SIGMUND user manual

This is the user manual of mutualistic model[1] simulator SIGMUND.

Github repository: <u>https://github.com/jgalgarra/sigmund</u>

Release: 2.03

Date: September 2014

Author: Javier García-Algarra, Complex System Groups, Universidad Politécnica de Madrid, Spain.

1   Installation

1.1   Requirements

SIGMUND requires *python* 3.2 or higher to run. Packages `numpy`, `scipy`, `matplotlib` and `PyQt` must be installed. A good option is downloading the `Anaconda` distribution[2] that includes all the required software. In addition, `git` must be enabled. The simulator has been succesfully tested under *Windows* and *Ubuntu.*

Go to the directory where you want to install SIGMUND. From the UNIX shell or git bash in Windows, invoque:

```
git clone https://github.com/jgalgarra/sigmund
```

SIGMUND is ready to run. Navigate to the directory `sigmund/src/pak_tfm` that has been created under the directory from where you invoked the `git clone` command. Under Windows double click on the sigmund_tool.py icon to launch the program. Under Linux write the command:

```
./sigmund_tool.py
```

---

1 The model is described in Javier García-Algarra et al. (2014) "Rethinking the logistic approach for population dynamics of mutualistic interactions", *Journal of Theoretical Biology,* 363, 332-343.

2 https://store.continuum.io/cshop/anaconda/

1.2 User interface

SIGMUND user interface is built with the `Qt` framework, so details may look slightly different depending on your machine and Operating System.
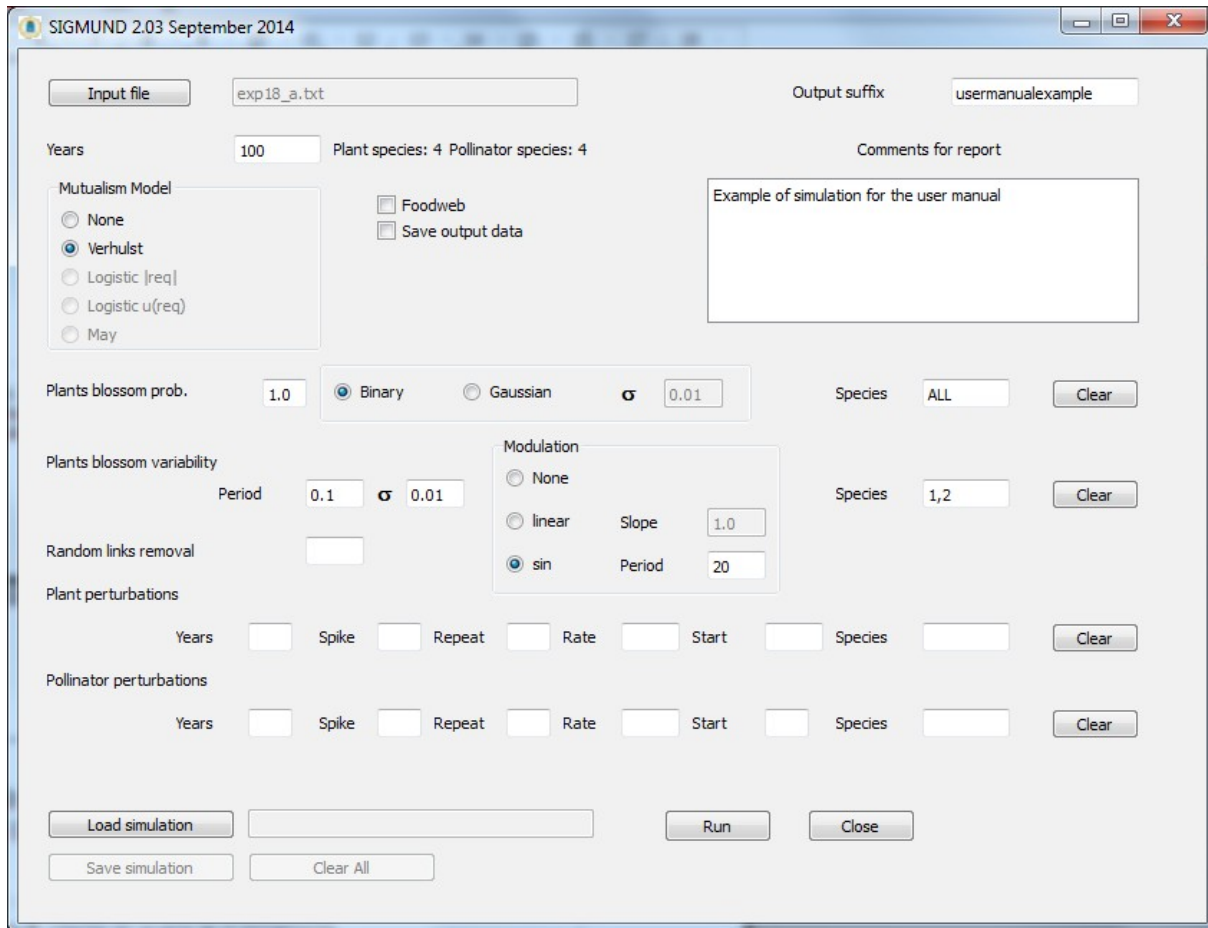
SIGMUND User interface

