Coding for Astronomy

Answering astronomical question with Python code

Tony Rice NASA Ambassador & WRAL Science Contributor

What is coding?

MetPy

Siphon

GeoPandas

Cartoony

Questions

- How rare are Halloween full Moons?
- When does the book and movie "The Martian" take place?
- When is the best time to see Mercury?
- When will the International Space Station next visible?
- When will the next total eclipse pass through Alabama? Where will the path be?

Eclipse Paths

When will the next total solar eclipse pass through southern Alabama? Where will the path be?

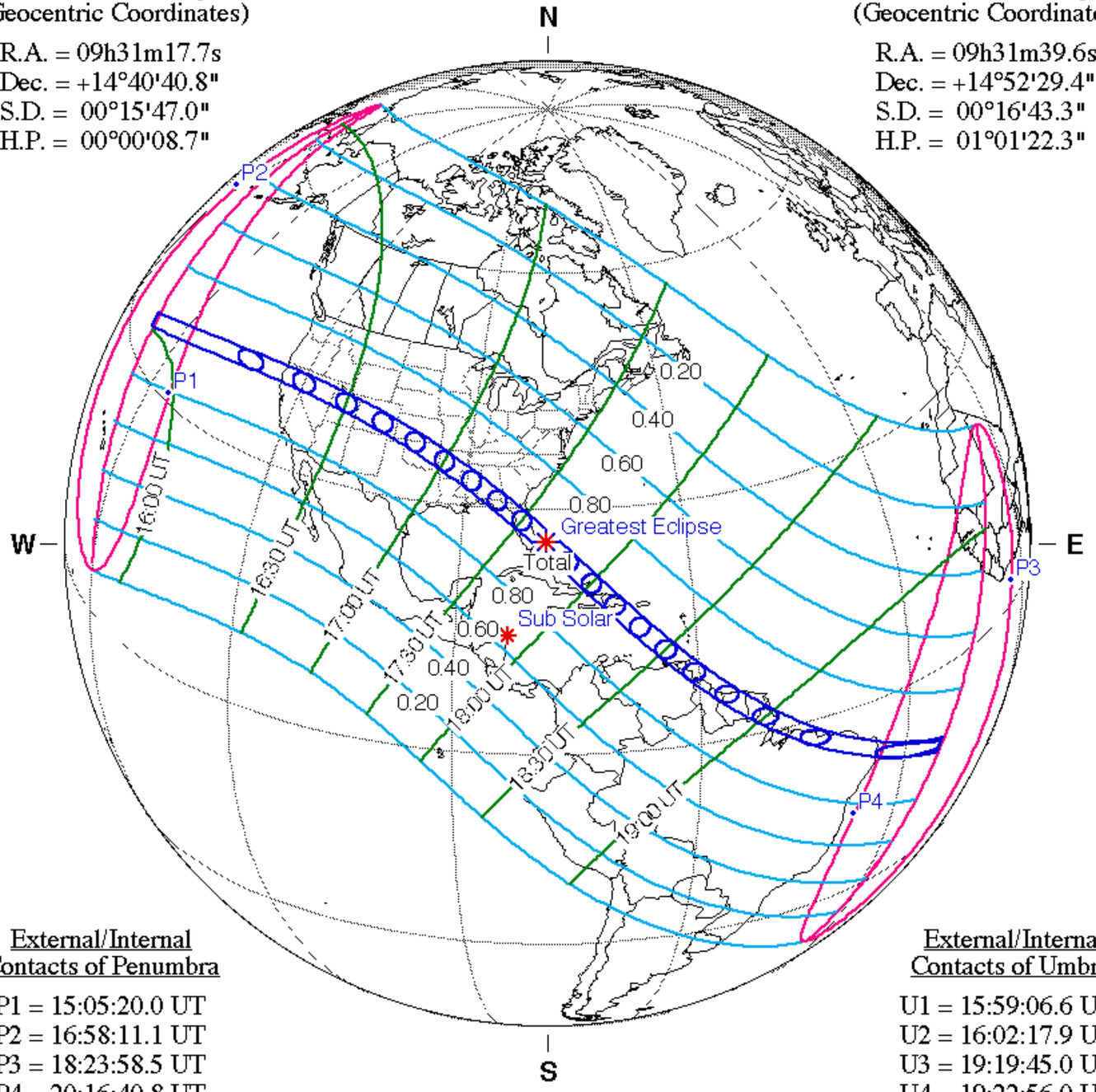
Image: NASA Scientific Visualization Studio

Sun at Greatest Eclipse
(Geocentric Coordinates)

R.A. = 09h31m17.7s
Dec. = +14°40'40.8"
S.D. = 00°15'47.0"
H.P. = 00°00'08.7"

Moon at Greatest Eclipse
(Geocentric Coordinates)

R.A. = 09h31m39.6s
Dec. = +14°52'29.4"
S.D. = 00°16'43.3"
H.P. = 01°01'22.3"



