# Tess Follow Up Observations 1

February 1, 2025

## 1 Tess Follow Up Obeservations 1

This notebook is the first 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 this notebook, I filter the TESS ground based follow-up data from the Exoplanet Follow-up Observing Program (ExoFOP) based on the following criteria: - Sufficient depth of transit > 1500 ppm - Faint enough: Tess Mag > 9 - No pervious time series photometry - Declination < +28 degrees

After this, I find all stars with at least 2 stars in the Mookodi field of view. The output of this notebook is a .csv file which includes all of the columns from the original TESS TOI file, but two new columns which list the number of comparison stars with Mookodi and with SHOC.

### 1.0.1 Filtering

[ ]:

```
# import necessary libraries
import pandas as pd
```

[ ]:

```
# reading the TESS TOI file and displaying the first few rows
toi = pd.read_csv('Tables/TESS_TOI_28Jan2025.csv')
toi.head()
```

[ ]:

```
# function to convert sexagesimal ra and dec to decimal (realized I could've␣
  ↪just used the astropy function, but oh well)
def sexagesimal_to_decimal(ra, dec):
    ra = ra.split(':')
    dec = dec.split(':')
    ra = (float(ra[0]) + float(ra[1])/60 + float(ra[2])/3600) * 15
    dec_sign = -1 if dec[0].startswith('-') else 1
    dec[0] = dec[0].lstrip('+-')
    dec = dec_sign * (float(dec[0]) + float(dec[1])/60 + float(dec[2])/3600)
    return ra, dec
```

[ ]:

```
# creating new columns for ra and dec in decimal form
toi['RA (deg)'], toi['Dec (deg)'] = zip(*toi.apply(lambda x:␣
  ↪sexagesimal_to_decimal(x['RA'], x['Dec']), axis=1))

# filtering the data
```

```
toi_filtered = toi[(toi['Depth (ppm)'] > 1500) & (toi['TESS Mag'] > 9) &␣
 ↪(toi['Dec (deg)'] < +28) & (toi['Time Series Observations'] == 0)]

toi_filtered.head()
```

Now that I have the filtered stars, I can search for fidn comparison stars in the Mookodi field of view.

### 1.0.2 Finding Comparison Stars

[ ]:

```
# import necessary libraries
import pyvo as vo
from tqdm import tqdm
```

[ ]:

```
# function definitions
# function to query the skymapper database
def skymapper_query(ra, dec):
    my_tap_query = ("SELECT DISTANCE(POINT('ICRS', raj2000, dej2000),␣
 ↪POINT('ICRS', "
              + ra + ", " + dec +
              ")) AS dist, "  +
              "m.object_id,m.raj2000,m.dej2000,m.g_flags,m.g_nimaflags,m.
 ↪i_flags,m.i_nimaflags, " +
            "m.g_psf,m.e_g_psf,m.i_psf,m.e_i_psf  FROM dr2.master AS m " +
              "WHERE 1 = CONTAINS(POINT('ICRS', raj2000, dej2000), "
              "CIRCLE('ICRS',"  +
               ra +", "+ dec +
              ", 0.083 )) ORDER BY dist " )

    tap_service = vo.dal.TAPService("https://api.skymapper.nci.org.au/public/
 ↪tap/")
    tap_results =  tap_service.search(my_tap_query)
    astropy_table = tap_results.to_table()
    df = astropy_table.to_pandas()

    #make sure that we really have found out target star
    if df.empty:
        print("Target star not found for ({},{})".format(ra, dec))
        # null out df
        df = pd.DataFrame()

    elif df['dist'].empty:
        print("Target star not found for ({},{})".format(ra, dec))
        # null out df
        df = pd.DataFrame()

    elif df['dist'].iloc[0] > 0.00056 :
        print("We have a problem, distance > 2 arcsconds for ({},{})".
 ↪format(ra, dec))
        # null out df
        df = pd.DataFrame()

    return df

# function to clean the photometric data
def photometric_cleaning(df) :
    # handle null case (when star was not found)
    if df.empty:
        return df

    # filter out bad photometry
    g_good1 = df['g_flags'] == 0
    g_good2 = df['g_nimaflags'] == 0
    g_good3 = df['e_g_psf'] < 0.022
    i_good1 = df['i_flags'] == 0
    i_good2 = df['i_nimaflags'] == 0
    i_good3 = df['e_i_psf'] < 0.022

    #combine these comparisons to make a new dataframe
    df_good = df[g_good1 & g_good2 & g_good3 & i_good1 & i_good2 & i_good3]
    return df_good


# function to search for comparison stars in the Mookodi field of view
def mookodi_search(df_good, G_mag):
    # handle null case (when star was not found)
    if df_good.empty:
        return 0, 0

    i_brightcut = G_mag - 0.5
    i_faintcut = G_mag + 4.0

    i_bright = df_good['i_psf'] > i_brightcut
    i_faint = df_good['i_psf'] < i_faintcut

    df_mookodi = df_good[i_bright & i_faint]

    shoc_comp = df_mookodi['dist'] < 0.0233   #SHOC field of view is 2.8x2.8␣
 ↪arcminutes, so use r=1.4 arcminutes
    df_shoc = df_mookodi[shoc_comp]
    num_mookodi = df_mookodi.shape[0]
    num_shoc_comp = df_shoc.shape[0]
    return num_mookodi, num_shoc_comp
```

