Informe Laboratorio 5: Ecuaciones Diferenciales Ordinarias

Nicolás Ibagón Rivera
Santiago Uribe Pastás
Pontificia Universidad Javeriana Cali

Diciembre 2020

Abstract

En el siguiente documento se presenta un informe acerca de la implementación en lenguaje de programación Python de diferentes métodos para resolver problemas de valor inicial, problemas de valor de frontera, con ecuaciones diferenciales ordinarias (ODE por sus siglas en ingles). Posteriormente se presentarán los métodos utilizados para la realización de dichas implementaciones junto con los resultados y su respectivo análisis después de probar los algoritmos con diferentes entradas. Los algoritmos aqui presentados fueron ejecutados en un computador con un procesador Intel Core i5-7200U y 16 GB de RAM.

1. Introducción

Una ecuación diferencial es una ecuación matemática que relaciona una función con sus derivadas. Estas ecuaciones pueden describir fenómenos, evolución y variación dinámicamente cambiantes; usualmente representan cantidades físicas, las derivadas representan sus razones de cambio, y la ecuación define la relación entre ellas.

Una ecuación diferencial ordinaria es una ecuación diferencial que contiene una o más funciones de una variable independiente y las derivadas de esas funciones. El término "ordinario"se usa en contraste con el término ecuación diferencial parcial que puede ser con respecto a más de una variable independiente.

El documento se encuentra organizado de la siguiente manera: en la sección 2 se hablara acerca de los materiales y métodos usados para la implementación de los algoritmos, en la sección 3 se presentarán los resultados obtenidos por lo algoritmos después de haberlos evaluado con diferentes funciones, y finalmente en la sección 4 se hablara sobre las conclusiones a partir de los resultados obtenidos.

2. Materiales y métodos

Primeramente como materiales fueron usados el lenguaje de programación Python junto con librerías que facilitaron en algunos aspectos la implementación del código, como lo son *numpy* y *matplotlib*. Además se hizo uso de las diapositivas y vídeos de clase para solucionar dudas acerca de teoría y métodos de resolución.

A continuación se hablará sobre la teoría matemática detrás de los métodos implementados.

2.1. Problemas de Valor Inicial

Una ecuación diferencial ordinaria $y' = f(t, y)$ no tiene una solución única. En general, hay una familia infinita de funciones que satisfacen la ecuación diferencial. Para encontrar una solución particular se debe especificar el valor denotado por y_0 en algún punto t_0. Entonces, se requiere conocer:

$$y(t_0) = y_0$$

Esto se conoce como el problema de valor inicial. La ODE gobierna la evolución dinámica del sistema en el tiempo, desde su valor inicial y_0 en el tiempo t_0. Lo que se busca es una función $y(t)$ que describe el estado de un sistema como función del tiempo. Los valores de la solución aproximada de la ODE se generan paso a paso en incrementos discretos en el tiempo. Por esta razón, los métodos numéricos para resolver ODEs se conocen como métodos de variable discreta.

2.1.1. Método de Euler

Se busca simular el comportamiento del sistema gobernado por la ODE. Empezando en t_0 con el valor inicial dado y_0, se desea seguir la trayectoria dictada por la ODE. La evaluación de $f(t_0, y_0)$ nos dice la pendiente o inclinación de la trayectoria en ese punto. Se usa esta información para predecir el valor y_1 de la solución en el tiempo futuro $t_1 = t_0 + h$, siendo h un incremento adecuado. Considere la siguiente serie de Taylor:

$$y(t+h) = y(t) = y'(t)h + \frac{y''(t)}{2}h^2 + ...$$
$$y(t+h) = y(t) + f(t, y(t))h + \frac{y''(t)}{2}h^2 + ...$$

