Isoplot

Release 1.3.1

Loïc LE GREGAM

Dec 07, 2021

USAGE


ISOPLOT

CHAPTER

ONE

WELCOME TO ISOPLOT'S DOCUMENTATION!

Isoplot is a software for the visualisation of MS data from C13 labelling experiments. It takes as input corrected MS data from IsoCor in csv or tsv format. Isoplot outputs static and interactive plots that are then used for quality assessment, biological interpretation, project reports, published papers and so forth.

It is one of the routine tools used on the MetaToul platform.

The code is open-source, and available on GitHub under a GPLv3 license .

Key Features

- **Creation of static and interactive plots** (barplot, heatmap, clustermap, etc…)
- **Command-line interface**
- **Jupyter Notebook graphical interface** using ipywidgets
- **Open-source, free and easy to install** everywhere where Python 3 is available
- **Biologist-friendly**

The documentation relative to Isoplot's installation and usage can be found on ReadTheDocs.

1.1 Quickstart

1.1.1 Installation

Isoplot requires Python 3.7 or higher. If you do not have a Python environment configured on your computer, we recommend that you follow the instructions from Anaconda. Then, open a terminal (e.g. run Anaconda Prompt if you have Anaconda installed) and type :

```
pip install isoplot
```

If you have an old version of the software installed, you can update to the latest using the following command:

```
pip install -U isoplot
```

You are now ready to start Isoplot.

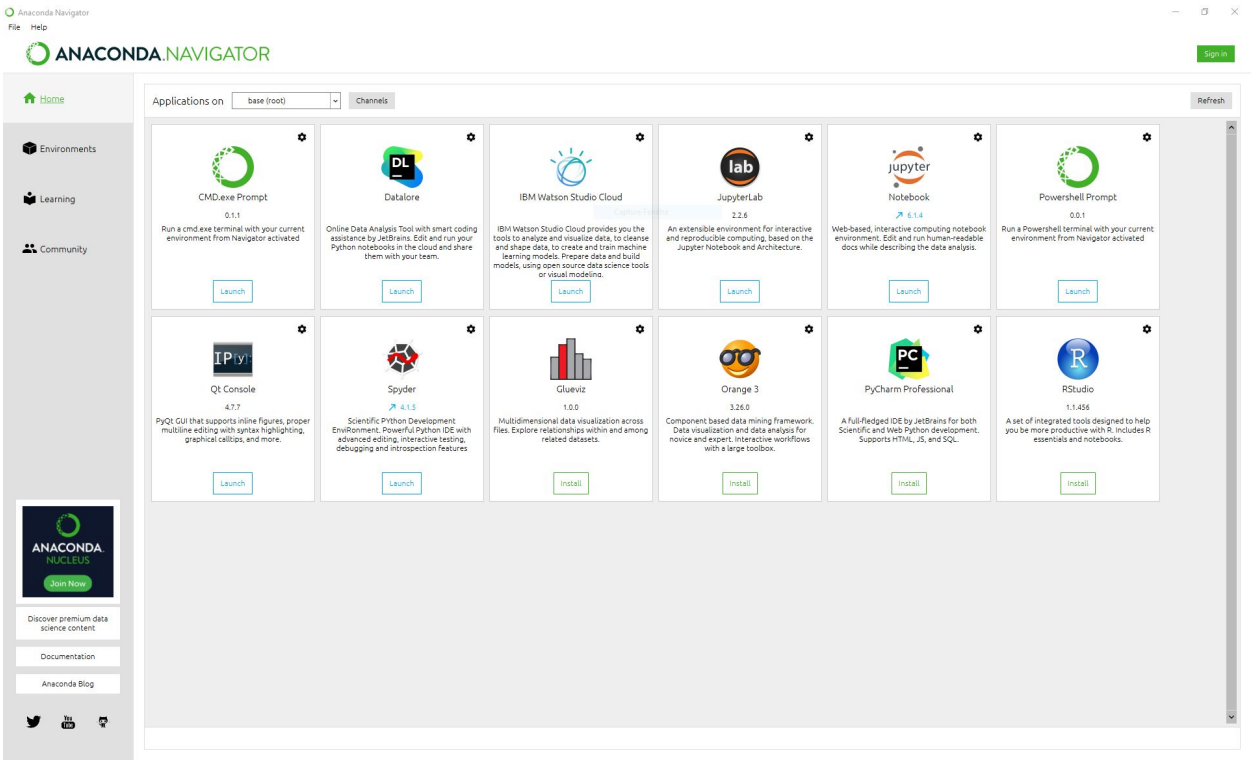
There are two ways to use isoplot : through the dedicated jupyter notebook (recommended) or through the command line.

1.2 Environment installation

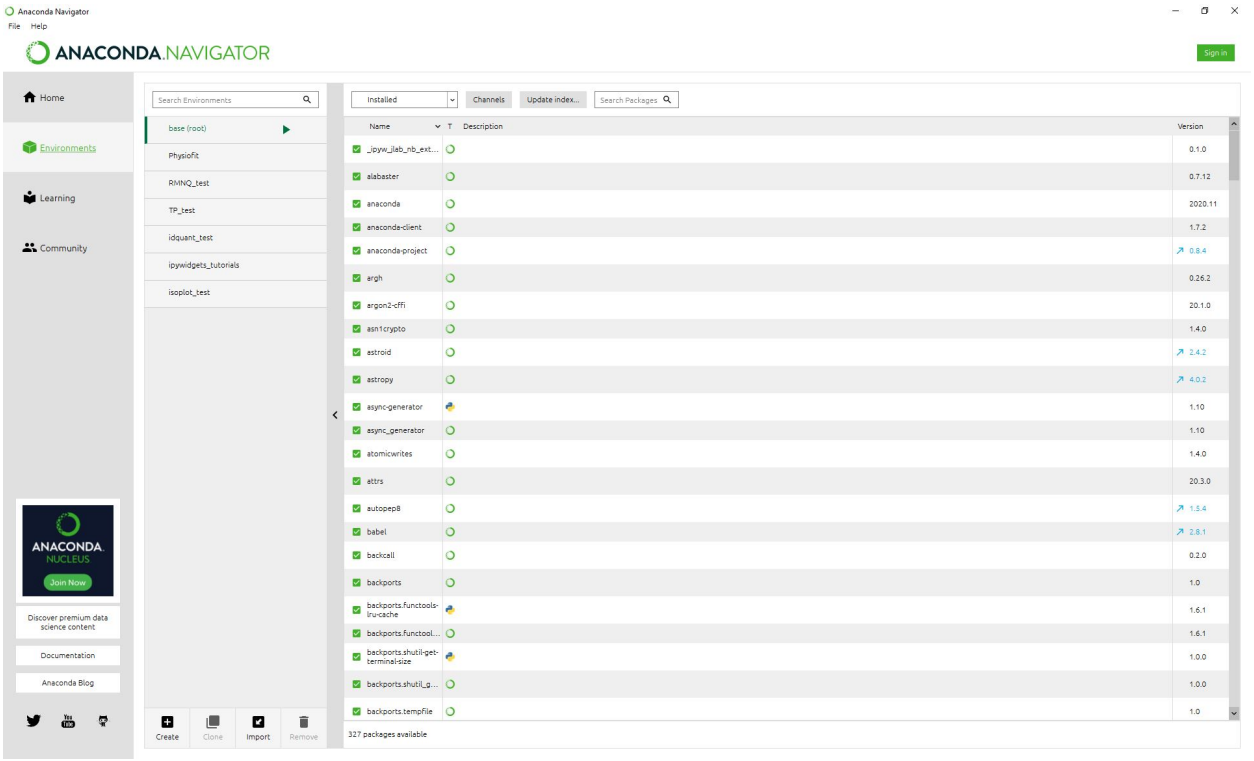
One of the advantages of the Anaconda Suite is that it gives access to a user-friendly GUI for the creation and maintenance of python environments. Python environments give the user a way to separate different installations of tools so that different package dependencies do not overlap with each other. This is especially useful if packages share the same dependencies but in different versions. The Anaconda Suite provides a quick and intuitive way of separating these installations.

