REvoSim Documentation

Release 2.0.0

Mark Sutton, Russell Garwood, Alan R.T. Spencer

Feb 13, 2019

Contents

The [R]apid [Evo]lutionary [Sim]ulator program.

REvoSim is an individual-based evolutionary model, using a simplified first-principles evolutionary model to facilitate high computational efficiency, enabling the simulation of large populations incorporating space, over geological time, using modest computer hardware. It can simulate populations of 10^5 –10^7 digital organisms over geological timescales, and incorporates spatial and temporal environmental variation, recombinant reproduction, mutation and dispersal. Speeds attainable depend on the computer hardware in use, the size of the populations simulated, and details of the experimental setup (most especially on whether species tracking and fitness recalculation are activated). With a typical 2018 desktop computer, speeds of between 500,000 and 1,000,000 iterations per hour can be achieved for populations of around 250,000.

REvoSim has been in development since 2008, and has been released with the intention that it can be used as a multipurpose platform for the study of many evolutionary phenomena (both macro- and microevolution). While it was designed with macroevolutionary studies in mind, it is also applicable to microevolutionary problems. As such it is complementary to the many other approaches of studying evolution on a range of different timescales, and will be continually developed by the core team to expand its capabilities.

t:@palaeoware

w:https://github.com/palaeoware.

CHAPTER 1

Relevant references

Garwood, R.J., Spencer A.R.T. and Sutton, M.D., 2019. REvoSim: Organism-level simulation of macro- and microevolution. Palaeontology.

CHAPTER 2

Table of Contents

2.1 Introduction

2.1.1 Overview

REvoSim can be used to study a range of evolutionary processes. It is based on 64-bit digital organisms, within an environment defined by the RGB values of a two-dimensional image. It is highly abstracted, and is designed for computational efficiency. For versatility there are a large number of user-defined variables: an overview is provided below. This assumes the software has already been installed (instructions can be found on the page *Compiling, Installation, and Requirements*). We recommend reading the paper below for full discussion of REvoSim's approach, potential limitations, and its strengths:

Garwood, R.J., Spencer A.R.T. and Sutton, M.D., 2019. REvoSim: Organism-level simulation of macro- and microevolution. Palaeontology.

A utility program, EnviroGen, is available to generate environments for REvoSim and thus provide control of a number of the environmental controls on the simulation. This is documented separately.

In brief, controls for the simulation are found on the toolbar at the top of the main window. Hovering a mouse over each toolbar button (or any other areas for user input in REvoSim) will provide an overview of what it does. The buttons are as follows:

Run Launch a simulation

Run for Launch a simulation and then allow it to continue for a set number of iterations.

Batch Repeat Run for *n* times.

Pause Pause a simulation.

Stop Cancel a simulation.

Reset Reset the simulation and reseed with a random digital organism in the central pixel.

Reseed Launch a dialogue to allow the simulation to be reseeded with a known genome, or with two individuals that share a (random one, or user defined) genome.

Genome Launch Genome Comparison Dock which allows genomes to be inspected and compared.

Settings Launch Settings Dock which allows variables to be defined.

About Launch dialogue with information about REvoSim.

The main part of the window comprises two panels:

Population view Provides an overview of the population alive at any given polling iteration. The information shown can be selected with a drop down menu at the top.

Environment Shows the RGB environment which is used to calculate organism fitness.

Below this is the *Information Bar*, which shows a number of statistics for the given run, updated each polling iteration. These include population size, number of species, iterations and speed. You can find more information on the *Window Layout* pages.

2.1.2 Variables

Variables can be defined within the settings dock on the right. Full descriptions of these and their implications can be found in the REvoSim paper. Clicking settings on the toolbar at the top of a window toggles the visibility of this dock. At the bottom of the dock are three tabs, each of which has variables associated with different aspects of the simulation.

Organism tab This includes the variables which dictate the behaviour of the digital organisms in a REvoSim run. This includes chance of mutation, starting age (i.e. length of life), breed threshold and cost, mode of breeding, and breed settings. More information: *Configuring your Organisms*

Simulation tab This includes the settings for the environment and associated files, simulation size, fitness target (i.e. the nature of the fitness lanscape), energy input, settle tolerance, and species tracking. More information: *Setting up the Simulation*

Output tab This includes output options for the simulation: save directory, refresh rate, and logging/output options. More information: *Configuring your Outputs*

2.1.3 Quick start

A simulation - using default settings and environment - can be started by hitting the Run button. In addition to the visualisation, runs can be analysed using log files which are placed by default in a folder called *REvoSim_output* on the desktop for all operating systems. A log is written during a run when "Write Log Files" (Settings dock, Output tab) is checked, and a the phylogenetic tree and other more detailed statistics for a run can be written at any point by clicking the button "Write data".

2.2 Compiling, Installation, and Requirements

2.2.1 Compiling from Source

Windows 64-bit

QT Creator + QT v5.x using MSYS2 (64-bit) and MinGW (64-bit) We recommend you install and use MSYS2 (64-bit) a Windows package manager, based on modern Cygwin (POSIX compatibility layer) and MinGW-w64, that allows easy installation of QT v5.x 64-bit.

1. Download and run the latest version of MSYS2 for 64-bit Windows. This will be name "mysys2-x86_64-..." for the 64-bit installer.

2. Follow the install instructions. We have used the default install location of "C:mysys64" and it is here that includes required in the .pro files point. If you install MSYS2 to another location the .pro files will need to be updated to your install location.
3. Once installed open up MSYS2 shell and run the pacman update command: pacman -Syu Note that as this will almost certainly update pacman itself you may have to close down and restart the MYSYS2 shell before re-running the command to finish.
4. Once MSYS2 and pacman are fully updated run the following command to install QT 5.x and its dependencies: pacman -S mingw-w64-x86_64-qt-creator mingw-w64-x86_64-qt5
5. Optional - if you intend on debugging the software in QT and wish to use GDB then run the following to install the matching GBD debugger: pacman -S mingw-w64-x86_64-gdb
6. **At this stage you should have the following under the MYSYS2 install location:**
    - {install location}/mingw64 (Main ming64 folder)
    - {install location}/mingw64/bin/qmake.exe (QMake for QT version)
    - {install location}/mingw64/bin/g++.exe (C++ complier)
    - {install location}/mingw64/bin/gcc.exe (C complier)
    - {install location}/mingw64/bin/gdb.exe (Debugger | OPTIONAL)
7. You should now be able to find the required libraries under "{install location}/mingw64/bin" and the required header (.h) files for QT v5.x.
8. Open the .pro file in QT Creator, and then use the information above to setup a new 64-bit ming64 kit. Follow standard QT Creator debug/release procedure.

Ubuntu 18.04 64-bit *QT Creator + QT v5.x and using GCC (64-bit)*

1. The simplest way to install Q5.X on your system is to download and run the installer from Qt. Further instructions are available from the Qt website
2. Open the .pro file in QT Creator, set up an appropriate kit using the installed version of QT, and follow the standard debug/release procedure.

MacOS *QT Creator + QT v5.x*

The above (Linux) approach should also work for MacOS builds.

2.2.2 Installation

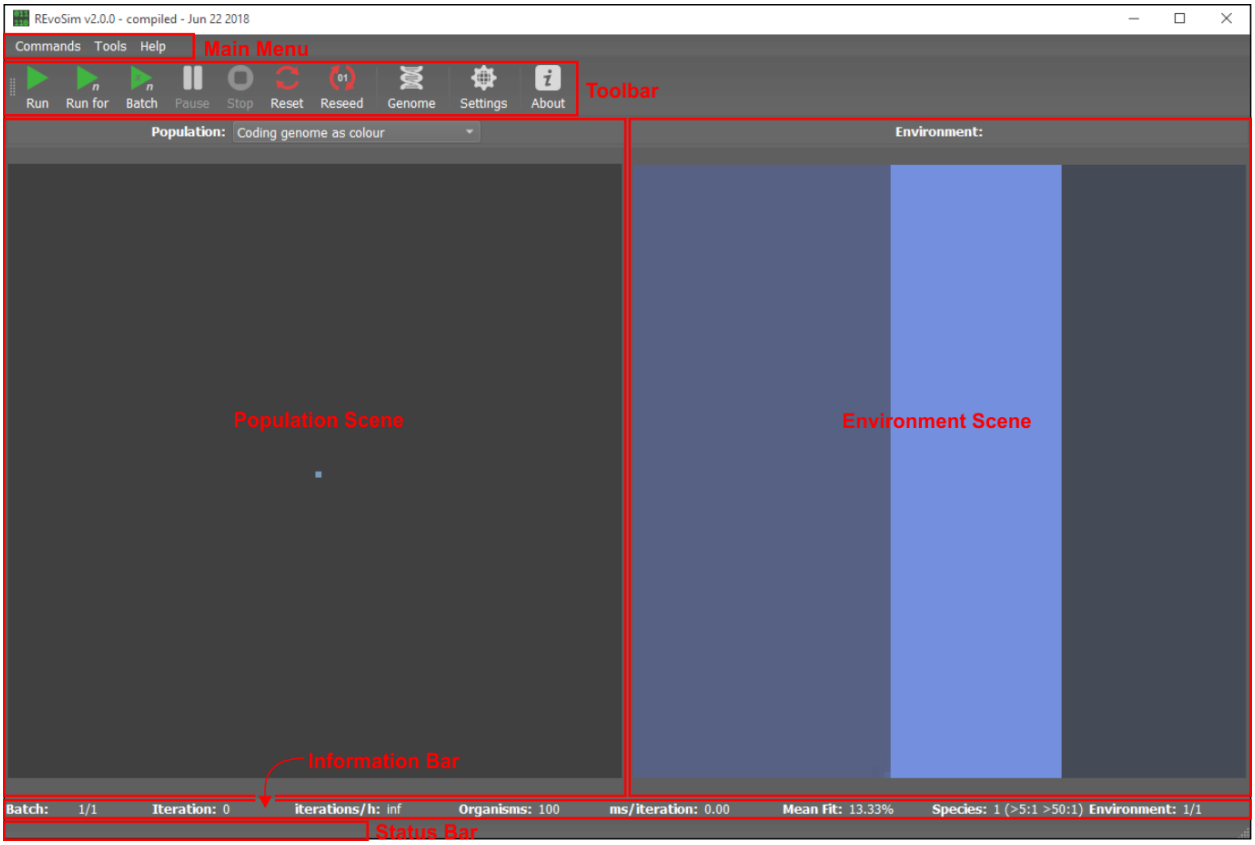
From the REvoSim GitHub repository pre-compiled binary releases and packaged installers can be downloaded. For Windows users we provide both a portable binary release (.zip) - which just needs extracting to a convinent location - and a self contained installer.

2.2.3 Requirements

REvoSim has no minimum requirements as such, and will run on most standard systems (Windows/Linux/Mac); it however has not been tested on versions of Windows older than Windows 10, Ubuntu 16.04, and macOS High Sierra. Performance will benefit from high processor speed and increased number of processor cores, with large amounts (>4GB) of available RAM recommended for large simulations. Graphics card performance is not relevant as GPUs are not currently used in the program's calculation pipeline. A fast hard drive (e.g. SSD) is recommend when intensive logging is enabled; as slow I/O response time can affect the iteration cycle speed.

We recommend a minimum of 1GB RAM and a 1.8 GHz or faster, ideally multicore processor. We also recommend a minimum screen resolution of 1280x720 if using the software without the genome comparison docker (and 1920x1080 if this is enabled).

