Checking correctness

Boris Prochnau

5. Mai 2014

Inhaltsverzeichnis

Introduction

To validate that the program works correctly I have made a suite of tests. Some of the test try to verify the correctness of the algorithm and therefore could be interesting for you. Most of the calculations work with the same instance and every tests resets its result before the next test starts. The intention is to make them as independent as possible. For making calculations and changes to the population only the implemented functions of the program were used. The testing functions themselves do not effect the population in any sense.

1 Verify correct rates

The first tests work with this instance:

$$\text{Mutation probability} = 0.1 \qquad \text{Amount of traits} = 3$$

$$\text{Competition rates} = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 0.5 \\ 0 & 2 & 2 \end{pmatrix}$$

$$Members = \begin{pmatrix} 100 \\ 100 \\ 100 \end{pmatrix} \qquad Brithrates = \begin{pmatrix} 10 \\ 10 \\ 10 \end{pmatrix} \qquad Deathrates = \begin{pmatrix} 5 \\ 5 \\ 5 \end{pmatrix}$$

1.1 calculate intrinsic death rates

Testresult: **intrinsic death rates =** $\begin{pmatrix} 500 \\ 500 \\ 500 \end{pmatrix}$

To verify the result we consider:

$$\text{Deathrate*Members} = 5 \cdot 100 = 500$$

1.2 competition death rates

Testresult: **competition death rates =** $\begin{pmatrix} 30000 \\ 25000 \\ 40000 \end{pmatrix}$

To verify the result we consider a calculation for the trait „2“:

$$\text{Members(2)} \cdot \sum_j \text{Competitiondeathrate(2,j)*Members(j)}$$
$$= 100 \cdot (2 \cdot 100 + 0.5 \cdot 100) = 25000$$

1.3 calculate total deathrates

Testresult: **intrinsic death rates =** $\begin{pmatrix}30500\\25500\\40500\end{pmatrix} = \begin{pmatrix}500\\500\\500\end{pmatrix} + \begin{pmatrix}30000\\25000\\40000\end{pmatrix}$
This result is not surprising after the previous results and will not be explained further.

1.4 calculate total birthrates

Testresult: **total death rates =** $\begin{pmatrix}1050\\1100\\1050\end{pmatrix}$
To verify the result we consider a calculation for one trait „2“:

Births(2)*Members(2) + mutation*(Births(1)*Members(1)+Births(3)*Members(3))
$= 10 \cdot 100 + 0.1 \cdot (10 * 100 + 10 * 100) = 1000 + 0.1 \cdot 2000 = 1000 + 100$

For „i = 1“ and „i = 3“ we consider that the mutation happens only for the one existing neighbor what explains the loss of 50.

1.5 calculate total trait rates

Testresult: $\begin{pmatrix}31550\\26600\\41550\end{pmatrix} = \begin{pmatrix}1050\\1100\\1050\end{pmatrix} + \begin{pmatrix}30500\\25500\\40500\end{pmatrix}$
This result is also not surprising after the previous results and consists of birth and deathrates.

1.6 calculate total event rate

Testresult: **total event rate =** 99700 = 31550 + 26600 + 41550 This is just the summation over the total trait rates.

2 sampling of event time

Here is only one test mentionable where I verified that many samples from the same total event rate don't repeat and their average converges to the mean of the corresponding exponential distribution $\frac{1}{99700}$.

3 changing a trait

Here we will test whether the trait is chosen correctly and the correct event will be executed correctly on it. In order to chose traits and events, i.e. to extract the first ringing clock, we use superposition of the Poisson Point Process. For doing so the results of section 1 (the rates) are used.

3.1 choose a trait

To verify that the traits are chosen like planed, I choose 100000 traits with the same rates and verify their correct distribution among each other.

Testresult: **approximation** $= \begin{pmatrix} 0.31633 \\ 0.26867 \\ 0.415 \end{pmatrix}$ and **expected** $= \begin{pmatrix} 0.316449 \\ 0.266800 \\ 0.416750 \end{pmatrix}$

The approximation was calculated through a histogram of the occurrence of the chosen traits divided by the number of iterations. The expected value was calculated through

$$\frac{\text{total trait rate of trait i}}{\text{total event rate}} = \frac{1}{99700} \cdot \begin{pmatrix} 31550 \\ 26600 \\ 41550 \end{pmatrix}$$

3.2 chose event type

To verify that the events are chosen correctly I summed 100000 birth events and a chosen trait and compared it with the expected distribution.