1.2.1 How to create an environment in Anaconda

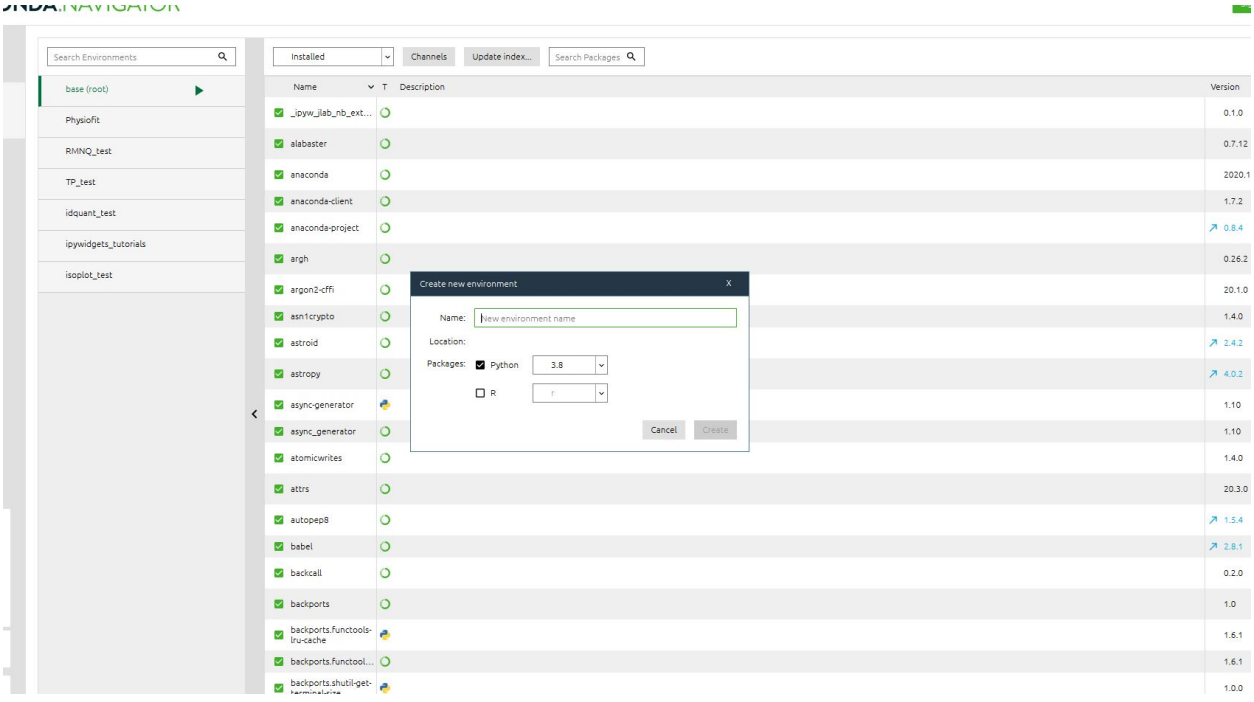
When the user opens up the Anaconda software, she/he ends up on the main menu:



The main window shows all the tools available for installation in the Navigator. To get to the environments page, the user must click on the “Environments” panel that is in the left-side menu.



Once on the Environments page, the user can click on the "create" button that is present at the bottom left of the screen. A pop up menu will then appear and allow the user to select a python version and a name for the environment.



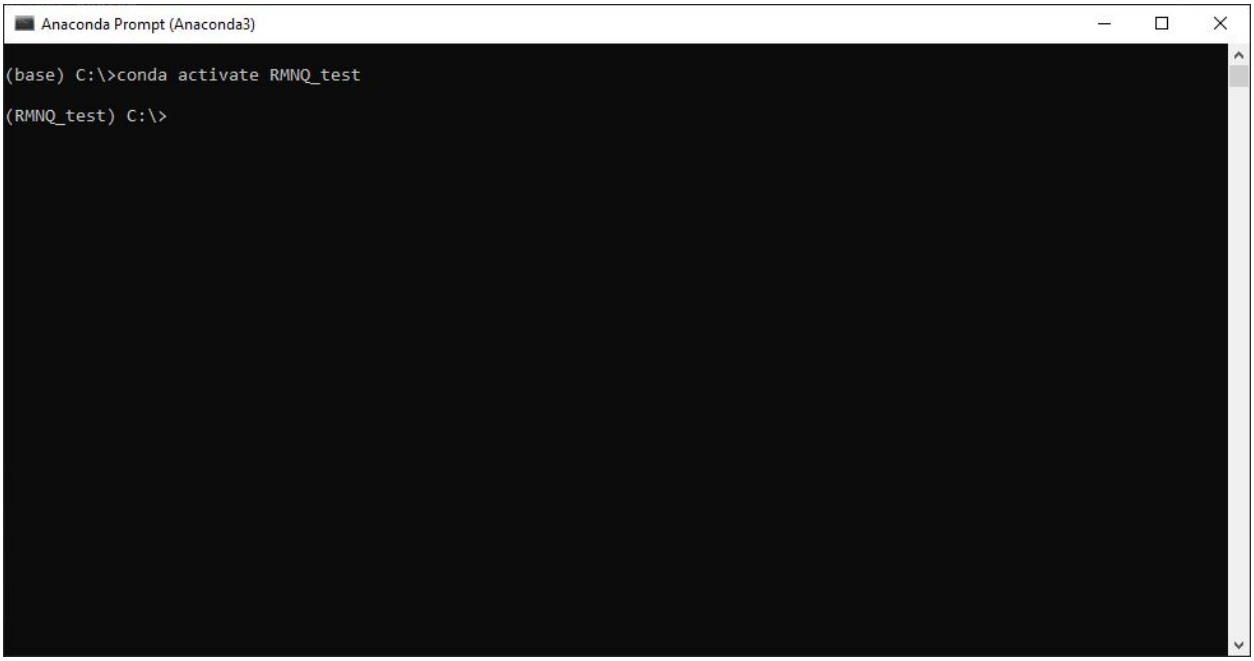
Once the user clicks on the "create" button the environment is created and ready for use!

