Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio

Máster Universitario en Sistemas Espaciales

Milestone 1: Prototipos de integración de órbitas sin funciones

Ampliación de Matemáticas I:
Cálculo numérico

24 de septiembre de 2022

Autor:

- Daniel Cerón Granda

ÍNDICE

PÁGINA

1.Introducción

El objetivo de este trabajo es la realización de un código que sea capaz de simular de manera básica el recorrido de una órbita. Para ello se aplicarán dos esquemas numéricos implícitos (Euler y Runge-Kutta de orden 4) a un problema de Cauchy o de valor inicial.

Un problema de Cauchy es una ecuación de la forma:

$$\frac{dU}{dt} = F(U,t)$$

Ecuación 1: Expresión general del problema de Cauchy

Además, este problema debe tener una condición inicial en t = 0 tal que: U(0) = U_0. Tanto F como U son vectores (o escalares, si fuera el caso) formados por números reales en sus componentes.

La expresión de una órbita circular es la siguiente:

$$\ddot{\vec{r}} = \frac{-\vec{r}}{|\vec{r}^3|}$$

Ecuación 2: Expresión de la ecuación de una órbita circular

Que a simple vista parece no ajustarse a la expresión del problema de Cauchy. Para transformar esta expresión, expresamos el vector de posición en función de sus componentes (x, y) y también incluimos en la U del problema de Cauchy a las derivadas de las componentes del vector de posición, de tal forma que el vector U queda:

$$U = \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix}$$

Ecuación 3: Expresión del vector U del problema de valor inical

Y el problema de valor inicial queda:

$$\frac{dU}{dt} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \ddot{x} \\ \ddot{y} \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dfrac{-x}{(x^2+y^2)^{3/2}} \\ \dfrac{-y}{(x^2+y^2)^{3/2}} \end{pmatrix} = F(U,t)$$

Ecuación 4: Expresión del problema de valor inicial en función de x e y

Y expresándolo en función de las coordenadas de U:

$$\frac{dU}{dt} = \begin{pmatrix} U(3) \\ U(4) \\ \frac{-U(1)}{(U(1)^2 + U(2)^2)^{3/2}} \\ \frac{-U(2)}{(U(1)^2 + U(2)^2)^{3/2}} \end{pmatrix} = F(U,t)$$

Ecuación 5: Expresión del problema de valor inicial en función de las componentes de U

Que es la expresión de F que se utilizará en este ejercicio. Ahora el problema reside en cómo obtener el valor de U. De forma matemática, despejando de la ecuación del problema de Cauchy:

$$dU = F(U,t)dt \rightarrow \int_{U(t_1)}^{U(t_2)} dU = \int_{t_1}^{t_2} F(U,t)dt$$

Ecuación 6: Integración del problema de Cauchy

La integral de dU es directa, y la integral de F(U, t) se puede obtener, con mayor o menor dificultad a mano, conocida F(U). Sin embargo, un programa no tiene capacidad de realizar la integración directa de una función arbitraria. Conocida F, se podría obtener su integral a mano e introducirla en el ordenador, pero eso llevaría un tiempo importante y sólo sería válido para una F determinada, variar la función llevaría a tener que repetir todo el proceso. Por tanto, lo que se hace es realizar una aproximación de la integral de la función con esquemas numéricos, que discretizan la función a estudiar y calculan la integral de forma aproximada, a partir de los valores de las funciones para un intervalo de tiempo formado por n (siendo n un número elevado, generalmente) puntos equiespaciados entre ellos por un Δt pequeño (normalmente, es un valor de uno o varios órdenes de magnitud inferiores al segundo).

Si el esquema numérico es preciso, el error cometido es muy bajo. Se denomina orden de un esquema temporal al orden de magnitud del error cometido. Si un esquema numérico es de orden 1, su error local de truncamiento (error cometido en un paso del esquema), su error es del orden del Δt utilizado, si es de orden 2, de ese Δt elevado al cuadrado… Cuanto mayor sea el orden de un esquema, más preciso será respecto a la solución real.

