MESS GUI Documentation - Last Updated: April 30th

# Introduction

Originally Known as CAMO-S GUi, this software is not titled Meteor Elemental Spectra Software(MESS) was developed starting in May 2021. It is designed to provide a graphical user interface (GUI) and processing pipeline for extracting and analyzing meteor emission spectra from observational video data. MESS integrates stellar calibration, extinction modeling, and plasma emission modeling to estimate elemental abundances in meteor events. This program has only be tested in Linux.

# Installation Requirements

Before installing and running MESS, ensure the following dependencies are available in your Python environment.

### Python Version

* Python 3.9.19

### Required Python Libraries

The following are the key packages required, based on the verified working environment:

|  |  |
| --- | --- |
| **Library** | **Version** |
| numpy | >= 2.0.2 |
| matplotlib | >= 3.9.2 |
| pyqtgraph | >= 0.13.7 |
| PyQt5 | >= 5.15.10 |
| imageio | >= 2.6.1 |
| scipy | >= 1.12.0 |
| scikit-learn | >= 1.5.2 |
| cython | >= 3.0.11 |
| astropy | >= 5.1 |

# Installation Instructions

1. Create and activate an anaconda virtual environment (optional but recommended)

|  |
| --- |
| conda create -n mess python=3.9 conda activate mess |

1. Install RMS and WMPL following instructions from their github

|  |
| --- |
| Western Meteor Physics Library (WMPL) Github link: https://github.com/wmpg/WesternMeteorPyLib RMS Github link: https://github.com/CroatianMeteorNetwork/RMS |

1. Ensure Python Packages are the correct version

|  |
| --- |
| conda install -c conda-forge pyqtgraph imageio scikit-learn matplotlib scipy cython numpy astropy |

1. Verify package versions (optional but recommended):

|  |
| --- |
| python -m pip list |

1. Clone the MESS repository **within the RMS folder structure**:

|  |
| --- |
| cd ~/path/to/RMS git clone https://github.com/yourrepo/MESS cd MESS |

1. Build the spectral library:

|  |
| --- |
| cd spectral\_library make && make install |

1. Launch MESS:

|  |
| --- |
| python MESS.py |

### Files for GUI functionality:

* MESS.py: Main application GUI
* CAMO-Spectral\_Library.py: Python-C++ interface library for spectral computation
* spectral\_library/: Contains compiled C++ shared libraries, config files, and data
* DriverInputFiles/: Contains spectral config and star catalog files
* SPCAL/: Contains responsivity and extinction calibration files
* Vids/: Sample Videos can be used for

# CAMO-S.py overview (sequential order)

See script for full doc string associated with each function.

1. Imports and support functions

* Standard imports from python and pyqt5
* Import functions from the wmpl library
* Cython init is there to supporting loading and applying flast later on
* Support functions and classes
  + Adjust levels: called later to adjust the levels of the image seen by the user (not the levels used to analyze the measured spectrum)
  + loadImage: load an image using imageio
  + binImage: will take a given image and return a binned image array
  + FlatStruct (class): structure containing flat field, used to load and apply the flat later on
  + loadFlat: reads in file and path name
  + applyFlat: takes output from loadFlat, applies flat to actual image

