Studying the selection for - or against - mutator alleles

Computational Biology

Adin Pablo Bartoli, Jules Exbrayat

Project of the second part of the course
Computational Biology 2024/25
Master of Science in Industrial and Applied Mathematics

University Grenoble Alpes
Grenoble-INP ENSIMAG
Date of submission: 31 January 2025 at 23:59pm
Professor: FRENOY Antoine

Contents

1 Introduction

We designed and implemented a simple individual-based simulation model of evolution and use it to study certain phenomena. Our model is built with simple demographics; it is a form of **non-neutral Moran process**, which is explained later in this report. An individual is represented as a list of traits (a vector), meaning we only use its phenotype. For the phenotype-to-fitness mapping, we use **Fisher's Geometric Model**. To model mutations, we introduce changes to the phenotype at each time step using a normal distribution. As a result, **both beneficial and deleterious mutations occur continuously**. Our aim with this individual-based simulations of evolution is to study the selection for (or against) mutator alleles. In the end, we studied the influence of other factors/parameters, namely: population size, environmental changes, spatial structure. In order to investigate the impact of spatial structure, we use a 2D grid of size $n \times n$, where each cell represents one individual. This 2D structure enables to visualize in a pleasant and intuitive way the outcomes of our evolutionary experiments. We chose to code the simulation in Java which provide a multi-agent visual representation tool.

2 Modeling the evolutionary process

This section is an overview of the model that we decided to implement. We present the mathematical formulas that we used and an overview of our algorithm.

2.1 Notations

We adopt the geometric Fisher model to measure fitness. Let:

- $d \in \mathbb{N}$ be the dimension of the phenotype space,
- $X \in \mathbb{R}^d$ represent the phenotype.

The optimum fitness for a phenotype is $[0.5]^d$.

2.1.1 Fitness function

The fitness function is defined as:

$$f : \mathbb{R}^d \to [0, 1], \quad X \mapsto e^{-kr(X)} \tag{1}$$

where:

- $k \in \mathbb{R}$ is the slope parameter,
- W_0 is the optimum phenotype vector,
- $r(X) = \|X - W_0\|$ represents the distance to the optimum phenotype.

2.1.2 Random mutation

Mutations are modeled as:

$$\mu : \mathbb{R}^d \to \mathbb{R}^d, \quad \mu(X) \mapsto \max(0, \min(1, X + \mathcal{N}_d(0, I_d\sigma_X))) \tag{2}$$

where σ_X denotes the mutation variance (mutation rate). If **two types** of individuals exist, we may have separate variances σ_{X_1} and σ_{X_2}. More generally, mutation variance can be a time-dependent random process $\sigma_{X_{1,t}}, \sigma_{X_{2,t}}$, allowing mutation rates to evolve over time. This parameter will be varied in our experiments.

2.1.3 Reproduction probability

The reproduction score is defined as:

$$m_r : [0,1] \to [0,1], \quad m_r(W) = e^{c \cdot y} + \epsilon \tag{3}$$

where:

- $c \in \mathbb{R}$ is a scaling factor,
- $\epsilon \sim \mathcal{N}(0, \sigma_r)$ is a noise term with variance σ_r.

Here, we assume a linear relationship between fitness and reproduction score, though this can be modified in our experiments (we mostly used an **exponential relationship**, which is what it is used in the code). The reproduction matrix M_n is defined as:

$$(M_n)_{i,j} = m_r(W_{i,j}) \tag{4}$$

Each individual receives a **reproduction score**, which we normalize to obtain a **probability of reproduction**:

$$p^r_{i,j} = \max\left(\frac{(M_n)_{i,j}}{\sum_{i,j=1}^{n} M_{i,j}}, 1\right) \tag{5}$$

Each time an individual dies, another reproduces according to these probabilities.

2.1.4 Death probability

The probability of death is uniformly distributed:

$$p^d_{i,j} = \frac{1}{n^2} \tag{6}$$

Thus, the number of individuals dying at each time step follows a random process with expectation 1. We can adjust this expectation to c by scaling the probability:

$$p^d_{i,j} = c \times \frac{1}{n^2} \tag{7}$$

2.1.5 Algorithm overview

The algorithm that we set up is quite forward, and is inspired by the Moran process.

