ADVANCED SPECTRAL SOLVER

Author:
Martin Jindra

21.10.2025 (v2.2.0)

Abstract

The Advanced Spectral Solver is an application specifically developed for conducting analyzes of spectroscopic data. This app is equipped with a range of functionalities that not only cover fundamental spectrum analysis but also incorporate a set of advanced capabilities, such as data filtering, visualization, and processing.

Contents

1 Introduction

This open-source Python application is designed to simplify the analysis of spectroscopic data, with a particular focus on multi-peak fitting, batch data processing, and spectroscopy mapping (1D and 2D). Originally developed for academic and research use, the app provides a powerful, user-friendly graphical interface built with Tkinter, enabling scientists and students to interactively explore and interpret spectra without writing code. So far, this app can work with data acquired from HORIBA and Witec spectroscopes, but it can also define your own data file structure with the "User format" setting.
The application supports:

- Importing and visualizing Raman spectra, including 1D and 2D spatial maps (e.g. line scans, hyperspectral images, etc.)
- Spectrum processing features such as minimum or linear subtraction, data filtering, or spectrum subtraction .
- Flexible spectral fitting using standard peak models (Gaussian, Lorentzian, Voigt, etc.) with bounds and constraints
- Exporting and importing defined models significantly reduces the time required to recreate fitting models from scratch for new datasets or similar spectral analyzes.
- Batch fitting routines for consistent analysis of multiple spectra
- Export options for peak parameters, fitted curves, and processed datasets
- Analysis of the Raman map in 2D (spatial) and 1D (time scan, line scan, and electrochemical potential) modes

Whether you're working on a few spectra or hundreds of Raman maps, this app aims to accelerate insights, reduce manual labor, and improve reproducibility in spectroscopic analysis.

2 Main Window

2.1 Environment of the Main Window

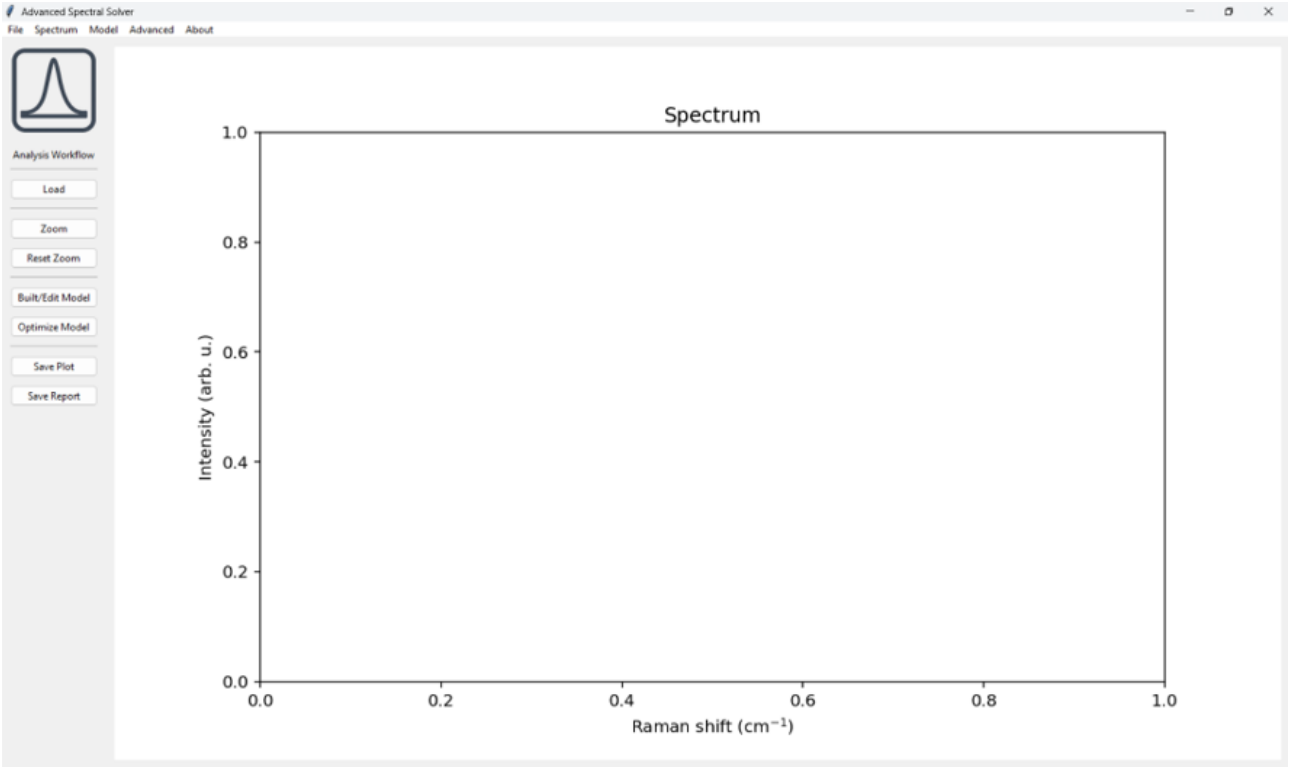

Figure 2.1: Graphical User Interface of the Advanced Spectral Solver

The primary window of this application presents a straightforward and user-friendly workflow. Within the left panel (figure 2.1), there is a button sequence designed to guide you step by step through the essential analysis of the spectrum. These buttons are:

- Load
- ZOOM
- Reset Zoom
- Build/Edit Model
- Optimize Model
- Save Plot
- Save Report

Each of these functions will be described separately.
Another method to access various functions, including the more advanced options, is through the top menu.

2.2 Load - Import of Data

This app is programmed to read `.txt` files exported from the HORIBA and WITec spectrometers. Since the structure of exported files can vary between instruments and measurement modes, import routines are designed to be flexible and adaptable.
To simplify the workflow for repeated analysis or data sharing, the app also supports its own lightweight **Default format**. This format contains no headers or metadata — just raw spectral data — making it easy to export a specific region of interest and re-import it later for further analysis. This approach enables users to focus on the parts of the spectrum that matter most, without the overhead of parsing complex file structures. Additionally, this format is readily interpretable by various software applications such as Excel, Origin, and similar programs.

Note: The "User defined" format allows users to customize and define their own structure of data files. Formats like .txt, .csv, or Excel .xlsx files can be imported with definition separator, skip_rows (to bypass file headers), usecols (to determine which columns are loaded), and file encoding ("ansi" and "utf-8" are the most common). These options can be individually configured through "File/Modify user spectrum details," where you can input your settings and save them using the "Apply" button.