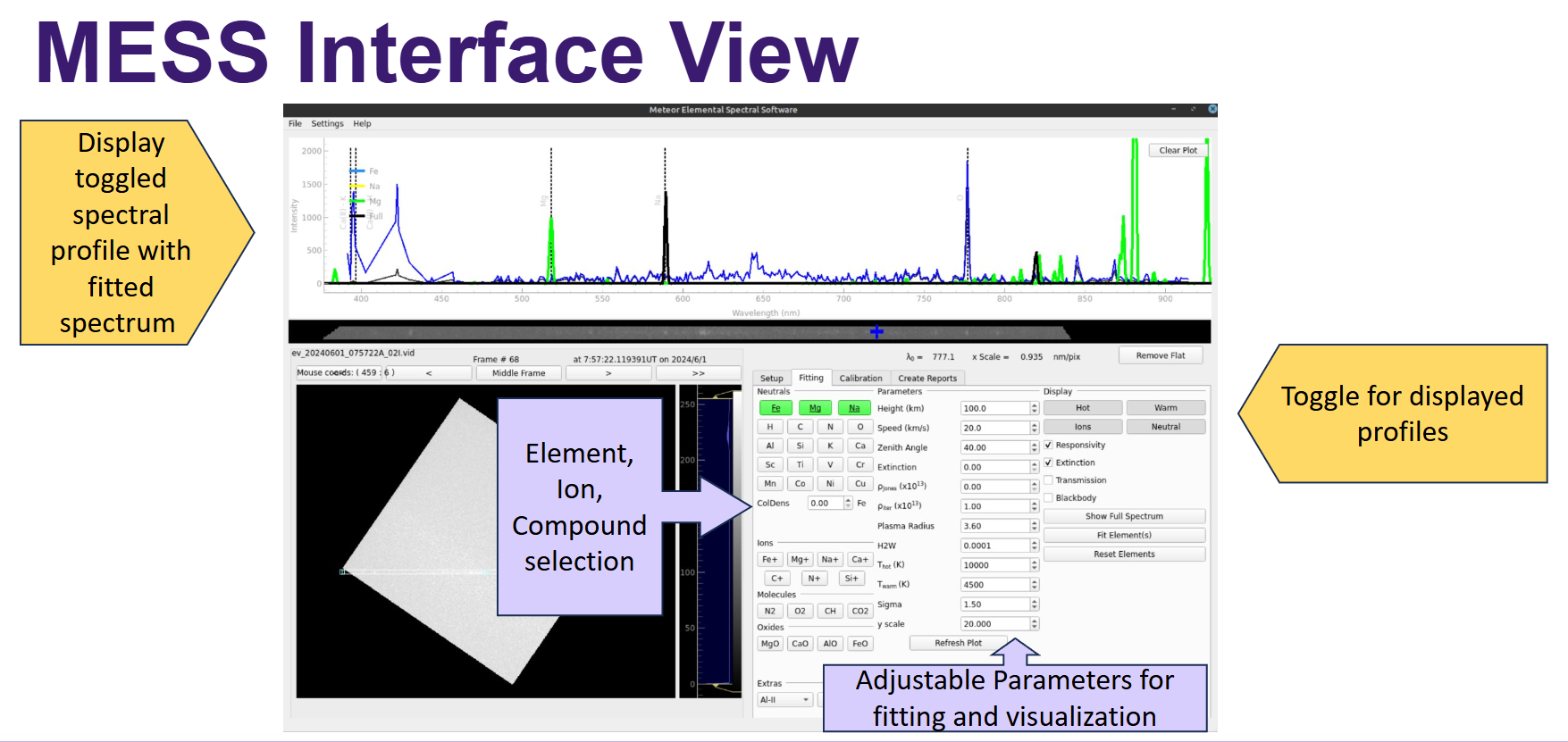
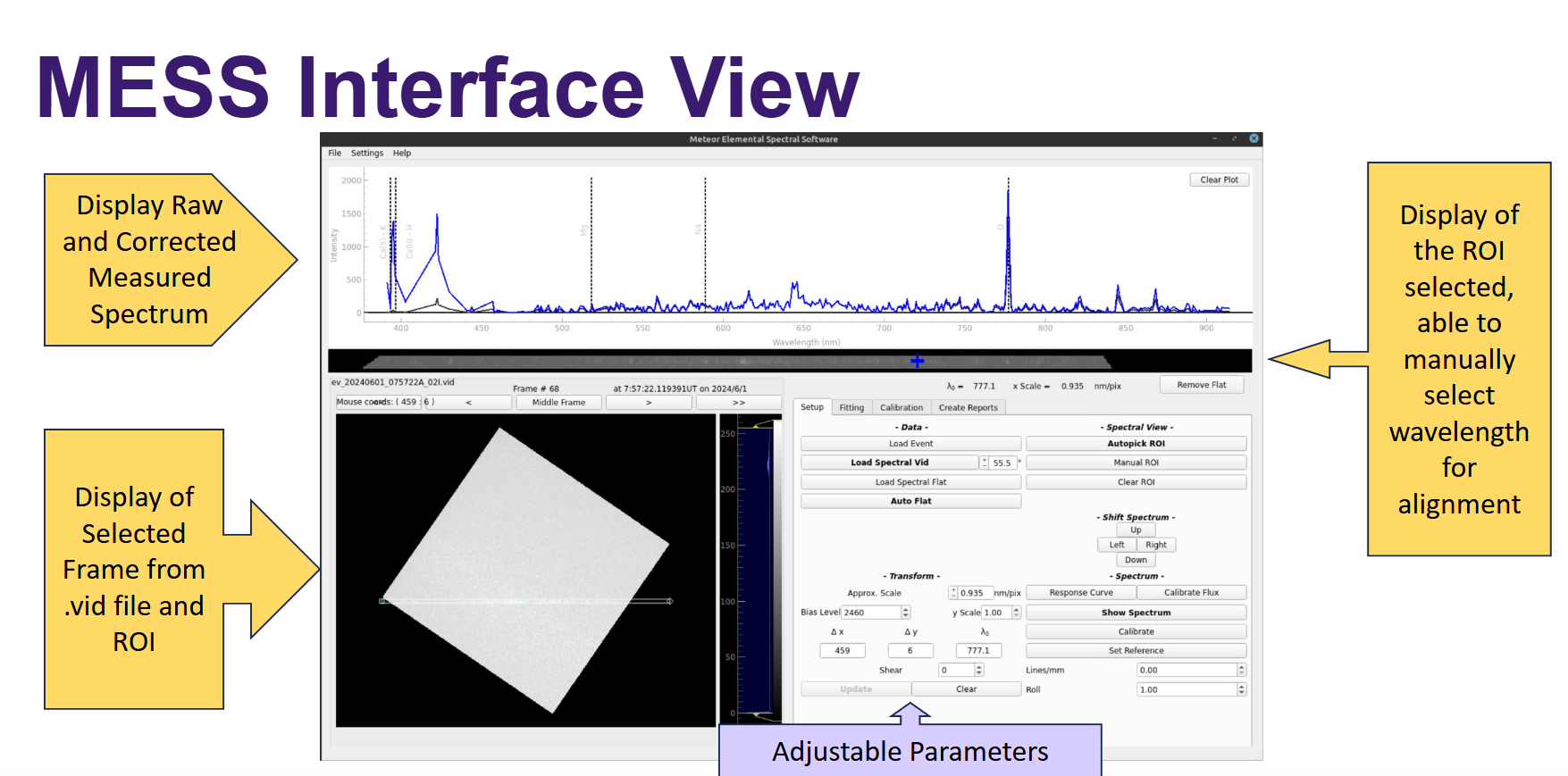
2. Primary Class - contains GUI functionality