2.3 Window Layout



Each of the above mentioned features is documented in more detail on the following pages:

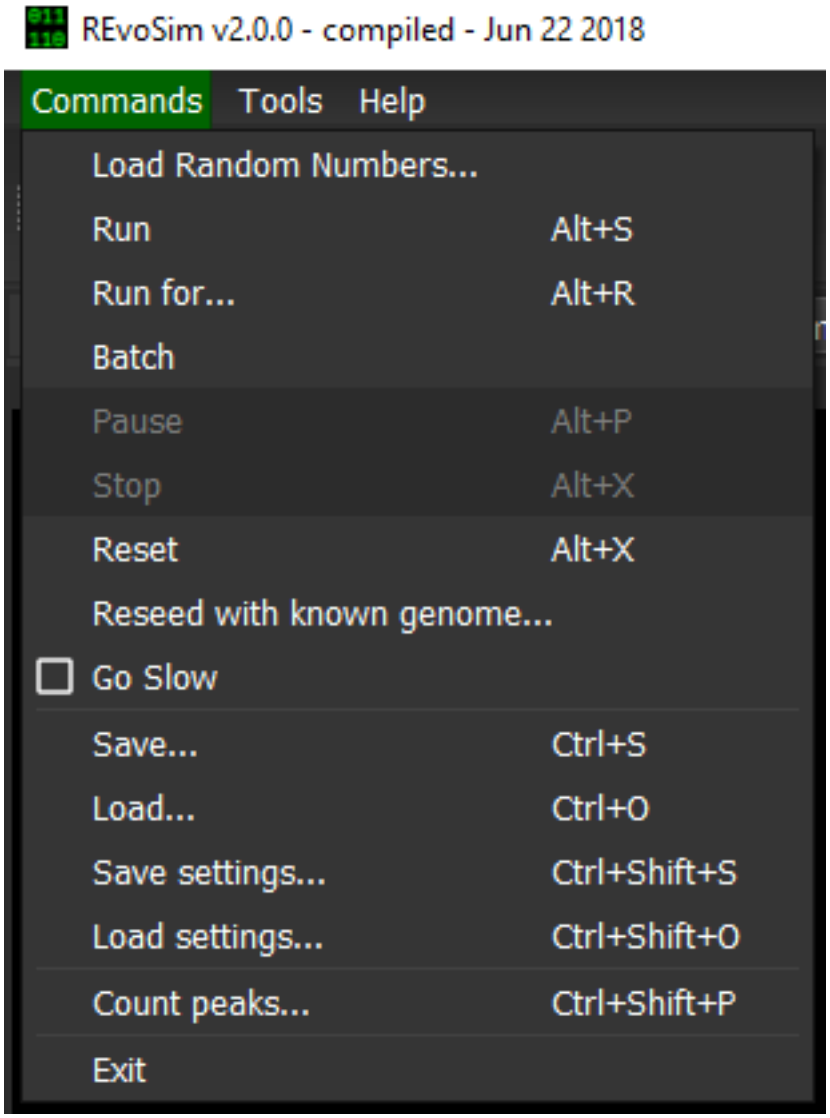
2.3.1 Main Menu

The Main Menu, located at the top of the program window, allows access to all program actions, functions, and settings. The menu is currently sub-divided into three sections:

1. *Commands*
2. *Tools*
3. *Help*

Commands

The Commands menu holds the majority of available actions and program functions related to running of one or more simulations. The options are as follows:


REvoSim v2.0.0 - compiled - Jun 22 2018
Commands
Tools
Help
Load Random Numbers...
Run Alt+S
Run for... Alt+R
Batch
Pause Alt+P
Stop Alt+X
Reset Alt+X
Reseed with known genome...
Go Slow
Save... Ctrl+S
Load... Ctrl+O
Save settings... Ctrl+Shift+S
Load settings... Ctrl+Shift+O
Count peaks... Ctrl+Shift+P
Exit

Load Random Numbers Allows the random numbers incorporated into the REvoSim binary to be overridden with a custom file. See *Custom Random Numbers*.

Run .... Reseed with known These options are provided as alternatives to the buttons on the top toolbar of the GUI.See *Main Toolbar*.

Go slow This option slows the simulation to allow environmental changes and the visualisation in the population view to be viewed more clearly. It achieves this by adding a 30ms delay to every iteration.

Save This saves the current state of the REvoSim simulation, allowing it to be loaded later, including the masks, organisms, and all settings. This is saved as a binary file to allow the minimum file size possible.

Load This loads the above REvoSim file.

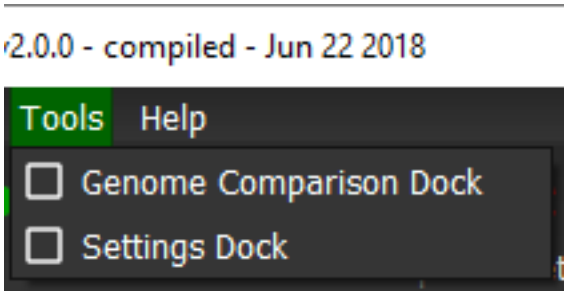
Save settings This saves the settings of REvoSim in a given state. This includes all user-defined variables, but nothing else. These are saved as a human-readable XML file.

Load settings Loads a settings file.

Count peak This is provided to help the user understand the fitness landscape of their run (albeit in simple terms). See *Count peaks*.

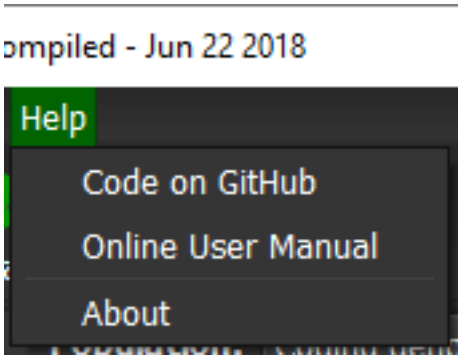
Exit Quits REvoSim.

Tools



The Tools menu allows access to the built-in dockable widgets (called 'Docks') which alter or extend the core program functions. This includes the main simulation settings dock, described in *Configuring your Organisms*, *Setting up the Simulation* and *Configuring your Outputs*, and the genome comparison dock (*Genome comparison dock*).

Help



The Help menu contains links to useful program information, the Palaeoware code repository, and the REvoSim documentation.

2.3.2 Main Toolbar

The main toolbar consists of the following buttons:

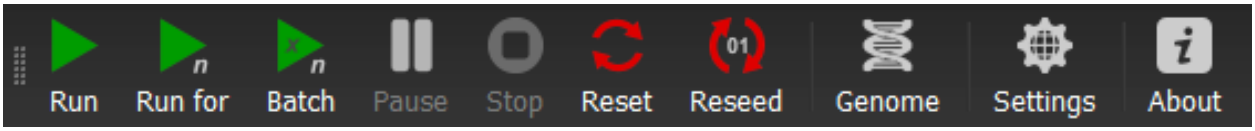


Fig. 1: Figure 3.1.2.1 - Main toolbar (no simulation running).

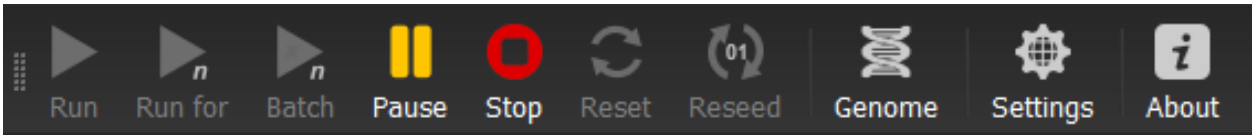


Fig. 2: Figure 3.1.2.2 - Main toolbar (with simulation running).

The first four ("Run", "Run for", "Batch", "Pause" and "Stop") control the initiation and cessation of simulation runs. These commands can also be accessed from the *Commands* menu.

Run This button launches a simulation, and then runs it until it is either pasued or stopped.

Run for This launches a simulation and runs it for a user-defined number of iterations.

Batch For repeated runs using the same settings, REvoSim provides a batch mode: this provides the option of repeating the environment, or continuing from the last environmental file loaded. Logs in batch mode will be labelled accordingly. The number of runs, and for how many iterations these should last, are requested on launching batch mode.

Pause Pauses a simulation, allowing it to be continued when requested.

Stop Stops a simulation and resets the GUI, but leaves the simulation in its current state.

Reset Resets the simulation by removing all digital organisms, and then placing a random individual capable of surviving in the central pixel.

Reseed Launches a dialogue to allow the simulation to be reseeded with a known genome, or with two individuals that share a (random, or user-defined) genome. Not all genomes are capable of surviving in a REvoSim run: if reseeded with a geome incapable of survival, REvoSim will provide an error. To allow reseeding with a known genome - but one which can survive in a given environment, the dialogue provides a list comprising the top ten genomes from the genome comparison dock (which can be populated prior to a given run). See *Genome comparison dock*.

Genome Launch Genome Comparison Dock, described in *Genome comparison dock*.

Settings Launch Settings Dock which allows variables to be defined. See *Configuring your Organisms*, *Setting up the Simulation* and *Configuring your Outputs*.

About Launch dialogue with information about REvoSim, including version number, authors, license information, and contact details for the Palaeoware team.

The toolbar itself can be moved to one of four available positions using drag and drop: top (default), left, right, and bottom of the window. The toolbar can also be undocked from the main window and used as a floating toolbar (i.e. an independent window). To move the toolbar or to undock it as a floating window use the left mouse button on the three

dotted handle (far right of the toolbar by default), then holding the mouse button down drag the window to it desired position.

2.3.3 Population Scene

The Population Scene is the program's visual output showing what is taking place within the defined siumlation space. When the program is first started, or after a Reset is called, the Population Scene will default to showing a single coloured pixel in the center of the population grid, unless Dual Reseed is selected in which case two coloured pixels with be shown. This pixel represents the starting genome of the seeding organism: black pixels are those without any living digital organisms.

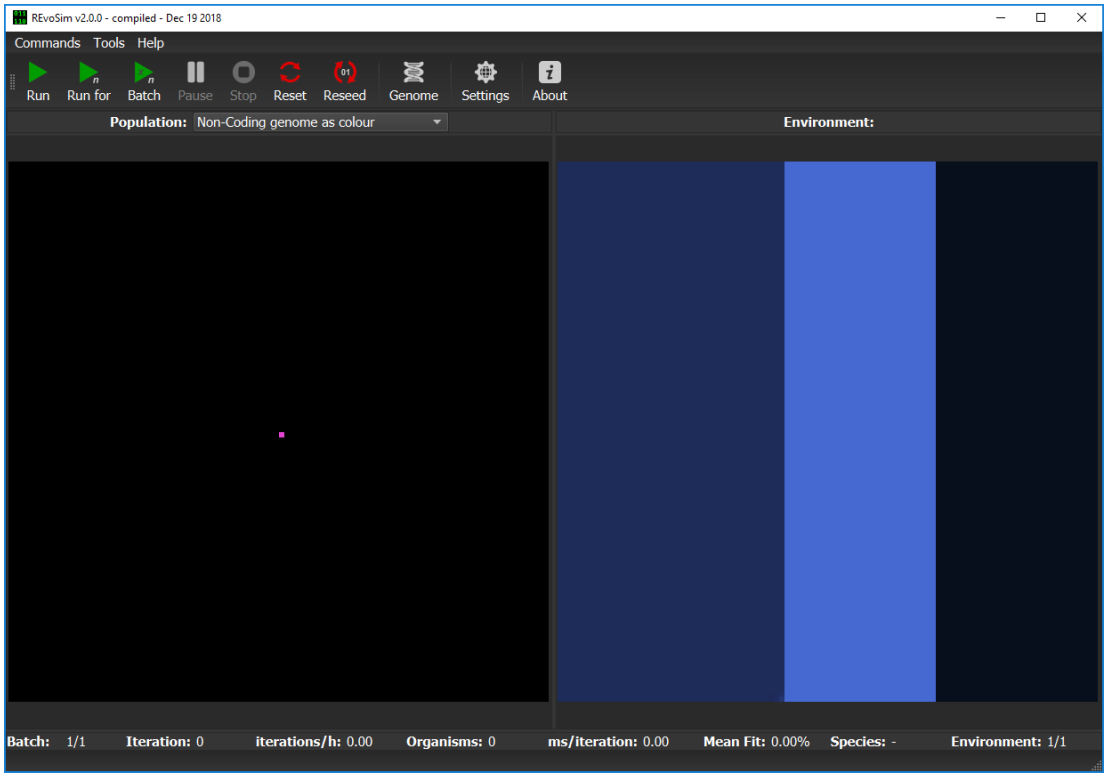


Fig. 3: Figure 3.3.1 - Default Population Scene showing a single starting genome in the centre.

