webb_scraping Documentation

Release 0.3.0-beta

Arjun Savel

Jun 18, 2020

Getting Started

CHAPTER 1

About webb_scraping

The webb_scraping code is intended to supplement explorations of exoplanet data for James Webb purposes. The program pulls from a variety of sources (ExoFOP-TESS, MAST, and SIMBAD) to aggregate data relevant to a potential JWST proposal.

webb_scraping is being actively developed on GitHub. If you find bugs or would like to make a contribution, please open an issue!

CHAPTER 2

Installation

webb_scraping, as of now, does not include an automated testing suite; however, this is an issue that will soon be fixed!

2.1 Installing with pip

The most straightforward way to download the code is too `pip install` it. It is recommended to run the below lines in a fresh conda environment.

```
python3 -m pip install -U pip
python3 -m pip install -U setuptools setuptools_scm pep517
pip install webb_scraping
```

2.2 Installing from source

webb_scraping is developed on GitHub, where you can find the latest developer version of the code. It is recommended to run the below lines in a fresh conda environment.

```
python3 -m pip install -U pip
python3 -m pip install -U setuptools setuptools_scm pep517
git clone https://github.com/arjunsavel/webb_scraping
cd webb_scraping
python3 -m pip install -e .
```

[3]:
```
from webb_scraping import target
```

CHAPTER 3

Will JWST observe my target?

Let's say that we're interested in observing TRAPPIST-1 with JWST. To determine whether this target's already been included in an ERS or GTO proposal, we can call `target.Target` on our target of interest. This instantiates a `Target` object, allowing us to perform scrapes relevant to our target of interest.

[7]:
```
test = target.Target("TRAPPIST-1")
```

Next, we can call `test.scrape_webb_MAST` to query MAST regarding this target. Note:

[9]:
```
test.scrape_webb_MAST()
test.webb_approved
```

[9]:
```
True
```

Because `test.webb_approved` returned as `True`, there's already a GTO or ERS proposal logged for this target. Let's see what it is.

[13]:
```
test.webb_proposal_names
```

[13]:
```
['MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
```

(continues on next page)

(continued from previous page)

```
 'Transit Spectroscopy of TRAPPIST-1e',
 'Transit Spectroscopy of TRAPPIST-1e',
 'Transit Spectroscopy of TRAPPIST-1e',
 'Transit Spectroscopy of TRAPPIST-1e',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'MIRI observations of transiting exoplanets',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'NIRISS Exploration of the Atmospheric diversity of Transiting exoplanets (NEAT)',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Thermal emission from Trappist1-b',
 'Transit Spectroscopy of TRAPPIST-1e',
 'Transit Spectroscopy of TRAPPIST-1e',
 'Transit Spectroscopy of TRAPPIST-1e',
 'Transit Spectroscopy of TRAPPIST-1e']
```

Looks like a number of studies are taking relevant data!

[ ]:

[3]:
```
from webb_scraping import target
```

CHAPTER 4

What's in the literature about my target?

Let's say, once more, that we're interested in observing TRAPPIST-1 with JWST. To set up the answer to this proble, we again call `target.Target` on our target name. This instantiates a `Target` object, allowing us to perform scrapes relevant to our target of interest.

[4]:
```
test = target.Target("TRAPPIST-1")
```

Given we'll be scraping a number of different sources, we can use the `test.scrape_all` function.

[5]:
```
test.scrape_all()
```

```
Scraping arXiv: 100%|| 10/10 [00:05<00:00,  1.87it/s]
```

Let's see if there are any asssociated HST or JWST proposals.

[6]:
```
test.webb_approved, test.hst_approved
```

[6]:
```
(True, True)
```

[10]:
```
test.hst_data
```

[10]:
```
{'Two Birds One Stone: Simultaneous Atmospheric Pre-Screening of Two Temperate Earth-
↪Sized Exoplanets During Their Double Transit': 'mast:HST/product/id4301r4q_drz.fits
↪',
 'Collecting the Puzzle Pieces: Completing HSTs UV+NIR Survey of the TRAPPIST-1␣
↪System ahead of JWST': masked,
 'A search for low-mass companions to ultracool dwarfs': 'mast:HST/product/u64tb803r_
↪c0f.fits',
 'Exploratory observations of the TRAPPIST-1 system: essential prelude to an␣
↪immediate JWST follow-up': 'mast:HST/product/idded1p9q_drz.fits',
 'The Mega-MUSCLES Treasury Survey: Measurements of the Ultraviolet Spectral␣
↪Characteristics of Low-mass Exoplanetary Systems': 'mast:HST/product/ld1mz6010_
↪x1dsum.fits',
 'UV exploration of two Earth-sized planets with temperate atmospheres': masked,
 'Confirming the Presence of an Hydrogen Exosphere around the Earth-sized Temperate␣
↪Planet TRAPPIST-1c': masked,
 'UV irradiation of the Earth-sized planets orbiting TRAPPIST-1': masked}
```

We can also examine what aliases exist for this target.