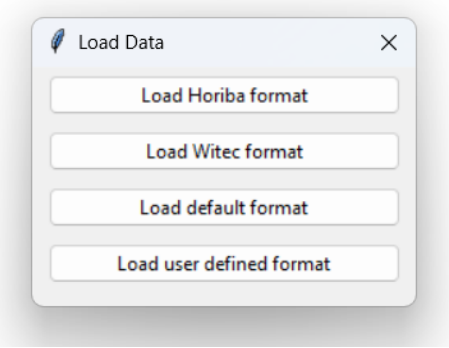

Figure 2.2: Loader window with Horiba, Wited, Default, and User option

Activating the Load button will launch a window that prompts the user to choose the file type to be loaded (HORIBA, WITec, Default, or User defined, figure 2.2). Upon selecting the file type, the user selects the desired file using a file browser. The import process for each file type can be accelerated by using shortcut keys that will open the file browser directly: 'Ctrl+h' for HORIBA, 'Ctrl+w' for WITec, 'Ctrl+d' for Default, and 'Ctrl+u' for User.

An equivalent outcome can be achieved via the "File" button in the top menu(figure 2.1). The only difference is that this way allows toggling the "User format definition" environment.

2.3 ZOOM

Clicking on a ZOOM button will trigger the span selector, enabling the user to choose a range along the Raman shift axis for analysis. Once activated, the user can select the region by holding down the left mouse button and dragging the cursor across the desired Raman shift spectrum. Following the release, the graph will automatically refresh along both the Raman shift axis, as determined by the span selector, and the intensity axis in accordance with the displayed data.
If it is necessary to store data for future reference, you can right-click on the plot and choose the "Save spectrum" option. This action enables you to save the currently displayed segment of the spectrum as a text file in the "default format".

2.4 Reset Zoom

Clicking the "Reset Zoom" button will simply reset the boundaries set by the ZOOM span selector.

2.5 Build/Edit model

Pressing this button opens a new window where users can construct the composite model using predefined profile functions. A new function can be incorporated into the model with the "Add function" button. Subsequently, a new dropdown menu will appear on the right panel of the model builder window. After selecting a function, a function label and an input field for function parameters will be available. The GUI of the model builder can be seen in the figure 2.3.

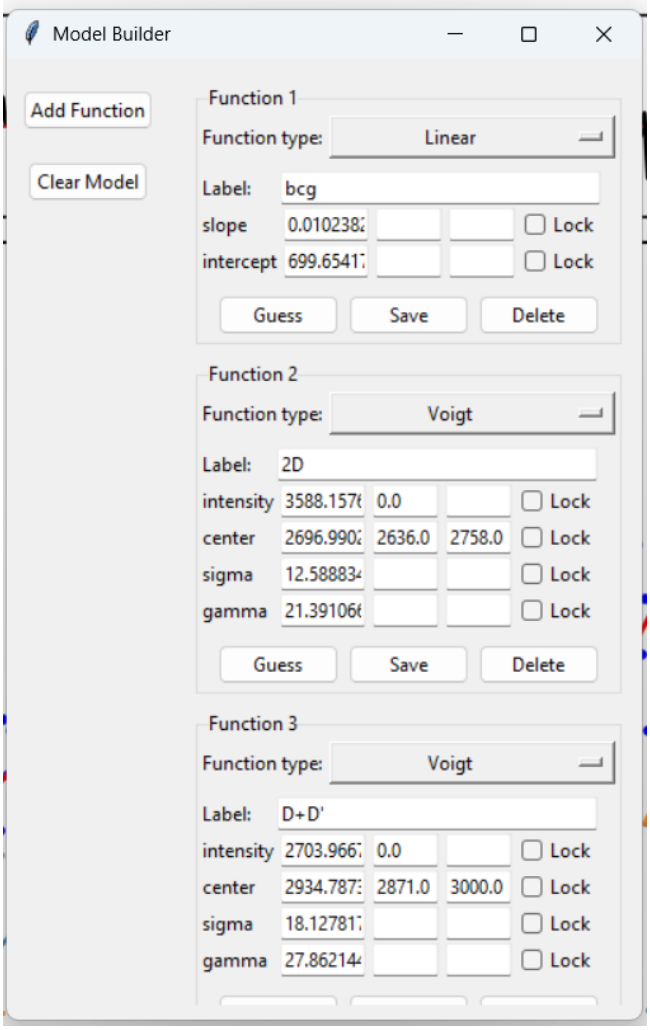

Figure 2.3: Model builder window with already defined functions

While the label is optional, it helps to navigate the composite model, particularly in complex models. Labels support basic mathematical formatting using `matplotlib-style`-style syntax: use underscores for subscripts (e.g., `"G_{2D}"` produces G_{2D}), the circumflex sign (^) for superscripts (e.g., `"I^2"` gives I^2), and prepend a backslash to insert Greek letters (e.g., `"\alpha"` gives α). When using the mathematical format whole label need to be closed between dollar signs, for example: $random label$. By default, the math formatted text is written in italics. if you want to have your label written in normal text, you need to add \mathtm{} at the beginning of the label. All together label can look like $\mathrm{\Gamma_{2D}}$ which will produce label Γ_{2D}.

The first column for parameter entry is for the parameter value, with the next two columns specifying the lower and upper boundaries, preventing the parameter from surpassing these limits during fitting. Users can input these boundaries manually or opt for the "Guess" feature, selecting the spectral region to be described by the chosen profile function with the span

selector. Peak parameter estimation is based on the residual spectrum (raw spectrum minus the defined model), making it crucial to define the background function as an initial step in model building. Following selection, the function values and some boundaries are automatically populated. The peak intensity's lower boundary is set at 0 to avoid fitting negative peaks, and the peak position's boundaries are confined within the selection limits, preventing the peak from shifting outside its designated area. If the estimation is imprecise or requires adjustment, all parameters can be manually modified. It **is** important to note that a function won't appear in the composite model until it is saved using the save button.
When the app hosts an active composite model, it automatically displays the residual spectrum in the upper panel of the app's graph area, while the main plot showcases the raw data alongside the composite model and all individual functions (like in the figure 2.4). All peak functions are charted against the secondary right axis, with the left axis reserved for raw data, the composite model, and background functions (Linear and Sigmoid at present). To eliminate the function from the composite model, you just need to press the "Delete" button.

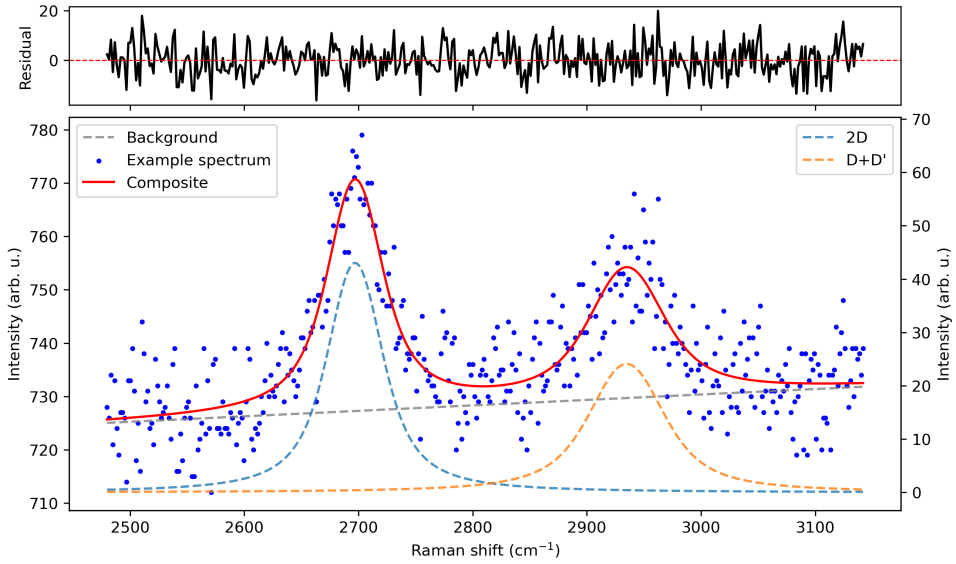

Figure 2.4: Multipeak fit of the defective graphene sample in region of 2D and D+D' band.

