Análisis de Sistemas Electromagnéticos en Ciencias

Módulo 4: Tarea 1

Mauricio Andrés Flores Pérez - A01639917

1.- En un circuito RC como el mostrado en la figura 1 se puede calcular el cambio de voltaje (v) con respecto al tiempo (t) empleando las leyes de Kirchoff. Esto plantea la siguiente edo:

$$RC\frac{dy}{dt} + y = v(t)$$

Suponga que $RC = 0,1$ y se quiere calcular la solucion adiministrando cero Voltios al sistema, esto implica que $v(t) = 0.$ Así mismo, el voltaje inicial del capacitor es 2V. Resuelva la edo empleando RK 4. Compare el resultado con la solución analítica:

$$y(t) = 2e^{-10t}$$

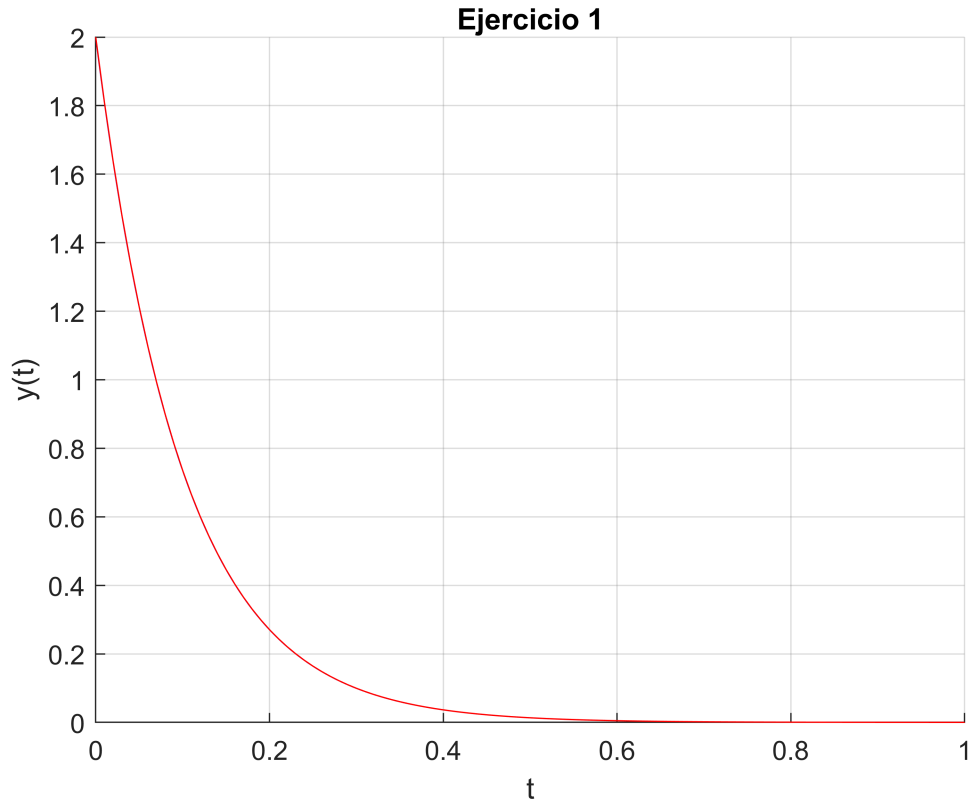
Es decir, genere una gráfica que superponga ambas soluciones.

Solución:

En primer lugar, hay que reacomodar los términos de la ecuación diferencial tomando en consideración que $v(t)$ es una función constante que en todo momento tendrá un valor de 0 como dicta el problema. Esto es:

$$\frac{dy}{dt} = -\frac{y}{RC}$$

```
% Obtencion de la solucion numerica por medio de Rk 4
RC = 0.1;
deq = @(x, y) -(y/RC);

[xpoints, ypoints] = rk_main(deq, @rk_4, 2, 0, 1, 0.1, 0.01);

% Solucion analitica
t = linspace(0, 1, 1000);
y = 2.*exp(-10.*t);

% Graficacion y comparacion de las soluciones
figure(1);
hold on
plot(t, y, 'b'), xlabel('t'), ylabel('y(t)'), grid on, title('Ejercicio 1')
plot(xpoints, ypoints, 'r')
```