The Population Scene can be set using the dropdown menu to show different output modes. The the Population Scene currently supports the following visual modes:

1. Population Count
2. Mean Fitness
3. Coding Genome as Colour
4. Non-Coding Genome as Colour
5. Gene Frequencies
6. Settles
7. Breed/Settle Fails (R=Breed; G=Settle)
8. Species

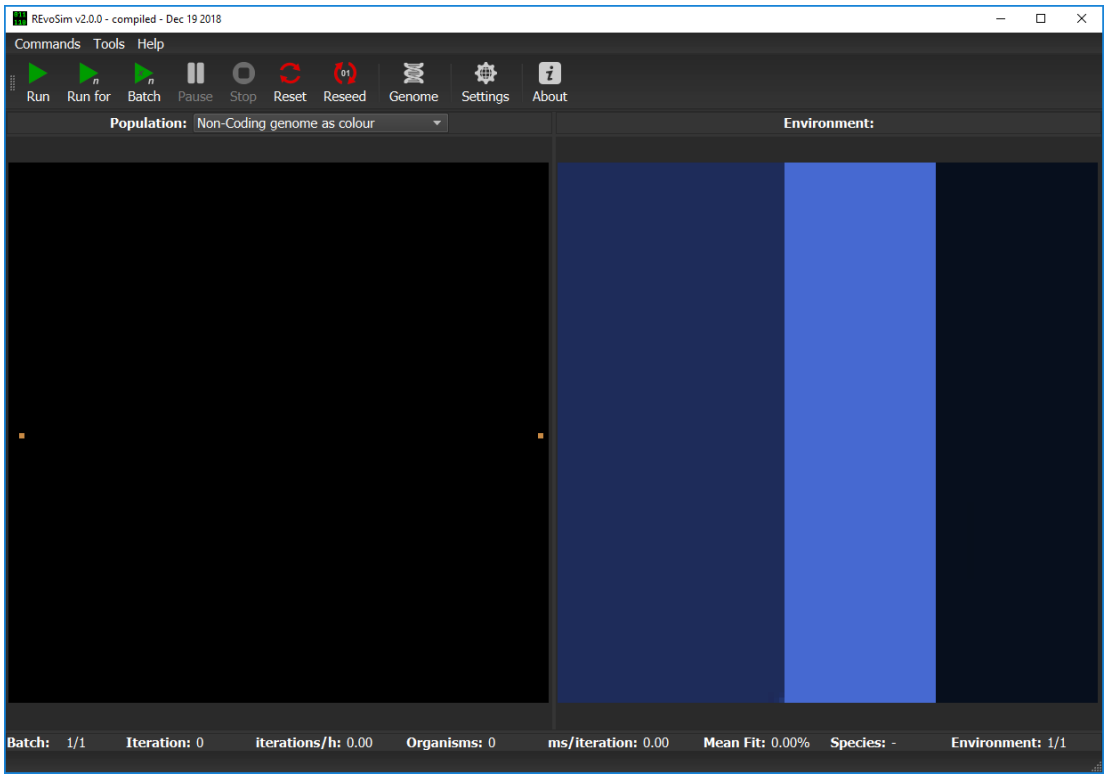
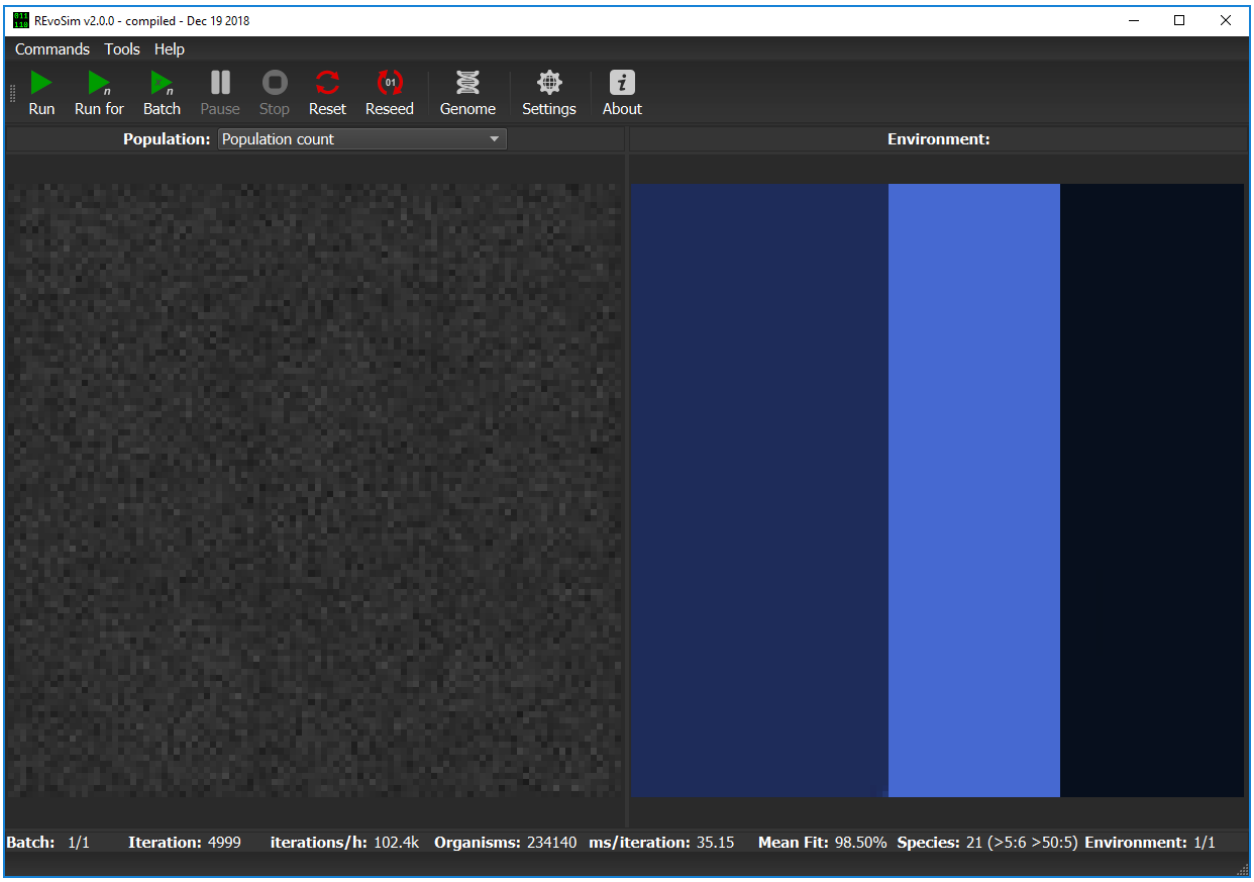


Fig. 4: Figure 3.3.2 - Default Population Scene showing a two genomes, one on left and another on the right, as set by the Duel Reseed option.

Population Count

Each grid-square is visualized with a grey-level representing the current number of creatures alive in the square



Mean Fitness

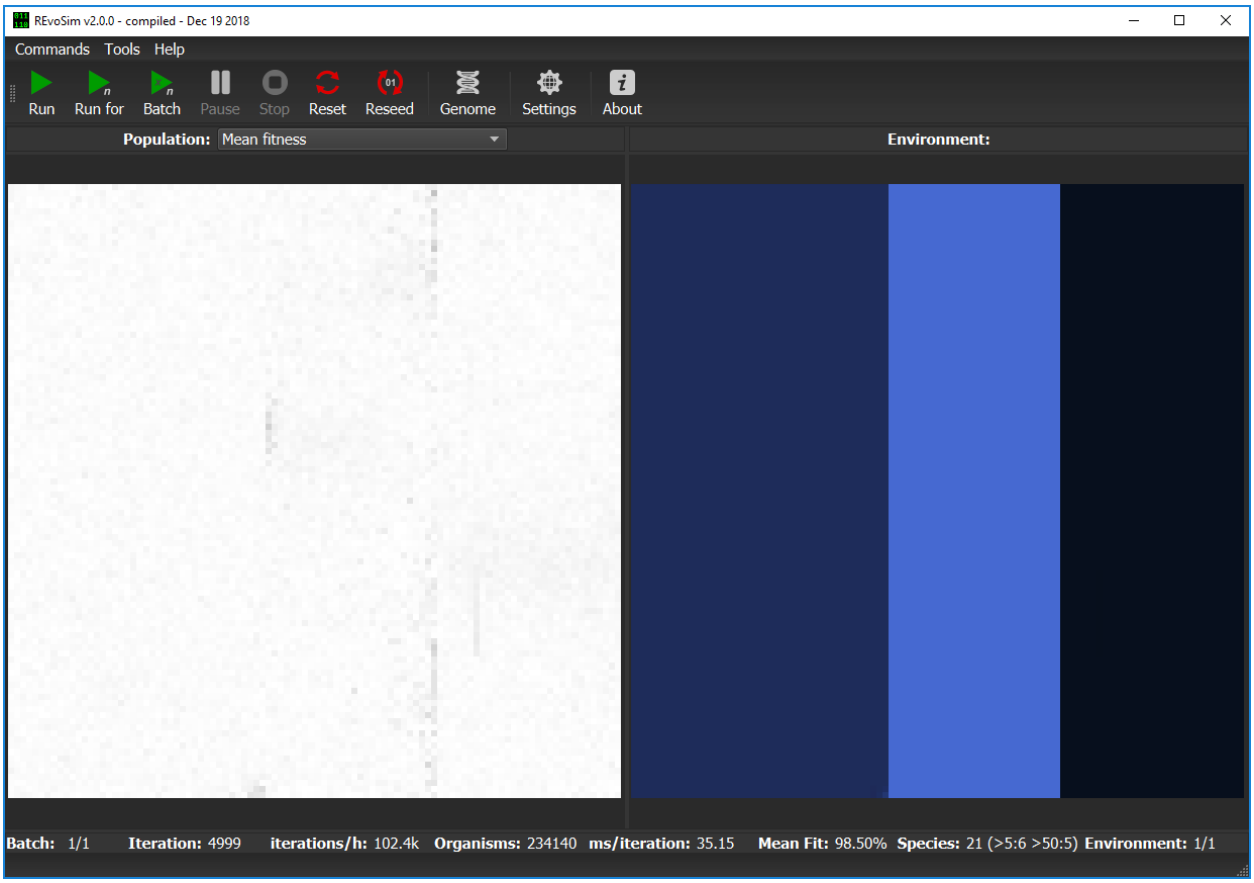
Each grid-square is visualized with a grey-level representing the mean fitness of all creatures alive in the square, scaled so that white is maximum fitness (=‘Settle Tolerance’).

