SyncPy2 UIs documentation

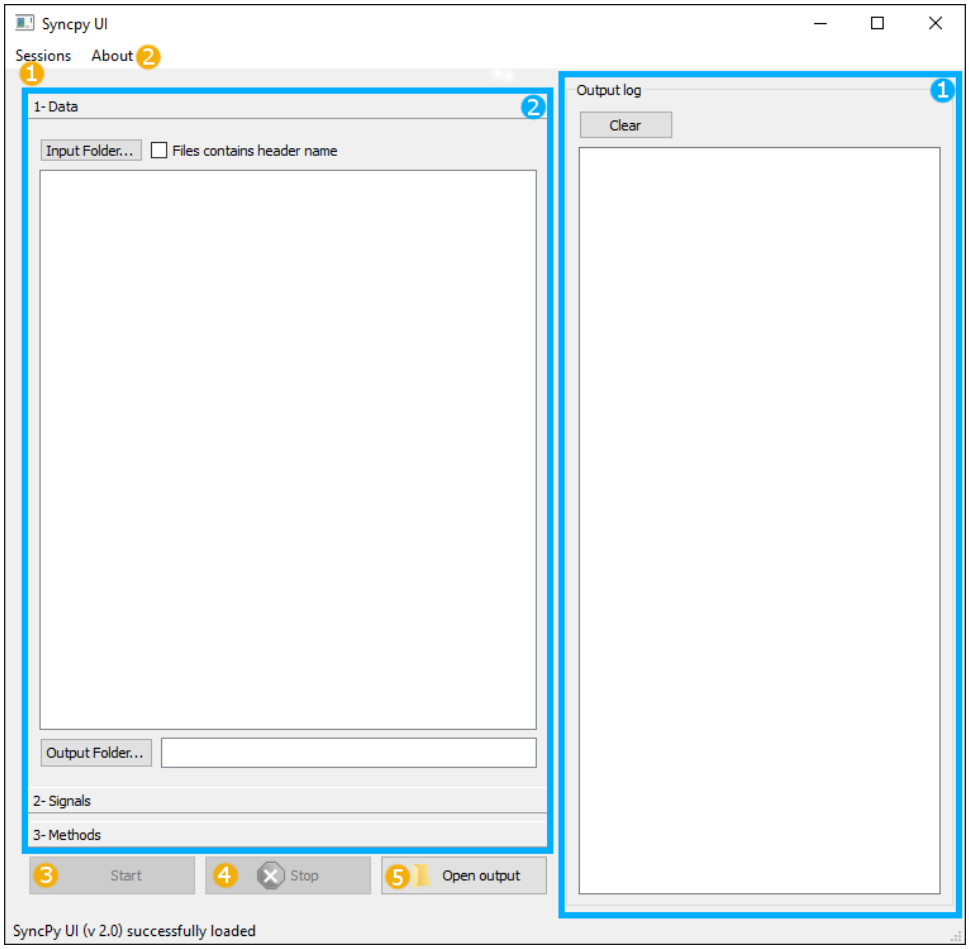
Part 1 – Installation

You need to install anaconda to be able to launch the entire SyncPy suite. If you download the SyncPy installer from the website, it will also install anaconda for you.

You can access the UIs either from the src folder inside the SyncPy installation folder or shortcuts on your desktop if you are on Windows/Mac, or in your home folder if you are on Linux, if you installed SyncPy with the installer.

Part 2 - SyncPy2 UI

1. The main interface



Panel 1 is the log panel. Here you will have important information, messages and results.

Panel **2** is the main panel. Here you will be able to load your data, select the method you want to apply on them, …

Button **1** gives you the possibility to save and load your current work. The file created is a ***.session*** and contains all the information you have entered such as: the input files, the names of the signals, the method selected, …

Button **2** gives you a link to the website and information of the current version of SyncPy you are using.