2.- Resuelva la siguiente e.d.o. y grafique su aproximación numérica empleando: Euler, RK2 y RK4. No resuelva de manera analítica; Solo de manera numérica.

$$10\frac{dy}{dt} + y = 20 + sin(2t), y(0) = 15$$

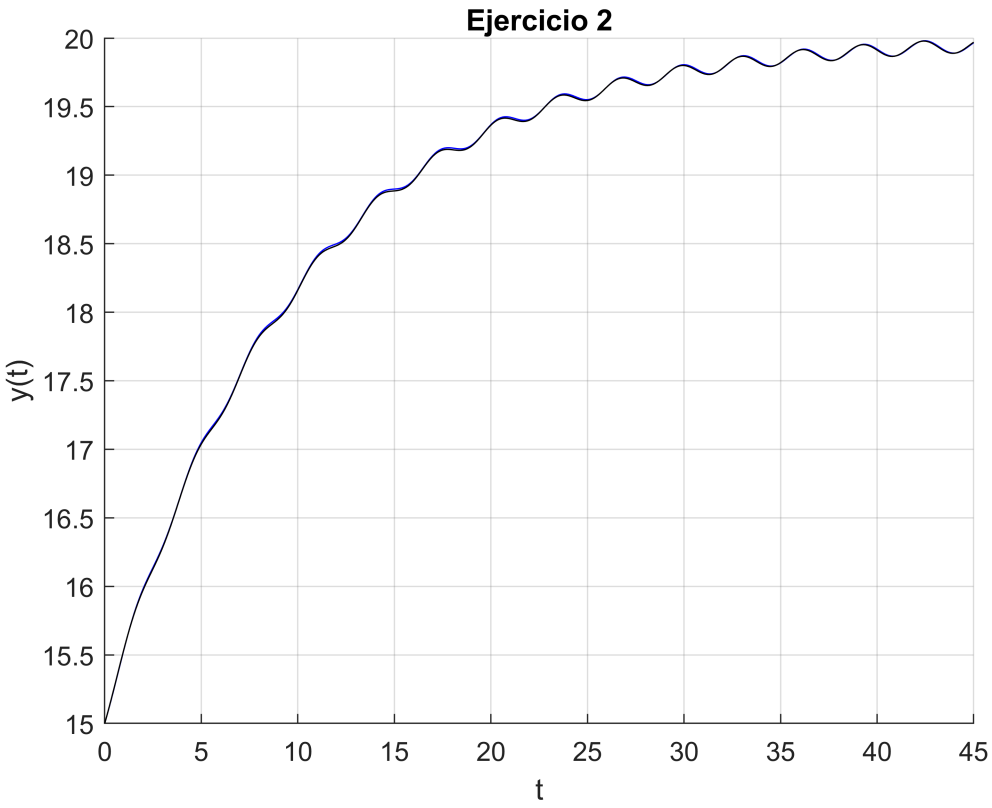
Solución:

Reacomodamos los términos de la ecuacion diferencial para poder utilizarla con nuestro métodos:

$$\frac{dy}{dt} = \frac{20 + sin(2t) - y}{10}, y(0) = 15$$

```
% Ecuaciones
dy =  @(t, y) (20 + sin(2.*t) - y)/10;

% Obtencion de la solucion por metodo de Euler
[epointsx, epointsy] = rk_main(dy, @eulerInt, 15, 0, 45, 0.1, 0.1);

% Obtencion de la solucion por metodo RK2
[rk2pointsx, rk2pointsy] = rk_main(dy, @ralstonInt, 15, 0, 45, 0.1, 0.1);

% Obtencion de la solucion por metodo RK4
[rk4pointsx, rk4pointsy] = rk_main(dy, @rk_4, 15, 0, 45, 0.1, 0.1);

% Graficacion de resultados
figure(2);
hold on
```

```
plot(epointsx, epointsy, 'b'), xlabel('t'), ylabel('y(t)'), grid on, title('Ejercicio 2')
plot(rk2pointsx, rk2pointsy, 'r')
plot(rk4pointsx, rk4pointsy, 'k')
```



3. - Investigue acerca de la función ode45 de Matlab, su uso y diga en qué escenarios es mejor emplear esa que un método de RK.

