Métodos Numéricos para la Ciencia e Ingeniería
FI3104-1

Tarea 4

Maximiliano Dirk Vega Aguilera
18.451.231-9

1 Introducción

La tarea 4 consistió en estudiar el tutorial de Software Carpentry para mejorar el diseño del software, y en resolver de forma numérica la ecuación de movimiento de un planeta con una órbita cercana al Sol, a partir del potencial gravitatorio corregido por la relatividad general y comparando con el potencial gravitatorio de Newton clásico.

2 Software Carpentry

La primera parte consistió en estudiar unos vídeo-tutoriales desarrollados por la compañía Software Carpentry, cuyo objetivo es ayudar a científicos a mejorar sus habilidades de programación, enfocándose en el diseño del software, de modo que este sea más eficiente.

Para esto, se contestaron las preguntas de a continuación:

2.1 Describa la idea de escribir el main driver primero y llenar los huecos luego. ¿Por qué es buena idea?

La idea de escribir el main driver primero es crear una estructura que defina lo que se espera que haga el programa, de manera que se vaya completando con las funciones que se necesiten mientras se va avanzando. Esto es una buena idea ya que permite mantener un orden y da una idea de las tareas faltantes.

2.2 ¿Cuál es la idea detrás de la función mark_filled? ¿Por qué es buena idea crearla en vez del código original al que reemplaza?

La idea detrás de la función mark_filled es hacer que el código este haciendo lo que realmente debe estar haciendo, permitiendo identificar los errores de forma más fácil en caso de que exista alguno.

2.3 ¿Qué es refactoring?

Refactoring es el proceso de cambiar la estructura del código sin cambiar su comportamiento, de forma que quede más ordenado o más eficiente.

2.4 ¿Por qué es importante implementar tests que sean sencillos de escribir? ¿Cuál es la estrategia usada en el tutorial?

Es importante implementar tests que sean sencillos de escribir porque así se evita introducir un error en la implementación del test y hace que otra persona pueda entenderlo con más facilidad. La estrategia usada en el tutorial es utilizar fixtures y luego comparar con los test.

2.5 El tutorial habla de dos grandes ideas para optimizar programas, ¿cuáles son esas ideas? Descríbalas.

La primera idea para optimizar es reducir el tiempo que se tarda en ejecutar a cambio de espacio en la memoria. Esta idea consiste en guardar en la memoria resultados de procesos que se van repitiendo, de modo que no sea necesario volver a calcularlos.

La segunda idea para optimizar es mejorar el código a cambio de tiempo de programación. Esta idea consiste en gastar más tiempo programando un código más eficiente, de modo que este corra más rápido.

2.6 ¿Qué es lazy evaluation?

Lazy evaluation es no evaluar o calcular algo a no ser que sea necesario, ahorrando así tiempo de ejecución.

2.7 Describa la other moral del tutorial (es una de las más importantes a la hora de escribir buen código).

Para escribir un programa que corra rápido, se debe partir escribiendo un programa simple e ir modificándolo de a poco, testeándolo con cada modificación.

3 Órbitas Planetarias Cercanas al Sol

La segunda parte consistió en encontrar la ecuación de movimiento de un planeta que orbita cerca del Sol bajo la acción del potencial gravitacional corregido por relatividad general, donde si $\alpha = 0$ se recupera el potencial gravitacional de Newton clásico.

$$U(r) = -\frac{GMm}{r} + \alpha\frac{GMm}{r^2} \tag{1}$$

La órbita de un planeta bajo este potencial sigue siendo plana, pero el perihelio gira alrededor del Sol.

Para calcular la ecuación de movimiento, se utilizó el formalismo lagrangeano para las coordenadas x e y. Donde las energías cinética y potencial toman la forma:

$$T = \frac{1}{2}m(\dot{x}^2 + \dot{y}^2) \tag{2}$$

$$V = -\frac{GMm}{\sqrt{x^2+y^2}} + \alpha\frac{GMm}{(x^2+y^2)^2} \tag{3}$$

Obteniéndose,

$$\ddot{x} = -\frac{GMx}{(x^2+y^2)^{\frac{3}{2}}} + 2\alpha\frac{GMx}{(x^2+y^2)^2} \tag{4}$$

$$\ddot{y} = -\frac{GMy}{(x^2+y^2)^{\frac{3}{2}}} + 2\alpha\frac{GMy}{(x^2+y^2)^2} \tag{5}$$

Estas ecuaciones de movimiento se resuelven numéricamente para obtener la orbita del planeta.