El método de Euler elimina los términos de segundo orden y superior para obtener los valores de la solución aproximada:

$$y_{k+1} = y_k + f(t_k, y_k)h_k$$

Lo cual permite saltar del tiempo t_k a $t_{k+1} = t_k + h_k$. Equivalentemente, si se reemplaza la derivada en la ecuación diferencial $y'(t) = f(t, y)$ por la representación en diferencias finitas, se obtiene una ecuación algebraica:

$$\frac{y_{k+1} - y_k}{h_k} = f(t_k, y_k)$$

2.1.2. Método de Series de Taylor

Vimos anteriormente que el método de Euler se obtiene de la expansión en series de Taylor. Si se toman más términos en la serie de Taylor, se pueden obtener más métodos de paso sencillo (single-step) para órdenes más altos. Por ejemplo, tomando un término adicional en la serie de Taylor se obtiene el metodo de segundo orden:

$$y_{k+1} = y_k + y'_k h_k + \frac{y''_k}{2} h_k^2$$

Nótese que se requiere la segunda derivada de y''. Esto se puede obtener diferenciando $y' = f(t, y)$ usando la regla de la cadena:

$$y'' = f_t(t, y) + f_y(t, y)y' = f_t(t, y) + f_y(t, y)f(t, y)$$

2.1.3. Método de Runge-Kutta de Orden 2

Son métodos de paso sencillo que son similares a los métodos de serie de Taylor, pero no requieren la computación de derivadas de orden superior. En vez de eso, se simula el efecto de las derivadas superiores evaluando f varias veces entre t_k y t_{k+1}.

Recordemos que la segunda derivada de y esta dada por:

$$y'' = f_t + f_y f$$

Se puede aproximar el termino de la derecha expandiendo f en series de Taylor de 2 variables:

$$f(t + h, y + hf) = f + hf_t + hf_y f$$

Obteniendo:

$$f_t + f_y f = \frac{f(t + h, y + hf) - f(t, y)}{h}$$

Con esta aproximación, el método de Taylor de segundo orden se convierte a:

$$y_{k+1} = y_k + \frac{f(t_k, y_k) + f(t_k + h_k, y_k + h_k f(t_k, y_k))}{2} h_k$$

El cual puede ser implementado como:

$$y_{k+1} = y_k + \frac{1}{2}(k_1 + k_2)$$

donde:

$$k_1 = f(t_k, y_k)h_k,$$
$$k_2 = f(t_k + h_k, y_k + k_1)h_k$$

Este metodo tiene exactitud de orden 2.

2.1.4. Método de Runge-Kutta de Orden 4

El método de Runge-Kutta mejor conocido y más empleado es el esquema clásico de orden 4:

$$y_{k+1} = y_k + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

donde:

$$\begin{aligned} k_1 &= f(t_k, y_k)h_k, \\ k_2 &= f(t_k + h_k/2, y_k + k_1/2)h_k, \\ k_3 &= f(t_k + h_k/2, y_k + k_2/2)h_k, \\ k_4 &= f(t_k + h_k, y_k + k_3)h_k \end{aligned}$$

2.1.5. Métodos Multi-Paso

Estos métodos usan información en más de un punto previo para estimar la solución en el siguiente. Por esta razón, a veces se conocen como métodos con memoria. Los métodos lineales multi-paso tienen la siguiente forma general:

$$y_{k+1} = \sum_{i=1}^{n} \alpha_i y_{k+1-i} + h \sum_{i=0}^{n} \beta_i f(t_{k+1}, y_{k+1-i})$$

Los parámetros α_i y β_i se determinan con interpolación polinómica. Si $\beta_0 = 0$, el método es explicito, de lo contrario es implícito

La fórmula explicita del método de dos pasos es

$$y_{k+1} = y_k + \frac{1}{2}(3y'_k - y'_{k-1})h$$

Uno de los métodos multi-paso más populares es el método explicito de cuarto orden de Adams-Bashforth:

$$y_{k+1} = y_k + \frac{1}{24}(55y'_k - 59y'_{k-1} + 37y'_{k-2} - 9y'_{k-3}h)$$

2.2. Problemas de Valor de Frontera

Un problema de valor de frontera (PVF) para una ODE especifica más de un punto en el cual la solución o sus derivadas deben tener valores dados. Por ejemplo, un PVF para una ODE de segundo orden tiene la forma:

$$y'' = f(t, y, y'), a \leq t \leq b,$$

con condiciones de frontera:

$$y(a) = \alpha, y(b) = \beta$$

En general, para hallar la solución particular, deben existir tantas condiciones como el orden de la ODE.

