Testing the HDF5_BLS package for creating BrimX files

Pierre Bouvet

2025

Contents

1 Installation of the HDF5_BLS package

The HDF5_BLS package is a Python package for creating BrimX files. It is compatible with Python 3.10 and above.

We recommend setting up a virtual environment to install the package. This can be done using the following command:

- On Mac terminal

```
python -m venv HDF5_BLS_venv
source HDF5_BLS_venv/bin/activate
```

- On Windows terminal

```
python -m venv HDF5_BLS_venv
HDF5_BLS_venv\Scripts\activate
```

On both OS, the package can be installed using pip:

```
pip install HDF5_BLS
```

2 An example

Here we propose to create a synthetic Brillouin dataset that we'll store in a BrimX file. We will store 3 different types of synthetic data:

- A single spectrum
- A synthetic time evolution
- A spatial mapping in 2D
- A z-stack of a sphere

2.1 Creating the data

We define a DHO function to create the Brillouin spectra.

```
def DHO(nu, nu0, gamma, a, b):
    return b + a * (gamma*nu0)**2/((nu**2-nu0**2)**2+(gamma*nu)**2)
```

We define a frequency axis to create the spectra.

```
nu = np.linspace(-15, 15, 1024)
```

We define the values of shift and linewidth for the different datasets to create:

- For the single spectrum, we set shift to 5 GHz and linewidth to 1 GHz, with an amplitude of 1 and no offset.

```
shift_0D = 5
linewidth_0D = 1
PSD_0D = DHO(nu, shift_0D, linewidth_0D, 1, 0)
```

- For the synthetic time evolution spectrum, we consider a shift rising quadratically from 5 to 6 GHz over time, a linewidth raising from 1 to 2 GHz linearly, and an amplitude of 1 with no offset.

```
time_1D = np.linspace(0, 10, 100)
shift_1D = 5 + time**2/100
linewidth_1D = np.linspace(1, 2, 100)
for s, l in zip(shift_1D, linewidth_1D):
    PSD_1D = DHO(nu, s, l, 1, 0)
```

- For the synthetic spatial mapping, we consider a diagonal shift gradient from 5 to 6 GHz over time, and a diagonal linewidth gradient from 1 to 2 GHz, and an amplitude of 1 with no offset.

```
gradient_x = np.linspace(0, 1, 50)
gradient_y = np.linspace(0, 1, 50)
temp = (np.outer(gradient_y, gradient_x) + np.outer(gradient_y, gradient_x)
    )/2
shift_2D = temp + 5
linewidth_2D = temp + 1
PSD_2D = np.zeros((50, 50, 1024))
for i in range(50):
    for j in range(50):
        PSD_2D[i, j] = DHO(nu, shift_2D[i, j], linewidth_2D[i, j], 1, 0)
```

- For the synthetic z-stack mapping, we consider a sphere with a shift of 6GHz and linewidth 2GHz in a solution of shift 5GHz and linewidth 1GHz, with an amplitude for all spectra of 1 and an offset of 0. The z-stack is made of 50 slices

```python
x = np.linspace(-1, 1, 50)
y = np.linspace(-1, 1, 50)
z = np.linspace(-1, 1, 50)
shift_3D = np.zeros((50, 50, 50))
linewidth_3D = np.zeros((50, 50, 50))
PSD_3D = np.zeros((50, 50, 50, 1024))
for i in range(50):
    for j in range(50):
        for k in range(50):
            if (x[i]**2 + y[j]**2 + z[k]**2) > 1:
                shift_3D[i, j, k] = 5
                linewidth_3D[i, j, k] = 1
            else:
                shift_3D[i, j, k] = 6
                linewidth_3D[i, j, k] = 2
            PSD_3D[i, j, k] = DHO(nu, shift_3D[i, j, k], linewidth_3D[i, j,
                k], 1, 0)
```

We have provided in the joined Python script, options to visualize the created datasets.

2.2 Adding the data to a BrimX file

Here we're going to structure our file to store the four different datasets we've created. Each dataset will be stored in a separate group, and the group will be named with example names.

