

# M\_spec, Python manual

Martin Dubs, FMA

## Summary

This manual explains the use of a Python script for the analysis of video meteor spectra. The theory behind it has been described in (Dubs and Schlatter, 2015):

[meteor\\_spectroscopy\\_wgn43-4\\_2015.pdf](#)).

Until now, the method had been applied successfully to analyse meteor spectra obtained with high sensitive video cams such as the Watec 902 H2 ultimate or colour cameras such as the Sony alpha 7S, which allow video recording. Unfortunately the method required the use of different software for the various steps of the spectra processing, since standard spectroscopy was not adapted to the special requirements of meteor spectroscopy with a fast moving spectral source and nonlinear spectra caused by oblique incidence of the light to be analysed. By combining the different steps of the analysis in a single pipeline, using Python, the analysis should be easier, faster and more reliable. The processing steps include extraction of the images from the video file, defining a background image, subtraction of background, transformation of the images to orthographic projection, resulting in linear spectra. These are superposed (registered) and converted to 1-dimensional spectra by summing over several rows. The intermediate images of the processing are stored as FITS-images. The advantage of this format is that it also contains information about the image creation and processing, important for spectra of professional quality. The last steps are wavelength calibration and plotting of the final spectra. The procedure follows the processing as described in (Dubs and Maeda, 2016: [Calibration of Meteor Spectra Dubs IMC2016.pdf](#)).

At the moment some processing steps are not implemented yet, such as the flat correction and the instrument response. These will be added later if needed.

The program is still in a development stage, so the user is asked to report problems and errors, for correction in later versions. The present version 0.9.15 changes from a pipeline based approach to a windows GUI, using PySimpleGUI (<https://pysimplegui.readthedocs.io/en/latest/>) for creating the windows interface. This allows more flexibility in solving the tasks of meteor spectra processing. The style is similar to ISIS, a well known and widely used astronomical spectroscopy software.

## Introduction

For amateurs, the analysis of meteor spectra has been quite difficult. Standard spectroscopy software is not particularly suited for meteor spectra. Therefore different software had to be used for parts of the processing, which discouraged potential users and made the analysis quite complex (see the manual: [processing-meteor-spectra-v151.pdf](#) for the old method). Most of the processing steps are the same for different meteors, therefore a processing pipeline which combines the different steps can simplify the meteor spectra processing. The pipeline is written as a Python script.

Python was chosen, because

- it contains all the necessary tools to do the analysis
- it finds widespread use in the astronomy community
- it is free
- it runs on different platforms

I was inspired to use Python by Giovanni Leidi, who was giving a talk about the use of Python for the analysis of spectra in the spectroscopy workshop at OHP 2018 (<https://www.shelyak.com/ohp-spectro-star-party-2018/>).

For the processing of meteor spectra it is necessary to calibrate the equipment beforehand. For this a separate script, also in Python, was written (see: [Calib Python manual 2.pdf](#))

Note of caution: I am new to Python, so the script presented here may not be the best solution. Some things have been done in a complicated way, copying examples from different sources

and trying to make it work. I try to improve it for clarity and safety of operation. Therefore I hope you will suggest improvements.

The manual is divided into three main sections. First a general description of the processing steps is given, which are followed by any processing software. The next section describes the installation of Python, such that the specific Python scripts for analysing meteor spectra can be run. In the following section a detailed description how to process a video file into a calibrated meteor spectrum is given, using the script `m_spec.py`. At last some hints and tricks are given for efficient use of the script.

`M_spec` is intended for a first analysis of meteor spectra. For more detailed studies and reanalysis of processed meteor spectra standard spectroscopy such as ISIS or Vspec may be used. The registered images and spectra produced by the Python script are in a format, which is suitable for these spectroscopy programs.

## Processing of meteor spectra, overview

### Image extraction from video sequence

The starting point for the analysis is a video file. Typically these are recorded with UFO capture, a program to detect meteors in real time. The program uses a pretrigger to record one second before the meteor appears until the end of the meteor. This video is separated into an image sequence which is stored as \*.BMP images. In Python this is done with a call to `ffmpeg` (<https://www.ffmpeg.org/>). For the Watec camera, the images are either single frames or the interlaced frames can be separated into two fields, each containing a half frame (even and odd scan lines). These have to be arranged in the correct order (bottom field first or top field first). The use of fields with double image rate is useful for fast meteors, having a large velocity component in the dispersion direction, which reduces the spectral resolution.

For colour videos, there are two processing methods, in colour or black and white. Processing in colour produces nice colour images of the spectra and makes the assignment of spectral lines easier. On the other hand it is three times slower and uses lots of disk space. For the extraction of 1d-spectra the colour images have to be converted to b/w images. Therefore it is usually more convenient to do the whole processing in b/w. Another possibility to speed up the processing is by binning the images in the video extraction in case of 4k videos (only 2x2 binning supported at present).

### Background image

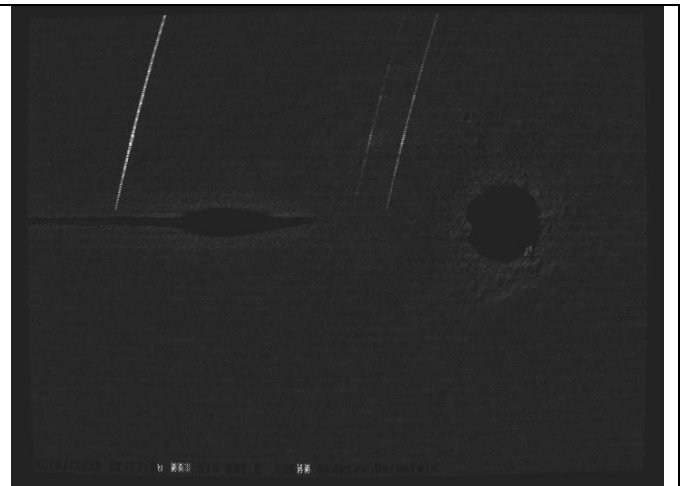
An important step in the processing of meteor spectra is background subtraction. The video records not only the meteor spectrum, but also other light sources such as sky background, starlight and light pollution from different sources. Most of this does not vary rapidly and can be subtracted before further processing of the spectra.

The background image is made by averaging a number of images without the meteor. For this, the first images of the video can be used. The pretrigger in UFO capture is adjusted normally to about one second. At a frame rate of 25 images per second this means that the first 20 images can be used for the background, leaving some safety margin if the detection algorithm misses the first few meteor images. Averaging of 20 images reduces the statistical noise in the images by a factor square root of 20 or about 4.5. In the case that the video is converted to fields the image rate is doubled and also the number of images for the background can be doubled.

This background image is subtracted from the remaining images containing the meteor spectra. The effect of the background subtraction in the case of moonlight:



Peak image of meteor images as displayed by UFO capture (no background subtraction)



Peak image of meteor images after background subtraction and orthographic transformation (see next section)

### Image transformation to the orthographic projection

As described in the above mentioned paper: [meteor spectroscopy wqn43-4 2015.pdf](#), the recorded spectra have a nonlinear dispersion and are curved differently in different image areas. These spectra are linearized and made parallel by a transformation to an orthographic projection. This transformation can also correct lens distortions. In the same processing step the spectra can be rotated, so that the resulting linear spectra are parallel to the image x-axis. The transformation is described by an axial symmetric radial transformation of the form  $r = r' * (1 + a3*r'^2 + a5*r'^4)$  (plus higher terms in the future), where  $r$  and  $r'$  are polar coordinates in the original and transformed images. The equation describes the inverse transformation, since for every image point in the resulting image the original coordinates have to be calculated. The polar coordinates are measured from the coordinates of the optical axis ( $x_0$ ,  $y_0$ ) and the polar angle is corrected by the spectrum rotation. The transformation is done in three steps, conversion to polar coordinates, radial and angular transformation, conversion back to Cartesian coordinates. For the interested reader the Python code is given here:

```
x, y = xy.T
x0, y0 = center
y0 = y0 * yscale # the center in the original image has to be scaled as well
# y has been scaled in a previous step with resize image
rp = np.sqrt((x - x0) ** 2 + (y - y0) ** 2)
phi = np.arctan2(y - y0, x - x0) + rotation
r = rp*(1 + rp**2*(a3+a5*rp**2))
xy[... , 0] = x0 + r * np.cos(phi)
xy[... , 1] = y0 + r * np.sin(phi)
```

The parameters  $x_0$ ,  $y_0$ ,  $a_3$ ,  $a_5$  and rotation have to be determined beforehand in a one time calibration for each camera – lens – grating configuration. In addition, in the case of non-square pixels a transformation (stretching in y-direction) by a factor  $yscale$  has to be applied before the transformation to orthographic coordinates. When fields (half frames) were extracted from the interlaced video, the images have to be scaled by an additional factor 2 in y-direction, in order to compensate for the missing lines. For the laser calibration, a script `M_calib.py` is available, which still needs some testing. It was written before `m_spec`, but as a first attempt with a windows GUI it needs some improvement. However, the previous calibration `s_calib.py` is fully compatible with `m_spec`.

The radial transformation conserves the dispersion or image scale in the center of the image (at  $x_0$ ,  $y_0$ ). The dispersion `disp0` can be used to convert the spectra from pixel to wavelength (more about this see below).

## Image registration

In stellar spectroscopy, the position of the star image can be kept in a fixed place, usually defined by a narrow slit, which allows a constant calibration of the spectrum. With meteor spectra this is hardly possible, the meteor appearing without warning and moving fast enough to make capture and tracking impossible, at least for the amateur. Therefore the spectra may be recorded anywhere in the image plane, in a different position from image to image. For the analysis they can be shifted in such a way that they overlap. This works only because they are all parallel and have the same linear dispersion. Aligning therefore the zero order will automatically align the whole spectrum. Summing these aligned (registered) spectra allows to improve signal to noise ratio. The aligning is done by measuring the position of a prominent line (the zero order if visible, or another prominent line if the zero order is outside the image area) in each image. This position is determined by fitting a Gaussian peak shape to the line and using the centre as line position. The line position in the first spectrum is used as a reference, to which all the following spectra are shifted.

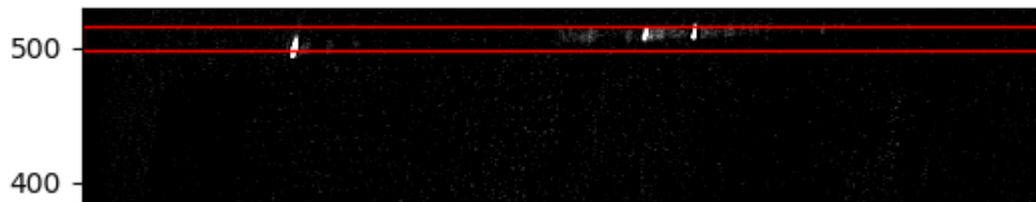
In the next step a suitable number of images are added, which results in a 2-D spectrum image, which can be analysed further in the following steps.

## Correction of tilt and slant

The deviation of the meteor spectrum from the horizontal axis is called tilt. For the extraction of the spectrum from the 2-dimensional image a number of rows corresponding to the width of the spectrum are added. This works correctly only if the spectrum is aligned parallel to the rows. During calibration this angle has been determined and the images were rotated so that the spectra should be aligned parallel. In the case that the grating has been rotated since the calibration, the spectra will not be parallel to the rows. This can be corrected by applying a "rotation" to the added spectrum. Mathematically a shear is applied; the columns are shifted vertically in proportion to the column number:

$$dy = \text{tilt} * (x - x_0)$$

$x_0$  is selected in the centre of the image.