2.2.1. Método de Diferencias Finitas

El método de diferencias finitas convierte los PVF en sistemas de ecuaciones algebráicas, reemplazando algunas derivadas por aproximaciones con diferencias finitas. Por ejemplo, para resolver un PVF de dos puntos:

$$y'' = f(t, y, y'), a \leq t \leq b,$$

con CF: $y(a) = \alpha, y(b) = \beta$. Primero se divide el intervalo $[a, b]$ en n subintervalos igualmente espaciados. Sea $t_i = a + ih$, $i \in [0, n]$ donde $h = \frac{b-a}{n}$. Se busca una aproximación $y_i \approx y(t_i)$ en cada uno de los puntos de la malla dados por t_i.

Ya tenemos $y_0 = \alpha, y_n = \beta$. Ahora reemplazamos las derivadas con aproximaciones de diferencias finitas:

$$y''(t_i) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}$$

Este reemplazo nos lleva a un sistema de ecuaciones algebraicas el cual debe ser resuelto para las incognitas $y_i, i \in [1, n-1]$. El sistema de ecuaciones puede ser lineal o no, dependiendo de que f sea lineal en y y y' .

2.2.2. Método de Elementos Finitos

Este método aproxima la solución de un PVF a una combinación lineal de funciones base ϕ_i, típicamente monomios. La aproximación tiene la forma:

$$y(t) \approx u(t) = \sum_{i=0}^{n} x_i \phi_i(t)$$

Los coeficientes x_i están determinados mediante la imposición de requerimientos del residuo, que se define como la diferencia entre el lado izquierdo y derecho de la ODE. El método que se implementara es el de colocación en el cual el residuo es 0, la ODE se satisface exactamente en los n puntos discretos.

3. Resultados

Para la ejecución de los algoritmos se escogieron 2 diferentes ecuaciones diferenciales con diferentes complejidades, las cuales son:

- $y_1(t) = 2t^4$, $y_1' = 8t^3$, $y_1'' = 24t^2$
- $y_2(t) = cos(t)$, $y_2' = -sin(t)$, $y_2'' = -cos(t)$

Es importante decir que para las gráficas de las soluciones analíticas fueron calculadas manualmente y posteriormente graficadas. También que las gráficas que involucren a la ODE y_2'' el eje x se encuentra en radianes.

3.1. Tiempos de Ejecución

En las siguientes tablas se muestra los tiempos de ejecución para los algoritmos.

ODE	Euler	Taylor	R.K. 2	R.K. 4	M.P. 2	M.P. 4
y_1'	$2{,}19{\cdot}10^{-5}$	$2{,}55{\cdot}10^{-5}$	$2{,}31{\cdot}10^{-5}$	$3{,}52{\cdot}10^{-5}$	$5{,}41{\cdot}10^{-5}$	$8{,}03{\cdot}10^{-5}$
y_2'	0,00018	0,00073	0,00039	0,00087	0,00055	0,00116

Tabla 1: Tiempos de ejecución para algoritmos de PVI

ODE	Dif. Finitas	Elem. Finitos
y_1''	0,436	0,401
y_2''	0,430	0,398

Tabla 2: Tiempos de ejecución para algoritmos de PVF

3.2. Gráficas

A continuación se presentan algunos resultados gráficos de algunas simulaciones de los algoritmos con las funciones y diferentes valores de h y de n.