The function that has been defined can be saved and retrieved as a .json file. This capability provides an opportunity to reuse models that were previously created, which is particularly useful when conducting analyses on similar datasets. Users can save and access the model through the menu at the top, under the "Model" section.
If a user needs to remove the entire model, for instance, to analyze a different spectral region or a separate dataset, the "Clear model" option will delete all functions defined in the "Model builder window".
A list of the available functions with explanations of their parameters can be find below:

2.5.1 Lorentzian

The Lorentzian function models peaks with long tails, typical of lifetime broadening:

$$L(x) = \frac{I}{\pi \cdot \Gamma \left[1 + \left(\frac{x - x_0}{\Gamma}\right)^2\right]} \tag{2.1}$$

Parameters:

- I – Integrated intensity (area under the curve)
- x_0 – Peak center
- FWHM $= 2\Gamma$ – Full width at half maximum

2.5.2 Gaussian

The Gaussian function is ideal for symmetric peaks influenced by noise or instrument resolution:

$$G(x) = \frac{I}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - x_0)^2}{2\sigma^2}\right) \tag{2.2}$$

Parameters:

- I – Total area under the curve
- x_0 – Peak center (mean)
- FWHM $= 2\sqrt{2\ln 2} \cdot \sigma$

2.5.3 Voigt

The Voigt profile is a convolution of Gaussian and Lorentzian broadening:

$$V(x) = I \cdot \text{Voigt}(x - x_0, \sigma, \gamma) \tag{2.3}$$

Parameters:

- I – Area under the peak
- x_0 – Peak center
- σ – Standard deviation (Gaussian part)
- γ – Half-width at half-maximum (Lorentzian part)

2.5.4 Asymmetrical Lorentzian

This variation introduces asymmetry to the Lorentzian profile:

$$AL(x) = \frac{I}{\pi \cdot \Gamma(x) \left[1 + \left(\frac{x - x_0}{\Gamma(x)}\right)^2\right]}, \quad \Gamma(x) = \frac{\text{FWHM}}{2}(1 + \alpha(x - x_0)) \tag{2.4}$$

Parameters:

- I – Integrated intensity
- x_0 – Peak center
- FWHM – Full width at half maximum for symmetric case
- α – Asymmetry parameter ($\alpha = 0$ gives symmetric Lorentzian)

2.5.5 Fano Resonance

The Fano function describes an asymmetric resonance due to quantum interference:

$$F(x) = I \cdot \frac{(q + \epsilon)^2}{1 + \epsilon^2}, \quad \epsilon = \frac{x - x_0}{\Gamma} \tag{2.5}$$

Parameters:

- I – Area under the resonance
- x_0 – Resonance position
- FWHM $= 2\Gamma$ – Width of the resonance
- q – Asymmetry parameter

2.5.6 Linear

Simple linear trend used for baseline correction:

$$y(x) = m \cdot x + b \tag{2.6}$$

Parameters:

- m – Slope
- b – Intercept

2.5.7 Sigmoid

Smooth transition often used for step-like backgrounds:

$$S(x) = \frac{A}{1 + e^{-(x-x_0)/s}} + B \tag{2.7}$$

Parameters:

- A – Amplitude of the step
- x_0 – Center of the transition
- s – Steepness (lower = sharper)
- B – Baseline offset

2.6 Optimize model

Once the model is defined, you can start the fitting process by clicking the "Optimize" button. This will trigger an optimization procedure that minimizes the sum of squared residuals using a non-linear approach. This is accomplished with the SciPy (scientific Python library) function `curve_fit`, and additional information is available in the official documentation.
After finalizing the fit, users are presented with multiple methods to save the results. These can be accessed by right-clicking on the graph.

- Save plot - saves the visual representation as a .png file
- Save report - generates a .txt file containing the optimized parameters and fitting statistics, which will be explained in detail later
- Save both - concurrently stores both the Plot and Report in the chosen directory
- Save data - exports an Excel file containing the raw data shown, the composite model, and each individual function
- Save spectrum - save the currently displayed spectrum in a default format (no header, just two columns of data) for possible later use

2.6.1 Structure of the report file

The report file contains the values of the optimized parameters, including their errors from the fit. To help evaluate the quality and reliability of the fitted model, the application also computes several statistical metrics based on the residuals between the data and the model.

These include R^2, adjusted R^2, RMSE, AIC, and BIC. Also, the covariance and correlation matrices are included.

Parameter Errors (Uncertainties)

Each fitted parameter is reported with a corresponding standard error, which reflects the uncertainty in its value based on the local curvature of the cost function. Smaller errors indicate higher confidence in the parameter estimate.

Coefficient of Determination (R^2)

$$R^2 = 1 - \frac{\mathrm{SS}_{\mathrm{res}}}{\mathrm{SS}_{\mathrm{tot}}}$$

- $\mathrm{SS}_{\mathrm{res}} = \sum(y_i - f_i)^2$: residual sum of squares.
- $\mathrm{SS}_{\mathrm{tot}} = \sum(y_i - \bar{y})^2$: total variance of the data.

R^2 indicates how well the model explains the variance in the data. A value close to 1 suggests a good fit.

Adjusted R^2

$$R^2_{\mathrm{adj}} = 1 - (1 - R^2) \cdot \frac{n-1}{n-p}$$

- n – Number of data points
- p – Number of fitted parameters

Unlike R^2, the adjusted R^2 accounts for the complexity of the model. It penalizes the use of unnecessary parameters and is better suited for comparing models with different numbers of variables.

Root Mean Square Error (RMSE)

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum(y_i - f_i)^2}$$

RMSE gives an absolute measure of the average deviation between the data and the model. Lower values indicate a better fit.

Akaike Information Criterion (AIC)

$$\mathrm{AIC} = n \cdot \ln\left(\frac{\mathrm{SS}_{\mathrm{res}}}{n}\right) + 2p$$

AIC evaluates model quality by balancing goodness of fit and complexity. Lower AIC values indicate a more parsimonious (efficient) model. Useful for comparing different model structures.

Bayesian Information Criterion (BIC)

$$\mathrm{BIC} = n \cdot \ln\left(\frac{\mathrm{SS}_{\mathrm{res}}}{n}\right) + p \cdot \ln(n)$$

BIC serves a similar purpose to AIC but penalizes model complexity more heavily. Particularly useful when the goal is to identify the simplest adequate model.