*Spectrum before tilt correction*



*Spectrum after tilt correction, tilt = -0.04, spectrum "rotated" clockwise.*

The same effect would be obtained by changing the rotation angle in the configuration file by increasing rot by 0.04 (angle measured in radians)

Notice that the spectral lines are not vertical. This is caused by the movement of the meteor. The deviation from the vertical is called slant and is defined as the slope

$$\text{Slant} = dx/dy$$

For the correction of the slant the rows are shifted horizontally in proportion to the row number.

$$Dx = \text{slant} * (y - y_0).$$

$y_0$  is chosen as the center of the selected rows for adding the spectrum.

Therefore the spectrum is shifted only minimally by the slant operation.



*Spectrum after slant correction, slant = -0.2, spectral lines "rotated" anti-clockwise.*

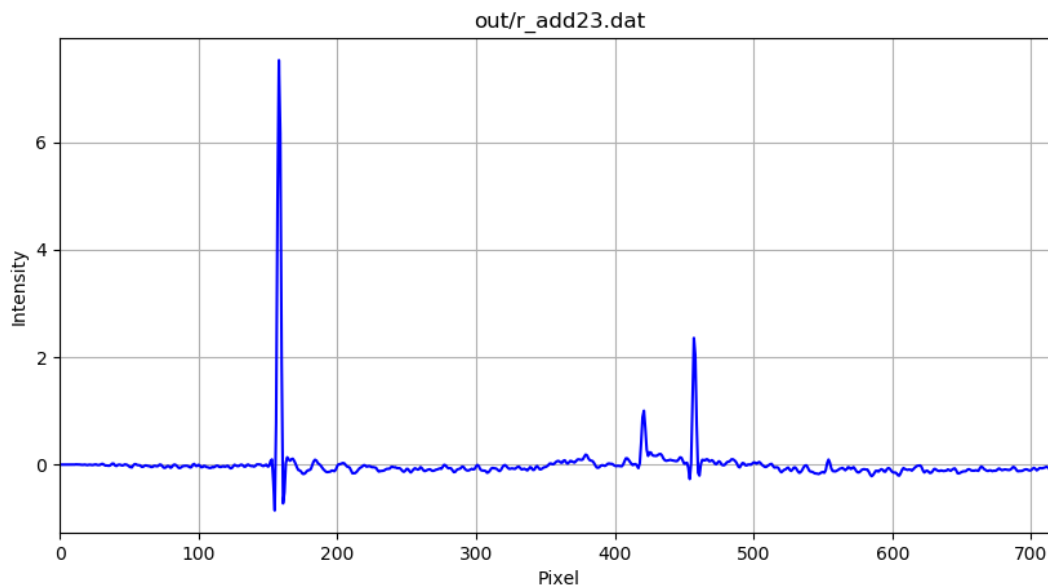
It is advisable to perform the operations in this order and adjust the row selection before the slant correction.

With these corrections, the processing may continue with the spectrum extraction.

### Spectrum extraction

The sum image can be converted to a 1-d spectrum by adding a suitable number of rows of the image which contain the spectrum. The number of rows is determined by the width of the spectrum. Ideally one wants to add all the rows which contain the information but avoid the rows which contain only background noise. The choice is somewhat arbitrary and depends on the skill of the operator, but not too critical. There exist methods of optimal extraction of the spectrum by measuring the width of the spectrum and signal to noise ratio in comparison with the background. This has not been implemented yet.

The result of this summing over rows is a 1-d spectrum: intensity versus pixel.



With the known dispersion  $\text{disp0}$  [nm/pixel] wavelength intervals can be calculated by measuring separations in pixels of prominent features (zero order, Na-line, Mg-line etc.). This helps in a coarse assignment of spectral features.

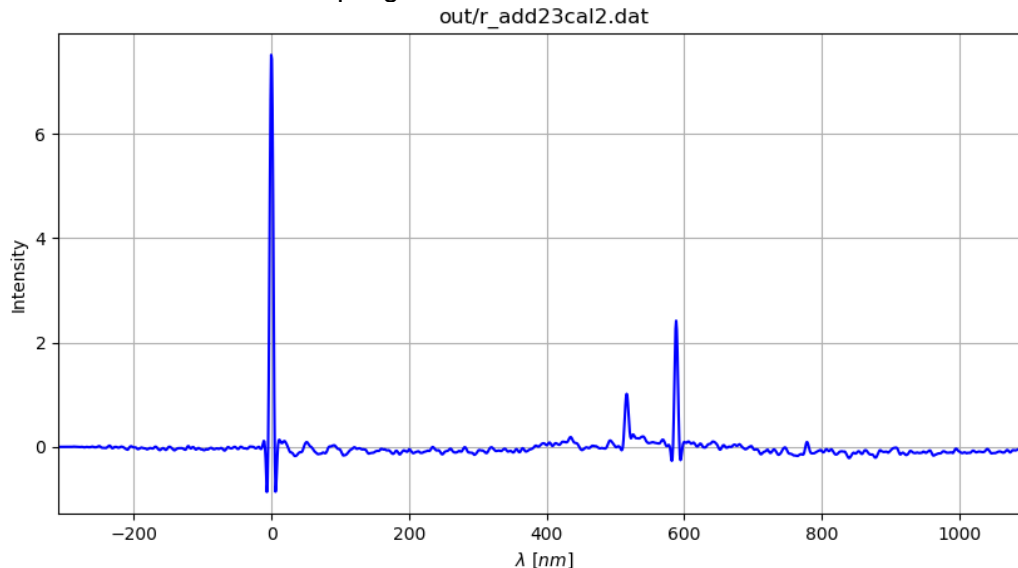
### Spectrum calibration

In principle, the meteor spectrum can be calibrated in wavelength by setting the zero point at the zero order and convert the pixels to nm by multiplying the distance in pixels of a spectral line from the zero order by the factor  $\text{disp0}$ , the (inverse) dispersion. In practice, some changes in alignment of the grating, changes of focus and focal length (for zoom lenses) may occur and it is advisable to calibrate the spectrum with the help of known spectral lines of the meteor.

Although in theory a linear function should fit the spectrum, in order to correct for small calibration errors a second order polynomial may give a better result. Before fitting a higher order polynomial it is advisable to check if a lower order polynomial works as well. Checking with more lines than required for a fit is always advisable in order to be able to eliminate measurement errors or assignment of the wrong lines.

## Plotting the spectra

Once calibrated, the spectrum can be plotted and saved as \*.cal.dat file. This is a common data format, a text file with two columns for wavelength and intensity, each data point in a separate row. Some programs which read spectra expect a constant wavelength interval; therefore the spectra are resampled with a constant wavelength interval, about half as large as the average original sampling interval. Quadratic interpolation is used. This helps to prevent loss of information with the resampling.



*Calibrated spectrum, the zero order peak is at 0nm, the Mg line at 515.5 nm and the Na line at 589 nm are also clearly visible.*

For the calibration in this example a linear fit of the zero order and the two spectral lines of Mg and Na were used.



## Python Installation

Python comes in different versions. Apart from the base version it requires additional packages, in particular:

Numpy  
Astropy  
Etc.

Some versions are not quite compatible. For that reason it is highly recommended to install a pre-packaged version, such as Anaconda, unless you are familiar with Python installing packages.

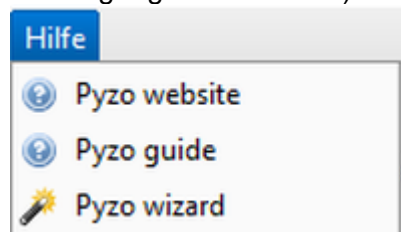
For running and editing Python scripts you need a development environment (IDE). I use PYZO, as recommended by Giovanni Leidi. It is installed from <http://www.pyzo.org/start.html>.



To get started with Pyzo, you need to install the Pyzo IDE (in which you write your code) and a Python environment (in which you run your code).

Download and install it. Pyzo recommends installing Miniconda afterwards. I had some problems with Miniconda and other packages; therefore I recommend using Anaconda instead. The installation of Anaconda is described next. You are free to use whatever version of Python you like, but do not complain if it does not work (**do not use Python 2.7; it does not work with my script**). I strongly recommend using Python 3.6, which worked well for me.

If you are not familiar with Pyzo, you may look at the Pyzo wizard. (There you can also change the language of the editor). Also the guide or website are quite instructive:



### Installation of Anaconda

The advantage of using Anaconda is that it comes with almost all of the required packages needed for running the spectroscopy scripts. This saves a lot of trouble with different versions of Python libraries. Important: use the **Python 3.6 version** (version 2.7 and 3.7 **do not work!**)

Go to <https://www.anaconda.com/download/>

Install the 3.7 version (64-bit in my case). The 3.6 version also works, if already installed:

## Python 3.7 version

Download

64-Bit Graphical Installer (466 MB)

32-Bit Graphical Installer (423 MB)

Follow the tips for installation

<https://docs.anaconda.com/anaconda/install/windows>

- Required system architecture: Windows- 64-bit x86, 32-bit x86; MacOS- 64-bit x86; Linux- 64-bit x86, 64-bit Power8/Power9.
- Install Anaconda in C:\, otherwise you may have problems finding it on your computer (makes it easier to write path to anaconda, but you are free to install it elsewhere).

Note: Install Anaconda to a directory path that does not contain spaces (such as: Program Files) or unicode characters.

This will take a while. At this point, if everything has gone well the Pyzo IDE should recognize without telling it the environment Anaconda you chose and you can start writing in python. To install packages simply type in the shell (you will find it on the right in Pyzo): install numpy, install scipy, install astropy and so on.

### Load libraries

If the scripts do not run, it may be that a library is missing. You have to load libraries, which are not contained in your Python distribution.

With Anaconda the command is, e.g. for the installation of PySimpleGUI:

```
>>> conda install -c conda-forge pysimplegui
or
```

```
>>>conda install -c conda-forge pillow (for the image library PIL)
```

You find the installation command by going to the Anaconda cloud <https://anaconda.org/> and search for “PySimpleGUI” or “PIL”.

The installation checks for the compatibility with the different already installed packages and makes the necessary updates.

In addition you need to copy other scripts with additional functions (l\_config2.py, m\_specfun.py etc.).