1.2.2 Installing packages in the environment

Now that the environment exists, it is time to populate it with the tools needed. The first thing to do is to open up a command-line interface, preferably Anaconda Prompt (it is the one that will be used in this tutorial. Other command-line interfaces might use different names for commands). Once the interface is open, the first thing to do is to activate the desired environment. The command for this is as follows:

```
conda activate <name-of-environment>
```

Once this is done the environment name should be seen on the left of the screen behind the name of the directory the interface is open in.



Once the environment is activated, the user can install using pip or conda any of the desired tools. The dependencies and the tool itself will now be installed in a safe and separate set of folders which will ensure that other installations are not affected by anything happening in the environment. Once the user is done, she/he can now close the prompt.

1.2.3 Jupyter Notebook

First install jupyter notebook through the Anaconda Navigator or through the dedicated website and launch the notebook.

Navigate to the **Isoplot.ipynb** file that you can download from the Github.

Launch the first cell to initiate the **« upload data »** and **« submit data »** buttons and use them to load in the tsv or csv IsoCor output file and generate the **template file** (ModifyThis.xlsx)

Modify the template as needed, save it and load it into the notebook after launching the second cell and initiating the **« Upload template »** and **« Submit template »** buttons.

Launch the next cells and generate plots !

Note: For more information on how to setup a python tool in a specific environment (recommended) using jupyter notebooks, check out this documentation.

1.2.4 Command-line interface

To process your data, type in a terminal :

```
isoplot [command line options]
```

Here after the available options are enumerated and detailed.

```
usage: isoplot [-h] --value
               [{corrected_area,isotopologue_fraction,mean_enrichment} [{corrected_area,
↪isotopologue_fraction,mean_enrichment} ...]]
               [-m METABOLITE] [-c CONDITION] [-t TIME] [-gt]
               [-tp TEMPLATE_PATH] [-sa] [-bp] [-mb] [-IB] [-IM] [-IS] [-hm]
               [-cm] [-HM] [-s] [-v] [-a] [-z ZIP] [-g]
               input_path run_name format
```

Positional Arguments

input_path Path to datafile

run_name Name of the current run

format Format of generated file

Named Arguments

--value Possible choices: corrected_area, isotopologue_fraction, mean_enrichment

Select values to plot. This option can be given multiple times

Default: “isotopologue_fraction”

-m, --metabolite Metabolite(s) to plot. For all, type in ‘all’

Default: “all”

-c, --condition Condition(s) to plot. For all, type in ‘all’

Default: “all”

-t, --time Time(s) to plot. For all, type in ‘all’

Default: “all”

-gt, --generate_template Generate the template using datafile metadata

Default: False

-tp, --template_path Path to template file

-sa, --stacked_areaplot Create static stacked areaplot

Default: False

-bp, --barplot Create static barplot

Default: False

-mb, --meaned_barplot Create static barplot with meaned replicates

Default: False

-IB, --interactive_barplot Create interactive stacked barplot

Default: False

-IM, --interactive_meanplot Create interactive stacked barplot with meaned replicates

Default: False

-IS, --interactive_areaplot Create interactive stacked areaplot

Default: False

-hm, --static_heatmap Create a static heatmap using mean enrichment data

Default: False

-cm, --static_clustermap Create a static heatmap with clustering using mean enrichment data

Default: False

-HM, --interactive_heatmap Create interactive heatmap using mean enrichment data

Default: False

-s, --stack Add option if barplots should be unstacked

Default: True

-v, --verbose Turns logger to debug mode

Default: False

-a, --annot Add option if annotations should be added on maps

Default: False

-z, --zip Add option & path to export plots in zip file

-g, --galaxy Option for galaxy integration. Not useful for local usage

Default: False

1.3 Tutorial

1.3.1 Input data

Isoplot takes as input **IsoCor corrected MS data** from **labelling experiments** in **tabular form**. The tabular data must contain a few key columns for each row :

- **« sample »** : Name of the sample containing the analyzed metabolite
- **« metabolite »** : name of the metabolite
- **« isotopologue »** : number of the metabolite's *Isotopologue*
- **« corrected area »** : area of the analyzed metabolite's isotopologue after correction by Isocor
- **« isotopologue fraction »**: see *Isotopologue fraction*.
- **« mean enrichment »** : see *Mean enrichment*.

The input data file is used by Isoplot to generate a **template file** that the user must modify with the desired names for each sample's parameters .

1.3.2 Template file

Once the data has been loaded into Isoplot through the notebook or the command-line, the template is generated by either clicking on the **« submit data »** button in the notebook or by not giving the command-line the template option. The template is a xlsx document that can be opened with excel, and contains a number of columns :

- **Sample** : this columnn corresponds to **the names given to the samples in the original dataset** and **must not be modified**,

or else Isoplot will not be able to pair the template's information with the data.

- **Condition** : The different conditions for each sample
- **Condition order** : Order in the plots of each condition+time. When different replicates are present, **they must be given the same condition order if meaned plots are to be created**. For default order leave 1 on all rows.
- **Time** : The different time points for each sample.
- **Replicate number** : Different replicate numbers for each sample. **They must be numbered starting from 1 to n (n being number of replicates)** for each condition+time.

Note: If for any reason one of the samples must not be plotted or taken into account by Isoplot, remove the associated line from the template file

---

Warning: It is important to properly fill the template table as it is what Isoplot will use to define which data to group together for the plots.

1.3.3 Output files

Once the template is loaded and submited to Isoplot, a Data Export.xlsx file is created with the merged data that is used for the generation of the plots.

If static plots are created, the plots are outputed in the format given by the user (jpeg, png, svg or pdf).

If interactive plots are created, plots are outputed in html format.

1.4 Walkthrough

This part will walk the user through the usage of Isoplot. There are 2 ways to use the tool: through a GUI hosted on a **Jupyter Notebook** or through a **Command Line Interface (CLI)**.