Note: AIC and BIC are only meaningful when comparing fits on the same dataset. They are not absolute indicators of fit quality, but they help guide model selection.

Covariance Matrix

The covariance matrix describes how the uncertainties of different parameters are interrelated. For parameters a_i and a_j, the covariance $\mathrm{Cov}(a_i, a_j)$ measures how variations in one parameter correspond to variations in the other.

- **Diagonal elements:** Variances of each parameter, i.e., $\mathrm{Var}(a_i) = \sigma_i^2$.
- **Off-diagonal elements:** Covariances between pairs of parameters.

Large off-diagonal values suggest that the corresponding parameters are not independently determined; a change in one may be partially compensated by a change in another.

Correlation Matrix

The correlation matrix is a normalized version of the covariance matrix. Each element is calculated as:

$$\mathrm{Corr}(a_i, a_j) = \frac{\mathrm{Cov}(a_i, a_j)}{\sigma_i \, \sigma_j}$$

where σ_i and σ_j are the standard errors of parameters a_i and a_j.

- Values range from -1 to $+1$.

- $\mathrm{Corr}(a_i, a_j) = 0$ indicates statistical independence.
- Values close to ± 1 indicate a strong linear dependence between the parameters.

Note: Strong correlations (e.g., $|\mathrm{Corr}| > 0.95$) may indicate that the model is over parameterized or that the parameters are physically redundant.

2.7 Save plot

This button will save currently rendered graphics; everything that the app displays on the main canvas will be saved as a .png picture. This option is also accessible by right-clicking on the canvas and selecting the particular function.

2.8 Save report

This button will save just the report of the fit in .txt format. This option is also accessible by right-clicking on the canvas and selecting the particular function.

3 Spectrum processing

Spectrum tile in the top menu offers various options for spectrum analysis and presentation. The "Update plot" option is primarily for manually refreshing the canvas when it doesn't update automatically, which should be uncommon but possible. If such an issue arises, kindly report it via the "About/Report bug" option or the main GitHub repository.

3.1 Spectrum compare

The spectrum comparison feature primarily serves the visual comparison of different spectra. It allows for comparing spectra measured under different conditions and samples, and even with different spectroscopes, since each spectrum is loaded with its own loader.

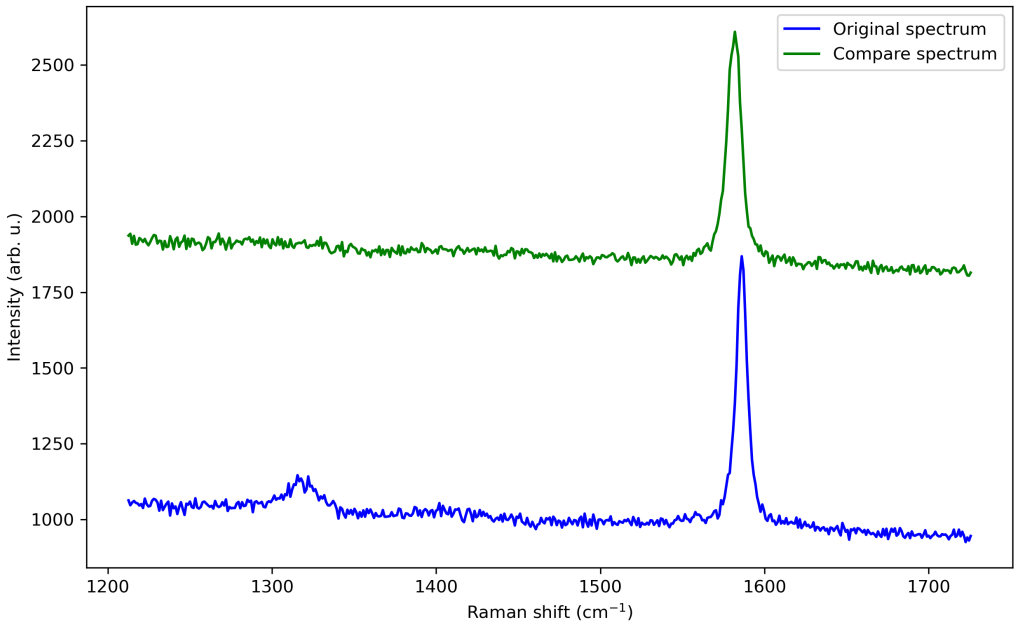

Figure 3.1: Example of spectrum comparison

Activating the "Compare spectrum" option and selecting a file results in a green overlay plot on the main canvas. While the compared spectrum is loaded, all other features remain enabled, except that analysis (aside from data filtering) will focus exclusively on the main spectrum. To remove it, choose the "Delete compare" option. Figure 3.1 illustrates the possible results of the spectrum comparison.

3.2 Signal filtering

To reduce noise and enhance the interpretability of spectral data, the application offers several filtering methods. Each filter has its own characteristics and is suitable for different types of data and noise profiles. For the application of the filter to the data, simply select the type of filter you wish to apply, input the filter parameters, and click "Apply"; this filter will be automatically applied to all data plotted in the Main window. To deactivate the filter, the "Disable" button needs to be clicked.

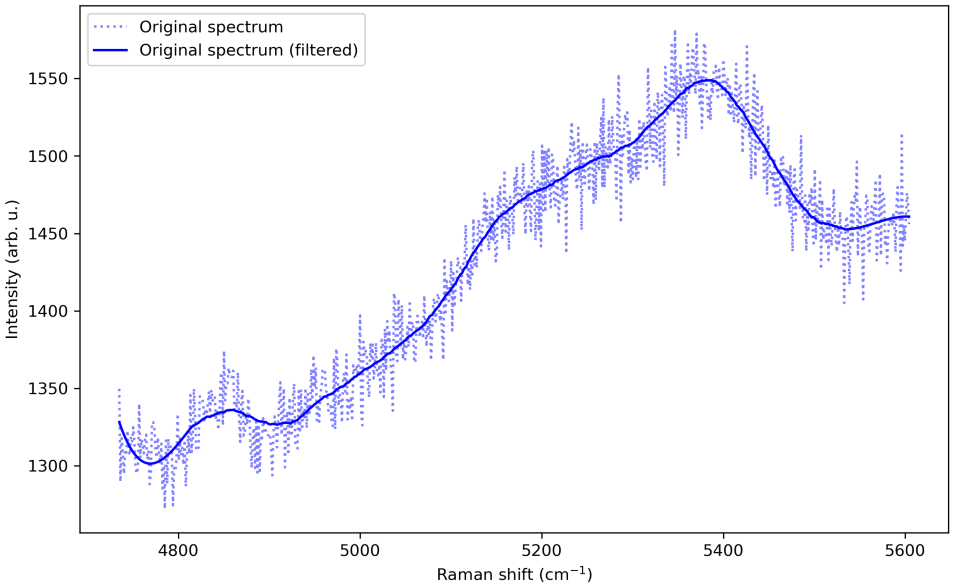

Figure 3.2: Example of Savitzky-Golay filter applied on the noisy data

3.2.1 Savitzky–Golay Filter

The Savitzky–Golay filter smooths data by fitting a low-degree polynomial to a moving window of points using least squares. Unlike a simple moving average, it preserves the original shape and features of the signal, such as peak height and width.