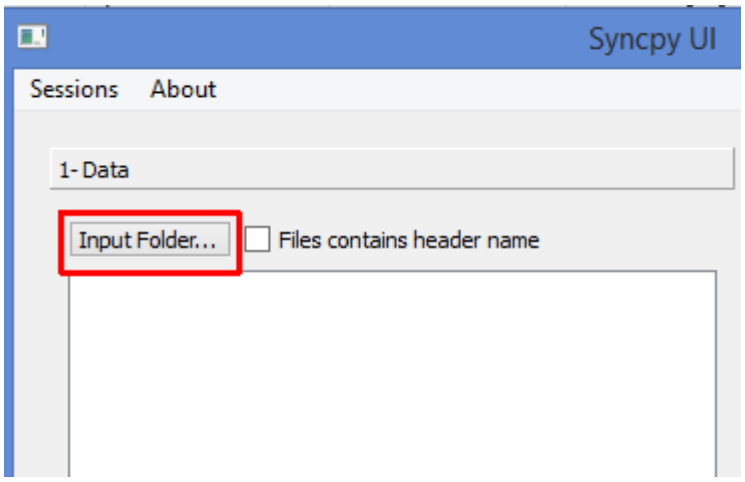
Buttons **3** and **4** are used to apply the selected method on your data and stop the process if you want.

Button **5** opens the out folder where lots files and plotted graph of the method executed.

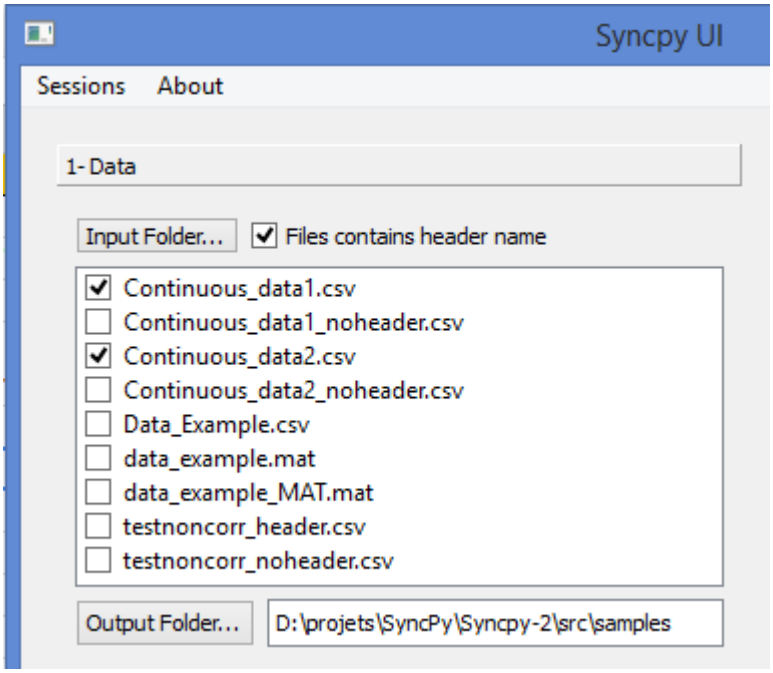
2. The main panel

a. Select your data

First you have to select the folder where your data are stored clicking on the *Input folder* button.

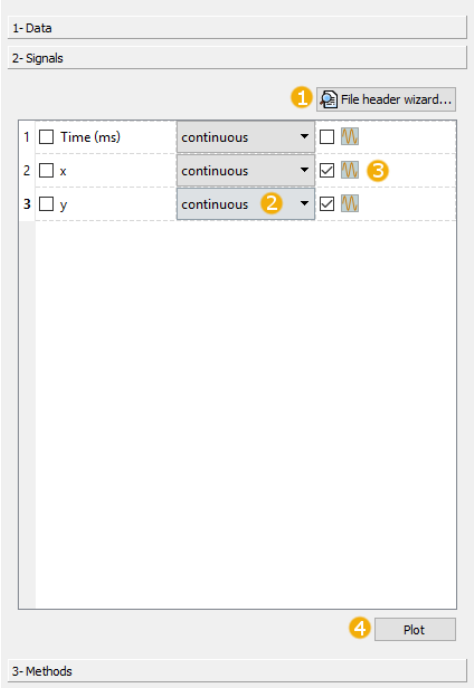


Then you can select the files you want to process. If the file contains a header **you must** check the box *Files contains header name*.