Algorithm 1 Individual-Based Evolution Simulation

Input:
- Population of n individuals
- Mutation model with normal distribution
- Fitness function based on phenotype
- Simulation time steps: T

Output: Evolution of the population over T time steps

Steps: **for** $t = 1$ ***to*** T **do**
  for *each individual* **do**
    Apply a random mutation to the phenotype Update fitness based on the new phenotype
  end
  for *each individual* **do**
    Compute death probability Compute reproduction probability
  end
  Normalize reproduction probabilities
  for *each individual* **do**
    Execute a stochastic process for deaths
  end
  for *each deceased individual* **do**
    Select a parent for reproduction based on probabilities Replace the deceased individual with an offspring
  end
end

2.2 The code

To build and execute the code, use the following commands:

```
make build
make run
```

In the Makefile, there are parameters to **enable visualization** and specify **how many times you want to repeat the experiment**.

In the run section of the Makefile, the first parameter determines whether you want to visualize the process on a 2D grid (1 for yes, 0 for no). The second parameter specifies the number of times the experiment should be repeated.

Each time the code is run, we save the results into the file `class_count.csv` in order to be able to run our python plot scripts.

3 Experimental results

3.1 When there is no mutator / both mutation rates are equal

Having the **same phenotype** ($[0]^d$) and **equal population size** $n = 400, n_1 = n_2 = 200$ at the beginning and no mutation, see Figure: 1. The process is simply a random symmetric walk.

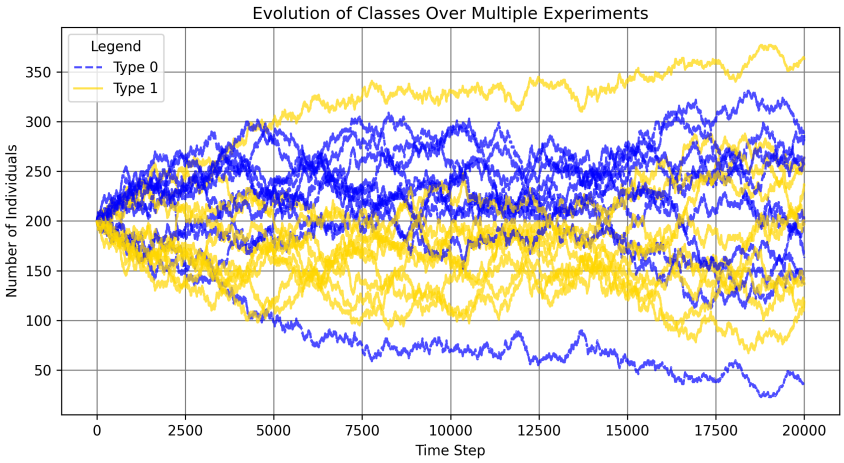

Figure 1: Evolution of the number of individuals with equal population size/phenotype at the beginning and no mutation.

When we vary the mutation rate (μ_0, μ_1) of the 2 populations, we obtain some **first dominance results**. Table 1 illustrates this. We can also look at another kind of graphic to understand what is happening here, we can look at the evolution of the average fitness of both population Table 1, we observe that a very **little fitness advantage makes the populations totally impose themselves**. This illustrates well the fact that **the variance of the mutator allele is very important** when some individuals sample some more beneficial mutations, it gets a better chance of reproduction.