External/Internal
Contacts of Penumbra

P1 = 15:05:20.0 UT
P2 = 16:58:11.1 UT
P3 = 18:23:58.5 UT
P4 = 20:16:40.8 UT

External/Internal
Contacts of Umbra

U1 = 15:59:06.6 UT
U2 = 16:02:17.9 UT
U3 = 19:19:45.0 UT
U4 = 19:22:56.0 UT

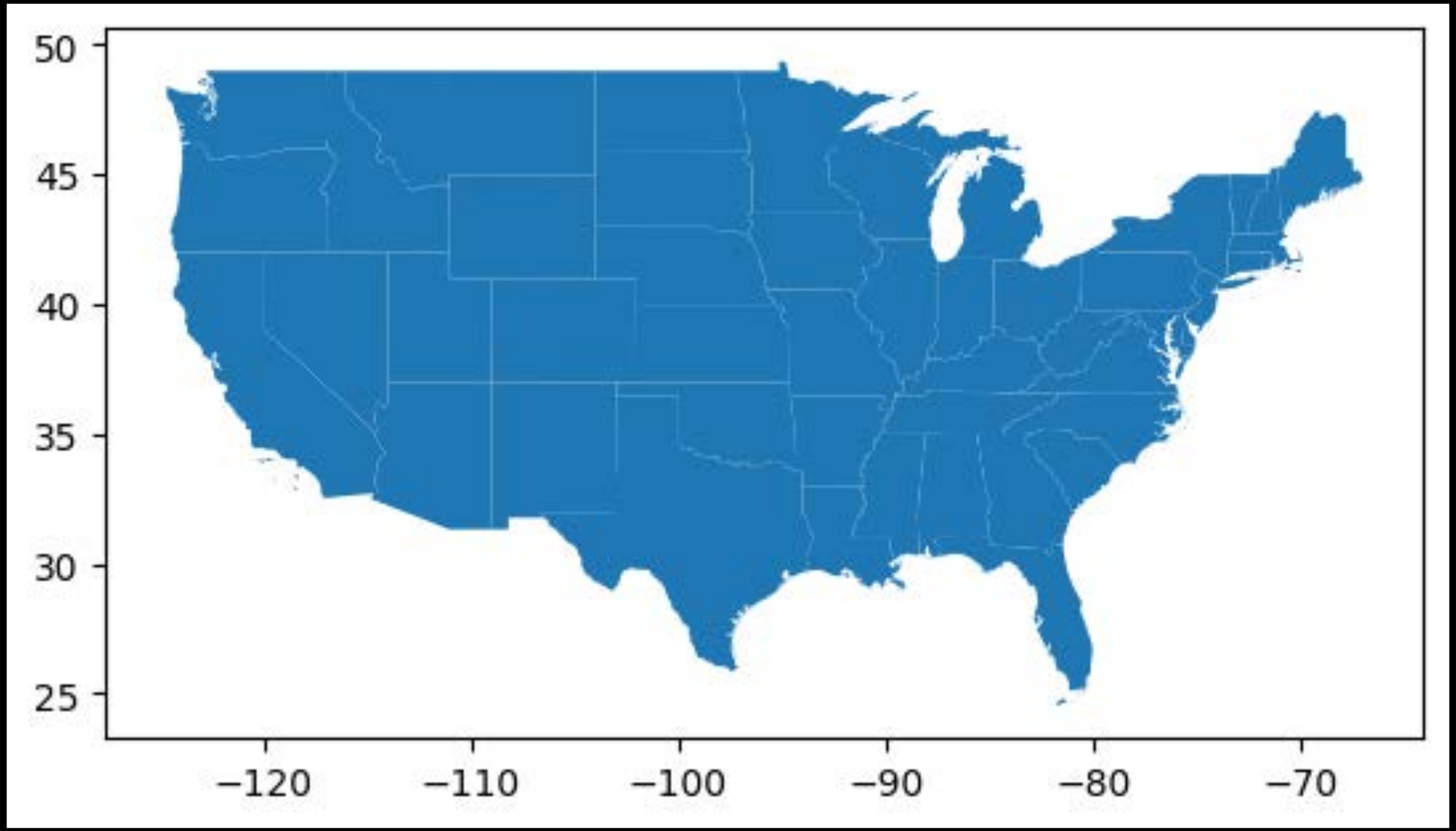
Local Circumstances at Greatest Eclipse

Lat. = 25°53.8'N Sun Alt. = 77.7°

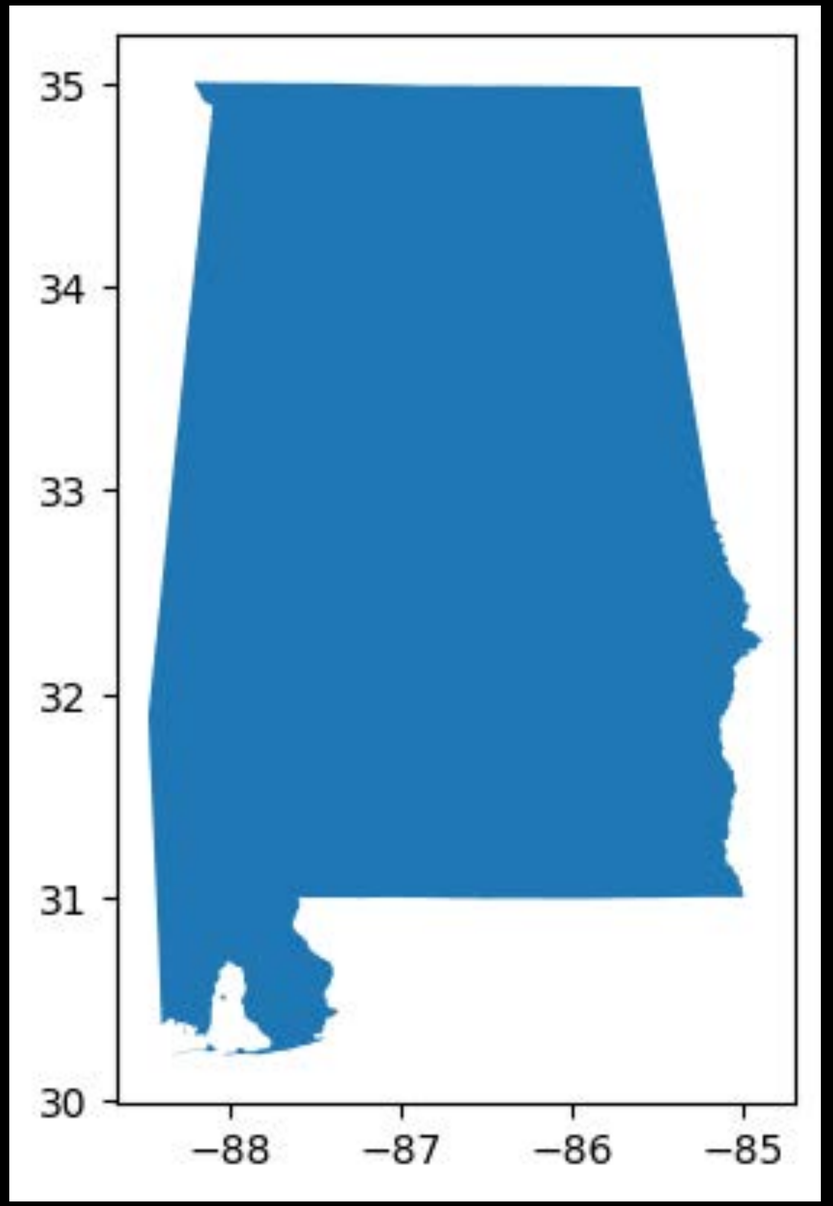
```
import geopandas as gpd
import matplotlib.pyplot as plt
from shapely.ops import unary_union


# fetch US Census Bureau shape files
url="https://github.com/rtphokie/ncmns_astronomy_days_demo/raw/main/cb_2018_us_state_500k/cb_2018_us_state_500k.shp"
us_states = gpd.read_file('/vsicurl/'+url)

us_states = us_states.to_crs('EPSG:4326') # mercator projection

lower48 = us_states[~us_states.NAME.isin(['Alaska', 'Hawaii', 'United States Virgin Islands',
                                          'American Samoa','Puerto Rico','Guam',
                                          'Commonwealth of the Northern Mariana Islands'])]
```

STUSPS	NAME	LAND AREA (m²)	WATER AREA (m²)	Geometry
AL	Alabama	131,174,048,583	4,593,327,154	MULTIPOLYGON (((-88.05338 30.50699, -88.05109 ...

```
lower48.plot()
```