The data files have to be formatted in columns where the first column is the time of your experiment, one file per subject/experiment.

b. Select the signals

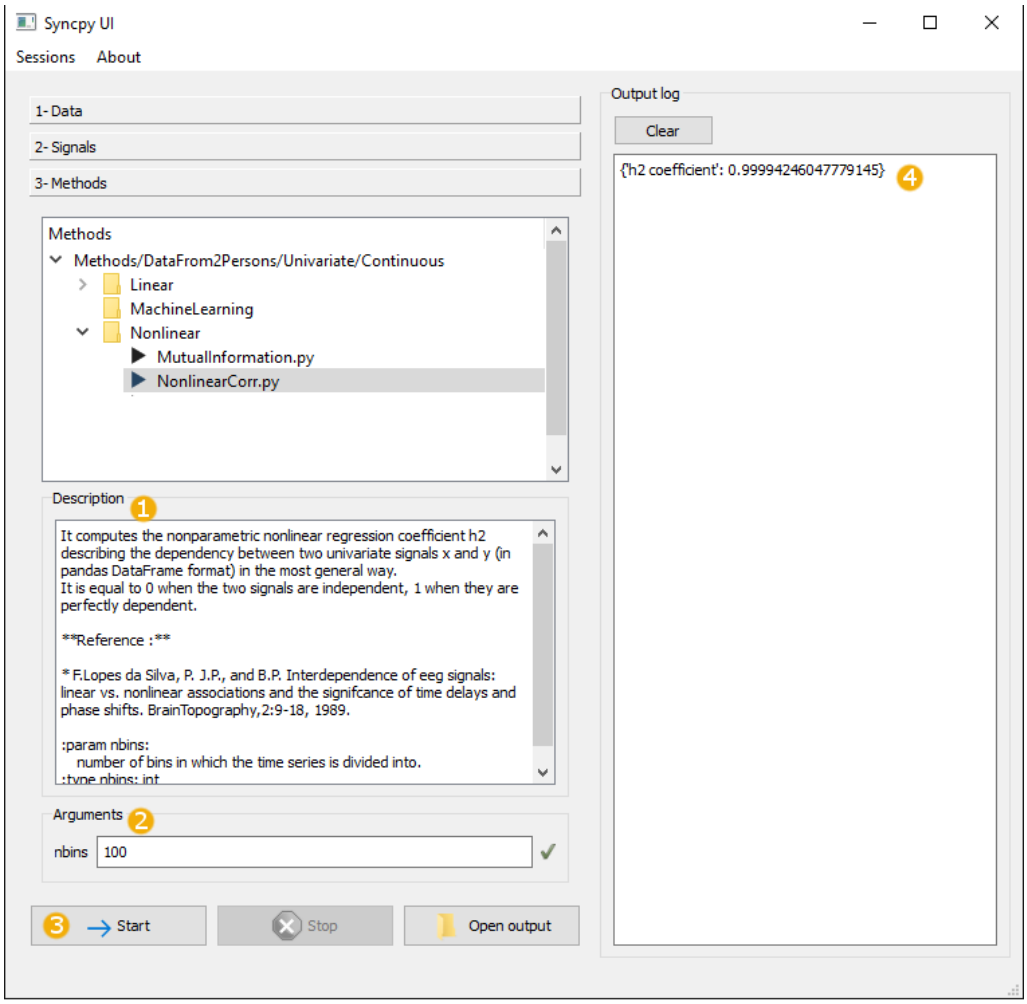


The first time you click on the *Signals* tab the signal header wizard will appear. You can rename your signals or set a name to your signals if you don't have a header in your data file. You can access to this wizard any time by clicking on the *File header wizard* button (1). Remember that the first column must be Time.

The program will detect automatically if the signal is continuous or categorical. If the program is wrong, you can select the type of the signal at any moment (2).

