genotype information. After that, each column stores the genotype data for a single locus. The columns are separated by spaces or other whitespace.

The elements of the input file are as follows. If present, they must be in the following order, however some are optional (as indicated) and may be deleted completely. A separate file *mainparams* (see Section 4), allows the user to indicate which of these items are present in the input file. The user must also specify the number of individuals and the number of loci, in *mainparams*.

3.1 Components of the data file:

1. **Label** (Optional) A string of integers or characters used to designate each individual in the sample.
2. **PopData** (Optional) An integer designating a user-defined population from which the individual was obtained (for instance these might designate the geographic sampling locations of individuals).

<i>Label</i>	<i>Pop</i>	<i>Flag</i>	<i>Phen</i>	<i>ExtraCols</i>	<i>Loc 1</i>	<i>Loc 2</i>	<i>Loc 3</i>	<i>....</i>	<i>Loc L</i>
$L^{(1)}$	$g^{(1)}$	$f^{(1)}$	$\phi^{(1)}$	$y_1^{(1)}, \dots, y_n^{(1)}$	$x_1^{(1,1)}$	$x_2^{(1,1)}$	$x_3^{(1,1)}$	\dots	$x_L^{(1,1)}$
$L^{(1)}$	$g^{(1)}$	$f^{(1)}$	$\phi^{(1)}$	$y_1^{(1)}, \dots, y_n^{(1)}$	$x_1^{(1,2)}$	$x_2^{(1,2)}$	$x_3^{(1,2)}$	\dots	$x_L^{(1,2)}$
$L^{(2)}$	$g^{(2)}$	$f^{(2)}$	$\phi^{(2)}$	$y_1^{(2)}, \dots, y_n^{(2)}$	$x_1^{(2,1)}$	$x_2^{(2,1)}$	$x_3^{(2,1)}$	\dots	$x_L^{(2,1)}$
$L^{(2)}$	$g^{(2)}$	$f^{(2)}$	$\phi^{(2)}$	$y_1^{(2)}, \dots, y_n^{(2)}$	$x_1^{(2,2)}$	$x_2^{(2,2)}$	$x_3^{(2,2)}$	\dots	$x_L^{(2,2)}$
\dots									
$L^{(i)}$	$g^{(i)}$	$f^{(i)}$	$\phi^{(i)}$	$y_1^{(i)}, \dots, y_n^{(i)}$	$x_1^{(i,1)}$	$x_2^{(i,1)}$	$x_3^{(i,1)}$	\dots	$x_L^{(i,1)}$
$L^{(i)}$	$g^{(i)}$	$f^{(i)}$	$\phi^{(i)}$	$y_1^{(i)}, \dots, y_n^{(i)}$	$x_1^{(i,2)}$	$x_2^{(i,2)}$	$x_3^{(i,2)}$	\dots	$x_L^{(i,2)}$
\dots									
$L^{(N)}$	$g^{(N)}$	$f^{(N)}$	$\phi^{(N)}$	$y_1^{(N)}, \dots, y_n^{(N)}$	$x_1^{(N,1)}$	$x_2^{(N,1)}$	$x_3^{(N,1)}$	\dots	$x_L^{(N,1)}$
$L^{(N)}$	$g^{(N)}$	$f^{(N)}$	$\phi^{(N)}$	$y_1^{(N)}, \dots, y_n^{(N)}$	$x_1^{(N,2)}$	$x_2^{(N,2)}$	$x_3^{(N,2)}$	\dots	$x_L^{(N,2)}$

Table 1: Format of data file. $L^{(i)}$ is the label for individual i , $g^{(i)}$ is the geographic origin of individual i (PopData); $f^{(i)}$ is a flag used to incorporate learning samples (PopFlag); $\phi^{(i)}$ can store a phenotype for individual i ; $y_1^{(i)}, \dots, y_n^{(i)}$ are for storing extra data (ignored by the program); $(x_l^{i,1}, x_l^{i,2})$ stores the genotype of individual i at locus l . The data for each individual are stored in two consecutive lines in the file. All of the columns preceding the genotype data $(x_l^{(i,j)})$ are optional, but any that are present must be in the order shown. See text for further details.