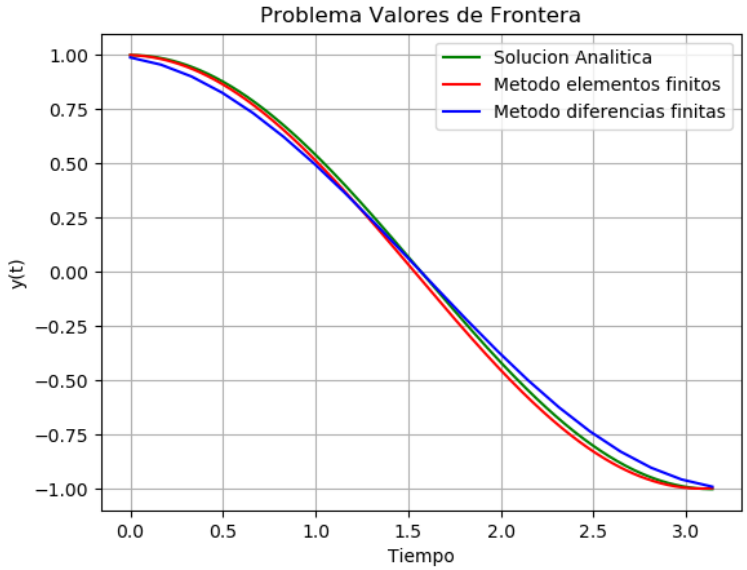

Figura 1: PVF para y_2'' y $n = 20$

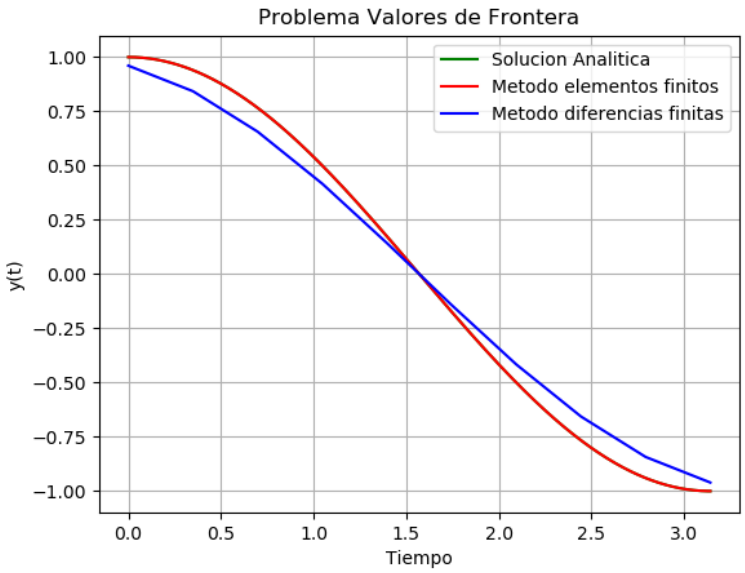

Figura 2: PVF para y_2'' y $n = 10$

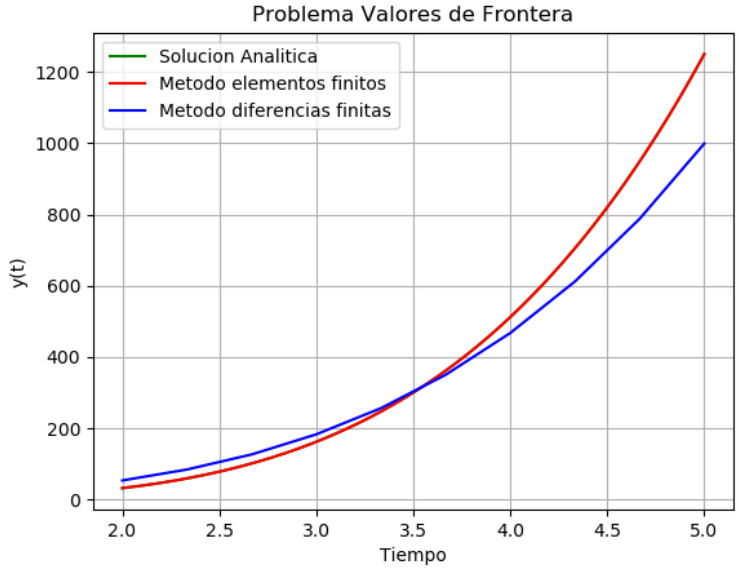

Figura 3: PVF para y_1'' y $n = 8$

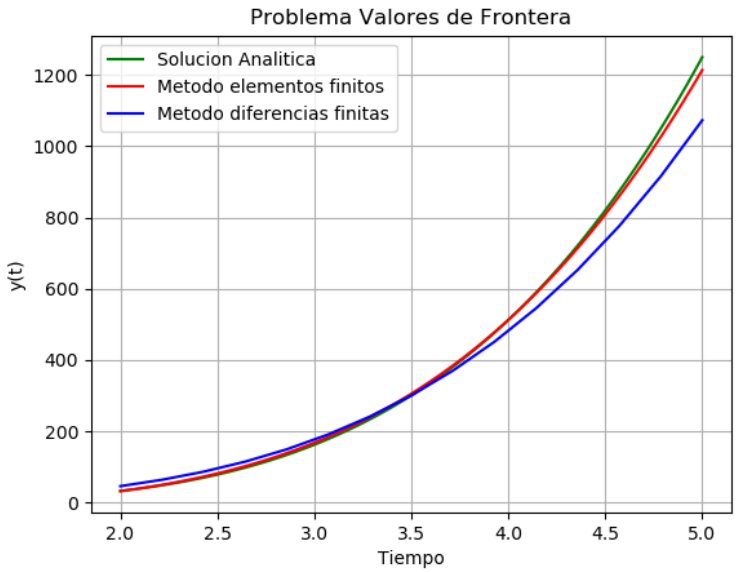

Figura 4: PVF para y_1'' y $n = 16$

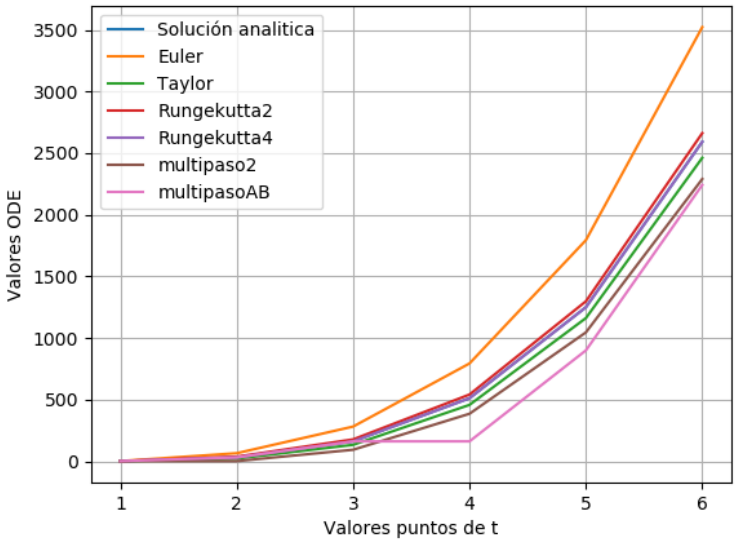

Figura 5: PVI para y_1' y $h = 1$

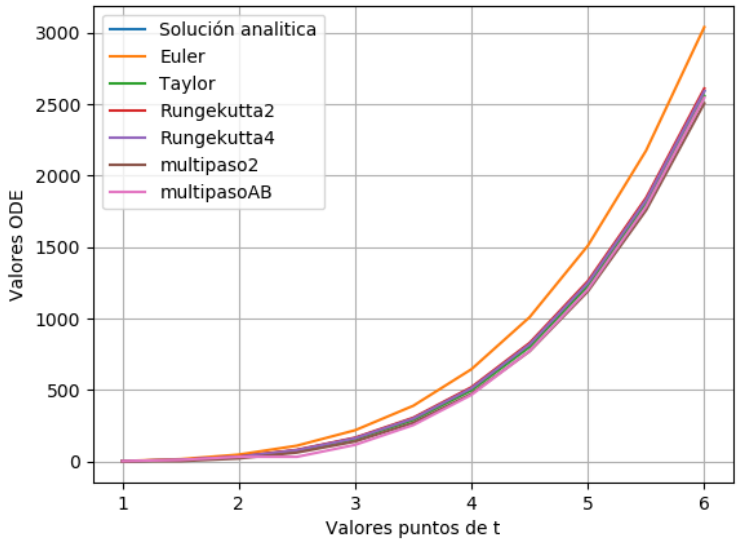

Figura 6: PVI para y_1' y $h = 0,5$

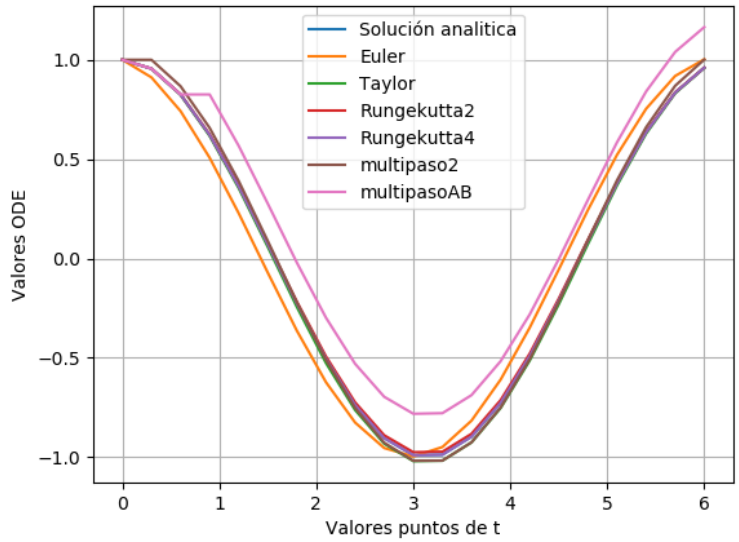

Figura 7: PVI para y_2' y $h = 0,3$

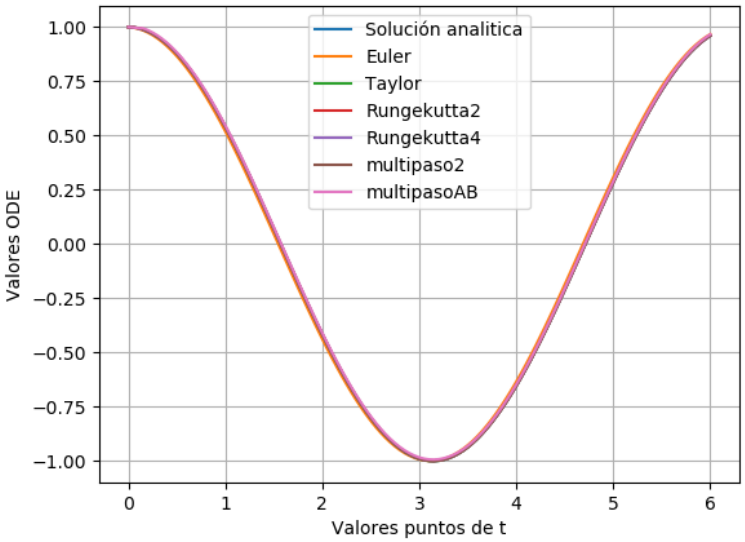

Figura 8: PVI para y_2' y $h = 0,05$

3.3. Error Relativo Promedio

A continuación se presentan el error relativo promedio para algunas de las simulaciones realizadas.

ODE	n	Dif. Finitas	Elem. Finitos
