PyToxo user manual

PyToxo 1.0
English version

Borja González Seoane
C.e.: `borja.gseoane@udc.es`

Facultade de Informática
Universidade da Coruña

Wednesday 23rd June, 2021

Contents

List of Codes

List of Figures

1 Requirements

In order to install PyToxo it is necessary to have installed Python 3.8 or higher previously. It is recommended to follow the official instructions of the Python Software Foundation [5] to install Python 3.8.10, which is the latest version of Python 3.8. The user's Python installation must also include the PIP tool [3].

In this manual it will be assumed that the user has completed the Python installation process and correctly configured the PATH variable of their environment, so that both Python itself, as well as PIP and the packages installed as a result of it, are reachable from a command terminal.

PyToxo works in any of its forms —programmatic library, CLI, and GUI— on Linux, Mac OS, and Windows systems. Specifically, it has been verified on Linux Lubuntu 20, Apple Mac OS Big Sur and Microsoft Windows 10. However, its few dependencies make PyToxo a very portable application that is unlikely to present compatibility problems even on older platforms.

In addition to the above, if you want to use the PyToxo graphical interface, you must have Tk [6]. Tk has been adapted to work on most variants of Linux, Mac OS, and Windows, and is included in most modern Python distributions for the older platforms. This is not the case for some Linux distributions, where an additional package must be installed. However, PyToxo itself detects this deficiency when trying to launch the interface, and suggests to the user the command to use to install the module, which in Debian-based operating systems is:

```
sudo apt install python3-tk
```

2 Installation

PyToxo is available from PyPI [4], the official Python repository, so once you have completed the steps in the previous section, just run the following command to install it:

```
pip install pytoxo
```

Once this is done, the `pytoxo` library and the executables `pytoxo_cli` and `pytoxo_gui` will be available on the user's machine.

3 Use as a Python library

When using PyToxo, and as it is quite idiosyncratic in the Python community, the best documentation is made up of the source code headers themselves, which in PyToxo have been written in great detail according to the officially recommended convections.

For illustrative purposes, below we are going to address the use of PyToxo as a Python library through a series of examples. This demo is available in the form of a Jupyter Notebook [2] in the PyToxo repository.

The first thing we have to do to use PyToxo is to import the library:

```
import pytoxo
```

Then, from a model in a CSV file, we can generate a PyToxo `Model` object with:

```
model = pytoxo.Model(filename="../models/additive_3.csv")
```

And immediately afterwards we can generate a penetrance table using the appropriate method of the `model` object. This method will be `find_max_prevalence_table` or `find_max_heritability_table`, depending on whether we want to maximize prevalence or heritability, respectively.

```
# Definimos los parámetros del experimento
mafs = [0.4, 0.4, 0.4]
heritability = 0.85

table = model.find_max_prevalence_table(mafs=mafs, h=heritability)
```

The object `table`, of class `PTable`, contains our penetrance table. Now we could print it on the screen or save it as a file, for example with:

```
table.print_table(format="gametes")
```

Using PyToxo as a library we also have the possibility of entering the data of the original epistatic model directly, without using to an existing CSV file. In the Code 1 is a complete usage example that starts from two lists with the model data.

4 Use from the command line interface

To invoke the CLI, just use the command:

```
pytoxo_cli
```

We can summarize the use of the interface in the following POSIX [1] specification, which we break down below, starting with the optional arguments:

```
pytoxo [-h] [--gametes] (--max_prev | --max_her) <model> <prev_or_her> <maf>
    [<maf> ...]
```

- `-h`: using this optional argument we display the command line help.

```python
import pytoxo
import numpy

genotypes = ["AABB", "AABb", "AAbb", "AaBB", "AaBb", "Aabb", "aaBB", "aaBb",
    "aabb"]
probabilities = numpy.array(
    [
        "x",
        "x",
        "x",
        "x",
        "x*(1+y)",
        "x*(1+y)",
        "x",
        "x*(1+y)",
        "x*(1+y)",
    ]
)  # We also can use Numpy arrays instead of lists
model = pytoxo.Model(
    definitions=genotypes,
    probabilities=probabilities,
    model_name="other_model",
)
table = model.find_max_heritability_table(mafs=[0.1] * model.order, p=0.96)
table.print_table()
```

Code 1: Example using PyToxo as a library and manually entering the model

- `--gametes`: optional argument that implies that the output table will be composed in the format of GAMETES [7]. If this argument is not used, the table will be configured as CSV, by default.
- `--max_prev` or `--max_her`: using the first will maximize the prevalence and using the second, the heritability. These options are mutually exclusive, specifying one is mandatory, and implies that the last argument entered before the MAF will be treated as heritability, if prevalence is maximized; or as prevalence, if heritability is maximized.