Fortunately most of the needed library packages are already contained in Anaconda, namely:

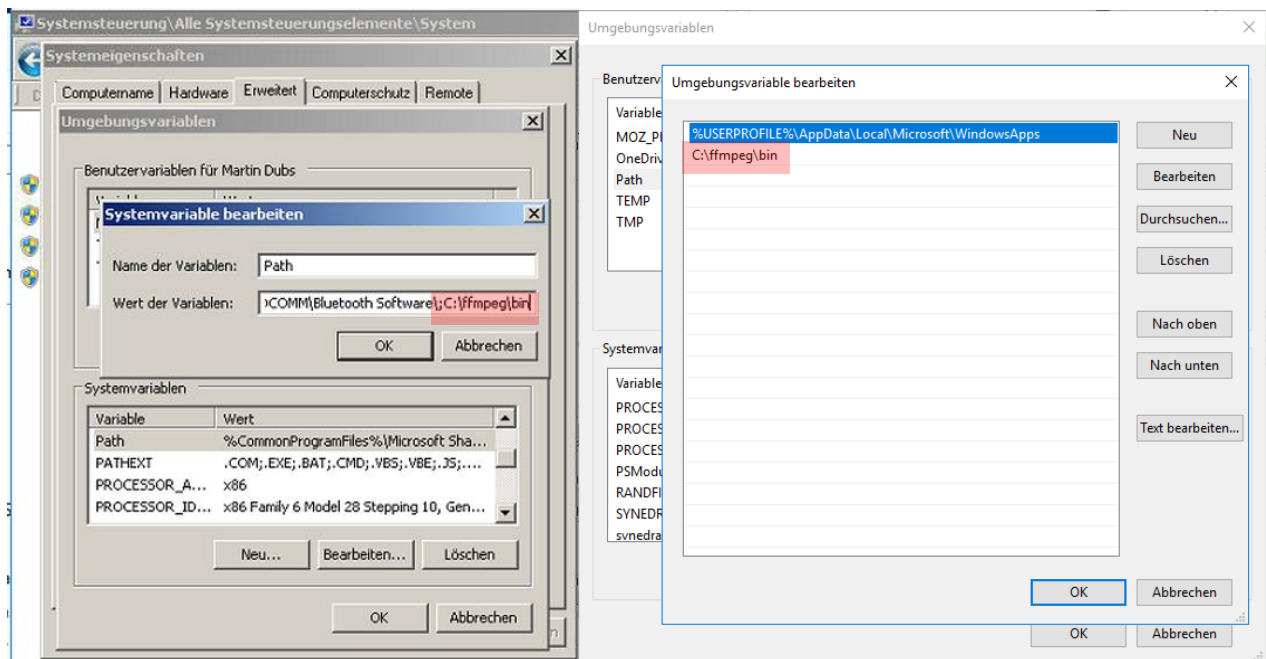
- numpy
- matplotlib
- astropy
- scipy
- skimage (is contained in scikit-image)

### Install ffmpeg

The script uses ffmpeg.exe for the conversion of the AVI-files to BMP-images.

Download ffmpeg.exe (e.g. from: <https://ffmpeg.zeranoe.com/builds/>) select the correct Windows version (32 or 64-bit), e.g. ffmpeg-4.0.2-win64-static.exe and install it. Add the ffmpeg.exe directory to the Windows system path variable. This is slightly different in Windows7 and 10. In system variable, double click on the path and add the location of ffmpeg.exe:



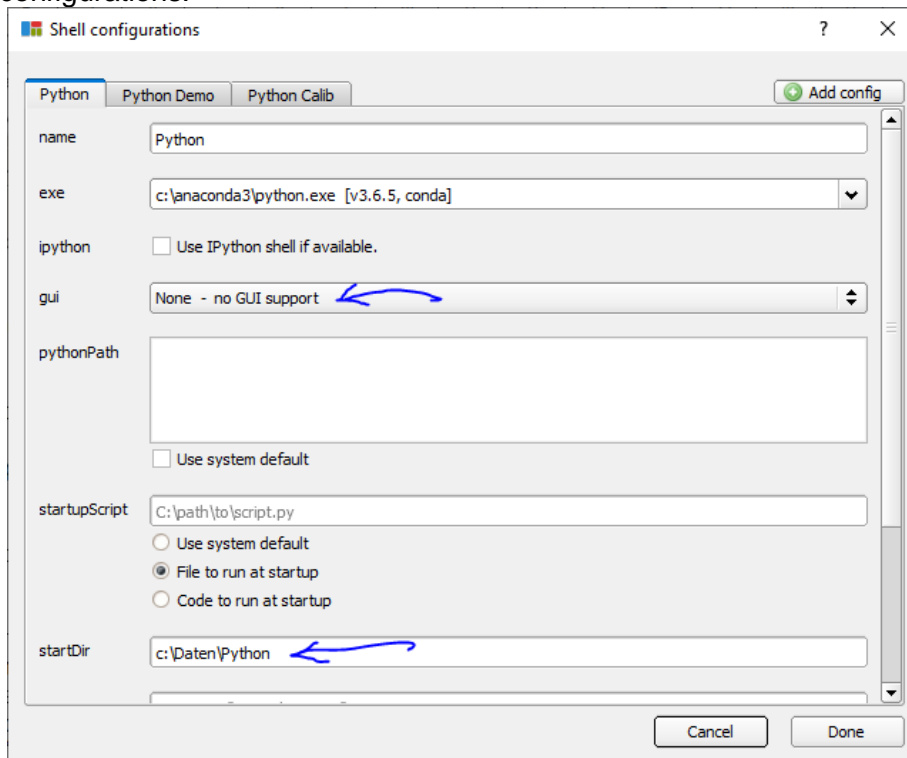


## Setup Pyzo

The python script should run in different environments. I show how to run it with Pyzo, which is fairly simple but quite convenient. I am new to python, so I cannot tell how it compares with other environments.

When starting Pyzo, one can select several windows. Of these the shell is the most important, there the script is run. Also useful is the editor window, where one can make changes to the scripts and the workspace, where one sees the actual values of variables while running the script. Other windows can be selected under Tools:

When starting Pyzo, one should check the configuration, under Shells – Python... - edit shell configurations:



Important are the path to the python.exe file, which with correct installation is automatically set. Gui should be set to none, otherwise some strange things happen with the plots. The start directory should be set to the directory where the python scripts such as m\_pipe6.py and m\_pipefun5.py are located, C:\Daten\Meteor\Python in my case. The data and results I put in subdirectories of this "root" directory. In this directory some other files are needed, i.e. mset.ini, the configuration created in the calibration script (See: Calib\_Python\_manual.pdf).

Important: Before running this script, run the laser calibration l\_calib6.py or edit a configuration file mset.ini with the correct parameters for your camera – lens - grating configuration (e.g. if you already made your calibration with EXCEL). It is good to start with the calibration first, until you are a bit familiar with Python.

### **Edit text files**

At some points it may be necessary to edit text files. The Windows editor is not particularly useful, it is recommended to use a better editor, such as Notepad++:

<https://notepad-plus-plus.org/>

It can also be used to view Python scripts with highlighting, if you do not want to execute the scripts.

### **Installing Python in LINUX**

The installation under Linux (Ubuntu 18.04) without Anaconda works also:

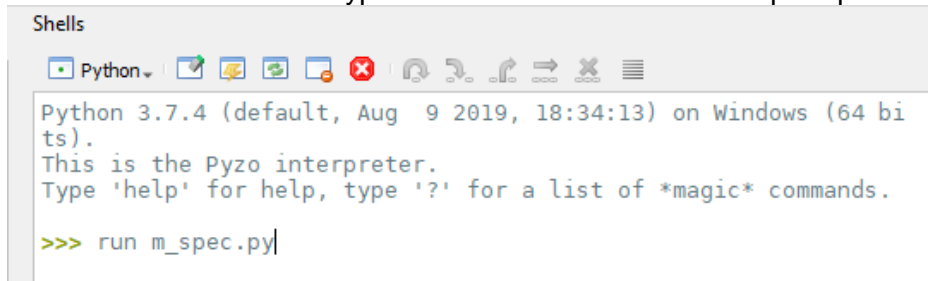
This has not been tested yet for m\_spec, but worked well for M\_calib according to a Linux user of Python.

## Processing of meteor spectra, step by step


### Starting the script

There are two methods to start a script. Either you open the script, at present m\_pipe6.py, in the editor window, select it and start it with: Run – Run file as script (Ctrl shift E)

The second method is to type in the shell window after the prompt >>> run m\_pipe6.py ↵ (enter)



```
Shells
Python
Python 3.7.4 (default, Aug 9 2019, 18:34:13) on Windows (64 bi
ts).
This is the Pyzo interpreter.
Type 'help' for help, type '?' for a list of *magic* commands.
>>> run m_spec.py
```

If you do not see the prompt, you are running something already, then it may be necessary to restart with: .

When you start the script for the first time you will probably see some error messages of the type ... package not found.

Required packages are included in Anaconda:

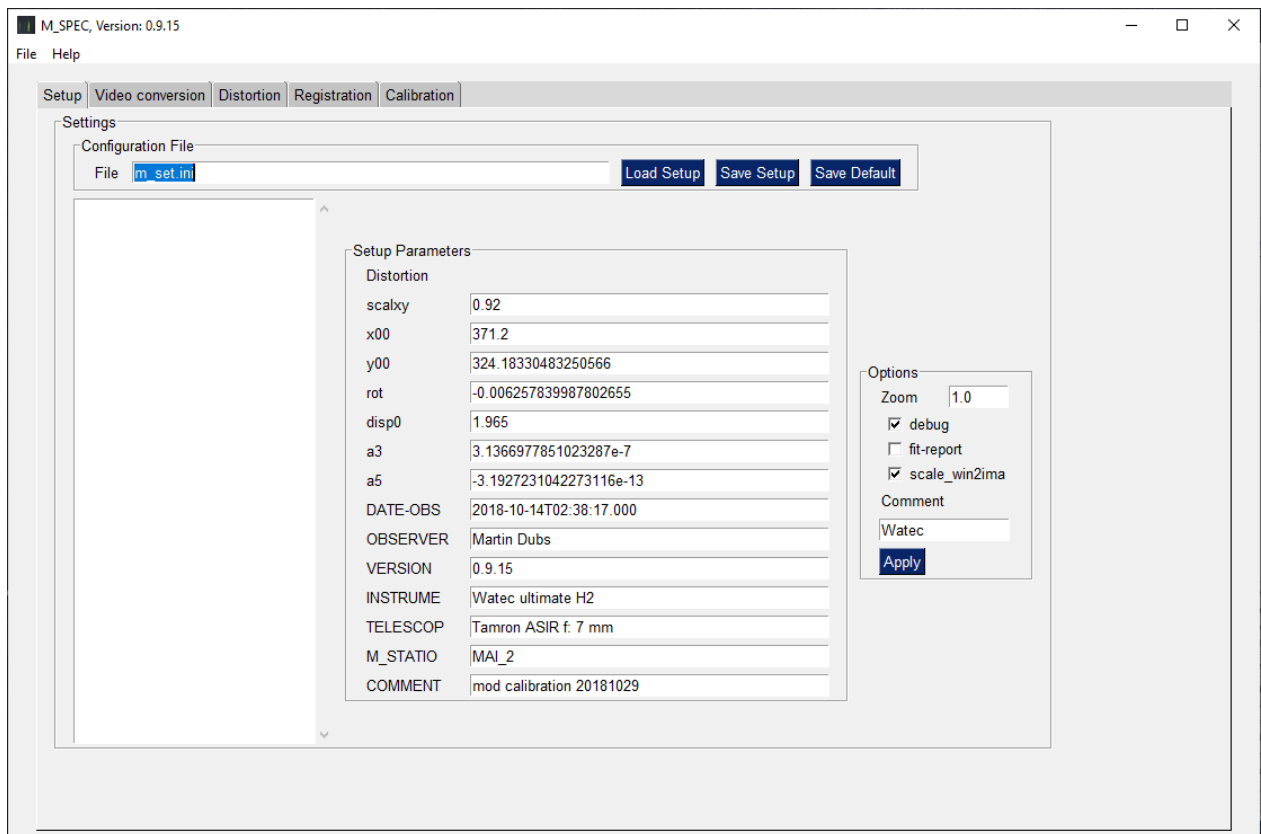
- numpy
- matplotlib
- astropy
- skimage (contained in scikit-image)
- pathlib
- scipy

If I have forgotten something, look in Anaconda if you can find it and install it (send a little note to <https://meteorspectroscopy.org/contact/> so that I can fix this manual).