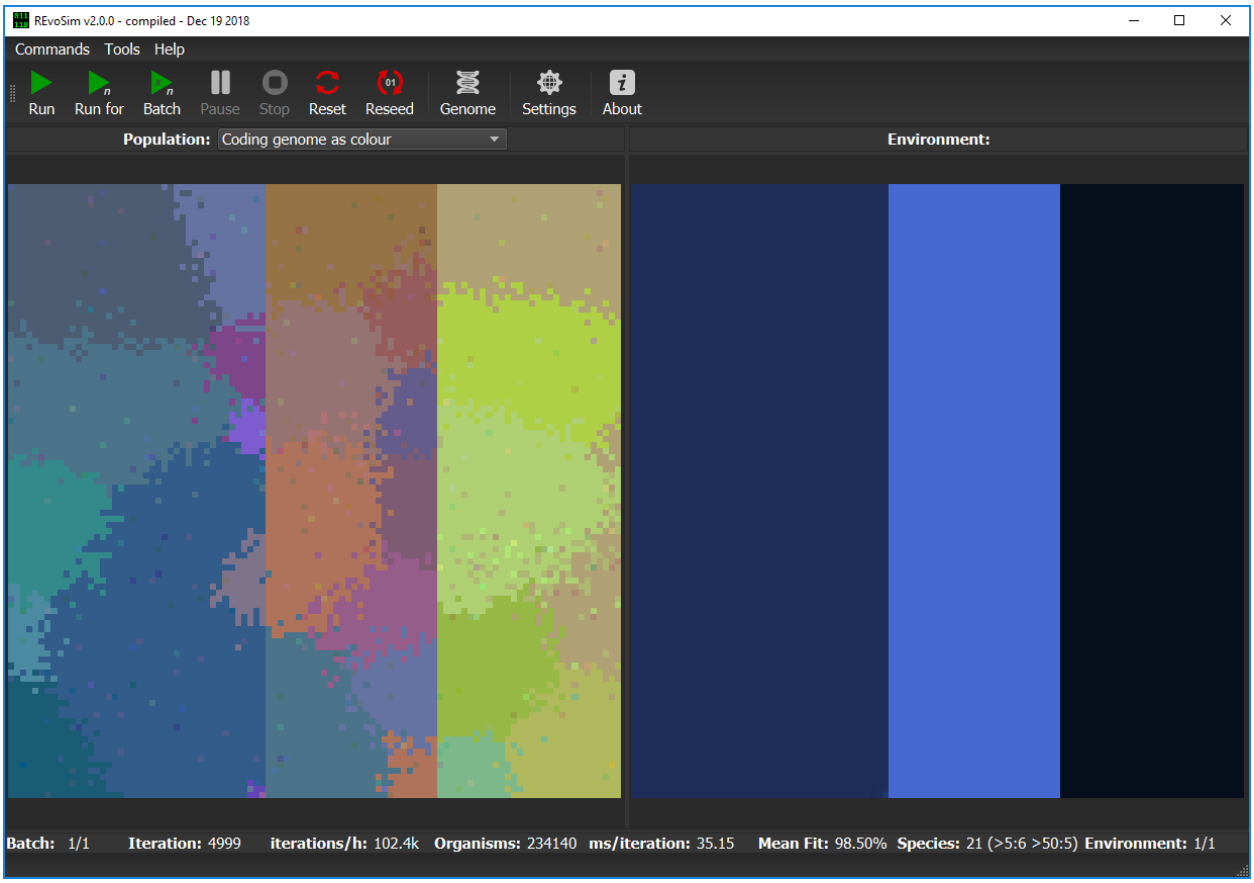
Coding Genome as Colour

For each grid-square, the most commonly occurring (modal) genome is computed. The 32-bit coding part of this genome is then converted to a colour, where the level of red is the (scaled) count of 1’s in the least significant 11 bits, the level of green is the (scaled) count of ‘1’s in the next least significant 11 bits, and the level of blue is the (scaled) count of ‘1’s in the most significant 10 bits. This visualization approach provides a quick means of distinguishing genomes, ensuring that small genomic changes result in small changes in colour. It does not, however, guarantee that the same colour will in all cases represent the same genome (as many very different genomes may possess the same bit-counts in each of the three 11/10 bit segments).

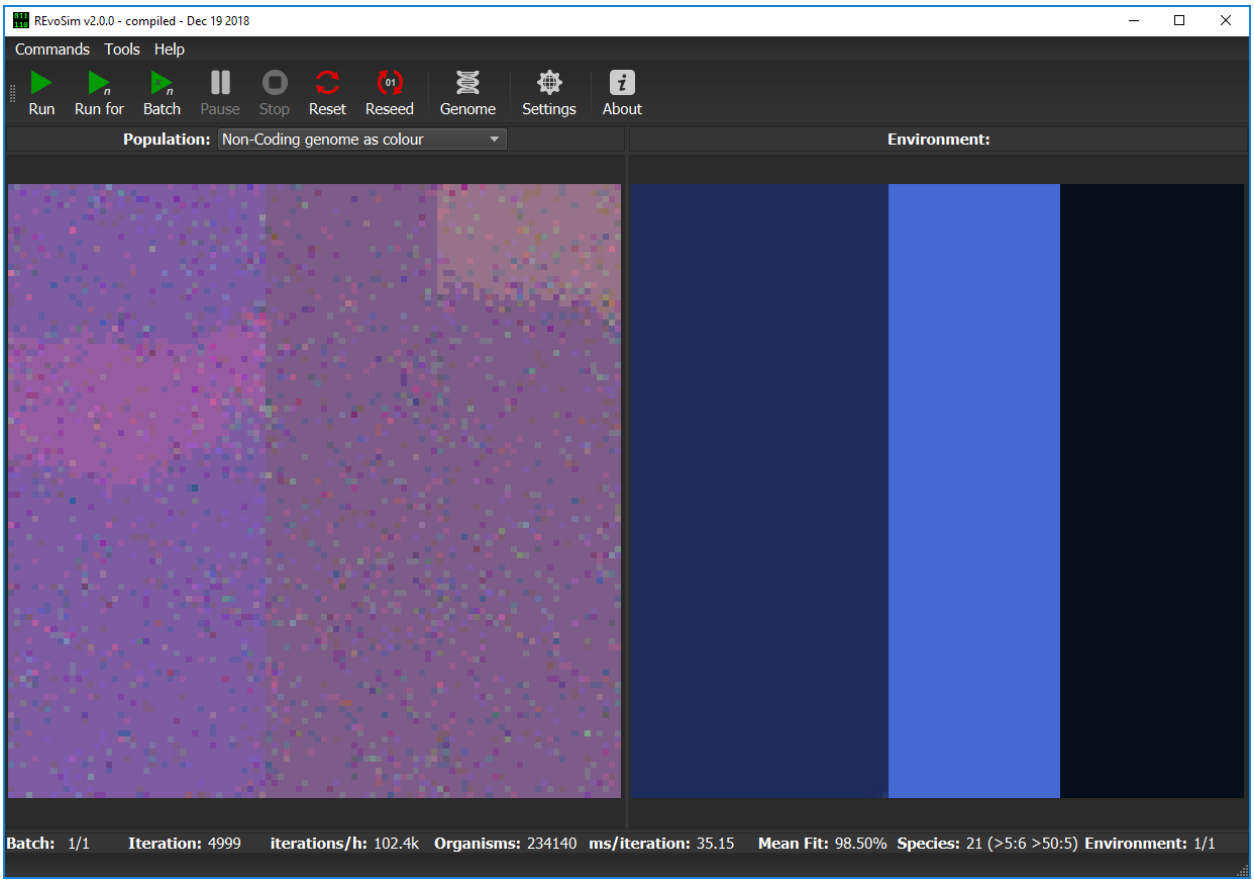
Non-coding Genome as Colour

For each grid-square, the most commonly occurring (modal) genome is computed. The 32-bit non-coding part of this genome is then converted to a colour, where the level of red is the (scaled) count of 1’s in the least significant 11 bits,


REvoSim v2.0.0 - compiled - Dec 19 2018
Commands Tools Help
Run
Run for
Batch
Pause
Stop
Reset
Reseed
Genome
Settings
About
Population: Mean fitness
Environment:
Batch: 1/1 Iteration: 4999 iterations/h: 102.4k Organisms: 234140 ms/iteration: 35.15 Mean Fit: 98.50% Species: 21 (>5:6 >50:5) Environment: 1/1


REvoSim v2.0.0 - compiled - Dec 19 2018
Commands Tools Help
Run
Run for
Batch
Pause
Stop
Reset
Reseed
Genome
Settings
About
Population: Coding genome as colour
Environment:
Batch: 1/1 Iteration: 4999 iterations/h: 102.4k Organisms: 234140 ms/iteration: 35.15 Mean Fit: 98.50% Species: 21 (>5:6 >50:5) Environment: 1/1

the level of green is the (scaled) count of '1's in the next least significant 11 bits, and the level of blue is the (scaled) count of '1's in the most significant 10 bits. This visualization approach provides a quick means of distinguishing genomes, ensuring that small genomic changes result in small changes in colour. It does not, however, guarantee that the same colour will in all cases represent the same genome (as many very different genomes may possess the same bit-counts in each of the three 11/10 bit segments).



Gene Frequencies

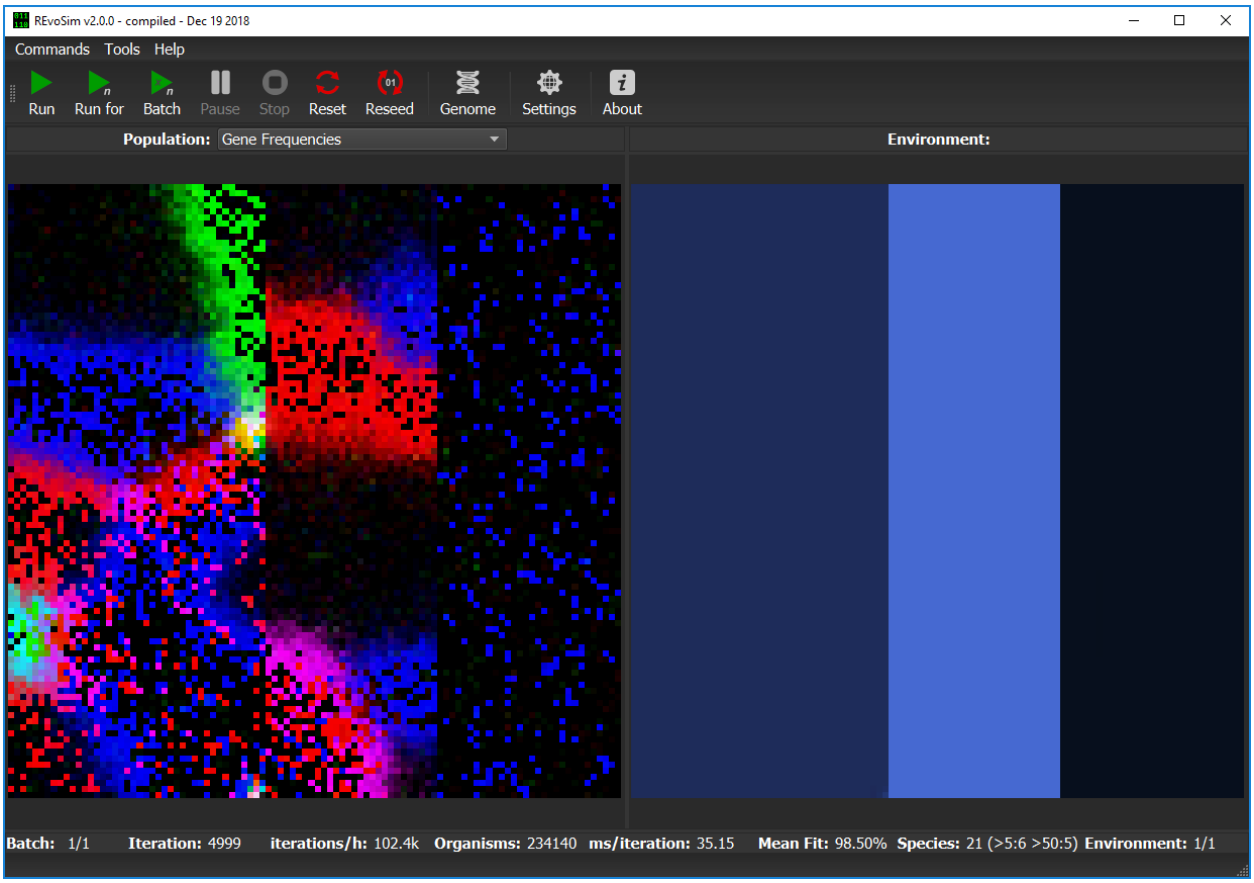
For each grid-square, the mean value of the first three coding bits is computed, and a composite colour created using the mean of the least significant bit as red, the mean of the next least significant bit as green, and the mean of the third least significant bit as blue.