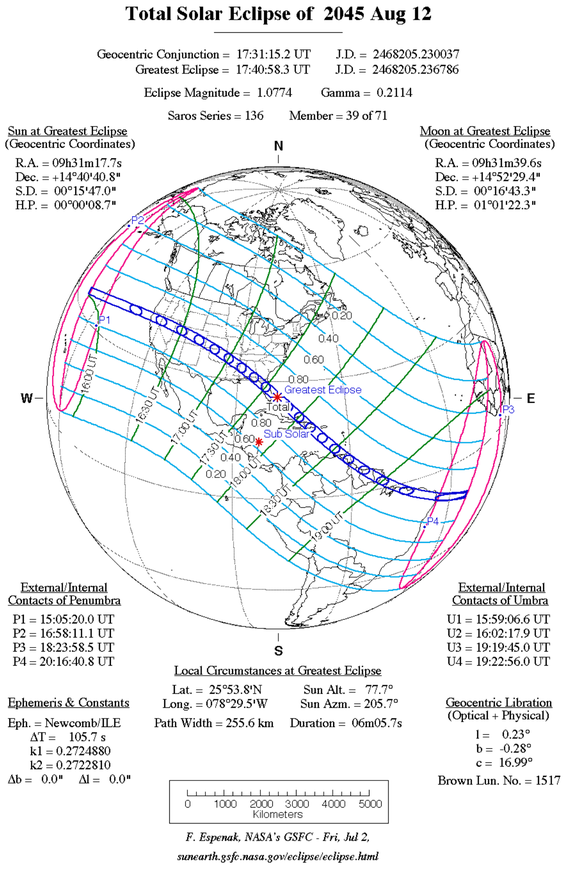
```
AL = lower48[lower48.NAME=='Alabama']

AL.plot()
```



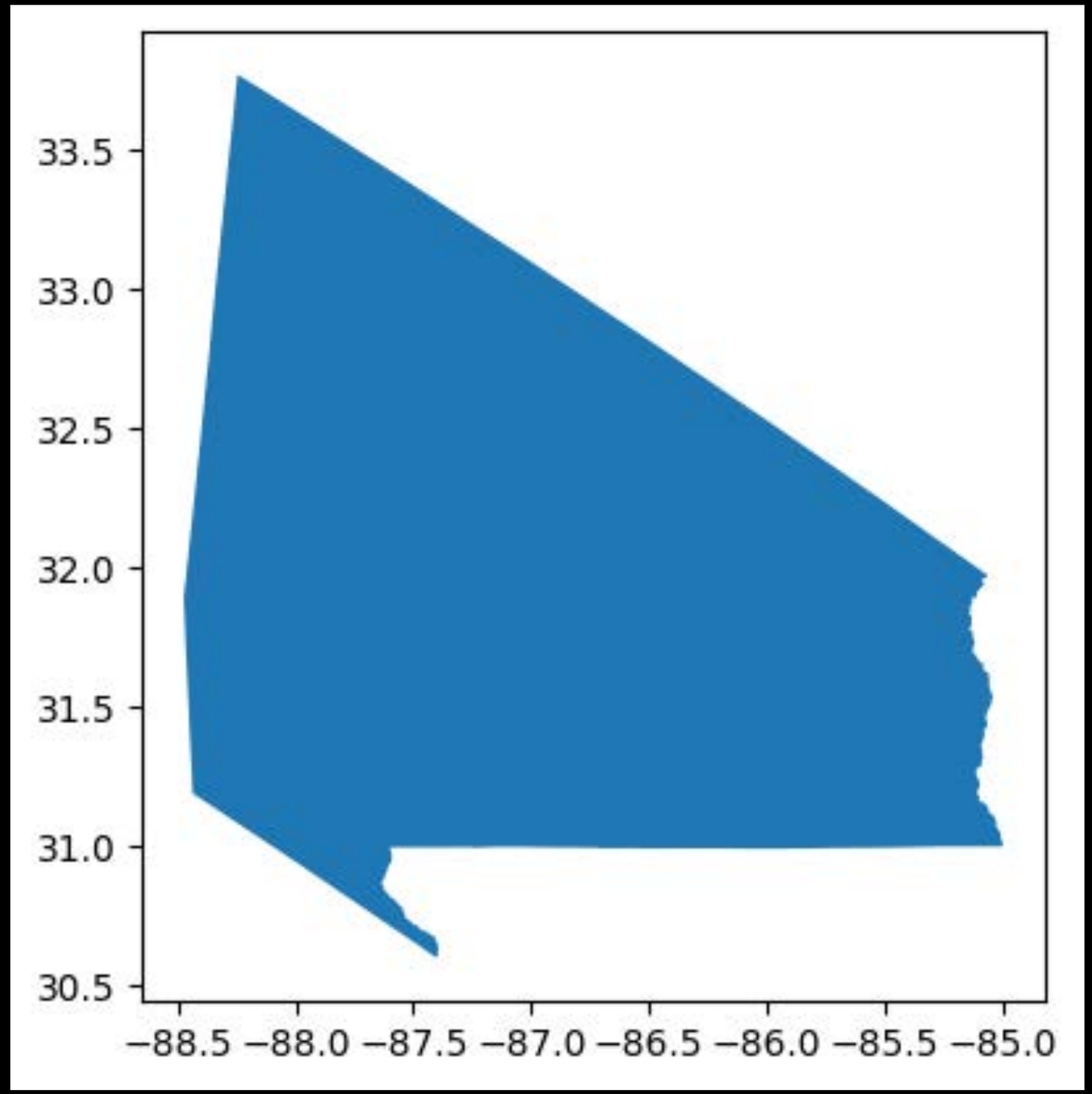
```
# fetch eclipse paths
url='https://github.com/rtphokie/ncmns_astronomy_days_demo/raw/main/SE_1950_to_2200/SE_1950_to_2200.shp'
eclipse_gdf = gpd.read_file('/vsicurl/'+url).to_crs('EPSG:4326')
```

eclipsetype	linelabel	date	geometry
total	path	2045-08-12	POLYGON ((48.01758 -72.26721, 48 -72.26739, 47...

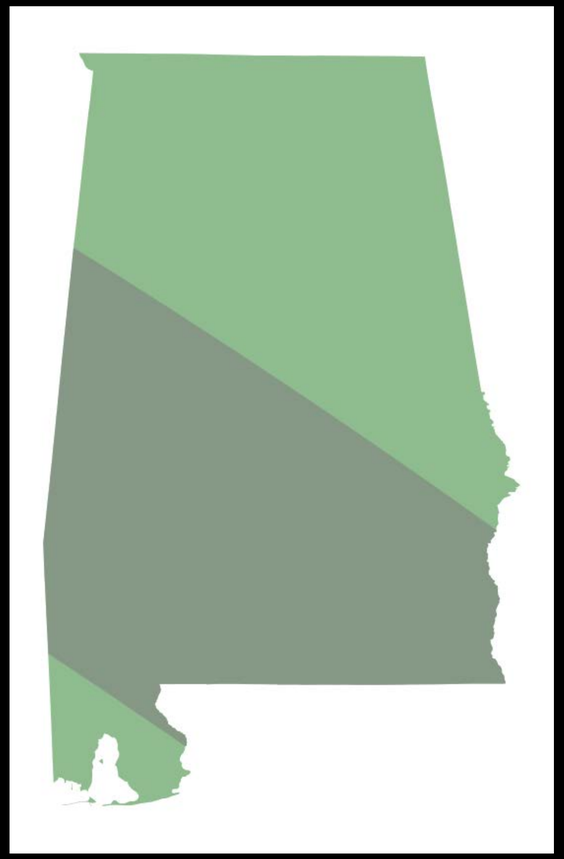
```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(12, 9))
ax.axis('off')

lower48.plot(ax=ax, color='darkseagreen')

eclipse_paths[eclipse_paths.date=='2045-08-12'].plot(ax=ax, alpha=.5)
```



```
AL_eclipses[AL_eclipses.date=='2045-08-12'].plot()
```



```
AL.plot(ax=ax, color='darkseagreen')
AL_eclipses[AL_eclipses.date=='2045-08-12'].plot(ax=ax, color="grey", alpha=0.6)
```



```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(12, 9))
fig.tight_layout()
ax.axis('off')

AL.plot(ax=ax, color='darkseagreen')
AL_eclipses[AL_eclipses.eclipsetype == 'Total'].plot(ax=ax, color="grey", alpha=0.6)
AL_eclipses[AL_eclipses.eclipsetype == 'Annular'].plot(ax=ax, color="orange", alpha=0.6)
AL_eclipses[AL_eclipses.eclipsetype == 'Hybrid'].plot(ax=ax, color="red", alpha=0.6)
```

