TREvoSim Documentation

Release 2.0.0

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

Jul 25, 2022

Contents

1 Relevant references **3**

2 Table of Contents **5**

2.1 Compiling, Installation, and Requirements . . . 5
2.1.1 Compiling from Source . . . 5
2.1.2 Installation . . . 6
2.1.3 Requirements . . . 7
2.2 Window Layout . . . 7
2.2.1 Toolbar . . . 7
2.2.2 Species list . . . 8
2.2.3 Newick string . . . 8
2.2.4 Status bar . . . 8
2.3 Settings dialogue . . . 8
2.3.1 Organisms and simulation tab . . . 8
2.3.2 Environment . . . 10
2.4 Logging the Simulation . . . 11
2.4.1 Logging system . . . 11
2.4.2 Keywords . . . 11
2.5 Menu commands . . . 12
2.5.1 Save Current Settings . . . 12
2.5.2 Save settings as… . . . 13
2.5.3 Load settings from file… . . . 13
2.5.4 Restore default settings . . . 13
2.5.5 Fitness histogram . . . 13
2.5.6 Run tests . . . 13
2.5.7 Set uninformative factor . . . 14
2.5.8 Random Seed . . . 14

The [Tr]ee [Evo]lutionary [Sim]ulator program.

TREvoSim is an individual-based evolutionary model, focussing on the simulation of evolutionary trees and associated character data through a simplified first-principles approach. It shares a number of elements in common with the model REvoSim (see Garwood et al. 2019, relevant references), but differs in species concept, and eschews both space and sexual reproduction.

TREvoSim has been released with the intention that it can be used as a multipurpose platform for the study of many evolutionary phenomena, but in particular, acts as one of the few methods for simulating character data and tree topology at the same time, and one of a highly limited number of individual-based simulations capable of doing so. We note that, as with REvoSim, this package 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.

A full description of the algorithm can be found in Keating *et al.* (2020), referenced below. In brief, a simulation employs digital organisms comprising binary strings. These are used to calculate fitness relative to the environment, and ultimately provide the character matrices output by the simulation. Organisms exist in a playing field and compete with each other; one is selected, the probability of which being based on a fitness algorithm, for reproduction each iteration. When this occurs, the organism is mutated, and returned to the playing field, overwriting the least fit organism. A new species is defined based on hamming distance to the parent species, or last species to originate within the lineage. Species are kept in a list, and their genomes are recorded on extinction. The simulation runs until the requested number of species have existed, and then writes data in the requested format.

t: @palaeoware

e: palaeoware@gmail.com

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 62(3),339-355. DOI: 10.1111/pala.12420

Keating, J.N., Sansom, R.S., Sutton, M.D., Knight, C.G. and Garwood, R.J., 2020. Morphological phylogenetics evaluated using novel evolutionary simulations. Systematic Biology 69(5):897-912. DOI: 10.1093/sysbio/syaa012

Koch, N.M., Garwood, R.J. and Parry, L.A., 2020. Fossils improve phylogenetic analyses of morphological characters. bioRxiv.

CHAPTER 2

Table of Contents

2.1 Compiling, Installation, and Requirements

2.1.1 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 using GCC (64-bit)

To compile from command line.

1. Install GCC and Qt using system packages:

```
sudo apt-get install build-essential libgl1-mesa-dev
sudo apt install qt5-default
```

2. Download source code and navigate to folder, or alternatively clone using Git:

```
git clone https://github.com/palaeoware/trevosim.git
cd trevosim
```

3. Within REvoSim folder create makefile:

```
qmake ./trevosim.pro
```

4. Build by running the make command:

```
make
```

5. Navigate to bin folder (e.g. trevosim/bin) and launch software by double clicking on file.

Using Qt creator.

1. Install Q5.X on your system by running the installer from Qt: https://www.qt.io/download Further instructions are available here: https://wiki.qt.io/Install_Qt_5_on_Ubuntu
2. Download source code, launch Qt Creator, and open the .pro file. Configure build and follow standard debug/release procedure.

