Universidad de los Andes
Departamento de Ingeniería Eléctrica y Electrónica
IELE 2009L – Laboratorio de computación científica para IEE
Gerardo Andrés Riaño Briceño (201112388)

Informe laboratorio 10

Teniendo en cuenta que la solución para la ecuación diferencial es

$$y(x) = 2 + x - C_1 e^{-x}$$
$$y(0) = 0 \rightarrow C_1 = 2$$

Se halla la solución numérica y se compara con el valor real de la función $y(x)$. Para ello, se evalúan tres métodos de Euler (hacia adelante, hacia atrás y modificado) y el método de Runge-Kutta de cuarto orden. Para cada método, se toma,

$$f(x, y) \equiv \frac{dy(x)}{dx} = 3 + x + y$$

La solución numérica se halla iterativamente, avanzando con pasos de h = 0.03 > 0.02. A continuación, se muestran los resultados obtenidos y los valores de error absoluto y relativo de cada método.

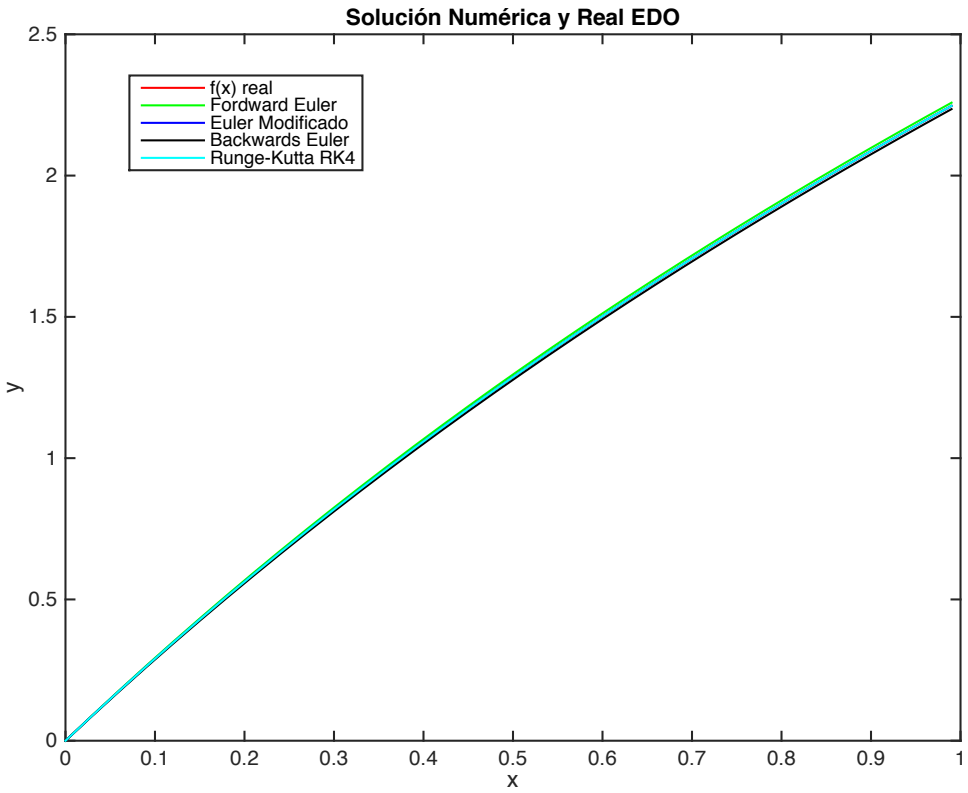

Figura 1. Solución numérica EDO.

