Interactive Spectrum Assigner

User Manual and Technical Summary

September 30, 2025

Contents

1 What this app does (high-level)

Interactive Spectrum Assigner is a Dash application running in your browser for assigning observed spectral lines to simulated transitions and exporting assignments to `.lin` files. It does **not** do the job for you, but lets you do the job faster. It overlays a measured spectrum (CSV) with one or more simulated or already fitted catalogs (`.cat`) in the spcat format, lets you fit peaks (with single or multi-Gaussian) in a selected region, and then assigns all simulated lines in that region to a chosen fitted center frequency (μ). The assigned line can then be exported

with the quantum number in a `.lin` file format that can be used by `spfit`. The app computes per-line uncertainty components and a total uncertainty, maintains per-catalog intensity scaling, and persists fit context for reproducible recomputes.

2 Quick start

2.1 1. Clone the Repository

Clone the repository from GitHub:

```
git clone https://github.com/berger-lukas/Hyperfine-Spectrum-Analyzer
```

and navigate into the new created Folder Hyperfine-Spectrum-Analyzer

2.2 2. Set Up a Virtual Environment

Create and activate a virtual environment:

```
python -m venv venv

# On macOS/Linux
source venv/bin/activate

# OR on Windows
venv\Scripts\activate
```

2.3 3. Install Dependencies

Install required Python packages:

```
pip install -r requirements.txt
```

3 Project Structure

The project directory is organized as follows:

```
Hyperfine-Spectrum-Analyzer/
|-- plotcomparison.py # Main script
|-- config.json # Configuration file (input)
|-- requirements.txt # Dependencies needed to run the project
|-- assets/
| \-- dark.css # Optional CSS styling for visual output
|-- molecules/
| \-- acenaphthene
| \-- \-- acenaphthene.cat # test .cat file to try
| \-- \-- acenaphthene_measurement.csv # test .csv file to try
| \-- ethanol
| \-- \-- ethanol.cat # test .cat file to try
\-- .venv/ # Local virtual environment (excluded from Git)
```

4 Configuration

The app expects a `config.json` in the working directory. In the config file you can set the paths to your `.csv` and `.cat` files and related settings:

`cat_files`
: Array of one or more simulated `.cat` catalogs. A single `cat_file` is also accepted.

`csv_file`
: Path to measured spectrum CSV, with semicolon separator "`;`" and columns: frequency (MHz), intensity.

`qn_labels`
: Optional mapping of QN column names (e.g., `UpperJ`) to display labels.

`uncertainty`
: Object with:
  - `fid_time_us` (default 10.0): sets $\Delta F = 1/T_{\mathrm{FID}}$ [MHz]; used in the interference kernel.
  - `base_sigma_instr_mhz` (default 0.01): instrumental baseline uncertainty floor.

Temporary caches (auto-managed)

- `spectrum_assigner_scales.json`: per-catalog simulated intensity scales.
- `spectrum_assigner_fitcache.json`: per-assignment fit context by key `cat{idx}/obs{freq}/uid{SimUI`

Note: Ensure your virtual environment is activated before running the script to guarantee that the correct dependencies are used.

5 Input formats

5.1 Measured CSV

Two columns separated by semicolon; intensities are normalized internally to $[0, 1]$.

5.2 Simulated `.cat`

Each line is parsed for frequency, $\log I$, E_{low}, and a count of QNs. QNs are read into columns like `UpperJ, UpperKa, UpperKc, LowerJ, ...`. The app builds a stable `SimUID` for each line and a hover tooltip with QNs.

5.3 `.lin` assignments

On save, each assignment row writes QNs (ordered like the active catalog), observed frequency, uncertainty, and weight. On load, rows are matched back to simulated lines using QNs; uncertainties and weights are restored; the app recomputes components.

6 Running the Script

After setting up the virtual environment and editing `config.json`, run the tool using:

```
python plotcomparison.py
```

Ensure your virtual environment is activated before running the script. Open your browser at http://127.0.0.1:8053

7 The UI, controls, and workflow

7.1 Top controls

Active catalog
: shows which catalog is primary. Press `s` to cycle the active catalog.

Simulated Intensity Scale
per-catalog multiplier (persisted). Click *Apply Scale*.

Default X-axis Range
set a working MHz window and click *Apply X-range*.

Flip Simulated Spectrum
draws simulated sticks below zero (useful for contrast).