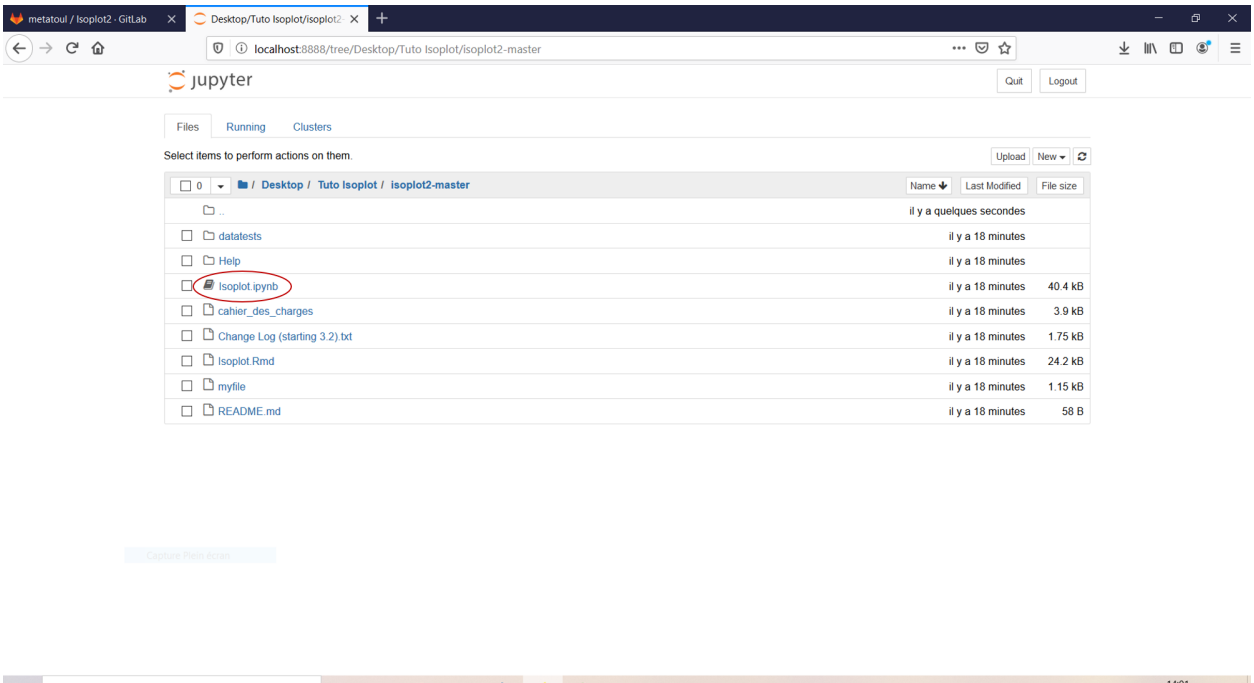
1.4.1 Jupyter Notebook GUI

Once Isoplot is installed and the interactive notebook file **isoplot.ipynb** is downloaded and placed in the desired directory, launch Jupyter Notebook through the **Anaconda Launcher** or through the command line.

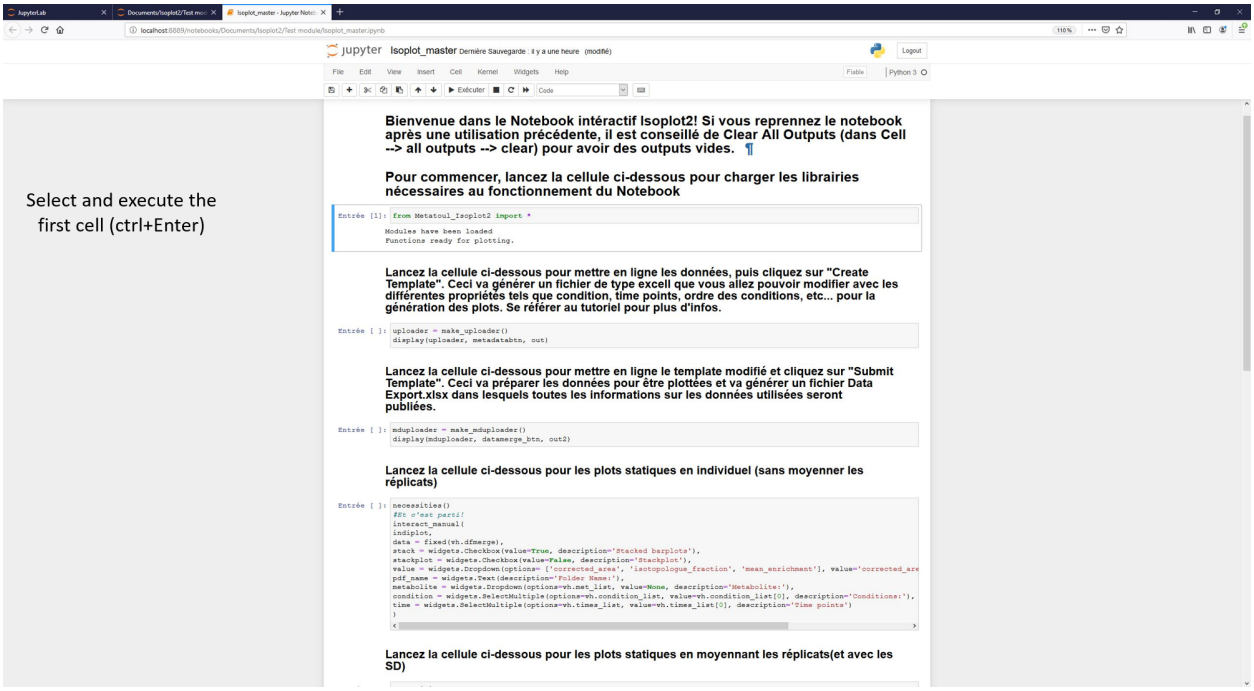
Note: The Isoplot Notebook interface outputs the created directories and files in the directory where the notebook is contained

---

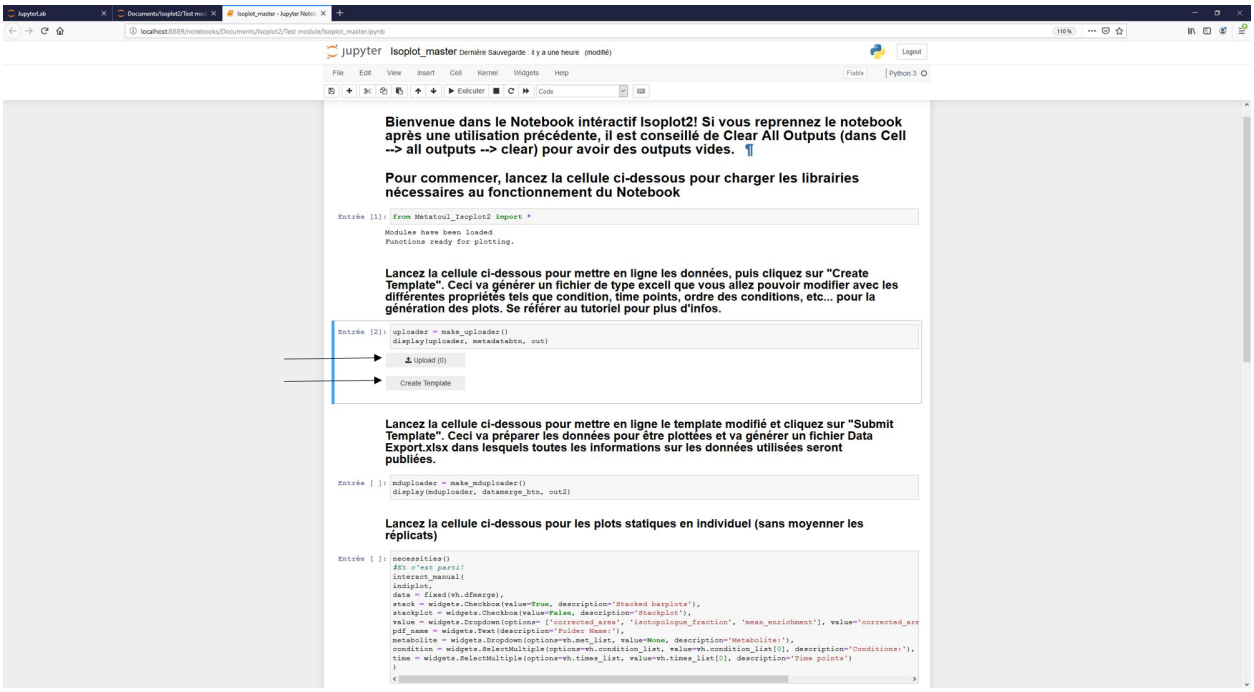
Once in the Notebook, navigate to the appropriate folder and open isoplot.ipynb