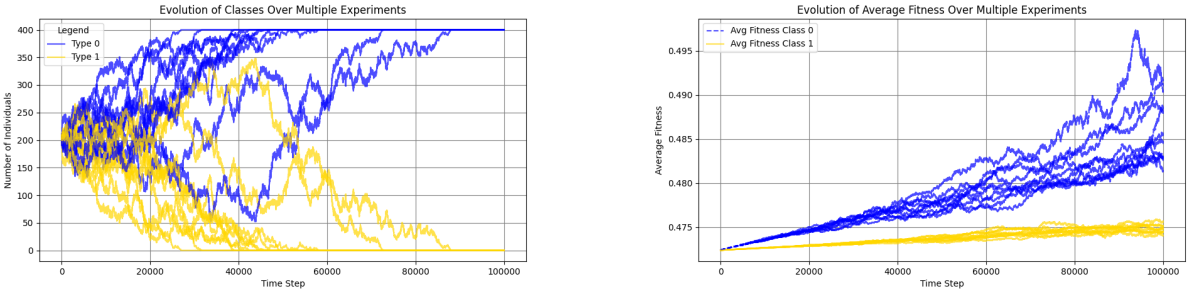

Table 1: Evolution of the number of individuals with equal population size/phenotype at the beginning and $\mu_0 = 10^{-6}, \mu_1 = 10^{-5}$ (left). Evolution of the average fitness of the two types of population (right).

We can also plot the change in the **average fitness**, when the 2 populations have different phenotypes at the beginning. First of all if Type 1 is fitter than Type 0 but then, through mutations the

fitness of phenotype 0 **grows faster towards the fitness pic**. See Figure:2.

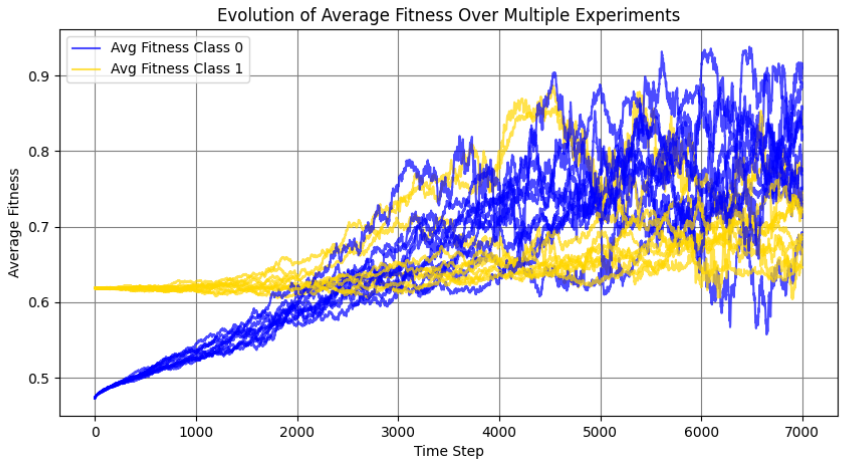

Figure 2: Evolution of the average fitness when the phenotype of the 2 population are different at the beginning.

3.2 The advantage of a higher mutation rate

While genome replication machinery must be reliable for life to reproduce and spread, the mutator phenotype can provide a **meta-fitness advantage**. A population carrying a mutator allele can rapidly reach optimal fitness, outcompeting its less adaptable adversaries. However, **if the mutation rate becomes too high**, the mutator phenotype turns into a **disadvantage**, leading to population decline due to non-viability or deviation from the optimum.

The simulation below demonstrates how a population quickly overtakes its competitor after acquiring the mutator phenotype. Both populations start with the same phenotype (and thus the same fitness advantage) and the same mutation rate. At time step 5000, Population 0 acquires the mutator phenotype.