[7]:
```
test.aliases
```

[7]:
```
['EPIC 246199087',
 'K2-112',
 '2MUCD 12171',
 '2MASS J23062928-0502285',
 '2MASSI J2306292-050227',
 '2MASSW J2306292-050227',
 'Gaia DR2 2635476908753563008',
 'WISEA J230630.02-050234.1',
 'TRAPPIST-1',
 'EPIC 200164267']
```

To see what arXiv articles reference our target (or its aliases), we can use `test.arxiv_links`.

[8]:
```
test.arxiv_links
```

[8]:
```
['https://arxiv.org/pdf/1708.02200',
 'https://arxiv.org/pdf/1706.02018',
 'https://arxiv.org/pdf/2003.11590',
 'https://arxiv.org/pdf/2002.10950',
 'https://arxiv.org/pdf/2002.05892',
 'https://arxiv.org/pdf/2002.04798',
 'https://arxiv.org/pdf/2002.02015',
 'https://arxiv.org/pdf/2001.11605',
 'https://arxiv.org/pdf/2001.08946',
 'https://arxiv.org/pdf/2001.06225',
 'https://arxiv.org/pdf/2001.04606',
 'https://arxiv.org/pdf/2001.01338',
 'https://arxiv.org/pdf/1912.05749',
 'https://arxiv.org/pdf/1912.02313',
 'https://arxiv.org/pdf/1912.02132',
 'https://arxiv.org/pdf/1911.09132',
 'https://arxiv.org/pdf/1911.08878',
 'https://arxiv.org/pdf/1911.08596',
 'https://arxiv.org/pdf/1911.02051',
 'https://arxiv.org/pdf/1910.09871',
 'https://arxiv.org/pdf/1909.13859',
 'https://arxiv.org/pdf/1909.13331',
 'https://arxiv.org/pdf/1909.12320',
 'https://arxiv.org/pdf/1909.09158',
 'https://arxiv.org/pdf/1908.10873',
 'https://arxiv.org/pdf/1908.04166',
 'https://arxiv.org/pdf/1907.13145',
 'https://arxiv.org/pdf/1907.06451',
 'https://arxiv.org/pdf/1907.05710',
 'https://arxiv.org/pdf/1907.02112',
 'https://arxiv.org/pdf/1906.09866',
 'https://arxiv.org/pdf/1906.06797',
 'https://arxiv.org/pdf/1906.05250',
 'https://arxiv.org/pdf/1906.03527',
 'https://arxiv.org/pdf/1906.00669',
 'https://arxiv.org/pdf/1905.12821',
 'https://arxiv.org/pdf/1905.11419',
 'https://arxiv.org/pdf/1905.11298',
 'https://arxiv.org/pdf/1905.07070',
 'https://arxiv.org/pdf/1905.06035',
```

(continues on next page)

(continued from previous page)

```
 'https://arxiv.org/pdf/1905.02560',
 'https://arxiv.org/pdf/1905.00512',
 'https://arxiv.org/pdf/1903.04501',
 'https://arxiv.org/pdf/1902.08772',
 'https://arxiv.org/pdf/1902.04026',
 'https://arxiv.org/pdf/1902.03867',
 'https://arxiv.org/pdf/1902.03732',
 'https://arxiv.org/pdf/1901.04057',
 'https://arxiv.org/pdf/1901.02747',
 'https://arxiv.org/pdf/1901.02041',
 'https://arxiv.org/pdf/1901.02015',
 'https://arxiv.org/pdf/1901.00219']
```

[ ]:

[1]:
```
from webb_scraping import target
```

CHAPTER 5

Is my target observable?

Let's say, once more, that we're interested in observing TRAPPIST-1 with JWST. To set up the answer to this proble, we again call `target.Target` on our target name. This instantiates a `Target` object, allowing us to perform scrapes relevant to our target of interest.

[2]:
```
test = target.Target("TRAPPIST-1")
```

We'll first need to aggregate the relevant data, so we can use the `test.scrape_all` function.

[3]:
```
test.scrape_all()
```
```
Scraping arXiv: 100%|| 10/10 [00:06<00:00,  1.64it/s]
```

Next, we can calculate the TSM and ESM, two proxies of SNR introduced in Kempton+ 18, to determine how observable our target is.