MacOS

QT Creator + QT v5.x

The above (Linux, using qtcreator) approach should also work for MacOS builds. This will require xcode to be installed, which you can do using the app store, followed by QtCreator, which can be achieved through the Qt online installer. To build the software, download source code, launch Qt Creator, and open the .pro file. Configure build and follow standard debug/release procedure.

2.1.2 Installation

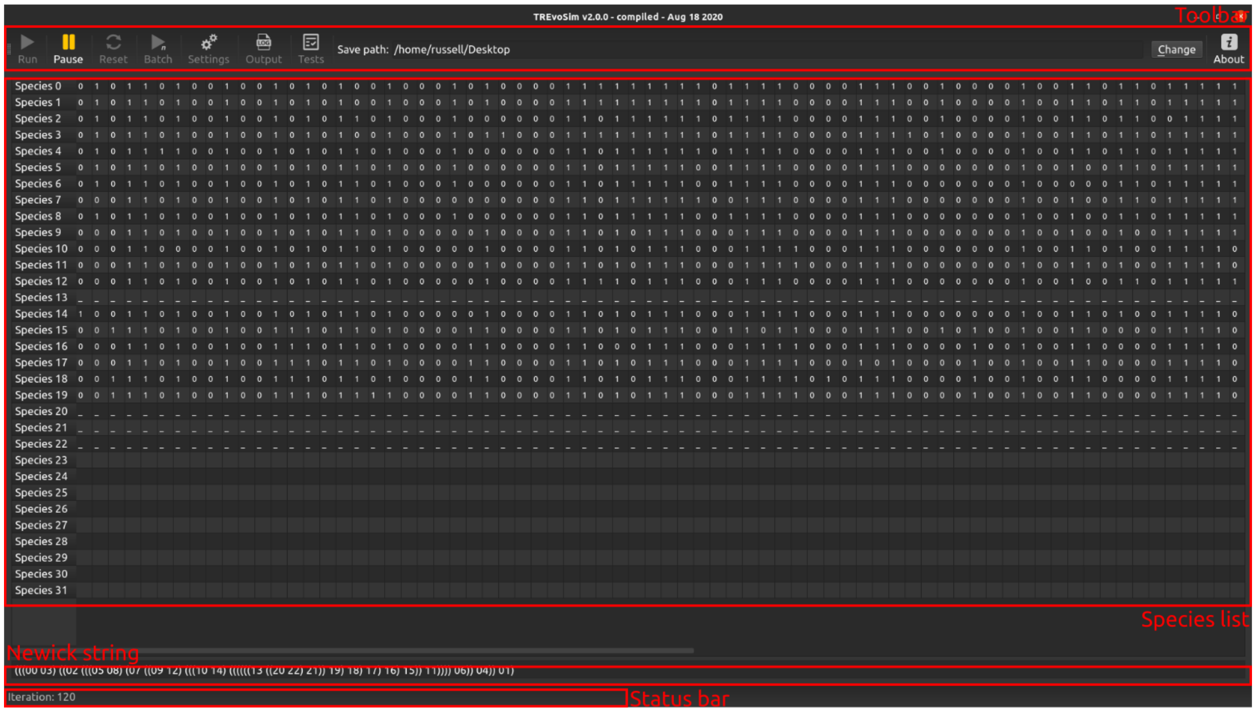
From the TREvoSim GitHub repository pre-compiled binary releases and packaged installers can be downloaded. For Windows users we provide both a portable binary release (.zip) - which just needs extracting to a convenient location - and a self contained installer. For Mac we provide a zip containing the TREvoSim program that can be downloaded from the TREvoSim GitHub repository. To install the software, drag and drop the required .app folder(s) into the Applications folder. You may be required to the approve the software in security and privacy settings before it will launch. For Linux users, the above instructions will allow the software to be built using a limited number of lines of bash. Please contact palaeoware@gmail.com if you encounter any issues.

2.1.3 Requirements

TREvoSim 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.

