Project 1 : Genetic of Populations (Wright-Fisher)

[Team n°1]

I. Theory

The main goal of this program is to simulate the temporal evolution of a population represented by N individuals with a genome. In order to do that, we will follow the model of Wright-Fisher. There is a fixed number A of alleles in the population and each individual has a type of allele.

At each generation, we will sort a new population with a size N among the individuals of the last generation according to the multinomial distribution. This distribution is generated by a succession of binomial distribution : if at the time t the number of the alleles 1,…,A are n_1,…,n_A then at the time t + 1 the new number of these alleles are given by the random numbers k_1,…,k_A generated by this multinomial distribution.

$$
\begin{aligned}
k_1 &\sim \text{Binom}\left(N, \frac{n_1}{N}\right) , \\
k_2 &\sim \text{Binom}\left(N - k_1, \frac{n_2}{N - n_1}\right) , \\
k_3 &\sim \text{Binom}\left(N - k_1 - k_2, \frac{n_3}{N - n_1 - n_2}\right) , \\
&\dots \\
k_A &= N - \sum_{i=1}^{A-1} k_i ,
\end{aligned}
$$

The alleles can be represented by 4 nucleotides (A, T, G or C) if we have a fasta reader or just by numbers (1, 2, 3, etc.) if it is not the case.

We chose as en extension the mutations and selection. The selection is possible on all case whereas the mutations are possible only if there is a fasta file. At each generation, each nucleotide in each individual has a probability μ to mutate to another nucleotide and a probability 1-μ not to mutate (μ is very small). We have two models of mutations :

$$
\begin{array}{c} \\ \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array}
\begin{array}{c}
\begin{array}{cccc} \text{A} & \text{C} & \text{G} & \text{T} \end{array} \\
\begin{pmatrix} 0 & 1/3 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 1/3 \\ 1/3 & 1/3 & 0 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 \end{pmatrix}
\end{array}
\times \mu
$$

Jukes-Cantor: no parameter

$$
\begin{array}{c} \\ \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array}
\begin{array}{c}
\begin{array}{cccc} \text{A} & \text{C} & \text{G} & \text{T} \end{array} \\
\begin{pmatrix} 0 & \frac{1-\delta}{2} & \delta & \frac{1-\delta}{2} \\ \frac{1-\delta}{2} & 0 & \frac{1-\delta}{2} & \delta \\ \delta & \frac{1-\delta}{2} & 0 & \frac{1-\delta}{2} \\ \frac{1-\delta}{2} & \delta & \frac{1-\delta}{2} & 0 \end{pmatrix}
\end{array}
\times \mu
$$

Kimura model:
1 parameter
(δ>1/3)

For each nucleotide, the number of mutations will be a random binomial number with a mean equal to μ_{site} and a size equal to the number of this nucleotide in the population.

Selection means that each allele has a fitness coefficient s_j which can increase or decrease the probability of transmission to the next generation :

- $0 < s_j$ for positive selection (favorable allele)
- $-1 < s_j < 0$ for negative selection (unfavorable allele)
- $s_j = 0$ is neutral, $s_j = -1$ is lethal

We will generate the next population by replacing each frequency by : $\frac{n_j(1+s_j)}{N+\sum_k n_k s_k}$

II. Examples

There are 2 ways to perform the simulation : with a fasta file or with the others parameters given by TCLAP. The name of the program is PopulationGenetic and the name of the tests are PopulationTest. You can see the results in the folder “build” you created in the file “display.txt” and the documentation Doxygen in the folder “doc” in the file “index.html” from the folder “html”.

1) If you want to use the TCLAP parameters (so without a fasta file), you need to precise:

- The time of simulation or the number of generations with **-T** (default value : 10)
- The number of repetitions of the simulation with **-R** (default value : 2)
- The size of the population with **-N** (default value : 100)
- The number of alleles of the population with **-A** (default value : 2)
- The frequencies of the alleles with **-f** (no default value)
- The fitness coefficient of the alleles with **-S** (default value : 0 for each allele)

The number of frequencies must be equal to the number of allele and the sum of all the frequencies must be equal to 1. However, only the frequency of the allele is obligatory, the other parameters are set by default.