Satellite Passes

Whe will the ISS next be visible from Mobile, AL?

Image: NASA/JSC

Fetch NORAD satellite

```
from skyfield.api import load, wgs84
import pytz
import pylab as pl
import matplotlib.pyplot as plt

ts = load.timescale()

stations_url = 'http://celestrak.com/NORAD/elements/stations.txt'
satellites = load.tle_file(stations_url)
```

```
Loaded 16 Satellites
<EarthSatellite ISS (ZARYA) catalog #25544 epoch 2025-03-27 04:07:34 UTC>
<EarthSatellite ISS DEB catalog #62376 epoch 2025-03-27 03:10:14 UTC>
<EarthSatellite CYGNUS NG-21 catalog #60378 epoch 2025-03-26 20:23:24 UTC>
<EarthSatellite CREW DRAGON 10 catalog #63204 epoch 2025-03-26 20:23:24 UTC>]
<EarthSatellite SOYUZ-MS 26 catalog #61043 epoch 2025-03-26 20:23:24 UTC>
<EarthSatellite PROGRESS-MS 29 catalog #62030 epoch 2025-03-26 20:23:24 UTC>
<EarthSatellite FREGAT DEB catalog #49271 epoch 2025-03-26 10:04:27 UTC>
<EarthSatellite PROGRESS-MS 30 catalog #63129 epoch 2025-03-26 20:23:24 UTC>
<EarthSatellite CSS (TIANHE) catalog #48274 epoch 2025-03-27 04:15:51 UTC>
<EarthSatellite 1998-067XC catalog #62296 epoch 2025-03-27 01:29:06 UTC>
```

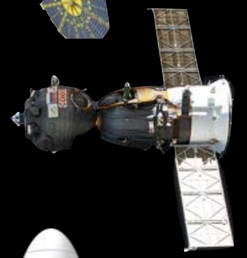
ISS (ZARYA)
1 25544U 98067A   25086.17192136 .00035516 00000+0 62421-3 0 9998
2 25544 51.6371 353.2392 0003699 54.8873 305.2462 15.50144996502445

CSS (TIANHE)
1 48274U 21035A   25086.17767694 .00061137 00000+0 66955-3 0 9995
2 48274 41.4656 159.9905 0005012 105.6102 254.5291 15.62349305223333

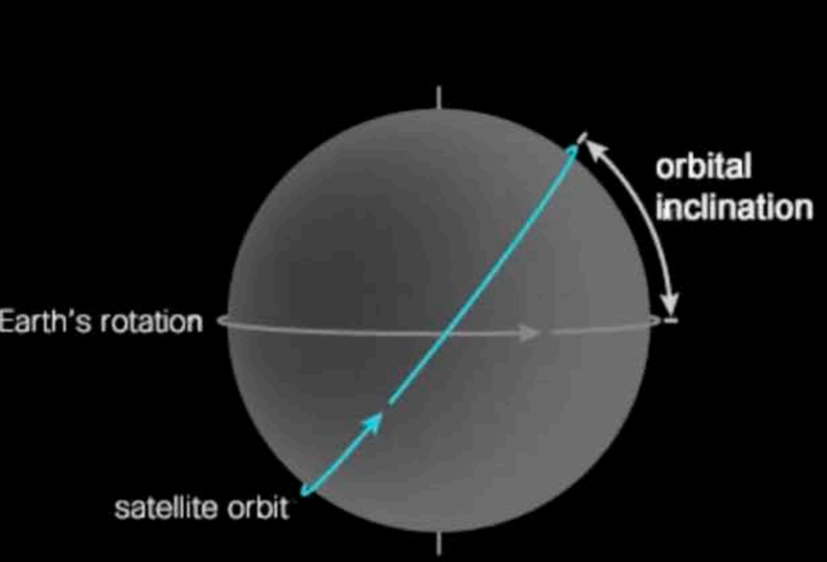
FREGAT DEB
1 49271U 11037PF 25085.41975788 .00006664 00000+0 15491-1 0 9996
2 49271 51.6296 249.4253 0892588 13.7782 348.5808 12.28238727169735

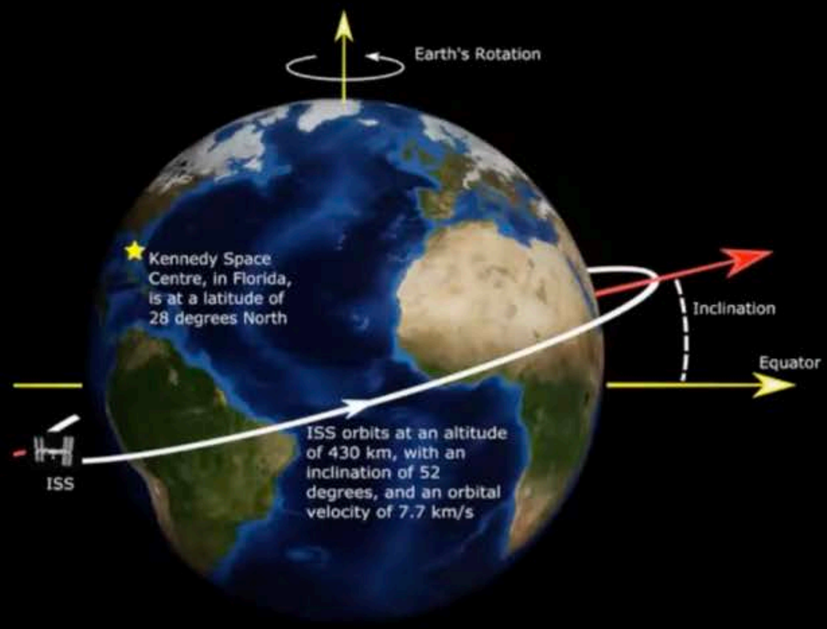
CYGNUS NG-21
1 60378U 24139A   25085.84958103 .00035509 00000+0 62456-3 0 9998
2 60378 51.6370 354.8360 0003638 54.8357 305.2972 15.50123203501533