Ode45 es un solver de Matlab para ecuaciones diferenciales de la forma $y' = f(t, y)$, o bien problemas que impliquen una matriz de masas, $M(t, y)y' = f(t, y)$. El solver se basa en una fórmula explícita de Runge-Kutta(4,5), el par Domand-Prince. Esta función de matlab puede ser usada por medio de distintas sintaxis, por ejemplo:

[t, y] = ode45(odefun, tspan, y0), donde tspan = [t0 tf] integrando el sistema de t0 a tf con codiciones iniciales y0. Cada fila del array de soluciones y corresponde a un valor devuelto en el vector de columna t.

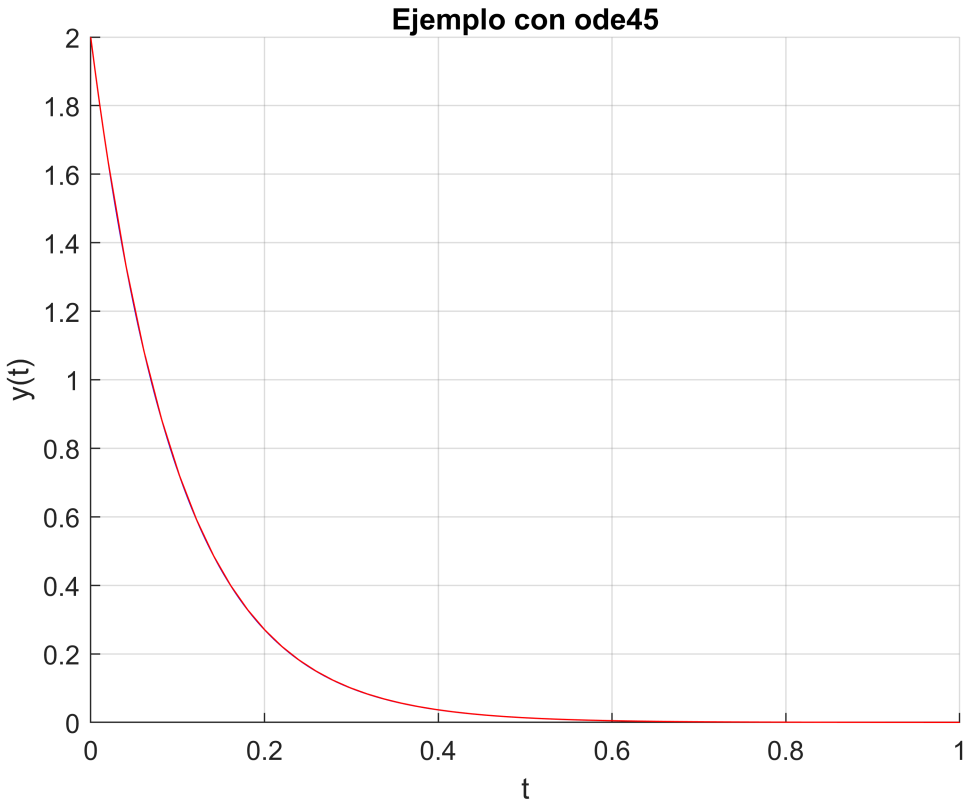
[t, y] = ode45(odefun, tspan, y0, options), esta función agrega el parámetro de options, el cual es un argumento creado mediante la función odeset y por el cual se pueden especificar las tolerancias de error absoluto y relativo.

Existen otras dos formas de utilización de este solver, pero dichas formas son utilizadas para propósitos mas especifícos en la aplicación, además de resolver la ecuación diferencial.

Ejemplo de uso resolviendo la ecuación diferencial del punto 1.

```
% Obtencion de la solucion numerica por medio de Rk 4
RC = 0.1;
deq = @(x, y) -(y/0.1);
```

```
[xpoints, ypoints] = ode45(deq, [0 1], 2);

% Solucion analitica
t = linspace(0, 1, 1000);
y = 2.*exp(-10.*t);

% Graficacion y comparacion de las soluciones
figure(3);
hold on
plot(t, y, 'b'), xlabel('t'), ylabel('y(t)'), grid on, title('Ejemplo con ode45')
plot(xpoints, ypoints, 'r')
```



¿Cuándo es mejor ode45 y cuándo es un método de RK?