This is the meaning of the different fields:

- **Input File**. Name of simulation configuration files stored in /input . You must choose it or it may be automatically picked if you load an stored simulation. The UI will show the number of species of each guild.

- **Years.** Span of simulation, mandatory.

- **Output suffix.** Optional suffix that will be appended to the results files that are written in /output.

- **Comments for report**. Optional field to include a free text describing the experiment that will be included in the report file.

- **Mutualism Model.** Allows choosing between no mutalism and logistic like (Verhulst)

mode of simulation.

- **Food web**. Check if there is a superimposed food web.

- **Save output data**. Check if you want to save numeric simualtion data in text files.

The system may be attacked with four classes of perturbations (optional).

- **Random links removal.** Percentage of mutualistic links that are randomly removed (No effect if the field is left blank or set to zero).

- **Plants blossom probability**. Simulation of random strength of plants blossom.

  - **Probability**. Blossom probability, (range 0 to 1.0, default 1.0).

  - **Type of distribution**. Binary or Gaussian. Binary performs a Bernouilli trial with the set probability for each year. Intensity is 100% is result is successful or zero otherwise. Gaussian modulates intensity with a normal distribution, with mean equal to the probability parameter and the set standard deviation.

  - **Species**. List of affected plant species (from 1 to n, or ALL).

- **Plants blossom variability**. Simulation of random blossom period.

  - **Period**. Fraction of year.

  - **Deviation**. Variability of blossom period.

  - **Type**. None, linear (define by slope) or sinusoidal (defined by period).

  - **Species**. List of affected plant species (from 1 to n, or ALL).

- **Forced external perturbations**. Exogen attacks (epidemics, droughts, migrations, etc) that may lead to step death rate increases.

  - **Years**. Perturbation span.

  - **Spike**. Fraction of the perturbation span where the perturbation is active (useful for repetitions).

  - **Repeat**. Number of times the perturbation is repeated.

  - **Rate**. Yearly death rate increase.

  - **Start**. Year when the perturbation starts.

  - **Species**. List of affected species.

- **Load simulation**. Load stored simulation parameters.

- **Save simulation**. Store simulation parameters.

1.3   Experiment results

Each experiment produces a set of results. The system console shows real time information on the experiment behavior. These data are written to the HTML report file in the directory /output.. Figures and data of population and growth rates are also stored in that directory.

```
Binomial simulated mutualistic interaction. Input file: exp18
==============================================================
User Comment: Example of simulation for the user manual
Span: 100 years
ALGORITHM: Verhulst
Release 2.03
Plants matrix: exp18_a.txt
Pollinators matrix: exp18_b.txt
Plant species: 4
Plant initial populations [800, 900, 700, 270]
Pollinator species: 4
Pollinator initial populations [1100, 700, 400, 350]
Blossom probability 1.0, type Binary. Plant affected species:['ALL']

Blossom variability active. Period: 0.10 (% of year), Initial moment standard
dev.: 0.0100 (% of year), Type: sin, Period 20.00 . Affected species:[1, 2]
Elapsed time 6.51 s


Created 2014-10-03 23:52:18.363000
```