Para el desarrollo numérico, se utilizó python junto a sus bibliotecas numpy y matplotlib. Lo primero que se hizo fue crear una clase planeta, completando el archivo planetas.py. En esta clase se define el objeto planeta dadas ciertas condiciones iniciales de posición y velocidad; además se define el vector de la ecuación de movimiento como ecuaciones de primer orden para el desarrollo de los métodos numéricos, a partir de las ecuaciones (4) y (5). También se definen las funciones que hacen avanzar los métodos de Euler, Runge-Kutta 4 y Verlet en un paso, y una función que entrega la energía total del sistema en las condiciones en las que se encuentra el planeta.

Para simplificar la implementación, se consideró $G = M = m = 1$ durante todo el desarrollo del problema.

Una vez creada la clase planeta, se calculó la órbita y la energía total del sistema para el potencial clásico ($\alpha = 0$) utilizando los 3 métodos anteriormente mencionados. Obteniéndose los gráficos de las figuras 1 al 6.

Luego, se calculó considerando $\alpha = 10^{-2.321}$, y se obtuvieron los gráficos de las figuras 7 al 12.

Cabe mencionar que se varió la velocidad inicial en y y el numero total de pasos para tratar de cumplir con el número de orbitas pedido.

4 Conclusión

Respecto a la primera parte de la tarea 4, el tutorial de Software Carpentry entrega buenas estrategias para la realización de un mejor código, de forma que quede más ordenado, permitiendo comprender cada parte del código y/o encontrar errores de manera más sencilla; y a la vez dando la posibilidad de hacerlo más eficiente.

Respecto a la segunda parte, a partir de los gráficos obtenidos, se ve que el método de euler genera error sistemáticamente, lo que concuerda con la inestabilidad del método para largos intervalos. El metodo de Runge-Kutta 4 presenta una mejor estabilidad respecto al metodo de Verlet, lo que nos indica que probablemente que el método de Verlet fue mal implementado.

5 Figuras

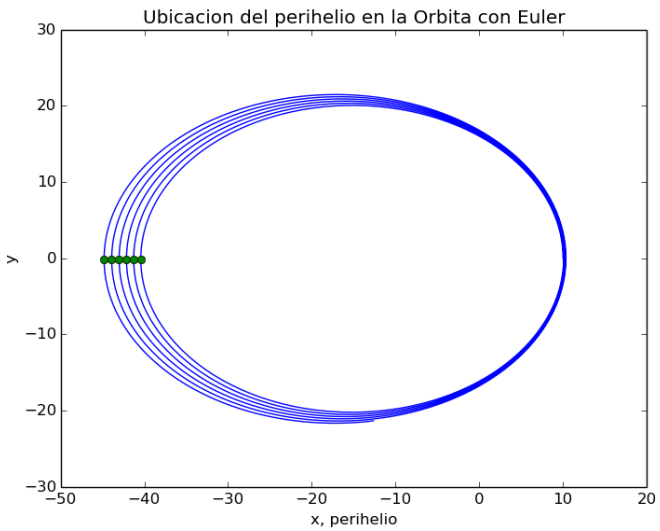

Figure 1: Sin α

References

[1] Nicolás Troncoso, compañero.

