Documentation for UNIX version of DMLE+

Release 2.2, 14 January, 2005.

Executing the program: The program is executed by entering the following command in a shell

```
<path>/DMLE+2.2 <filename>
```

where <path> is the directory containing the program binary DMLE+2.2 and `<filename>` is an input file located in the same directory.

Files Created by Program:

The program's output files are named `<filename.ext>` (where ".ext" will be .dat, .sig, .hap, .log, .tre, .mat, or .hpf; see description of output below). If a contig file is used for sequence information, an additional file (exons.txt) is created, which shows the exon/intron/non-genic boundaries, in Morgans, translated from the contig file.

Input file format:

In general, the input file consists of alternating lines of descriptors then data. Avoid having any blank lines in the input file. Data may over-run a line as long as no new line character is used (i.e. don't use the "Enter" key to make an extra new line). Avoid blank lines in the input file. All fields with multiple entries may be delimited with space(s) or tabs. In the following description, each line from the input file (a combination of input descriptors and data) is given, immediately followed by a description of the meaning of input surrounded by square brackets [].

Step-by-step Guide to Input File Entries:

```
Data as genotypes?:(0=haplotypes, 1=genotypes):
0
```

[If the patient data are completely haplotyped, use 0.]

```
Genetic model: (dominant=0,recessive=1)
0
```

[Only used for genotype data. If haplotypes are used, this field is ignored, and it is assumed that all chromosomes included below in "Patient haplo- or genotypes" are disease bearing, whatever the genetic basis for the disease.]

```
Read old file?: (0=no, 1=yes):
0
```

[If the tree is to start from a saved tree from the end of a previous run, set this to 1, and change the name of the output matrix file (see below) "*.mat" to "inmat.txt".]

```
Use fixed random seed?:(0 = no (=random), negative integer
= yes (=fixed)):
0
```

[The random number generator either starts with a random negative integer (if set to 0), or uses the negative integer supplied here if you want to be able to reproduce the same sequence of pseudo-random numbers.]

```
# chromosomes (N):
148
```

[Number of chromosomes in the disease sample. If the data are genotypes, this must be an even number.]

```
# loci per chromosome (L):
5
```

[Number of marker loci.]

```
Numbers of haplotypes in the normal (base) population:
45 1 1 1 1 1
35 1 1 1 1 6
15 1 1 2 1 2
16 1 1 2 2 2
10 1 2 2 3 3
5 2 2 2 3 4
2 2 -1 3 4 5
1 -1 3 -1 4 2
```

[Each row starts with a frequency (count) of the haplotype, followed by that haplotype's alleles. Unknown markers types are indicated with a -1. For any particular marker, when the control sample and patient sample are considered together (pooled), the allele codes for that marker should start at 1 and make use of consecutive positive integers, with no gaps. Gaps in the numbering system (e.g. 1,2,3,7, and 8 as the collection of alleles at a marker in the controls + disease-bearing chromosomes) are not allowed and will produce an error message. The program uses these normal population counts only to check against normal allele counts - genotypes are exactly equivalent to haplotypes. Therefore, if you have genotype data, split each genotype into any of the possible haplotypes. The frequency column must be included even if it is always "1". Haplotypes do not have to be grouped, e.g. you can have
23 1 1 2 1 1
14 1 1 2 1 1
(Note that earlier versions of the program ended the list of haplotypes with a row reading "-99".)]

```
Map distances:
0.0 0.00013 0.000165 0.000195 0.000205
```

[The first map distance must be set to 0.0, with the rest given in Morgans (e.g. given that 1 cM =~ 1000 kb in humans, the final entry above is ~20.5kb). Ensure that there are as many entries here as there are markers indicated above. These entries are overridden if a contig file is being read for sequence information (see below).]

```
Run simulation?:
0
```

[If analyzing actual marker data, this should be zero. Set this to be 1 if you wish to simulate chromosomes for an analysis (under a population coalescent process) using the allele frequencies specified in the population of normal chromosomes and the other parameters as specified in the input file. If a simulation is run, the actual location of the mutation (for the simulated data) is read from the variable below. Regardless of whether 0 or 1 is chosen here, the initial tree topology, and the ancestral and internal states, are simulated in an identical fashion. Only the tips (extant chromosomes) are not simulated when this parameter is set to zero.]

```
Mutation location:
0.0003
```

[Position of the mutation relative to the first marker (used for simulations). This number can be negative. The value of this number is ignored if no simulation is run, but some value must still be present.]

```
Mutation's low and high boundaries
-0.05 0.05
```