You can also visualize your data by checking the box at the end of the signal line (3) and click on the *Plot* button (4).

c. Process the data

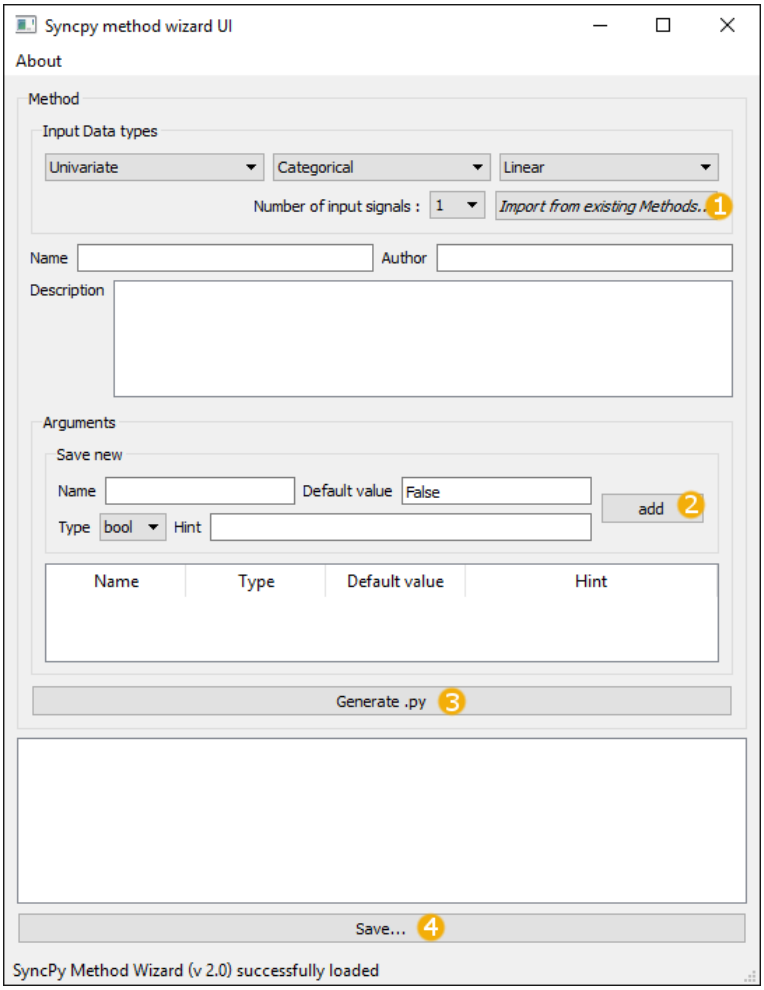


Once you have select the signal to process you can click on the *Method* tab. The program will automatically suggest the methods that you can apply on the signals selected.

Clicking on a method will show its description (1) and display the arguments needed (2). You can then click on the *Start* button (3) to start processing the data. At the end of the processing the output of the method will appear on the log panel (4). Also a file with basics information and the result is created in the output folder.

Part 3 - SyncPy2 method wizard UI

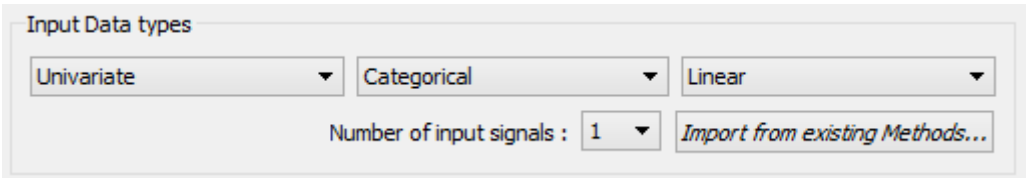
The purpose of this tool is to help to create, from scratch or an example, python coder to implement new Methods that the interface can use to process data.



