task2

January 24, 2024

```
[ ]: from astropy.timeseries import LombScargle
     from astropy import units as un
     import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
```

```
[ ]: data = pd.read_csv('./data/sksolartimevariation5804d.txt', skiprows=13,␣
      ↪sep='\s+', names=['t_mean(s)', 't_mean-t_start(s)', 't_end-t_mean(s)',␣
      ↪'nu_flux(1e6cm-2s-1)', 'flux_up_error(1e6cm-2s-1)',␣
      ↪'flux_down_error(1e6cm-2s-1)'])
```

```
[ ]: data
```

```
[ ]:        t_mean(s)  t_mean-t_start(s)  t_end-t_mean(s)  nu_flux(1e6cm-2s-1)  \
     0      833654760             170100           277380                 2.74
     1      834127080             175500           210060                 2.83
     2      834550800             213180           230160                 2.30
     3      834997020             199380           212640                 1.79
     4      835380420             170520           265680                 3.15
     …            …                  …                …                    …
     1338  1525315550             172739           172774                 2.36
     1339  1525703838             215064           215054                 2.26
     1340  1526138206             216970           216028                 1.88
     1341  1526588224             232102           226109                 1.90
     1342  1527014775             199299           208324                 2.60

           flux_up_error(1e6cm-2s-1)  flux_down_error(1e6cm-2s-1)
     0                          0.63                         0.53
     1                          0.75                         0.62
     2                          0.53                         0.45
     3                          0.55                         0.44
     4                          0.74                         0.61
     …                           …                            …
     1338                       0.36                         0.33
     1339                       0.31                         0.29
     1340                       0.33                         0.29
     1341                       0.38                         0.28
```

```
1342                          0.35                          0.33

[1343 rows x 6 columns]
```

[ ]:
```
times = data['t_mean(s)'].values * un.s
flux = data['nu_flux(1e6cm-2s-1)'].values * un.cm**-2 * un.s**-1 * 1e6
flux_err_up = data['flux_up_error(1e6cm-2s-1)'].values * un.cm**-2 * un.s**-1 *␣
 ↪1e6
flux_err_down = data['flux_down_error(1e6cm-2s-1)'].values * un.cm**-2 * un.
 ↪s**-1 * 1e6
```

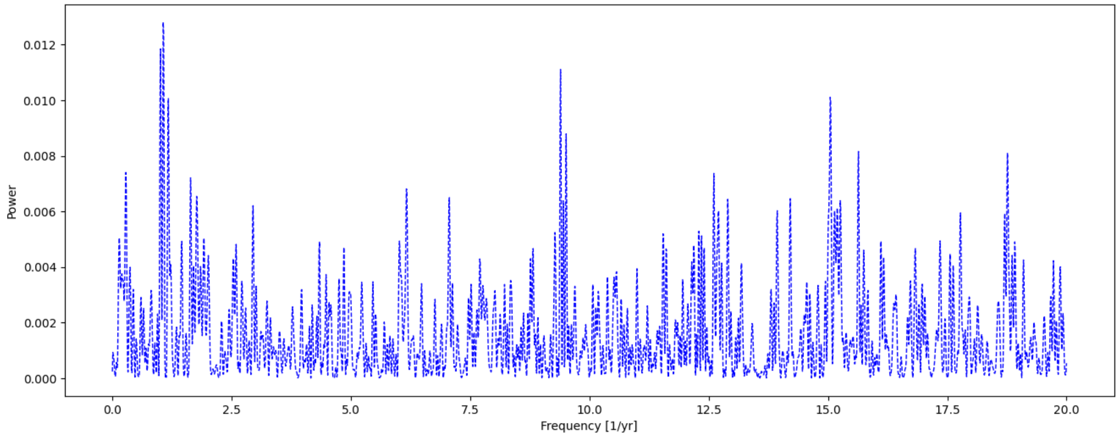
[ ]:
```
lsp = LombScargle(times, flux, dy=0.5*(flux_err_up+flux_err_down),␣
 ↪normalization='log')
```

[ ]:
```
freq, power = lsp.autopower(minimum_frequency=1e-8/un.year,␣
 ↪maximum_frequency=20/un.year)
```

[ ]:
```
freq
```

[ ]: $[1 \times 10^{-8},\ 0.0091028135,\ 0.018205617,\ \ldots,\ 19.980654,\ 19.989757,\ 19.998859]\ \frac{1}{\mathrm{yr}}$

[ ]:
```
plt.figure(figsize=(16, 6))
plt.plot(freq, power, '--', lw=1, c='b')
plt.xlabel('Frequency [1/yr]')
plt.ylabel('Power')
plt.show()
```