1.4   Offline script

SIGMUND may be launched from the command line in batch mode to perform multiple experiments. The script is called `sigmund_standalone.py` . The simulation will take the parameters stored in the simfile unless they are changed using the optional command arguments:

```
usage: sigmund_standalone.py [-h] [-simfile SIMFILE_NAME] [-g] [-v]
                             [-years SIM_YEARS] [-fw] [-outsf OUTSF] [-ds]
                             [-dsdir DSDIR] [-stop] [-rlink RLINK]
                             [-Blprob BLPROB] [-Blsd BLSD] [-Bltype BLTYPE]
                             [-Blspecies BLSPECIES] [-Bssvarper BSSVARPER]
                             [-Bssvarsd BSSVARSD] [-Bssvartype BSSVARTYPE]
                             [-Bssvartype_param BSSVARTYPE_PARAM]
                             [-Bssvarspecies BSSVARSPECIES]
                             [-pl_ext_period PL_EXT_PERIOD]
                             [-pl_ext_spike PL_EXT_SPIKE]
                             [-pl_ext_numperiod PL_EXT_NUMPERIOD]
                             [-pl_ext_rate PL_EXT_RATE]
                             [-pl_ext_start PL_EXT_START]
                             [-pl_ext_species PL_EXT_SPECIES]
                             [-pol_ext_period POL_EXT_PERIOD]
                             [-pol_ext_spike POL_EXT_SPIKE]
                             [-pol_ext_numperiod POL_EXT_NUMPERIOD]
                             [-pol_ext_rate POL_EXT_RATE]
                             [-pol_ext_start POL_EXT_START]
                             [-pol_ext_species POL_EXT_SPECIES]


optional arguments:
  -h, --help            show this help message and exit
  -simfile SIMFILE_NAME  Simulation parameters file
  -g                    Generate and store graphs
  -v                    Verbose output
  -years SIM_YEARS      Simulation span in years
  -fw                   Superimposed food web
  -outsf OUTSF          Append suffix to output file names
  -ds                   Results data save
  -dsdir DSDIR          Data save directory
  -stop                 On extinction stop simulation
  -rlink RLINK          Randomlinks removal
  -Blprob BLPROB        Blossom probability
  -Blsd BLSD            Blossom deviation
  -Bltype BLTYPE        Blossom type: Binary, Gaussian
  -Blspecies BLSPECIES  Blossom species
  -Bssvarper BSSVARPER  Blossom variability period
  -Bssvarsd BSSVARSD    Blossom variability deviation
  -Bssvartype BSSVARTYPE Blossom variability modulation type: None, linear, sin
  -Bssvartype_param BSSVARTYPE_PARAM Blossom variability modulation parameter
  -Bssvarspecies BSSVARSPECIES Blossom variability affected species
  -pl_ext_period PL_EXT_PERIOD Plants external perturbation, period in years
  -pl_ext_spike PL_EXT_SPIKE Plants external perturbation, spike (fraction of
period)
  -pl_ext_numperiod PL_EXT_NUMPERIOD Plants external perturbation, repeat
perturbation
  -pl_ext_rate PL_EXT_RATE Plants external perturbation, rate
  -pl_ext_start PL_EXT_START Plants external perturbation, initial year
  -pl_ext_species PL_EXT_SPECIES Plants external perturbation, affected species
  -pol_ext_period POL_EXT_PERIOD Pollinators external perturbation, period in years
  -pol_ext_spike POL_EXT_SPIKE Pollinators external perturbation, spike (fraction
of period)
```

```
  -pol_ext_numperiod POL_EXT_NUMPERIOD Pollinators external perturbation, repeat
perturbation
  -pol_ext_rate POL_EXT_RATE Pollinators external perturbation, rate
  -pol_ext_start POL_EXT_START Pollinators external perturbation, initial year
  -pol_ext_species POL_EXT_SPECIES Pollinators external perturbation, affected
species
```

Usage example:

```
./sigmund_standalone.py -simfile exp18_Verhulst_oso_20.sim -y 50 -g -rlink 0.1 -v
```

This example loads the simulation conditions stored in `exp18_Verhulst_oso_20.sim` sets a sapn of 50 years, generates and stores the pictures, removes a 10% of links and runs in verbose mode (the same console output as if it was launched from the User Interface).

If flag -v is not activated, the program is run in quiet mode. It returns either `SURVIVAL` or `EXTINCTION`. In combination with the `-stop` flag this feature may be used to run multiple simulations to check survavility of the whole system under certain parameters.

This is an example of a python script to compute the rate of survival when a 70% of links are removed at random. The simulation span is set to 50 years and the scripts ends in the moment extinction is detected:

```python
import subprocess
survival_success  =  0
no_experiments  =  50
com_base =  "python3 sigmund_standalone.py -simfile exp18_10.sim -
y 50 -stop -rlink 0.7"
for i in range(0,no_experiments):
      b = subprocess.check_output(com_base+str(rrate), shell=True)
      if str(b).find("EXTINCTION") == -1:
          print("remove links "+str(rrate)+" survival")
          survival_success += 1
      else:
          print("remove links "+str(rrate)+" extinction")
print("Experiments "+str(no_experiments)+"/ Network survived " +
str(survival_success)+" times")
```

Please, note that we are invoking the sigmund_standalone.py as an external process. You can get the same results writing your own script in your preferred programming language or scripting.