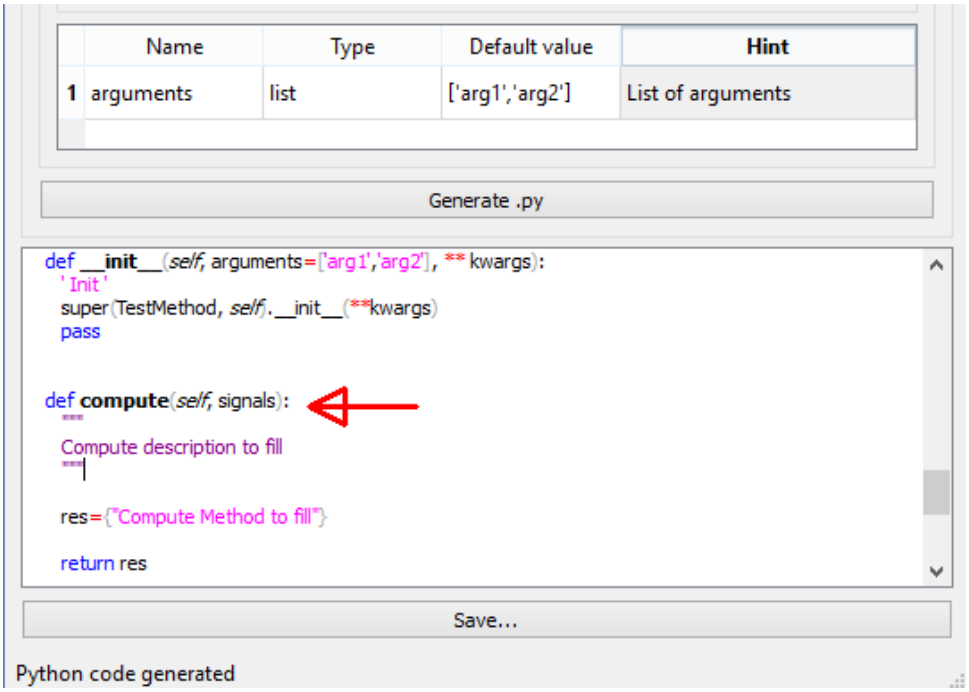
1. Create a method from scratch

a. Choose the input data

First you have to choose on which kind of data the method will work. In the *Input Data types* panel choose the types of the signals, and the number of signals to process.



Or you can use the button to import an existing Method and it will deduce the types for you (see tion do to what you want to do.



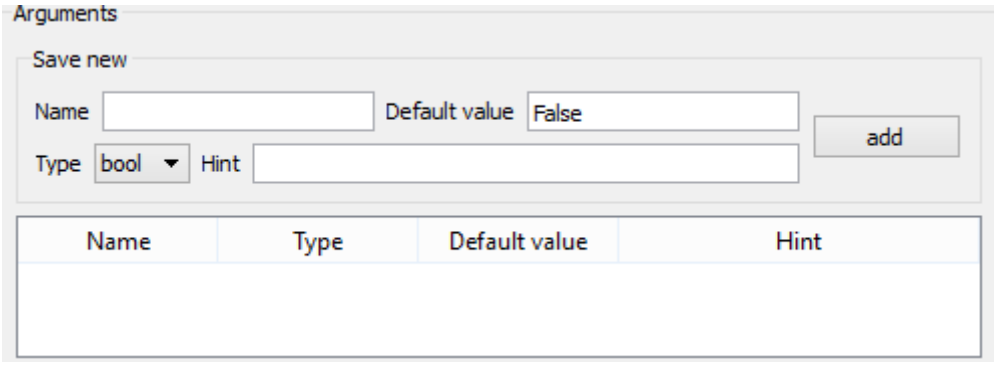
Warning: generating the python file will not save it! To do it click on the *Save* button (4), it will automatically suggest you to save the python file in the right place to be able to use it in the SyncPy UI.

Import an existing method section).

b. Set the main information

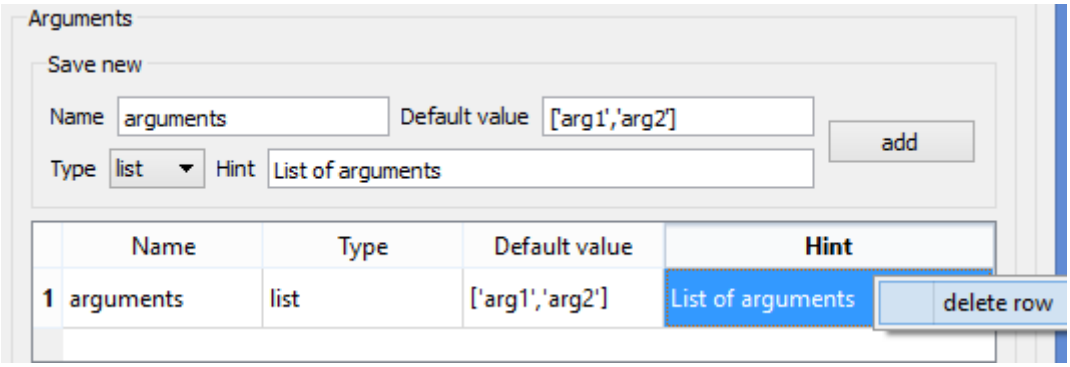
Then you have to choose a name for your method. This will be the name of the generated file and the name displayed in the SyncPy UI. Set the name of the author and a description of what your method is doing.