[Limits to the range of mutation positions (relative to marker 1, in Morgans) that will be considered in the Markov chain]

```
# simultaneous runs:
1
```

[Number of chains run simultaneously in the MCMC analysis. If more than one chain is run simultaneously, the program will generate a "square root of R" statistic that can be used to assess whether the multiple runs have converged (currently implemented only for convergence amongst iterated mutation locations, not for age of mutation).]

```
Starting value(s) for mutation location for each
simultaneous run (-99 for random):
-99
```

[The value for the mutation location used in iteration 1 of the Markov chain (not the true location of the mutation). If -99, a random uniform number between the mutation’s low

and high boundaries (see above) will be generated. If not –99, there must be one number for each simultaneous run, and these must be between the low and high boundaries]

```
Population growth rate:
0.085
```

[Population growth rate, estimated from historical data]

```
Proportion of population sampled:
0.001825
```

[Proportion of disease chromosomes in sample, estimated from disease incidence and number of disease chromosomes sampled]

```
Iterate root state, mutation age, mutation location, allele
freq. (0=no, 1=yes*):
1 0 1 0
```

[Choose 0 to hold the parameter fixed, 1 to iterate over the parameter. Internal nodes are always iterated. If the mutation location is known and you want to estimate the mutation's age, use 1 1 0 0 (or 1 1 0 1). If the allele frequencies are iterated, the frequencies are initially set to be equal for all alleles present in the normal and disease sample at each marker. If they are not iterated, frequencies from the normal population are used.

* In version 2.2, there are 3 methods of nominating new mutation locations. The default value "1" (omit the quotes!) uses the standard method. "2" uses slice sampling, which can be tried if the likelihood surface is multimodal and the peaks are relatively far apart. It is slower than the standard method. "3" samples uniformly across the range of the mutation boundaries, and is NOT recommended unless you are having extreme difficult getting multiple chains to converge to the same locations.]

```
Flip (potentially) all loci? (0=no, 1=yes):
1
```

[When the alleles of the internal nodes are changed in the MCMC, a single random allele from each node can be (potentially) altered (= 0), or the alleles at all markers can be (potentially) altered (=1).]

```
Adjustment level for tree, rootage, mutation_location,
ancestral states, internal states, allele frequencies, and
method:
1.0 3.0 0.005 0.15 0.15 0.1 1
```

[The size of the adjustment factor that determines the range of new values that are nominated at each iteration of the MCMC, for the six variables (tree topology; mutation age, mutation location; ancestral haplotype alleles; internal node haplotype alleles; and allele frequencies), if they are set to be iterated- see above. Mutation age, if iterated, is

adjusted by the same factor as are the other nodes of the tree. The method variable determines how transition probabilities are calculated. 0 uses marginal allele frequencies, 1 uses marginal haplotype frequencies. It is recommended that the default value of 1.0 be used, as simulations over a wide range of input datasets indicate that it will produce more accurate results roughly 75% of the time. If the control population markers are thought to be in near-perfect linkage equilibrium, allele frequencies can be used, and the program will run faster]

```
Burn-in iterations:
1000
```

[Number of iterations to perform in the MCMC analysis before the data for the posterior distribution histograms starts to be tabulated.]

```
Iterations:
1000
```

[Number of iterations over which data is tabulated in the MCMC analysis to construct the posterior distribution histograms.]

```
Screen and file update intervals, "acceptance details",
save population haplotype frequencies:
50 100 1 1
```

[Data is sent to the screen and the output files at intervals corresponding to the first two entries respectively (data from the first and last iterations are also shown). A high rate of screen updates may slow down program execution. Large numbers of file updates produce large data files, and may be difficult to read completely in spreadsheets that can store only limited number of rows, e.g. ~65,000 for Microsoft Excel. The third number indicates whether details about the acceptance rate for changes in the five variables (tree topology, mutation location, root states, internal states, and allele frequencies) should be printed to the output file *.dat (see below). If 0, they are not saved, if 1 they are. In either case, these numbers are printed to the screen. (Note that earlier versions only had 3 entries in this input row. “Save pop. haplotypes” is a new options - See output.]

```
Number of histogram bars:
200
```

[The number of bins into which the mutation locations are sorted]

```
ALPHA level for histograms:
0.05
```

[Significance level for the credible set recorded in the *.sig and *.tre output files (see below)]

```
Mutation age:
100
```

[Estimated age (in generations) of the original mutation in the population. If age is being iterated, this number will be ignored and the initial age will be a random number no more than 100 generations older that the minimum set below.]

```
Mutation age boundaries:
0 1000
```

[Sets boundaries for mutation age if it is being iterated.]

```
Star genealogy (0=no, 1=yes):
0
```