2.2 Window Layout



Save path: /home/russell/Desktop

```
Species 0   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 0 0 1 0 0 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1
Species 1   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 0 0 1 0 0 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1
Species 2   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 0 0 1 0 0 0 0 1 0 0 1 1 0 1 1 0 0 1 1 1 1
Species 3   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 0 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1
Species 4   0 1 0 1 1 1 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 0 0 1 0 0 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1
Species 5   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 1 1 1 1
Species 6   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 1 1 1 1 1
Species 7   0 0 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 1 1 1 1
Species 8   0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 1 1 1 1
Species 9   0 0 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 1 1
Species 10  0 0 0 1 1 0 0 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0 1 0 0 1 1 1 1 0
Species 11  0 0 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0 1 0 0 1 1 1 1 0
Species 12  0 0 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 1 1 0 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 1 1 1 1
Species 13  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
Species 14  1 0 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0 1 0 0 1 1 1 1 0
Species 15  0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 1 1 0 1 1 0 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 0 0 0 0 1 1 1 1 0
Species 16  0 0 0 1 1 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 0 1 1 1 1 0
Species 17  0 0 0 1 1 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 0 1 1 1 1 0
Species 18  0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 0 1 1 1 1 0
Species 19  0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 1 1 1 0 0 0 0 1 1 0 0 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 0 1 1 1 1 0
Species 20  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
Species 21  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
Species 22  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
Species 23
Species 24
Species 25
Species 26
Species 27
Species 28
Species 29
Species 30
Species 31
```

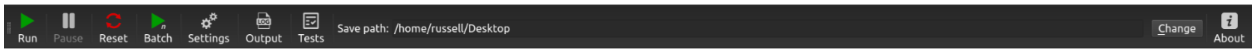
```
(((00 03) ((02 (((05 08) (07 ((09 12) (((10 14) ((((((13 ((20 22) 21)) 19) 18) 17) 16) 15)) 11)))) 06)) 04)) 01)
```

Iteration: 120

The main window comprises a number of elements, outlined below.

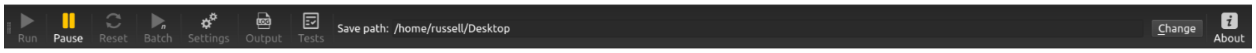
2.2.1 Toolbar

The main toolbar appears as follows when the simulation is not running:



Save path: /home/russell/Desktop

And when it is running, as below:



Save path: /home/russell/Desktop

The buttons control the simulation, as well as launching dialogues, and defining program output location.

Run This button launches a simulation, and then runs it until the requested number of species has evolved (see *Settings dialogue*), the simulation is paused, or cancelled (escape key).

Pause Pauses simulation, which can be resumed when requested by pressing pause again.

Reset Resets the simulation by removing all digital organisms from the playing field and species list.

Batch For repeated runs using the same settings, TREvoSim provides a batch mode. The number of runs is requested on launching batch mode, and output files are labelled accordingly.

Settings Launches *Settings dialogue* dialogue.

Output Launches *Logging the Simulation* dialogue.

Tests Runs the TREvoSim test suite, reporting test results in a new panel that appears on the right of the main window.

Save path All files created by the program are written to a folder titled *TREvoSim_output*, created in this location.

Change Launches a file explorer to change the save path.

About Opens a dialogue with information about TREvoSim, and links to the code and for bug reporting.

2.2.2 Species list

This panel updates as a run of the simulation progresses: it shows the binary string for the species in a simulation (up to the first 128 characters are displayed), once available. The panel starts as blank; once speciation occurs and any given species remains extant (i.e. there is a living representative in the playing field), the species is represented by a series of dashes. Once the species is extinct, and characters have been recorded (see algorithm description in Keating *et al.* 2019 for more details), these are printed to the panel.

2.2.3 Newick string

This text box updates with a Newick string showing the tree of the current simulation run, which updates as speciation events occur. This provides an immediate picture of, for example, the tree asymmetry within a run.