Once all packages are installed, you should be able to start the script and the following message appears:

```
>>> run m_spec.py
pysimplegui 4.15.2
version m_spec, m_specfun 0.9.15 0.9.15
```

At the same time the window for processing the meteor spectra appears:



The window allows to select the different processing steps in separate tabs. At startup the tab Setup is shown, where the parameters from the configuration file are displayed. By default, the configuration file `m_set.ini` from the work folder (where `m_spec.py` resides) is loaded. This file is also stored when you leave the program with File – Exit in the menu or you can store changes to it with “Save Default”. This file contains many program settings and, most important, also the distortion parameters from a calibration with `s_calib.py` or `m_calib.py`. In the case you have different cameras or setups, it is advisable to store them in separate configuration files. The content of the configuration file you see in the left memo. Since the same configuration file is used for the calibration and for the spectrum processing, it starts with parameters for the laser calibration, followed by the distortion parameters and fits-header values and finally the settings of `m_spec` under [Options].

The most important setup parameters are shown in the center frame. Most values can be edited, some values such as `DATE_OBS`, `VERSION` and `M_STATIO` are updated by the program in case you use UFO Capture for recording the meteor spectra (`VERSION` shows the actual version of the script as written to the fits-header)

## Fits-header

A good practice in astro photography and spectroscopy is to record information about the image content in the Fits-header, in particular observation time and date, observer, location and equipment. In the case of the orthogonal transformation also the used transformation parameters may be useful for a later check on the processing history.

`DATE-OBS` is a standard key, its value is derived from the UFO capture filename.

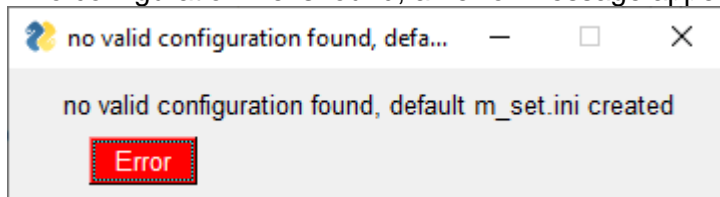
`M_STATIO` is also derived from this filename.

`D_DISP0` is the dispersion from the laser calibration. If you have a better value from actual meteor spectra, you may enter the correct value here.

Others, such as `COMMENT` you may edit to your taste.

If you save the changes in the default directory with ↵ you can use them next time you start the script. It is good practice to save them also in the output directory, together with the other processed files, in case you would like to reprocess some files later on.

If no configuration file is found, an error message appears:



Load a valid configuration file or edit the default values with correct values.

If you use different cameras or lenses, make sure you are using the correct configuration. I use different directories for different cameras. Also make a backup of the configuration in a safe place, it is possible to overwrite or erase the file by accident, if something goes wrong.

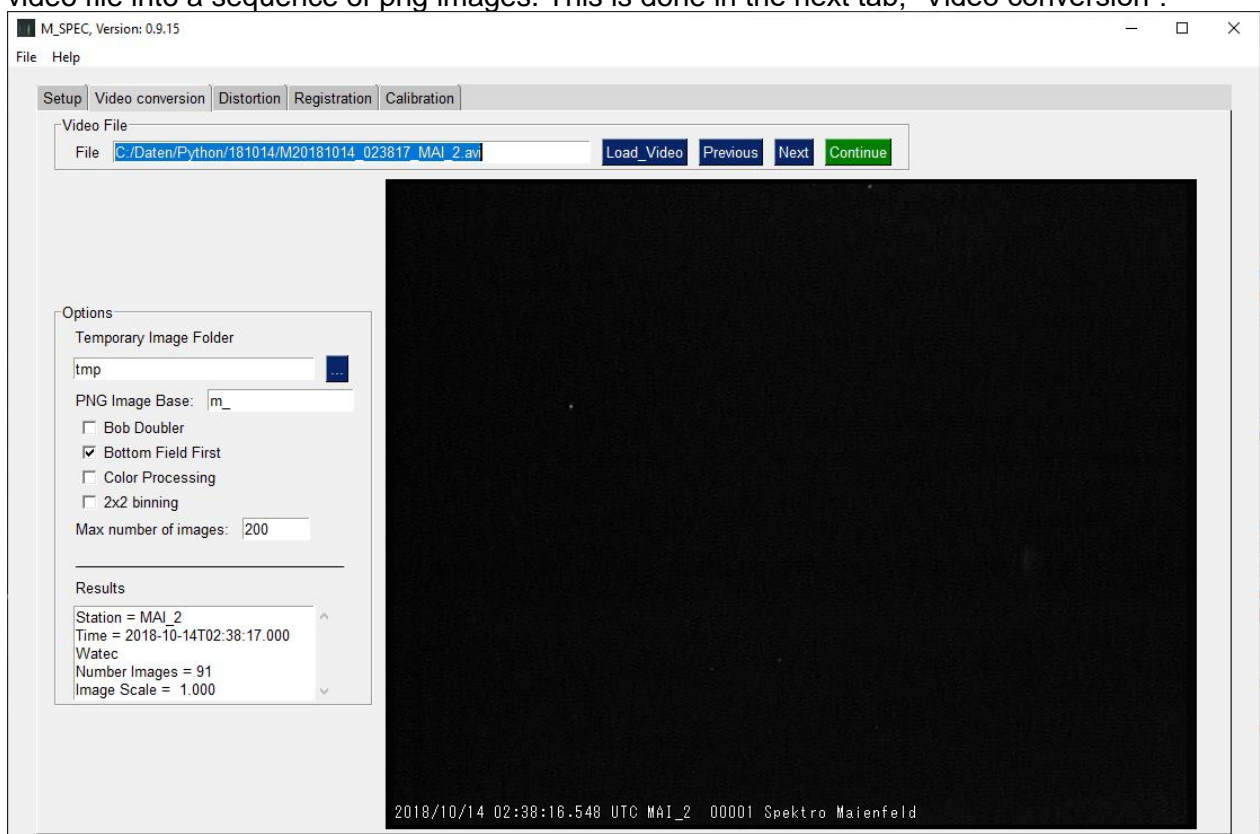
(The script is not yet foolproof, not even without bugs. While improving it constantly, it is possible that new errors are introduced, typical for scripts only tested by the creator).

On the right side is a frame for options (fit-report and debug are not active, used in m\_calib.py).

With zoom you can adjust the scale of the displayed images (works after video conversion or distortion). If it does not work correctly, save configuration and restart program. Another possibility is to deselect scale win2ima, then you can adjust the window size manually and the images should fit. If they do not fit, restart the script. Check the other settings, then save the configuration under a meaningful name.

## Conversion of \*.AVI to PNG

With the correct configuration we can start the meteor spectra processing, first converting the video file into a sequence of png images. This is done in the next tab, "Video conversion":



**Before** you load the video file you have to set the options!

- You can select a folder to store the images. Since they are used only once they are stored in a temporary folder tmp.
- You can also choose the filename of the images, the index starting from 1 is added: m\_1.png, m\_2.png, ...

This is the same naming convention as in IRIS, so you may use these files also in IRIS,

for processing steps not included in this pipeline. You can change these names, but this is not really useful as these are only intermediate files you need once or a few times if some errors occurs later in the processing.

The images are erased before a new video conversion, so do not select a folder with images you want to keep.

- With Bob Doubler you can read half frames for interlaced videos, in this case you also have to choose which field is read first. For Watec cameras this is usually Bottom Field First. With the wrong selection, the meteor jumps around and it is difficult to register the images correctly. This increases time resolution and therefore spectral resolution if the meteor has a velocity component parallel to the dispersion direction (recommended for WATEC).

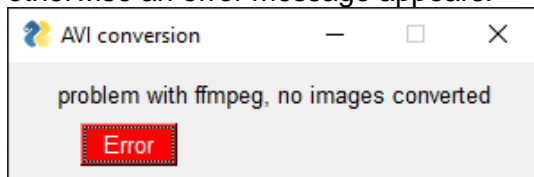
**Important:** the calibration parameters are adjusted automatically; in particular the factor  $\text{scalxy}$  is multiplied by 2 to correct for the different aspect ratio.

- 2x2 binning is useful to reduce the file size for 4k images (for 4k color cameras the color pixels are often interpolated, so you loose little information by reducing the size to 2k).

**Important:** The distortion parameters are not adjusted automatically, it is assumed that you use the same binning as for the calibration (keep 2 sets of calibration parameters for 2k and 4k images!)

- It is also possible to limit the length of the video sequence. This is helpful in case of a corrupted file or when you want to save time or disk space.

With the options set, you can load an AVI-file. The conversion starts automatically and the button Continue turns green when completed. For this step ffmpeg.exe has to be installed, otherwise an error message appears:



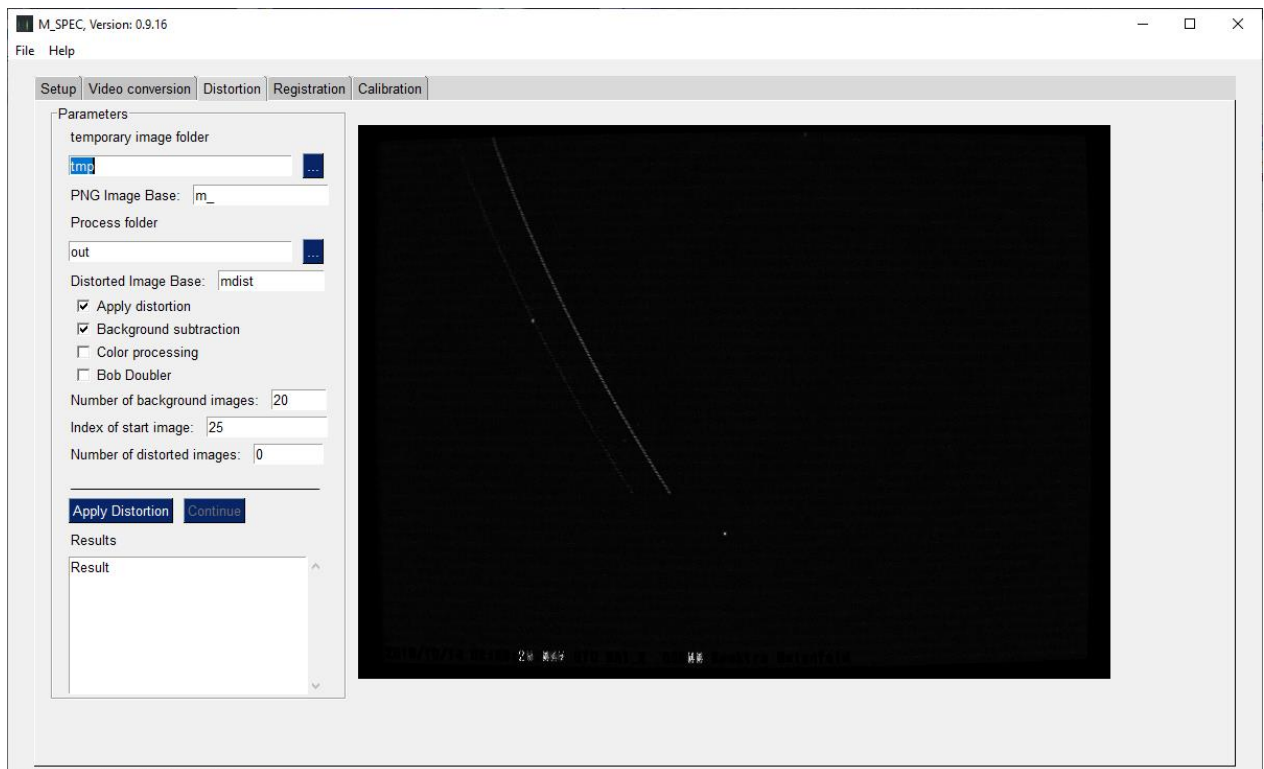
Notice that the path of the selected file is written with forward slash: /, so users of LINUX should not have a problem with it. In most cases also a backslash \ is accepted. I personally use a different directory for each meteor, with the image files having standard names.

