Documentation for DRIFTER, version 2.2

Murray P. Cox[1]

Arizona Research Laboratories, University of Arizona

December 2005

[1]Address for correspondence: Arizona Research Laboratories – Biotechnology, University of Arizona, Bioscience West – Room 246, 1041 East Lowell Street, Tucson, AZ 85721, USA.

Contents

1.1 Introduction

Drifter is a Perl script designed to simulate genetic drift and sampling error in population genetic samples. This programme is intended primarily to simulate drift in haploid populations of varying size given known (or predicted) starting conditions (*i.e.*, allele frequencies and time of population origin). These conditions are met for some well-studied human populations, but may also hold for various non-human populations, as well as controlled laboratory experiments (*e.g.*, genetic studies with *Drosophila* spp.). Drifter is relatively flexible, and is also suitable for allele frequencies from diploid systems, multiple population samples (either hypothetical or from the real-world), and multiple alleles. Drifter is useful as a teaching aide in practical genetics courses with a focus on genetic drift and/or sampling effects.

1.2 Algorithms

Drifter implements two processes: random genetic drift and sampling error effects. These processes may be run separately or together.

1.2.1 Random Genetic Drift

Genetic drift is usually defined (in a Wrightian sense) as variation in the allele frequencies of population $t+1$ arising from random sampling of gametes from population t (Wright 1931). This definition holds well for diploid genetic systems. Intergenerational variation in haploid allele frequencies is primarily a function of differential fecundity across individuals, but this can also be modelled in Drifter if the underlying processes are assumed to be random. Drifter takes a population of fixed size (N_e), and generates the next generation's cohort by randomly selecting alleles from the previous generation's allele frequency profile. The extent of genetic drift varies with population size (Kimura 1968). For a newly-arisen, neutral mutation:

$$p_{(fixation)} = \frac{1}{2N_e} \tag{1.1}$$

$$p_{(loss)} = 1 - \frac{1}{2N_e} \tag{1.2}$$

For polymorphisms with an allele frequency (p), the probability of fixing the polymorphism is p, and the probability of the allele becoming extinct is $1-p$. All these effects are automatic outcomes of the random sampling of N_e alleles between generations.

1.2.2 Sampling Effects

Sampling effects arise from randomly choosing a small number of individuals to infer allele frequencies for the entire population. This effect can be addressed analytically:

$$var(p) = \frac{p(1-p)}{n} \tag{1.3}$$

…where p is the allele frequency and n is the sample size. Assuming a normal approximation, the extent of a 95% confidence region can be ascertained using:

$$SD(p) = \sqrt{\frac{p(1-p)}{n}} \tag{1.4}$$

$$95\%\text{CI}(p) = \pm 1.96\sqrt{\frac{p(1-p)}{n}} \tag{1.5}$$

Drifter can estimate these parameters empirically during a simulation run. In other words, the data from a drift simulation can be 'sampled' before being output. This may be a simpler approach to generating 95% confidence regions. As $p \to 0$ or $p \to 1$, the 95% confidence region falls asymmetrically about p, and calculating boundaries of the 95% confidence regions from simulated data avoids the need to adjust for this effect analytically.

1.3 Running Drifter

1.3.1 Programme Setup

You must have Perl language support installed to run Drifter. PERL usually comes pre-installed on most UNIX systems, as well as Apple systems running Mac OS X or higher. Windows users should download 'ActivePERL,' which is freely-available from: `http://www.perl.org/`.

It should also be noted that Drifter is not fast when evaluating drift in large populations or over many generations. In such cases, running the script in the background on a UNIX workstation or Macintosh G5 would be preferable. In addition, use of the -a flag can generate large output files (say, 10-100 MB), and this option should be used with care.

1.3.2 Input File

Input files can be given any alphanumeric name without spaces (underscores are acceptable). The input file should start with a line containing only the phrase '#drifter,' and subsequent lines should contain population information. The end of a population subsection is indicated by a semicolon.

The `poplabel` line contain the population name in square brackets (*e.g.*, in the following example, `Malaita` and `Rendova`). `Ne` and `Ns` are the effective population size and the sample size, respectively. Subsequent lines (`A1`, `A2`, . . . , `An`) contain allele names (first square bracket) and frequencies (second square bracket). There is no restraint on the number of alleles, and allele frequencies can vary from 0 to 1. (Alleles with frequency 0 or 1 are best excluded from the input file. They do not affect the output, but will add to the calculation time).

Allele names should contain only alphanumeric characters (*i.e.*, avoid names such as K* or K-M230). Additionally, allele names are used internally as regular expressions, so care should be taken that namespaces do not collide. For instance, 'K' and 'KM' are a poor combination of allele names, because 'K' is a component of 'KM.' Using the alphabet to represent alleles is an easy solution to this problem. Alternately, namespaces with a larger number of alphanumeric characters should rarely collide (*e.g.*, seven, eight, nine; or 012, 120, 210).

