Instituto Politécnico Nacional

Escuela Superior de Cómputo

Tarea 8

Sesión 11 del parcial 2 de Métodos Numéricos.

Tema: Método de Runge - Kutta.

Nombre del alumno: De Luna Ocampo Yanina

Fecha de entrega: 26/10/2021

Introducción:

Utilizar este método de cuarto orden, sabemos que el error por truncamiento es del orden de h5. A partir de este conocimiento es posible encontrar una expresión preciosa para este error, sin embargo, esto es complicado.

Estos métodos evitan el cálculo analítico de las derivadas que aparecen en la serie de Taylor, haciendo una suposición donde hay constantes que se determinan de forma que la anterior expresión coincida con la serie de Taylor hasta el orden P.

Sus principales ventajas es que podemos cambiar el paso de integración en un momento dato, la estabilidad, la facilidad de programación y la posibilidad de estimar los errores de truncado en los métodos más sofisticados de estos. Pero como todo, hay una desventaja y es que de estos métodos es que se debe calcular la función f en puntos que no pertenecen a la red de integración.

Descripción: Aplique el método de Runge – Kutta de orden 4 al siguiente problema de valores iniciales:

$$y' = \frac{2 - 2yt}{t^2 + 1}, \qquad 0 \leq t \leq 1, \qquad y(0) = 1$$

Para lo anterior emplee $h = 0.1$, es decir, $N = 10$.

Comparando con la solución real que es:

$$y(t) = \frac{2t + 1}{t^1 + 1}$$

<u>Problema 1 del ejercicio 4.3:</u>

$$y' = \frac{2 - 2yt}{t^2 + 1}, \qquad 0 \leq t \leq 1, \qquad y(0) = 1$$

<u>Procedimiento:</u>

Debemos determinar nuestros parámetros dados por el ejercicio previo, tenemos nuestro punto inicial, punto final, condición inicial y el número de pasos que hará.

```
a = 0
b = 1
ci = 1
n = 10
```

Escribimos nuestra función f(t,y) para definirla y poder utilizarla. Se denota de la siguiente manera.

```
def fty(t,y):
    fty = (2 - 2*t*y)/(t**2 + 1)
    return fty
```

Determinamos el tamaño de nuestro salto.

```
h = (b - a) / n
```

Creamos nuestro arreglo y definimos nuestra función T^4. Una vez definida esta, hacemos lo mismo con nuestra solución real.

```
def ftyR(t):
    ftyR = (2*t + 1)/(t**2 + 1)
    return ftyR
```

Sacamos nuestros valores exactos y el error de aproximación. Acomodándolos en una tabla, podemos observar que nuestro error de aproximación se encuentra dado por:

Resultado:

La aproximacion obtenida se encuentra dada por:

Punto	Aproximacion	Real	Error Absoluto
0.0	1.0	1.0	0.0
0.1	1.18811876	1.18811881	5e-08
0.2	1.34615361	1.34615385	2.4e-07
0.3	1.46788934	1.46788991	5.7e-07
0.4	1.55172317	1.55172414	9.6e-07
0.5	1.59999866	1.6	1.34e-06
0.6	1.61764544	1.61764706	1.61e-06
0.7	1.61073648	1.61073826	1.77e-06
0.8	1.58536403	1.58536585	1.82e-06
0.9	1.54695954	1.54696133	1.79e-06
1.0	1.4999983	1.5	1.7e-06

}

Podemos observar que realmente el error va disminuyendo conforme avanza.

¿Qué aprendí?

Este es uno de los métodos más genéricos que nos sirve para la resolución numérica de ecuaciones diferenciales. Es un método de cuarto lo cual significa que el error por paso es del orden: 0(h 5) mientras que el error total acumulado tiene el orden: 0(h4).

Este método es mejor que el de Euler, pero aún así es posible aumentar la precisión disminuyendo los pasos entre los puntos o implementando este método, el de orden superior.