earth_state_vector

May 29, 2021

William D. Taylor

Space Science with Python: Part 1

Compute Earth's position and velocity vector for tonight at midnight.

[1]:
```
# imports
import spiceypy
import datetime
import math
```

Thorughout the course, use the SPICE docs to see which functions to use: https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/index.html

For this exercise, we'll need the `spkgeo()` function.
parameters: target body (Earth, in our case), ephemeris time, reference frame, and observer (Sun)

[2]:
```
# first, let's calculate ephemeris time (ET) for midnight tonight

# get today's date as a string, replace current time with UTC midnight
TODAY = datetime.datetime.today()

TODAY = TODAY.strftime("%Y-%m-%dT00:00:00")
```

Here we need to use the `furnsh()` (furnish) function from the SPICE library
to load a kernel.

[3]:
```
# furnish the necessary kernel
spiceypy.furnsh("../kernels/lsk/naif0012.tls")

# convert UTC midnight to ET
ET_TODAY_MIDNIGHT = spiceypy.utc2et(TODAY)
ET_TODAY_MIDNIGHT
```

[3]:
```
675518469.1849737
```

Next we need to use `spkgeo()` to compute the Earth's state vector.
A state vector is a position and velocity vector.

Recall that `spkgeo()` takes in the target body and observer.

For this example, Earth and the Sun respectively. SPICE uses NAIF IDs instead of strings to identify these bodies: https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/req/naif_ids.html

[4]:
```python
# Earth's NAIF ID is 399, and the Sun's is 1
# our reference frame is "ECLIPJ2000," Earth's ecliptic plane
# for the year 2000. (this is the Sun's apparent path of orbit)

# here we need an spk (Spacecraft and Planet Kernel)
spiceypy.furnsh("../kernels/spk/de432s.bsp")

# and we can finally run our computation
EARTH_STATE_WRT_SUN, EARTH_SUN_LT = spiceypy.spkgeo(targ = 399,
                                                    et = ET_TODAY_MIDNIGHT,
                                                    ref = "ECLIPJ2000",
                                                    obs = 10)
EARTH_STATE_WRT_SUN
```

[4]:
```
array([-5.78507631e+07, -1.40147949e+08,  7.04099522e+03,  2.70372213e+01,
       -1.14793582e+01,  1.51605913e-03])
```

Let's check our results.
To check our position vector, compute the distance between the Sun and the Earth.
Convert this from km to AU, and we should get a number close to one.

[5]:
```python
EARTH_SUN_DISTANCE_KM = math.sqrt(EARTH_STATE_WRT_SUN[0] ** 2 +
                                  EARTH_STATE_WRT_SUN[1] ** 2 +
                                  EARTH_STATE_WRT_SUN[2] ** 2)

# now we'll use the SPICE convert function to convert from km to AU
EARTH_SUN_DISTANCE_AU = spiceypy.convrt(EARTH_SUN_DISTANCE_KM, "km", "AU")

# this is near one!
EARTH_SUN_DISTANCE_AU
```

[5]:
```
1.0135068265992886
```

Last, to check our velocity vector, let's use a formula.
The theoretical expectation of Earth's orbital velocity around the Sun can be approximated by
$V_{orb} \approx \sqrt{\frac{GM}{r}}$,
where G = gravitational constant, M = mass of Sun, r = distance betw. earth and sun

[8]:
```python
# we'll need one last kernel, which contains the G * M values
# for a collection of objects

# here we deviate from the tutorial because the suggested kernel doesn't work
# instead we use a different kernel that contains body GMs
spiceypy.furnsh("../kernels/pck/de-403-masses.tpc")
```

```
# next use the bodvcd command to get the GM for the sun
# again, bodyid = 10, we want the "GM", and maxn sets the number of expected␣
↪return values
# since it returns an array
_, GM_SUN = spiceypy.bodvcd(bodyid = 10, item = "GM", maxn = 1)

# now compute the orbital speed using a lambda fn
V_ORB_FUNC = lambda gm, r: math.sqrt(gm / r)
EARTH_THEORETICAL_ORB_SPEED_WRT_SUN = V_ORB_FUNC(GM_SUN[0],␣
↪EARTH_SUN_DISTANCE_KM)

# we get a value close to 30 km/s as we should!
EARTH_THEORETICAL_ORB_SPEED_WRT_SUN
```

[8]: 29.58555849620625