c. Set the arguments



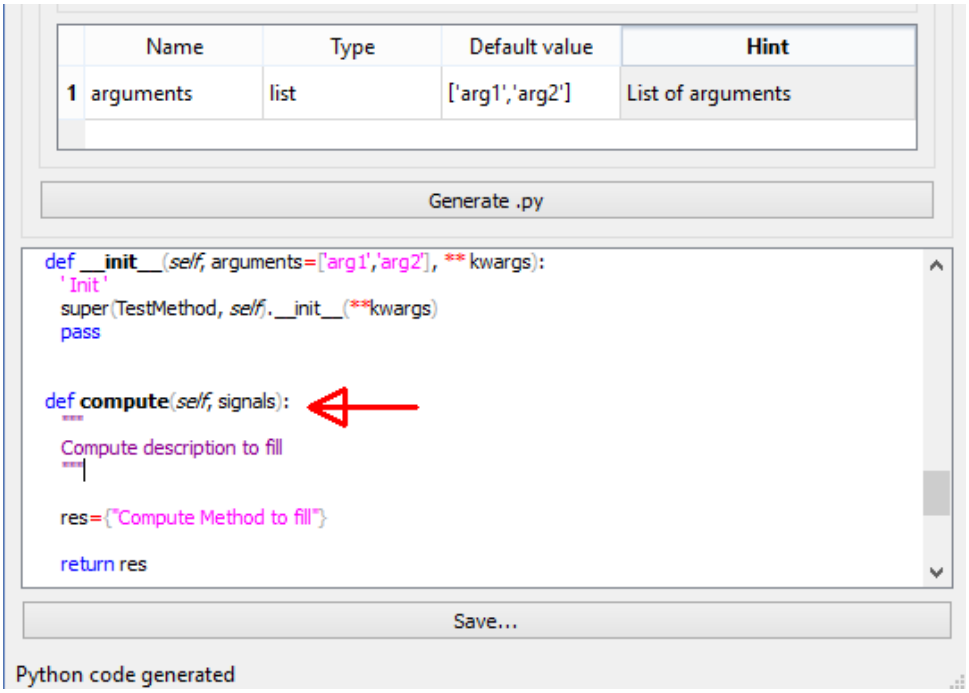
Now you can start to add the arguments of your method. For each argument you **must** specify a name and a default value. For each type of argument, a default value is set to show you the format you must enter. **Warning: if you change the type of the argument you will lose the default value you set!**

If an argument with the same name is already added you can't add it. The hint is optional but useful for the community. Click on the *add* button (2) to add your argument in the arguments list. You can erase at any moment an argument in the list by right clicking on it and choose *delete row*.



d. Generate and save the method

Clicking on the *Generate .py* button (3), the wizard will create your method with all the information you gave. The final step is to edit the *compute* function do to what you want to do.



Warning: generating the python file will not save it! To do it click on the *Save* button (4), it will automatically suggest you to save the python file in the right place to be able to use it in the SyncPy UI.

2. Import an existing method

You can also import an existing method by clicking on the *Import from existing Methods* button (1). Before that you can set the type of the signals and the number of signals, the wizard will only show methods working on the same signals you want to process.

The wizard will fill all the information, add the arguments to the argument list and generate the python file. **Warning: you will have to rewrite the compute method, the wizard will not copy it**.

Be careful to **change the name of your method**, generate the python file and modify the compute function before saving the method.