ASTRODINÁMICA PARA TODOS (1ª PARTE)

PYTHON COMO HERRAMIENTA CLAVE DE APRENDIZAJE EN CIENCIA

—

¿QUIERES DAR TUS PRIMEROS PASOS EN EL MUNDO DE LA PROGRAMACIÓN EN PYTHON A LA VEZ QUE REPASAS LAS LEYES DE KEPLER? DE LA MANO DE POLIASTRO, LA ASTRODINÁMICA ESTÁ A NUESTROS PIES.

CARLOS MORALES SOCORRO, DANIEL MARÍN ARCONES Y MIGUEL LEÓN SANTANA

Python se ha convertido en uno de los lenguajes más usados en la actualidad en el mundo de la astronomía y de las ciencias en general. La educación no es una excepción y, con Poliastro, el Sistema Solar está al alcance de todos, grandes y pequeños.

Sin duda, la astronomía es un atractor universal de motivación para el alumnado. Su estudio en el aula permite que el árido análisis global de funciones, con su dominio, recorrido, máximos y mínimos, periodicidad, etc., o la estadística, con sus medias, medianas y desviaciones típicas se conviertan en protagonistas de algo tan maravilloso como el descubrimiento de una estrella variable, con su correspondiente registro en la AAVSO; hecho que ya contamos en el número 245, de noviembre de 2019, de esta revista.

En este breve artículo desplazaremos el foco a nuestra vecindad más inmediata, dando nuestros primeros pasos en la simulación del movimiento de cuerpos en el Sistema Solar y explorando la formulación newtoniana de las Leyes de Kepler, hecho que puede hacerse tanto formalmente en las materias de física o matemáticas como, más informalmente, en el club de astronomía. Usaremos para ello la librería **Poliastro** *[docs.poliastro.space/en/stable]*, desarrollada por Juan Luis Cano Rodríguez, a quien agradecemos la inestimable ayuda proporcionada a los autores de este artículo así como al alumnado del Club de Astronomía del IES José Frugoni Pérez, en Telde, Gran Canaria.

Antes de nada, debemos tener claro qué son las Leyes de Kepler, o, mejor dicho, qué nos dicen sobre el Sistema Solar. La primera Ley de Kepler nos dice que todas las órbitas de los cuerpos que giran alrededor del Sol son elípticas (las órbitas parabólicas o hiperbólicas, por definición, no son de cuerpos que viajen alrededor del Sol). Las elipses tienen dos focos y el Sol está siempre en uno de ellos. Cuando los focos de una elipse están muy juntos, la órbita se puede considerar circular. La mayoría de los planetas tienen órbitas que se pueden considerar circulares, con la excepción de Mercurio y Marte. Por contra, multitud de cuerpos menores –asteroides y cometas– tienen órbitas claramente elípticas. La segunda Ley nos indica que un cuerpo se mueve en su órbita más rápidamente cuando pasa cerca del punto más cercano al Sol —perihelio— que cuando está cerca del más lejano –afelio–. La tercera Ley nos permite calcular la velocidad a la que se desplaza un cuerpo en su órbita. Si suponemos órbitas circulares, algo que hemos visto que es factible en el caso de los planetas, la tercera Ley de Kepler nos dice que cuanto más lejos esté un planeta del Sol, más despacio se moverá en su órbita. Por supuesto, todas las leyes de Kepler se derivan directamente de la Ley de la Gravitación Universal de Newton, pero siguen siendo una buena introducción para entender los fundamentos de la mecánica celeste.

1. JUGANDO CON MARTE

Como primer paso, instala Jupyter *[jupyter.org/install]* y abróchate el cinturón. ¡Comienza nuestra aventura! Crea una celda con este código de carga de librerías y teclea *shift-Enter* para ejecutarla y desplegar la siguiente celda:

```
In [2]: from astropy import units as u
        from astropy import time
        from astropy.time import Time
        from poliastro.bodies import Mars, Sun
        from poliastro.twobody import Orbit
        from poliastro.plotting import StaticOrbitPlotter
        from poliastro.util import norm
```

FIGURA 1A. Carga de librerías. (Todas las imágenes son cortesía de Carlos Morales)

Crea a continuación la órbita de Marte según los parámetros clásicos: Semieje mayor (a), Excentricidad (ecc), Inclinación de la órbita (inc), Longitud del nodo ascendente (raan), Argumento del perihelio (argp) y Anomalía verdadera (nu):

```
In [4]: # Extraído de https://docs.poliastro.space/en/stable/user_guide.html

        a = 1.523679 * u.AU
        ecc = 0.093315 * u.one
        inc = 1.85 * u.deg
        raan = 49.562 * u.deg
        argp = 286.537 * u.deg
        nu = 23.33 * u.deg

        mars = Orbit.from_classical(Sun, a, ecc, inc, raan, argp, nu)
```

FIGURA 1B. Creación de la órbita de Marte a partir de parámetros orbitales clásicos.

