Tess Follow Up Observations 2

February 5, 2025

1 Tess Follow Up Observations 2

This notebook is the second part of a follow-up observation on TESS objects of interest (TOIs). The ultimate goal is to create a list of TOIs observable from Sutherland, South Africa. The list will be used to plan follow-up observations with the SAAO 1.0m telescope. In part 1, I gathered a list of transiting exoplanets which have more than 2 comparisons in the Mookodi field of view. In this notebook, I determine which of these objects have transits which are visible at night from Sutherland during my observing run in mid-February.

[ ]:

```
# import necessary packages
import numpy as np
import pandas as pd

import astropy.units as u
from astropy.coordinates import AltAz, EarthLocation, SkyCoord
from astropy.time import Time,TimeDelta

from astroplan import Observer
from astroplan import FixedTarget
from astropy.coordinates import EarthLocation
from astropy.time import Time
from astroplan.plots import plot_airmass
from astroplan import EclipsingSystem
from astroplan import (PrimaryEclipseConstraint, is_event_observable,
                       AtNightConstraint, AltitudeConstraint,␣
 ↪LocalTimeConstraint)
```

[ ]:

```
# function definitions

def astroplan_init(observing_site, ra, dec, id_name, eclipse_time,␣
 ↪orbital_period, eclipse_duration, obs_time):
    # base cases
    if orbital_period == 0:
        return 0, ""

    # define observing site
    obs_location = EarthLocation.of_site(observing_site)
    observer = Observer.at_site(observing_site)
    #convert our RA and DEC into an astropy Sky Coordinate
    star_coordinates = SkyCoord(ra, dec, unit="deg")

    # define details of transit time and such
    primary_eclipse_time = Time(eclipse_time, format='jd')
    orbital_period = orbital_period * u.day
    eclipse_duration = eclipse_duration * u.hour

    # let astroplan know the name and location of our target star
    star= FixedTarget(name=id_name, coord=star_coordinates)

    #let astroplan know we have a transiting system
    curr_toi = EclipsingSystem(primary_eclipse_time=primary_eclipse_time,
                           orbital_period=orbital_period,␣
↪duration=eclipse_duration,
                           name=id_name)

    # the start time of our observing run
    obs_time = Time(obs_time)

    #approximate number of transits which would be visible in a two week (14␣
↪day) period
    num_transits = 14.0/orbital_period.value
    n_transits=round(num_transits - 0.5)

    constraints = [AtNightConstraint.twilight_civil(),
               AltitudeConstraint(min=30*u.deg)]

    ing_egr = curr_toi.next_primary_ingress_egress_time(obs_time,␣
↪n_eclipses=n_transits)

    # using our constraints, determine if the both the ingress and egress are␣
↪observable
    # the function is_event_observable returns a boolean array which lets us␣
↪know which transits are observable.

    can_observe=is_event_observable(constraints, observer, star,␣
↪times_ingress_egress=ing_egr)
    num_observable = can_observe.sum()

    can_observeT=can_observe.T
    can_obs=np.insert(can_observeT,1,can_observeT[:,0],axis=1)
    observe_times=ing_egr[can_obs]

    t2 = ""
    if( any(observe_times) ) :
        #convert to a more easily readable format
        t2 = Time(observe_times,format='isot')

    return num_observable, t2
```

[ ]:

```
# reading csv file created in part 1
toi = pd.read_csv('Tables/TESS_TOI_28Jan2025_Mookodi.csv')
toi.head()
```

[ ]:

```
# creating empty lists to be turned into columns
trans_num = []
trans_time = []

# looping over all rows in the dataset
for index, row in toi.iterrows():
    num, time = astroplan_init('Sutherland', row['RA (deg)'], row['Dec (deg)'],␣
  ↪row['TOI'], row['Epoch (BJD)'], row['Period (days)'], row['Duration␣
  ↪(hours)'], '2025-02-12 16:00')
    trans_num.append(num)
    trans_time.append(time)
```

[ ]:

```
# copying the dataframe
toi_filtered = toi.copy()

# appending the new columns to the dataframe
toi_filtered.loc[:,'Num Transits'] = trans_num
toi_filtered.loc[:,'Transit Times'] = trans_time

# keeping only rows with at least one observable transit
toi_trans = toi_filtered.loc[toi_filtered['Num Transits'] >= 1]
```

[ ]:

```
# saving to a new csv file
toi_trans.to_csv('Tables/TESS_TOI_28Jan2025_Mookodi_Transits.csv', index=False)
```

1.0.1 Final lists

below is the code for the lists provided in the writeup

[ ]:

```
# first 10 potentially observable and number of transits for each
toi_trans[['TOI', 'Num Transits']].head(10)
```

[ ]:

```
# list the number of TOIs with 1 observable transit
len(toi_trans[toi_trans['Num Transits'] == 1])
```

[ ]:

```
# list the number of TOIs with 2 observable
len(toi_trans[toi_trans['Num Transits'] == 2])
```

[ ]:

```python
# list the number of TOIs with 3 observable
len(toi_trans[toi_trans['Num Transits'] == 3])
```

[ ]:

```python
# list the number of TOIs with > 3 observable
len(toi_trans[toi_trans['Num Transits'] > 3])
```