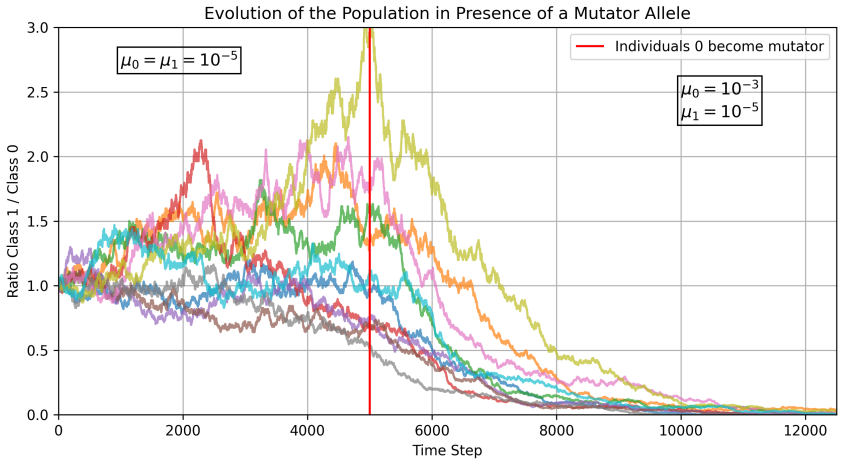

Figure 3: Both populations start with the same size, equal fitness, and a reasonably low initial mutation rate. The *status quo* is disrupted at step 5000 when a mutator allele is introduced in Population 0. Population 0 quickly gains a fitness advantage, ultimately outcompeting Population 1.

3.3 The flaw of excessive mutations

The framework built for our simulations allows us to model the effects of various mutation rates. As shown above, a higher mutation rate can be beneficial, leading to a faster convergence to the fitness optimum. However, it can also be detrimental, as it may prevent the population from remaining at the optimum and produce non-viable offspring.

Our model defines fitness on a scale from 0 to 1, derived from a phenotype-to-fitness mapping based on the Fisher geometric model of evolution. The graph below illustrates how a strain with an excessively high mutation rate eventually becomes less fit, as it fails to reach or maintain the optimum, unlike its competitor.

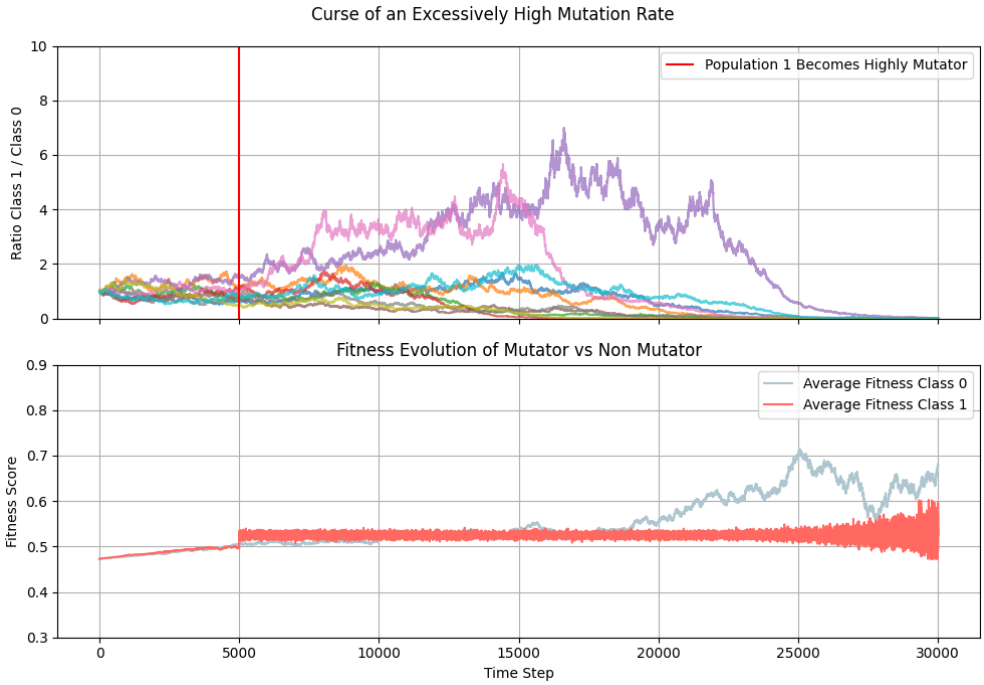

Figure 4: Both populations start with the same size, equal fitness, and a low initial mutation rate. The simulation shows Population 1 declining after transitioning to a highly mutator phenotype. The purple and pink curves illustrate its inevitable loss, despite initially maintaining an advantage, due to luck and stochasticity, until late in the simulation.

