“SWEET-Cat: Spectroscopic Characterization of Stars with Exoplanets”

Center for Astrophysics of the University of Porto

27 de julho de 2020

Henrique Miguel Marques Luís Legoinha

Supervisors:

Sérgio Sousa
João Camacho

Abstract:
This report presents the methodology used in the construction of a Python code that acts on the form of the TNG telescope - Telescopio Nazionale Galileo. The built-in function allows to comfortably store files associated with extensive lists of stars on your computer. The code was implemented within the context of the SWEET-Cat project so functionalities such as the automatic selection of the HARPN instrument and selection of optional parameters of the stellar spectra were added as well.

1) INTRODUCTION

The characterization of stars is a fundamental step in the study of the associated exo-planets. When observing a planet passing through a star, it is important that the stellar radius is precisely calculated in order to be able to estimate the radius of the planet. The calculation of the star radius will depend on a good derivation of stellar parameters through the observed spectra. The chemical composition of the planet and its atmosphere are also related to the chemical composition of the star. Therefore, this characteristic of the star must also be rigorously derived through the observed spectra.

The extraction of stellar parameters from the spectra can be done by different methods. However, it is important to ensure uniformity in its derivation. The SWEET-Cat project aims to create a list of stellar parameters whose calculation is processed in the same way for all available stars. Thus, a list of homogeneously derived parameters is available to the scientific community and can be consulted on the website. The results are compared with those of other catalogues allowing greater confidence and better perception of errors associated with its calculation.

To obtain stellar spectra one must search for the star on the form website (for example) of ESO (European Southern Observatory). However, it is not possible to gather all the observations of a given star on one site. The spectra obtained through the HARPN instrument must be consulted on the TNG (Telescopio Nazionale Galileu) form website. It is the observations of this instrument for each of the stars already present in the SWEET-Cat catalogue that we intend to obtain in this work. Deriving the stellar parameters of each star from the HARPN spectra will further improve the quality of the SWEET-Cat catalogue.

Since SWEET-Cat gathers a very large number of stars the only way to obtain the spectra for all of them is by using code that automatically performs the search, extracts, and saves all the found spectra. However, for the TNG website there is no code available to carry out this procedure so there is a need to build one.

2) CODE

The developed code acts on the HTML structure of the TNG form website in order to command the search and the extraction operations of the stelar spectra. The code is in Python and is available in this GitHub repository. It has been ensured that both Windows and Linux users are able to use the built function. The method is sensitive to the browser used, working only on Chrome and Mozilla Firefox. The preliminary instructions for first use are available in the "read me" of that repository. The function's name is inspired by the instrument it explores (the Harps North) being so called HARPN.

The function arguments are only 6, of which the first three are mandatory: path to the folder where the spectra will be saved, path to the desired browser executable, star list, fiber, file type and download. The latter (download), when defined as ‘False’, allows for just a search (without download) to identify which stars in the list of stars provided are not recognized by the TNG telescope website database. Thus it is possible to distinguish unrecognized stars from recognized stars but without any observation in the database. Next follows a brief explanation of the construction of the code and the libraries used in this construction.

2.1) Code Building

The first approach attempts to replicate the logic of the existing function class to the ESO website form (the ESO class). Although these are different websites they are both built in html and therefore there will be similarities. The objective is to build a ‘payload’ (set of information to be sent to the website) through a specific function (requests) to be processed by the TNG website. However, two problems arise that will prevent the simple and direct adaptation of the ESO class. The presentation of research results on both websites is totally different: TNG spectra appear on the same page of the form and on ESO's website they appear on a new page. In addition, the type of button that must be pressed to activate the search is not easily identified and commanded in the code for the TNG website (the difference is visible only in the html code). These two situations make it impossible for the TNG website to correctly process the ‘payload’

recieved. However, some of the Python libraries used by the ESO class are possible to use in the second attempt to build the code and will even prove to be important.