2.2.4 Status bar

The status bar is updated with messages during and after a run: for example, during a run it displays the iteration number, and afterwards it can provide messages about the data (e.g. number of uninformative characters, if these are not set to be stripped out, number of identical terminals). It also displays progress bars where one is required.

2.3 Settings dialogue

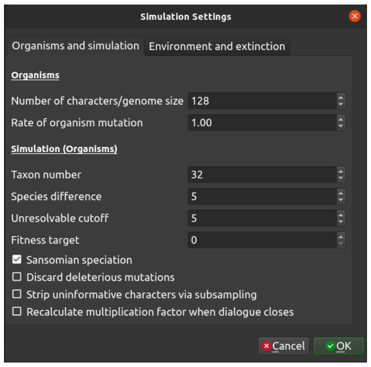
Clicking on the settings button of the toolbar will launch a settings dialogue which has two tabs - one with settings for the organisms in the simulation, and one controlling the environment for the simulation.

2.3.1 Organisms and simulation tab

Organisms

Number of characters/genome size This defines the number of bits in character string for each organism, and ultimately the character number of the matrix output by the software.

Rate of organism mutation This is the rate of mutation for the organisms in the simulation, in units of mutations per hundred characters per iteration.



Simulation Settings

Organisms and simulation | Environment and extinction

Organisms

Number of characters/genome size: 128

Rate of organism mutation: 1.00

Simulation (Organisms)

Taxon number: 32

Species difference: 5

Unresolvable cutoff: 5

Fitness target: 0

- ☑ Sansomian speciation
- ☐ Discard deleterious mutations
- ☐ Strip uninformative characters via subsampling
- ☐ Recalculate multiplication factor when dialogue closes

Simulations (Organisms)

Taxon number A simulation will run until this number of species has evolved, and then terminate.

Species difference The hamming distance between a selected organism, post mutation, and either the character string when the species originated, or the last species to originate within the lineage (see paper for rationale) is used within the TREvoSim species concept. This setting defines the required hamming distance for a speciation to have occurred.

Unresolvable cutoff With low character numbers, especially when TREvoSim is set to strip out uninformative characters, terminals can have the same character string within a matrix, and thus be unresolvable. This setting defines upper limit for identical terminals. If there are more unresolvable taxa than this number in a single run, output files are not written and a warning is provided. If batch mode is underway, the current run is discarded and started again.

Fitness target This is the target value for the count of 1s in the fitness calculation. See Garwood et al (2019) or Keating et al (2020) for details of the fitness algorithm. At low and high values (max being mask number multiplied by character number) fewer genomes will have peak fitness, at half max value, a larger number of peak fitness organisms exist. This can be quantified using fitness histogram menu command.

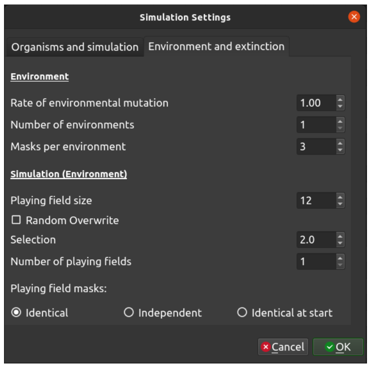
Sansomian speciation This option dictates when the character string of a species is recorded: at speciation, or at extinction. Sansomian speciation, the default, is to record the this when a species goes extinct. This ensures if it is a long-surviving species that the recorded characters string best reflects that closest to its sister group. When the option is not selected, genome is recorded at speciation.

Discard deleterious mutations Optionally, TREvoSim can accept only mutations which are neutral, or improve the fitness of an organism.

Strip uninformative characters via subsampling TREvoSim gives the option of writing matrices of only parsimony informative characters. When this option is checked, the software attempts to provide the requested character number of only informative characters. It achieves this by multiplying the number of characters and species difference by an empirically calculated factor (see below) at the start of a run. After a run has completed, informative characters are randomly subsampled to the requested number of characters; if there are not enough characters to achieve this, in batch mode the run is discarded and restarted, in single run mode an error message is provided.