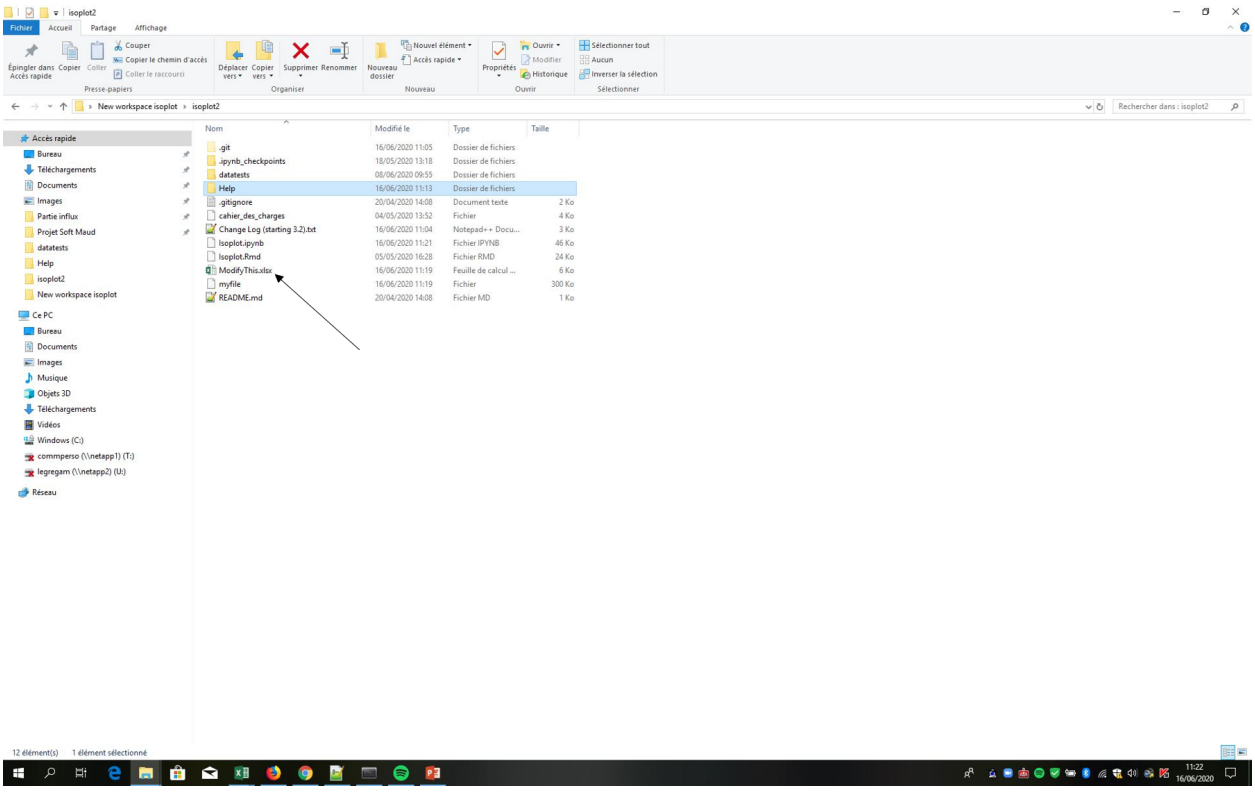
Next, select the first cell and execute it.



Once the buttons have appeared, use the first one to select the **datafile** and then the second one to submit it.

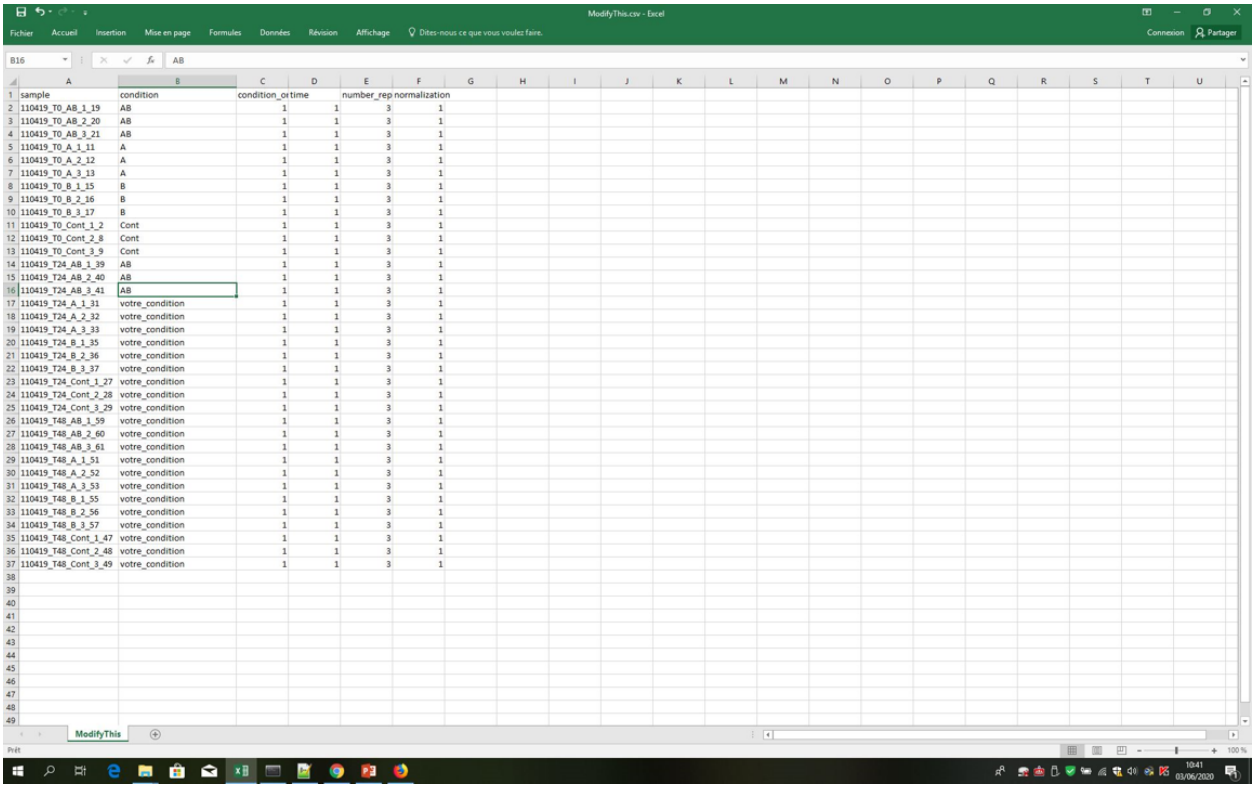


In the file explorer, navigate to the generated **ModifyThis.xlsx** template file and open it.