First we create the BrimX file at a given path, and define an object "wrp" to interact with it.

```python
wrp = wrapper.Wrapper('path/to/file.h5')
```

For this example, we will structure the file as follows:

```
file.h5
|- Brillouin
|  |- A single spectrum
|  |- A time series
|  |- A mapping
|  |- A z-stack
```

In each of these groups, we will store the corresponding Power spectral density datasets, frequency array and shift and linewidth arrays. To do so, we will use the following code:

```python
# Adding the 0D datasets
wrp.add_frequency(data=nu, parent_group="Brillouin/A single spectrum", name = "
    Frequency")
wrp.add_PSD(data=PSD_0D, parent_group="Brillouin/A single spectrum", name = "PSD
    ")
wrp.add_treated_data(parent_group="Brillouin/A single spectrum", name_group = "
    Treated Data", shift = shift_0D, linewidth = linewidth_0D)

# Adding the 1D datasets
wrp.add_frequency(data=nu, parent_group="Brillouin/A time series", name = "
    Frequency")
wrp.add_PSD(data=PSD_1D, parent_group="Brillouin/A time series", name = "PSD")
wrp.add_treated_data(parent_group="Brillouin/A time series", name_group = "
    Treated Data", shift = shift_1D, linewidth = linewidth_1D)

# Adding the 2D datasets
wrp.add_frequency(data=nu, parent_group="Brillouin/A mapping", name = "Frequency
    ")
wrp.add_PSD(data=PSD_2D, parent_group="Brillouin/A mapping", name = "PSD")
wrp.add_treated_data(parent_group="Brillouin/A mapping", name_group = "Treated
    Data", shift = shift_2D, linewidth = linewidth_2D)

```

```
# Adding the 3D datasets
wrp.add_frequency(data=nu, parent_group="Brillouin/A z-stack", name = "Frequency")
wrp.add_PSD(data=PSD_3D, parent_group="Brillouin/A z-stack", name = "PSD")
wrp.add_treated_data(parent_group="Brillouin/A z-stack", name_group = "Treated Data", shift = shift_3D, linewidth = linewidth_3D)
```

You are now the proud owner of a BrimX file containing synthetic Brillouin datasets!

2.3 Adding metadata to the BrimX file

To add metadata to the BrimX file, we will choose the easy way for this example, and just edit the standard spreadsheet given with the package and downloadable at this link: https://github.com/bio-brillouin/HDF5_BLS/blob/main/spreadsheets/attributes_v1.0.xlsx.

A list of established parameters is given, their name, units and definition is written in the different columns. For this example, we are filling the datasheet as presented on figure 1.

