
Goodman Focus Documentation

Release 2.0.2

Simon Torres

Apr 08, 2022

HOW TO INSTALL

1	Overview	3
1.1	Install Using PYPI	3
1.2	Install Using GitHub	3
1.3	Getting Data	4
1.4	Running from terminal	5
1.5	Using it as a library	5
1.6	Interpreting Results	6
1.7	Change History	9
1.8	License	11
2	Indices and tables	13

OVERVIEW

This is a tool to obtain the best focus from a series of images obtained at different focus values. It works for imaging and for spectroscopy.

This tool requires python 3.6 at least to work. It will not install with 3.5.

Note: We recommend using a virtual environment management system such as [astroconda](#)

1.1 Install Using PYPI

Create a virtual environment using *conda* and specify python version 3.6.

```
conda create -n goodman_focus python=3.6
```

Install using *pip*

```
pip install goodman-focus
```

1.2 Install Using GitHub

Clone the latest version using:

```
git clone https://github.com/soar-telescope/goodman_focus.git
```

Move into the new directory

```
cd goodman_focus
```

Create a virtual environment using the *environment.yml* file and activate it.

```
conda env create python=3.6 -f environment.yml
```

This will create a virtual environment named *goodman_focus* with all the important parts on it.

```
conda activate goodman_focus
```

Install using *pip*

```
pip install .
```

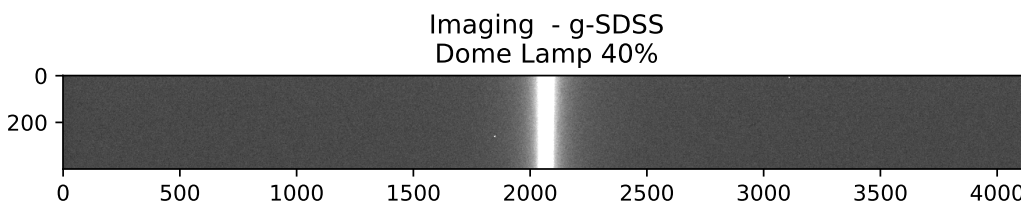
1.3 Getting Data

The goodman focus value ranges from -2000 to $+2000$ the best practice (from experience) is to sample every 200. If you want to refine you should not go below a 100 step. If you already know the approximate focus value for a given setup, you don't need to do the full range, just take enough samples so it is possible to make a fit or see the trends. At least five points.

Note: Always use the narrowest slit available

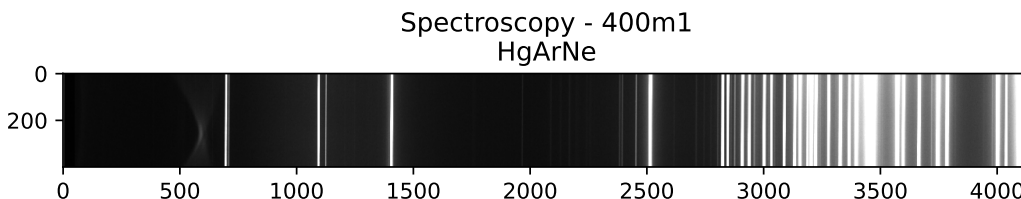
1.3.1 Imaging

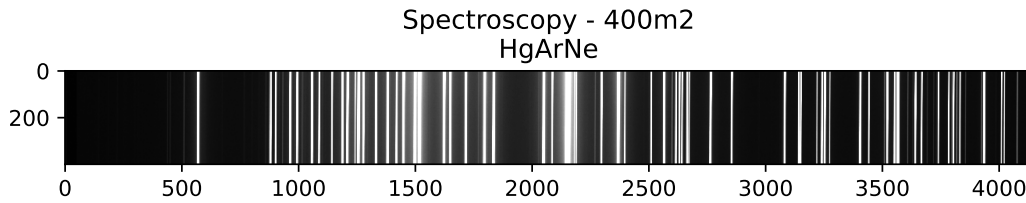
For imaging, an image of the slit is obtained, illuminated by an uniform light such as a quartz lamp. A special ROI is used for faster reading. This does not affect the focus of course.



1.3.2 Spectroscopy

In spectroscopy a comparison lamp is observed and after every line is measured and fitted, an average is obtained with sigma clip rejection.





1.4 Running from terminal

There is an automatic script that will obtain focus from a folder containing a focus sequence.

If you have *fits* files you can simply run.

```
goodman-focus
```

It will run with the following defaults:

Table 1: Default values for arguments

Argument	Default Value	Options
--data-path <input>	Current Working Directory	Any valid path
--file-pattern <input>	*.fits	Any
--features-model <input>	gaussian	moffat
--plot-results	False	True
--debug	False	True

Where <input> is what you type.