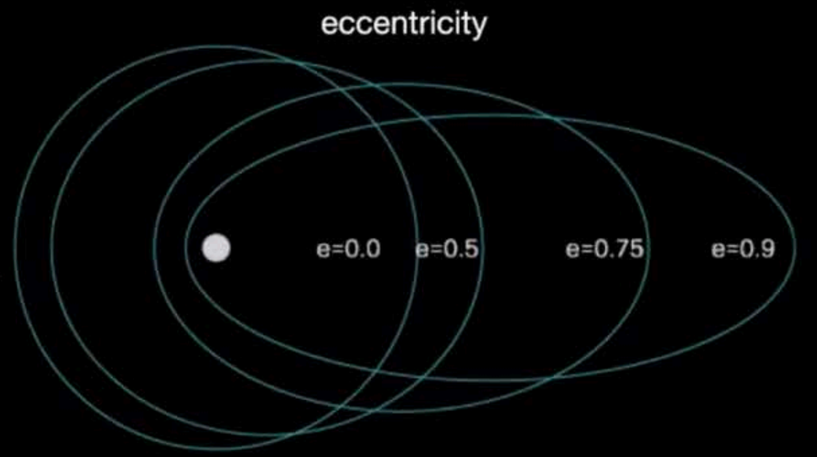
SOYUZ-MS 26
1 61043U 24162A   25085.84958103 .00035509 00000+0 62456-3 0 9994
2 61043 51.6370 354.8360 0003638 54.8357 305.2972 15.50123203501533



CREW DRAGON 10
1 63204U 25049A   25085.84958103 .00035509 00000+0 62456-3 0 9990
2 63204 51.6370 354.8360 0003638 54.8357 305.2972 15.50123203501534





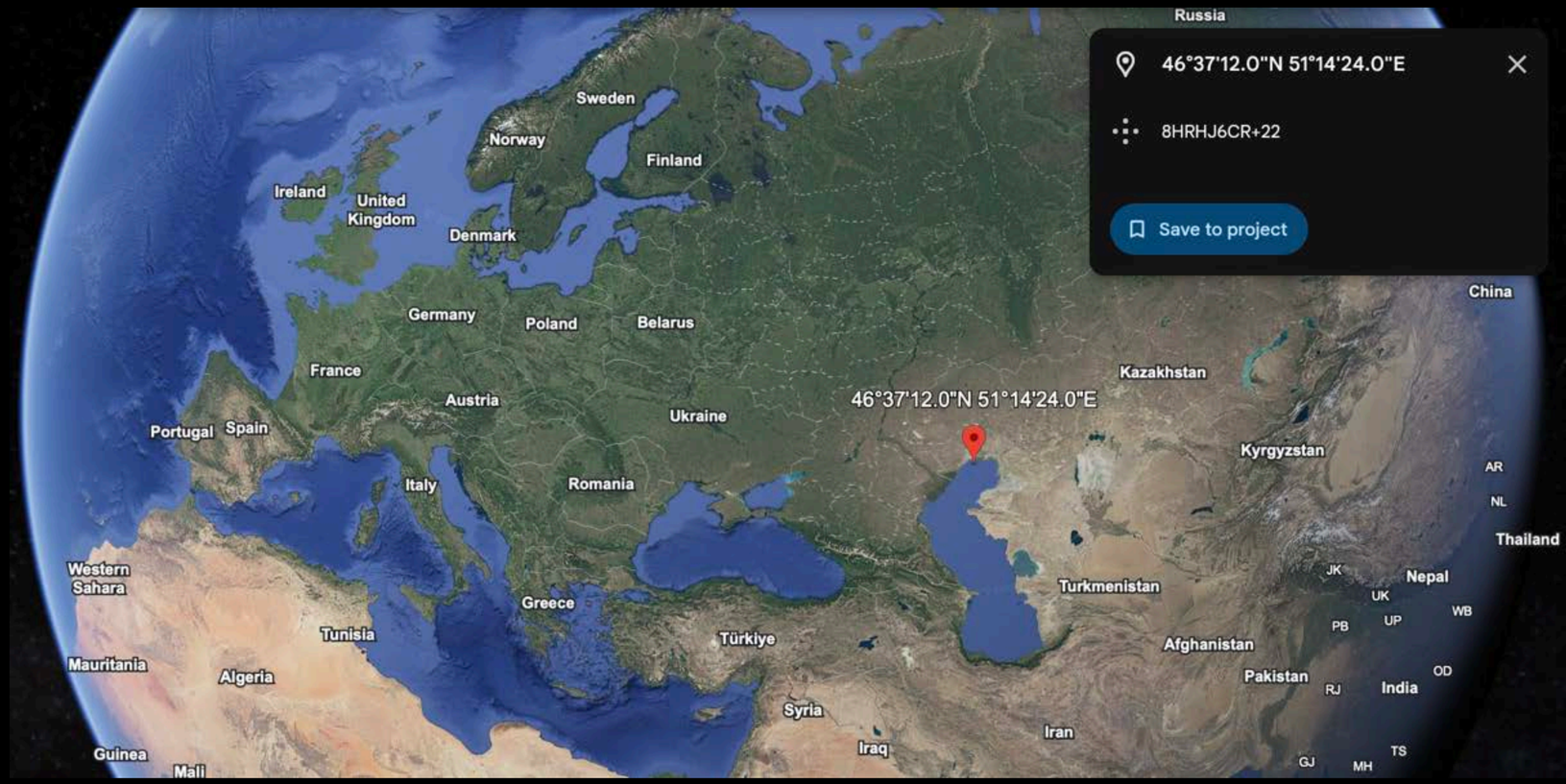


	Satellite #	Intl Designator	2021-01-30 12:01:55.0.1634	change in revs/day	orbit decay	drag term		counter
1	25544U	98067A	21030.50133291	.00003153	00000-0	65512-4	0	9995
2	25544	51.6465	303.7326	0002392	315.8456	215.8385	15.48928800	267250
		Inclination	right ascension	eccentricity	perigee	anomaly	orbits per day	checksum

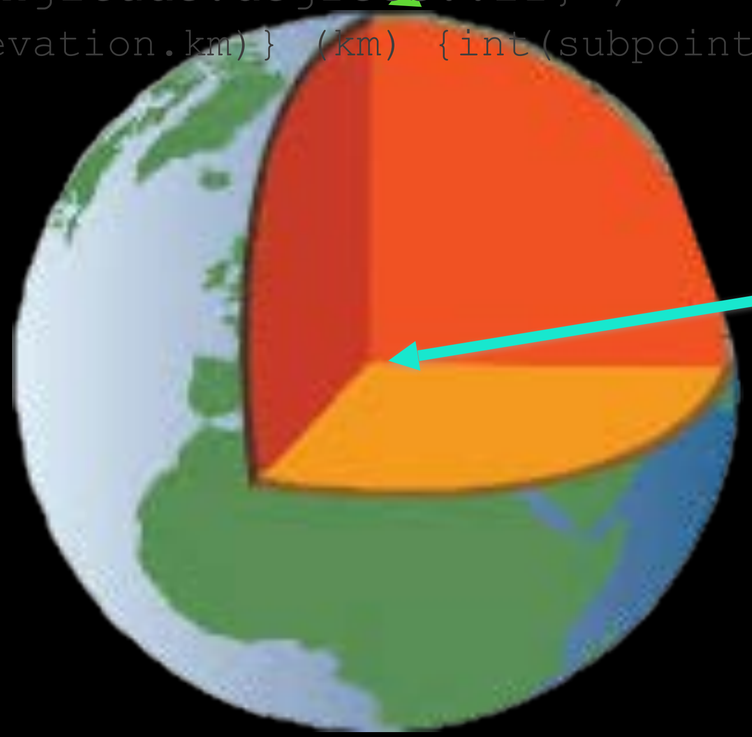
Where is the ISS now?

