Project presentation

Pierre Bouvet

February 10, 2025

1 Introduction

This document is made as to present the vision of the HDF_BLS library and the software approach for the unification of all BLS data. This file is meant to be evolutive with the different needs encountered by researchers. It however presents succintly the current vision of the project and allows to better understand its development and its parts.

2 HDF5 file structure

2.1 Multiple measures of same nature

The vision of the project is to unify all BLS data into a single HDF5 file. This file will be used to store all the data and metadata of the BLS. The data will be stored in a hierarchical structure, with the root of the file being the "Data" group. This group will contain all the data. These data can either be arranged in groups, whose identifiers will be "Data_i" or directly as numerical arrays. We propose to only allow one measure per group or file, and to call the array corresponding to the raw data recovered from the spectrometer "Raw_data". For multiple measures, we propose to create multiple groups, following this structure:

```
file.h5
+-- Data
|   +-- Data_1
|   |   +-- Raw_data
|   +-- Data_2
|   |   +-- Raw_data
|   +-- ...
```

2.2 Attributes

The attributes will follow a hierarchical structure. The attributes that apply to all "Data_i" will be stored in the "Data" group, while the parameters that apply to a specific ”Data_i” will be stored in the ”Data_i” group. All the attributes are stored as text. Attributes are then divided in three categories:

- Attributes that are specific to the spectrometer used, such as the wavelength of the laser, the type of laser, the type of detector, etc. These attributes are recognized by the capital letter word ”SPECTROMETER” in the name of the attribute.
- Attributes that are specific to the sample, such as the date of the measurement, the name of the sample, etc. These attributes are recognized by the capital letter word ”MEASURE” in the name of the attribute.
- Attributes that are specific to the original file format, such as the name of the file, the date of the file, the version of the file, the precision used on the storage of the data, etc. These attributes are recognized by the capital letter word ”FILEPROP” in the name of the attribute.

The name of the attribues contains the unit of the attribute if it has units, in the shape of an underscore followed by the unit in parenthesis. Some parameters that can be represented by a series of norms will also be defined in a given norm, such as the ISO8601 for the date. These norms are however not specified in the name of the attribute. Here are some examples of attributes:

- ”SPECTROMETER.Detector_Type” is the type of the detector used.
- ”MEASURE.Sample” is the name of the sample.
- ”MEASURE.Exposure_(s)” is the exposire of the sample given in seconds
- ”MEASURE.Date_of_measurement” is the date of the measurement.
- ”FILEPROP.Name” is the name of the file.

To unify the name of attributes, we will create and maintain a spreadsheet containing all the attributes and their units. This spreadsheet will be updated as new attributes are added to the project and defined with a version number that will also be stored in the attribtutes of each data attributes. This spreadsheet is meant to be exported in a CSV file that can be used to update the attributes of the data.

2.3 Meta-files

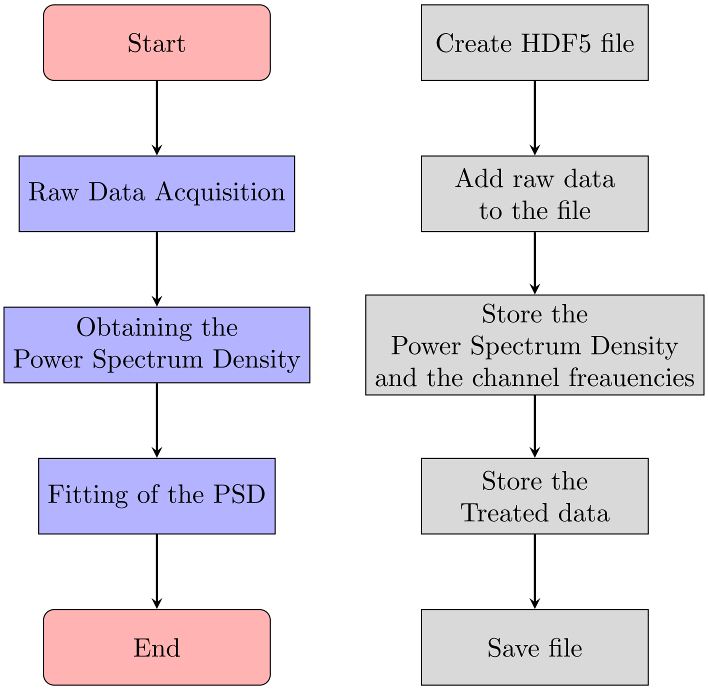
It can be useful to store in a same file, measures coming from different instruments, taken in different conditions, or that we just want to separate from other groups of measures. In that end, we propose a tree-like structure of the HDF5 file, where each group can contain sub-groups, which can also contain sub-groups etc. In order to unify the way we access these groups, we propose to identify them by a unique identifier of the form ”Data_i”, where ”i” is an integer. Here is an example of a meta-file:

```
file.h5
+-- Data
|   +-- Data_1
|   |   +-- Data_1
|   |   |   +-- Raw_data
|   |   +-- Data_2
|   |   |   +-- Raw_data
|   |   +-- ...
|   +-- Data_2
|   |   +-- Data_1
|   |   |   +-- Data_1
|   |   |   |   +-- Raw_data
|   |   |   +-- Data_2
|   |   |   |   +-- Raw_data
|   |   |   +-- ...
|   |   +-- Data_2
|   |   |   +-- Raw_data
|   +-- ...
```