Parameters:

- `window length` – Number of points in the smoothing window. Must be odd and larger than the polynomial order.
- `polyorder` – Order of the polynomial used for fitting (e.g., 2 for quadratic, 3 for cubic).

Pros:

- Retains peak shapes and derivatives.
- Effective in reducing random noise without distorting the signal structure.

Cons:

- Sensitive to window size; too large a window can flatten peaks.
- May introduce artifacts at the edges.

Recommended Use: Ideal for general-purpose smoothing when the signal shape must be preserved (e.g., Raman or IR spectra).
Avoid If: The signal has sharp spikes or highly irregular noise; consider median filtering instead.

3.2.2 Moving Average Filter

The moving average (boxcar) filter replaces each point with the mean of its surrounding neighbors, effectively smoothing high-frequency noise.
Parameters:

- `window_size` – Width of the averaging window in the number of points.

Pros:

- Simple and fast to compute.
- Smooths out random fluctuations effectively.

Cons:

- Can blunt sharp features and reduce resolution.
- Distorts peak height and width.

Recommended Use: Suitable for baseline smoothing or reducing random noise when precise peak shapes are not critical.
Avoid If: The data contains closely spaced or narrow peaks that must be preserved accurately.

3.2.3 Median Filter

The median filter replaces each point with the median of the surrounding window, making it very effective at removing sharp outliers (e.g., cosmic rays or spikes).
Parameters:

- `window_size` – Width of the median window. Must be an odd integer.

Pros:

- Excellent for removing impulsive noise (spikes).
- Preserves edges better than moving average.

Cons:

- Not ideal for reducing continuous Gaussian noise.
- Can distort small peaks if the window is too large.

Recommended Use: Use when data contains discrete spike artifacts or experimental glitches.
Avoid If: Your data has small-amplitude features that could be suppressed by the median operation.

3.2.4 Fourier Low-Pass Filter

This method filters the signal in the frequency domain by removing high-frequency components above a specified cutoff. It is suitable for uniformly spaced data.
Parameters:

- `cutoff` – Cutoff frequency (in units of inverse x-axis units), above which components are removed.

Pros:

- Effective for removing high-frequency noise.
- Retains the overall spectral shape well.

Cons:

- Requires uniformly spaced data.
- Can introduce ringing artifacts (Gibbs phenomenon).

Recommended Use: Effective when noise is dominated by high-frequency components and the data are evenly spaced.
Avoid If: The data is unevenly sampled, or if phase preservation is critical (e.g., for derivative spectroscopy).

3.3 Spectral subtraction

There are several operations that could be performed with the spectrum before the analysis itself. Such preprocessing steps can often help reduce model complexity and enhance the comparability of results between different fits.

3.3.1 Minimum subtraction

Spectral data often contain an artificial offset to prevent the detection system from returning negative intensity. This simple function subtracts the minimum intensity value (y axis) from the whole dataset and sets the lowest point to zero.

3.3.2 Linear subtraction

Linear subtraction calculates the slope of the data from the first and last display points and then subtracts this slope from the displayed data.

3.3.3 Spectrum subtraction

This feature will allow the user to subtract the entire spectrum file from the displayed spectrum. In cases where the spectra have different ranges and varying point densities, the subtraction will be performed only at the overlapping spectra, and the spectrum with lower point density will be used as the grid on which the other spectrum will be interpolated.

3.4 Spectrum normalization

This function allows the user to normalize the spectrum according to the selected range in the data. It uses the same logic as data zoom; instead of focusing on the selected range, it normalizes the displayed spectrum to the maximum amplitude in that range.

3.5 Batch spectrum processing

The single spectrum processing does not overwrite the original data files, but the user has the option to save them manually using the "Save spectrum" option. Performing the preprocessing steps in batch (more data files together) requires different options. All of the preprocessing techniques mentioned above could be performed in batch; the user just needs to define the input folder (the folder where the data on which the preprocessing step will be performed is located) and the output folder (where to save the data after the preprocessing step). It is done by two consecutive dialog windows, which makes it user friendly. To distinguish between the original and modified data, the modified data are saved with the same filename and extension corresponding to the preprocessing step that was performed (for example, subtraction of the minimum = *original filename*_sub_min.txt. The data are saved in the Default format, so they can later be loaded into the ASS easily with the "Load function" or with the "Ctrl+d" shortcut.

Since the normalization logic also needs the operating range on which the normalization will be performed, it is triggered even before setting the input and output folders through dialog windows.

3.6 Spectrum configuration

The final option in the "Spectrum" menu lets users customize details of the rendered image, including the plot title, axis labels, data labels, and the dataset comparison offset. These features enable plot customization according to user preferences and allow shifting the app's focus from Raman to other spectroscopy techniques by simply changing axis labels, as the analysis workflow remains consistent. Once these changes are applied, they stay in the apps settings even after the restart.

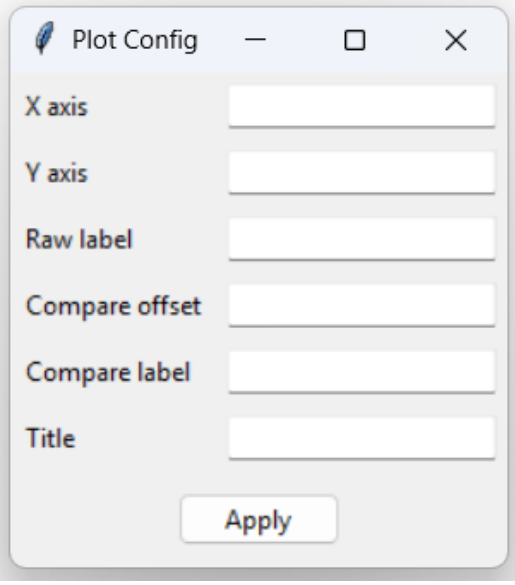

Figure 3.3: Plot configuration window

4 Model

4.1 Building a model

How to build the model is described in detail in chapter 2 in section 2.5. From the top Menu, in the Model section, the "Model builder" can also be called.

4.2 Model storage

The ASS provides a "Save model" feature, storing the model as a .json file with all values like guesses, boundaries, and labels. The "Load model" function allows reusing the model, eliminating the need to build a new one for routine measurements and simplifying repetitive analysis. To remove the model completely without saving it, the "Clear model" feature could be used. It could be called from the "Model builder" window itself or from the top menu.

4.3 Model visibility

For cases when the model is currently not in use but will be used later, "Enable/Disable Model" could be used. It will not delete the model, it will just change its visibility on the canvas.

5 Advanced

This section of the help document outlines the app's advanced features. These functionalities are all available through the "Advanced" option in the top menu.

5.1 Batch analysis

Batch analysis offers an automated way of analyzing all the spectra in the selected folder. For using this functionality, just load one spectrum, select the desired range of analysis, and build/load the appropriate model.
Following the execution of the Batch analysis, the application simply requests the format of the spectrum files it will process and the directory where they are located. It then proceeds to analyze each file in the folder, using the same boundaries as the spectrum used to build the model, optimizing the model parameters, and storing the images in a subfolder within the chosen directory. Once all spectra have been fitted, the optimized parameter values and their errors are documented in an Excel file within the same subfolder.