The following example input file (`solomons.input`) uses data from Cox and Lahr (2006). Your own data should confirm to this general pattern:

```
#drifter
poplabel = [Malaita]
Ne = [10000]
Ns = [12]
A1 = [K][0.667]
A2 = [S][0.083]
A3 = [M][0.250]
;
poplabel = [Rendova]
Ne = [1000]
Ns = [20]
A1 = [K][0.550]
A2 = [O][0.450]
;
```

1.3.3 Command Line Options

To run Drifter, open a command line terminal. The simplest input command is:

```
perl drifter input.file
```

Advanced UNIX users can set the path to the Perl executable using the #! flagline in the `Drifter` source file, change the file permissions to executable, and invoke the programme directly.

```
drifter input.file
```

The simulation run can be modified with the following flags:

```
-b          Verbose.  Outputs verbose comments.
-h          Horizontal.  Outputs a horizontal format output file.
-v          Vertical.  Outputs a vertical format output file.
-a          All.  Outputs frequency values for each generation (one population only).
-t          Timefile.  Outputs a log file containing run time information.
-s          Sampling.  Creates only a sampling profile for the input populations.
-d NUMBER   Seed.  Allows the user to enter a random number seed.
-g NUMBER   Generations.  The number of generations to drift the input populations.
-i NUMBER   Iterations.  Defines the number of independent replicates.
```

For instance, to run the solomons dataset with sampling only, the input command is:

```
perl drifter -s solomons.input
```

To drift a dataset for 10 generations and 20 iterations, the command would be:

```
perl drifter -g 10 -i 20 solomons.input
```

It is possible to add as many of these commands as you wish:

```
perl drifter -b -h -v -t -d 2793926932 -g 10 -i 20 solomons.input
```

The only exceptions are that 1. `-a` allows only one population sample in the input file (multiple populations can be run independently from multiple input files); and 2. `-s` overrides `-g`, because if you are only sampling from the present generation, the number of generations (g) to drift is by definition 0. If you wish to sample from the simulated data, you should provide a sample size (`Ns`) in the input file. (Alternately, if you do not want to sample from the drifted simulates, `Ns` can be set to `Ne` in the input file). For all options other than `-s`, you will be prompted for the number of generations `-g` and the number of iterations `-i`, if these are not provided on the command line.

1.3.4 Output Files: Time Log Output File

The time log out file `solomons_timefile.out` can be overwritten, and will look something like this:

```
Operations started -- day 1:  hr 10:  min 4:  sec 16
Operations finished -- day 1:  hr 10:  min 4:  sec 18
Operations took 2 seconds
```

This file shows the day, hour, minute and second on which Drifter started and finished its calculations. The final line indicates the total time taken in seconds. (The output files shown above for `solomons.input` actually took much less than one second to run). The time log option may be useful for calculating how long a large simulation will take based on a test simulation with fewer iterations.

1.3.5 Output Files: Vertical Output File

The output of the simulation runs can be output in two forms: *vertical* and *horizontal.* Drifter automatically defaults to *vertical* format, unless another format is specified on the command line. The output file in the *vertical* format has `_vertical.out` appended to the input filename. Output data files cannot be overwritten – if you try to do so, the programme will abort.

For the input file `solomons.input`, the *vertical* format output file `solomons_vertical.out` will look something like this:

```
Number of iterations:  2
Number of generations:  2

                         K        M        O        P
Malaita                  0.08333  0.25000  0        0.66667
Rendova                  0        0        0.45000  0.55000
;
Malaita                  0.16667  0.08333  0        0.75000
Rendova                  0        0        0.65000  0.35000
;
```

The first line indicates the number of iterations run by Drifter, and the second line indicates the number of generations for which the dataset underwent genetic drift. The following sections, separated by semicolons, list the population data. Final allele frequencies are given for all alleles in the study, including alleles that are not present in all populations (*e.g.*, allele M for the Rendova sample).

1.3.6 Output Files: Horizontal Output File

The *horizontal* output format `solomons_horizontal.out` also cannot be overwritten, and will look something like this:

```
        K        M        O  P                K  M  O        P
Malaita 0.08333  0.25000  0  0.66667  Rendova 0  0  0.45000  0.55000
Malaita 0.16667  0.08333  0  0.75000  Rendova 0  0  0.65000  0.35000
```

The same information is contained in the *vertical* and *horizontal* output files. However, the columnar nature of the *horizontal* output format is better suited to subsequent analyses of the data in spreadsheet software (for instance, if calculating average values, confidence intervals, etc.)

1.3.7 Output Files: All (Generations) Output File

The *all generations* output format `solomons_allgen.out` also cannot be overwritten, is only available for a single population at a time (due to clarity and size concerns), and will look something like this:

```
    K        O
G1  0.55200  0.44800  G2  0.53900  0.46100  Rendova  0.50000  0.50000
G1  0.54700  0.45300  G2  0.56700  0.43300  Rendova  0.40000  0.60000
```

This file presents frequency information for each generation (G1, G2, ... , Gn) for each iteration, and therefore quickly becomes sizable (say, 10-100 MB) with large numbers of generations and/or iterations. When this file is small (say, alleles x generations < 250), it can be manipulated using standard spreadsheet software. Some scripting ability is required to parse larger files. (This author uses a Perl script called Chopper, which may become available on request).

1.4 References

1. Cox, M.P. and Lahr, M.M. 2006. Y-chromosome diversity is inversely associated with language affiliation in paired Austronesian– and Papuan-speaking communities from Solomon Islands. *American Journal of Human Biology* 18:35–50.

2. Kimura, M. 1968. Evolutionary rate at the molecular level. *Nature* 217: 624–626.

3. Wright, S. 1931. Evolution in Mendelian populations. *Genetics* 16: 97–159.