The second approach uses a library (selenium) that starts a browser (without being visible to the user) and is equipped with a set of functions that allow access specifically to the html code of the started browser. These functions create an enormous freedom of programming as it is very easy to specify elements of the html code and interact with them after. It is possible to simulate any command coming from the keyboard or mouse on a specific html element. Thus, once on the TNG form website it is necessary to identify which element of the html code is associated with the fields of interest to be filled, with the selection lists or with the action buttons. For this, selenium functions must be given either the name or the ID of the html elements. This information is comfortably obtained through the 'inspect source code' option present on any web page. Therefore, it is possible to select the desired search parameters (HARPN instrument and other specifications associated with spectra such as fiber A and type 'sıd'). The code writes each element of the list of stars given in the field associated with the name of the star and then simulates pressing the 'resolve' and 'search' buttons. This starts the search and once the results are visible, all spectra that are free to use are extracted and saved in a folder with the name of the star (within the directory specified by the user). These last steps combine simple 'if' cycles and conditions.

The difference between the first and the second approach is essentially due to the way information is sent to the TNG website. While in the first it is a set of information submitted in a block (the 'payload') in the second it is directly simulated the filling of the fields and the necessary interactions with the page.

2.2) Used Libraries

The following table summarizes the libraries and functions of the Python programming language used during the construction of the Code:

Python programming language libraries and functions used	
Selenium	Function library that allows launching and operating a browser via its html. It is possible to start and control it without it being visible on the computer. It offers a set of tools for identification and interaction with the html code. The most useful (in general) are the click and send keys commands.
Gzip	Function that allows the extraction of .gz files. This is an extension of zipped files.
time	Function that allows you to specify a time interval for the code to wait. It is useful to ensure that the page's html updates correctly
wget	Function that allows to download files through a link. It is used in the code to process downloads of stellar spectra without the code having to click on each link. This would lead to a new window that in addition to spending time to be presented can be associated with a non-free spectrum breaking the cycle of code downloads.
astropy	Collection of software packages designed for use in astronomy. It is used in the code to open the spectrum files after they have been extracted and to access the S / N values.
numpy	Package that supports multidimensional arrays and matrices, having a wide collection of mathematical functions to work with these structures. In the code is used to help calculate the S / N averages of the spectra found for a given star.
BeautifulSoup	Library of functions that allow for searching and extracting information from the html source of a page. It is used in the code (together with the functions of the Selenium package) to find out if the star is present in the database or if there are spectra (and how many there are) for that same star.
OS	Function library that allows to reproduce commands associated with the operating system. It is used to read paths, create folders or view items within a folder.
Shutil	Used in conjunction with the functions of the module above (OS) it allows to perform tasks, such as copying or moving files from one directory to another (it also allows to delete them).

Table 1 - Libraries, functions and their contribution to the code

2.3) Code application

The following lines of code show how the function should be started for the different operating system and browser cases. In red, in the first example, are the mandatory arguments that must always be specified by the user. In green are 2 of the 3 optional arguments. The third optional argument (the 'download') is pre-defined as 'True' so, in this example, spectra of the stars present in the list provided are being downloaded where the observations are made by the 'A' fiber and the type of file is 's1d':

Windows, Chrome:

```
HARPN('C:/Users/hlego/Desktop/estrelas2/' , r'C:/bin/chromedriver' , lista_de_estrelas , 'A' , 's1d')
```

Windows, Mozzilla Firefox:

```
HARPN('C:/Users/hlego/Desktop/estrelas2/', 'C:/bin/geckodriver', lista_de_estrelas, 'A', 's1d')
```

Linux, Chrome:

```
HARPN('/home/henrique/Downloads' ,  `/home/henrique/ChromeDriver/chromedriver' ,  'A' , 's1d')
```

Linux, Mozzilla Firefox:

```
HARPN('/home/henrique/Downloads' , '/usr/local/bin/geckodriver' , 'A' , 's1d' )
```

The second argument of the function is easily specified following the steps in the tutorial presented in the «read-me» file in the GitHub repository. Installing the driver to start the browser is just as simple. These steps and more details regarding the function's arguments can be found here: read-me. In the previous example, a list of stars was considered to show the different possible results depending on the number of free spectra to use. The results presented are as follows:

```
1)  HD1445: not in database
2)  Corot-7: 1/1 spectra extracted
3)  Vega: 210/211 spectra extracted
4)  Arcturus: 0/9 spectra extracted
DONE!
```