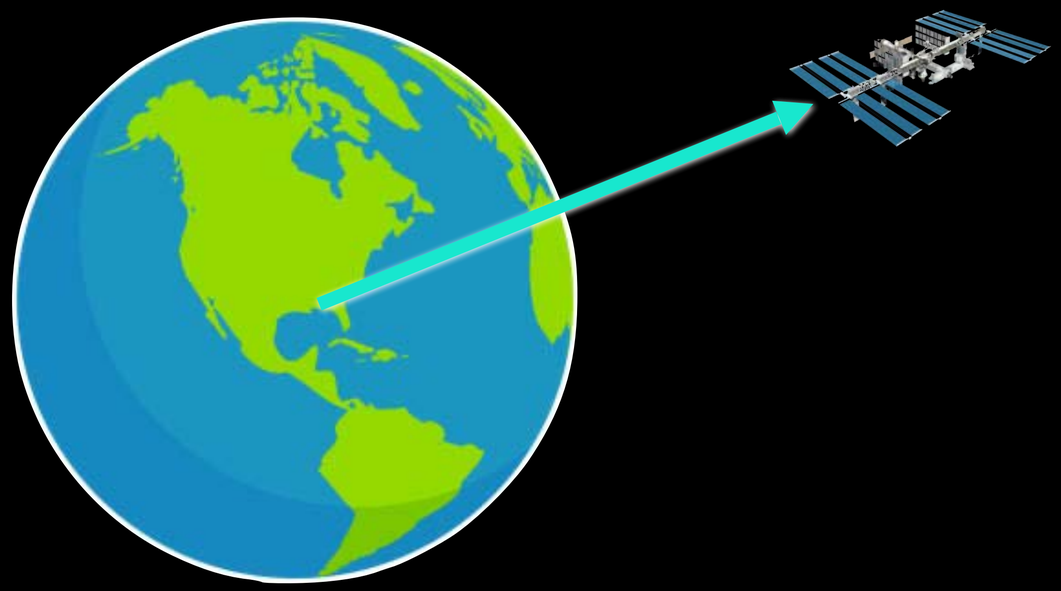
```
by_name = {sat.name: sat for sat in satellites}
satellite = by_name['ISS (ZARYA)']

t = ts.now()
geocentric = satellite.at(t)
subpoint = wgs84.subpoint(geocentric)

print(f'Latitude:  {subpoint.latitude.degrees:.2f}')
print(f'Longitude: {subpoint.longitude.degrees:.2f}')
print(f"Elevation {int(subpoint.elevation.km)} (km) {int(subpoint.elevation.km*0.621371)} (mi)")
```

```
Latitude:  -48.15
Longitude: -99.22
Elevation 435 (km) 270 (mi)
```



```
by_name = {sat.name: sat for sat in satellites}
satellite = by_name['ISS (ZARYA)']

t = ts.now()
geocentric = satellite.at(t)
subpoint = wgs84.subpoint(geocentric)

print(f'Latitude:  {subpoint.latitude.degrees:.2f}')
print(f'Longitude: {subpoint.longitude.degrees:.2f}')
print(f"Elevation {int(subpoint.elevation.km)} (km) {int(subpoint.elevation.km*0.621371)} (mi)")

Latitude:  -48.15
Longitude: -99.22
Elevation 435 (km) 270 (mi)
```





```
local_lat=30.6959
local_lng=-88.1842
local_tz = pytz.timezone('US/Central')
```

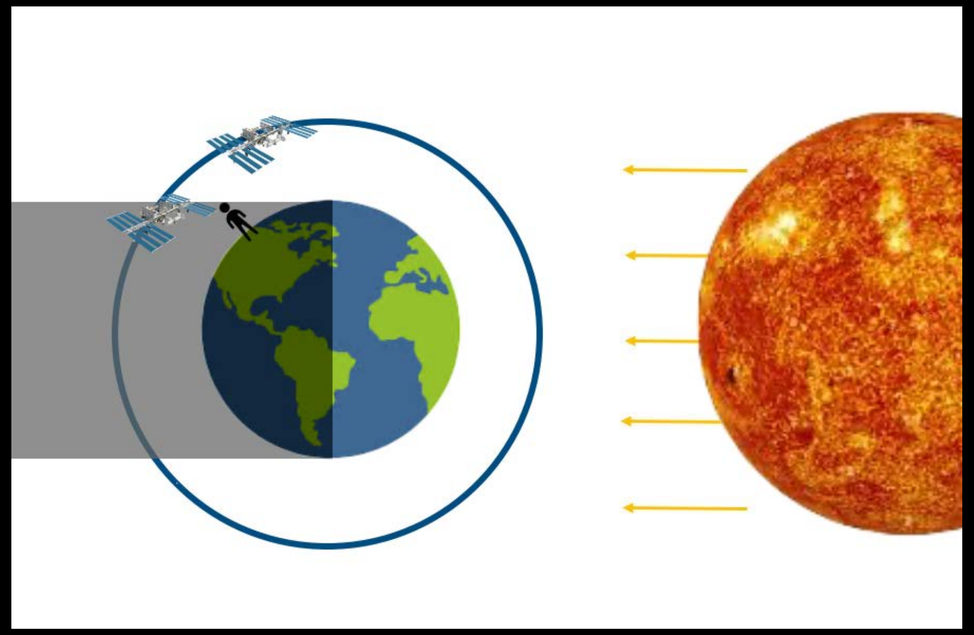
```
local_coords = wgs84.latlon(local_lat,local_lng, elevation_m=5)

difference = satellite - local_coords
```

```
topocentric = difference.at(t)
alt, az, distance = topocentric.altaz()
print(f"alt {alt}, az {az} distance {int(distance.km)} (km)
{int(distance.km*0.621371):,} (mi)")
```

```
alt -39deg 57' 47.0", az 196deg 35' 51.9" distance 8800 (km) 5,468 (mi)
```

```
t0 = ts.utc(2025, 3, 29)
t1 = ts.utc(2025, 3, 30)

t, events = satellite.find_events(local_coords, t0, t1,
                                  altitude_degrees=10)

for ti, event in zip(t, events):
    name = ('rise above 10°', 'culminate', 'set below 10°')[event]
    print(ti.utc_strftime('%Y %b %d %H:%M:%S'), name)
```

```
2025 Mar 29 01:30:35 rise above 10°
2025 Mar 29 01:33:12 culminate
2025 Mar 29 01:35:49 set below 10°

2025 Mar 29 03:07:03 rise above 10°
2025 Mar 29 03:09:53 culminate
2025 Mar 29 03:12:43 set below 10°

2025 Mar 29 18:08:25 rise above 10°
2025 Mar 29 18:11:41 culminate
2025 Mar 29 18:14:58 set below 10°
```