The previous optional arguments can be distributed in any position of the command without affecting it. However, positional arguments, or arguments that do not match with any of the previously defined options, are interpreted according to the order in which they are provided:

- The first positional argument will be the path to the CSV file containing the epistatic model.
- The second positional argument will correspond to the heritability or the prevalence to be set, depending on what has been selected with `--max_prev` or `--max_her`.
- The positional arguments that follow the previous ones will all be interpreted as each of the MAFs, respectively. The number of MAFs provided has to match the order of the model previously introduced. MAFs are separated by spaces, just as any other argument.

```
pytoxo_cli --gametes --max_prev models/additive_4.csv 0.6 0.2 0.3 0.3 0.4

pytoxo_cli models/additive_4.csv --max_prev --gametes 0.6 0.2 0.3 0.3 0.4

pytoxo_cli models/additive_4.csv 0.6 0.2 0.3 0.3 0.4 --max_prev --gametes

pytoxo_cli models/additive_4.csv 0.6 --max_prev 0.2 0.3 0.3 0.4 --gametes

pytoxo_cli models/threshold_8.csv --max_her 0.88 0.01 0.01 0.01 0.01 0.01 0.01
    0.01 0.01

pytoxo_cli models/additive_2.csv --max_her 0.4 0.45 0.5
```

Code 2: Examples of well-formed commands that make use of the PyToxo CLI

In the Code 2 are some examples of well-formed commands that make use of the PyToxo CLI. The first four cases above are different ways of writing the exact same experiment.

Once the penetrance table is calculated from the command line interface, it is printed directly to the standard output configured in the terminal. To save the table to a file, you must follow any of the usual methods in this type of workflow, such as redirecting the output of the command:

```
pytoxo_cli models/additive_3.csv --max_prev 0.75 0.5 0.45 0.5 > my_table.csv
```