[ ]:

```
# looping over all stars in the filtered data and to generate mookodi and shoc comparison star counts
# the tqdm stuff just let me see a progress bar and gave me an estimate of how long it would take
mookodi_comp = []
shoc_comp = []

# iterating over the filtered TOIs and searching for Mookodi
for index, row in tqdm(toi_filtered.iterrows(), total=toi_filtered.shape[0], desc="Mookodi Search"):
    df = skymapper_query(str(row['RA (deg)']), str(row['Dec (deg)']))
    df_good = photometric_cleaning(df)
    mookodi, shoc = mookodi_search(df_good, row['TESS Mag'])
    mookodi_comp.append(mookodi)
    shoc_comp.append(shoc)

print(mookodi_comp)
```

[ ]:

```
# adding the comparison star counts to the filtered TOI dataframe
toi_filtered.loc[:, 'Mookodi Comp'] = mookodi_comp
toi_filtered.loc[:, 'SHOC Comp'] = shoc_comp

# keeping only rows with Mookodi count > 2
toi_mookodi = toi_filtered.loc[toi_filtered['Mookodi Comp'] > 2]

toi_mookodi.head()
```

[ ]:

```
# saving to a new csv file
toi_mookodi.to_csv('Tables/TESS_TOI_28Jan2025_Mookodi.csv', index=False)
```

### 1.0.3 Final lists

Below is the code for the lists provided in the writeup

[28]:

```
# first 10 potentially observable
toi_filtered['TOI'].head(10)
```

[28]:

```
0      101.01
2      103.01
6      107.01
7      108.01
8      109.01
9      110.01
12     113.01
13     114.01
14     115.01
16     117.01
```

```
Name: TOI, dtype: float64
```

```
[30]: # list the number of potentially observable stars from the initial cut
      len(toi_filtered['TOI'])
```

```
[30]: 975
```

```
[31]: # list the number of TOIs with >2 comparison stars in Mookodi field of view
      len(toi_mookodi['TOI'])
```

```
[31]: 706
```

```
[32]: # list the number of TOIs with >4 comparison stars in Mookodi field of view
      len(toi_mookodi[toi_mookodi['Mookodi Comp'] > 4]['TOI'])
```

```
[32]: 644
```

```
[33]: # list the number of TOIs with >2 comparison stars in SHOC field of view
      len(toi_mookodi[toi_mookodi['SHOC Comp'] > 2]['TOI'])
```

```
[33]: 330
```

```
[34]: # list the number of TOIs with >4 comparison stars in SHOC field of view
      len(toi_mookodi[toi_mookodi['SHOC Comp'] > 4]['TOI'])
```

```
[34]: 178
```