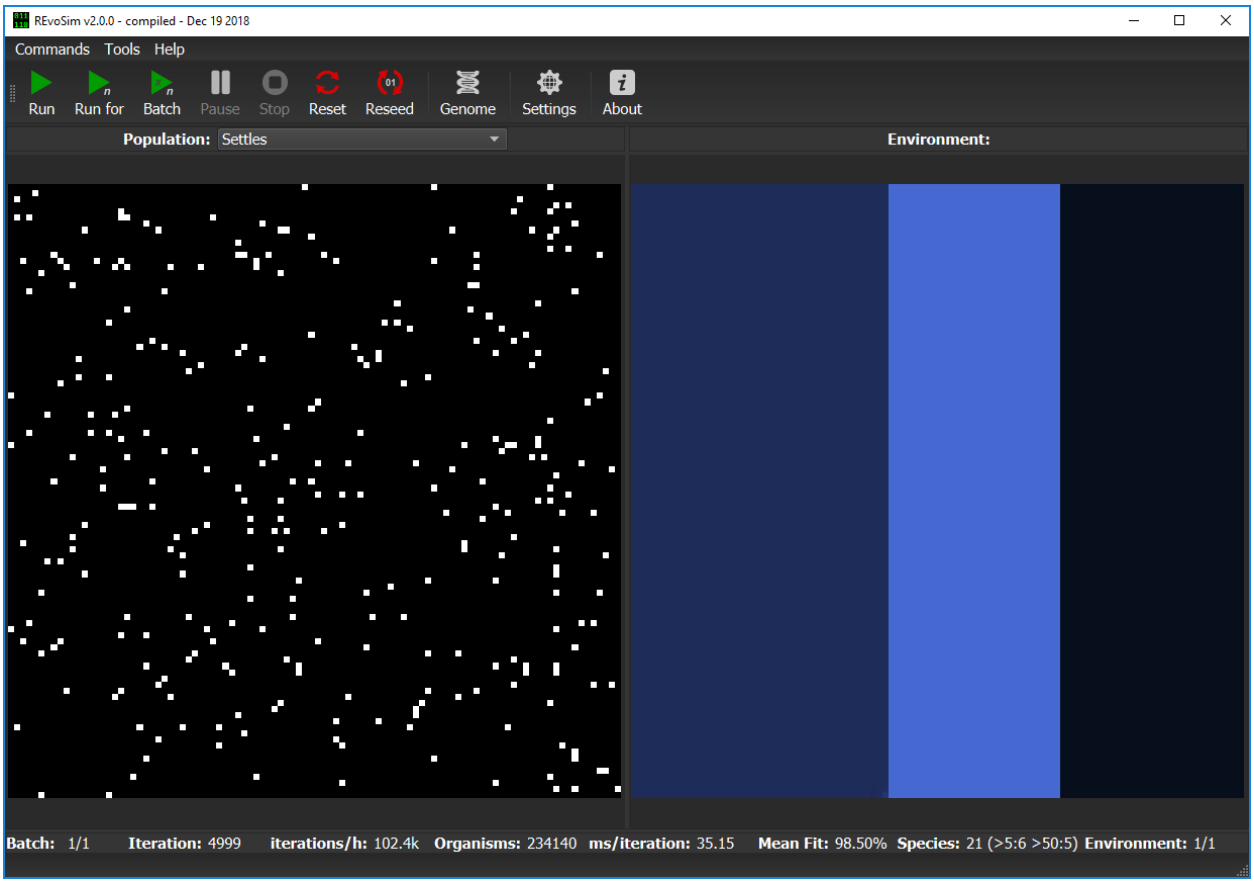
Settles

Each grid-square is visualized with a grey-level representing the number of successful 'settling' events in that square since the last visualization.

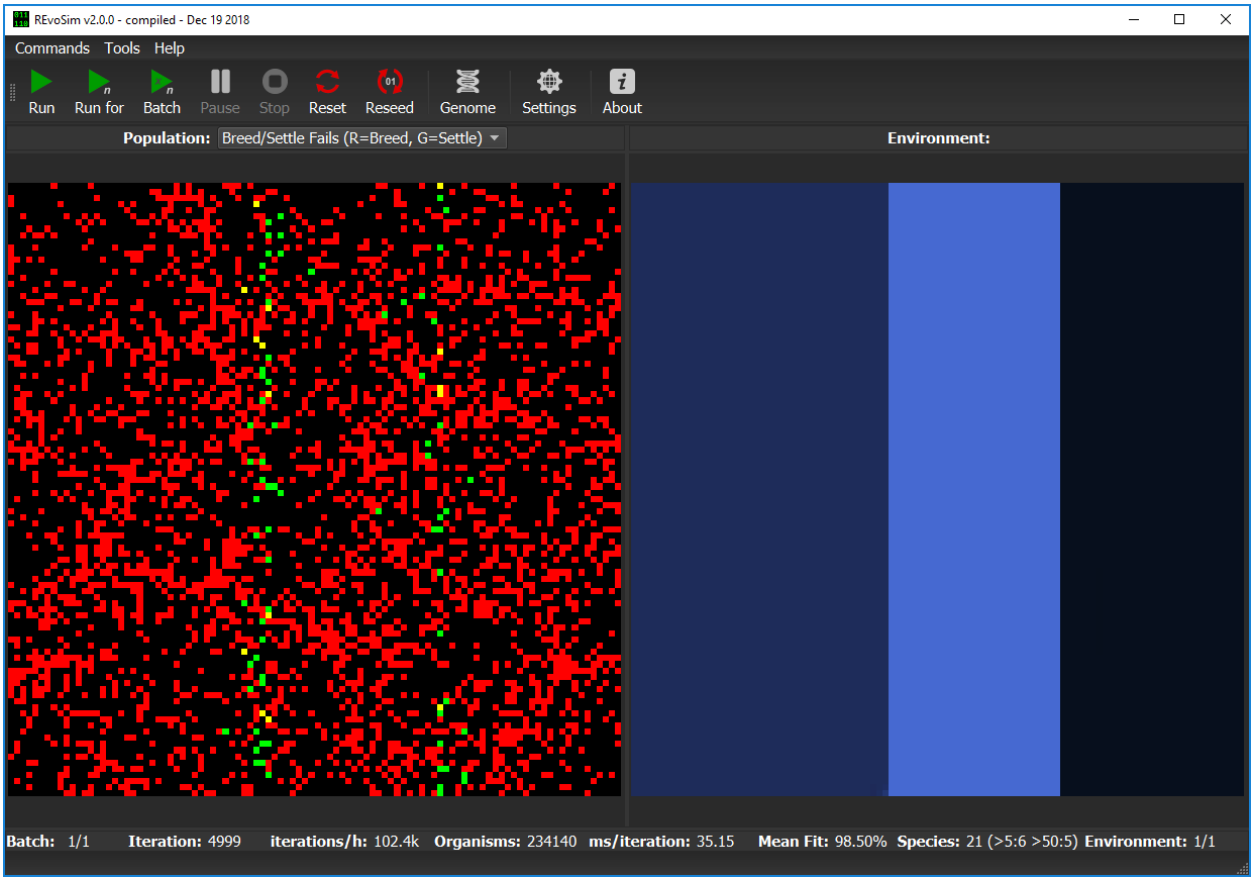
Breed/Settle Fails (R=Breed; G=Settle)

Each grid-square is assigned a colour representing the number of 'fails' since the last visualization. The number of 'breed fails' (attempts to breed that were aborted due to compatibility) provides the red level, and the number of 'settle fails' (attempts to settle that resulted in a fitness of 0 and hence the death of the settling creature) provides the green level. Fail visualizations are scaled non-linearly (using v = 100*f*0.8, where f = mean fails per iteration, and v is


REvoSim v2.0.0 - compiled - Dec 19 2018
Commands Tools Help
Run
Run for
Batch
Pause
Stop
Reset
Reseed
Genome
Settings
About
Population: Gene Frequencies
Environment:
Batch: 1/1 Iteration: 4999 iterations/h: 102.4k Organisms: 234140 ms/iteration: 35.15 Mean Fit: 98.50% Species: 21 (>5:6 >50:5) Environment: 1/1


REvoSim v2.0.0 - compiled - Dec 19 2018
Commands Tools Help
Run
Run for
Batch
Pause
Stop
Reset
Reseed
Genome
Settings
About
Population: Settles
Environment:
Batch: 1/1 Iteration: 4999 iterations/h: 102.4k Organisms: 234140 ms/iteration: 35.15 Mean Fit: 98.50% Species: 21 (>5:6 >50:5) Environment: 1/1

visualized intensity on a scale 0-255). Fail visualization maps the edge of species or subspecies ranges, as it highlights cells where gene-flow is restricted by any mechanism.



Species

REvoSim simulations often produce discrete species, i.e. reproductively isolated gene-pools, and for many macroevolutionary studies the identification and tracking of the fate of species is a requirement. This visualisation option assigns a unique colour to each species, and colours each grid-square with the species in the lowest occupied slot of that square.

2.3.4 Environment Scene

Every grid-square of the Population Scene possess an 'environment', which consists of three integer parameters in the range 0-255. These are visualized as colour, the three parameters representing red, green and blue values (further details of how this is achieved are available in the REvoSim paper). The environment for the entire grid can thus conveniently be visualised as a raster (bitmapped) 24-bit colour image in the Environmental Scene. Environments can be static (specified by a single raster image), or dynamic (specified by a sequence of raster images).

The default environmental scene consists of a static image with three colour zones. This image can be changed from the Simulation Settings pannel.

Dynamic environments can be enabled via the Simulation Settings pannel. There are currently three modes of transformation between image sequences, these are:

1. Once - each image in the sequence is shown in order but only once

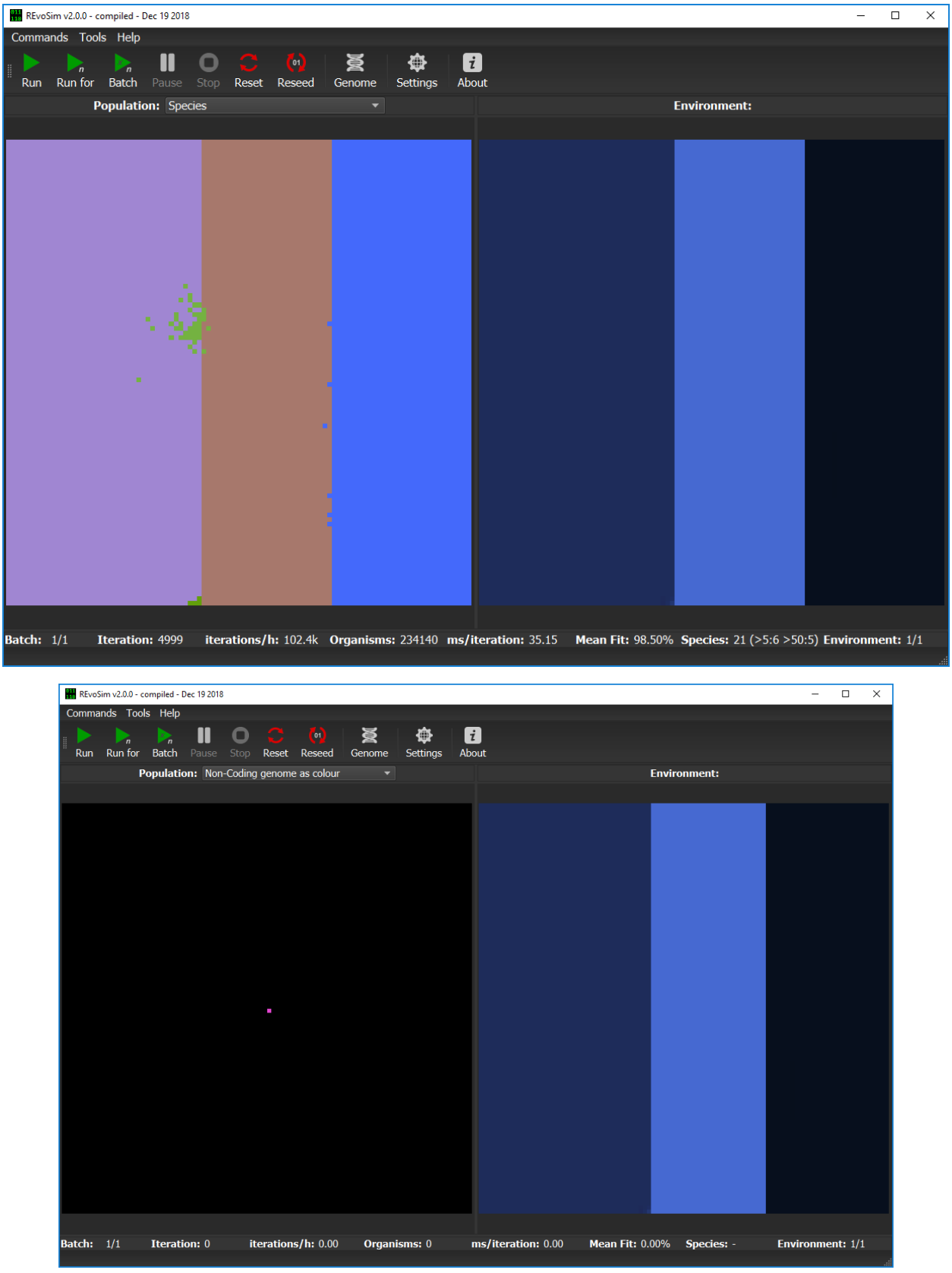


Fig. 5: Figure 3.4.1 - Default Environmental Scene showing three blocks of colour in vertical strips.

2. Loop - each image in the sequence is shown in order, after displaying the last image in the sequence the environment loops back to the first image (etc.)
3. Bounce - each image is shown in order, after displaying the last image the order is reversed, until the first image is once again shown, the order is then reversed (etc.)