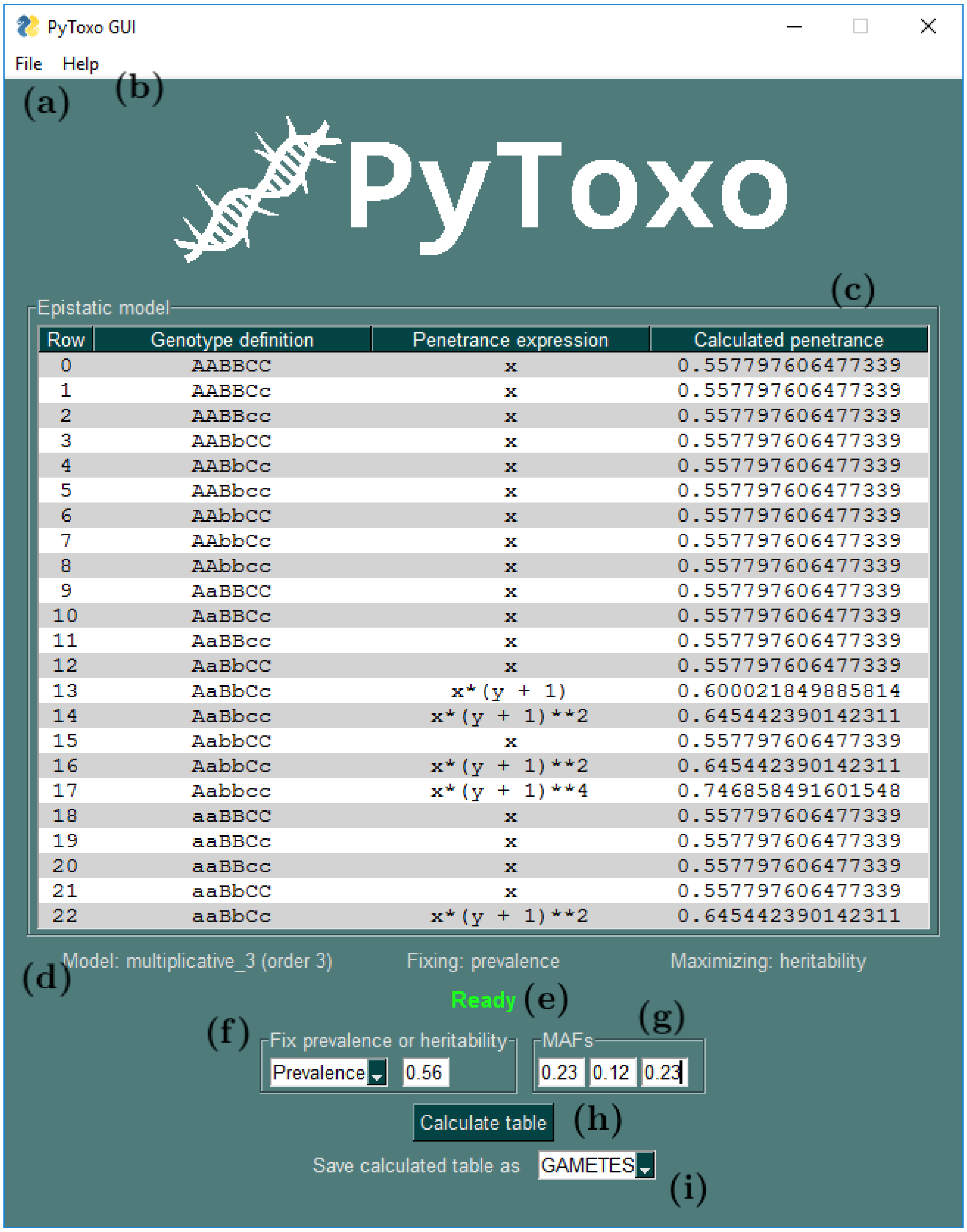
5 Use from the graphical user interface

To start the GUI, just use the command:

```
pytoxo_gui
```

In Figure 1 we have marked the different components that make up the graphical interface, and we comment on them below:

- **a**: file contextual menu, from which to load a model, delete it, save a penetrance table as a file or close the application.
- **b**: contextual help menu, from which useful information about the application can be accessed.
- **c**: main grid of the screen, with the representation of the model. In the case of Figure 1 it also contains the penetrations already calculated in the third column.
- **d**: informative texts to support the user, which are dynamically updated.
- **e**: program status indicator, which toggles when the table calculation process starts.



Epistatic model

Row	Genotype definition	Penetrance expression	Calculated penetrance
0	AABBCC	x	0.557797606477339
1	AABBCc	x	0.557797606477339
2	AABBcc	x	0.557797606477339
3	AABbCC	x	0.557797606477339
4	AABbCc	x	0.557797606477339
5	AABbcc	x	0.557797606477339
6	AAbbCC	x	0.557797606477339
7	AAbbCc	x	0.557797606477339
8	AAbbcc	x	0.557797606477339
9	AaBBCC	x	0.557797606477339
10	AaBBCc	x	0.557797606477339
11	AaBBcc	x	0.557797606477339
12	AaBbCC	x	0.557797606477339
13	AaBbCc	x*(y + 1)	0.600021849885814
14	AaBbcc	x*(y + 1)**2	0.645442390142311
15	AabbCC	x	0.557797606477339
16	AabbCc	x*(y + 1)**2	0.645442390142311
17	Aabbcc	x*(y + 1)**4	0.746858491601548
18	aaBBCC	x	0.557797606477339
19	aaBBCc	x	0.557797606477339
20	aaBBcc	x	0.557797606477339
21	aaBbCC	x	0.557797606477339
22	aaBbCc	x*(y + 1)**2	0.645442390142311

Model: multiplicative_3 (order 3) Fixing: prevalence Maximizing: heritability

Ready

Fix prevalence or heritability: Prevalence 0.56

MAFs: 0.23 0.12 0.23

Save calculated table as GAMETES

Figure 1: PyToxo GUI from Windows 10 with its different components marked for reference

- **f**: drop-down to select the parameter to be set between prevalence and heritability, and space to enter its value.
- **g**: spaces to fill in the MAF to use. The number of spaces is dynamically adapted to the order of the loaded model.
- **h**: button to calculate the penetrance table with the completed configuration.
- **i**: drop-down to select the table format to save it as a file, from CSV and GAMETES [7] format.

The GUI is very easy to use because it dynamically adapts to the state of the application workflow. The buttons are enabled or disabled depending on whether a model is loaded, the relevant fields filled in, etc. For example, we cannot fill in MAFs until we have loaded a model and we cannot use the calculate button until all parameters have been filled.

Bibliography

[1] IEEE Computer Society and The Open Group, "IEEE Standard for Information Technology–Portable Operating System Interface (POSIX(TM)) Base Specifications, Issue 7," *IEEE Std 1003.1-2017 (Revision of IEEE Std 1003.1-2008)*, Jan. 2018.

[2] Project Jupyter, "Jupyter," https://jupyter.org, accessed: 16 June 2021.

[3] Python Packaging Authority, "PIP Documentation v21.1.2," https://pip.pypa.io/en/stable/, accessed: 3 June 2021.

[4] Python Software Foundation, "PyPI," https://pypi.org, accessed: 28 May 2021.

[5] ——, "Python 3.8.10," https://www.python.org/downloads/release/python-3810/, accessed: 31 May 2021.

[6] Tcl Core Team, "Tcl Developer Xchange," http://www.tcl.tk, accessed: 9 June 2021.

[7] R. J. Urbanowicz, J. Kiralis, N. A. Sinnott-Armstrong, T. Heberling, J. M. Fisher, and J. H. Moore, "GAMETES: a fast, direct algorithm for generating pure, strict, epistatic models with random architectures," *BioData Mining*, vol. 5, no. 1, p. 16, Dec. 2012. [Online]. Available: http://biodatamining.biomedcentral.com/articles/10.1186/1756-0381-5-16