Fill the file with your conditions, times, replicate numbers, etc… then save it.

> **Warning:** What you input here will be used by Isoplot to generate the groups for plotting (ex: condition AB at T0 replicate 1). **If you have multiple replicates for one same time and condition, number them starting from 1**.

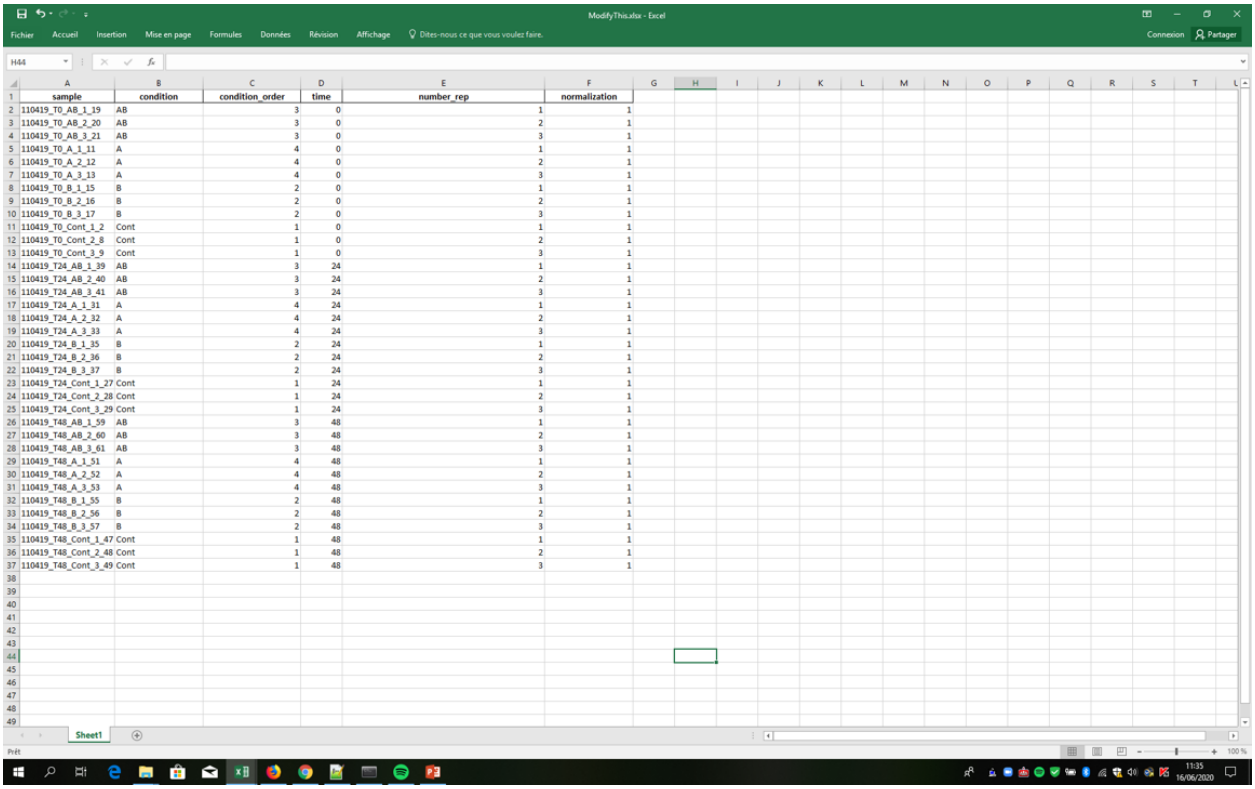


	A	B	C	D	E	F
1	sample	condition	condition_or	time	number_rep	normalization
2	110419_T0_AB_1_19	AB	1	1	3	1
3	110419_T0_AB_2_20	AB	1	1	3	1
4	110419_T0_AB_3_21	AB	1	1	3	1
5	110419_T0_A_1_11	A	1	1	3	1
6	110419_T0_A_2_12	A	1	1	3	1
7	110419_T0_A_3_13	A	1	1	3	1
8	110419_T0_B_1_15	B	1	1	3	1
9	110419_T0_B_2_16	B	1	1	3	1
10	110419_T0_B_3_17	B	1	1	3	1
11	110419_T0_Cont_1_2	Cont	1	1	3	1
12	110419_T0_Cont_2_8	Cont	1	1	3	1
13	110419_T0_Cont_3_9	Cont	1	1	3	1
14	110419_T24_AB_1_39	AB	1	1	3	1
15	110419_T24_AB_2_40	AB	1	1	3	1
16	110419_T24_AB_3_41	AB	1	1	3	1
17	110419_T24_A_1_31	votre_condition	1	1	3	1
18	110419_T24_A_2_32	votre_condition	1	1	3	1
19	110419_T24_A_3_33	votre_condition	1	1	3	1
20	110419_T24_B_1_35	votre_condition	1	1	3	1
21	110419_T24_B_2_36	votre_condition	1	1	3	1
22	110419_T24_B_3_37	votre_condition	1	1	3	1
23	110419_T24_Cont_1_27	votre_condition	1	1	3	1
24	110419_T24_Cont_2_28	votre_condition	1	1	3	1
25	110419_T24_Cont_3_29	votre_condition	1	1	3	1
26	110419_T48_AB_1_59	votre_condition	1	1	3	1
27	110419_T48_AB_2_60	votre_condition	1	1	3	1
28	110419_T48_AB_3_61	votre_condition	1	1	3	1
29	110419_T48_A_1_51	votre_condition	1	1	3	1
30	110419_T48_A_2_52	votre_condition	1	1	3	1
31	110419_T48_A_3_53	votre_condition	1	1	3	1
32	110419_T48_B_1_55	votre_condition	1	1	3	1
33	110419_T48_B_2_56	votre_condition	1	1	3	1
34	110419_T48_B_3_57	votre_condition	1	1	3	1
35	110419_T48_Cont_1_47	votre_condition	1	1	3	1
36	110419_T48_Cont_2_48	votre_condition	1	1	3	1
37	110419_T48_Cont_3_49	votre_condition	1	1	3	1

The ModifyThis.xlsx file contains a table with 5 columns. They are detailed in *Tutorial - Template file*. Once you have finished modifying the template, save it.