In the memo at left some details of the conversion are given. With Continue you go to the next tab, "Distortion". Before you do that you may look at the images with the very basic image browser. With the Next button you jump from frame 1 to 25 and then advance in steps of 1. With Previous you go backward in the sequence. This is because the first 25 images only contain background and the meteor appears around frame 25. With Bob Doubler selected, the Next button selects field 50 after the first field, corresponding to frame 25.

## Distortion

In the distortion tab you can apply the transformation to an orthographic projection, which linearizes the spectra. For this to work, you have to set the correct distortion parameters in the setup.





In addition you have to set some additional parameters in the frame at left.

- Temporary image folder, PNG image base: if you have just done the video conversion, these values are preset, otherwise you can choose them to convert other images. Note that they should conform to the naming convention, file base index .png, eg. m\_1.png, m\_2.png, ...
- Process folder and distorted image base: Folder and name for the processed images. Naming convention as above. Notice that files with the same name are removed from this folder before a new distortion is performed. This prevents a mix of different spectra. For the process folder I use a separate folder for each date, where I also store the AVI file(s), setup-file, processed spectra and logfile(s) for later reference.
- Apply distortion, background subtraction: These are normally checked, but you can do only one operation if desired.
- Color processing is mostly used for making nice images, the spectra are evaluated from b/w images. Use b/w, it is three times faster than colour processing and for the extraction of the spectra you have to convert the images anyway.
- Bob Doubler: This is set the same value as in Video Conversion. Be careful to set the right value when you do a distortion correction at a later time. It is advisable to do the distortion right after Video Conversion.
- Number of images: For background and start images use the default values. The number of distorted images is calculated from the number of available images after video conversion. Select a reasonable number. It will be adjusted to a lower value, if not enough images are found.
- Background images:  
As described in the general processing section, a background image is created from the first second of video. The default value of the number of images used for the averaging is given for a pretrigger of 1 sec and a frame rate of 25 images/second. With bob doubling, the image rate (fields/second) is doubled, therefore also the double number of images can be used for the background. Make sure that the default value does not use any images with the meteor appearing early.

Once everything seems ok you press the button "Apply Distortion" and wait for the end of the processing. This is indicated by the button Continue turning green (means enabled) and some information is output in the result memo at left bottom:

and the peak image of the transformation is displayed:

Number of distorted images: 67

Apply Distortion Continue

Results

67 images processed of 67  
M2018-10-14T02:38:17.000\_MAI\_2  
check time!  
process time 13.61 sec  
for single distortion: 0.16 sec

The process time is the time between the start of the background image and the end of the transformation. The log shows also the average process time per image.

The contrast and brightness of the peak image has not been adjusted, so you should not be disappointed by this result. If you continue you can adjust the brightness of the transformed images in the registration tab.

### Registering the meteor spectra

As explained above, the meteor spectra of the different images can be added after registration, because they are all aligned parallel to a common dispersion direction and they all have the same linear dispersion. This is because of the transformation in the previous step. The meteor movement causes a displacement of the spectra. This is removed by registration, which will be explained in detail.

M\_SPEC, Version: 0.9.16

File Help

Setup Video conversion Distortion Registration Calibration

Registration

Process folder out Previous Next Current Index: 3 Max Images: 67 Darker Brighter

Parameters

Distorted Image Base: mdist  
Registered Image Base: r  
Index of start image: 3  
Number of registered images: 65

Sel Start Sel Last  
Register Show Sum show registered  
r\_add Load Radd  
Add Rows Save raw spectrum Calibrate

Results

Result

The idea is to identify a clearly visible line in the meteor spectrum and shift the following images so that this line is in the same position in all images. The position is determined with a Gaussian line fit, similar to the determination of line position in the calibration script. The problem is that this determination of line centre is not fool-proof, it may be disturbed by a missing line in one or several images, by saturation or overlap with something else. If the registration does not work the first time it is possible to repeat it with different parameters. Important is the selection of the first image used for registration. If the 1<sup>st</sup> image is not useable a different one may be chosen.

The registration frame at the top is a simple image browser, which helps to select the correct images.

- The process folder and file base are preset from the distortion tab. If you want to register some other files you can choose the folder with the button “...” and change the base file name of the images you would to register (here mdist, for the files mdist1.fit, mdist2.fit, ..., with the now familiar file naming convention)
- You can also change the name of the registered file base, r for r1.fit, r2.fit, ...
- With the buttons “Next” and “Previous” you can click through the images to find a suitable start image, here mdist3 was selected
- With the buttons “Brighter”, “Darker” the image contrast can be changed
- The current index is shown and the number of images can be set. If this is larger than the actual number of images it is adjusted automatically.

Notice that the numbering of the meteor spectra mdist is offset from the image number in the original video file by: first meteor image – 1

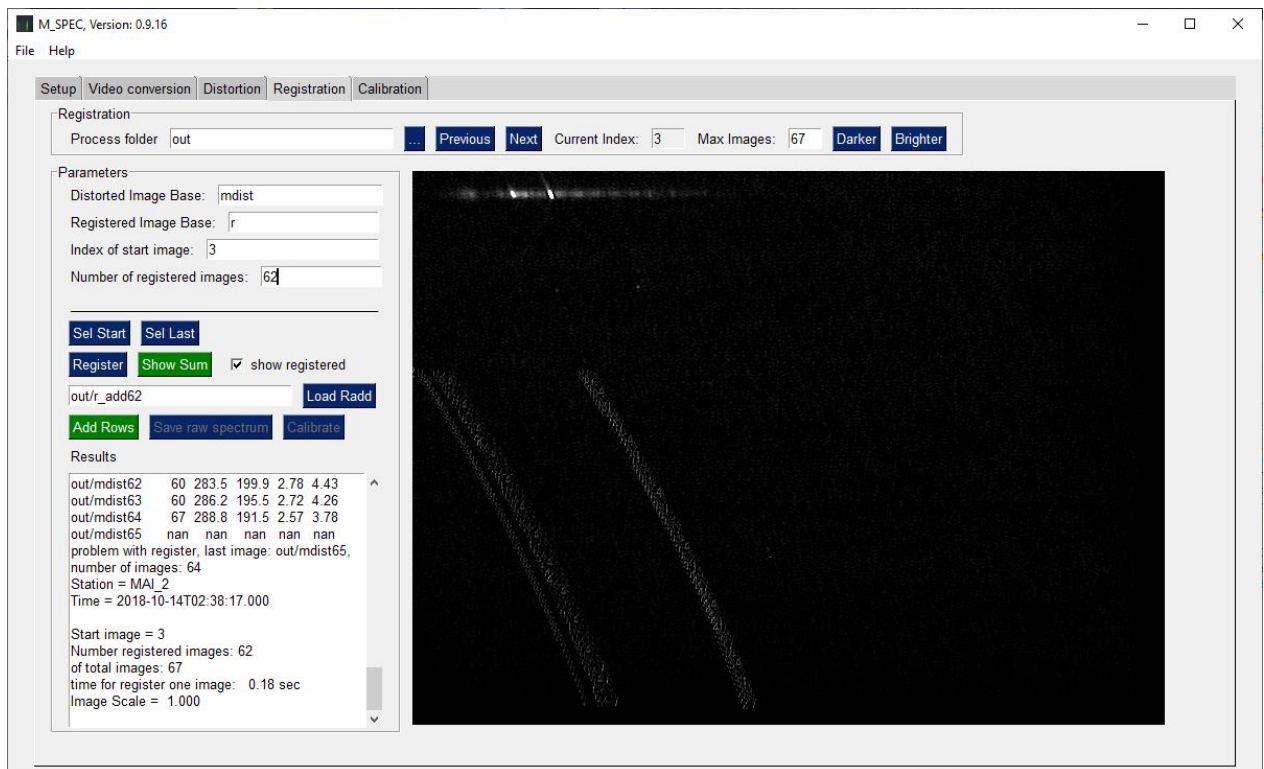
If you have found the correct start image, press the button “Sel Start” or enter the image number (here 3) into the text box “Index of start image”. Browse through the images with next until your last image to register and press “Select last” or enter the corresponding number of images (not the index) into the box “Number of registered images”.

With all settings done you can start the registering procedure by pressing “Register”.

A new window appears with the start image. Here you can draw with the mouse a rectangle enclosing the zero order or whatever spectral line you would like to use for the registering:



If you do not see the zero order either the image is too dark or you selected the wrong image. With Cancel you can try different parameters. With Ok, the registering starts. In case of success you see the sum spectrum of the registered images:



With Previous and Next you can look through the registered images, Show Sum shows the sum image again and by unchecking “show registered” you can browse through the original images. In the Results section at left you can check if the registering was successful. A list of the selected files, line intensities, positions and line widths of the Gaussian fit is given. Apparently there was a small problem with one of the last images. Try again with a different (smaller) rectangle size or leave it as it is. The size of the rectangle should be at least as large as the size of the line and the movement from frame to frame (whichever is larger), but not too large. This need a bit of practice, the algorithm to determine the line centre (x, y) could probably be improved. Notice that in this particular case not the zero order was chosen but the Na-line, often one of the strongest lines in the spectrum. The zero order was outside of the image area to the left.

## Spectrum extraction

Now you are almost there. Adjust the contrast of the coadded spectra until you see the spectrum strongly but only the strongest lines saturated. Continue by pressing “Add Rows” or load a previously registered image by pressing “Load Radd”. A new window with the sum spectrum appears. Next you can select the rows you would like to add, by dragging the mouse over the width of the spectrum:





For the width of the selection it is important to add all the rows which contain the spectrum, but not the rows which contain only noise or other signals (The stripes in the lower half were produced by the text in the images, avoid these). If you do not like your choice, try again. In this case mdist3 to mdist67 were registered as r1 to r65 (notice the IRIS compatible naming) and added to r\_add65.fit. In this particular spectrum you may notice that the spectral lines are not vertical because of the movement of the meteor. You can improve this by processing fields with half exposure time (bobdoubler) or by correcting the slant (making the lines vertical). More about this later.

A look at the FITS-header (e.g. with IRIS: image info or Audela:) shows that the information about the time and observation as well as the distortion parameters are stored for documentation.