Recalculate multiplication factor on dialogue close The multiplication factor required to achieve a set number of informative characters via subsampling depends on the settings of any given run. Selecting this option calculates and sets this factor empirically after the settings dialogue is closed by conducting a ten-run batch and working out the proportion of informative characters within those runs. This needs to be recalculated after any settings are changed. If it is not, or has not been set, the software will run with a large factor, and thus be slower than necessary.

2.3.2 Environment



Environment

Rate of environmental mutation	1.00
Number of environments	1
Masks per environment	3

Simulation (Environment)

Playing field size	12
☐ Random Overwrite	
Selection	2.0
Number of playing fields	1

Playing field masks:

[✓ Identical] Independent Identical at start

Environment

Rate of environmental mutation The environment in TREvoSim - which comprises a series of random numbers, or masks - also mutates, allowing lineages to track fitness peaks. This setting dictates the rate of mutation for the environment, in units of mutations per hundred characters per iteration.

Number of environments From v2.0.0 TREvoSim allows multiple environments (i.e. numerous sets of masks). During a simulation the fitness of each organism is calculated for every environment, and the organism's overall fitness is defined as that from the environment to which it is best suited (i.e. for which it has the highest fitness).

Masks per environment This dictates the number of masks (random numbers) in the environment. The more there are the flatter the fitness landscape is likely to be.

Simulation (Environment)

Playing field size This dictates the size of the playing field within the simulation, i.e. the number of individuals alive at any given time. Small sizes will lend themselves to asymmetrical trees with short tips.

Random overwrite When this is checked, when a new organism is returned to the playing field (see Keating et al. 2020 for algorithm details), it will overwrite an individual at random. When it is not checked it overwrites the organism with the lowest fitness (or one of these at random if multiple individuals share the lowest fitness).

Selection This dictates the probability of choosing any given individual when moving down the playing field in the coin toss to select an individual to duplicate. The probability of selecting an individual is the reciprocal of this (i.e. 1 / this number). If, e.g., this is 2.0 there is a 50% chance of selecting the first organism in the playing field, then 50% selecting the next, and so on.

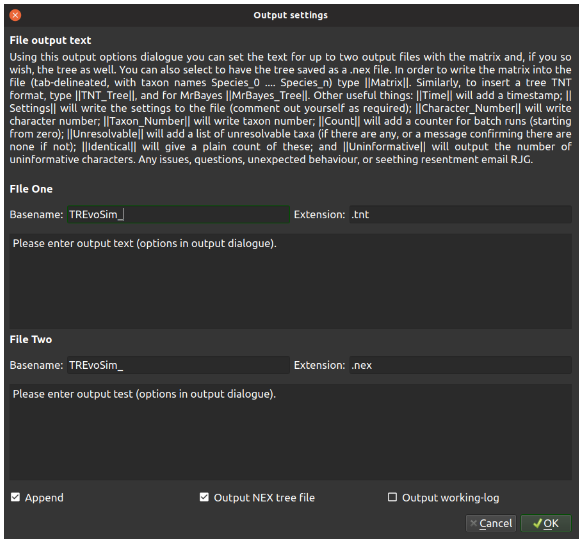
Number of playing fields From v2.0.0 TREvoSim allows multiple playing fields. These are initialised with the same individual, and then operate individually (each playing field will form a clade). At present the released version of TREvoSim does not allow mixing between playing fields.

Playing field masks The options here define the behaviour of masks across playing fields. They can be identical, start identical and then evolve away from each other, or be independent. If the last is selected the playing fields are initialised with the individual that has the highest mean fitness across all playing fields after 5000 initialisations (with a single playing field the initialising organism is within the top 10% of possible fitnesses for the starting masks).