3.4 High population size vs mutation advantage

All else being equal, a population **starting with more individuals has a higher chance of outcompeting other populations**. In this section, we analyze how an outnumbered population can still prevail by converging more quickly to the fitness optimum.

The graphs below show that the smaller population initially declines, overwhelmed by the larger one. However, in the vast majority of cases, a beneficial mutation leading to the mutator phenotype **reverses this trend**, ultimately causing the initially larger population to disappear. The two instances where Population 1 became fixed occurred because Population 0 did not have enough time to acquire the mutator phenotype.

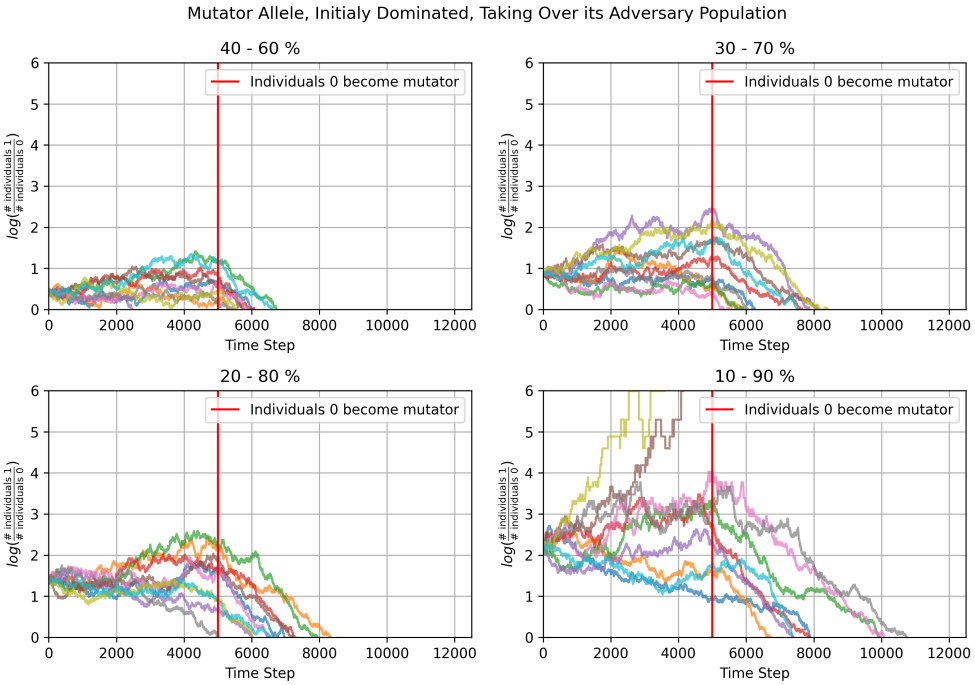

Figure 5: Evolution of populations starting with different sizes in a constrained environment, i.e., with a fixed total population size. The mutator allele enables Population 0 to dominate in nearly all cases, except twice in the 10–90% scenario, highlighting the advantage of a high mutation rate.

3.5 How environmental changes affect evolution

Catastrophic environmental changes can reverse trends, similarly to the emergence of mutator phenotypes, as shown in 3.4. To test this, **we simulated a disease outbreak in Population 1**, which initially had a fitness advantage (a better starting phenotype and a slightly higher mutation rate) compared to Population 0. We wanted to determine whether this advantage would allow the affected population to recover from the disease.

The results were **highly variable**. In four out of ten simulations, Population 1 recovered and even exhibited exponential growth, similar to the scenario depicted in Figure 3 of the Nature article *Role of Mutator Alleles in Adaptive Evolution*[1]. However, in the other six cases, Population 1 went extinct despite having gained a significant advantage before the outbreak.