Figure 1 - Results of the function for a list of test stars [HD1445, Corot-7, Vega, Arcturus]

The other feature of the function starts with the third optional argument (green in the next line of code) set to 'False'. This is identical for any browser or operating system so only one case is shown (Windows, Chrome):

```
HARPN('C:/Users/hlego/Desktop/estrelas2/', r'C:/bin/chromedriver', lista_de_estrelas, download=False)
```

For the list of previous stars, the results indicate that they are all recognized by the database (the test star HD1445 simply does not have spectra although it is recognized).

```
Stars not recognized: 0
```

Figure 1 – survey results

3) FINAL REGARDS

The built function meets the need of the SWEET-Cat project to obtain the spectra of several stars automatically. The extraction time of the spectra per star depends exclusively on the number found and can vary from a few seconds to a few minutes. The list of stars not recognized by the database that the function returns can in turn be used to automate their identification through libraries such as SIMBAD. Although the construction of a function that does this has been started it is not yet possible to cover all the names of the stars so this functionality is not yet available in the function.

The function was applied to a large part of the SWEET-Cat catalogue. Its code was verified to operate correctly since there were no problems in downloading the spectra of the stars present in the database. Many of the stars have no files and a small number are not recognized. As some of the recognized stars have hundreds of spectra, it would not be possible to process them all with the available memory of the computer. The time spent doing so would be approximately 8 hours. Since it is now more than confirmed that everything is working correctly in what respects the downloads, the code was interrupted. The identification feature of unrecognized stars is much faster. Its application to the SWEET-Cat catalogue results in a list of stars not recognized by the form on the TNG website which can be consulted in the annex (they are 98 stars in all).

ANEX

List of stars not recognized by the TNG website, in a format ready to be copied to python:

lista = ['1SWASP J1407', '2M0805+48', '2M1059-21', '2MASS J0249-0557 (AB)', 'AD 3116', 'Cha Ha 8', 'DS Tuc A', 'EPIC 249893012', 'EPIC-206011691', 'EPIC-210490365', 'eps Ind A', 'gamma 1 Leo', 'HAT-P-10 A', 'HATS-71A', 'HIP 65A', 'IC 4651 9122', 'J1433', 'K2-293', 'K2-294', 'KELT-19 A', 'KELT-24/MASCARA-3', 'Kepler-1319 A', 'Kepler-16(AB)', 'Kepler-1647 (AB)', 'Kepler-1651 A', 'Kepler-1661 (AB)', 'Kepler-34(AB)', 'Kepler-35(AB)', 'Kepler-38(AB)', 'Kepler-413 (AB)', 'Kepler-420 A', 'Kepler-453 (AB)', 'Kepler-469', 'Kepler-47(AB)', 'Kepler-64 (AB)', 'LTT 1445A', 'MARVELS-10', 'MARVELS-11', 'MARVELS-12', 'MARVELS-13', 'MARVELS-16', 'MARVELS-17', 'MARVELS-18', 'MARVELS-19', 'MARVELS-8', 'NGTS-11', 'NGTS-3A', 'Pr 0211', 'Pr0201', 'Qatar-1', 'Qatar-10', 'Qatar-2', 'Qatar-3', 'Qatar-4', 'Qatar-5', 'Qatar-6', 'Qatar-7', 'Qatar-8', 'Qatar-9', 'SAND364', 'SDSS J080531+481233', 'SDSS J1411+2009', 'TOI-1130', 'TOI-1235', 'TOI-132', 'TOI-1338', 'TOI-1406', 'TOI-150', 'TOI-157', 'TOI-163', 'TOI-169', 'TOI-203', 'TOI-216', 'TOI-257', 'TOI-263', 'TOI-270', 'TOI-402', 'TOI-421', 'TOI-564', 'TOI-677', 'TOI-700', 'TOI-813', 'TOI-849', 'TOI-905', 'TYC-3667-1280-1', 'WASP-102', 'WASP-143', 'WASP-146', 'WASP-148', 'WASP-150', 'WASP-176', 'WASP-178/KELT-26', 'WASP-70 A', 'WASP-85 A', 'WASP-87 A', 'YBP1194', 'YBP1514', 'YBP401']