Testresult: **approx. births** $= \begin{pmatrix} 0.03346 \\ 0.04057 \\ 0.02562 \end{pmatrix}$ and **expected births** $= \begin{pmatrix} 0.0332805 \\ 0.0413534 \\ 0.0252708 \end{pmatrix}$

The approximation was calculated through the number of births for trait „i“ divided by the number of iterations. The expected value was calculated through

$$\frac{\text{total birth rate of i}}{\text{total trait rate of i}} = \begin{pmatrix} 1050/31550 \\ 1100/26600 \\ 1050/41550 \end{pmatrix}$$

3.3 executing event type on trait

To verify that the chosen trait was changed like the chosen event told it to, I simply checked if a birth/death has been occurred after using the corresponding function. The chosen trait has changed its members by 1 so everything works fine here.

3.4 changing a trait

This tests simply executes the „changing a trait“ function and verifies that the chosen trait has been changed. It simply ensures the collaboration of the previous steps.

4 convergence tests

I know how to calculate the convergence of populations without mutations and in competition to each other, up to 2 different traits. I made 3 different tests. One tests the convergence of 3 independent (no competition or mutation) traits

and the other two tests 2 Traits in competition to each other with no mutation. The last one only differ in the evaluation technique.

4.1 Monomorphic population

I used the following input:

$$\text{Mutation probability} = 0 \qquad \text{Amount of traits} = 3$$

$$\text{Competition rates} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$Members = \begin{pmatrix} 110 \\ 110 \\ 110 \end{pmatrix} \qquad Brithrates = \begin{pmatrix} 100 \\ 120 \\ 140 \end{pmatrix} \qquad Deathrates = \begin{pmatrix} 10 \\ 10 \\ 10 \end{pmatrix}$$

To verify the convergence I made 10 000 000 iterations. I created an array that sums the changed members for each trait separately and calculated their average.
Testresult:

$$\textbf{average members} = \begin{pmatrix} 89.8175 \\ 110.337 \\ 130.398 \end{pmatrix} \text{ and } \textbf{expected members} = \begin{pmatrix} 90 \\ 110 \\ 130 \end{pmatrix}$$

My observations where that the computed average is usually slightly higher than the expected and does not improve much with more iterations. One of the next goals would be to figure out where this behavior comes from. Besides that the actual average is calculated with a summation of changed members like described above. After the summation the results got divided by the number of changes for each trait individually. The expected is just

$$\frac{birthrate(i) - deathrate(i)}{competitionrate(i)}$$

4.2 Dimorphic population

I used the following input:

$$\text{Mutation probability} = 0 \qquad \text{Amount of traits} = 2$$

$$\text{Competition rates} = \begin{pmatrix} 2 & 0.5 \\ 1 & 1.5 \end{pmatrix}$$

$$Members = \begin{pmatrix} 100 \\ 100 \end{pmatrix} \qquad Brithrates = \begin{pmatrix} 170 \\ 150 \end{pmatrix} \qquad Deathrates = \begin{pmatrix} 15 \\ 5 \end{pmatrix}$$

To verify the convergence I made 10 000 000 iterations. todo
Testresult:
average members $= \begin{pmatrix} 64.7513 \\ 55.8959 \\ a \end{pmatrix}$ nd **expected members** $= \begin{pmatrix} 64 \\ 54 \end{pmatrix}$ My observations for the dimorphic case are the same as for the monomorphic.
But when I used another method of calculating the average I got this result:
average members $= \begin{pmatrix} 63.7036 \\ 53.904 \\ a \end{pmatrix}$ nd **expected members** $= \begin{pmatrix} 64 \\ 54 \end{pmatrix}$ These results are usually slightly lower(!) and their precision is better. I will bring this topic up in the next meeting on the 06.05 because this issue could disappear with the usage of an smoothing „K“ value.