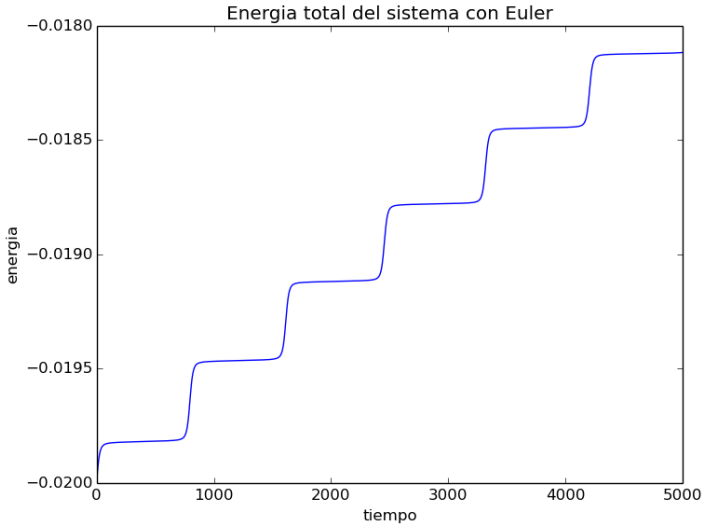

Figure 2: Sin α

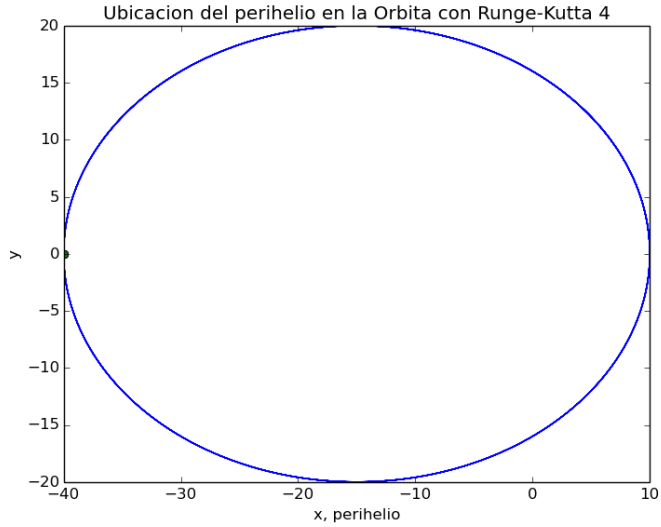

Figure 3: Sin α

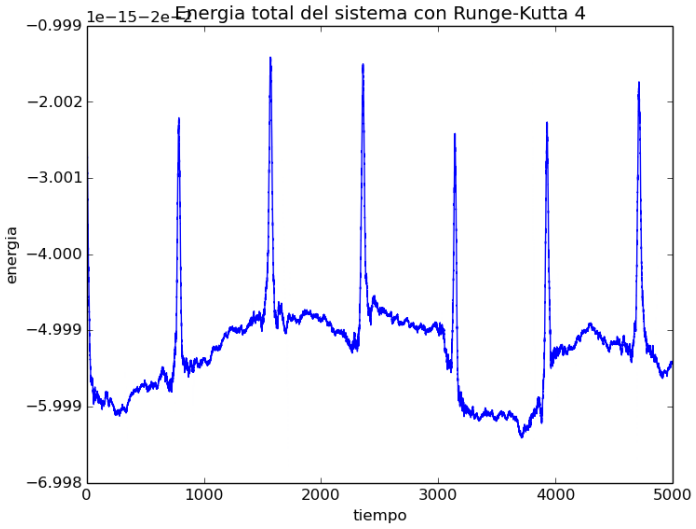

Figure 4: Sin α

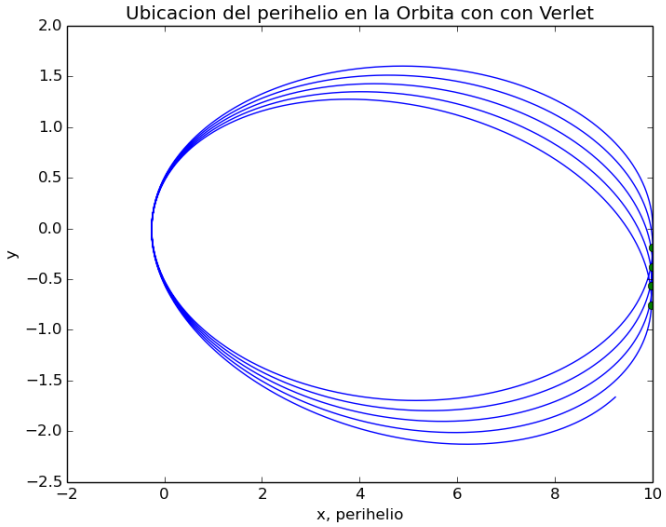