5.2 1D mapping

One-dimensional (1D) Raman mapping refers to the analysis of spectral evolution as a function of a single varying parameter. In the context of this application, 1D maps are typically acquired by recording Raman spectra sequentially while changing an external variable, such as electrochemical **potential**, elapsed **time**, or **position** along a linear spatial scan.
This type of analysis enables users to monitor dynamic changes in materials, detect chemical or structural gradients, and track reaction progress over time or space. The app supports the import, visualization, and automated processing of 1D datasets in a user-friendly format designed to allow efficient exploration of such trends.
Currently, this app is capable of importing only 1D files, specifically those from Horiba .txt files that have been exported as "split" arrays. To export the timemap and line scans in this format, navigate to the save button in the LabSpec6 software, click the small arrow next to it, and select the "Split array" option. This will automatically export the map into separate files, including time or distance information in both the filename and the file header. For potential maps (SEC - SpectroElectroChemistry), the files must include the potential values applied to the working electrode during spectrum acquisition, with mV as the locator string for reading the potential. For example, a filename such as $sample\_12\_-800mV.txt$ indicates a spectrum recorded at -800 mV.
The loading of the Witec format will be added in the future.

To load the dataset, first specify the source spectroscope (Horiba or Witec) along with the measurement mode to be analyzed (timemap, linemap, SEC). Clicking the "Load" button will then prompt a window where you can select the directory containing the .txt files.
Another section allows the user to specify the map's display details. The left and right cutoffs set the spectral range to be plotted, while the lower and upper indices define the index region (such as time, length, or potential). The colorscale indicates the color scheme for data visualization. The correction option provides basic background correction for data display: "none" for raw data, "zero" sets the lowest point of the spectrum to zero, and "linear" calculates and subtracts a line profile from the raw data using the first and last points of each spectrum. Orientation specifies whether data is shown from top to bottom or the reverse. Interpolation offers graphical smoothing for the image, applicable only to heatmaps, with various strengths available (these are built-in functions of the 'imshow' function in matplotlib; further details are in the matplotlib documentation).

5.2.1 Heatmap

The heatmap presentation of the 1D map data is based on restructuring the input data into a 2D matrix, using the Raman shift and the index as columns and the index, respectively. The intensity values are then used as the variable for the color coding of the dataset. The setting of the smoothing can help to lower the spectral noise and steps in the picture caused by small index sampling. To mark the desired area, right-click on the menu and select the "Select region" option. You can then click and drag the mouse over the image to highlight the area of interest. Upon releasing the mouse button, the spectral and index limits will be updated accordingly.

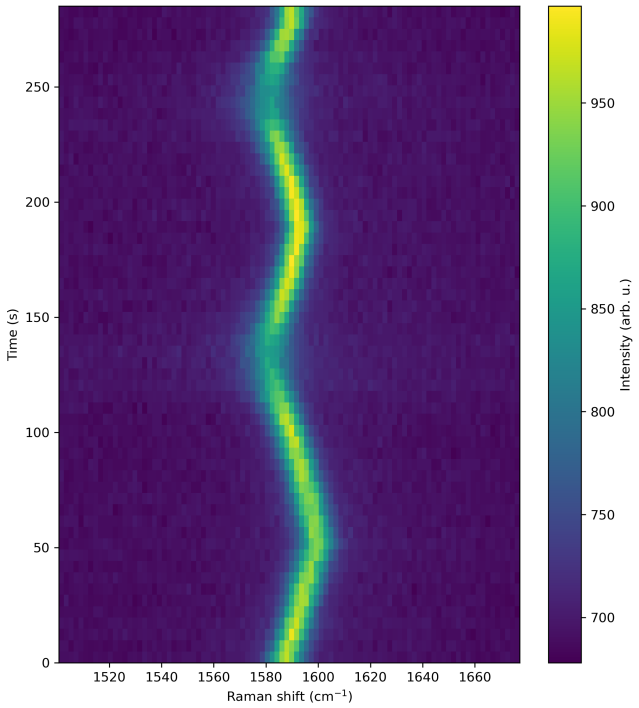

Figure 5.1: Example of timescan mesurement in a form a heatmap

An additional capability provided by this heatmap environment is the option to plot the spectrum directly onto the Main window. To enable this feature, simply perform a right-click on the desired spectrum image that you wish to examine thoroughly, and then choose the "Plot in Main" option from the context menu.

5.2.2 Lineplot

Another way to present a 1D Raman map is through a line plot. This method displays the spectra as a line plot with an offset determined by the slider. The slider adjusts the offset based on a percentage of the dataset's maximum intensity, making it dependent on the correction settings. The spectra are subsequently color-coded according to their indices. In this case, by right-clicking and choosing "Select region," only the spectral range will be selected. The exclusion of the indexes must be performed manually.

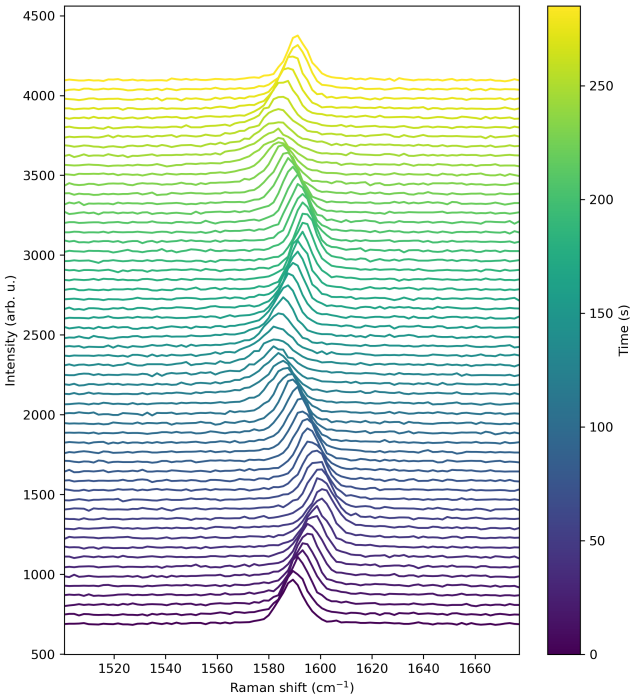

Figure 5.2: Example of timescan mesurement in a form a lineplot

5.2.3 Fitting of the 1D map

The Fit button triggers the fitting process for the one-dimensional map across the specified spectral range using the composite model set in the main window. In case a model is not yet specified, you may export the spectrum from the heatmap to the main window, where you can either manually define the model or import it from an existing .json file. Once the fitting process concludes, the resulting fit parameters and their corresponding indices are saved into an Excel file.