* GUI Modifications (initialize graphics view items, buttons, etc.)
  + All *direct* *video modifications* will come first, in the order of: image view, image level histogram, mouse cursor
    - Direct file image view: “window” in which the user will see the video frames
    - Direct image histogram: histogram to the right of the image where the user can manually adjust image levels
    - Initialize direct mouse: change the cursor to a crosshair in the direct image view widget, and allow the user to get the image coordinates when they click the mouse
  + All *spectral video modifications* will come next, in the order of: image view, image level histogram, mouse cursor and region of interest, background sets, markers, and spectral flat
    - Spectral file image view: same as direct file image view, for the spectral video frames
    - Spectral image histogram: same as direct image histogram. Note that if the user adjusts the levels in the GUI, it will not affect analysis of the measured spectrum
    - Initialize spectral mouse and region of interest: similar to direct mouse, make the cursor a crosshair in the spectral image view. *Also* initialize the spectral region of interest, which will be used later to analyze a certain part of the image and measure the spectrum
    - Initialize background sets: frame numbers for set 1 and set 2 (both of which should *not* have the spectrum visible) to be summed and averaged to provide an image background. This background can be viewed by the user, and will be subtracted from the spectrum to provide a clearer measurement.
    - Spectral markers
      * Affine marker: will plot the affine transform from the direct image view so the user can confirm which spectra they want to measure, provided more than one is visible on the spectral image view (i.e. if fragments are discernable)
      * Profected affine marker: used by the GUI to project the affine marker onto the spectrum, as the affine marker is never exactly on the desired spectrum. This marker is not seen by the user, but used by the plotting function to get the scale of the plot.
    - Load spectral flat: allows user to input file and path name, followed by a function call to load the spectral flat to every spectral video frame.
* Button triggers
  + Direct file control buttons (see code for button names/associated functions)
  + Spectral file control buttons (see code for button names/associated functions)
  + Plotting buttons (see code for button names/associated functions)
* Functions to support image upload and analysis
  + Direct file control functions
    - uploadDirectVid: reads video into GUI, displays 0th frame
    - updateDirectFrames: re-sets the frame being shown, updates associated time, date, and frame number. Adjusts image levels using adjustLevels function (as seen in section 1).
    - nextDirectFrame: increase the frame number by 1
    - lastDirectFrame: decrease the frame number by 1
    - affineTransform: use affine transform file, scale plate, and plateScaleMap to run the affine transform *from* direct *to* spectral images. At present, the affine transform file is only loaded in the script: ensure the file is in the *same folder as the CAMO-S.py file*.
  + Spectral file control functions
    - uploadSpectralVid: same as uploadDirectVid, but for the spectral file
    - updateSpectralFrames: same as updateDirectFrames, but instead of using image levels it applies the spectral flat.
    - nextSpectralFrame: same as nextDirectFrame
    - lastSpectralFrame: same as lastDirectFrame
    - spectralROI: introduces a square region of interest, which can be resized/rotated as needed to fit the spectrum.
    - checkSpectralBackground:
      * Uses frame sets 1 and 2, initialized in section 1, to define a frame range, build and fill an array, find the median of the array, and create a background image. Includes flat in background calculations.
    - showSpectralBackground: shows background in a pop-up window so the user can confirm it is satisfactory. Not a necessary step.
    - checkSpectralRegion: get the array region in the spectral ROI box to define image data
    - clearSpectralROI: removes ROI box on spectral image view and clears all associated data. Resets spectral roi.
    - clearAffine: removes the affine marker associated with the affineTransform function in the direct file control functions. Re-initializes the marker so the affineTransform function can be performed again later, with new data.
    - removeSpectralFlat: Very similar to updateSpectralFrame button, but uses the adjustLevels function instead of applying the image flat. Will only remove the flat from the current frame. Background and spectrum will still be calculated with the flat applied.
  + Joint file control functions
    - nextFrame: moves both direct *and* spectral frames forward by 1 frame.
    - lastFrame: moves both direct *and* spectral frames back by 1 frame.
    - nextTimeFrame: moves to next frames that are as close in time as possible.
    - lastTimeFrame: moves to last frames that are as close in time as possible.
  + Plotting functions
    - profectAffine: projects the affine marker from its mapped location on the spectral image to a point on the spectrum
    - plotMeasuredSpec: calculates the background and the region of interest, even if the user hasn’t manually checked them. Uses scaling parameters given (2.85 nm/pixel) and projected point on spectrum to plot the spectrum in as intensity vs wavelength (nm).
    - clearSpec: clears all measured spectra shown on the graph.