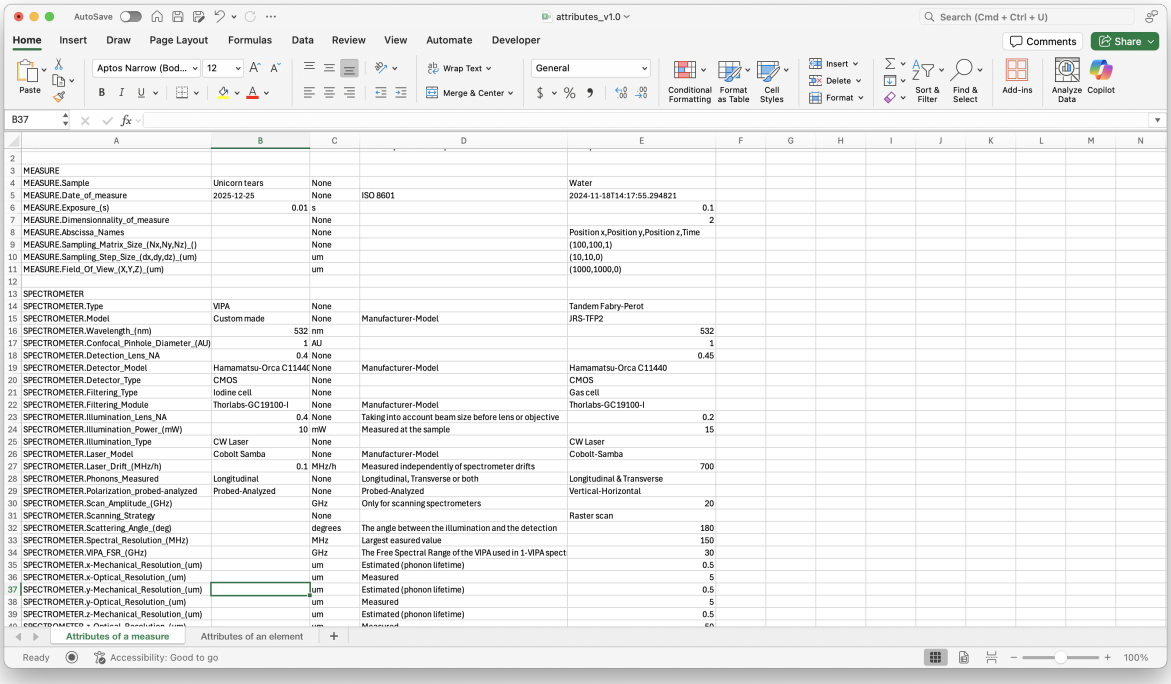

Figure 1: Example of the filled datasheet used to add metadata to the BrimX file

You can of course play with this file, add your own parameters, change the values for the ones given, add values for the ones not given, etc. We recommend however not changing the name of the parameters so that all the community calls the same parameters the same way.

You can then save the file at your prefered location, and apply it to the BrimX file you just created. To do so, you can use the following code:

```
wrp.import_properties_data(filepath='local_use/example_attributes.xlsx')
```

This line of code applies the metadata to the whole BrimX file. You can also apply metadata to a specific group by adding the argument "parent_group" to the function:

```
wrp.import_properties_data(filepath='local_use/example_attributes.xlsx', path = "Brillouin/A z-stack")
```

3 Template for integrating BrimX files to your workflow

You can now try integrating the use of BrimX files in your workflow. Note that the example we showed above is limited to 4 kinds of datasets (frequency, PSD, shift and linewidth). The format can handle a total of 13 different kinds of datasets that are defined in the documentation of the package: https://github.com/bio-brillouin/HDF5_BLS/blob/main/guides/Tutorial/Tutorial.pdf.

Here is a base template to get you started:

```
import HDF5_BLS as bls

# Create a BrimX file
wrp = bls.Wrapper(filepath = "path/to/file.h5")

##############################################################################
# Existing code to extract data from a file
##############################################################################
# Storing the data in the HDF5 file (for this example we use a random array)
data = np.random.random((50, 50, 512))
wrp.add_raw_data(data = data, parent_group = "Brillouin", name = "Raw data")

##############################################################################
# Existing code to convert the data to a PSD
##############################################################################
# Storing the Power Spectral Density in the HDF5 file together with the
    associated frequency array (for this example we use random arrays)
PSD = np.random.random((50, 50, 512))
frequency = np.arange(512)
wrp.add_PSD(data = PSD, parent_group = "Brillouin", name = "Power Spectral
    Density")
wrp.add_frequency(data = frequency, parent_group = "Brillouin", name = "
    Frequency")

##############################################################################
# Existing code to fit the PSD to extract shift and linewidth arrays
##############################################################################
# Storing the Power Spectral Density in the HDF5 file together with the
    associated frequency array (for this example we use random arrays)
shift = np.random.random((50, 50))
linewidth = np.random.random((50, 50))
wrp.add_treated_data(parent_group = "Brillouin", name_group = "Treat_0", shift =
     shift, linewidth = linewidth)
```

4 Extracting data from the BrimX file

You can now interact directly with the BrimX file as any other file storing data. To extract a dataset located at a known path in the file, you can use the following line of code:

```
dataset = wrp["path/to/dataset/in/the/file"][:]
```

For example, using the example BrimX file we created in this tutorial, you can extract the PSD of the first spectrum using the following code:

```
psd = wrp["Brillouin/A single spectrum/PSD"][:]
```

The paths can be obtained by visualizing the file as explained in the next section.

5 Visualizing the BrimX file

5.1 Online

You can visualize the BrimX file online using the myhdf5 web application: https://myhdf5.hdfgroup.org. This application runs locally in your web browser so you can use it safely even to observe sensitive data.

When you open the application, you can upload your BrimX file and explore its contents either by clicking the "Select HDF5 File" button or by dragging and dropping your file in the window (figure 2).