3. **PopFlag** (Optional) A Boolean flag which indicates whether to use the PopData when using learning learning samples (see USEPOPINFO, below). (*Note: A Boolean variable (flag) is a variable which takes the values TRUE or FALSE, which are designated here by the integers 1 (use PopData) and 0 (don't use PopData), respectively.*)
4. **Phenotype** (Optional) An integer designating the value of a phenotype of interest, for each individual. ($\phi^{(i)}$ in table.) (The phenotype information is not actually used in this program. It is here to permit a smooth interface with the program STRAT which is used for association mapping.)
5. **Extra Columns** (Optional) It may be convenient for the user to include additional data in the input file which are ignored by the pro-

gram. These go here, and may be strings of integers or characters.

6. **Genotype Data** Each allele at a given locus should be coded by a unique integer (eg microsatellite repeat score).

The data for each individual are stored in two consecutive rows in the input file. All rows must have the same number of data entries (the pre-genotype data columns are simply recorded twice for each individual). At each locus, one allele is entered on the first line, and one on the second line. The order of the two alleles does not matter.

Missing genotype data should be indicated by a number which doesn't occur elsewhere in the data (I use -9). This number can also be used to include haploid loci (eg X chromosome in males). The missing-data value is set along with the other parameters (See "MISSING", Section 4).

Except for the "Label" and the "Extra Columns", which can contain any characters, the data file should only contain integers. It is critical that the data for each individual contain the correct number of entries. Extra whitespace (eg spaces, tabs or returns in the middle of lines) are ok. Other punctuation (eg commas) are not permitted.

I have implemented reasonably careful error checking to make sure that the data set is in the correct format, and the program will terminate for most problems. However, there are some cryptic problems that only the user could catch, so I recommend occasionally turning on the "ECHODATA" option, which prints out some of the data (see Section 4).

Table 2 shows a sample data file.

4 Parameters set by User

There are a number of program parameters which are set by the user. These are in two files (*mainparams* and *extraparams*), which are read every time the program executes. You will need to set all the values in *mainparams*, while the default values in *extraparams* are probably ok to begin with. Note that the default model assumes admixture, and does not make use of the user-defined PopData. The parameters in *extraparams* are described in an appendix.

Each parameter is printed in all-caps in one of these two files, preceded by the word "#define". (They are also printed in all-caps throughout this document.) The value is set immediately following the name of the parameter (eg "#define NUMREPS 1000" sets the number of MCMC repetitions to 1000).

George	1	-9	145	66	0	92
George	1	-9	-9	64	0	94
Paula	1	106	142	68	1	92
Paula	1	106	148	64	0	94
Matthew	2	110	145	-9	0	92
Matthew	2	110	148	66	1	-9
Bob	2	108	142	64	1	94
Bob	2	-9	142	-9	0	94
Anja	1	112	142	-9	1	-9
Anja	1	114	142	66	1	94
Peter	1	-9	145	66	0	-9
Peter	1	110	145	-9	1	-9
Carsten	2	108	145	62	0	-9
Carsten	2	110	145	64	1	92

Table 2: Sample data file. Here LABEL=1, POPDATA=1, NUMINDS=7, NUMLOCI=5, and MISSING=-9. Also, POPFLAG=0, PHENOTYPE=0, EXTRACOLS=0. The second column shows the geographic sampling location of individuals.

Following each parameter definition, there is a brief comment (marked “//”), describing the parameter. This includes an indication of what sort of value is expected. These include: “(str)”, for string (used for the names of the input and output files); “(int)”, for integer; “(d)”, for double (i.e., a real number such as 3.14); and “(B)”, for Boolean (i.e., the parameter takes values TRUE or FALSE by setting this to 1 or 0, respectively).

The program is insensitive to the order of the parameters, so you can re-arrange them or add comments, etc. The values of all parameters used for a given run are printed at the end of the output file.

4.1 Parameters in file *mainparams*.

The user will need to set all of these parameters before running the program. Several of these parameters (LABEL, POPDATA, POPFLAG, PHENOTYPE, EXTRACOLS) indicate whether particular types of data are present in the input file; these are described in Section 3.