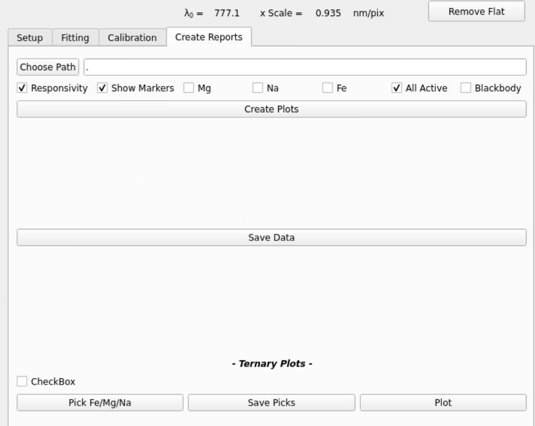
# GUI Overview

The overview of mess can be seen as below, they are generally separated to 4 different tabs:

* Set up
* Fitting
* Calibration
* Export

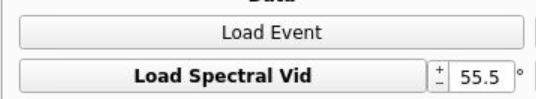






# Workflow to Analyze a Meteor Spectrum:

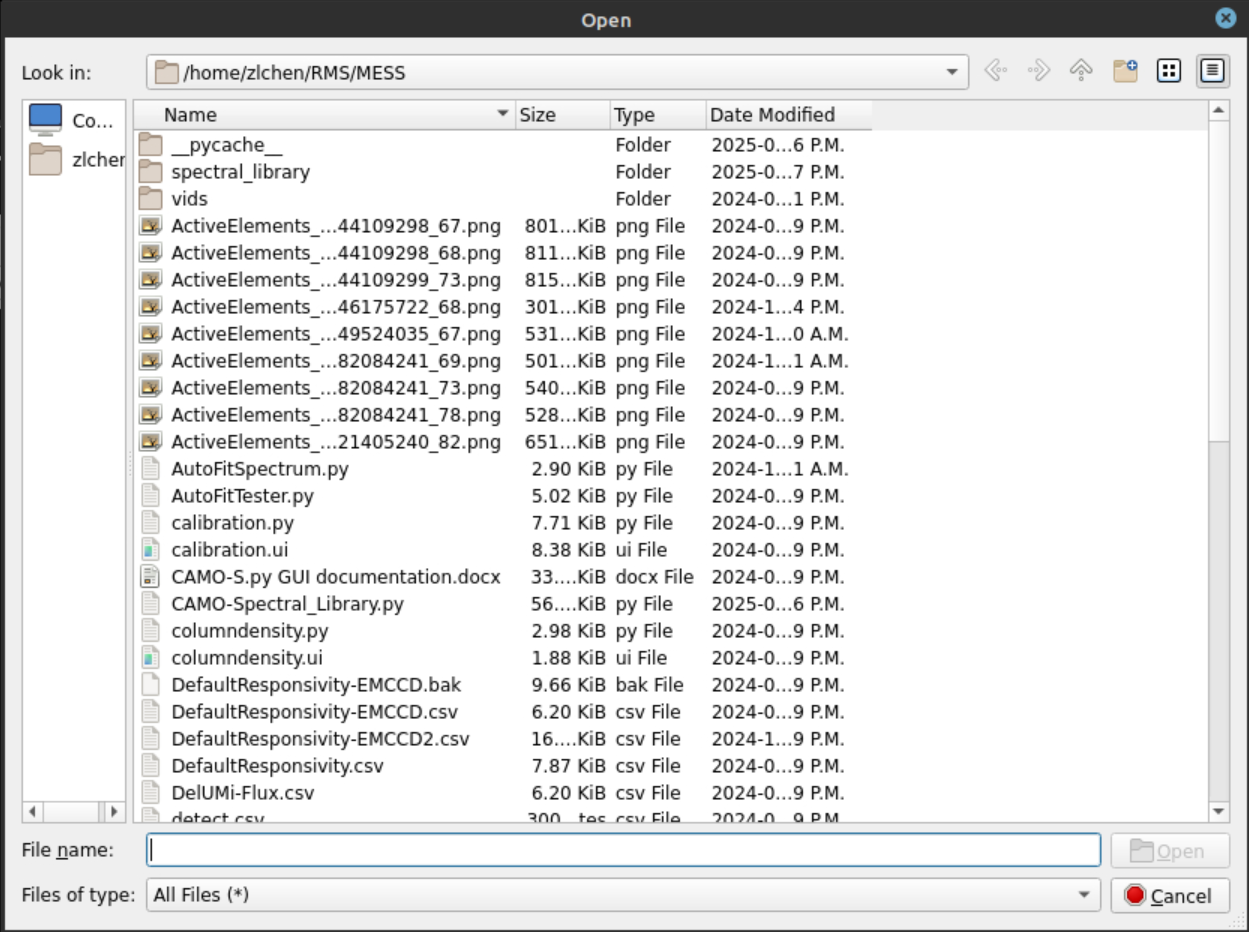
1. Upload .vid file

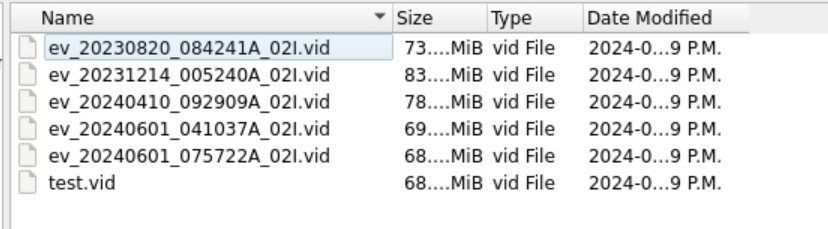


This can be done in 2 different ways, you can either chose to upload a directory with the **load event** button, in which most commonly meteor files have the directory format of