[If set to 1, the original tree will be given a star phylogeny (all branch lengths equal). If this is combined with an adjustment level of 0.0 for the tree topology (see above), the star topology will remain throughout the run.]

```
Loci for the ancestral state (-99 for random):
1 1 2 1 1
```

[The contents of this field are only important if the "iterate ancestral states" variable (see above) is set to 0.]

```
Patient haplo- or genotypes:
79 1 1 1 1 1
31 1 1 1 1 2
18 1 1 1 2 2
5 1 1 1 2 1
5 2 2 2 2 3
4 1 2 1 1 1
2 1 2 1 1 2
1 2 2 2 -1 2
1 2 2 2 2 2
1 -1 2 2 3 2
```

[These entries can be haplotypes or genotypes. Missing alleles (-1) are allowed in both cases.
1) Haplotypes (shown above). The first column is the frequency of the haplotype, followed by the allele at each marker. Columns are delimited by spaces or tabs. If simulated data is being used, you may fill in the above with arbitrary haplotypes or genotypes with the constraint that they must add up to the total number of chromosomes indicated in the # chromosomes field above (in this case the haplotypes you provide are not used, but their frequencies are needed to know when to stop reading this field and move on to the next one). E.g.
148 1 1 1 1 1
would be fine. The total number of rows in this table of haplotypes does not matter. As with the normal population haplotypes, all identical disease haplotypes do not have to be grouped.
2) Genotypes. There is no first column for frequencies. Alleles may be separated with “/”, “\”, or “,”. E.g.

1/2 1/1 1/3 2/4 1/1
3/2 4/3 4/2 5/1 2/1
etc…
Note that for genotype data, if the number of chromosomes is 148, there should only be 74 rows of genotype data, whether dominant or recessive.]

```
Use sequence weights?
1
```

[When this is set to 1, priors will be assigned to the DNA sequence depending on whether the region is an exon, intron, or non-gene (see below). If set to 0, a uniform prior is used]

```
Weights for exons, introns, non-genes
1 0.17 0.02
```

[These weights only used if the above sequence weight parameter is set to 1. The first entry is for exons, the second for introns, and the last for non-genic regions. These are followed immediately by either:]

```
A)
-0.00208767 e
-0.00208449 i
-0.00206302 e
```

etc.

[The first column indicates the starting position, relative to the first marker (0.0), of the initial element in the DNA sequence in the area of interest. The second column indicates what type of element starts at this location (e = exon, i = intron, n = nongenic).

or B)
Input file:
filename
301500
301700
303234
311111
312456
[Where filename is the filename for a Genbank/NCBI contig file, for instance NT_029406, saved in text format (see sample file). This file must be placed in the same directory as DMLE+. Following the filename are the positions of your markers in the contig (which starts at base #1).] (Note that earlier versions of the program had a line that read “99999 Input file:”)

Output:

Standard output (screen):

Each time the standard output is updated (at an interval determined by the screen update parameter – see above) one row with either 9 or 10 columns is displayed for each chain being run:

Column 1: the iteration number.

Column 2: the location of the mutation (relative to the first marker).

Column 3: the log-likelihood for the entire tree (topology, mutation location, internal and ancestral states of marker alleles).

Column 4: the depth of the tree.

Column 5: the proportion of iterations in which the change in tree topology has been accepted.

Column 6: the proportion of iterations in which the change in mutation location has been accepted.

Column 7: the proportion of iterations in which a change in the original mutation's haplotype has been accepted.

Column 8: the proportion of possible internal node haplotype changes in which one (or more) changes in an internal node's haplotype has been accepted (i.e. total number of changed haplotypes/(iterations x internal nodes)).

Column 9: the proportion of changes accepted in allele frequencies.

Column 10: This column is only present if the number of simultaneous runs is set to be greater than 1. It is the value for the "square root of R" statistic. Its value only appears at the end of the first row of any set of results for multiple chains.

When the program is done, it will print "Finished". In a large sample, there may be a small delay between the screen output for the final iteration and the "Finished" signal, due to writing potentially large files to disk.

File output:

Six to eight output files are created on each run of the program. Each has the same filename (+ the extension if any) of the input file, followed by an extension of either .dat, .sig, .hap, .log, .tre, .mat., and .hpf (For example, an input file called test1 will produce test1.dat, test1.sig, test1.hap, test1.log, test1.tre, test1.mat., and test1.hpf) The contents of these files are as described below. The columns of these files are space delimited, and are therefore easily parsed in a spreadsheet. If the output files from a previous run are present in the directory in which the program is run, and the new input file has the same name as

one previously used, the original contents of the old output files will be overwritten at the start of the new run.