INFILE (string) Name of input data file. Max length 30 characters (or possibly less depending on operating system).

OUTFILE (string) Name for program output files (the suffixes “_1”, “_2”, ..., “_m” (for intermediate results) and “_f” (final results) are added to this name). Existing files with these names will be overwritten. Max length of name 30 characters (or possibly less depending on operating system).

NUMINDS (int) Number of diploid individuals in data file.

NUMLOCI (int) Number of loci in data file.

LABEL (Boolean) Input file contains labels (names) for each individual. 1 = Yes; 0 = No.

POPDATA (Boolean) Input file contains a user-defined population-of-origin for each individual. 1 = Yes; 0 = No.

POPFLAG (Boolean) Input file contains an indicator variable which says whether to use popinfo when USEPOPINFO==1 (see below). 1 = Yes; 0 = No.

PHENOTYPE (Boolean) Input file contains a column of phenotype information. 1 = Yes; 0 = No.

EXTRACOLS (int) Number of additional columns of data after the Phenotype before the genotype data start. These are ignored by the program. 0 = no extra columns.

MISSING (int) Value given to missing genotype data. Must be an integer, and must not appear elsewhere in the data set. I use -9.

MAXPOPS (int) Number of populations assumed for a particular run of the program. Pritchard *et al.* (2000) call this K . Sometimes (depending on the nature of the data) there is a natural value of K that can be used, otherwise K can be estimated by checking the fit of the model at different values of K (see Section 7).

BURNIN (int) Length of burnin period before the start of data collection. (See Section 6.)

NUMREPS (int) Number of MCMC reps after burnin. (See Section 6.)

5 Interpreting the output

5.1 Output to screen during run

Rep#:	Alpha	Corr	D1,2	Ln Like	Est Ln P(D)
4500:	0.009	6.1	0.831	-2730	-2776
4525:	0.010	9.1	0.806	-2716	-2776
4550:	0.008	7.3	0.792	-2734	-2775

The example above shows part of the output to the screen during a run. Here, “rep” gives the number of MCMC iterations so far, including the burnin; “Alpha” is the current value of α ; “Corr” is the current mean (across loci) value of the correlation coefficient $f^{(l)}$ (see FREQSCORR); “D1,2” is a measure of divergence between populations 1 and 2 (see PRINTKLD); “Ln Like” is the log likelihood of the data given the current values of P and Q ; “Est Ln P(D)” is the current estimate of $\ln(P(X|K))$ (averaging over all iterations since the end of the burnin period). Some of these columns may not be printed, depending on the program options chosen. If K gets large, it may be worth setting PRINTKLD=0 in order to make the output easier to read.

5.2 Printout of Q

Label	(%Miss)	Pop:	Inferred clusters	(and 90% probability intervals)
1 17	(0)	2 :	0.977 0.023	(0.829,1.000) (0.000,0.171)
2 1219	(7)	2 :	0.997 0.003	(0.988,1.000) (0.000,0.012)
3 1223	(0)	2 :	0.833 0.167	(0.003,1.000) (0.000,0.997)
4 1329	(7)	1 :	0.005 0.995	(0.000,0.020) (0.980,1.000)
5 15	(0)	3 :	0.006 0.994	(0.000,0.016) (0.984,1.000)

When the program is run without using prior population information, the results for Q are presented in the format shown above (here K was set to 2). This is read as follows. Reading from row 1: Individual label (taken from data file) = 17; percentage of missing data for this individual = 0%; user-assigned population = 2; estimated membership in clusters 1 and 2 = 0.977 and 0.023 respectively (these are the mean values of $q^{(17)}$); 90% probability intervals on $q_1^{(17)}$ and $q_2^{(17)}$ are (0.829,1.000) and (0.000,0.171), respectively. (Note that the probability intervals are shown only if ANCESTDIST is turned on.)