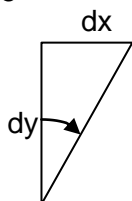
```

7% FITS Header (visu1) - C:/Daten/Python/out/r_add65.fit
BGMEAN = 1.456867089189018e-006 mean value for background pixels adu
BGSIGMA = 24.378311715904609 std sigma value for background pixels adu
BITPIX = -32 array data type
BSCALE = 32767
BZERO = 0
COMMENT = mod calibration 20181029
CONTRAST = -1.071756579981142e+007 Pixel contrast adu
D_A3 = 3.136697728223220e-007
D_A5 = -3.192723049914100e-013
D_DISPO = 1.9650009870529199
D_ROT = -0.0062578399665653697
D_SCALXY = 0.92000001668930098
D_X00 = 371.20001220703102
D_Y00 = 324.18331909179699
DATAMAX = 7986.99560546875 maximum value for all pixels adu
DATAMIN = -662.0494384765625 minimum value for all pixels adu
DATE-OBS = 2018-10-14T02:38:17.000
EXTEND = T
INSTRUME = Watec ultimate H2
M_NIM = 65
M_STARTI = 3
M_STATIO = MAI_2
MEAN = 1.7485718958855889 mean value for all pixels adu
MIPS-HI = 243.783111572 High cut for visualisation for MiPS adu
MIPS-LO = -146.269866943 Low cut for visualisation for MiPS adu
NAXIS = 2 number of array dimensions
NAXIS1 = 720
NAXIS2 = 530
OBSERVER = Martin Dubs
SIGMA = 58.772258764504933 std sigma value for all pixels adu
SIMPLE = T conforms to FITS standard
TELESCOP = Tamron ASIR f: 7 mm
TT1 = IMA/SERIES STAT TT History
VERSION = 0.9.16

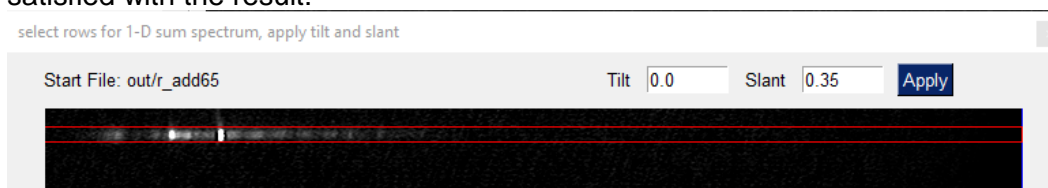
```

With Ok you could select the rows and the resulting 1-d spectrum is stored as r\_add65.dat, with the summed intensities as a function of column number (starting at column 0 to 719) for further processing with any spectra software (ISIS, Vspec, EXCEL ...). Before doing that you should read the next paragraph.

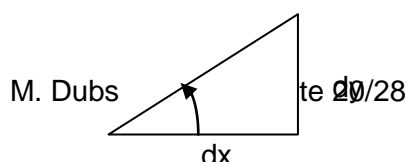
You can adjust tilt and slant by typing a suitable value into the text boxes for Tilt and Slant. For a clockwise change of slant you enter a positive value. The value is the tangens of the angle or the slope  $dx/dy$  applied to the image



You confirm the value with Apply. The modified image is displayed. If you do not like the result you may type r and start all over or you type a new value for the slant and ↵ until you are satisfied with the result:



For the tilt the function is very similar. It is defined as the the tangens of the angle of the direction of the dispersion with respect to the x-axis or the slope  $dy/dx$ :





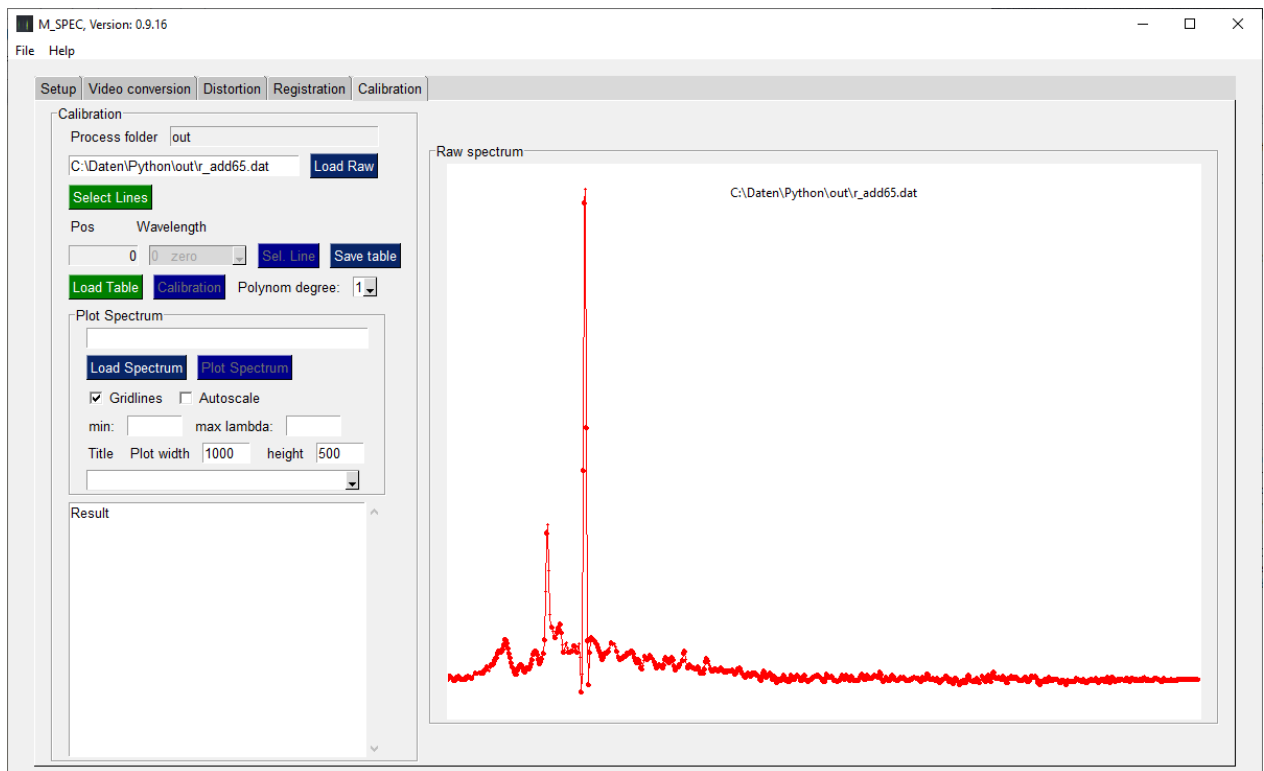
A positive value of the tilt rotates the spectrum anti-clockwise, as shown for demonstration in the next image:



Of course, in this case a tilt of zero is the correct value, since the spectrum was aligned initially.. Notice that you can restore the original image by entering 0 for tilt and slant. When everything is adjusted, you type Ok to save the modified image and the 1-dimensional spectrum obtained by adding the rows column by column. The corrected image is stored as `r_add65st.fit` for processing with other software and the 1-dimensional spectrum obtained by adding the selected rows is stored as `r_add65.dat` (uncalibrated spectrum). You may also save it under a different name with "Save Raw Spectrum"

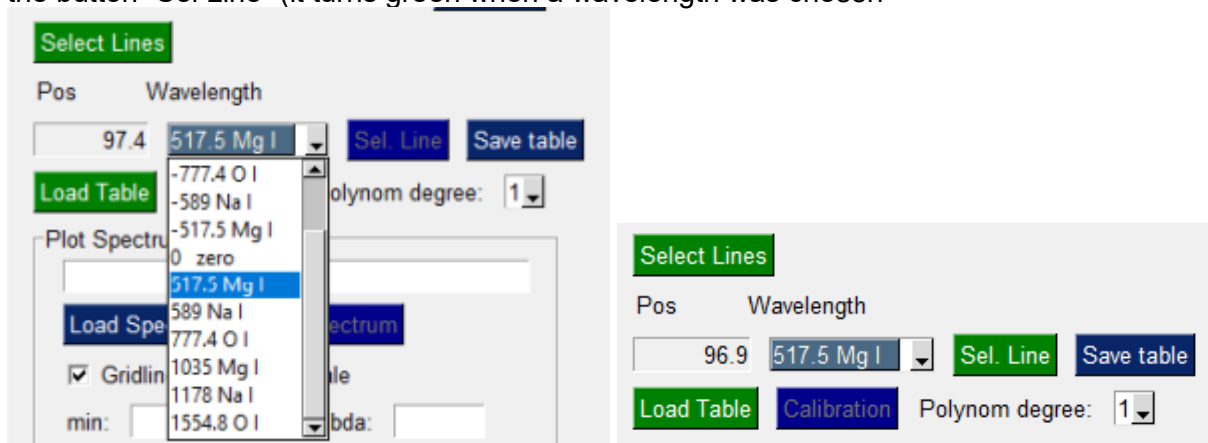
The slant is needed for correction of the meteor moving diagonally across the image field. Making the spectral lines vertical improves the spectral resolution. The tilt is used to correct for misalignment of the dispersion direction. This can also be corrected by adjusting the angle in the setup parameters. If the grating rotates slightly in front of the camera, a correction of tilt during the analysis is easier here, where the result is seen immediately. If a consistent tilt angle is observed in the spectra, it is advisable to correct the angle "rot" in `m_set.ini`. For small angles the tangens of the tilt is equal to the required correction of "rot", measured in radians. If you are satisfied with the result, you finish this process step with "Ok". This saves the corrected image as `r_add65st.fit`. The image is normalized to a maximum of 32767 ( $2^{15}-1$ ) in order to be viewed easily in IRIS. In the fits-header of this image the selected values of slant and tilt as well as the selected rows and the selected width (half width) are displayed: The original image `r_add65.fit` is simply the average of the added images, with 255 ADU's scaled to 32767, for readable display in any FITS-viewer.

By addition of the selected rows for each column the 1-dimensional spectrum is calculated and displayed when you press "Calibrate". This brings you to the last tab for wavelength calibration of the raw (Intensity vs. pixel) spectrum:



## Wavelength calibration

In the graph the raw spectrum is shown. Start the calibration with “Select Lines”. This activates the mouse clicks in the graph and starts a new table with wavelengths and positions of calibration lines. Select a line in the raw spectrum by dragging the mouse with pressed left button across it. 5 points around the maximum intensity are used for a quadratic fit. The position of its maximum determines the line position in pixels. In the text window Pos is the center of the selected range given. Make sure the highest point of the line is within the range indicated by green lines. Select the correct wavelength from the table “Wavelength” or enter a wavelength not listed in the table (you have to enter the table with a mouse click). Confirm the choice with the button “Sel Line” (it turns green when a wavelength was chosen



In the memo below the position, width (calculated from the parabolic fit and selected calibration wavelength with information about the line are displayed. Continue by selecting more lines (those already used are shown in blue). When finished, press “Save Table”. The list of positions and wavelengths is stored under the raw file name.TXT