¡Ya tenemos la órbita de Marte y su posición en un momento concreto o efeméride! ¿Cuál es esa fecha?:

```
In [20]: mars.plot()
         mars.epoch.iso
Out[20]: '2000-01-01 12:00'
```

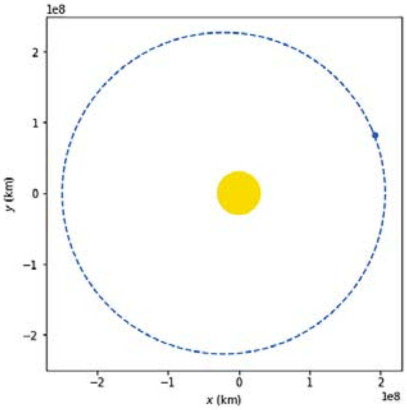

FIGURA 1C. Obtención de la efeméride de la órbita de Marte.

¿Qué forma geométrica tiene la órbita? ¿Encaja con el valor de la excentricidad? Ejecuta en una nueva celda la instrucción *mars.ecc* para averiguarlo y compáralo con la siguiente tabla:

Excentricidad	Tipo de órbita
e = 0	Circunferencia
0 < e < 1	Elipse
e = 1	Parábola
e > 1	Hipérbola

Estimemos ahora el periodo de Marte a partir de la versión simplificada de la formulación newtoniana de la Tercera Ley de Kepler:

$\frac{P^2}{a^3} = \frac{4\pi^2}{GM_S} \rightarrow P = \sqrt{\frac{4\pi^2 a^3}{GM_S}}$, que sustituyendo valores: P ≈ 689 días.

Sin embargo, al usar Poliastro, se obtiene un valor diferente. ¿Cómo explicas esa diferencia?:

```
In [9]: mars.period.to(u.day)
Out[9]: 686.97138 d
```

FIGURA 1D. Obtención del periodo orbital de Marte.

Averiguaremos ahora la posición de Marte cien días después y dibujaremos su posición en la misma gráfica:

```
In [11]: mars100d = mars.propagate(100 * u.day)
         plotter = StaticOrbitPlotter()
         plotter.plot(mars, label="Marte")
         plotter.plot(mars100d, label="Marte, 100 días después")
Out[11]: [<matplotlib.lines.Line2D at 0x7f0f06bf3b50>,
          <matplotlib.lines.Line2D at 0x7f0f06bf3190>]
```

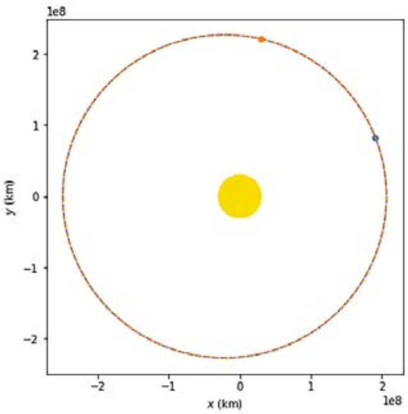

FIGURA 1E. Propagación de la órbita de Marte en cien días y dibujo de la órbita.

¿Cómo vas hasta ahora? ¿Qué debería ocurrir si propagásemos la órbita de Marte (mars) 699 días y representásemos su posición en el diagrama anterior? Hazlo.

2. SOBRE ASTEROIDES Y COMETAS

Los asteroides y cometas son cuerpos pequeños, restos de la formación del Sistema Solar. Tradicionalmente, se consideraba cometas a aquellos cuerpos con órbitas muy elípticas (muy excéntricas) con una gran cantidad de hielo y otros volátiles en su composición que, al pasar cerca del Sol, desarrollaban llamativas colas de gas y polvo. Por contra, los asteroides son cuerpos rocosos, la mayoría de ellos con órbitas menos excéntricas que las de los cometas que están situadas entre Marte y Júpiter. En realidad, hoy en día no existe una línea divisoria definida entre asteroides y cometas. Simplemente, hay todo un espectro de cuerpos según su contenido en hielo y otros volátiles. Actualmente conocemos objetos con órbitas muy excéntricas que no son cometas y, al contrario, otros situados en el cinturón principal de asteroides que desarrollan colas ocasionalmente.