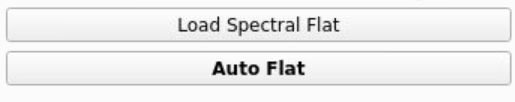
**/srv/meteor/klingon/evcorr/EVENT\_DATE/EVENT\_DATE+EVENT\_TIME+A/event.txt**

If you want some sample .vid file to upload and see, they are in **MESS/vids**





1. Auto-pick or define ROI

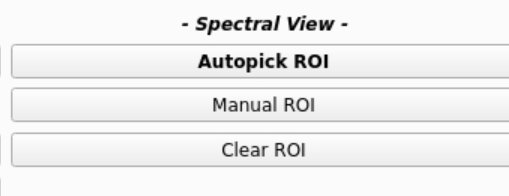


Spectral Flat Field can be manually uploaded in .png format using the **Load Spectral Flat** button, there are two sample flats in MESS/ you can use



Or an Automatic Flat can be applied by clicking **Auto Flat**

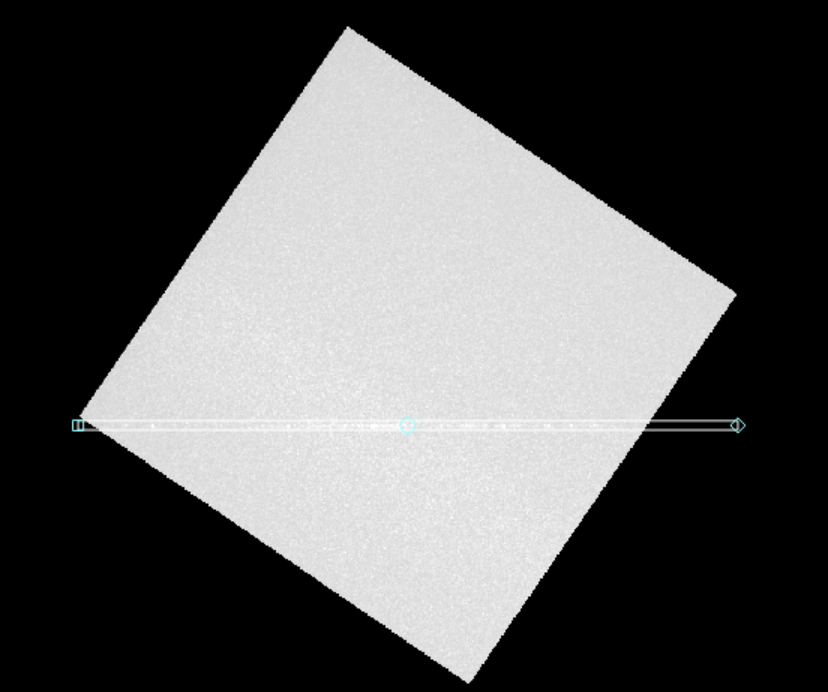
1. Choose the Region of Interest either Manually or Automatically



The region of interest can be selected automatically with **Autopick ROI** button, which is the **Spectral line** on the image

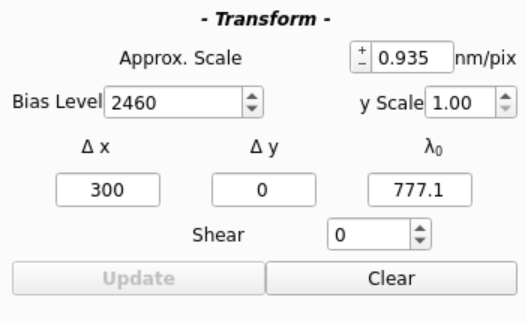
This can also be done manually by clicking **Manual ROI**