2.Integración de órbitas con el método de Euler

El esquema de Euler es un método sencillo de aplicar. Realiza la siguiente aproximación para una integral:

$$\int_{U^n}^{U^{n+1}} dU = \int_{t_n}^{t_{n+1}} F(U,t)dt \;\rightarrow\; U^{n+1} - U^n \cong F(U^n, t_n) * (t_{n+1} - t_n)$$

Ecuación 7: Aproximación de una integral por el método de Euler

Siendo n el paso de tiempo que se esté analizando. Como es lógico, el error cometido será totalmente dependiente del intervalo entre los dos t. Se elige un intervalo pequeño de tiempo que se denomina Δt que será constante para cada paso de tiempo. La expresión de U en el paso de tiempo siguiente al analizado será:

$$U^{n+1} = U^n + F(U^n, t_n) * \Delta t$$

Ecuación 8: Expresión de U en el paso siguiente en función del valor del paso anterior

Se puede observar que es un método explícito, obteniendo el valor de U^{n+1} sólo en función de variables definidas en el paso anterior n. Es un esquema de orden 1, el error cometido será del orden de Δt.

Los valores elegidos para el estudio de este problema son n = 200 y Δt = 0,1 segundos. El código de este apartado se muestra en la imagen siguiente:

```
#APARTADO 1: EULER

#Definición de condiciones iniciales
n = 200 # Número de particiones
deltat = 0.1 #Intervalo de tiempo entre iteraciones
U0 = array([1,0,0,1]) #Condiciones iniciales
matrizU = array(zeros([n,4])) #Matriz que contendrá en sus columnas los valores de las variables, cada fila es un paso de tiempo
F = array([0,1,-1,0])#Valor de F en el instante inicial
U = zeros(4)
x = zeros(n)
y = zeros(n)

#Aplicación del método de Euler
for i in range(n):
    for j in range(4):
        U[j] = U0[j]+ deltat*F[j] #Método de Euler
        continue
    matrizU[i,:] = U #Almacena en una columna de la matriz el valor de las variables para el paso de tiempo analizado
    U0 = U #Cambio las condiciones iniciales del paso
    d = ((U[0])**2+(U[1])**2)**1.5
    F = array([U[2], U[3], -U[0]/d, -U[1]/d]) #Obtengo mi nuevo valor de F(U,t)
    continue

#Las dos primeras columnas de la matriz son x e y de la órbita
x = matrizU[:,0]
y = matrizU[:,1]

#Representación gráfica
plt.plot(x,y)
plt.show()
```

Figura 1: Código de la modelización de una órbita por el método de Euler

Como aspectos destacables del mismo, se dan las condiciones iniciales para U y F. Se supone que la órbita empieza en (1,0) con velocidad (0,1). Para almacenar la información, se crea una matriz que va almacenando los valores de posición y velocidad para todos los pasos de tiempo de la simulación.

Para cada paso, se calcula el valor de F a partir de los valores de U para ese paso de tiempo. Con F, se calcula el valor del paso siguiente para U y así sucesivamente. También se define un parámetro d que es el módulo del vector posición elevado al cubo, para facilitar la definición de F.

Finalmente, para la representación se extraen las dos primeras columnas de la matriz, que representan las coordenadas de posición x e y para todos los pasos de tiempo. La representación es la siguiente:

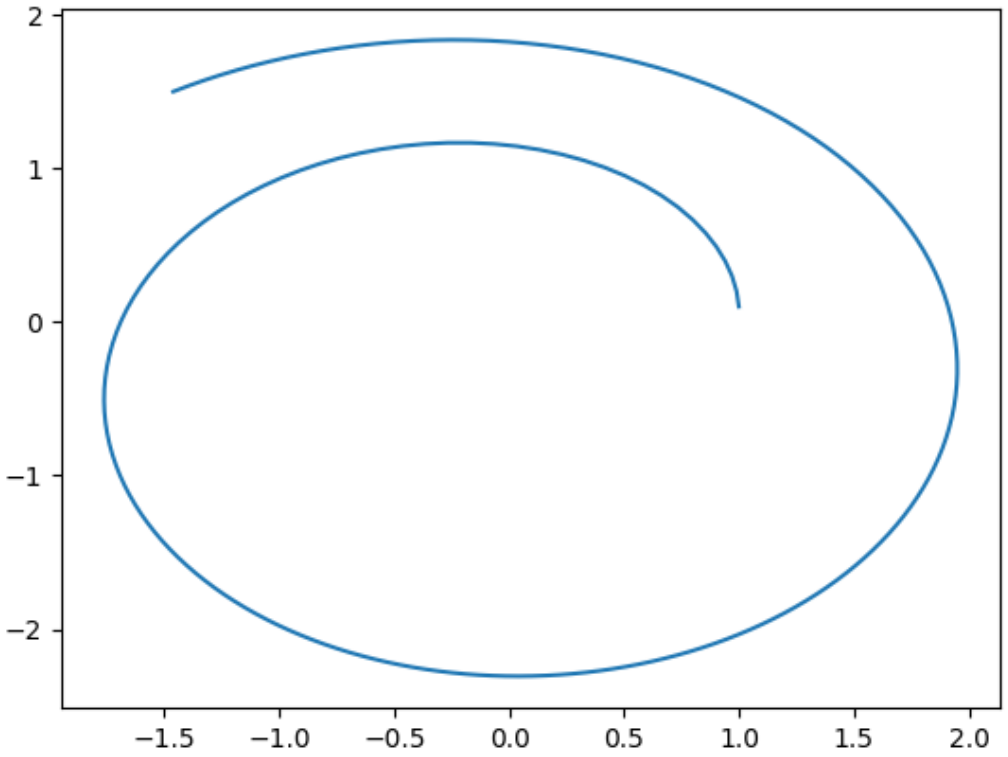

Figura 2: Representación gráfica por el método de Euler

Se puede observar una desviación muy rápida y de gran magnitud respecto a la órbita real al integrar por este método con estos valores. Esto es lógico, puesto que es un esquema de orden 1. Una solución posible sería disminuir el Δt y aumentar el número de iteraciones, lo que debería disminuir el error, ya que éste es proporcional al Δt.

3.Integración de órbitas con el método RK4

Los métodos de Runge-Kutta (RK) son un conjunto de métodos implícitos y explícitos para aproximar problemas de valor inicial. Se determinan según su orden. La expresión general de un método RK de orden s es la siguiente:

$$U^{n+1} = U^n + \Delta t * \sum_{i=1}^{s} b_i * k_i$$

Ecuación 9: Expresión general de un método RK de orden s

Donde los coeficientes k son términos de aproximación intermedios evaluados en F de manera local:

$$k_i = F(U^n + \Delta t * \sum_{j=1}^{s} a_{ij} * k_j$$

Ecuación 10: Expresión general de los coeficientes k en un método RK de orden s

Siendo los coeficientes a, b y c coeficientes propios de cada esquema. Los coeficientes a serán los que determinarán si el método es implícito o explícito. Si la matriz formada por estos coeficientes es triangular inferior con todos los elementos de la diagonal principal iguales a 0, el método será explícito. Si la matriz no es así, el método será explícito.

Para este trabajo, se utilizará un Runge-Kutta de orden 4. La expresión de U en este método es la siguiente:

$$U^{n+1} = U^n + \Delta t * \frac{k_1 + 2 * k_2 + 2 * k_3 + k_4}{6}$$

Ecuación 11: Expresión de U para el método RK4

Donde los coeficientes k son los siguientes:

$$k_1 = F(U(t_n), t_n)$$

$$k_2 = F\left(U\left(t_n + \Delta t * \frac{k_1}{2}\right), t_n + \frac{\Delta t}{2}\right)$$

$$k_3 = F\left(U\left(t_n + \Delta t * \frac{k_2}{2}\right), t_n + \frac{\Delta t}{2}\right)$$

$$k_4 = F(U(t_n + \Delta t * k_3), t_n + \Delta t)$$

Ecuaciones 12-15: Expresiones de los coeficientes k para el método RK4

Es un método explícito, cada coeficiente k se puede obtener en función de los coeficientes anteriores y el valor de U se puede obtener en función únicamente de los coeficientes del paso anterior. Es de orden 4, por lo que el error escala con el Δt elevado a la cuarta. Por tanto, debería dar una órbita mucho más precisa que el Euler