[4]:
```
test.run_all_calculations()
```

[6]:
```
test.TSM, test.ESM
```
[6]:
```
(41.288771043260134, 3.915150286416691)
```

[ ]:
```

```

CHAPTER 6

API

class webb_scraping.target.**Target**(*input_name*)

Bases: object

Performance of web reconnaisanse on an interesting target.

Attributes:

aliases (list of strings) names by which the target is known in other catalogs.

input_name (str) name of target you're interested in.

webb_approved (bool) whether or not the target has been included in approved webb program.

hst_approved (bool) whether or not the target has been included in a public HST program.

webb_proposal_link (list of strings) if there are associated JWST proposals, these are the associated URLs.

webb_proposal_names (list of strings) if there are associated JWST proposals, these are the associated proposal names.

hst_data (dict) keys are HST proposals, vals are links to associated data producs.

exoplanet_archive_data (dict)

arxiv_links (list) list to PDFs of arxiv papers that have self.input_name or self.aliases in their abstracts

__init__ initializes.

scrape_all master run method.

find_aliases finds aliases.

search_webb_site manual scraping, not preferred.

find_aliases()

Uses astroquery and Simbad to find any aliases of input_name; these are then put into the self.aliases list.

run_all_calculations(*verbose=False*)
Calculates the TSM and ESM (Kempton+ 18) for this target, using known planet properties.

scrape_HST()
Checks MAST for the target's relevant HST proposals/data. Modifies hst_approved: if there are observations, sets it to True; otherwise False. Appends links to relevant HST data to hst_data.

scrape_all()
The preferred scraping method. This calls all other main scraping methods.

scrape_arxiv(*progress=False*)
Searches through arXiv abstracts for the target. Appends links of relevant arXiv pdfs to arxiv_links. If progress=True, outputs a tqdm progress bar.

scrape_exoFOP_aliases(*ticid*)
This manually scrapes exoFOP for aliases, given a TICID.

scrape_exoplanet_archive()

scrape_planet_properties()
Uses exo_MAST to get planet properties for this target.

scrape_webb_MAST()
Checks MAST for the target's relevant JWST proposals/data. Modifies webb_approved: if there are relevant proposals, sets it to True; otherwise False. Appends the names of these proposals to webb_proposal_names.

search_ERS()
Manually scrapes the JWST ERS page.

search_GTO()
Manually scrapes the JWST GTO page.

search_webb()
Manually scrapes both the JWST ERS and GTO pages.

search_webb_site(*URL*)
Checks whether self has been approved via GTO or ERS. Needs debugging as missing above targets still. Adds any links to webb_proposal_links. changed webb_approved. not validated to ERS.

webb_scraping.calculations.**ESM**(*planet_properties*, *verbose=False*)
Takes in planet properties, computes ESM (Kempton+ 18) for it. Need to double-check units?

Inputs:

planet_properties (dict) contains planet orbital_distance (in AU), stellar radius (solar radii), stellar effective temperature (K), planet mass (Jupiter masses), orbital period (days), stellar K band magnitude (mag), and planet-star radius ratio.

verbose (bool) False. Dtermines whettehr properties are printed after computaiton.

Outputs:

ESM (float) SNR proxy for emission spectroscopy introduced by Kempton+ 18.

webb_scraping.calculations.**TSM**(*planet_properties*, *verbose=False*)
Takes in row of dataframe, computes TSM (Kempton+18) for it.

Inputs:

planet_properties (dict) contains planet orbital_distance (in AU), stellar radius (solar radii), stellar effective temperature (K), planet mass (Jupiter masses), stellar mass (solar masses), orbital period (days), stellar J band magnitude (mag), and planet-star radius ratio.

verbose (bool) False. Dtermines whettehr properties are printed after computaiton.

Outputs:

ESM (float) SNR proxy for emission spectroscopy introduced by Kempton+ 18.

webb_scraping.calculations.**blackbody**(*wav*, *T*)

Computes the blackbody curve at a given wavelength.

Inputs:

wav (float) wavelength of interest (m)

T (float) temperature of body in question

Outputs:

intensity (float) value of blackbody curve.

webb_scraping.calculations.**kepler_a**(*m1*, *m2*, *P*)

Computes semimajor axis with Kepler's Third Law.. Inputs:

m1 (float) mass of first body (kg)

m2 (float) mass of second body (kg)

P (float) orbital period (s)

Outputs:

a (float) semimajor axis of the orbit (m)

Python Module Index

W

Index