y_1''	8	0,2762	$1,2711{\cdot}10^{-10}$
y_1''	16	0,1266	1,5723
y_2''	10	0,1263	$1,9226{\cdot}10^{-9}$
y_2''	20	0,0661	0,1599

Tabla 3: Error relativo para algoritmos de PVF

ODE	h	Euler	Taylor	R.K. 2	R.K. 4	M.P. 2	M.P. 4
y_1'	1	0,5246	0,1185	0,06837	0,0001	0,3138	0,1831
y_1'	0,5	0,2843	0,0398	0,0214	0,0001	0,17007	0,1140
y_2'	0,3	0,3596	0,0418	0,0210	$7,89{\cdot}10^{-6}$	0,0451	0,5554
y_2'	0,05	0,0882	0,0016	0,0008	$8.66{\cdot}10^{-9}$	0,0013	0,0244

Tabla 4: Error relativo para algoritmos de PVI

4. Conclusión

Como se pudo observar en los diferentes métodos para resolver problemas de valor inicial para ODEs de primer orden, todos estos obtienen una buena aproximación con la condición de ingresar un valor de h bastante pequeño (cercano a 0). Sin embargo, se sugiere usar Runge-Kutta 4 pues este método tiene mayor precisión como se puede apreciar en la tabla de errores. Para los problemas de valor de frontera para ODEs de segundo orden podemos concluir que en el caso del método de elementos finitos a medida que el n es muy grande se tiene un error no tan bueno, esto debido a que este método aproxima polinomialmente si se tiene un n muy grande la solución sera un polinomio de grado muy alto, lo que no siempre es bueno. Para el caso de diferencias finitas se tiene un error bastante aceptable; a medida que n crece el error disminuye, por ende, se sugiere usar este método sobre el método de elementos finitos para tener una aproximación mejor.

Referencias

[1] "Mathematical functions - NumPy v1.19 Manual", numpy.org, 2008. [Online]. Available: `https://numpy.org/doc/stable/reference/routines.math.html`.