*.dat - This file stores information from both the burn-in and "good" iterations. The frequency at which the data is saved to this file is set by the fileupdate parameter. The data for multiple runs are saved in a single row per saved iteration, for easier handling with spreadsheets. If the "acceptance details" parameter (see above) is set to 0, columns 6 through 10 will not be printed to the *.dat file. The columns in the output file are as follows:

1: the iteration number.

2: the mutation location.

3: the log-likelihood for the entire tree (topology, mutation location, internal and ancestral states of marker alleles).

4: the current age of the mutation.

5: the log-likelihood for the tree structure itself, excluding priors.

6: the proportion of iterations in which the change in tree topology has been accepted.

7: the proportion of iterations in which the change in mutation location has been accepted.

8: the proportion of iterations in which a change in the original mutation’s haplotype has been accepted.

9: the proportion of possible internal node haplotype changes in which one (or more) changes in an internal node’s haplotype has been accepted (i.e. total number of changed haplotypes/(iterations x internal nodes)).

10: the proportion of possible allele frequency changes that have been accepted.

These ten columns are repeated for each chain. If more than 1 chain is run, a final column gives the value for the “square root of R” statistic.

*.sig – Histogram data for the posterior distribution of the mutation locations over only the "good" iterations, i.e. with the burn-in results excluded. The data for this histogram are recorded every iteration, and are not affected by the either of the "update" parameters. If the mutation location is not being iterated, this file will be all zeroes. If more than one chain is run, the histogram data for each chain are printed in the same file, separated by a blank line. The file is sorted by column 1. The columns in the output are as follows:

1: the bin midpoints for the mutation locations over the good iterations. The number of bins is from an input parameter (see above). The total range between the lowest and highest mutation locations from these iterations is divided by the number of intervals to determine the bin width.

2: the frequency for each bin.

3: the actual counts for each bin.

4: significance (1=sig, 0 = non-sig) based on the ALPHA value (see above). The sum of the frequencies of the non-significant bins is <= ALPHA.

*.hap - The frequencies (in actual numbers) of all haplotypes present in the sample. If the data is not being simulated, these haplotype frequencies are those from the input file. If the data were simulated, these values will (generally) be different from the input haplotype frequencies (which are ignored). If multiple chains are run simultaneously, only the haplotypes from chain #1 run are used. (With simulated data, the haplotypes would generally be different for all chains, whereas the haplotypes are identical if real data is used.)

*.log - This file contains a list of what the program has stored as the input parameters. This file should be checked to make sure the program is correctly reading in the input. Program crashes are often a result of incorrect input format.

*.tre – Histogram data for the posterior distribution of the time since the original mutation, over only the “good” iterations. Columns are as in the *.sig output file described above. If the depth of the mutation is not being iterated, this file will be all zeroes.

.mat - This file contains a matrix representation of the tree at the end of the "good" iterations. Such trees may then be loaded into the start of a new run. This might be done if it is suspected that the posterior distribution of the mutation locations had not reached stationarity by the start of the good iterations. If an old tree is to be used, ".mat" must be renamed "inmat.txt" and placed in the same directory as the executable. In addition, the "use oldfile?" input parameter (see above) must be set to 1. This matrix is sorted by the first column. If multiple chains are run simultaneously, only the matrix from the first run is saved. The columns in the output are as follows:

1: the node ID number.

2: the node’s left child. If there is no left child (i.e. the node is a leaf [= extant chromosome]), this is coded as –1.0.

3: the node’s right child. Coded as –1.0 as above if no right child.

4: the node’s parent. Coded as –1.0 if there is no parent (i.e. the node is the root node).

5: the node’s coalescence time (in generations). Leaves have a coalescence time of 0.0.

6: the length of time (in generations) between the coalescence time of the node and that of its parent. If the node is the root, this is the time between the root’s coalescence time and the age of the mutation.

7: the transition probability (log-likelihood) associated with that node’s current state, given the state of its parent (or that of the ancestral haplotype in the case of the root node).

7a: In the recessive model, this column pairs chromosomes by individual.

8 to 8 + (number of markers) -1: the alleles at each marker locus. In the dominant model, there are an extra (number of markers) columns, indicating that individual's alleles on the haplotype not currently in the disease tree.

If there were multiple runs of the chain, only the tree from chain #1 run is saved.

*.hpf - This file contains population haplotype numbers, frequencies, and alleles, for the non burn-in iterations of the first chain (results from additional chains, if present, are not stored). The haplotypes for these results are only recorded every “file update” generations (see Input file above).