2.4 Logging the Simulation

2.4.1 Logging system

TREvoSim has a versatile logging system which allows the user to define outputs in a range of formats required for phylogenetic inference packages, provided these allow plain text inputs. Clicking on the output button of the toolbar will launch the output dialogue:



File output text

Using this output options dialogue you can set the text for up to two output files with the matrix and, if you so wish, the tree as well. You can also select to have the tree saved as a .nex file. In order to write the matrix into the file (tab-delineated, with taxon names Species_0 .... Species_n) type ||Matrix||. Similarly, to insert a tree TNT format, type ||TNT_Tree||, and for MrBayes ||MrBayes_Tree||. Other useful things: ||Time|| will add a timestamp; ||Settings|| will write the settings to the file (comment out yourself as required); ||Character_Number|| will write character number; ||Taxon_Number|| will write taxon number; ||Count|| will add a counter for batch runs (starting from zero); ||Unresolvable|| will add a list of unresolvable taxa (if there are any, or a message confirming there are none if not); ||Identical|| will give a plain count of these; and ||Uninformative|| will output the number of uninformative characters. Any issues, questions, unexpected behaviour, or seething resentment email RJG.

FIle One

Basename: TREvoSim_ Extension: .tnt

Please enter output text (options in output dialogue).

File Two

Basename: TREvoSim_ Extension: .nex

Please enter output test (options in output dialogue).

[✓] Append [✓] Output NEX tree file [ ] Output working-log

This provides the options for two custom log files, which are placed, after a run, within the *TREvoSim_output* folder created on the save path. Basename defines the start of the filename, which then either includes a number, which iterates with subsequent runs when output is not set to append, or *_batch* if the outputs are set to append and a batch is being run. The file extension is also defined here. The two text boxes then allow custom file content to be written: text is written as provided to the file - for example with run instructions or program commands - and any of the keywords below (note this is case sensitive) included within two vertical bar (|) symbols are replaced as a file is written.

2.4.2 Keywords

Keywords within two vertical bars (e.g. |Matrix|) are replaced as a file is written as follows:

Matrix This is replaced with the matrix from the run.

TNT_Tree This writes a tree, if required, in TNT format (i.e. only brackets and terminal labels), e.g.

```
(((((00 ((((((((((((((05 20) 19) 18) 17) 16) 15) (14 ((((21 ((25 ((((27 31) 30) 29)␣
↪28)) 26)) 24) 23) 22))) 13) 12) 11) 10) 09) 08) 07) 06)) 04) 03) 02) 01)
```

MrBayes_Tree This writes a tree in standard Newick format, including branch lengths (these are based on iteration number throughout), e.g.

```
(S_01:13,(S_02:8,(S_03:13,(S_04:32,((S_06:4,(S_07:10,(S_08:6,(S_09:5,(S_10:10,(S_11:6,
↪(S_12:6,(S_13:2,(((S_22:11,(S_23:12,(S_24:9,((S_26:28,((S_28:37,(S_29:2,(S_30:2,(S_
↪31:1,S_27:1):1):38):23):20,S_25:12):31):41,S_21:31):24):27):1):70,S_14:13):42,(S_
↪15:21,(S_16:2,(S_17:11,(S_18:2,(S_19:10,(S_20:14,S_
↪05:25):7):1):4):2):1):2):1):13):58):6):2):2):5):2):36,S_00:20):3):1):3):7):85
```

Note The mechanism used for tree writing differs between the above requests - the tree topology is the same, but the taxon order differs.

Time Adds a timestamp.

Settings Writes the settings to the file (this is provided as a useful way to record, with any output data, the set up for TREvoSim for any given run).

Character_Number This outputs the character number.

Taxon_Number Writes taxon number.

Count This is replaced with a counter for batch runs; incrementing by one using C++ numbering (i.e. starting from zero).

Unresolvable This prints a list of unresolvable taxa (or a notice that there are none if required).

Uninformative Writes the number of uninformative characters.

So, for example, the following would output a block of text that could be run as a macro in tnt:

```
NSTATES nogaps;
xread
'Written on ||Time|| Variables: ||Settings||'
||Character_Number|| ||Taxon_Number||
||Matrix||
;
piwe-;
keep 0; hold 100000;
rseed *;
xmult = level 10; bbreak;
export - ||Count||_POUT.nex;
xwipe;
```

There are three further options at the base of the dialogue:

Append When checked, in batch mode, all the outputs are appended onto a single file, allowing them to be run as a single analysis in - for example - TNT and MrBayes.