5.3 Printout of Q when using prior population information

When the program is run with prior population information, the printout is a bit different. Here I have run the same data with $K = 3$, USEPOPINFO=1, and GENSBACK = 1:

```

Label (%Miss) Pop :
1 17      (0) 2 : 0.998 | Pop 1: 0.000 0.001 | Pop 3: 0.000 0.000
2 1219    (7) 2 : 1.000 | Pop 1: 0.000 0.000 | Pop 3: 0.000 0.000
3 1223    (0) 2 : 0.612 | Pop 1: 0.000 0.000 | Pop 3: 0.004 0.383
4 1329    (7) 1 : 1.000 | Pop 2: 0.000 0.000 | Pop 3: 0.000 0.000
5 15      (0) 3 : 0.999 | Pop 1: 0.000 0.001 | Pop 2: 0.000 0.000

```

In general, the first column of results (following “Pop : ”) shows the posterior probability that the individual in question is correctly assigned to the given population. The subsequent columns show the probabilities that it is from, or has ancestry in, the other populations. There are GENSBACK+1 entries for each of the other populations, showing the probability that an individual is from that population, has a parent, grandparent, great-grandparent,... etc, from that population (in this order).

For example, reading from row 3: Individual 1223 (who has 0% missing data) is actually from the presumed population (2) with probability 0.612. There is (approximately) zero posterior probability that this individual has recent ancestry in population 1, but it may have recent ancestry in population 3 (the probabilities are 0.004, and 0.383, that the individual is *from* population 3, or has a single *parent* from population 3, respectively).

5.4 Printout of estimated allele frequencies (P)

```

Locus 5
  3 alleles
 19.0% missing data
 2 0.511 0.821 0.656
 3 0.444 0.171 0.317
 1 0.045 0.008 0.027

```

This example shows a printout for the estimated allele frequencies (P) at locus 5. Columns 2,3, and 4 show the estimated frequency (in clusters 1,2 and 3, respectively) of the allele listed in column 1.

6 How long to run the program

The program is started from a random configuration, and from there takes a series of steps through the parameter space, each of which depends (only) on the parameter values at the previous step. This procedure induces correlations between the state of the Markov chain at different points during the run. The hope is that by running the simulation for long enough, the correlations will be negligible.

There are two issues to worry about: (1) burnin length: how long to run the simulation before collecting data to minimize the effect of the starting configuration, and (2) how long to run the simulation after the burnin to get accurate parameter estimates.

To choose an appropriate burnin length, it is really helpful to look at the values of summary statistics that are printed out by the program (eg α , the divergence distances among populations $D_{i,j}$, and the likelihood) to see whether they appear to have converged. I have found that in examples I have looked at, a burnin of 10,000–100,000 is usually more than adequate.

To choose an appropriate run length, you will need to do several runs at each K , possibly of different lengths, and see whether you get consistent answers. Typically, you can get good estimates of the parameter values (P and Q) with fairly short runs (eg 10,000–100,000), but accurate estimation of $\Pr(X|K)$ requires quite long runs (perhaps 10^6 or more). In practice your run length may be determined by your computer speed and patience as much as anything else.

If the estimate of α varies greatly throughout the run (i.e., not just during the burnin), you may get more accurate estimates of $\Pr(X|K)$ by increasing ALPHAPROPSD, which improves mixing. (See a related issue in section 7).

If your answers appear to be multimodal it is possible that the MCMC algorithm is converging to different solutions (see the example of Data Set 2A in the paper), and longer runs will probably not fix this. You can examine the results for Q to get an idea of whether this seems to be happening.

7 Estimation of K (the number of populations)

In our paper describing this program, we pointed out that this issue should be treated with care for two reasons: (1) it is computationally difficult to obtain accurate estimates of $\Pr(X|K)$, and our method merely provides an *ad hoc* approximation, and (2) the biological interpretation of K may not be

straightforward.

While these caveats should be taken seriously, I should point out that in our experience the method usually gives sensible results, provided that appropriate care is taken.