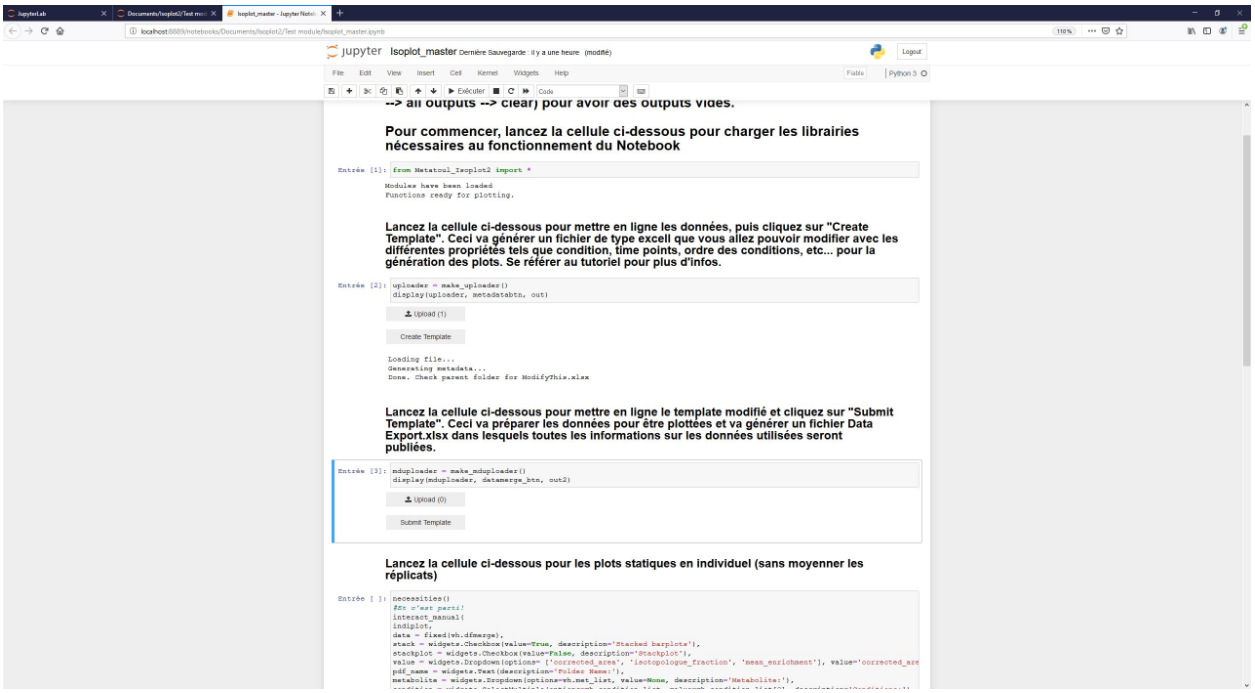
Note: It is good practice to change the name of the template file because if you press the "create template" button once more Isoplot will overwrite the old ModifyThis.xlsx file.

Example of finished template file:



	A	B	C	D	E	F
1	sample	condition	condition_order	time	number_rep	normalization
2	110419_T0_AB_1_19	AB	3	0	1	1
3	110419_T0_AB_2_20	AB	3	0	2	1
4	110419_T0_AB_3_21	AB	3	0	3	1
5	110419_T0_A_1_11	A	4	0	1	1
6	110419_T0_A_2_12	A	4	0	2	1
7	110419_T0_A_3_13	A	4	0	3	1
8	110419_T0_B_1_15	B	2	0	1	1
9	110419_T0_B_2_16	B	2	0	2	1
10	110419_T0_B_3_17	B	2	0	3	1
11	110419_T0_Cont_1_2	Cont	1	0	1	1
12	110419_T0_Cont_2_8	Cont	1	0	2	1
13	110419_T0_Cont_3_9	Cont	1	0	3	1
14	110419_T24_AB_1_39	AB	3	24	1	1
15	110419_T24_AB_2_40	AB	3	24	2	1
16	110419_T24_AB_3_41	AB	3	24	3	1
17	110419_T24_A_1_31	A	4	24	1	1
18	110419_T24_A_2_32	A	4	24	2	1
19	110419_T24_A_3_33	A	4	24	3	1
20	110419_T24_B_1_35	B	2	24	1	1
21	110419_T24_B_2_36	B	2	24	2	1
22	110419_T24_B_3_37	B	2	24	3	1
23	110419_T24_Cont_1_27	Cont	1	24	1	1
24	110419_T24_Cont_2_28	Cont	1	24	2	1
25	110419_T24_Cont_3_29	Cont	1	24	3	1
26	110419_T48_AB_1_59	AB	3	48	1	1
27	110419_T48_AB_2_60	AB	3	48	2	1
28	110419_T48_AB_3_61	AB	3	48	3	1
29	110419_T48_A_1_51	A	4	48	1	1
30	110419_T48_A_2_52	A	4	48	2	1
31	110419_T48_A_3_53	A	4	48	3	1
32	110419_T48_B_1_55	B	2	48	1	1
33	110419_T48_B_2_56	B	2	48	2	1
34	110419_T48_B_3_57	B	2	48	3	1
35	110419_T48_Cont_1_47	Cont	1	48	1	1
36	110419_T48_Cont_2_48	Cont	1	48	2	1
37	110419_T48_Cont_3_49	Cont	1	48	3	1

Next step is to launch the next cell to generate the buttons to upload and submit the template. Once this is done you can start plotting your figures!



1.5 Definitions

Some important definitions are provided in this section.

1.5.1 Isotopologue

Molecular entity that differs from it's parent molecule in that at least one atom has a different number of neutrons.

1.5.2 Isotopologue fraction

Fraction of an isotopologue relative to the total pool of the metabolite's isotopologues

1.5.3 Mean enrichment

Mean molecular content in isotopic tracer in the metabolite

1.6 Reference

1.6.1 dataprep.py

class isoplot.main.dataprep.**IsoplotData**(*datapath*, *verbose=False*)

Bases: object

Class to prepare Isoplot Data for plotting

Parameters **datapath** (*str*) – Path to .csv file containing Isocor output data

generate_template()

Generate .xlsx template that user must fill

get_data()

Read data from tsv file and store in object data attribute.

get_template(*path*)

Read user-filled template and catch any encoding errors

static load_isocor_data(*path*)

Function to read incoming data

static load_template(*template_input*, *excel_sheet=0*)

Function to read incoming template data

merge_data()

Merge template and data into pandas dataframe

prepare_data(*export=True*)

Final cleaning of data and export

1.6.2 plots.py

Module containing the plotting classes. The methods correspond to different types of plots to create

class isoplot.main.plots.**InteractivePlot**(*stack*, *value*, *data*, *name*, *metabolite*, *condition*, *time*, *display*, *rtrn*)

Bases: *isoplot.main.plots.Plot*

Class to generate the different interactive plots

mean_enrichment_meanplot()
Generate interactive mean_enrichment plots with meaned replicates

mean_enrichment_plot()
Generate interactive mean_enrichment plots

stacked_areaplot()
Generate interactive stacked areaplots

stacked_barplot()
Generate interactive stacked barplots

stacked_meanplot()
Generate interactive stacked barplots with meaned replicates

unstacked_barplot()
Generate interactive unstacked barplots

unstacked_meanplot()
Generate interactive unstacked barplots with meaned replicates

class isoplot.main.plots.**Map**(*data*, *name*, *annot*, *fmt*, *display=False*, *rtrn=False*)