# x lambda

95.78 517.50  
131.67 589.00

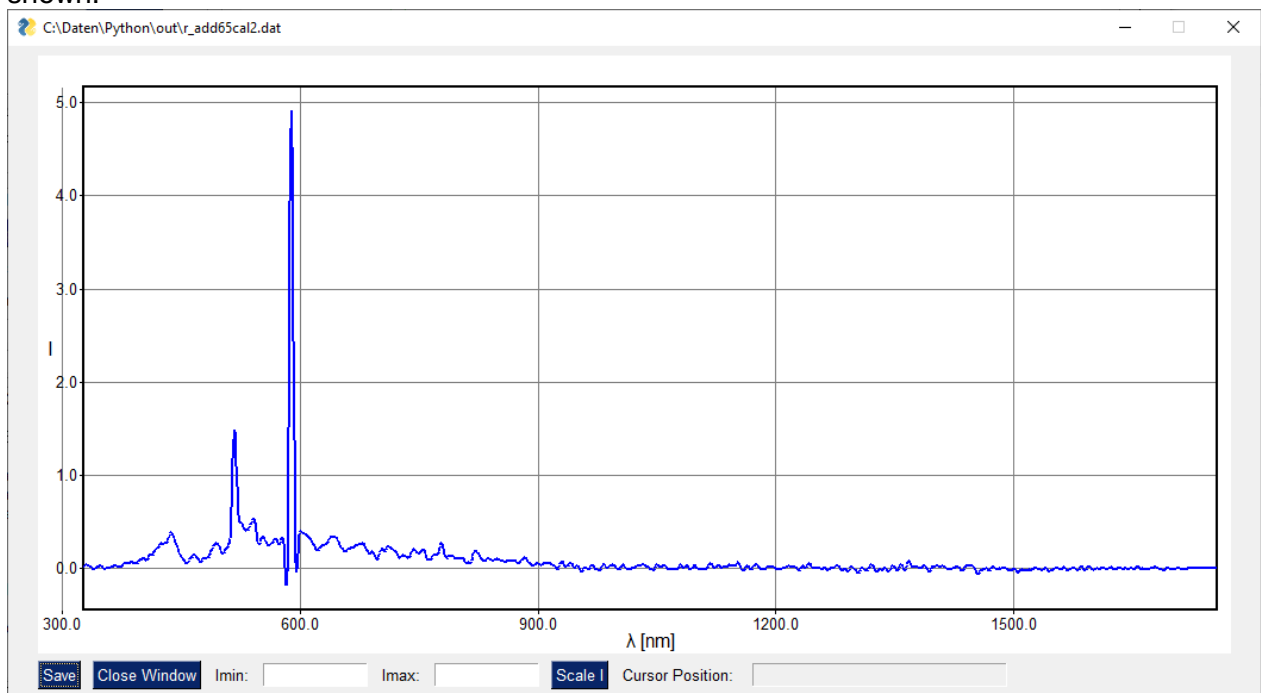
You may edit this table with a text editor, in case you have entered wrong lines or wavelengths. Load the raw spectrum again if you would like to make a new calibration. The table is overwritten if you do a new line measurement with the same raw spectrum.

With two lines only a linear fit is possible. Select 1 for polynom degree (default). The polynomial is of the form  $a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + \dots + a_1 \cdot x + a_0$ .

For the linear polynom:  $a_1 \cdot x + a_0$ , with dispersion  $a_1$  and offset  $a_0$ .

With "Calibration" (green if you have saved or loaded a table) a least square fit of the polynom to the selected line positions is done.

The error of the fit is zero, because with 2 lines a linear fit is determined, there are no additional lines for evaluating the error. The calibrated spectrum is saved as `r_add65cal.dat`, unless you give a different name to the raw spectrum. With "Plot Spectrum" the calibrated spectrum is shown:



You also have the possibility to select the plot range. The intensity scale you set in the Plot window, with "Scale I" you plot the new intensities. By closing the window and replotting it you also set the correct scales. The wavelength range and the window title are set in the main window. The title can be selected from a list of the latest converted avi-files and the latest raw spectrum or typed into the list manually. The spectrum can be saved with "Save". Choose a filename, a png image is saved. If no filename is entered, `r_add65cal_plot.png` or the corresponding raw file name + `"_plot.png"` is saved.

### Important:

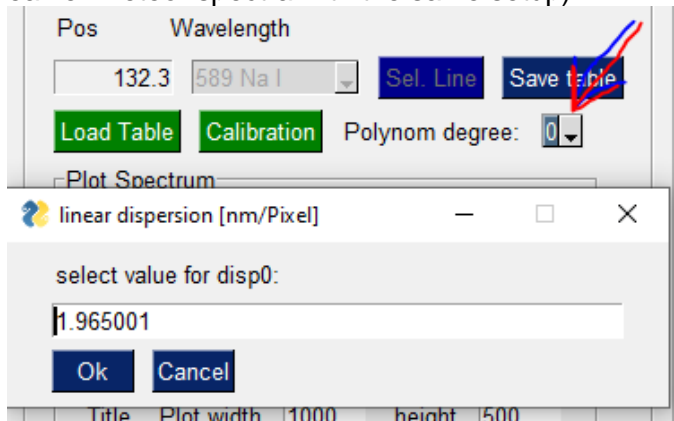
The script requires that the wavelength increases to the right, to higher pixel values. Therefore orient the grating in front of your camera that the first order is on the right side with respect to the zero order. The orders to the left of the zero orders have negative values, therefore the corresponding wavelengths are negative. Always enter the line wavelengths as `wavelength*order` (0 for zero order, `wavelength` for first order, `2*wavelength` for 2<sup>nd</sup> order, etc.).

### Single line calibration

The dispersion calculated from these two lines is similar to the dispersion from the laser calibration: `disp0 = 2.005`. However, it was derived from two lines in a narrow region, therefore the value is not very precise. In that case it may be better to use the dispersion `disp0` for the fit.

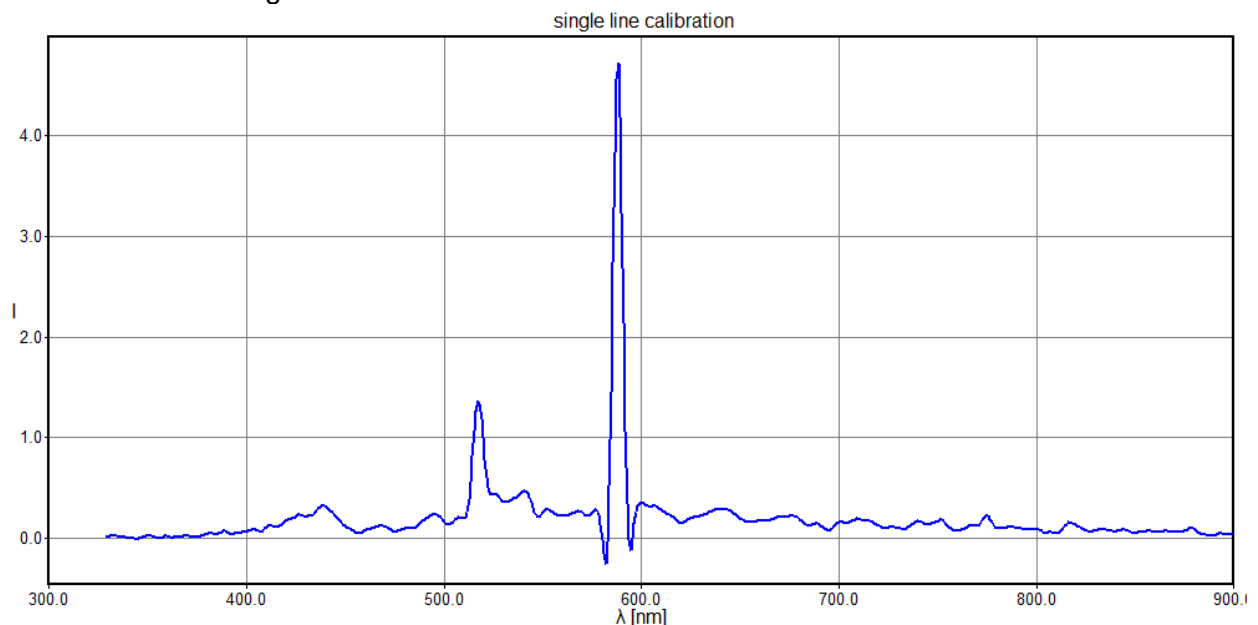
**This is done by entering 0 for the polynom degree:**

The value 2.005 from the setup file is suggested for disp0. In case you know the dispersion from other meteor spectra, you may enter here the correct value (1.965 was selected, based on earlier meteor spectra with the same setup):

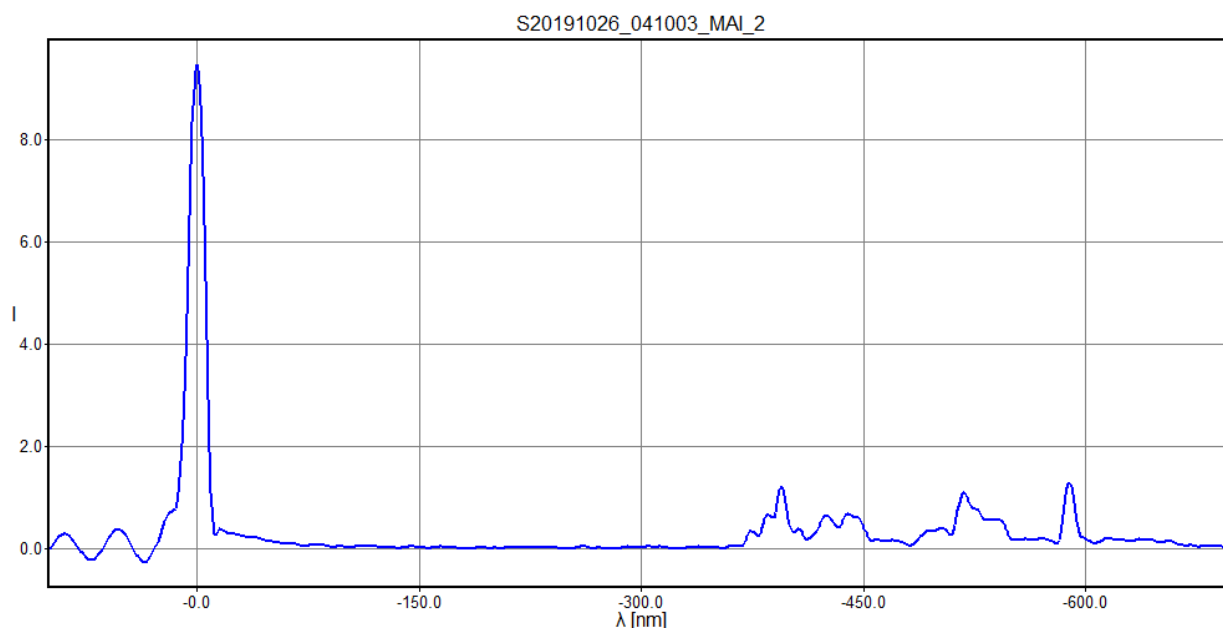


```
File C:/Daten/Python/out/r_add65.dat loaded
Pixel   lambda   fit   error
96.05   517.50   517.50   0.00
131.95   589.00   588.04  -0.96
polynom degree: 1
polynom for fit lambda c: [1.965001,
328.76165395]
rms_x = 0.6763
spectrum C:\Daten\Python\out\r_add65cal.dat
saved
```

We use the same linelist, but only the first entry is used, therefore the following lines will have an error. I use the Mg-line for calibration



For plotting a negative order you may reverse the plot by plotting from e.g. +100 nm to -900 nm, plotting the zero order and negative first order:



## Logfile

The main inputs and results of the script execution are stored in a logfile. You may review what you have done by reading the file `m_spec(yymmdd).log`, where (yymmdd) stands for the actual date in year, month day format. There you will find the main input and output for the complete processing from converting the AVI-file to the wavelength calibrated spectrum:

2020-04-03 15:59:51,525 M\_SPEC START

+++++

2020-04-03 15:59:51,525 M\_SPEC version 0.9.16, 0.9.16 +++++

2020-04-03 16:00:10,565 Platform: Windows

2020-04-03 16:00:13,779 converted C:/Daten/Python/181014/M20181014\_023817\_MAI\_2.avi  
91 images

2020-04-03 16:00:13,779 Station = MAI\_2 Time = 2018-10-14T02:38:17.000

2020-04-03 16:00:47,500 scalxy = 0.920

2020-04-03 16:00:47,500 x00 = 371.200

2020-04-03 16:00:47,500 y00 = 324.183

2020-04-03 16:00:47,500 rot = -0.006

2020-04-03 16:00:47,500 disp0 = 1.965

2020-04-03 16:00:47,500 a3 = 3.137e-07

2020-04-03 16:00:47,500 a5 = -3.193e-13

2020-04-03 16:00:47,500 'DATE-OBS' = 2018-10-14T02:38:17.000

2020-04-03 16:00:47,500 'M-STATIO' = MAI\_2

2020-04-03 16:00:57,735 67 images processed of 67

2020-04-03 16:00:57,735 process time for single distortion: 0.15 sec

2020-04-03 16:01:50,690 start x y, dx dy, file: 132 508,14 14, out/mdist

2020-04-03 16:01:50,700 out/mdist3 31.33 131.91 507.01 3.18 6.10

2020-04-03 16:01:50,776 out/mdist4 47.43 134.01 500.65 2.66 6.52

2020-04-03 16:01:50,998 out/mdist5 41.72 136.11 495.62 2.71 6.01

...