7.1 Steps in estimating K

1. Set COMPUTEPROBS and INFERALPHA to 1 in the file *extra-params*. Normally you will want FREQSCORR=0.
2. Run the MCMC scheme for different values of MAXPOPS (K). At the end it will output a line "Estimated Ln Prob of Data". This is the estimate of $\ln \Pr(X|K)$. You should run several independent runs for each K , in order to verify that the estimates are consistent across runs. If the variability across runs for a given K is substantial compared to the variability of estimates obtained for different K , you may need to use longer runs or a longer burnin period. If $\ln \Pr(X|K)$ appears to be bimodal, the MCMC scheme may be finding different solutions. You can check for this by comparing the Q for different runs at a single K . (cf Data Set 2A in the paper)
3. Compute posterior probabilities of K . For example, for Data Set 2A in the paper (where K was 2), we got

K	$\ln \Pr(X K)$
1	-4356
2	-3983
3	-3982
4	-3983
5	-4006

We can start by assuming a uniform prior on $K = \{1, \dots, 5\}$. Then from Bayes' Rule, $\Pr(K = 2)$ is given by

$$\frac{e^{-3983}}{e^{-4356} + e^{-3983} + e^{-3982} + e^{-3983} + e^{-4006}} \quad (1)$$

It's easier to compute this if we simplify the expression to

$$\frac{e^{-1}}{e^{-374} + e^{-1} + e^0 + e^{-1} + e^{-24}} = 0.21 \quad (2)$$

There are a couple of informal pointers which might be helpful in selecting K . The first is that it's often the situation that $\Pr(K)$ is very small for K less than the appropriate value (effectively zero), and then more-or-less plateaus for larger K , as in the example of Data Set 2A shown above. In this sort of situation where several values of K give similar estimates of $\Pr(X|K)$, it seems that the smallest of these is usually "correct".

The second pointer is that if there really are separate populations, there is typically a lot of information about the value of α , and once the Markov chain converges, α will normally settle down to be relatively constant (usually with a range of perhaps 0.2 or less in examples I have looked at). However, if there isn't any real structure, α will usually vary greatly during the course of the run.

Suppose that you have a situation with two clear populations, but you are trying to decide whether one of these is further subdivided (ie, the value of $\Pr(X|K = 3)$ is similar to, or perhaps a little larger than $P(X|K = 2)$). Then one thing you could try is to run *structure* using only the individuals in the population that you suspect *might* be subdivided, and see whether there is a strong signal about α .

Another issue is that departures from the model may lead to getting the wrong K . In particular, the presence of null alleles causes departures from Hardy-Weinberg equilibrium, which can be mis-interpreted as being the result of population structure.

In summary, I tend to be sceptical about splitting populations in cases where the divided population has fairly weak statistical support, and no clear biological interpretation.

8 Missing data, null alleles... and other such problems

The program ignores missing genotype data when updating Q and P . This approach is correct when the probability of having missing data at a particular locus is independent of what allele the individual has there. While estimates of Q for individuals with missing data are less accurate, there is no particular reason to exclude such individuals from the analysis, unless they have very little data at all.

A serious problem however arises when data are missing in a systematic manner, as with null alleles. These do not fit the assumed model, and can lead to apparent departures from Hardy-Weinberg even without population structure. One would not expect the assumed model to be robust to this

sort of violation.

Having multiple family members in the sample also violates the model assumptions. Based on quite limited experience, my impression is that this can sometimes lead to overestimation of K , but it may have little effect on the assignment of individuals to populations for fixed K .

Pritchard *et al.* (2000) discuss the issue of loci that are weakly linked (eg 5 cM apart). Briefly, this is not a serious issue when the populations are discrete, because one would not expect linkage disequilibrium between such markers within subpopulations. However, there may be correlations between these markers in an admixed population, and it should be possible to improve the inference by modelling this effect specifically.

8.1 Dominant loci

For some types of genetic markers, it is not possible to distinguish all the genotypes (eg, genotypes AA and Aa look the same on the gel, while aa is different). I have not implemented anything specific for this type of data, however the program *can* be applied to such data under the “no admixture” model (set NOADMIX=1).

This can be done by treating each class of genotypes as being, effectively, a haploid allele. Then we might designate an AA/Aa individual in the input file as (1,-9), and an aa individual as (2,-9), where -9 is the value used for missing data. This approach can also be used for more than two genotypic classes.