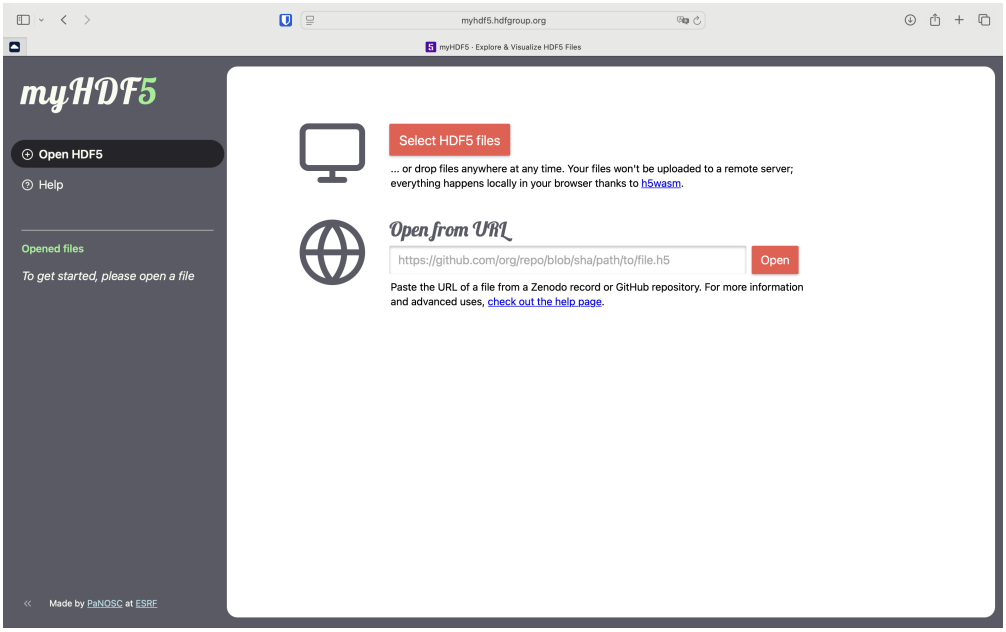

Figure 2: The welcome window of the myhdf5 application

Once your file is loaded, you can explore its contents on the left panel (figure 3).

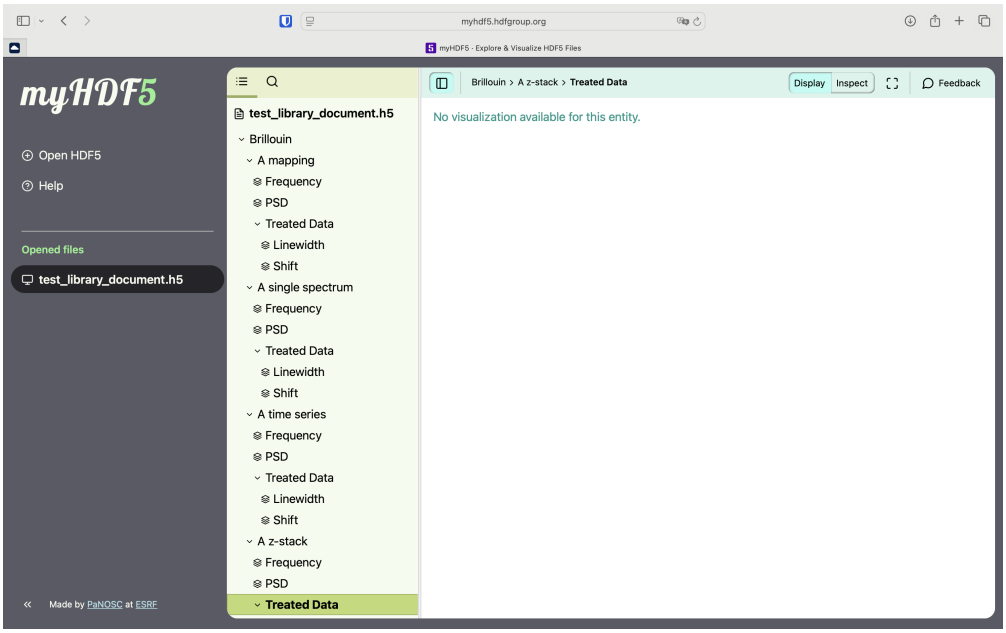

Figure 3: The developed structure view of your HDF5 file in the myhdf5 application

From there you can explore the file, visualize attributes (figure 4) and datasets of any dimension (figure 5).