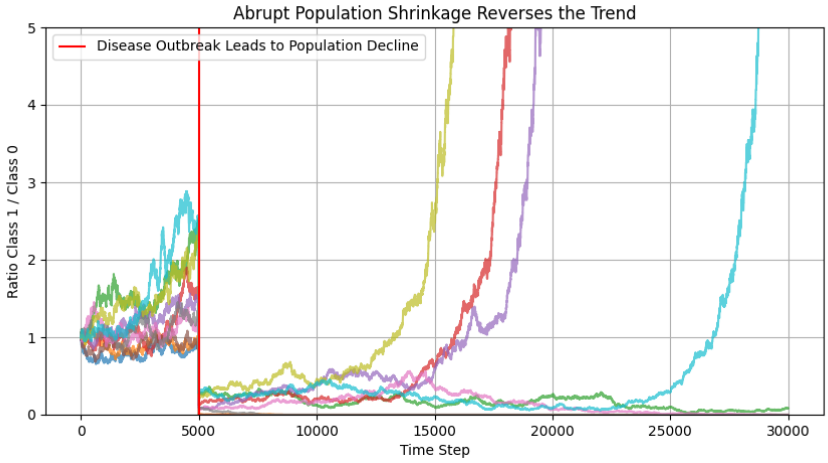

Figure 6: Effect of a catastrophic population decline in a constrained environment. The survival of an advantaged population becomes highly stochastic when its size is drastically reduced.

3.6 Effect of spatial structure

We want to include a spatial structure to our evolutionary model, so as individuals are arranged on an $n \times n$ grid. We want them to reproduce **following a spatially constrained process**. When an individual dies, it is replaced by **one of its eight neighboring individuals** (Moore definition of neighborhood). The probability of being chosen as a parent is proportional to its fitness, ensuring that stronger individuals are more likely to propagate (the same as in the first part of this project). This rule **introduces spatial structure, preventing highly fit individuals from dominating the entire grid too quickly while preserving local adaptation dynamics**.

We can then use **different spatial structure configuration at the beginning** to see which ones are more prone to high mutator dominance.

It is easy to run the different type of simulations by playing with `config` parameter in the main

3.6.1 Split the landscape in half

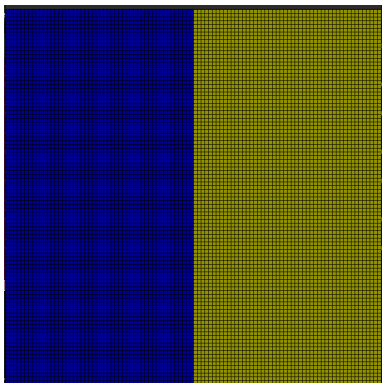

Figure 7: Landscape split in half

The takeover is very slow because types are **completely clustered**. This makes it very difficult for the mutators to invade the population. The main reason is the lack of contact between the two populations.

3.6.2 Cluster of mutators

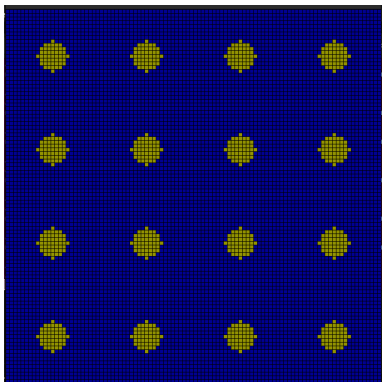

Figure 8: Big clusters

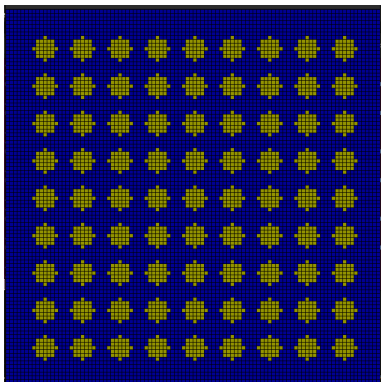

Figure 9: Small clusters

The more the clusters are small, the more the takeover is fast. **Due to the spherical boundary, the individuals on the boundaries have more contact with individuals of the other type, hence it makes it hard to survive when the cluster contour of the shape are smooth. In our opinions it will be better to have non smooth contours (squares, triangle, ...) for attack.**