5.3 2D mapping

2D Raman mapping is a procedure in which Raman spectra are measured at different points on a surface, creating a two-dimensional map. Each pixel at this map represents a Raman spectrum, providing detailed information about the sample at each location. Currently this app can analyses only the maps from Witec spectrometer, exported as table with header where columns contains the information about the pixel position (automatic export in Project5/6). Loading of the Horiba 2D maps will be added in the future.
In the upper segment of the right column, just beneath the loading button, users can specify the map parameters. This includes defining the spectral region intended for analysis, along with setting the x and y range, allowing for analysis restricted to a specific part of the map. The

spectral region requires manual setting; however, the x and y limits can be set by right-clicking on the map, selecting the "Select region" option, and then using a left-click to designate the desired area on the map. This function becomes particularly advantageous when a map is already displayed.

5.3.1 Metric evaluation

The initial assessment of the map involves computing the pixel metric. At this point, users have the option to choose among three metrics: the area (which refers to the integration of the spectrum area within a chosen spectral region), the maximum value (indicating the highest data value found within that spectral region), and max_position (which denotes the spectral location of this maximum value). To display the metric map, you need to click on the "Plot metric" button. Additionally, there are other parameters you can adjust: the "Colormap," which specifies the color palette for the map, and "Interpolation," similar to its application in a 1D heatmap. You have the option to focus on a specific region of the evaluated parameter by setting the Scale's minimum and maximum values. This approach can help highlight particular data trends. Any values exceeding the set maximum or falling below the minimum will appear as saturated pixels, with their colors corresponding to either the top or bottom of the chosen color scale. Once the metric analysis is done, it does not need to be run again unless the spectral region is changed. If the spectral range is changed, the program will automatically detect it and run the new metric analysis after the next "Plot metric" click.

5.3.2 Fitting of the 2D map

To fit the 2D map, it is essential to have the model loaded in the main window, which will be used for fitting. There are multiple approaches to achieve this. If you're analyzing known spectral features, you might already have a .json file containing the model from previous work. In that situation, simply load the .json model into the main window. If the model isn't available, you can manually build it in the main window. You can export the spectrum from a single pixel or the map's average by right-clicking on the map and choosing "Plot spectra in Main" or "Plot average in Main." It is necessary to create the metric map first to have some data to export.
Once the model setup is complete, pressing the "Fit" button initiates the fitting process. When it is finished, the options for plotting the fit variables become accessible. Simply select the desired variable to plot from the "Fit variable" dropdown menu, which is pre-filled with variables from the model used, and then click on the "Plot fit" button. The other configurations of the map are the same as those for the Metric map.
The fitted values can be saved as an excel file for later use using the button "Save fit".

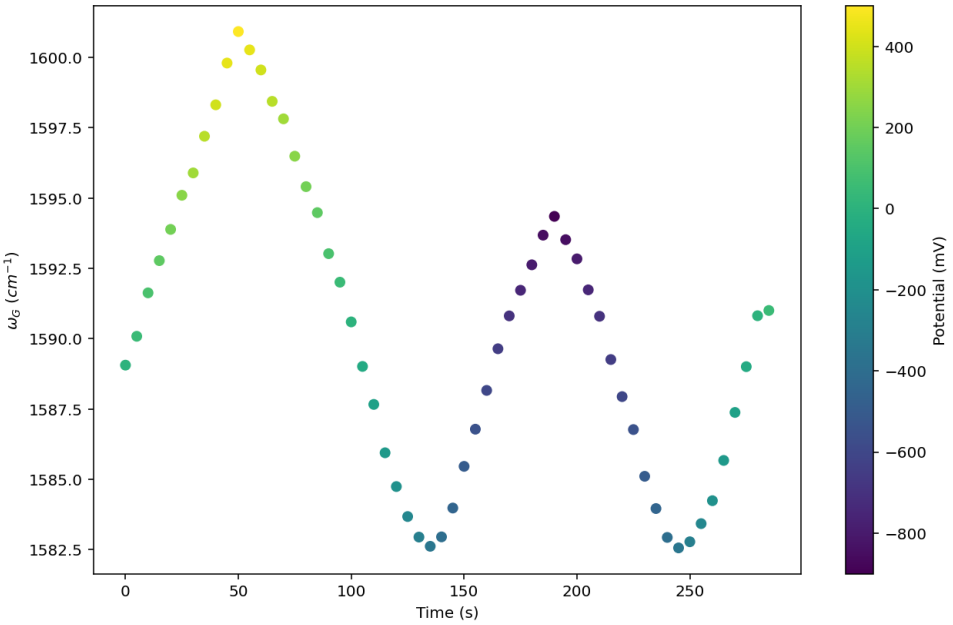

Figure 5.3: Example of plot made with "Excel plotter" enviroment

5.4 Excel plotter

The primary function of this feature is to swiftly allow for the visualization of map and batch fitting outcomes. Once the Excel file is loaded, users can effortlessly choose the data sources for both the x and y axes, assign appropriate labels, and proceed to create the plot. Assigning a third variable to define a color gradient is optional. Example of such plot could be seen in figure 5.3.

5.5 Principal Component Analysis

The PCA module (Principal Component Analysis, powerful multivariate statistical technique) enables users to perform statistical decomposition of spectral datasets to identify and quantify underlying sources of variance — such as changes in doping, strain, chemical composition, or other sample properties — that may not be immediately evident in raw Raman or electrochemical spectra. Within the GUI, the process is designed to be straightforward and fully automated while retaining full scientific control over the analysis parameters.

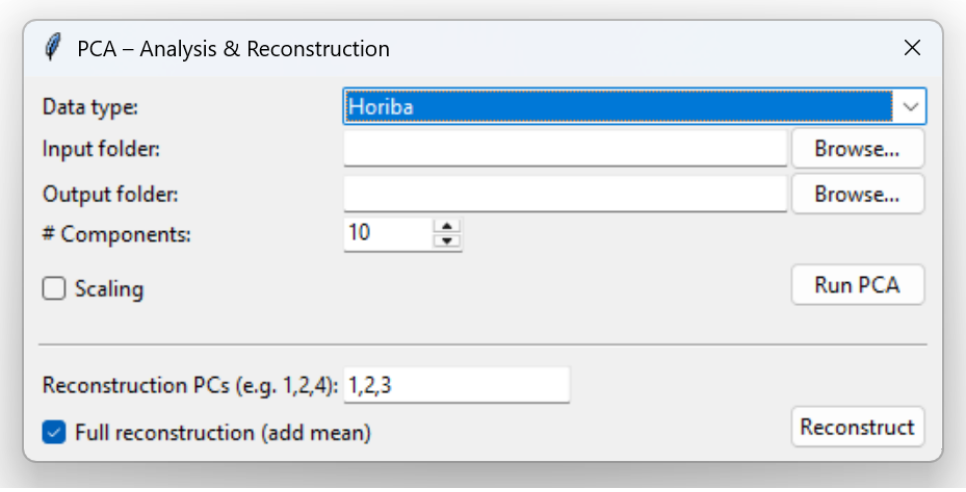

Figure 5.4: Caption of the PCA setting enviroment

The typical workflow begins by selecting a folder containing input spectra (in plain text format) and defining an output directory where all PCA results will be stored. The user can specify the number of principal components (PCs) to compute, as well as whether the data should be scaled (standardized to unit variance) or only mean-centered prior to decomposition. Once these options are set, pressing Run PCA will perform the analysis across all spectra in the selected folder.