Bases: object

Class to create maps from Isocor output (MS data from C13 labelling experiments)

Parameters **annot** (*Bool*) – Should annotations be apparent on map or not

build_clustermap()
Create a clustermap of mean_enrichment data across all conditions & times & metabolites

build_heatmap()
Create a heatmap of mean_enrichment data across all conditions & times & metabolites

build_interactive_heatmap()
Create an interactive heatmap of mean_enrichment data across all conditions & times & metabolites

class isoplot.main.plots.**Plot**(*stack*, *value*, *data*, *name*, *metabolite*, *condition*, *time*, *display*, *rtrn=False*)

Bases: object

Plot objects Master Class from which the rest inherit

Parameters

- **stack** (*Bool*) – Value to denote if barplots should stack
- **value** (*str*) – Data to be plotted. Can be 'isotopologue_fraction', 'corrected area' or 'mean_enrichment'
- **data** (*Pandas Dataframe*) – IsoplotData object containing clean data
- **name** (*str*) – Name for generated file directory where plots will go
- **metabolite** (*str*) – metabolite to be plotted
- **condition** (*list*) – List of conditions to be plotted

- **time** (*list*) – List of times to be plotted
- **display** (*Bool*) – Should plots be displayed when created
- **rtrn** (*Bool*) – Should figure object be returned or not

HEIGHT = 640

WIDTH = 1080

static split_ids(*ids*)
Function to split IDs and get back lists of conditions, times and replicates in order

Parameters **ids** (*list*) – IDs generated by IsoplotData Object

Returns lists containing conditions, times and replicates

Return type lists

class isoplot.main.plots.**StaticPlot**(*stack*, *value*, *data*, *name*, *metabolite*, *condition*, *time*, *fmt*, *display*, *rtrn*)
Bases: *isoplot.main.plots.Plot*

Class to generate the different static plots.

Parameters

- **fmt** (*str*) – Output format of static plots (pdf, svg, png or jpeg)
- **display** (*Bool*) – Should plots be displayed when created

barplot()
Creation of barplots

mean_barplot()
Creation of meaned barplots (on replicates)

mean_enrichment_meanplot()
Generate static mean_enrichment plots with meaned replicates

mean_enrichment_plot()
Generate static mean_enrichment plots

stacked_areaplot()
Creation of area stackplot (for cinetic data)

1.6.3 isoplotcli.py

Module containing the CLI class that will be used during the cli process to get arguments from user and generate the desired plots

class isoplot.ui.isoplotcli.**IsoplotCli**(*home=None*, *run_home=None*, *static_plot=None*, *int_plot=None*, *maps=None*, *args=None*)
Bases: object

dir_init(*plot_type*)
Initialize directory for plot

static get_cli_input(*arg*, *param*, *data_object*)
Function to get input from user and check for errors in spelling. If an error is detected input is asked once more. This function is used for galaxy implementation

Parameters

- **arg** – list from which strings must be parsed

- **param** (*str*) – name of what we are looking for
- **data_object** (*class: 'isoplot.dataprep.IsoplotData'*) – IsoplotData object containing final clean dataframe

Returns Desired string after parsing

Return type list

go_home()
Exit after work is done

initialize_cli()
Launch argument parsing and perform checks

plot_figs(*metabolite_list*, *data_object*, *build_zip=False*)
Function to control which plot methods are called depending on the arguments that were parsed

Parameters

- **metabolite_list** (*list of str*) – metabolites to be plotted
- **data_object** (*class: 'isoplot.main.dataprep.IsoplotData'*) – object containing the prepared data
- **build_zip** (*bool*) – should figures be returned and exported in zip

zip_export(*figures*, *zip_file_name*)
Function to save figures in figure list to zip file (taken from https://stackoverflow.com/questions/55616877/save-multiple-objects-to-zip-directly-from-memory-in-python)

Parameters

- **figures** (*list of tuples*) – storage of figures and their respective file names in tuples: (name, fig)
- **zip_file_name** (*str*) – name of the exported zip file

isoplot.ui.isoplotcli.**parse_args**()
Parse arguments from user input.

Returns Argument Parser object

Return type class: argparse.ArgumentParser

1.6.4 cli_process.py

Process that runs during Command-Line Interface usage

isoplot.main.cli_process.**main**()

1.6.5 isoplot_notebook.py

class isoplot.ui.isoplot_notebook.**ValueHolder**
Bases: object

x: int = None

isoplot.ui.isoplot_notebook.**build_map**(*data*, *name*, *map_select*, *annot*, *fmt*, *display*)

isoplot.ui.isoplot_notebook.**check_version**(*name*)

isoplot.ui.isoplot_notebook.**dataprep_eventhandler**(*event*)

isoplot.ui.isoplot_notebook.**indibokplot**(*stack*, *value*, *data*, *name*, *metabolites*, *conditions*, *times*, *display*, *stackplot=False*)

isoplot.ui.isoplot_notebook.**indiplot**(*stack*, *value*, *data*, *name*, *metabolites*, *conditions*, *times*, *fmt*, *display*, *stackplot=False*)

isoplot.ui.isoplot_notebook.**make_mduploader**()

isoplot.ui.isoplot_notebook.**make_uploader**()

isoplot.ui.isoplot_notebook.**meanbokplot**(*stack*, *value*, *data*, *name*, *metabolites*, *conditions*, *times*, *display*)

isoplot.ui.isoplot_notebook.**meanplot**(*stack*, *value*, *data*, *name*, *metabolites*, *conditions*, *times*, *fmt*, *display*)

isoplot.ui.isoplot_notebook.**metadatabtn_eventhandler**(*event*)

1.7 FAQ

1. Why are my replicates not being meaned properly?

The most frequent reason for this comes from the numbering of the replicates. Indeed, for each group of same condition+time (example: Wild Type T0) the replicates must be numbered *starting from 1 untill the last* (example: Wild Type T0 1, Wild Type T0 2, Wild Type T0 3, etc…)

2. There was a problem during the experiment for one of my samples. How do I tell Isoplot not to take it into account?

The easiest way to remove any samples from the dataset is not to modify the dataset itself, but rather to remove the corresponding line from the *template file*.

3. Sometimes the generated directory for my plots has multiple sub-directories with the same name. Why is this?

This happens when we generate multiple times the directory by using the same name twice in a row (in the case of an error for example on the first try, or a mistake when selecting data to be plotted). A quick fix is to restart the Kernel or change the name of the directory to be created before generating the plots.

**4. I am trying to create barplots with meaned data (so with error bars) for isotopologue_fractions, but some of my bars seem abherrent. They have error bars that go until under 0 and the total of the fractions is not equal to 1. Why is this?

This happens when data is missing for only some replicates of a given sample condition & time. An ulterior version of Isoplot will fix this by not using data points equal to 0 when calculating the mean.

PYTHON MODULE INDEX

i

INDEX

P

S

U

V

W

X

Z