Figure 5: Sin α

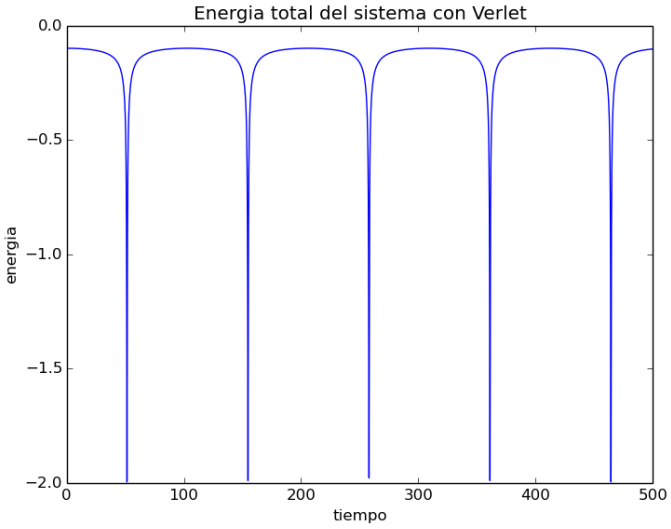

Figure 6: Sin α

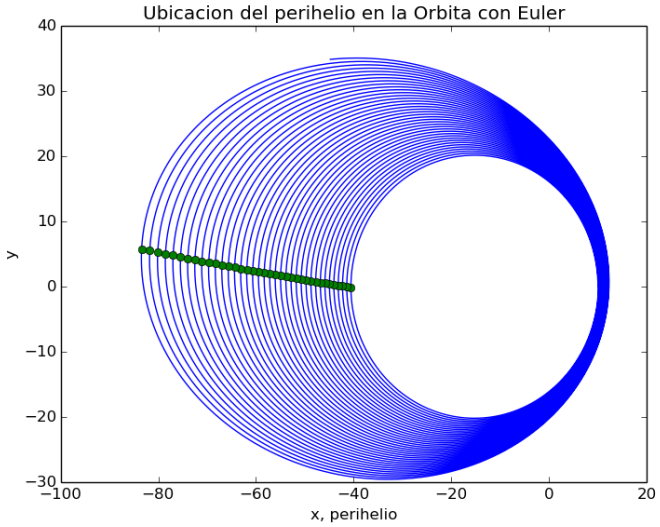

Figure 7: Con α

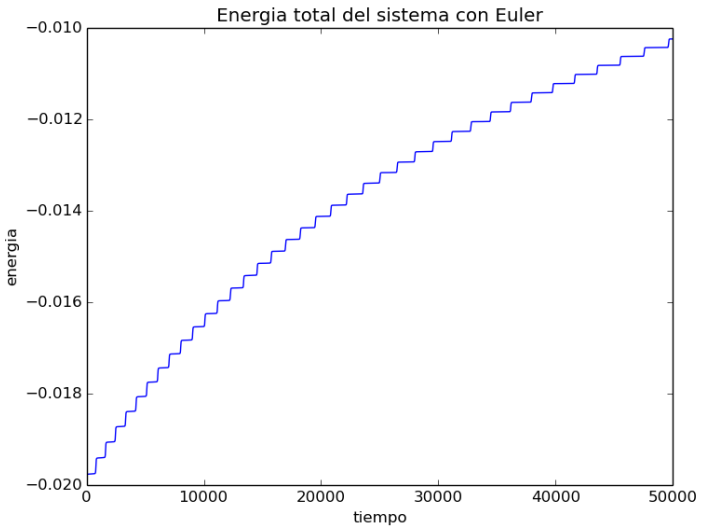

Figure 8: Con α