Podemos dividir a los asteroides según su composición o por su tipo de órbita. De acuerdo con la composición, los principales tipos de asteroides se denominan C, S y M. Los de tipo C o «condritas» son los más comunes y están formados principalmente por minerales rocosos, sustancias orgánicas y volátiles. Los de tipo S son más «rocosos» y tienen pocas o ninguna sustancia orgánica y hielo. Por último, los de tipo M son principalmente metálicos. En cuanto a los tipos de asteroides según su órbita, existen muchos grupos, pero el más numeroso es el que se halla en el cinturón principal de asteroides, entre Marte y Júpiter, seguido de los Troyanos, un amplio grupo de asteroides que se divide en dos grandes zonas situadas 60° por delante y por detrás de Júpiter en su órbita. Mención aparte merecen los asteroides cercanos a la Tierra o NEA *(Near Earth Asteroids)*, que son aquellos que se acercan a menos de 1,3 unidades astronómicas de nuestro planeta. Dentro de este grupo destacan los asteroides potencialmente peligrosos o PHA *(Potential Hazardous Asteroids)*, que, como su nombre indica, podrían chocar con la Tierra en las próximas décadas o siglos.

Los cometas de periodo más largo vienen de una reserva hipotética de cuerpos helados denominada Nube de Oort, localizada a casi un año luz del Sol. Como está tan lejos, al llegar cerca de nuestra estrella estos cometas tienen órbitas tan excéntricas que prácticamente son parábolas. Recientemente hemos descubierto dos objetos que han pasado por las proximidades del Sol procedentes del espacio interestelar. Su velocidad era tan alta que sus órbitas eran hiperbólicas y, por tanto, nunca más volverán a pasar por el Sistema Solar.

¿Podremos aproximarnos a estos objetos con Poliastro? Carga ahora un objeto menor, por ejemplo, el asteroide 433 Eros (A898 PA), del tipo Amor, con SPK-ID: 2000433. En esta ocasión lo seleccionaremos directamente desde la base de datos JPL Small-Body Database *[ssd.jpl.nasa.gov/sbdb.cgi]*. Pero antes, averigua qué es un asteroide tipo Amor. Carga también el asteroide Apophis. ¿Deberíamos estar asustados?:

```
In [1]:  from astropy.utils.data import conf
         conf.remote_timeout = 10000

In [2]:  from astropy import units as u
         from astropy import time
         from astropy.time import Time
         from astropy.coordinates import solar_system_ephemeris
         solar_system_ephemeris.set("jpl")

         from poliastro.bodies import Earth, Mars
         from poliastro.twobody import Orbit
         from poliastro.plotting import StaticOrbitPlotter
         from poliastro.util import norm
```

FIGURA 2A. Carga de librerías y ajustes de descarga de datos de JPL Small-Body Database.

```
In [26]: eros = Orbit.from_sbdb("433 Eros")
         #eros = Orbit.from_sbdb("2000433")
         apophis = Orbit.from_sbdb("99942 Apophis")
         #apophis = Orbit.from_sbdb("2099942")
         oumuamua = Orbit.from_sbdb("'Oumuamua")
```

FIGURA 2B. Carga de asteroides desde la base de datos sdbd, por nombre o por SPK-ID.

```
In [27]: epoch = Time("2017-11-21 12:00:00", scale="tdb")
         earth = Orbit.from_body_ephem(Earth, epoch)
         mars = Orbit.from_body_ephem(Mars, epoch)
         apophis = apophis.propagate(epoch).to_icrs()
         eros = eros.propagate(epoch).to_icrs()
         oumuamua = oumuamua.propagate(epoch).to_icrs()

In [28]: apophis, eros, earth, mars, oumuamua

Out[28]: (1 x 1 AU x 20.4 deg (ICRS) orbit around Sun (☉) at epoch 2458079.0 (TDB),
          1 x 2 AU x 30.8 deg (ICRS) orbit around Sun (☉) at epoch 2458079.0 (TDB),
          1 x 1 AU x 23.4 deg (ICRS) orbit around Sun (☉) at epoch 2017-11-21 12:00:00.000 (TDB),
          1 x 2 AU x 24.7 deg (ICRS) orbit around Sun (☉) at epoch 2017-11-21 12:00:00.000 (TDB),
          38795628 x -418529081 km x 143.5 deg (ICRS) orbit around Sun (☉) at epoch 2458079.0 (TDB))
```