Sun image: NASA/SDO

Show only visible passes

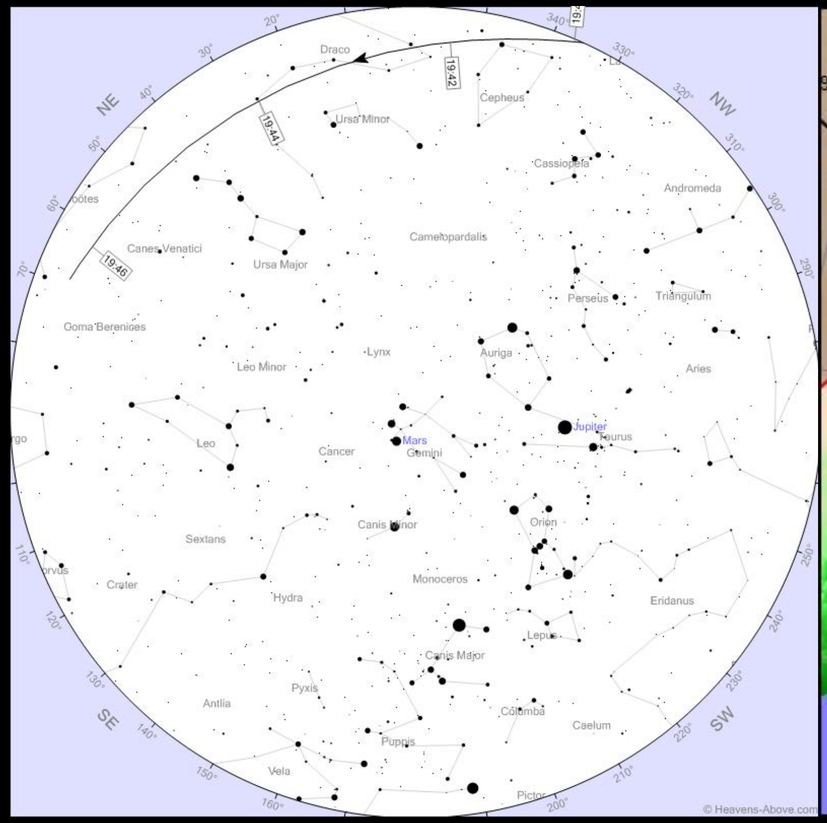
```
eph = load('de421.bsp') # ephemeris from NASA JPL

sun, earth = eph['sun'], eph['earth']

def sun_pos(t, topopos):
  obs = earth + local_coords
  astrometric = obs.at(t).observe(sun)
  apparent = obs.at(t).observe(sun).apparent()
  alt, az, dist = apparent.altaz()

  return alt, az, dist

for ti, event in zip(t, events):
    name = (f'rise above {horizon}°', 'highest point', f'set below {horizon}°')[event]
    sunalt, sunaz, sundist = sun_pos(ti, local_coords)
    if -12 <= sunalt.degrees <= -6:
      print(ti.utc_jpl(), name)
```

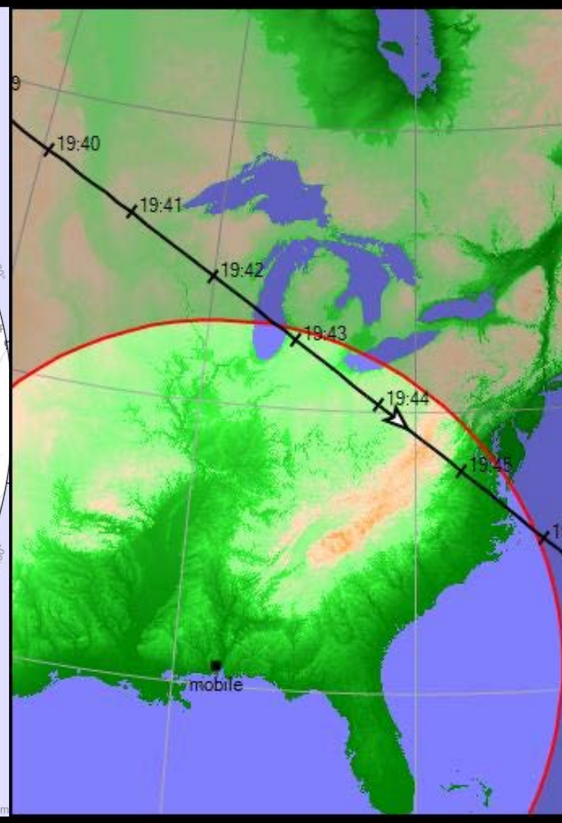
```
for ti, event in zip(t, events):
    name = (f'rise above {horizon}°', 'highest point', f'set below {horizon}°')[event]
    sunalt, sunaz, sundist = sun_pos(ti, local_coords)
    if -12 <= sunalt.degrees <= -6:
      print(ti.astimezone(local_tz).strftime('%a %b %d %-I:%M:%S %p'), name)
      if event == 0:
          print(ti.utc_jpl(), name)
```

```
A.D. 2025-Mar-30 00:42:45.0603 UTC rise above 10°
A.D. 2025-Mar-30 00:44:21.4859 UTC highest point
A.D. 2025-Mar-30 00:45:57.6806 UTC set below 10°

A.D. 2025-Apr-01 00:40:42.5949 UTC rise above 10°
A.D. 2025-Apr-01 00:43:45.9663 UTC highest point
A.D. 2025-Apr-01 00:46:49.0549 UTC set below 10°

A.D. 2025-Apr-03 00:39:21.2876 UTC rise above 10°
A.D. 2025-Apr-03 00:42:39.0706 UTC highest point
A.D. 2025-Apr-03 00:45:56.3080 UTC set below 10°
```

```
t, events = satellite.find_events(local_coords, t0, t1, altitude_degrees=10)
for ti, event in zip(t, events):
    name = (f'rise above {horizon}°', 'highest point', f'set below {horizon}°')[event]
    sunalt, sunaz, sundist = sun_pos(ti, local_coords)
    if -12 <= sunalt.degrees <= -6:
      print(ti.astimezone(local_tz).strftime('%a %b %d %-I:%M:%S %p %Z'), name)
```