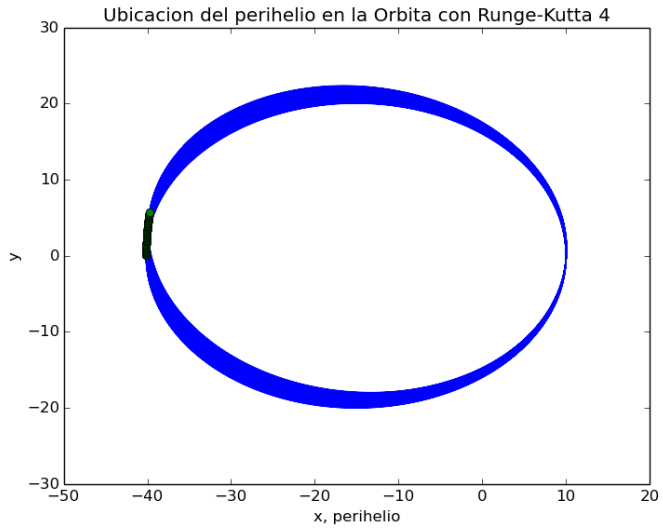

Figure 9: Con α

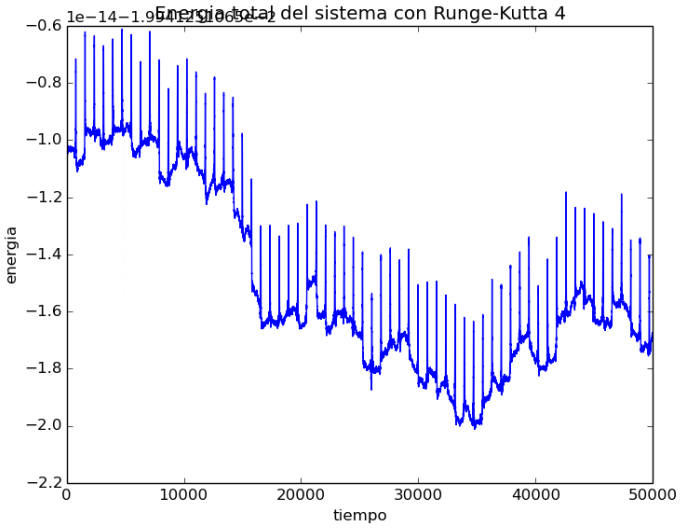

Figure 10: Con α

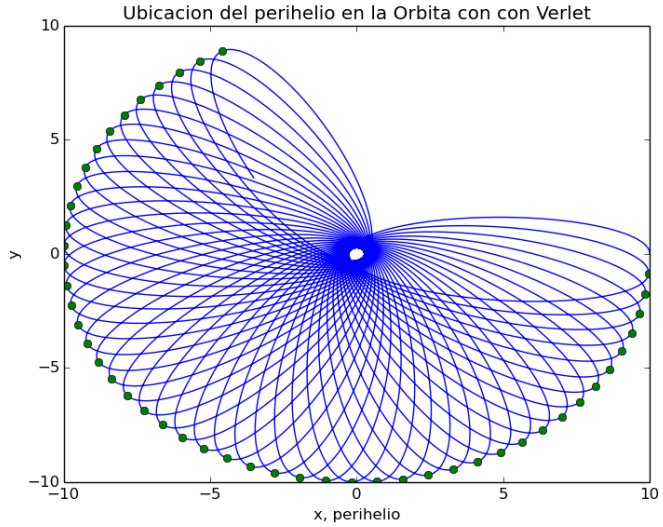

Figure 11: Con α

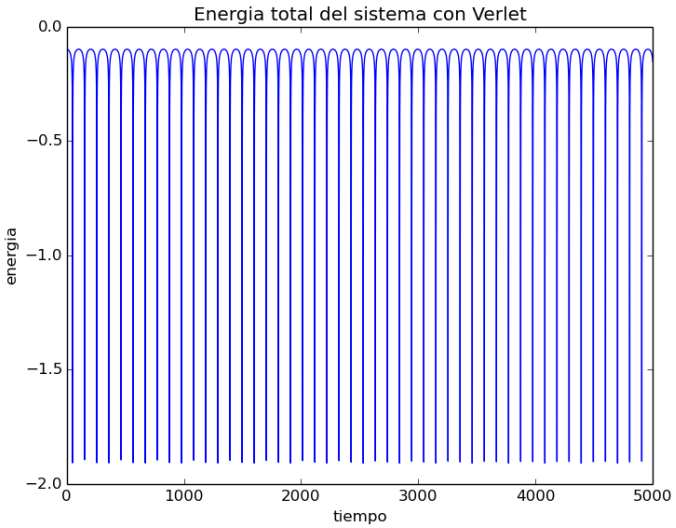

Figure 12: Con α