Output NEX tree file This outputs a tree for each run in a standard nexus format with the tree and translate block, as well as a comment with the settings of the run written to it. These do not append, but each file has the run number at the end of the file name.

Output working log When this is checked, TREvoSim outputs a text file outlining many of the steps each iteration, such as the state of the playing field, the environment, and the processes the software is going through. This helps understand and fact check any given run, but for significant playing field sizes, taxon numbers, or character numbers, it creates a significant (10s - 100s of MB) text file.

Should any other output options be required, please file a feature request. Keywords are not case sensitive.

2.5 Menu commands

TREvoSim has a series of options which can be selected using the Commands menu at the top left of the program (above the run button). These are as follows.

2.5.1 Save Current Settings

By default TREvoSim saves the simulation settings between sessions (it writes a settings file to the same folder as the software binary / executable on close). This menu command (or shortcut ctrl + s) rewrites that setting file without closing the program.

2.5.2 Save settings as. . .

This opens a file save dialogue, and then writes a TREvoSim settings file to the requested location (shortcut ctrl + shift + s). TREvoSim settings files are written in an XML document that can be opened in any text application.

2.5.3 Load settings from file. . .

Load settings file (ctrl + o) opens a saved settings file and updates the simulation settings accordingly.

2.5.4 Restore default settings

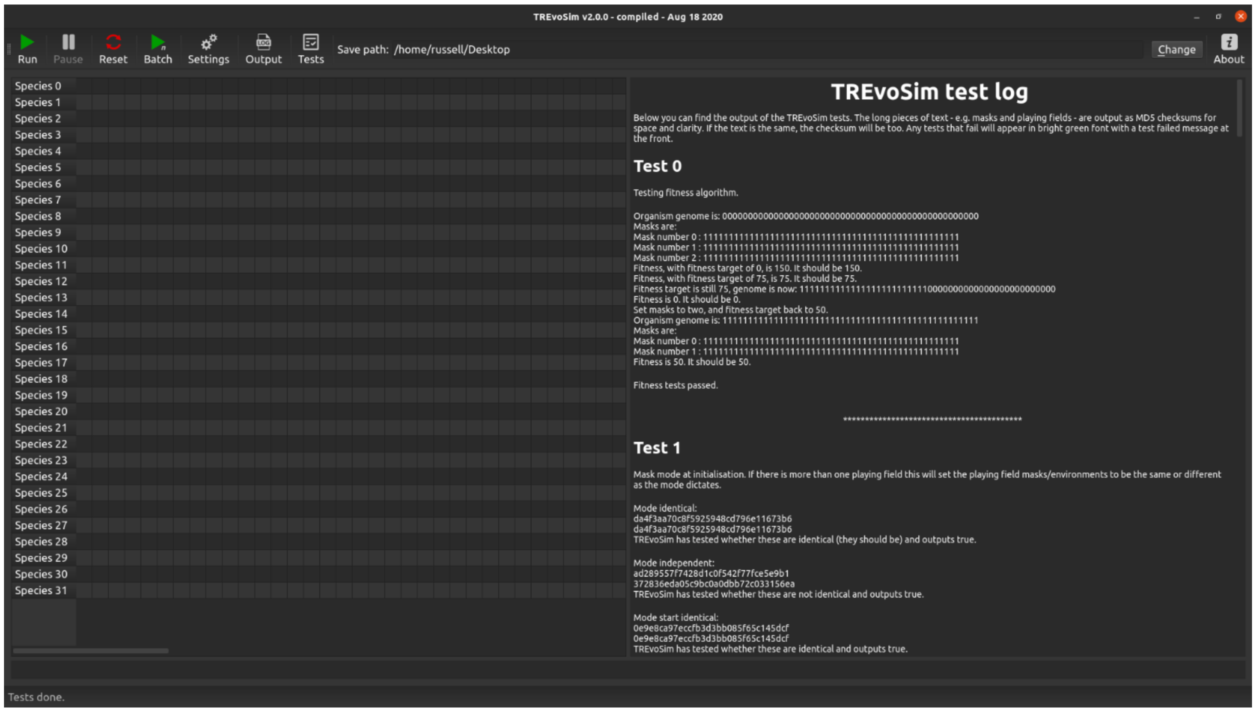
If at any point you require default settings, this menu option (or shortcut ctrl + shift + d) restores all settings to default.