You can check the validity by comparing your result to the two images below





1. Select the reference point to match the reference wavelength (default at Oxygen 777nm)



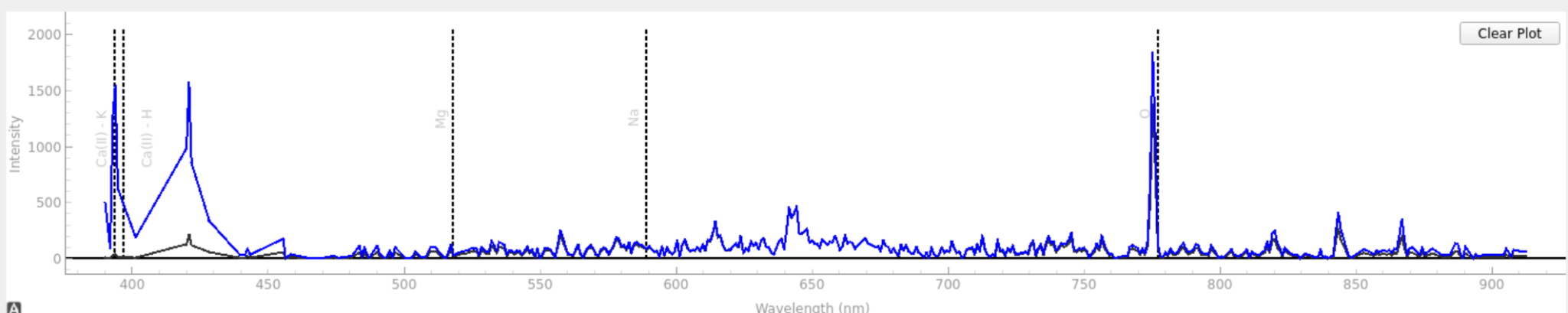
You select the point by clicking at the stripe display of the ROI



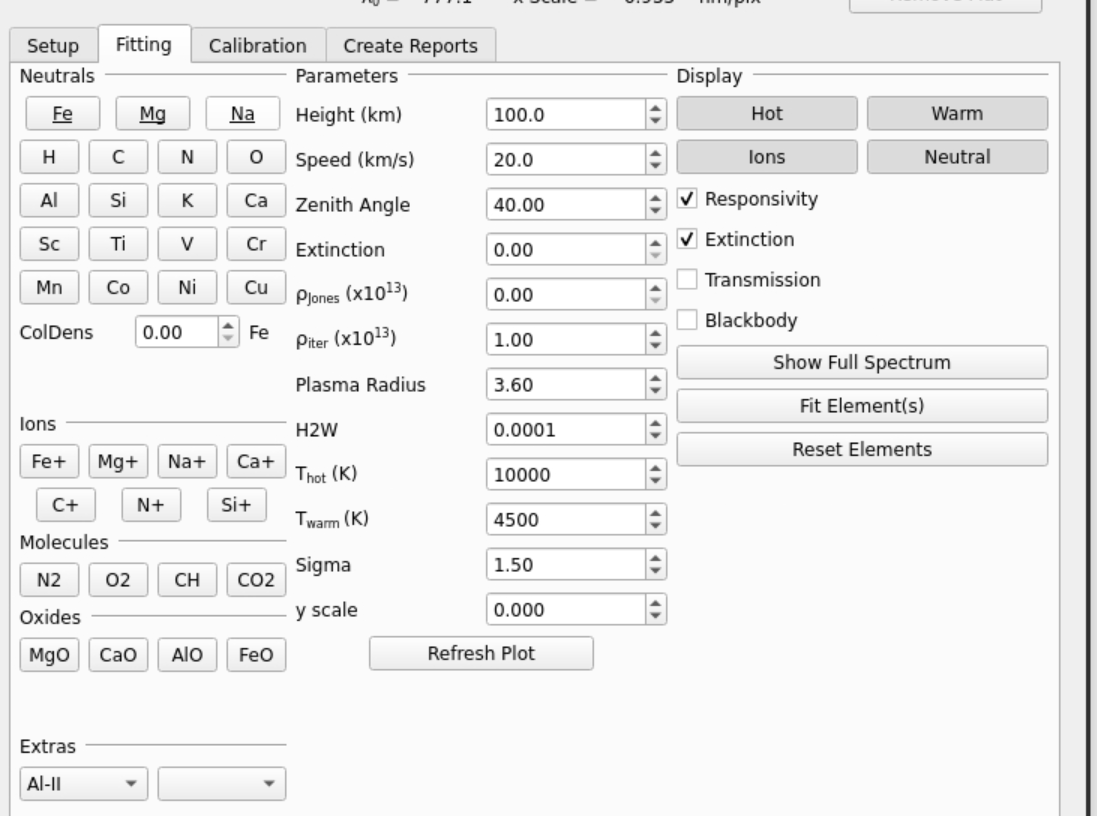
1. Click **show spectrum** to generate the calculated meteor spectrum, which should display on the top display of the software.