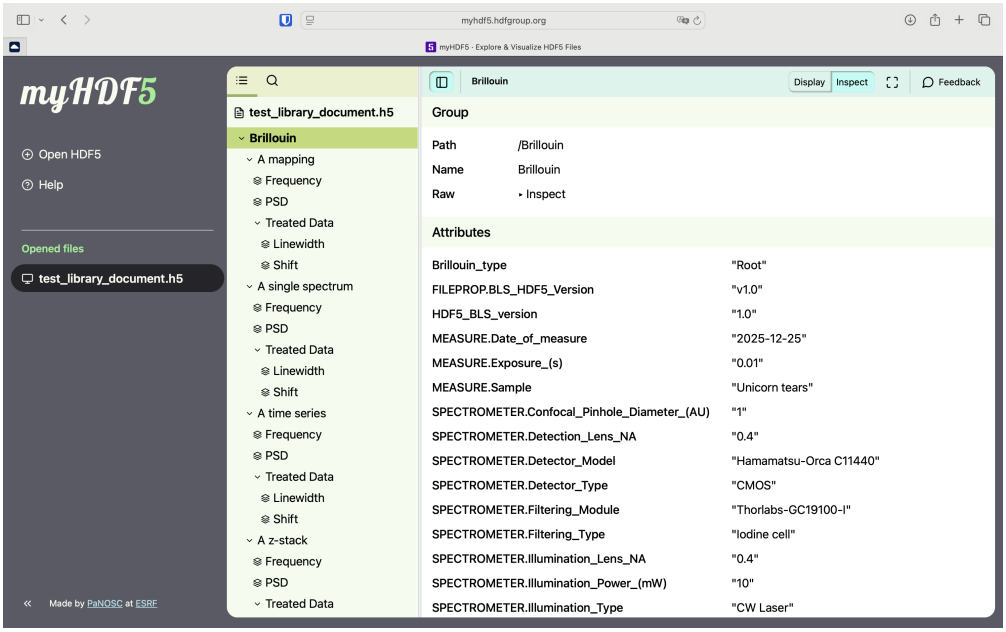

Figure 4: Visualizing the attributes stored in the "Brillouin" group

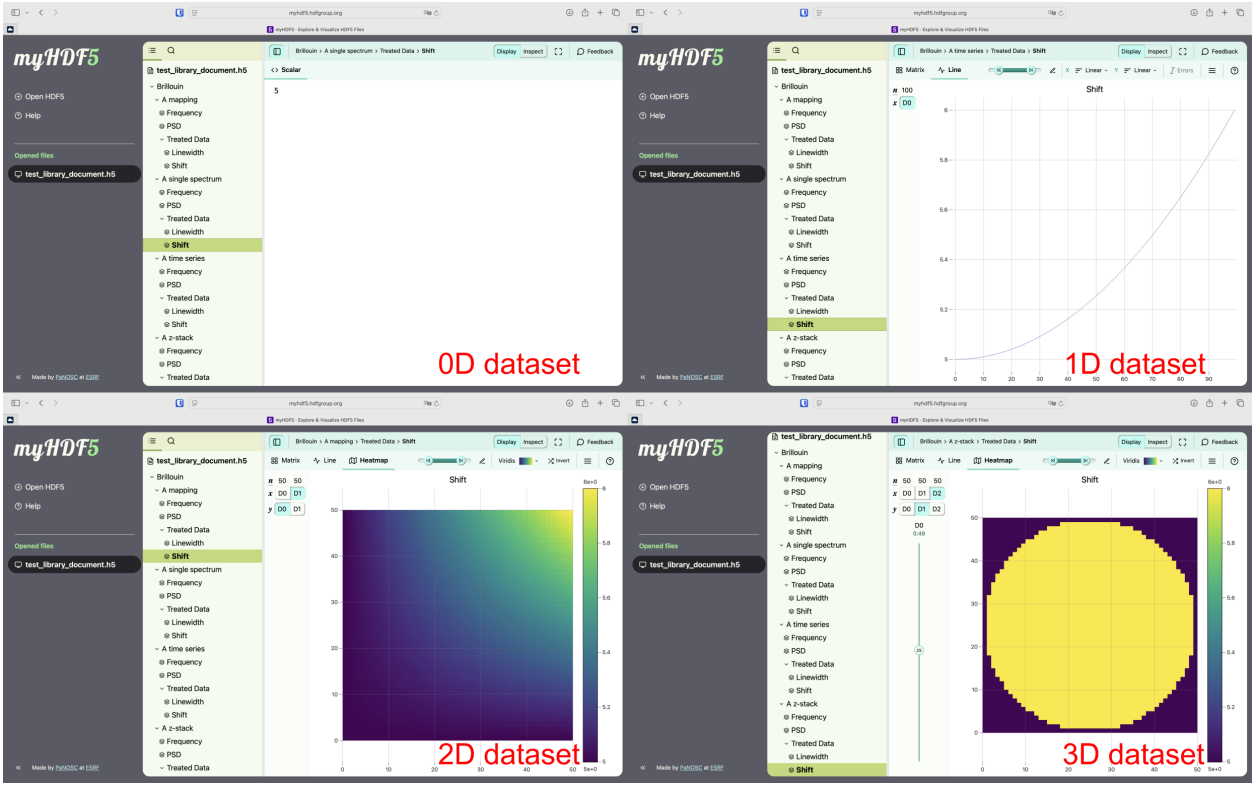

Figure 5: Visualizing the datasets using the MyHDF5 web application

5.2 Using Panoply

Panoply is an application developed by NASA to visualize geo-referenced data and more generally HDF5 files. It can be downloaded for free at this link: https://www.giss.nasa.gov/tools/panoply/.

Here are some screenshots of the application with the example file we created in this tutorial:

Figure 6: A simple visualization of the HDF5 file structure using the Panoply application

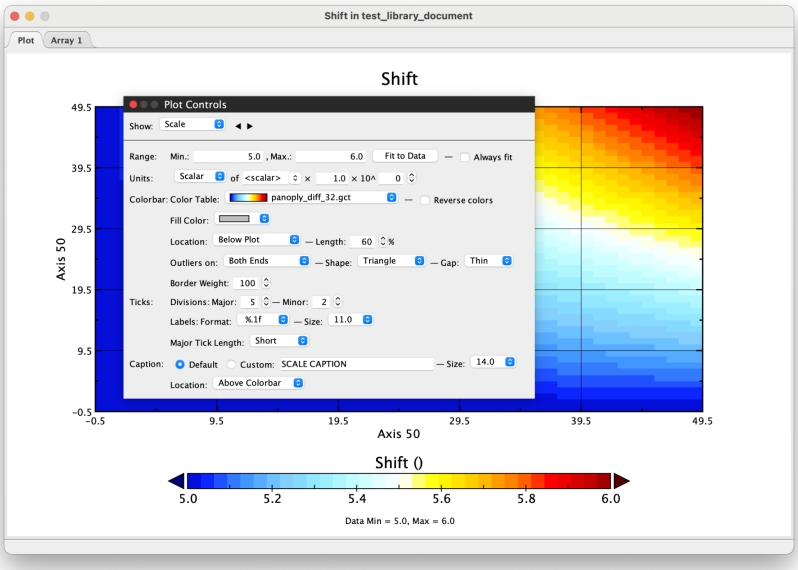

Figure 7: Visualizing datasets using the Panoply application

6 The HDF5_BLS_GUI

The HDF5_BLS_GUI is a graphical user interface for the HDF5_BLS package, designed to facilitate the exploration and analysis of HDF5 files. It provides a user-friendly environment for users to create and interact with their data. The GUI is meant to evolve to allow users to treat their data from it, but these developments are still in a beta phase.

To use the GUI, you need to install the package. Follow the instructions below:

- Clone the repository: `git clone https://github.com/yourusername/HDF5_BLS.git`
- Navigate to the directory: `cd HDF5_BLS`
- Install the packages for the library: `pip install -r requirements_library.txt`
- Install the packages for the GUI: `pip install -r requirements_GUI.txt`
- Run the GUI: `python HDF5_BLS_GUI/main.py`

You should see the HDF5_BLS_GUI window appear on your screen (figure 8).

