Métodos de Runge – Kutta

Martínez Navallez Daniel Isac

Noviembre 2018

1 Introducción

En este artículo veremos un poco del método más utilizado de la familia de Runge – Kutta conocido como RK4 para resolver ecuaciones diferenciales ordinarias.

Se presentará un código en Fortran de RK4 para resolver la ecuación del péndulo.

2 RK4

Utilizado en análisis numérico para aproximar las soluciones a ecuaciones diferenciales orinarias como método iteractivo, utliza la rutina llamada método de euler.

Éste es el método más conocido de la familia de Runge – Kutta donde un problema de valor inicial se expresa de la siguiente manera:

$$\dot{y} = f(t, y), \quad y(t_0) = y_0$$

Aquí y es una función que no se conoce, y es dependiente del tiempo t se nos dice que $\dot{y}$, la velocidad a la que cambia y, es una función de t y de y en sí misma. En el tiempo inicial t_0 el valor de y correspondiente es y_0; los valores de t_0 y y_0 son datos conocidos. De la siguiente manera se elije un tamaño de paso $h > 0$ y se define:

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

$$t_{n+1} = t_n + h$$

para n=0,1,2,3... usando:

$$k_1 = h\, f(t_n, y_n),$$

$$k_2 = h\, f\Big(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}\Big),$$

$$k_3 = h\, f\Big(t_n + \frac{h}{2}, y_n + \frac{k_2}{2}\Big),$$

$$k_4 = h\, f\Big(t_n + h, y_n + k_3\Big).$$

Tendremos que y_{n+1} es la aproximación de RK4 para $y(t_{n+1})$, y el siguiente valor y_{n+1} está determinado por el valor presente de y_n mas el promedio ponderado de cuatro elementos determinados por el tamaño del intervalo h.

El método RK4 es un método de cuarto orden, lo que significa que el error de truncamiento local es del orden de $O(h^5)$, mientras que el error total acumulado es del orden de $O(h^4)$.

3 Código Fortran

```
Program Pendulo
IMPLICIT NONE

Real :: l, a, h, m, Ang_0, grados
!Variables para rk4
Real :: k1, k2, k3, k4, l1, l2, l3, l4
Real :: aux, aux1, aux2, aux3, aux4
!Vectores sin dimension
Real, allocatable :: t(:), W(:), Teta(:), Ang(:)
Integer :: i, n
real, external :: func
character:: output2*12

Print*, "longitud de la cuerda"
Read*, l

Print*, "Angulo inicial del pendulo"
read*,grados
Ang_0= (3.1416*grados)/180

Print*, "Tiempo de oscilacion"
Read*, a

Print*, "Ancho de paso"
Read*, h

print*,"nombre archivo de salida t vs grados"
read*,output2
```

```
!SE calcula el numero de particiones
m=a/h
!se toma un numero entero de particiones
n=NINT(m)

!Se le da dimension a los vectores

!Valores del angulo(radianes)
Allocate(Teta(n))
!Valores del angulo(grados)
Allocate(Ang(n))
!Valores del tiempo
Allocate(t(n))
!Valores de la velocidad angular
Allocate(W(n))


Print*, "Gracias!"

!Ponemos valores iniciales en los arreglos



Teta(1)=Ang_0
Ang(1)=grados

t(1)=0
W(1)=0


!!!!
Do i=2,n

!Primer pendiente
k1= h*W(i-1)
l1= h*func(Teta(i-1),l)

!Segunda pendiente
aux2= Teta(i-1)+(k1/2)
k2= h*(W(i-1)+(l1/2))
l2= h*func(aux2,l)

!tercer pendiente
aux3= Teta(i-1)+(k2/2)
k3= h*(W(i-1)+(l2/2))
```

```
l3= h*func(aux3,l)

!cuarta pendiente
aux4= Teta(i-1)+k3
k4= h*(W(i-1)+l3)
l4= h*func(aux4,l)

!hacemos una suma para teta
Aux= k1+(2*k2)+(2*k3)+k4
!hacemos una suma para la rapidez angular
Aux1= l1+(2*l2)+(2*l3)+l4

!Calculamos las nuevas rapidez y angulo
W(i)= W(i-1) + (aux1/6)
Teta(i)= Teta(i-1) + (aux/6)
Ang(i) = Teta(i)*(180/3.1416)
!aux/6 es el promedio de las pendientes

!Calcula paso del tiempo
t(i)=h*(i-1)

End do

Open(3,file=output2)
Do i=1,n
Write(3,*)t(i), Ang(i)

End do
Close (3)

End program Pendulo

Function func(Teta,l)
implicit none
Real :: Teta, func, l

!Esta funcion se usa con angulos grandes
func=(-9.81/l)*(Sin(Teta))
```