3.6.3 Diagonal

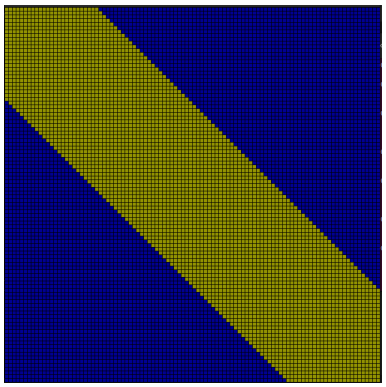

Figure 10: Diagonal set up

The takeover is quite fast has the mutators cuts the non mutators into 2 making them easy to attack. Almost two times faster then the splitting in 2 set up.

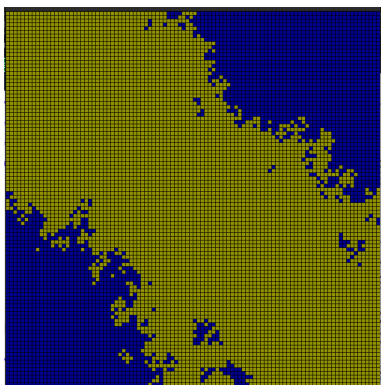

Figure 11: Diagonal set up after 600000 time step

3.6.4 Random

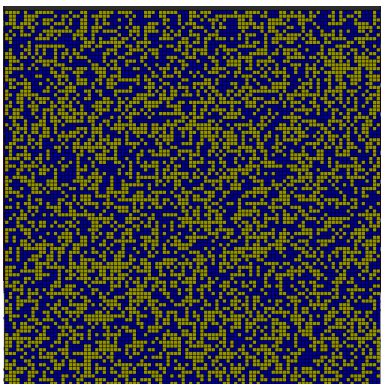

Figure 12: Random set up

The takeover is **extremely fast has the mutators are everywhere and impose themselves progressively**. It is approximately **10 times faster** than a diagonal set up.

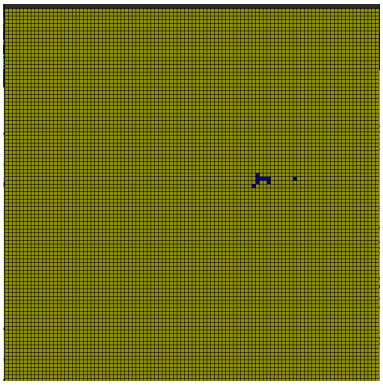

Figure 13: Random set up after 600000 time step

3.6.5 Ranking

Here is the ranking of how fast mutators invade the population in different initial set ups. The fastest is the Random set up, then the cluster set up, and the diagonal set up and finally the slipping into 2 set up.

Basically the speed of invasion of the mutators is dependent of the initial spatial set up of the population. **The more mixture there is between the 2 types the faster the mutators take over.** This could lead to some different study, like how can the non mutator, or the less fit individuals increase their chance of survival by positioning them self? Can we find pertinent strategies to win every time?

4 Conclusion

Basing ourselves on a very simple model, we tried to replicate the experiments from the two articles in the bibliography. We managed to get a good overview of the influence of mutator alleles, evaluating their strengths and weaknesses. Overall, in most setups, having a larger mutation rate is a significant advantage. It allows for more diverse mutations and hence a faster reach to the fitness peaks.

We also concluded that the spatial structure makes it much harder for mutators to conquer the space. Since reproduction is a local phenomenon, they need to win everywhere.

If you want to try the simulation, the commands are simply : `make build` then `make run`.

References

[1] Taddei, F., Radman, M., Maynard-Smith, J., Toupance, B., Gouyon, P.H., & Godelle, B. (1997). Role of mutator alleles in adaptive evolution. *Nature*, 387(6634), 700-702. `https://doi.org/10.1038/42696`

[2] Competition Between High and Low Mutating Strains of Escherichia coli, Author(s): Lin Chao and Edward C. Cox. `https://perso.crans.org/frenoy/compbiol/articles/Chao1983`.