This fix is valid under the no-admixture model, because the updates for P and Q are still correct even though we are treating genotypic classes rather than alleles. Of course, there is less information than if all the genotypes could be distinguished. The fix is incorrect under the admixture model.

9 Missing-data bug in original release

The earliest release version (before May 2000) did not perform properly when “0” was used to denote missing data. This has now been fixed.

References

- Beaumont, M., Gottelli, D., Barratt, E. M., Kitchener, A. C., Daniels, M., Pritchard, J. K. and Bruford, M. (2000) Genetic diversity and introgression in the scottish wildcat. Submitted to Molecular Ecology.

Pritchard, J. K., Stephens, M. and Donnelly, P. (2000) Inference of population structure using multilocus genotype data. *Genetics* (accepted). Available at <http://www.stats.ox.ac.uk/~pritch/papers/strucabs.html>.

10 Appendix: Parameters in file *extraparams*.

These options allow the user to refine the model in various ways, and do more involved analyses. The default values are probably fine to begin with. For Boolean options, type 1 for “Yes”, or “Use this option”; 0 for “No” or “Don’t use this option”.

10.1 Program Options

INFERALPHA (Boolean) Infer the value of the model parameter α from the data; otherwise α is fixed at the value ALPHA which is chosen by the user. This option is ignored under the NOADMIX model. (The prior for the ancestry vector Q is Dirichlet with parameters $(\alpha, \alpha, \dots, \alpha)$. Small α implies that most individuals are essentially from one population or another, while $\alpha > 1$ implies that most individuals are admixed.)

FREQSCORR (double) Use a model where the allele frequencies are correlated. More specifically, rather than assuming a prior in which the allele frequencies in each population are independent draws from a uniform Dirichlet distribution, we start with a distribution which is centered around the mean allele frequencies in the sample. This model is more realistic for very closely related populations (where we expect the allele frequencies to be similar across populations), and can produce better clustering. Specifically, suppose that locus l has j alleles. Then we model the allele frequencies at this locus in population k as being $\mathcal{D}(jf^{(l)}\mu_1^{(l)}, jf^{(l)}\mu_2^{(l)}, \dots, jf^{(l)}\mu_j^{(l)})$, where $\mu_i^{(l)}$ is the sample frequency of allele i . The parameter $f^{(l)}$ determines the amount of variation around the mean frequencies (large values lead to less variation). The prior of $f^{(l)}$ is set using CORRA, and CORRB. There may be a tendency to overestimate K when FREQSCORR is turned on.

COMPUTEPROB (Boolean) Print the log-likelihood of the data at each update, and estimate the probability of the data given K and the model (see section 7). This is used in estimating K , and is also a useful diagnostic for whether the burnin is long enough. The main reason for turning this off would be to speed up the program (~ 10 – 15%).

NOADMIX (Boolean) Assume the model without admixture (Pritchard *et al.*, 2000). (Each individual is assumed to be completely from one

of the K populations.) In the output, instead of printing the average value of Q as in the admixture case, the program prints the posterior probability that each individual is from each population. 1 = no admixture; 0 = model with admixture.

NOADMBURN (int) (For use when NOADMIX=1.) The first NOADMBURN iterations of the MCMC algorithm use the admixture model; after this, if NOADMIX=1, it switches to the no-admixture model. I have found that this can produce better convergence. Set NOADMBURN < BURNIN.

10.2 Using prior population information

The default mode for *structure* uses only genetic information to learn about population structure. However, there is often other information that can be used (e.g., physical characteristics of sampled individuals or geographic sampling location) to assist the clustering. In the section “Model with prior population information” in Pritchard *et al.* (2000), we described our framework for incorporating this type of information into the inference procedure.

There are at least two kinds of reasons for making use of this sort of extrinsic population information. One is that one may want to test whether any individuals in the sample are immigrants to their supposed populations, or have recent immigrant ancestors (see the example in Pritchard *et al.* (2000)).