$\nu = 9.43$ 1/yr is the frquency from 2016 paper

Checking the False Alarm Probability (FAP) for the 2016 paper

[ ]:

```
freq_943 = freq[np.argmin(np.abs(freq - 9.43/un.year))]
power_943 = power[np.argmin(np.abs(freq - 9.43/un.year))]
```

[ ]:

```
power_943
```

[ ]: 0.00051287299

[ ]:

```
lsp.false_alarm_probability(power_943, method='bootstrap')
```

[ ]:

```
1.0
```

[ ]:

```
lsp.false_alarm_probability(power_943, method='naive')
```

[ ]: 1

[ ]:

```
lsp.false_alarm_probability(power_943, method='baluev')
```

[ ]: 1

[ ]:

```
lsp.false_alarm_probability(power_943, method='davies')
```

[ ]: 1413.2765

The *davies* method is giving **FAP > 1**, so I am specifying the optional min and max frequency kwargs.

[ ]:

```
lsp.false_alarm_probability(power_943, method='bootstrap',␣
 ↪minimum_frequency=freq.min(), maximum_frequency=freq.max())
```

[ ]:

```
1.0
```

[ ]:

```
lsp.false_alarm_probability(power_943, method='naive', minimum_frequency=freq.
 ↪min(), maximum_frequency=freq.max())
```

[ ]: 1

[ ]:

```
lsp.false_alarm_probability(power_943, method='baluev', minimum_frequency=freq.
 ↪min(), maximum_frequency=freq.max())
```

[ ]: 1

[ ]:

```
lsp.false_alarm_probability(power_943, method='davies', minimum_frequency=freq.
 ↪min(), maximum_frequency=freq.max())
```

[ ]: 185.57355

Davies method still erroneous

However, now use the frequency for which the *power is maximum*

$\nu = \nu_{max \quad power}$

[ ]:

```
freq_max = freq[np.nanargmax(power)]
power_max = power[np.nanargmax(power)]

freq_max
```

[ ]: $1.065028\ \frac{1}{\mathrm{yr}}$

[ ]:

```
power.max()
```

[ ]: 0.012813377

[ ]:

```
lsp.false_alarm_probability(power_max, method='bootstrap')
```

[ ]:

```
0.897
```

[ ]:

```
lsp.false_alarm_probability(power_max, method='naive')
```

[ ]: 0.46610671

[ ]:

```
lsp.false_alarm_probability(power_max, method='baluev')
```

[ ]: 0.84534996

[ ]:

```
lsp.false_alarm_probability(power_max, method='davies')
```

[ ]: 1.8665905

Davies method still giving invalid FAP

Specifying the min and max frequency kwargs to see if it works

[ ]:

```
lsp.false_alarm_probability(power_max, method='bootstrap',␣
 ↪minimum_frequency=freq.min(), maximum_frequency=freq.max())
```

[ ]:

```
0.269
```

[ ]:

```
lsp.false_alarm_probability(power_max, method='naive', minimum_frequency=freq.
 ↪min(), maximum_frequency=freq.max())
```

[ ]: 0.078847271

[ ]:

```
lsp.false_alarm_probability(power_max, method='baluev', minimum_frequency=freq.
 ↪min(), maximum_frequency=freq.max())
```

[ ]: 0.21686123

[ ]:

```
lsp.false_alarm_probability(power_max, method='davies', minimum_frequency=freq.
 ↪min(), maximum_frequency=freq.max())
```

[ ]: 0.24444535

Using all powers from the LSP

[ ]:

```
fap_davies = lsp.false_alarm_probability(power, method='davies',␣
 ↪minimum_frequency=freq.min(), maximum_frequency=freq.max())
```

[ ]:

```
print(fap_davies.min(), fap_davies.max())
```

```
0.24444534729856113 191.3322277383209
```

[ ]:

```
fap_baluev = lsp.false_alarm_probability(power, method='baluev',␣
 ↪minimum_frequency=freq.min(), maximum_frequency=freq.max())
print(fap_baluev.min(), fap_baluev.max())
```

```
0.21686122583299355 1.0
```

[ ]:

```
fap_naive = lsp.false_alarm_probability(power, method='naive',␣
 ↪minimum_frequency=freq.min(), maximum_frequency=freq.max())
print(fap_naive.min(), fap_naive.max())
```

```
0.07884727093714422 1.0
```

[ ]:

```
fap_bootstrap = lsp.false_alarm_probability(power, method='bootstrap',␣
 ↪minimum_frequency=freq.min(), maximum_frequency=freq.max())
print(fap_bootstrap.min(), fap_bootstrap.max())
```

```
0.263 1.0
```

0.0.1 Davies method has some issues??