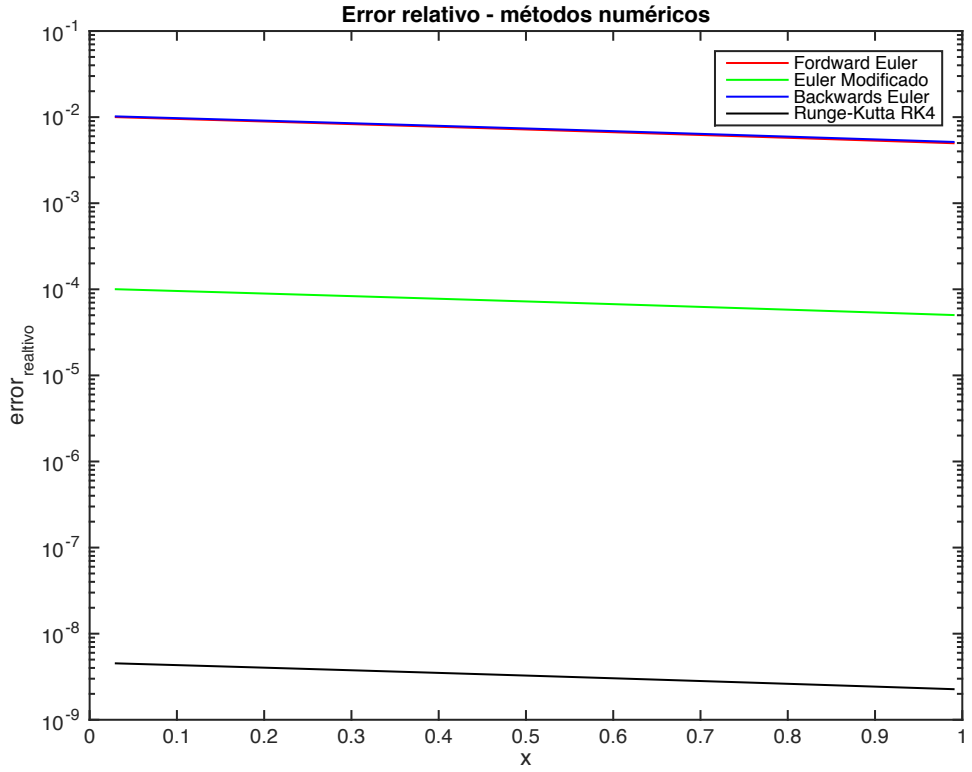

Figura 2. Error relativo de la solución numérica.

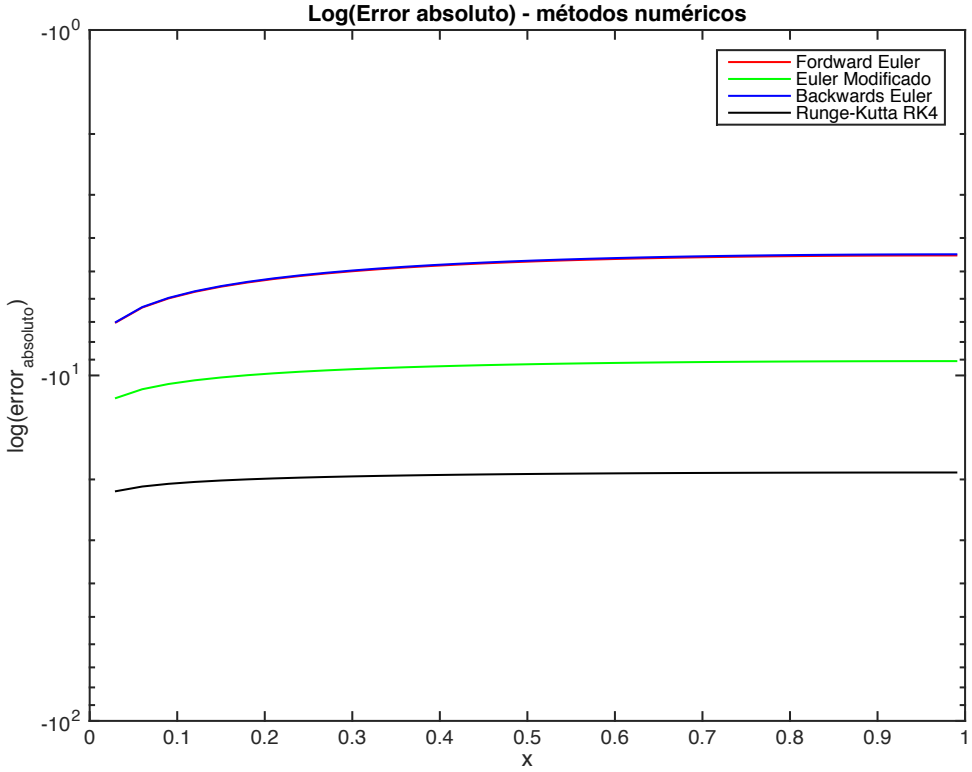

Figura 3. Error absoluto de solución numérica en escala logarítmica.

Gerardo Andrés Riaño Briceño (201112388)

Análisis de resultados

Los métodos de Euler hacia atrás y hacía adelante arrojan resultados similares en términos de precisión, ya que solo se diferencian por la posición donde se calcula el valor aproximado de la derivada. El método hacia atrás calcula la derivada al inicio del intervalo, y el método hacia delante lo hace al final del intervalo. De modo que el orden del error en cada iteración es cuadrático $O(h^2)$, teniendo en cuenta que al hacer una expansión por series de Taylor se tiene que,

$$y_{i+1} = y_i + hf(x_i, y_i) + O(h^2)$$

Sin embargo, el error global es $O(h)$, como se muestra en [1].