2) If you want to use a fasta file, you can decide to add mutations or not. So without mutations, you need to precise :

- The time of simulation or the number of generations with **-T** (default value : 10)
- The number of repetitions of the simulation with **-R** (default value : 2)
- The name of the fasta file with **-F** (default value : ../fasta/test_for_retrieveData.fasta)
- The marks of the nucleotides of the alleles with **-m** (no default value)
- The fitness coefficient of the alleles with **-S** (default value : 0 for each allele)

However, only the marks of the nucleotides are obligatory because the other parameters are set by default.

3) If you want to use a fasta file and to add mutations, you have two possible ways to perform the simulation :

- You can give the same mutation rate for all the sites indicated by the marks, in this case you only need to precise :
  - The parameters for a simulation with fasta file without mutations
  - The mutation rate by default with **-D** (default value : 0, should be between 0 and 1)
- You can specify the mutation rate of the sites one by one, in this case you also need to precise :
  - The parameters for a simulation with fasta file without mutations
  - The mutation rates with with **-M** for each site specified (should be between 0 and 1)
  - The marks of the sites mutated with **-s**
  - The mutation rate by default with **-D** (default value : 0, should be between 0 and 1) for all the sites you didn't specify

Examples :

- Command without a fasta file : ./PopulationGenetic -T 50 -R 2 -N 100 -A 3 -f 0.2 -f 0.3 -f 0.5 (without selection)

Here we set a population of size 100, with 2 replications, during 50 generations, with 3 alleles in which the frequencies are 0.2, 0.3 and 0.5.

Results : (10 first generations)

0	0.2|0.3|0.5|	0.2|0.3|0.5|
1	0.17|0.22|0.61|	0.32|0.18|0.5|
2	0.24|0.18|0.58|	0.31|0.16|0.53|
3	0.19|0.14|0.67|	0.39|0.1|0.51|
4	0.17|0.14|0.69|	0.47|0.06|0.47|
5	0.18|0.17|0.65|	0.42|0.07|0.51|
6	0.21|0.19|0.6|	0.4|0.04|0.56|
7	0.3|0.1|0.6|	0.42|0.03|0.55|
8	0.34|0.12|0.54|	0.39|0.04|0.57|
9	0.32|0.12|0.56|	0.38|0.04|0.58|
10	0.3|0.15|0.55|	0.33|0.05|0.62|

- Command without a fasta file : ./PopulationGenetic -T 50 -R 2 -N 100 -A 3 -f 0.2 -f 0.3 -f 0.5 -S -1 -S 0 -S 0.4 (with selection)

Same parameters as the previous command but we also add coefficients of selection for each allele (-1, 0 and 0.4).

- Command with a fasta file : ./PopulationGenetic -F ../test.fa -m 2 -m 4 -m 8 -T 100 -R 2 -M 0.01 -s 2 -d 0.5 -D 0.0001 (with mutation)

Here we work with the fasta file test.fa, and we will perform the simulation during 100 generations with 2 replications. We also add one coefficient of mutations (0.01) on the site “2”, a delta coefficient for the mutations 0.5 and a coefficient of mutations of 0.001 for the other sites of mutations.

III. Design and Test

1) Design

Our program consists in 5 classes : Simulation, Population, random, Display and FastaReader. The most important one, Simulation, makes the simulation and is called in the main.cpp with the TCLAP parameters. It has as attributes a vector of Population for the different repeats, the time of the simulation (or the number of generations), the different coefficients corresponding to the TCLAP parameters, the number of repeats and the number representing the current generation. When it performs the simulation, it calls the function “step” of the class Population which updates the number of alleles of the population and it calls the class Display which prints the number of alleles in the current flow. It also calls the function “mutation” of Population which generates mutations in case you enter a fasta file.

The class Population has as attributes a map of pair of string and double for the “name” and the frequency of each allele, and its size. Its function “step” updates the number of alleles replacing them by the numbers generated by the multinomial distribution.

In order to do that, it calls the function multinomial of the class random, which generate random numbers. The class FastaReader is called in the main.cpp to create a new Simulation if you want to use a fasta file. This class “reads” the fasta file you provided and give you the size of the population, the number of alleles and the “name” of the alleles (or simply the three nucleotides).

2) Tests

There are several tests in the program : we test the class FastaReader with its function which “reads” the fasta file and gives the alleles, the function multinomial and its operation, the function displayGen of the class Display which displays the number of the alleles and the mutations of one population during a simulation.

For example, we verify with the test “multinomial_average_freq” that the initial frequency of each allele of one population is equal to the average of all the frequencies generated afterwards by the multinomial distribution, if the size of the population and the number of generations (the time of simulation) are very large.