Ode45 es el método por excelencia para resolver ecuaciones diferenciales en la suite de matlab. Es un método bastante robusto y eficiente de la suite de Matlab y además está basado en un método Runge-Kutta de quinto orden conocido como el par Dormand-Prince. El mismo matlab recomienda su uso como el primer solver a intentar al momento de resolver un problema de ecuaciones diferenciales que pueda ser simplificado a un orden 5. Este método se destaca por su efectividad en la relación exactitud-tiempo de computo, ya que su precisión es media. Por todas estas virtudes, generalmente siempre debería ser la opción principal al tratar con alguna ecuación diferencial de orden 5 o menor.

Aún así existen casos, en donde ode45 puede resultar inestable y en donde escoger un método RK distinto puede ser muy beneficioso. Ode45 debe ser utilizado con ecuaciones diferenciales no rígidas. Una ecuacion diferencial rígida es aquella que incluye términos que puedan llevar a una rápida variación en la solución.

Aunque esto no es fácil de reconocer en la práctica, si ode45 muestra ser ineficiente o inestable, se puede optar por otros métodos de RK como los mencionados en clase u otros solvers que también provee la misma suite de matlab y que están optimizados para dichas ecuaciones diferenciales como lo pueden ser ode15s, ode23s, ode23t u ode23tb. La elección de cada uno de ellos dependerá de la precisión, además de algunas consideraciones adicionales que se pueden ver en la documentación de matlab.

Funciones auxiliares

```
% Funcion rk_main
```

```
function [xp,yp] = rk_main(eq, method, y, x_ini, x_fin, step, output_interval)
    x = x_ini;
    m = 1;
    elems = (x_fin - x_ini)/output_interval;
    xp = zeros(1, elems);
    yp = zeros(1, elems);
    xp(m) = x;
    yp(m) = y;
    while x < x_fin
        x_end = x+output_interval;
        if x_end > x_fin
            x_end = x_fin;
        end
        h = step;
        [x, y] = integrator(eq, method, x, y, h, x_end);
        m = m+1;
        xp(m) = x;
        yp(m) = y;
    end
end

% Funcion integrator
function [x_new,y_new] = integrator(eq, method, x, y, h, x_end)
    x_new = x;
    y_new = y;
    while x_new < x_end
        if (x_end - x_new) < h
            h = x_end - x_new;
        end
        [x_new, y_new] = method(eq, x_new, y_new, h);
    end
end

% Funciones para integrar
% Metodo de Euler
function [x, y]= eulerInt(eq, x_ini, y_ini, h)
    y = y_ini+eq(x_ini, y_ini)*h;
    x = x_ini+h;
end

% Rk de segundo orden ralston
function [x, y] = ralstonInt(eq, x_ini, y_ini, h)
```

```
    a1 = 1/3;
    a2 = 2/3;
    p1 = 3/4;
    q11 = 3/4;
    k1 = eq(x_ini, y_ini);
    k2 = eq(x_ini + p1*h, y_ini + q11*k1*h);
    y = y_ini + h*(a1*k1 + a2*k2);
    x = x_ini + h;
end

% Rk de cuarto orden
function [x, y] = rk_4(eq, x_ini, y_ini, h)
    k1 = eq(x_ini, y_ini);
    k2 = eq(x_ini+(1/2)*h, y_ini+(1/2)*k1*h);
    k3 = eq(x_ini+(1/2)*h, y_ini+(1/2)*k2*h);
    k4 = eq(x_ini+h, y_ini+k3*h);
    y = y_ini + (1/6)*h*(k1 + 2*k2 + 2*k3 + k4);
    x = x_ini + h;
end
```

Referencias

ode45 (s.f.) Recuperado el 24/05/2021 dehttps://la.mathworks.com/help/matlab/ref/ode45.html

Choose an ODE solver (s.f.) Recuperado el 24/05/2021https://la.mathworks.com/help/matlab/math/choose-an-ode-solver.html