2.5.5 Fitness histogram

The fitness algorithm for TREvoSim is described in Keating et al. (2020). The fitness landscape varies depends on the simulation settings. This menu command (shortcut ctrl + shift + h) provides a basic assessment of that landscape by showing the distribution of fitnesses for every possible genome for the current settings. It provides an option to select how many bits in the test organism's genome as larger genome sizes (e.g. >16) take exponentially longer given the need to try every potential genome rearrangement.

2.5.6 Run tests

TREvoSim includes a test mode, which is launched by clicking the Tests button on the main toolbar, this menu command, or the shortcut ctrl + shift + t. The main window is first split into two, and a series of tests are run, outputting the results to a panel on the right, shown below.



```
TREvoSim test log

Below you can find the output of the TREvoSim tests. The long pieces of text - e.g. masks and playing fields - are output as MD5 checksums for
space and clarity. If the text is the same, the checksum will be too. Any tests that fail will appear in bright green font with a test failed message at
the front.

Test 0

Testing fitness algorithm.

Organism genome is: 000000000000000000000000000000000000000000000000
Masks are:
Mask number 0 : 111111111111111111111111111111111111111111111111
Mask number 1 : 111111111111111111111111111111111111111111111111
Mask number 2 : 111111111111111111111111111111111111111111111111
Fitness, with fitness target of 0, is 150. It should be 150.
Fitness, with fitness target of 75, is 75. It should be 75.
Fitness target is still 75, genome is now: 111111111111111111111111000000000000000000000000
Fitness is 0. It should be 0.
Set masks to two, and fitness target back to 50.
Organism genome is: 111111111111111111111111111111111111111111111111
Masks are:
Mask number 0 : 111111111111111111111111111111111111111111111111
Mask number 1 : 111111111111111111111111111111111111111111111111
Fitness is 50. It should be 50.

Fitness tests passed.


                    ****************************************

Test 1

Mask mode at initialisation. If there is more than one playing field this will set the playing field masks/environments to be the same or different
as the mode dictates.

Mode identical:
da4f3aa70c8f5925948cd796e11673b6
da4f3aa70c8f5925948cd796e11673b6
TREvoSim has tested whether these are identical (they should be) and outputs true.

Mode independent:
ad289557f7428d1c0f542f77fce5e9b1
372836eda05c9bc0a0dbb72c033156ea
TREvoSim has tested whether these are not identical and outputs true.

Mode start identical:
0e9e8ca97eccfb3d3bb085f65c145dcf
0e9e8ca97eccfb3d3bb085f65c145dcf
TREvoSim has tested whether these are identical and outputs true.
```

This allows you to check TREvoSim is running as expected and described in the publications introducing the model (you can alternatively/additionally output a working log to check the internal workings of the software - see *Logging the Simulation*). The output in the test log describes each test, and also the expected output. The system the makes sure the output meets these expectations. Any failed tests will be highlighted in green.

2.5.7 Set uninformative factor

This menu command (or shortcut ctrl + shift + f) opens a dialogue that allows the strip uninformative factor to be set manually.

2.5.8 Random Seed

By default, the organism used to initialise a simulation is one near peak fitness for the starting environment(s), to prevent the resulting tree from documenting a lineage adapting to a fitness peak (this results in a highly asymmetrical tree). This menu command (shortcut ctrl + shift + r) toggles between this approach, and one in which a random individual is used to initialise the simulation.