Upon completion, the module automatically generates and saves a scree plot (showing the explained variance of each component), an Excel file containing the score values, and plots of score correlations between PC pairs to visualize clustering or sample evolution. The loading vectors are also saved — both as individual text files (matching the original spectral format) and as corresponding loading plots, allowing direct interpretation of which spectral regions contribute most to each component. All outputs are organized in the chosen output directory for traceability and further inspection.

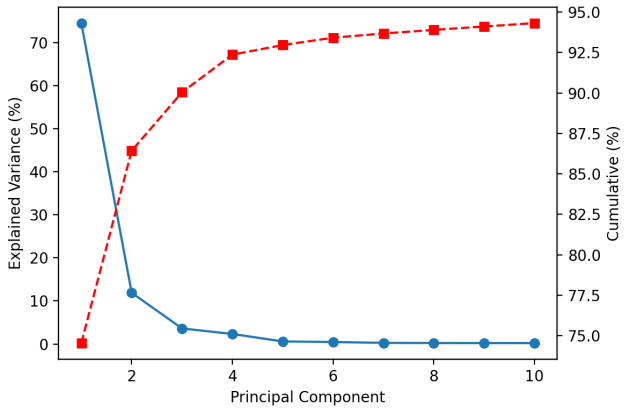

Figure 5.5: Scree plot showing Explianed variance and Cumulative variance

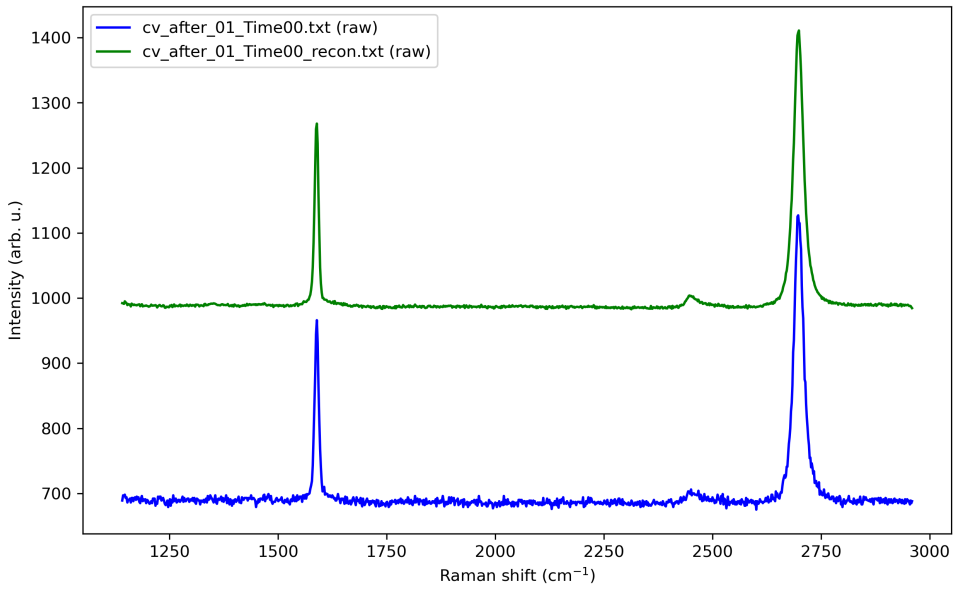

Figure 5.6: Comparison of original spectrum (blue) and reconstructed spectrum with only first three principal component (green)

A dedicated reconstruction window allows users to selectively rebuild spectra using only chosen principal components, rather than a fixed variance threshold. This makes it possible to isolate the spectral contribution of specific components and visualize their effect on the overall dataset. The reconstruction can be performed either with the mean vector included (yielding the best approximation of the original spectra) or without the mean to highlight only the pure influence of the selected PCs on the spectral shape or intensity.
Importantly, once a PCA calculation has been completed, the entire model — including scores, loadings, mean vector, and metadata — is saved in a dedicated PCA result file (pca_model.npz) located in the output folder. This allows the user to perform reconstructions later without repeating the full computation, simply by selecting the folder containing the saved PCA result file as the output directory. This ensures both reproducibility and flexibility, enabling efficient comparison of different reconstruction settings or datasets even after the main PCA run has been completed.

5.5.1 Recommended workflow

- Open the PCA module from the main application window.
- Select input folder containing the spectral data files (e.g., .txt spectra with identical Raman shift axes).
- Select output folder where all PCA results and plots will be saved.

- Set the number of components (N PCs) to compute — typically between 5 and 20 depending on dataset complexity.
- Choose preprocessing options:
  - Scaling: Normalize each variable to unit variance (recommended when absolute intensities differ greatly).
  - No scaling: Apply only mean-centering (recommended when absolute intensities carry physical meaning).
- Run PCA computation.
- The app will automatically:
  - Save the scree plot showing explained variance ratios.
  - Export scores as an Excel file and score correlation plots for PC combinations.
  - Save loading vectors both as .txt files and as corresponding loading plots.
- Perform reconstruction (optional):
  - Choose any subset of PCs to rebuild spectra.
  - Decide whether to include the mean vector (for full-spectrum reconstruction) or exclude it to visualize only the isolated spectral effect of selected components.
- Reopen results later:
  - Each PCA run generates a dedicated PCA result file stored in the output folder.
  - You can load this file later to perform reconstructions or comparisons without rerunning PCA on the raw data.

6 Chatbot

The application includes an optional Chatbot Assistant (accesible from the About section of the top menu), designed to provide instant, context-aware help directly within the graphical interface. The chatbot can answer questions about the program's functions, analysis workflows, and terminology, drawing information from the built-in help file and related documentation database. This allows users to quickly obtain guidance without leaving the application — for example, by asking, "How do I reconstruct spectra from PCA results?" or "What does the scaling option mean?"
The chatbot interface offers a simple conversational window where the user can type their question, press Send, and receive a structured and technically accurate reply. This system is especially useful for new users who are still learning the logic of the app or for experienced users who need a quick reminder of function-specific details.

6.1 Setup Requirements

To enable the chatbot feature, a local language model must be installed through the Ollama framework. The application does not include the model by default to keep distribution size small and comply with licensing requirements.

- Install Ollama:
  - Download and install Ollama from the official website: https://ollama.com/download
  - Follow the installation instructions for your operating system.
- Download a compatible model:
  - After installation, open a terminal (or command prompt) and pull a suitable model, for example: ollama pull llama3
  - You can choose any supported LLM; smaller models (e.g., llama3:8b) offer faster responses, while larger ones provide more detailed answers.
- Run Ollama locally:
  - Make sure Ollama is running in the background before launching the application.
  - The chatbot will automatically connect to the local Ollama server and use the installed model for dialogue.

Once configured, the chatbot becomes a fully offline, locally running assistant — no internet connection is required during use, and all processing happens on the user's computer for maximum privacy and speed.