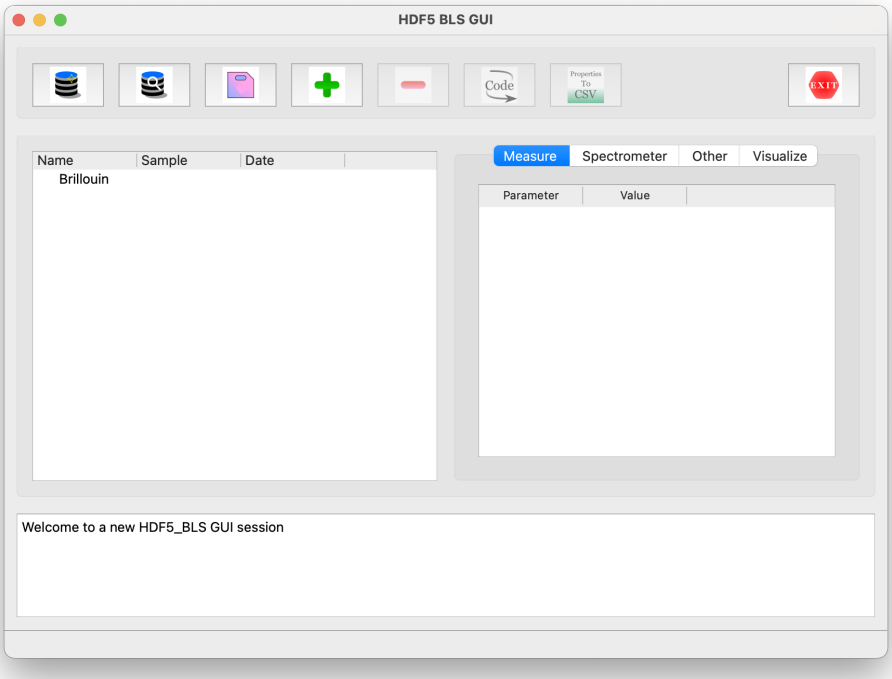

Figure 8: The welcome window of the HDF5_BLS_GUI

You can then drag and drop your data into the left panel if its addition is already supported by the GUI. If not, you'll be forced to add it from script.

You can then structure your file by creating groups, renaming them, dragging files from groups to groups, and generally use the GUI as you would a normal file explorer.

When adding a data, the GUI automatically creates a new group in the HDF5 file. You can change the name of the group by double clicking on it. You can also drag excel files with properties to the right panel to apply metadata to groups.

You can of course drag and drop the BrimX file we have created in this tutorial to the left panel to see its structure (figure 9).

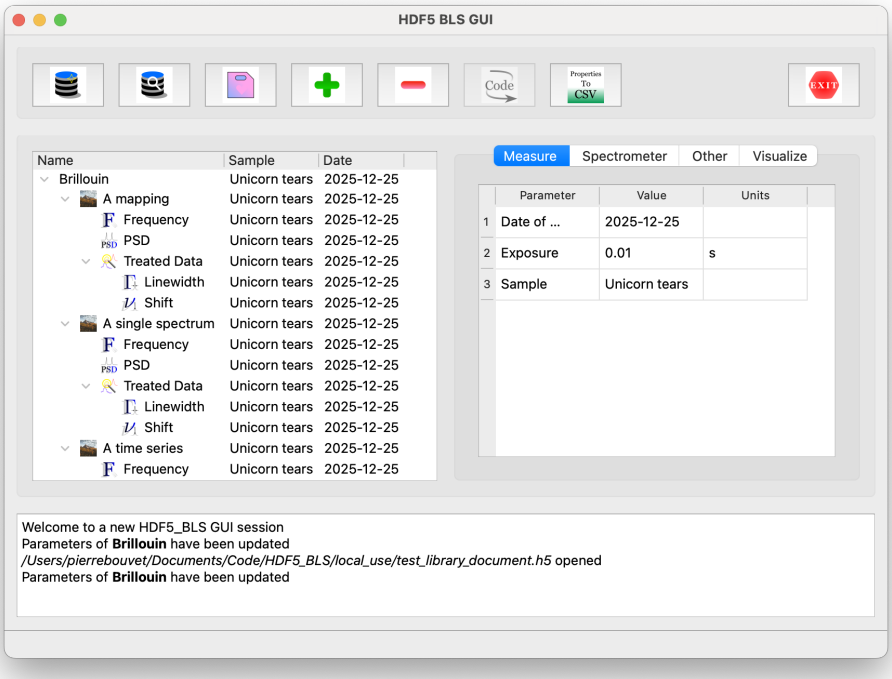

Figure 9: The structure of the example BrimX file in the HDF5_BLS_GUI

This GUI is still in development, so don't hesitate to report bugs or suggest features on the GitHub repository: https://github.com/bio-brillouin/HDF5_BLS/issues.

7 Ending remarks

In this tutorial, we have covered the basics of using the HDF5_BLS_GUI for exploring and managing HDF5 files. We encourage you to experiment with the GUI and provide feedback to help us improve it. You can directly contact me at pierre.bouvet@meduniwien.ac.at !

Thank you for your help! Happy Brillouin data handling!