To get some help and a full list of options use:

```
goodman-focus -h
```

1.5 Using it as a library

After installing *Install Using PYPI* you can also import the class and instantiate it providing a set of arguments and values or using default ones.

```
from goodman_focus import GoodmanFocus
```

If no argument is provided it will instantiate with the default values.

The list of arguments can be defined as follow:

```
import os
from goodman_focus import GoodmanFocus

goodman_focus = GoodmanFocus(data_path=os.getcwd(),
                              file_pattern='*.fits',
                              obstype='FOCUS',
                              features_model='gaussian',
```

(continues on next page)

(continued from previous page)

```
plot_results=False,  
debug=False)
```

Which is equivalent to:

```
from goodman_focus import GoodmanFocus  
  
goodman_focus = GoodmanFocus()
```

`features_model` is the function or model to fit to each detected line. `gaussian` will use a `Gaussian1D` which provide more consistent results. and `moffat` will use a `Moffat1D` model which fits the profile better but is harder to control and results are less consistent than when using a gaussian.

Finally you need to call the instance, here is a full example.

```
from goodman_focus import GoodmanFocus  
  
goodman_focus = GoodmanFocus()  
  
results = goodman_focus()
```

However since version *0.3.0* you can pass a list of files and all will only check that all files exists

1.6 Interpreting Results

The terminal version will print a message like this

```
Mode: IM__Red__VR Best Focus: -682.2891445722862 at FWHM: 2.610853168299203  
Best image: 0019_IM_FOCUS_VR-02-11-2019.fits with focus: -601 and FWHM: 2.  
618208314784383
```

Note: In version *2.0.0 21-10-2021* the format changed from the previous stable release in order to simplify the serialization process.

Using it as a library will return a list of dictionaries with the following values. Combination of settings for which the code is the same is called a *mode*, so the *mode name* is it's unique identifier, how the name is constructed is explained in `decoding-mode-name`. In the following example only one object is included for simplicity.

```
[  
  {  
    "date": "2019-08-10",  
    "time": "2019-08-10T20:06:15.884",  
    "mode_name": "IM__Red__VR",  
    "focus": -837.3191595797898,  
    "fwhm": 2.5831939867345586,  
    "best_image_name": "0062_FOCUS_IMG_VR-10-08-2019.fits",  
    "best_image_focus": -799,  
    "best_image_fwhm": 2.6170559494587478,  
    "focus_data": [  
      -1994,  
      -1801,
```

(continues on next page)

(continued from previous page)

```

-1601,
-1400,
-1201,
-997,
-799,
-601,
-400,
-200,
1
],
"fwhm_data": [
5.930451746635722,
5.306608292580763,
4.594963225848573,
3.555224927051857,
3.0090692421560217,
2.705695641392574,
2.6170559494587478,
2.7128327791088545,
3.02782407160001,
3.615781144676397,
4.265013680275817
]
}
]
```

It is also possible to obtain a plot, from terminal, use `--plot-results`. Below is a reproduction of results obtained with test data.

1.6.1 Decoding de mode name

The mode name is constructed using two letters to define the observing technique (Imaging or Spectroscopy) and values obtained from the header. The characters `<`, `>` and *blanks* are removed.

The mode name is different for Imaging and Spectroscopy, since for imaging the important settings are the instrument and the filter and for spectroscopy the important values come from the instrument, the grating and observing mode and filter from second filter wheel. Below, the word inside the parenthesis represents a keyword from the header.

Warning: Be aware that the separator string is a double underscore. This change was necessary to avoid confusion with single underscores used in certain keyword values.

For imaging:

IM__(INSTCONF)__(FILTER)