A second is that we may want to make use of learning samples: ie., we have some individuals of known origin, and we want to use them to help us classify individuals of unknown origin. For example in Beaumont *et al.* (2000), we wanted to learn about the ancestry of Scottish wildcats (many of which are hybridized with feral domestic cats). We had genetic data from a bunch of pet house cats which we defined as being in one population, while we inferred Q for the wildcats (with $K = 2$). Use of this sort of prior information will normally improve the accuracy of the inference.

To use these options you need to set USEPOPINFO to 1, and choose a value of MIGRPRIOR (which is ν in Pritchard *et al.*). You might choose something in the range 0.001 to 0.1 for ν . Even when using learning samples, it may be sensible to allow for some misclassification by setting MIGRPRIOR larger than 0.

The pre-defined population for each individual is set in the input data file (see PopData) and should be an integer between 1 and MAXPOPS, inclusive. If PopData for any individual is outside this range, their q will

be updated in the normal way (ie without prior population information, according to the model that would be used if USEPOPINFO was turned off.²). Learning samples are implemented through the use of the PopFlag column in the data file. The pre-defined population is used for those individuals for whom PopFlag=1, and it is ignored for individuals for whom PopFlag=0. If there is no PopFlag column in the data file, then when USEPOPINFO is turned on, PopFlag is set to 1 for all individuals.

In general, we advocate that the user should first run the program without population information to ensure that the pre-defined populations are in rough agreement with the genetic information.

USEPOPINFO (Boolean) Use prior population information to assist clustering. See also MIGRPRIOR and GENSBACK. Must have POPDATA=1.

GENSBACK (int) This corresponds to G (Pritchard *et al.*, 2000). When using prior population information for individuals (USEPOPINFO=1), the program tests whether each individual has an immigrant ancestor in the last G generations, where $G = 0$ corresponds to the individual being an immigrant itself. In order to have decent power, G should be set fairly small (2, say) unless the data are highly informative.

MIGRPRIOR (double) Must be in $[0,1]$. This is ν in Pritchard *et al.* (2000). Sensible values might be in the range 0.001—0.1.

10.3 Output options

PRINTQHAT (Boolean) When this is turned on, the point estimate for Q is not only printed into the main results file, but also into a separate file with suffix “q”. This file is required in order to run the companion program STRAT.

UPDATEFREQ (int) Number of MCMC iterations between printing each update on the screen. Value of 0 will cause this to be set automatically.

PRINTLIKES (Boolean) Print the current value of the likelihood to the screen in every iteration.

²If the admixture model is used to estimate q for those individuals without prior population information, α is updated on the basis of those individuals only. If there are very few such individuals, you may need to fix α at a sensible value.

INTERMEDSAVE (int) If you're impatient to see preliminary results before the end of the run, you can have results printed to file at intervals during the MCMC run. A total of INTERMEDSAVE such files are printed, at equal intervals following the completion of the BURNIN. Turn this off by setting to 0. Names of these files created using OUTFILE name.

PRINTKLD (Boolean) Print the "Kullback-Leibler" divergence "D" between populations. The output to the screen gets a bit wide for moderate K , so the user may want to switch this off to clarify the output. The distance between populations A and B is computed as

$$D_{A,B} = \sum_{l=1}^L \sum_{j=1}^{J_l} \hat{p}_{A,j}^{(l)} \ln\left(\frac{\hat{p}_{A,j}^{(l)}}{\hat{p}_{B,j}^{(l)}}\right), \quad (3)$$

where $\hat{p}_{A,j}^{(l)}$ is the estimated allele frequency of allele j at locus l in population A , and where L is the number of loci, and J_l the number of alleles at locus l . Notice that $D_{A,B}$ and $D_{B,A}$ are not, in general the same; however to save space in printing the running update, we print the average of these. This measure of divergence can be motivated as follows. Suppose that we want to infer whether an individual that is actually from population A is from A or B . Then it is natural to compute the ratio of the likelihood of their genotype in each population. $D_{A,B}$ gives the average contribution of a single genotyped allele to the log-likelihood ratio. Suppose for example that $D_{A,B} = 0.1$. Then by genotyping 10 (diploid) loci, the expected log-likelihood ratio is 2.0. $D_{A,B}$ provides a natural measure for the strength of evidence for assignments of individuals.