En cuanto al método de Euler modificado, la precisión mejora significativamente con respecto a los otros métodos de Euler; el valor del error relativo disminuye dos órdenes de magnitud con respecto al tradicional. Esto se debe a que el error en cada iteración es $O(h^3)$, y el global $O(h^2)$ [1].

$$y_{i+1} = y_i + hy_i' + \frac{h^2}{2}y_i'' + O(h^3)$$
$$= y_i + hf(x_i, y_i) + \frac{h^2}{2}(f_x(x_i, y_i) + f(x_i, y_i)f_y(x_i, y_i) + O(h^3)$$

Finalmente, el método de Runge-Kutta de cuarto orden permite obtener los resultados con menor error. El error relativo y absoluto disminuye varios órdenes de magnitud con respecto a los métodos descritos anteriormente, como se muestra en la **Figura 2** y la **Figura 3**. Se evidencia que el error global es $O(h^4)$ y el error en cada iteración $O(h^5)$ [2], ya que,

$$y_{i+1} = y_i + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5)$$

Tabla 1. Comparación de métodos de integración numérica para solución de EDO de primer orden.

Método	Costo computacional	Error de paso	Error global
Euler hacia atrás	$O(n)$	$O(h^2)$	$O(h)$
Euler hacia adelante	$O(2n)$	$O(h^2)$	$O(h)$
Euler modificado	$O(3n)$	$O(h^3)$	$O(h^2)$
Runge-Kutta 4	$O(4n)$	$O(h^5)$	$O(h^4)$

La complejidad de los algoritmos es lineal, y no hay grandes diferencias en la dificultad de implementación de los métodos. Por lo cual, se puede concluir que el método más apropiado de los cuatro, para hallar soluciones a ecuaciones diferenciales ordinarias de primer orden, es el de Runge-Kutta de cuarto orden. Con éste método el error es de menor orden y la magnitud del error relativo es cercana a la de errores por redondeo y truncamiento de la máquina, como se observa en la **Figura 2**.

Gerardo Andrés Riaño Briceño (201112388)

Anexos

```
clear all; clc; close all;

%% Forward Euler

h = 0.03;
x = (0:h:1)';
y1 = zeros(length(x), 1);

f = @(x,y) 3 + x - y;

for i=2 : length(x)
    y1(i) = y1(i-1) + h*f(x(i-1),y1(i-1));
end

%% Euler modificado

y2 = zeros(length(x), 1);

for i=2 : length(x)
    k1 = h*f(x(i-1), y2(i-1));
    k2 = h*f(x(i-1) + h, y2(i-1) + k1);
    y2(i) = y2(i-1) + (k1 + k2)/2;
end


%% Backwards Euler

y3 = zeros(length(x), 1);

for i=2 : length(x)
    k1 = h*f(x(i-1), y3(i-1));
    y3(i) = y3(i-1) + h*f(x(i), y3(i-1) + k1);
end


%% Runge-Kutta RK4

y4 = zeros(length(x), 1);
for i=2 : length(x)
    k1 = h*f(x(i-1), y4(i-1));
    k2 = h*f(x(i-1) + h/2, y4(i-1) + k1/2);
    k3 = h*f(x(i-1) + h/2, y4(i-1) + k2/2);
    k4 = h*f(x(i-1) + h, y4(i-1) + k3);
    y4(i) = y4(i-1) + (k1 + 2*k2 + 2*k3 + k4)/6;
end

%% Gráficas comparación con función real

yreal = 2 + x - 2*exp(-x);
plot(x,yreal,'r', 'DisplayName', 'f(x) real');
hold on;
plot(x,y1,'g', 'DisplayName', 'Fordward Euler');
plot(x,y2,'b', 'DisplayName', 'Euler Modificado');
plot(x,y3,'k', 'DisplayName', 'Backwards Euler');
plot(x,y4,'c', 'DisplayName', 'Runge-Kutta RK4');
```

```
legend('show');
xlabel('x');
ylabel('y');
title('Solución Numérica y Real EDO');

%% Gráfica error relativo

figure;
semilogy(x, abs(yreal-y1)./yreal,'r', 'DisplayName', 'Fordward Euler');
hold on;
semilogy(x, abs(yreal-y2)./yreal,'g', 'DisplayName', 'Euler Modificado');
semilogy(x, abs(yreal-y3)./yreal,'b', 'DisplayName', 'Backwards Euler');
semilogy(x, abs(yreal-y4)./yreal,'k', 'DisplayName', 'Runge-Kutta RK4');
legend('show');
xlabel('x');
ylabel('error_{realtivo}');
title('Error relativo - métodos numéricos');

%% Gráfica error absoluto

figure;
semilogy(x, log(abs(yreal-y1)),'r', 'DisplayName', 'Fordward Euler');
hold on;
semilogy(x, log(abs(yreal-y2)),'g', 'DisplayName', 'Euler Modificado');
semilogy(x, log(abs(yreal-y3)),'b', 'DisplayName', 'Backwards Euler');
semilogy(x, log(abs(yreal-y4)),'k', 'DisplayName', 'Runge-Kutta RK4');
legend('show');
xlabel('x');
ylabel('log(error_{absoluto})');
title('Log(Error absoluto) - métodos numéricos');
```