FIGURA 2C. Propagamos las órbitas a la fecha deseada y aseguramos sistema de referencia común (ICRS), baricentro del Sistema Solar.

```
In [29]: plotter = StaticOrbitPlotter()
         plotter.plot(apophis,label="(99942) Apophis")
         plotter.plot(eros,label="(433) Eros")
         plotter.plot(oumuamua, label="'Oumuamua")
         plotter.plot(earth,label="La Tierra")
         plotter.plot(mars,label="Marte")

Out[29]: [<matplotlib.lines.Line2D at 0x7f2d0ba27910>,
          <matplotlib.lines.Line2D at 0x7f2d0ba27f10>]
```

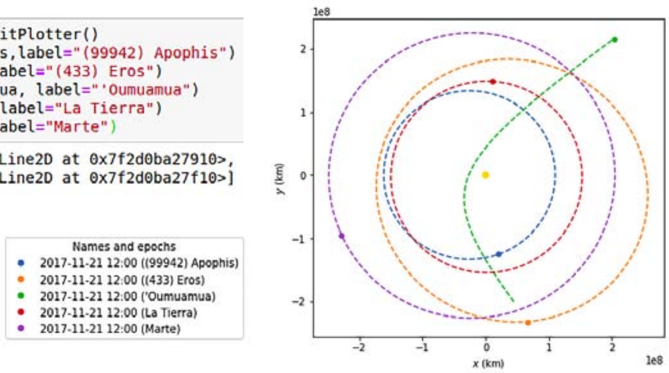

FIGURA 2D. Dibujamos las órbitas de Marte, la Tierra, Eros, Apophis y ´Oumuamua.

```
In [25]: norm(eros.r - earth.r) - Earth.R
Out[25]: 31361585 km
```

FIGURA 2E. Cálculo de la distancia de Eros a la superficie terrestre *[ver ssd.jpl.nasa.gov/sbdb.cgi]*.

De igual forma, podemos visualizar la Segunda Ley de Kepler con el cometa 1P/Halley:

```
In [46]: from poliastro.plotting import OrbitPlotter2D
epoch = Time("2019-11-10 12:00:00", scale="tdb")
halley = Orbit.from_sbdb("1000036")
halley = halley.propagate(epoch)
halley_10 = halley.propagate(10*u.year)
halley_20 = halley.propagate(20*u.year)
halley_30 = halley.propagate(30*u.year)
plotter = OrbitPlotter2D()
plotter.plot(halley,label="1P/Halley")
plotter.plot(halley_10,label="1P/Halley")
plotter.plot(halley_20,label="1P/Halley")
plotter.plot(halley_30,label="1P/Halley")
plotter.plot(Orbit.from_body_ephem(Earth,epoch), label="La Tierra")
```

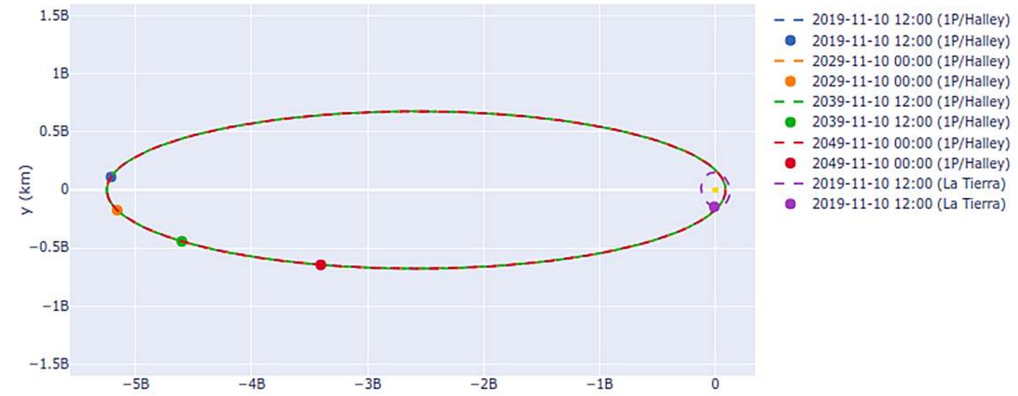

FIGURA 2F. Posición de 1P/Halley en cuatro momentos distintos.

En la segunda parte de este artículo el mes próximo avanzaremos en el uso de Poliastro, repasando conceptos como la velocidad de escape y las famosas órbitas de transferencia de Hohmann tan frecuentemente nombradas y que, como el lector podrá descubrir, no tienen ningún secreto con Poliastro. (**A**)

Carlos Morales Socorro y **Miguel León Santana** son profesores de matemáticas en el IES José Frugoni Pérez. **Daniel Marín Arcones** es astrofísico, uno de los mayores expertos en astronáutica de nuestro país. Los tres son miembros de la Agrupación Astronómica de Gran Canaria.