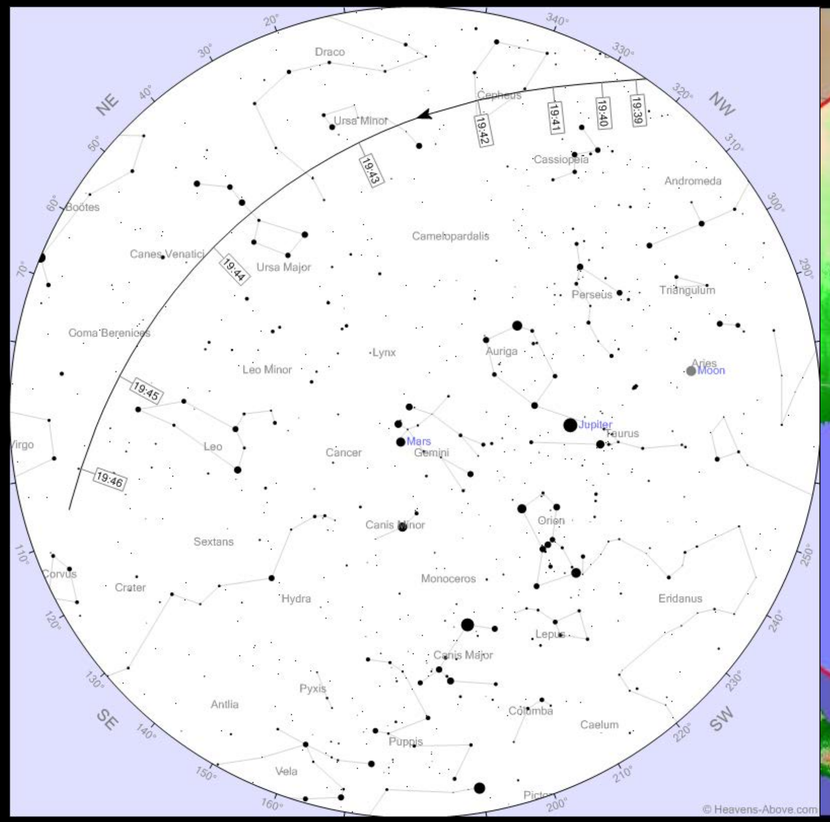
```
Sat Mar 29 7:42:45 PM CDT rise above 10°
Sat Mar 29 7:44:21 PM CDT highest point
Sat Mar 29 7:45:57 PM CDT set below 10°

Mon Mar 31 7:40:42 PM CDT rise above 10°
Mon Mar 31 7:43:45 PM CDT highest point
Mon Mar 31 7:46:49 PM CDT set below 10°

Wed Apr 02 7:39:21 PM CDT rise above 10°
Wed Apr 02 7:42:39 PM CDT highest point
Wed Apr 02 7:45:56 PM CDT set below 10°
```





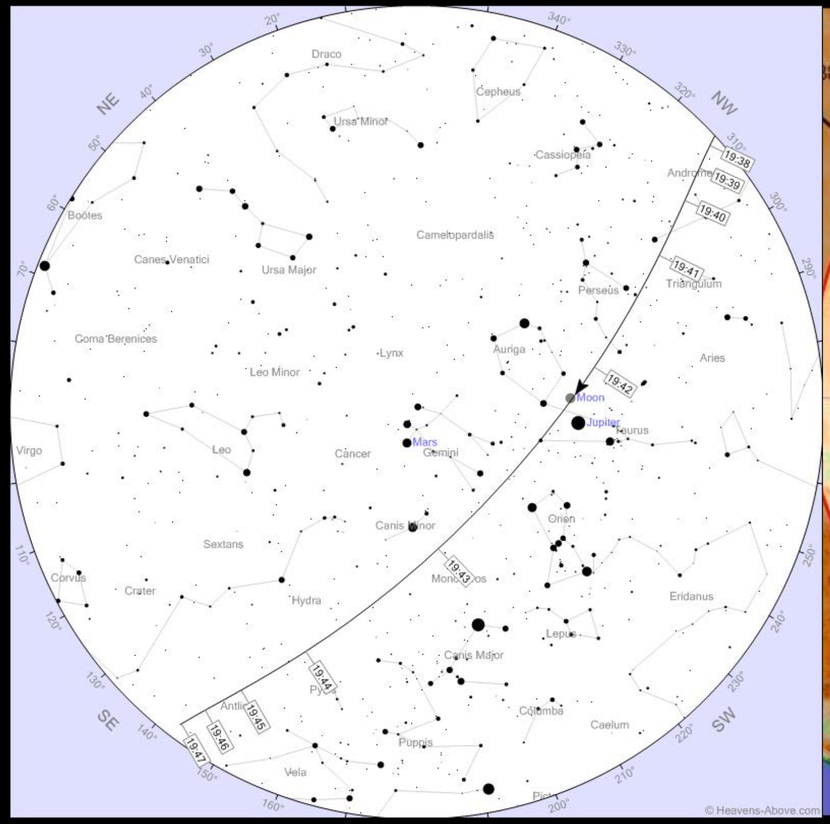
```
t, events = satellite.find_events(local_coords, t0, t1, altitude_degrees=10)
for ti, event in zip(t, events):
    name = (f'rise above {horizon}°', 'highest point', f'set below {horizon}°')[event]
    sunalt, sunaz, sundist = sun_pos(ti, local_coords)
    if -12 <= sunalt.degrees <= -6:
      print(ti.astimezone(local_tz).strftime('%a %b %d %-I:%M:%S %p %Z'), name)
```

```
Sat Mar 29 7:42:45 PM CDT rise above 10°
Sat Mar 29 7:44:21 PM CDT highest point
Sat Mar 29 7:45:57 PM CDT set below 10°

Mon Mar 31 7:40:42 PM CDT rise above 10°
Mon Mar 31 7:43:45 PM CDT highest point
Mon Mar 31 7:46:49 PM CDT set below 10°

Wed Apr 02 7:39:21 PM CDT rise above 10°
Wed Apr 02 7:42:39 PM CDT highest point
Wed Apr 02 7:45:56 PM CDT set below 10°
```



```
t, events = satellite.find_events(local_coords, t0, t1, altitude_degrees=10)
for ti, event in zip(t, events):
    name = (f'rise above {horizon}°', 'highest point', f'set below {horizon}°')[event]
    sunalt, sunaz, sundist = sun_pos(ti, local_coords)
    if -12 <= sunalt.degrees <= -6:
      print(ti.astimezone(local_tz).strftime('%a %b %d %-I:%M:%S %p %Z'), name)
```

```
Sat Mar 29 7:42:45 PM CDT rise above 10°
Sat Mar 29 7:44:21 PM CDT highest point
Sat Mar 29 7:45:57 PM CDT set below 10°

Mon Mar 31 7:40:42 PM CDT rise above 10°
Mon Mar 31 7:43:45 PM CDT highest point
Mon Mar 31 7:46:49 PM CDT set below 10°

Wed Apr 02 7:39:21 PM CDT rise above 10°
Wed Apr 02 7:42:39 PM CDT highest point
Wed Apr 02 7:45:56 PM CDT set below 10°
```



Learning more

- python.org/about/gettingstarted
- learnpython.org
- rhodesmill.org/skyfield
- ECMWF Summer of Weather Code
- GitHub Introduction to Python for Atmospheric Science & Meteorology