El código utilizado en este método es el siguiente:

```
43
44  #APARTADO 2: RK4
45
46  #Definición de condiciones iniciales
47  n = 200 # Número de particiones
48  deltat = 0.1 #Intervalo de tiempo entre iteraciones
49  U0 = array([1,0,0,1]) #Condiciones iniciales
50  matrizU = array(zeros([n,4])) #Matriz que contendrá en sus columnas los valores de las variables, cada fila es un paso de tiempo
51  F = array([0,1,-1,0])#Valor de F en el instante inicial
52  U = zeros(4)
53  x = zeros(n)
54  y = zeros(n)
55  k1 = zeros(4)
56  k2 = zeros(4)
57  k3 = zeros(4)
58  k4 = zeros(4)
```

Figura 3: Código del método RK4 I

```
60  #Aplicación del método RK4
61  for i in range(n):
62      k1 = F #Primer coeficiente
63      U2 = U0 +  k1*deltat/2
64      d2 = ((U2[0])**2+(U2[1])**2)**1.5
65      k2 = array([U2[2], U2[3], -U2[0]/d2, -U2[1]/d2]) #Segundo coeficiente
66      U3 = U0 +  k2*deltat/2
67      d3 = ((U3[0])**2+(U3[1])**2)**1.5
68      k3 = array([U3[2], U3[3], -U3[0]/d3, -U2[1]/d3]) #Tercer coeficiente
69      U4 = U0 +  k3*deltat
70      d4 = ((U4[0])**2+(U4[1])**2)**1.5
71      k4 = array([U4[2], U4[3], -U4[0]/d4, -U4[1]/d4]) #Cuarto coeficiente
72      U = U0+ deltat*(k1+2*k2+2*k3+k4)/6 #Método RK4
73      matrizU[i,:] = U #Almacena en una columna de la matriz el valor de las variables para el paso de tiempo analizado
74      U0 = U #Cambio las condiciones iniciales del paso
75      d = ((U[0])**2+(U[1])**2)**1.5
76      F = array([U[2], U[3], -U[0]/d, -U[1]/d]) #Obtengo mi nuevo valor de F(U,t)
77      continue
78
79  #Las dos primeras columnas de la matriz son x e y de la órbita
80  x = matrizU[:,0]
81  y = matrizU[:,1]
82
83  #Representación gráfica
84  plt.plot(x,y)
85  plt.show()
86
```

Figura 4: Código del método RK4 II

La estructura es muy similar a la utilizada para el método de Euler. Los valores de n y Δt son los mismos que en el caso anterior, para facilitar la comparación. Se definen los parámetros d y U para cada caso del coeficiente k (excepto k1, que por definición es la F del paso anterior) y posteriormente se define la U y la d del paso completo. Se aplica la misma estructura de la matriz para obtener la x y la y en cada paso.

La representación gráfica del movimiento es la siguiente:

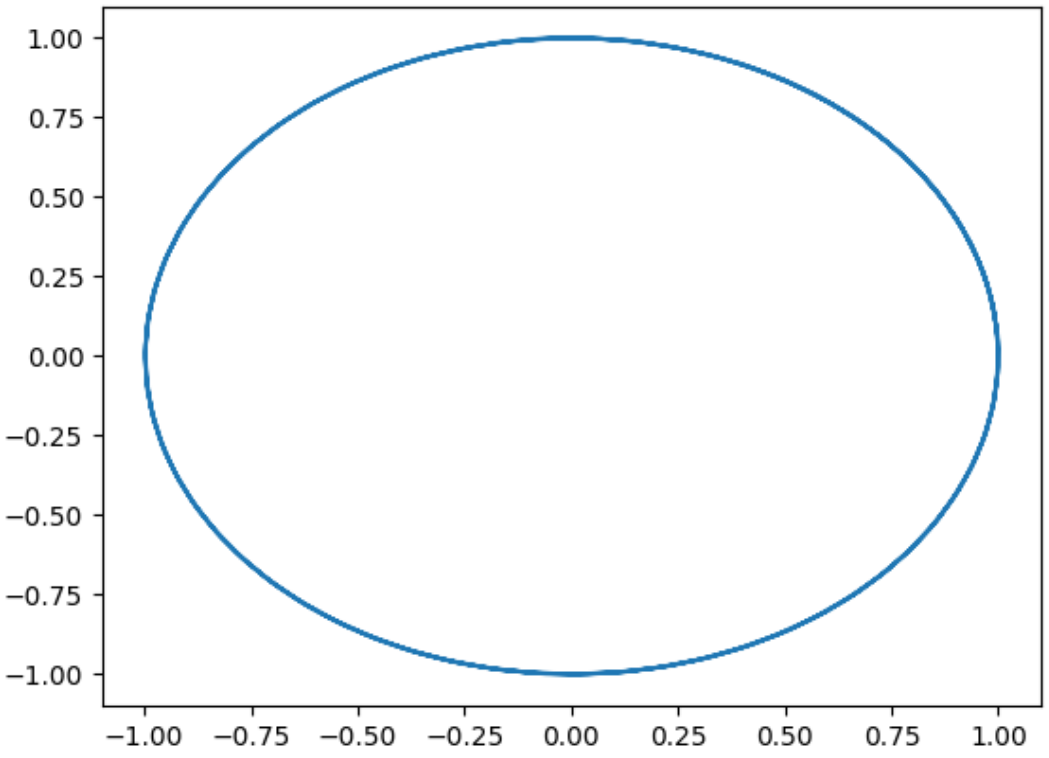

Figura 5: Representación gráfica del método RK4

Se observa una órbita mucho más precisa que en el caso de Euler, con una desviación casi imperceptible. Con un mayor número de iteraciones se podría observar que empieza a desviarse, pero para estos valores es una solución muy cercana a la real.

4.Efecto de cambios en el Δt requerido

a. Método de Euler

Como ya se ha dicho, el error cometido con un esquema depende del Δt elegido. A continuación, se va a analizar el efecto de cambiar este Δt. Para el método de Euler se va a analizar qué ocurre al disminuirlo, ya que el error cometido es muy elevado. Aplicando un código exactamente igual al del apartado 2, pero cambiando el Δt por 0,05 y aumentando el número de iteraciones a 400 (para que el tiempo final sea el mismo), se obtiene la gráfica siguiente:

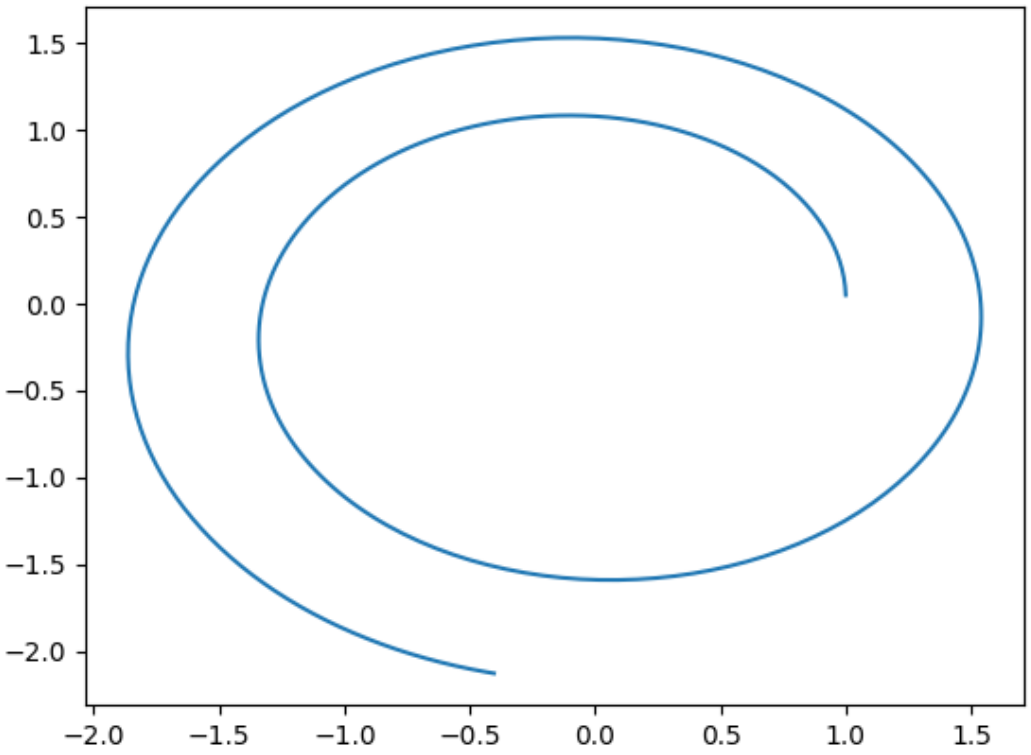

Figura 6: Representación de Euler con un menor Δt

Se observa que la desviación cometida con este método es menor (en la figura 2 la segunda vuelta en su punto de y máximo llega casi a 2, mientras que en esta se queda en 1,5), constatando que, como era de esperar, aumentar el Δt conlleva una reducción del error cometido. No acaba en el mismo punto, pero porque al no desviarse tanto, avanza más en la órbita.

b. Método RK4

Como la precisión del método RK4 es elevada de por sí, disminuyendo el Δt apenas va a proporcionar grandes diferencias a nivel visual. Por tanto, se seguirá el camino contrario, se va a observar qué ocurre al aumentar el Δt. Al ser de orden 4, debería aumentar muy rápidamente el error con el Δt. Aplicando un código exactamente igual al utilizado en el apartado 3, pero cambiando la n por 50 y el Δt por 0,04; se obtiene la siguiente representación:

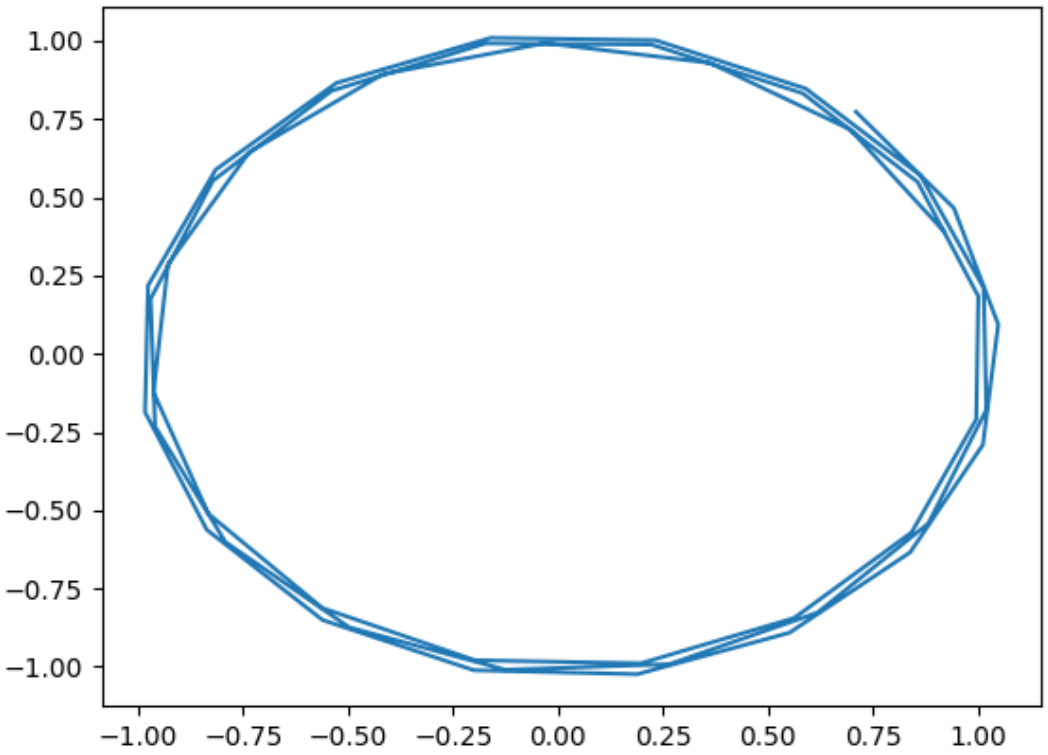

Figura 7: Representación deRK4 con un mayor Δt

Donde se observa la desviación de forma mucho más pronunciada que en el ejemplo anterior, constatando que, como era de esperar, al aumentar el Δt utilizado aumenta mucho el error.