3.1 Gráficas

A continuación. se presentan las gráficas con los diferentes ángulos en orden de menor a mayor iniciando con 15 grados:

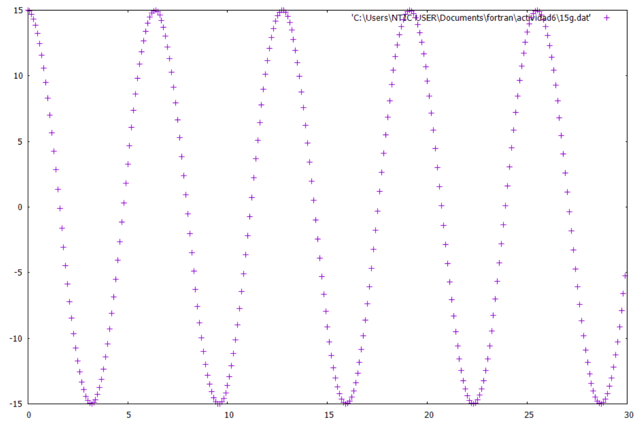

Figure 1: Gráfica con 15 grados

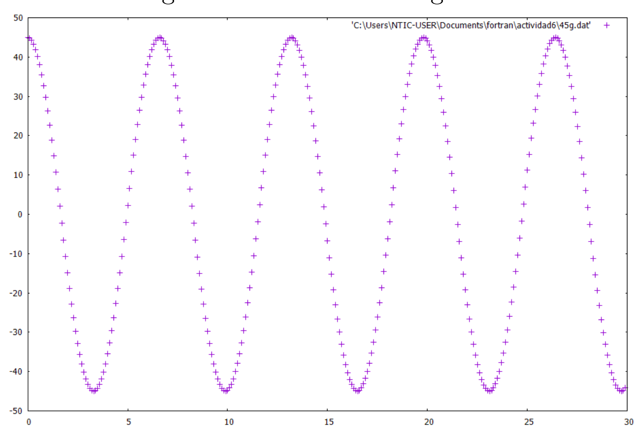

Figure 2: Gráfica con 45 grados

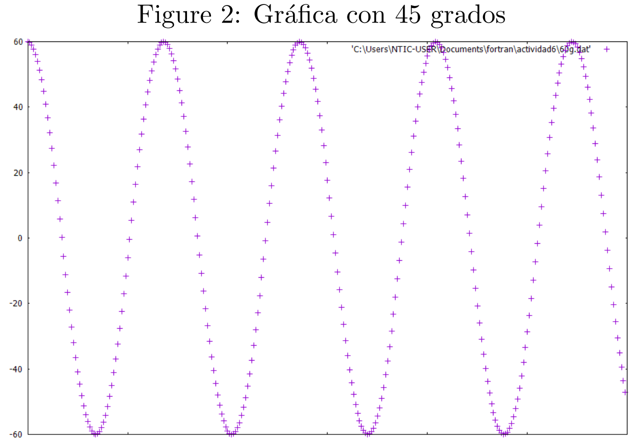

Figure 3: Gráfica con 60 grados

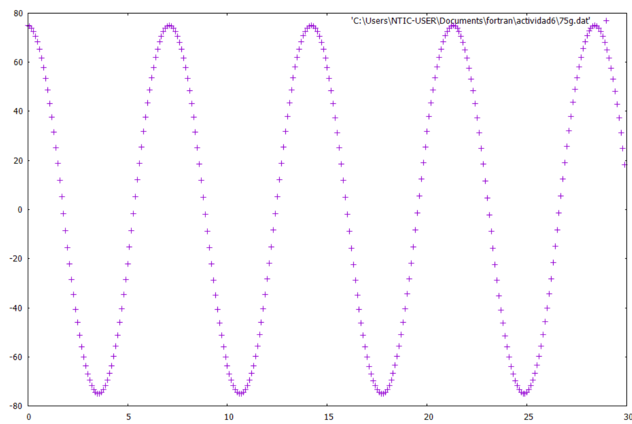

Figure 4: Gráfica con 75 grados