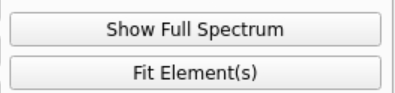
* Black spectrum - The raw spectrum calculated with pixel intensity
* Blue spectrum - The corrected spectrum after responsivity correction



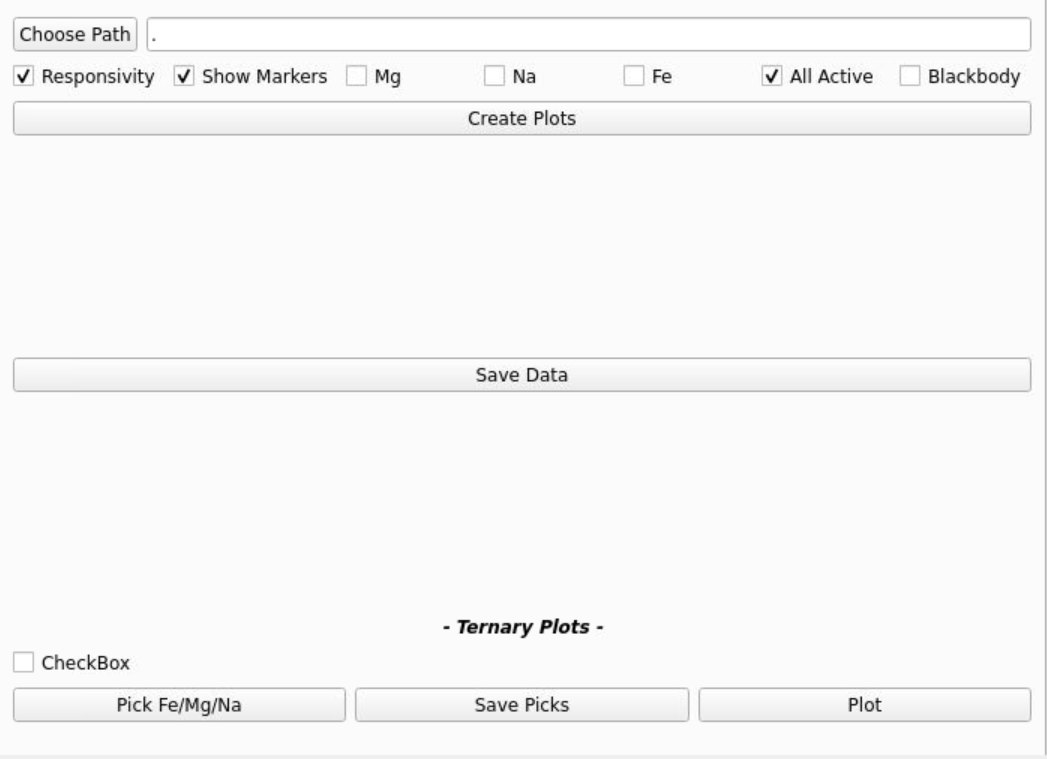


1. Select elements and put in parameters for fitting, click **Fit elements** to fit the selected elements/ion/molecules, the terminal will give the n warm values of the selected elements. Click **Show spectrum** to see the fitted spectrum





1. Save results by choosing a directory path and either save a matplotlib generated plot or save a .txt file of the spectrum



### Current issues to resolve

* The fitting doesn't work properly, check pete's code for fitting and extinction model
* CAMO-S Calibration data required to implement Extinction
* Calculation of extinction and trajectory from text/pickle files (Denis can provide more guidance on this)
* Checking for flat everytime before its applied

### **Common Glitches and issue**

**Any issue might be able to be solved by waiting a second, and retrying again, if not, relaunch MESS**

* If you are getting weird number or error messages such as 'Memory not allocated for…" or "segmentation faults"
  + Ensure that allocMemory() is called after readSpectralConfig(), it is worth relaunch MESS and check if the 'weird number' seem to be cumulative
* Mismatch between spectrum and the reference lines
  + clear the plot and choose the reference point on the ROI again
* GUi element button fail to toggle
  + make sure only select up to 3 buttons a a time
* Flat field mismatch or has no effect
  + plot out the flat itself to double check for that
* Video would not load
  + Check bits of the video and ensure it works with wmpl library's requirement
* Fitting is off, what should I check
  + Change the fitting range in the specalConfig file from 350nm – 1000nm to 400nm – 800nm