3 Pipelines

3.1 General view

The goal of this software is to go from a storaged data file to a usable data set with a reproducible, stable and unified treatment protocol. To schematize this treatment protocol, we propose the following diagram:



3.2 Conceptual issues and envisionned solutions

3.2.1 Raw Data Acquisition

Raw data will take a variety of form, and will sometimes need to be stored in combination with other data (e.g. array of time points). As such, we propose to store the raw data and all of the data it depends on for its understanding in the same group, naming "Raw Data" the array containing the measure of the energy of the signal, and "Abscissa_i" all the other arrays that are used to understand the data. The resulting will therefore be the following:

```
file.h5
+-- Data
|   +-- Data_1
|   |   +-- Raw_data
|   |   +-- Abscissa_1
|   |   +-- Abscissa_2
|   |   +-- ...
|   +-- ...
```

It is imprecindible therefore to store the technique used to acquire the Brillouin spectrum in the attribute of the file. This information will be stored in the "SPECTROMETER.Type" attribute.

3.2.2 Conversion of the data to a Power Spectrum Density

This is arguably the most important and controversial step of the pipeline. The goal of this step is to obtain the Power Spectrum Density of the data. Many different techniques can be used to do so, most of which will vary from laboratory to laboratory, and relying either on theoretical knowledge of the instrument or data calibration. As it is still unclear how to best convert the data to a PSD, we let the user choose the method they want to use. We propose to store the technique used in the "SPECTROMETER.PSD_Method" attribute, following the following nomenclature: "Laboratory_Method_version". For example, the following attributes could be used:

```
SPECTROMETER.PSD_Method = "MedUniWien_1VIPA_1.0"
SPECTROMETER.PSD_Method = "EMBL_2VIPA_0.5"
SPECTROMETER.PSD_Method = "CellSense_CS363_2.5"
```

Note that this conversion can be performed using an independent and even confidential pipeline, allowing firms to use this format without compromising their proprietary data. In the future, it might however be interesting to have a unified pipeline for all the techniques, which could automatically be adjusted to the "SPECTROMETER.Type" attribute. The end goal of this step is to obtain two arrays:

- The "PSD" array, containing the PSD of the data.
- The "Channel_frequencies" array, containing the channel frequencies of the data.

At this stage, the raw data might become irrelevant and costly in terms of storage space. We can therefore delete the "Raw_data" array and the "Abscissa_i" arrays associated to it at this stage. The resulting file would then take the following form:

```
file.h5
+-- Data
|   +-- Data_1
|   |   +-- PSD
|   |   +-- Channel_frequencies
|   |   +-- ...
|   +-- ...
```

Alternatively, if different treatments are to be compared, we recommend creating different groups. The resulting file would then take the following form:

```
file.h5
+-- Data
```

```
|   +-- Data_1
|   |   +-- Raw_data (optional)
|   |   +-- Abscissa_1 (optional)
|   |   +-- Data_1
|   |   |   +-- PSD
|   |   |   +-- Channel_frequencies
|   |   +-- Data_2
|   |   |   +-- PSD
|   |   |   +-- Channel_frequencies
|   |   +-- ...
|   +-- ...
```

3.3 Fitting of the PSD

The fit of the Power Spectrum Density (PSD) can be unified in a single function that can accept parameters to precise the fit. We encourage you to refer to the dedicated notebook: **Treatment_pipeline** found in the **guides** folder of the directory.

The treatment should result in a new group "Treat_i" composed of at least four arrays:

- The "Shift" array, containing the values of the fitted frequency shifts
- The "Linewidth" array, containing the values of the fitted linewidths
- The "Shift_var" arrays, containing the variance of the fitted frequency shifts, used to auqntify the uncertainty of the fitted values
- The "Linewidth_var" arrays, containing the variance of the fitted linewidths, used to auqntify the uncertainty of the fitted values

Additionnaly, other arrays can be added to the "Treat_i" group, such as the "SNR" and "Amplitude" arrays, or any other relevant information extracted from the PSD.

This treated group should in any case have as attributes all the parameters used in the treatment as well as the different steps followed for the treatment.

The returned structure should therefore be of the form:

```
file.h5
+-- Data
|   +-- Data_1
|   |   +-- PSD
|   |   +-- Channel_frequencies
|   |   +-- Treat_1
|   |   |   +-- Shift
|   |   |   +-- Linewidth
|   |   |   +-- Shift_var
|   |   |   +-- Linewidth_var
|   |   |   +-- ...
|   +-- ...
```

Other treatments on the same PSD can with this architecture be stored in the same file, by changing the index of the "Treat_i" group. For example, if two peaks are to be fitted independently, the resulting file would take the following form:

```
file.h5
+-- Data
|   +-- Data_1
|   |   +-- PSD
|   |   +-- Channel_frequencies
|   |   +-- Treat_1
|   |   |   +-- Shift
|   |   |   +-- Linewidth
|   |   |   +-- Shift_var
|   |   |   +-- Linewidth_var
|   |   |   +-- ...
|   |   +-- Treat_2
|   |   |   +-- Shift
|   |   |   +-- Linewidth
|   |   |   +-- Shift_var
|   |   |   +-- Linewidth_var
|   |   |   +-- ...
|   +-- ...
```