2020-04-03 16:01:55,362 out/mdist65 65.64 291.37 187.57 2.58 3.38

2020-04-03 16:01:55,414 out/mdist66 59.70 294.40 183.00 2.72 3.70

2020-04-03 16:01:55,488 out/mdist67 66.45 297.08 178.37 2.56 3.25

2020-04-03 16:01:55,532 time for register one image : 0.07 sec

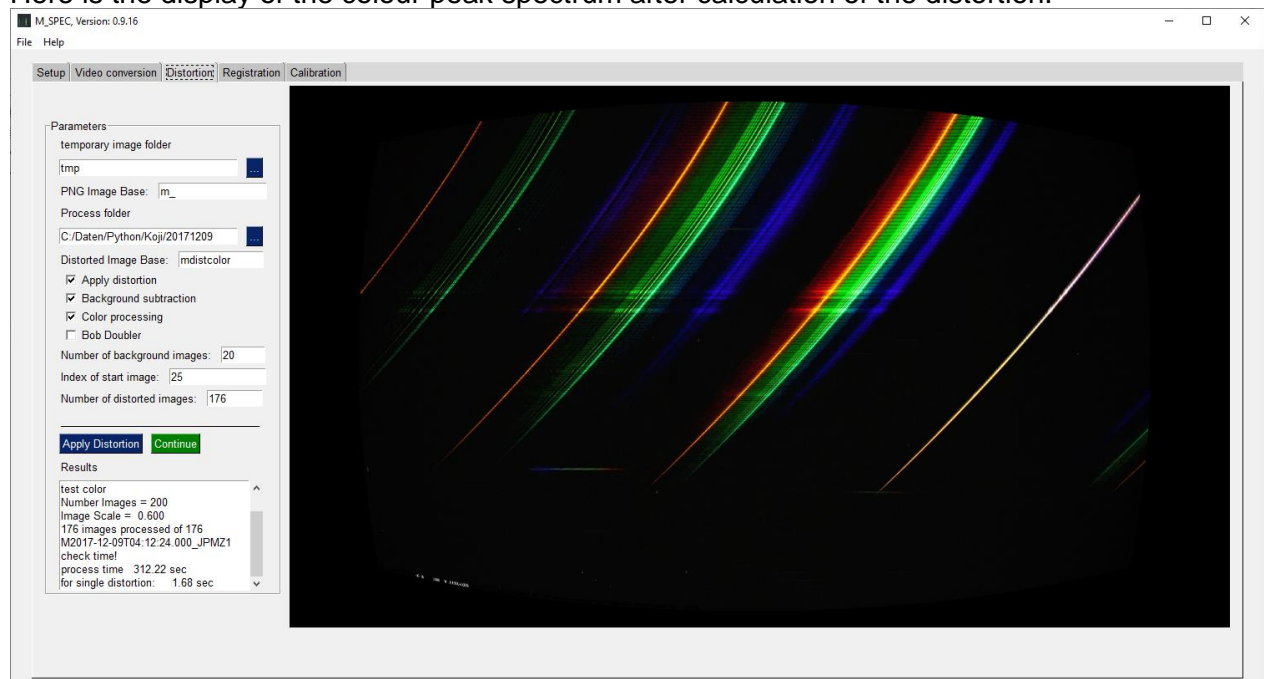
2020-04-03 16:02:43,981 start = 3, nim = 65

2020-04-03 16:02:43,981 added from 503 to 514, 12 rows  
 2020-04-03 16:02:43,981 tilt = 0.0000, slant = 0.400  
 2020-04-03 16:04:31,615 Pixel: 95.75 FWHMp = 3.540 lambda = 517.500 Mg I  
  
 2020-04-03 16:04:41,126 Pixel: 131.44 FWHMp = 2.910 lambda = 589.000 Na I  
  
 2020-04-03 16:04:45,728 polynom for fit lambda c: [ 2.003 325.678]  
 2020-04-03 16:04:45,729 pixel lambda fit error  
 2020-04-03 16:04:45,729 95.75, 517.50, 517.50, 0.0000  
 2020-04-03 16:04:45,730 131.44, 589.00, 589.00, 0.0000  
 2020-04-03 16:04:45,730 rms\_x = 0.0000  
 2020-04-03 16:04:45,738 spectrum C:\Daten\Python\out\r\_add65cal.dat saved  
 2020-04-03 16:04:45,758 spectrum C:\Daten\Python\out\r\_add65cal2.dat saved  
 2020-04-03 16:06:29,804 spectrum C:\Daten\Python\out\r\_add65cal2.dat plot saved as  
 20181014\_023817\_MAI\_2.png

## Processing of 4k meteor videos

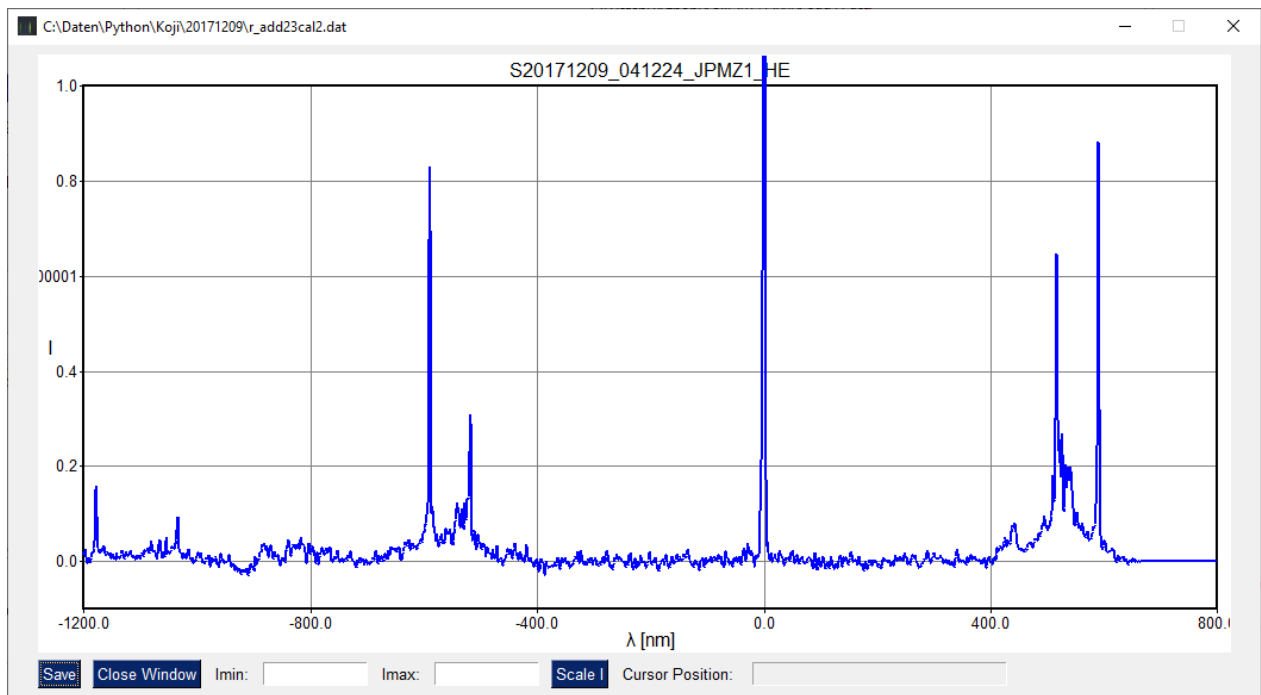
For processing of meteor spectra recorded with large format video cameras such as Sony Alpha 2 a fast computer is advisable. For faster processing 2x2 binning is recommended. Both colour and b/w processing is possible. Colour processing takes about 3 times longer, for registering a small portion of the images is converted to b/w. For the summation of rows and conversion to a 1-D spectrum the different colour channels are added, so the final result is the same as with b/w processing. As an example a spectrum recorded by Koji Maeda was processed (see also <https://meteorspectroscopy.org/2018/01/03/koji/>)

Here is the display of the colour peak spectrum after calculation of the distortion:



Notice the time for calculating the distortion (1.68 sec for images of size 1920x1080, 2x2 binning). An analysis of the images 28 to 50 (approximately the first second, at the bottom of the image, the meteor moving upwards) gave the following spectrum:





Typical for an early spectrum are the strong Na I lines (-2<sup>nd</sup> to +1<sup>st</sup> order), which were also used for calibration with M\_spec and for calibration of the actual spectrum. A linear fit was used:  
polynom for fit lambda c: [ 1.259e+00 -1.503e+03]

pixel	lambda	fit	error
257.95,	-1178.00,	-1177.83,	0.1668
725.21,	-589.00,	-589.48,	-0.4819
1193.86,	0.00,	0.62,	0.6196
1603.37,	517.50,	516.25,	-1.2451
1661.89,	589.00,	589.94,	0.9405

rms\_x = 0.7847

spectrum C:\Daten\Python\Koji\20171209\r\_add23cal.dat saved

## Content

Summary .....	1
Introduction .....	1
Processing of meteor spectra, overview .....	2
Image extraction from video sequence .....	2
Background image .....	2
Image transformation to the orthographic projection .....	3
Image registration .....	4
Correction of tilt and slant .....	4
Spectrum extraction .....	5
Spectrum calibration .....	5
Plotting the spectra .....	6
Python Installation .....	7
Installation of Anaconda .....	7
Load libraries .....	8
Install ffmpeg .....	8
Setup Pyzo .....	9
Edit text files .....	10
Installing Python in LINUX .....	10
Processing of meteor spectra, step by step .....	11
Starting the script .....	11
Conversion of *.AVI to PNG .....	13
Distortion .....	14
Registering the meteor spectra .....	16
Spectrum extraction .....	18
Wavelength calibration .....	22
Single line calibration .....	23
Logfile .....	25
Processing of 4k meteor videos .....	26