ECHODATA (Boolean) Print a brief summary of the data set to the screen and output file. (Prints the beginnings and ends of the top and bottom lines of the input file to allow the user to check that it has been read correctly.)

ANCESTDIST (Boolean) Collect information about the distribution of Q for each individual, as well as just estimating the mean. When this is turned on, the output file includes the left- and right-hand ends of the probability intervals for each $q(i)$. (A probability interval is the Bayesian analog of a confidence interval.) The values printed show the middle $100p\%$ of the probability interval, where p is a number in the

range 0.0 to 1.0 and is set using **ANCESTPINT**. The distribution of Q is estimated by recording the number of hits in each of a number of boxes between 0 and 1, to form a sort of histogram. The width of these boxes, which are of equal size, is set using **NUMBOXES**.

10.4 Priors

These values are used to parametrize the assumed probability models. In most cases they have little effect on the results, and the user may not want to worry about them. The values of **CORRA** and **CORRB** matter somewhat when **FREQSCORR**=1. The value of **ALPHA** has quite an effect if **INFERALPHA**=0.

ALPHA (double) Dirichlet parameter (α) for degree of admixture (this is the initial value if **INFERALPHA**==1).

CORRA, **CORRB** (double) See **FREQSCORR**. The prior for $f^{(l)}$ is taken to be Gamma with mean **CORRA*****CORRB**, and variance **CORRA*****CORRB**².

EPSILON (double) Dirichlet parameter for allele frequencies when **FREQSCORR**=0. A value of 1.0 (which produces a uniform distribution) seems to work fine.

UNIFPRIORALPHA (Boolean), **ALPHAMAX** (double) Assume a uniform prior for α which runs between 0 and **ALPHAMAX**. This model seems to work fine; the alternative model (when **UNIFPRIORALPHA**=0) is to take α as having a Gamma prior, with mean **ALPHAPRIORA*****ALPHAPRIORB**, and variance **ALPHAPRIORA*****ALPHAPRIORB**².

ALPHAPRIORA, **ALPHAPRIORB** (double) See **UNIFPRIORALPHA**.

10.5 Further miscellaneous options

ALPHAPROPSD (double) The Metropolis-Hastings update step for α involves picking a value α' from a Normal with mean α and standard deviation **ALPHAPROPSD**> 0. The value of **ALPHAPROPSD** does not affect the asymptotic behaviour of the Markov chain, but may have a substantial impact on the rate of convergence. If there is a lot of information about α , small values of **ALPHAPROPSD** are preferable to obtain a reasonable acceptance rate. If there's not much information about α , larger values produce better mixing.

FPROPSD (double) The standard deviation of the proposal for the allele frequency correlation $f^{(l)}$. Analogous to ALPHAPROPSD. Only relevant when $\text{FREQSCORR} = 1$.

STARTATPOPINFO (Boolean) Use given populations as the initial condition for population origins. (Need $\text{POPDATA}==1$). This option is here to provide another check that the Markov chain is converging adequately. This option only affects the starting point of the Markov chain and, in theory, it should not affect the final answers, provided that the BURNIN is long enough.

It is assumed that the PopData in the input file are between 1 and k where $k \leq \text{MAXPOPS}$. Individuals for whom the PopData are not in this range are initialized at random.

RANDOMIZE (Boolean) Use a different random number seed for each run (this is taken from the system clock).

METROFREQ (int) Frequency of using a Metropolis-Hastings step to update Q under the admixture model. When this is used, a new proposal $q^{(i)'} is chosen for each $q^{(i)}$. This proposal is sampled from the prior (ie $q^{(i)'} \sim \mathcal{D}(\alpha, \alpha, \dots, \alpha)$). The rationale for having this update is that it may improve mixing when alpha is quite small, by making it easier for individuals to jump between populations. The Metropolis-Hastings move is used once every METROFREQ iterations. If METROFREQ is set to 0, it is never used.$

REPORTHITRATE (Boolean) Report acceptance rate of Metropolis update for $q^{(i)}$ (see METROFREQ).