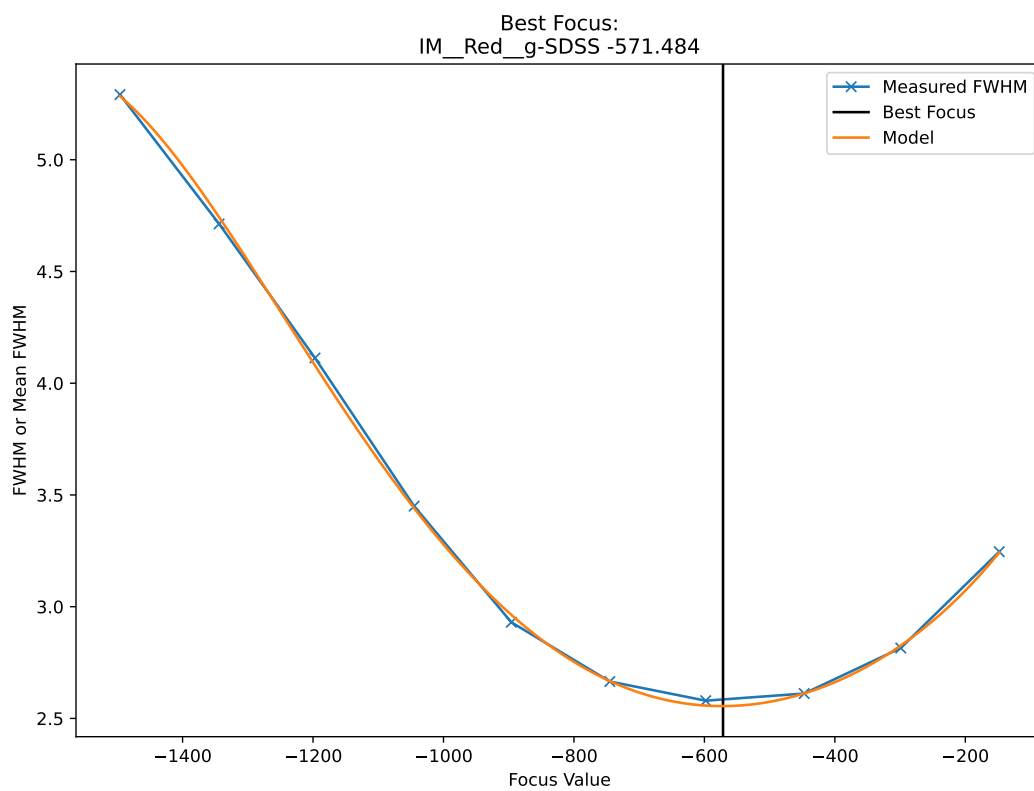
for example:

IM__Red__g-SDSS

For spectroscopy:

SP__(INSTCONF)__(WAVMODE)__(FILTER2)

for example:



SP__Red__400m2__GG455

1.7 Change History

1.7.1 2.0.2 08-04-2022

- Added exception handling for when it's not possible to use `scipy.optimize.brent`
- Added notes field to report special warnings.

1.7.2 2.0.1 23-02-2022

- Changed method of getting the best focus to `scipy.optimize.brent`
- Fixed bug regarding results report on terminal entry point

1.7.3 2.0.0 21-10-2021

- Added FWHM, Best image and data to results.
- Bumped version one major step due to the change on the format of the results.
- Returned result is a list of dictionaries now.
- Added Date and ISO timestamp as part of the json results.
- Focus and FWHM are rounded to 10 decimals.

1.7.4 1.0.0

- Removed Travis CI [#31]
- Replaced separator to double underscore [#34, #35]
- Updated documentation.

1.7.5 0.3.6

- Added python 3.7 and 3.8 to Travis CI
- Removed astroconda from environment.yml and specified python 3.8 to avoid 3.9

1.7.6 0.3.5

- Implemented Github Actions
- Removed astroconda channel from environment

1.7.7 0.3.4

- Fixed version of *ccdproc* to *1.3.0.post1*. *ccdproc==2.0.0* does have some problems reported on *astropy/ccdproc#699*

1.7.8 0.3.3

- Changed Sigma Clipping iterations from 1 to 3
- Added sigma clip iterations as argument to function *get_fwhm* though this is not exposed to the user.

1.7.9 0.3.2

- Changed logger setup
- Moved data directory validation from instantiation to execution.

1.7.10 0.3.1

- Fixed bug on the calculation of the pseudo-derivate used to find best focus value
- Updated hardcoded string that defines the Imaging wavmode from *Imaging* to the new *IMAGING*.
- Added docstrings

1.7.11 0.3.0

- Created dedicated documentation for readthedocs.
- Fixed bug where return was missing,
- GoodmanFocus need to be instantiated only once [#19]
- Calling instance of GoodmanFocus can receive a list of files as input [#19]
- Argument *-file-pattern* is now actually used in file selection [#18]
- Eliminated some warnings.
- Included plots in documentation.

1.7.12 0.2.0

- Added messages when no file matches the *-obstype* value, by default is *FOCUS* [#9]
- Replaced *parser.error* by *log.error* and *sys.exit* when the directory does not exist and when exists but is empty.
- Added test for cases when the directory does not exist, when is empty and when no file matches the selection on *-obstype* which by default is *FOCUS*.
- Replaced *logging.config.dictConfig* by *logging.basicConfig* which fixed several issues. For instance *-debug* was unusable, and also there were duplicated log entries for the file handler when used as a library in other application. [#10]
- Replaced the use of the function *get_args* by using arguments on class instantiation instead
- Created name for modes [#11]

1.7.13 0.1.3

- Fixed some issues with documentation
- Added `.readthedocs.yml` file for RTD builds
- Added `install_requires` field in `setup()`
- Removed python 3.5 from the supported versions.

1.8 License

BSD 3-Clause License

Copyright (c) 2019, SOAR Telescope All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`