Simulated Intensity Threshold
hides low-intensity simulated lines in the plot (assignments are always shown even if below threshold).

7.2 Modes and zoom

Choose among:

- **Zoom** (`q`): standard pan/zoom.
- **Select Region to Fit** (`w`): drag a region to fit Gaussians and baseline.
- **Assign by Region** (`e`): use selection and chosen μ to assign simulated lines.

Use buttons *X+ / X–, Y+ / Y–* to adjust axes; *Undo Zoom* or hotkey `r` to step through zoom history.

7.3 Fitting peaks

1. While in zoom (q) mode, zoom and pan to the peak you want to investigate.
2. Switch to *Select* mode (w), choose # Gaussians (1–10; key `0` means 10) and drag a region;.
3. The app fits a multi-Gaussian model with a baseline. The baseline is fitted on sidebands equal in width to the selected window on both sides. This is used as an offset for the for the gaussian fit and also to calculate the SNR. The window you draw provides the initial guesses for the Gaussian heights and widths. If you fit more than one Gaussian, the initial guess for the width of each Gaussian is the window width divided by the number of Gaussians.
4. Buttons labeled by the fitted center frequencies μ appear; click one to *choose the observed frequency* and peak you want to assign to. You can cycle through them with `d`.
5. The *Fit sum* and individual Gaussians are overlaid, plus baseline and optional $\pm\sigma$ band.

7.4 Assigning lines

1. Ensure a selection region exists and a μ is chosen.
2. Switch to assign mode (press `e`). Draw a rectangle over the lines from the active catalog you want to assign.
3. Click *Assign Region* or press `a`. The app assigns *all* selected simulated sticks from the **active** catalog to the chosen observed frequency.
4. Assigned sticks are drawn as dashed blue; unassigned sticks are red (active) or gray (inactive catalogs). The measured trace is black.

7.5 Assignments table

- Columns: `obs`, `sim`, `Delta (obs-sim)`, `Eu`, `logI`, QNs, component displays `Disp_SNR / Disp_Interf / Disp_Merge`, editable `Uncertainty`, and computed `Weight`.
- **Click** any of the colored component cells to toggle whether that component contributes to the total; the total is recomputed in quadrature without re-fitting.
- **Delete rows**: select a row to remove it.

- **Recalculate Weights**: recomputes relative weights per observed frequency cluster from $10^{\log I}$.

8 Keyboard shortcuts

`q`	Switch to Zoom mode
`w`	Switch to Select Region to Fit mode
`e`	Switch to Assign by Region mode
`a`	Assign region to current μ
`d`	Cycle through fitted μ values
`0-9`	Set number of Gaussians (0 = 10)
`r`	Undo zoom
`s`	Switch active catalog

9 Uncertainty model (how it's computed)

For each unique observed frequency (`obs`), the app estimates three components and combines them in quadrature

$$\sigma_{\text{total}} = \sqrt{\sigma_{\text{SNR}}^2 + \sigma_{\text{interf}}^2 + \sigma_{\text{merge}}^2}\,.$$

SNR / instrumental σ_{SNR}

From the fitted context if available: baseline RMS $\approx \sigma_{\text{baseline}}$ (from sidebands) and peak height = sum of fitted Gaussians at `obs`. With $SNR = \frac{\text{height}}{\text{baseline RMS}}$, the app uses

$$\sigma_{\text{SNR}} = \sqrt{\left(\frac{0.0575}{SNR}\right)^2 + \text{BASE\_SIGMA\_INSTR}^2}.$$

Interference σ_{interf}

Computed *globally* across all observed lines using a crowding kernel based on pairwise distance D and $\Delta F = 1/T_{\text{FID}}$:

$$C = \tfrac{1}{2} D\big(1 - \tanh(D/\Delta F)\big), \qquad \sigma_{\text{interf}} = \Big(\sum C^2\Big)^{1/2}.$$

Merge σ_{merge}

Half-range of simulated frequencies mapped to the same observed line ($\frac{\max - \min}{2}$).

Users can toggle each component's inclusion per row. The app also displays semi-transparent vertical uncertainty bands for visible observed lines (limited to small spans for clarity).

10 Saving and loading `.lin`

Save Click *Save .lin file*. A file named `<catalog>_assignments_YYYYMMDD_HHMMSS.lin` is written to `./assignments/`.

Load Provide a path and click *Load .lin File*. The app parses QNs, matches to simulated rows, restores `Uncertainty`/`Weight`, and recomputes components and totals.