For dynamic environments, the number of simulation-iterations between image updates is configurable but by default 50 (i.e. the environment image advances to the next in sequence every 50 iterations). REvoSim can optionally interpolate between environmental images to provide smooth transitions.

2.3.5 Information Bar

At the bottom of the GUI - below the population and environment views - is an information bar, which provides an overview of a number of elements of the current simulation. This is updated every polling generation. The statistics it provides are:

Batch When RevoSIm is running in batch mode this shows how many runs are completed out of the total number requested.

Iteration The number of iterations that had been completed since the start of the simulation at the last polling interval.

Iterations per hour This provides an indication of the speed at which RevoSim is running, and thus it is realtively easy to calculate how long any given run will take.

Organisms This is a count of the total number of digital organisms alive at the last polling iteration.

Milliseconds per iteration This is an alternative measure of speed.

Mean fitness This is the mean fitness of all living organisms in the simulation at the last polling iteration.

Species If species tracking is on, this will provide the number of species at last poll once a speciation event has occurred.

Environment This is the index of the current environmental image, along with the total number that have been loaded. By default RevoSim loads with a single environmental image.

2.4 Configuring your Organisms

At the core of the REvoSim simulation are 64-bit digital organisms. A number of properties of these organisms can be defined in the suimulation, in the Organism tab of the Settings dock. These settings are as follows:

Chance of mutation This dictates the chance of a mutation when an organism breeds (if it is set to n, there is a n/256 chance of mutation). The default is 10, which thus equates to a 10/256 chance that a randomly selected bit in the genome is flipped (from 0 to 1 or 1 to 0).

Start age Every iteration each organism loses one from its age counter. This setting dictates the value at which this counter is set when an organism is born. As such, this dictates generation times within the software, and will also impact on the number of times any individual can breed in its lifetime.

Breed threshold Within a pixel, the energy provided each iteration is split between the digital organisms living within the pixel based on their fitness. A full description of how this is achieved can be found in the REvoSim paper. When a digital organism has stored enough energy to pay the breed cost and still exceed the breed threshold, it attempts to breed.

Breed cost This is the amount of resource that is removed from an organisms when it successfully breeds.

Maximum difference to breed In sexual breeding mode, if two organisms attempting to breed have a hamming distance greater than this value when genomes are compared, breeding fails.

Use maximum difference to breed Depending on the nature of a study, maximum difference to breed may not be desired. This tickbox dictates whether it is enforced. If unticked, in sexual modes, breeding failure does not occur on the basis of genomic distance.

Breed only within species When this checkbox is ticked, during sexual selections, digital organisms can only breed with other members of the same species. This only occurs if species tracking is turned on.

Breed mode In sexual mode, organisms can breed with other individuals (or themselves) given the caveats outlined above. In asexual mode, self-breeding is enforced, and organisms are cloned when they have the required energy reserves to allow breeding.

Dispersal This figure dictates the extent to which juveniles disperse on settling. Small numbers equate to significant dispersal, larger numbers increase the likelihood that juveniles setting in the same pixel as their parent. How this is achieved is described in full in the REvoSim paper.

Nonspatial settling For some evolutionary phenomena, the impact of space/dispersal may have an unknown impact and not be the element of interest within a simulation, and thus be undesirable. This tickbox allows juveniles to be randomly placed within the simulation (note that with a non-uniform environment, space will still have some impact on the simulation).

2.5 Setting up the Simulation

REvoSim provides the user with control of many elements of the simulation. These are introduced herein, and can be modified within the the Simulation tab of the Settings dock.

Change environmental files REvoSim launches with a default environment comprising three stripes in different shades of blue. This can be changed by clicking on this button, which will launch a file explorer. Using this allows a single, or multiple environment images (any standard image format) to be loaded. Environment image files are currently limited to 100 x 100 pixels in size.

Environment refresh rate This dictates how many iterations there are between updates of the environmental images if an image stack has been loaded, and thus dictates the environmental rate of change (small numbers are faster).

Environment mode This dictates, if multiple files are loaded, the mode with which these are updated. Static uses just the first of the chosen images as the environment throughout a run. Once will run from the start of an image stack to the end, leaving the last image as the environment once this has been loaded. Loop returns to the first environment image once the last has been reached (which may result in a significant environmental change in the environment if the first and last image in a stack are different). Bounce will move from the start to end of an environmental image stack and then back again, repeating this for the duration of a run.

Interpolate between images This provide linear interpolation of the environment between refresh iterations.

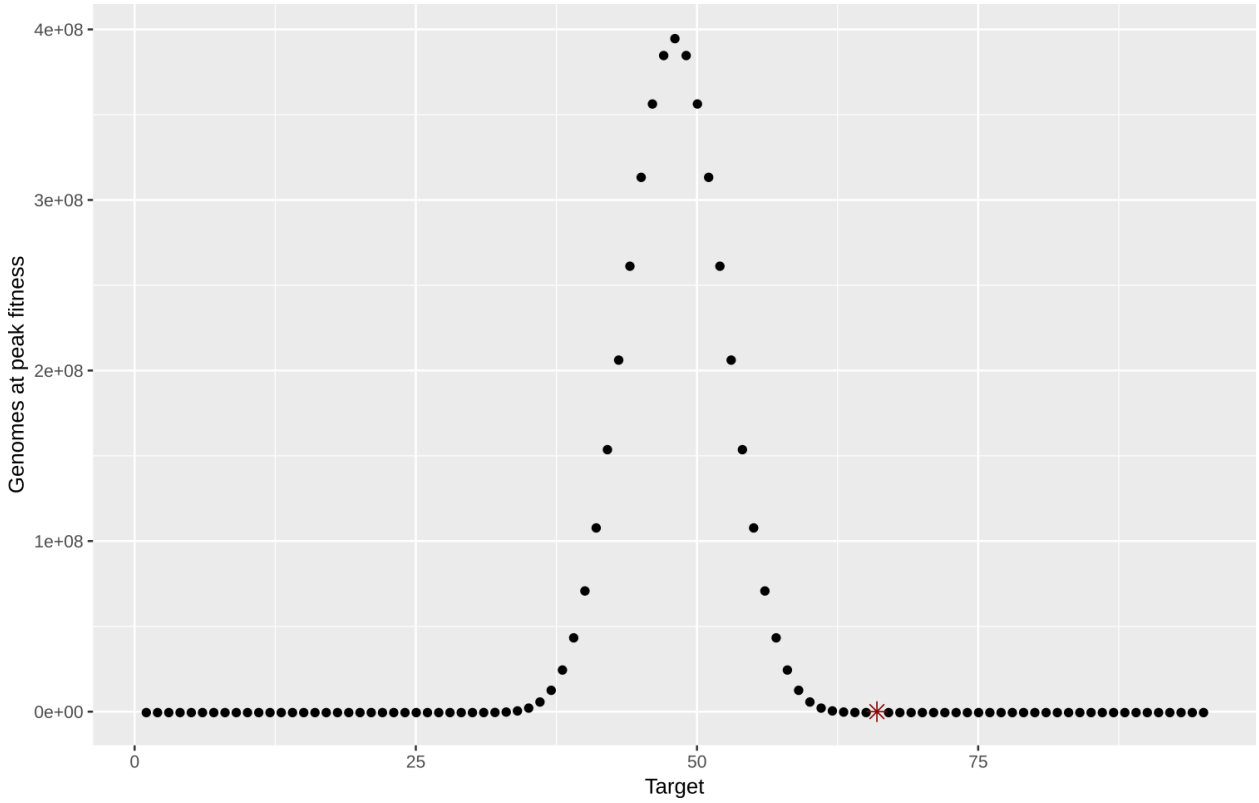
Toroidal environment By default, when a juvenile settles outside the limits of an environment, it dies. Such boundary effects can be avoided by selecting toroidal environment, which wraps around in both directions (e.g. juveniles settling off the left enter the simulation on the right; EnviroGen can simulate toroidal environments).

Grid X This dictates the size of the simulation grid in the X direction. If this is smaller than the environmental image it selects the left side of the image.

Grid Y This dictates the size of the simulation grid in the Y direction. If that is smaller than the environmental image it selects the top of the image.

Slots Every pixel in the simulation grid can have multiple digital organisms - each lives within a slot (this is equivalent to a Z axis). This dictates the number of slots (how these behave are treated during reproduction is outlined in the REvoSim paper).

Fitness target This is the target value used by the fitness algorithm as described in the REvoSim paper. In brief, it is the value that the hamming distance between the gemome and the environmental masks (numbers dictated by the colour) is optomised towards. Changing it modifies the fitness landscape: a value of 48 has the greatest number of organisms with peak fitness, and either side of this value, fewer organisms are capable of peak fitness this is shown in the graph below. In this, black points represent the number of genomes with peak fitness for all fitness targets, the red star is the default value 66; in this particular set of runs, there there are ~3000 genomes capable of peak fitness with the default target, but this can vary given differing masks. Note that there are some values for which no organisms will be capable of survival: if this occurs, REvoSim will provide a warning. Also note that the graph below can be created and distribution of fitnesses quantified using the count peaks option in the commands menu of REvoSim.



Energy input This is the amount of energy provided per pixel of the environment grid, which is then split between the digital organisms living in that pixel on the basis of their fitness. This is then used as part of the breeding system in REvoSIm.

Settle tolerance This it controls the distance from the fitness target at which organisms are no longer viable (i.e. have zero fitness and die). It also provides the maximum attainable fitness within the simulation.

Recalculate fitness For efficiency REvoSim only calculates organism fitness for any individual on initial settling: subsequent changes in the environment will not modify an individual's fitness. For some settings (e.g. rapidly changing enviironments or long-lived organisms) this may not be desirable. When checked, this option recalculates the fitness for every organiusm in the simulation every iteration, overcoming this limitation but also significantly slowing the simulation.

Phylogeny settings These radio buttons dictate the mode which by REvoSim tracks phylogeny. Off does not track phylogenies and is thus the fastest mode (this could be useful for - as an example - studies focussing on changes in fitness). Basic phylogeny identifies species in time slices to allow species to be coloured in the population view, and species diversity to be recorded. The option phylogeny identifies species, and then records their phylogeny, allowing a tree to be created at the end of a run. Phylogeny and metrics does this, and also records a number of other metrics for each species, also output (when requested) at the end of a run. Note that moving between off and any form of tracking has a significant performance cost: there is little computational overhead moving between the different tracking options. Moving from basic to phylogeny to metrics does, however, come with an increasing memory overhead, as the trees and metrics are by necessity stored in RAM during a run, and written when the run completes. This could have implications for runs with a significant number of organisms run for extended periods. See *Logging the Simulation* and *Configuring your Outputs* for more details REvoSim outputs.

2.6 Configuring your Outputs

REvoSim has multiple output options that allow runs to be recorded and analysed as required for any given study. These are controlled from the Output tab of the settings dock. Options are as follows:

Output save path This is the folder into which all outputs from REvoSim (logs, images, any other files) are saved. For consistency these are all placed within a newly created folder called *REvoSim_output*. Text files are placed within the root of this folder, and images are placed within their own folder within *REvoSim_output*.

Refresh/polling rate This is the frequency (in iterations) that the simulation is polled. Polling includes running the full clustering analysis intergral to the thorough species-identification algorithm (see RevoSim paper), writing the in-simulation log, writing any requested image files to disk, and updating the GUI (nless dsisabled). Because the species identification system has a significant computational overhad, if species tracking is on, frequent polling will significantly slow the simulation.

Logging Population/Environment Any of the simulation visualisations during a run can be saved as an image (.png stacks labelled by iteration number). The tick boxes in this part of the tab dictate which images are saved.

Write Log Files When this option is checked a log file is written during the course of every run. See *Logging the Simulation* for more details of REvoSim logs.

Automatically create detailed log on batch runs This option outputs the detailed log at the end of each run when REvoSim is operating in batch mode. See *Logging the Simulation* for more details of REvoSim logs.

Write data for current run This option outputs the detailed log for the currently running simulation at the point at which the button is pressed. See *Logging the Simulation* for more details of REvoSim logs.

Exclude species without descendents Under most settings a significant number of small, short-lived species may appear regularly within a REvoSim run. Given the significant amount of data REvoSim can generate, and the fact that these short lived species will be unimportant for many studies (potentially masking important obvservations), this option rationalises REvoSim detailed logs by only including species with descendents in the end run log and tree.

Minimum species size It is also possible to filter the species data in the log files so that only species above a certain number of individuals are included in the logs. This spin box dictates what that minimum cut-off is.

Don't update GUI This option allows runs to proceed without updating the GUI (although this prevents images being saved through a run). This allow REvoSim to run marginally faster, and may be of

utility for very long runs.

2.7 Logging the Simulation

RevoSim provides two primary logging options (as well as the opportunity to save images showing the progress of a run: see *Configuring your Outputs*).

2.7.1 Running log

When logging is enabled, REvoSim writes a file called REvoSim_log.txt to the output folder, and updates this by appending more data every polling iteration, giving an overview of the current run. This is structured as follows:

Timestamp The first line is a time stamp highlighting when the run was written, in the following format: 2018-12-30T11:57:51

Settings A printout of all REvoSim settings for this run then follows, divided into integers and then bools. This means that at any point it is possible to revisit and check all settings for that run.

Legend There is then an explanation of the structure of the log files. Every iteration, the log records data about the simulation to file in a format designed to be easy to parse into a rtange of analytical environments (e.g. R, Python). This structure is as follows for each iteration:

```
- [I] Iteration Number
- [P] Population Grid Data:
  - Number of living digital organisms
  - Mean fitness of living digital organisms
  - Number of entries on the breed list
  - Number of failed breed attempts
  - Number of species
- [S] Species Data:
  - Species ID
  - Species origin (iterations)
  - Species parent
  - Species current size (number of individuals)
  - Species current genome (for speed this is the genome of a randomly sampled␣
↪individual, not the modal organism)
```

Log data The log then begins. Iterations are separated by new line breaks. Every iteration has a single [I] line, one [P] line, and then an [S] line for every species above the minimum species size. We note that it does not exlude species without descendents because it is written during the log, appending to the file for speed. To filter out those species without descendents would introduce the need to store and then regularly filter the log data, and thus would come with a notable computational overhead.

2.7.2 Detailed log

REvoSim provides an alternative, more detailed log for anaylsis of runs. This can be automatically output at the end of every run in batch mode (see *Configuring your Outputs*), or dumped whenever required using the button in the Output tab of the Settings dock. This too is written to the current output folder, and is placed in a file called REvoSim_end_run_log.txt. This is structured as follows:

Timestamp The first line is a time stamp highlighting when the run was written, in the following format: 2018-12-30T11:57:51

Settings A printout of all REvoSim settings for this run then follows, divided into integers and then bools. This means that at any point it is possible to revisit and check all settings for that run.

Legend There is then an explanation of the contents of this log file: "This log features the tree from a finished run, in Newick format, and then data for all the species that have existed with more individuals than minimum species size. The exact data provided depends on the phylogeny tracking mode selected in the GUI."

Tree There then follows a tree in Newick format for the run to the point at which the log is written, excluding species below minimum species as requested, and also excluding species without descendents if requested. This can then be loaded into, e.g. FigTree to be rendered, or into e.g. R, for analysis. An example tree is shown below - note species labels are prefaced with id for clarity, and also include the maximum size of that species as part of their species name, after a hyphen:

```
((((((((((((((((((((((((((id27-81050:50,id28-2:50,id29-1:50,id30-1:50,id31-1:50,id32-
↪5:50,id33-2:50,id34-8:50,id35-1:50)id26-81050:50,id36-3:50,id37-2:50,id38-4:50,id39-
↪3:50,id40-2:50,id41-3:100,id42-4:100,id43-5:100)id25-81050:50,id44-17:150,id45-
↪4:100,(id47-23311:100,id48-2:50)id46-23311:50,id49-2:50,id50-2:100)id24-81050:50,
↪id51-4:50,id52-1:50,id53-5:100,id54-2:50,id55-10:100,id56-11:50,id57-61:200,id58-
↪49:200)id23-81050:50,id59-2:100,id60-2:50,id61-1:50,id62-4:50)id22-81050:50,id63-
↪13:250,id64-2:50,id65-8:50,id66-1:50,id67-1:50,id68-4:50,((id71-24648:50,id72-
↪1:50)id70-24648:50,id73-2:50,id74-3:100)id69-24648:200)id21-81050:50,id75-3:50,id76-
↪14:150,id77-3:50)id20-81050:50,id78-2:150,id79-8:50)id19-81050:50,id80-2:50)id18-
↪81050:50,id81-3:50,id82-2:50,id83-1:50)id17-81050:50,id84-1:50,id85-7:50,id86-2:50,
↪id87-9:150)id16-81050:51,id88-1:1)id15-81050:18,id89-18:69)id14-81050:16,id90-
↪3:35)id13-81050:13,id91-1:6)id12-81050:8,id92-2:42)id11-81050:3,id93-2:16)id10-
↪81050:14,id94-1:13)id9-81050:8,id95-2:11)id8-81050:1,id96-1:9)id7-81050:14,id97-
↪1:12)id6-81050:22,id98-1:12)id5-81050:87,id99-2:73)id4-81050:2,id100-1:12)id3-
↪81050:4,id101-2:141)id2-81050:64,id102-1:10)id1-81050:11,id103-3:82)id0-81050:237
```

Log - detailed species data The rest of the log file comprises detailed data for each species, in the same order they appear in the tree. For each species, for every polling iteration, REvoSim provides the following data:

- Species ID
- Species ID of Parent
- Iteration number (i.e. the polling iteration for which this was recorded)
- Number of individuals at polling iteration (size)
- A sample genome for the species, selected as for running log (i.e. randomly), presented as a 32-bit number
- The above genome as a binary string
- Genomic diversity - i.e. the number of different genomes in the species
- The number of pixels occupied by this species, subtracted 1 (i.e. real range is 1-65536, but -1 allows C++ style numbering: 0-65535)
- The geographic range in the form of the maximum distance between outliers
- The centroid of range in X - the mean of all X positions
- The centroid of range in Y - the mean of all Y positions
- Mean fitness of all members of species, stored multiplied by 1000 to allow small changes to be easily identified
- Minimum R, G, then B - the log then reports the minimum R, G and B values the species is found in
- Maximum R, G, then B - as above, buyt maximum values
- Mean R, G, then B - the final three numbers are the mean R, G, and B values the species inhabits

An example log thus appears:

```
id,ParentID,iteration,size,sampleGenome,sampleGenome_binary,diversity,cellsOccupied,
↪geog_range,centroid_x,centroid_y,mean_fit,min_env_red,min_env_green,min_env_blue,
↪max_env_red,max_env_green,max_env_blue,mean_env_red,mean_env_green,mean_env_blue
27,26,1073,34539,17476623570733825285,
↪1111001010001001011100001000101011011110110011101111110100000101,6780,4199,41,20,49,
↪9566,30,41,88,35,54,112,30,44,90
28,26,1073,1,18017055526017752864,
↪1111101000001001011100001000101011011110110001101101111100100000,1,1,0,29,44,10000,
↪30,44,90,30,44,90,30,44,90
29,26,1073,1,17441298461089501447,
↪1111001000001011111100001000101011011010110011101001000100000111,1,1,0,31,79,10000,
↪30,44,90,30,44,90,30,44,90
30,26,1073,1,17312242184062138796,
↪1111000001000001011100001000101011001110100011100111100110101100,1,1,0,5,6,8000,30,
↪44,90,30,44,90,30,44,90
31,26,1073,1,17726573350043672487,
↪1111011000000001011100001000101011011110010000100101111110100111,1,1,0,41,22,9000,
↪30,44,90,30,44,90,30,44,90
32,26,1073,5,18021559125636701700,
↪1111101000011001011100001000101011011110010001100101111000000100,1,4,2,3,83,9000,30,
↪44,90,30,44,90,30,44,90
33,26,1073,2,17439188498342378892,
↪1111001000000100011100011000101011011110110001100111110110001100,1,2,0,3,9,9000,30,
↪44,90,30,44,90,30,44,90
34,26,1073,3,17440594842165369120,
↪1111001000001001011100001001101011001110110001101101100100100000,1,2,1,36,49,9000,
↪30,44,90,30,44,90,30,44,90
35,26,1073,1,16358041978649091335,
↪1110001100000011011100001000001011001110010011101001110100000111,1,1,0,36,97,9000,
↪30,44,90,30,44,90,30,44,90
26,25,1023,34348,17476623570733825285,
↪1111001010001001011100001000101011011110110011101111110100000101,6582,4201,51,20,49,
↪9357,30,41,88,70,105,209,30,44,90
...
```

The two logs are designed to allow as many potential elements of a RevoSim run to be quantified as possible. Should any further measures or statistics be required, please file a feature request.

2.8 Genome comparison dock

The Genome Comparison Dock allows genomes to be inspected and compared within RevoSim, and can be launched from the main menu or the toolbar. By default it appears landscape on the bottom of the main window. Its landscape orientation is intended to allow whole genomes to be viewed without scrolling on most operating systems, however we note that on operating systems with low resolution displays, the REvoSim GUI may struggle to accommodate the docker in its default position: the environment and population views may be forced to be small, or the genome comparison docker may not have the vertical extent to show more than one genome. If this is the case, the docker can be floated, or alternatively docked to the right or left of the main window.

A genome can be added to the comparison dock by right clicking on the population scene. This will add the genome of the first digital organism living within that pixel to the comparison dock (i.e. that held in slot zero), rendering it as zeros and ones with coding genome on the left, as well as displaying the colour it is rendered as in genome views, and the colour of its environment. Multiple genomes can be selected and inspected in this manner, and a comparison of two can be achieved by selecting the desired genomes and selecting compare, which renders the two in a lower panel highlighting the differences between the genomes. By default, auto compare is enabled: this compares and highlights the differences between each entry on the genome list and that before it. See Video 8.1 for example of use.

The genomes can be deleted individually from the table by selecting a genome and using the Delete button. Alternatively all genomes in the table can be removed from Genome Comparison Dock the using the Reset All button (this will also remove any manually compared genomes).

The first ten genomes from the Genome Comparison Dock are pulled through to the reseed dialogue (Fig. 8.1): this allows one to be selected as the starting genome for a new run. This allows multiple repeats of simulations using the same genome, which is known to be capable of surviving within a given environment using the masks loaded at launch (note that these will not persist between sessions unless through loading a saved simulation).

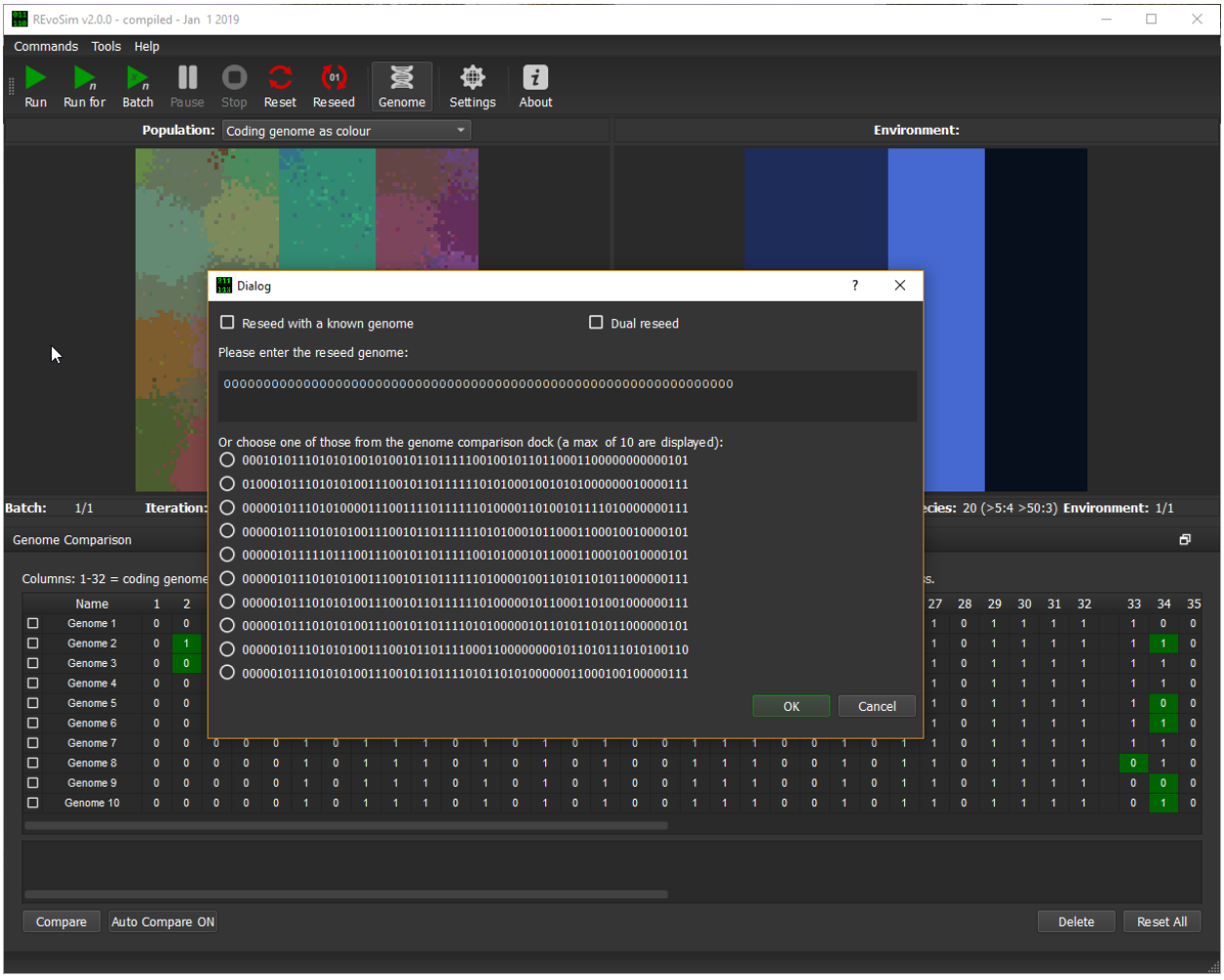


Fig. 6: Figure 8.1 - Reseed dialog showing first ten genomes as selected in the Genome Comparison Dock.

2.9 Advanced Options

2.9.1 Count peaks

A key element of REvoSim is its fitness algorithm, described in depth in the REvoSim paper. The countpeaks command in the REvoSim main menu is included to provide a simple quantification of the fitness landscape given the current masks. These change when the software is restarted, and thus in any given run there will be shifts in the fitness lanscape.

To provide quantification, the count peaks command cycles through every possible 32-bit number, calculating its fitness for the current masks, selected fitness target, and user-defined RGB values (these are requested from the user on initiation of the count peaks command). When this is complete, the software outputs these in a text file written to the output folder (as defined in the Output tab of the Settings dock) that lists fitness *vs* genome count. An example is shown below.

```
REvoSim Peak Counting 2018-12-17T16:56:57
==================

Below is a histogram showing the different fitnesses for all potential 32-bit␣
↪organisms in REvoSim under the user-defined RGB levels.

==================

Fitness counts for red=128, green=128, blue=128

0,3400873943
1,261711486
2,206660322
3,154146136
4,108257580
5,71295660
6,43807340
7,24949080
8,13052205
9,6196135
10,2627235
11,976050
12,310646
13,82512
14,18036
15,2930
```

The above result is for a typical run, which provides a distribution within those organisms capable of surviving as follows, for red=128, green=128, blue=128:

2.9.2 Custom Random Numbers

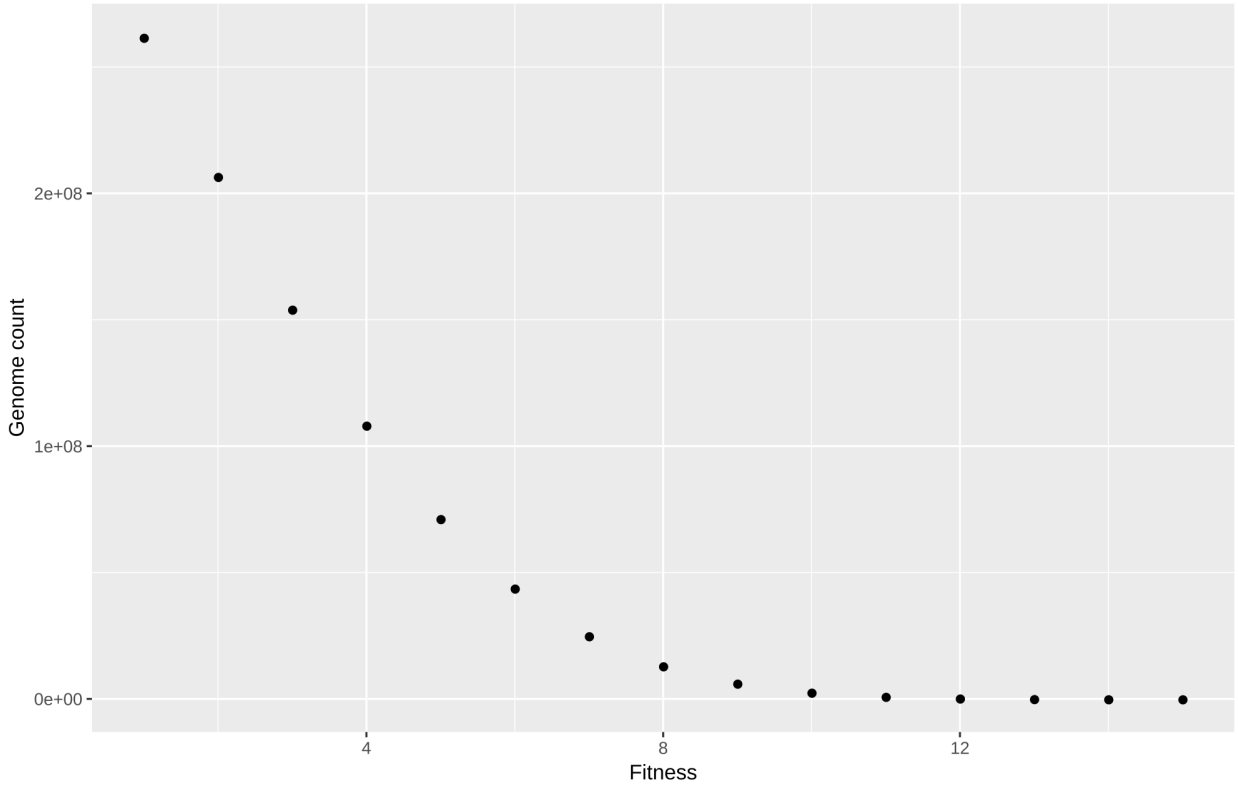
REvoSim employs a pre-generated table of 65,536 random numbers 0–255 which it uses by default during the simulation, both for speed and to avoid potential biases from pseudo-random number generators. 10Mb of quantum-generated random numbers from randomnumbers.info are packaged into the executable and used to generate this table on load (i.e. different runs will have different random numbers, based on the quantum-generated random numbers); these can be replaced with any other random number file preferred by the user.

To load a custom file of random numbers use the 'Commands > Load Random Numbers…' command from the main menu to open a file selection dialog.

The random number file should be encoded as a random binary string, and should be a minimum of 65536 bytes. Once the desired file is selected press the 'Open' button to import the new random numbers. REvoSim will then ask for a byte offset to read the file from (thus allowing runs to be repeated with the same random numbers, if desired).

Note that REVOSIM will always read 65536 bytes; and will throw an error message if it cannot.

On success a pop-up message reading "New random numbers read successfully" will appear.



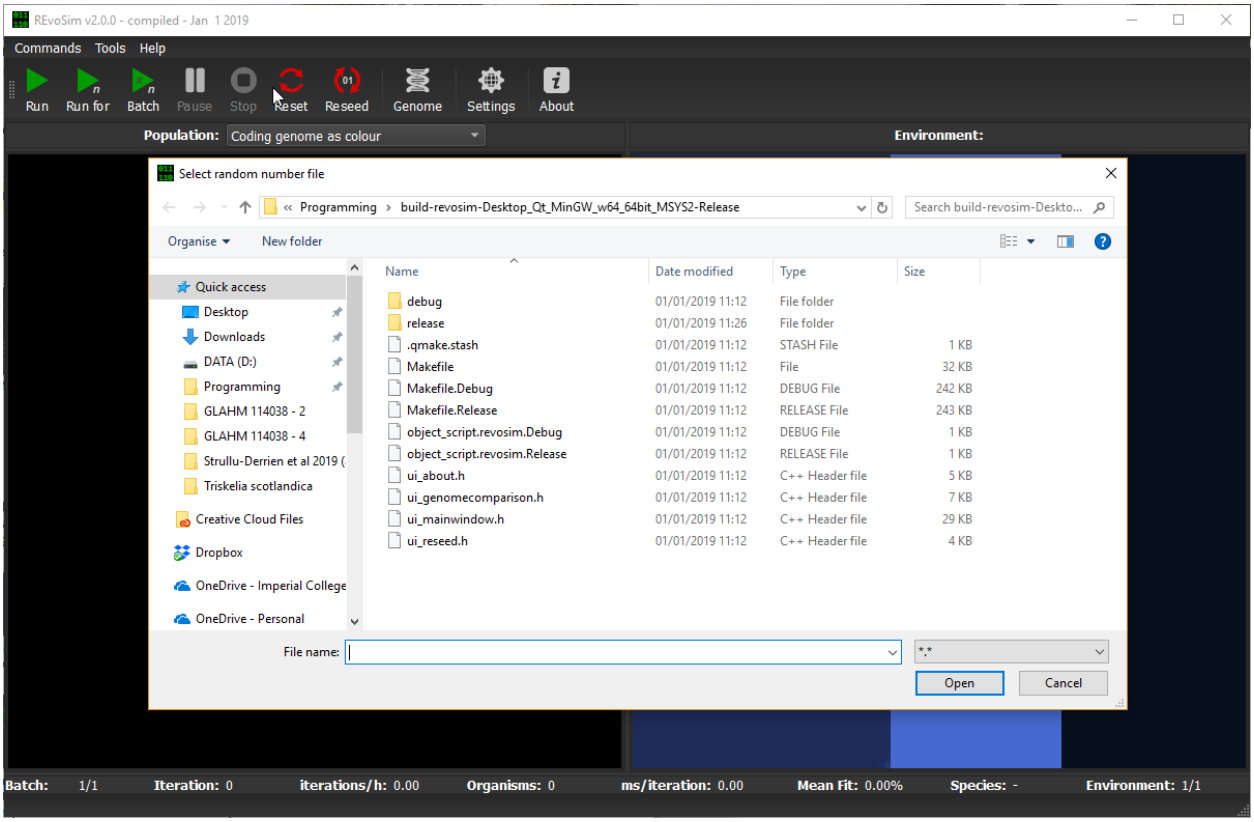


Fig. 7: Figure 9.2.1 - Custom Random Number file open dialog.

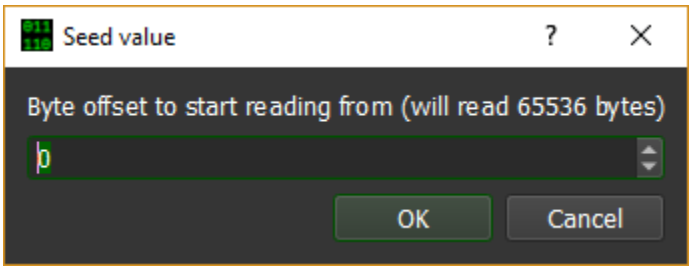


Fig. 8: Figure 9.2.3 - Custom Random Number byte offset form.

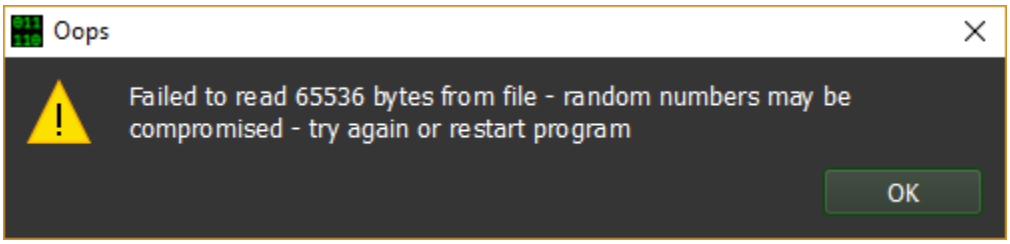


Fig. 9: Figure 9.2.3 - Custom Random Number error on load message.