Trabajo Práctico Entregable 4

Cálculo Numérico
Ingeniería Informática

Adjadj, Agustín
(agustin.adjadj@gmail.com)

DEFINICIÓN DE LA FUNCIÓN.

La trayectoria de una partícula que inicialmente se encuentra en el punto (-1,1) y se mueve en el plano está dada por la curva $(x_1(t), x_2(t))$, donde las funciones x_1 y $x_2$son la solución del siguiente sistema de ecuaciones diferenciales:

$$\begin{cases} x_1'(\mathrm{t}) = -\mathrm{t} * x_2(t) \\ x_2'(\mathrm{t}) = \mathrm{t} * x_1(t) - t * x_2(t) \end{cases}$$

- *Grafique la trayectoria de la partícula durante los primeros 20 segundos.*
- *Utilice el método de Runge-Kutta 4 con paso h=0.1 para determinar la posición de la partícula y su rapidez a los tres segundos.*
- *¿Cuántos dígitos correctos tienen los resultados del ítem anterior? Explique cómo lo determinó.*
- *Recuerde que la longitud de la trayectoria de la partícula durante los T primeros segundos está dada por $\int_0^T \sqrt{x_1'(\mathrm{t})^2 + x_2'(\mathrm{t})^2}dt$. Calcule la distancia recorrida por la partícula durante los primeros tres segundos. Explique como lo hizo.*
- *Determine el instante de tiempo a partir del cual la partícula esta siempre a una distancia menor a 0.01 del origen de coordenadas.*

CONSIDERACIONES INICIALES

Para resolver este problema, se utilizarán métodos de soluciones de EDO's(Ecuaciones Diferenciales Ordinarias) de problemas de valor inicial. Un problema de valor inicial se tiene a partir de una ecuación diferencial $\frac{dy}{dx} = f(x, y)$ valida en el intervalo a < x < b. Esa ecuación tiene infinitas soluciones, para precisar una es necesario dar una condición que se denominará condición inicial, es decir, el problema es hallar la función que satisface la ecuación diferencial en el intervalo y que además satisface la condición inicial.

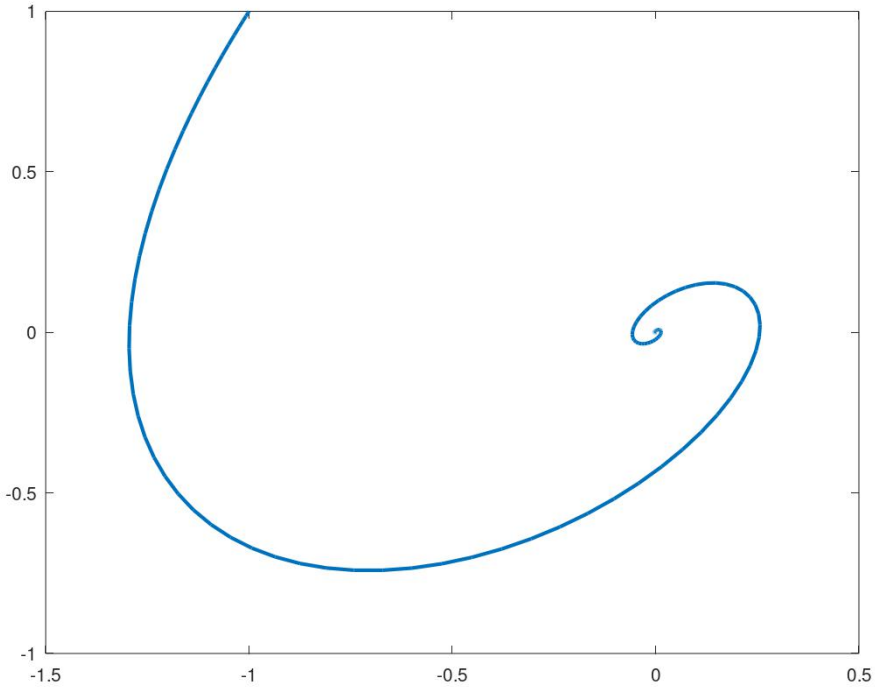
Para este trabajo se utilizarán métodos de un paso (también los existen en pasos múltiples). Dichos métodos obtendrán aproximaciones a y en determinados puntos en el intervalo [a,b]. Esos puntos están igualmente espaciados. Dividiendo (a,b) en n subintervalos, el tamaño del paso vendrá dado por $h = \frac{\mathrm{b}-\mathrm{a}}{\mathrm{n}}$ y la abscisa del nodo es $x_i = x_0 + i\,\mathrm{h}$. Los métodos de un paso permiten evaluar la solución numérica $y_{\mathrm{i}+1}$ en la abscisa $\mathrm{x}_{\mathrm{i}+1}$.

El método mas sencillo es el Método de Euler, que se utilizará aplicando Series de Taylor.

Otro método que vamos a utilizar será el de Runge-Kutta, que también tiene su desarrollo en serie de Taylor a partir de un punto. El mismo tiene varios ordenes de resolución, usaremos para este el de orden 4.

GRÁFICA DE LA PARTICULA.

Para graficar la trayectoria, utilizaré el método de Euler (Código 1) , con un h=0.05 para tener una grafica un poco más definida que si se usa un h=0.1 . En el código principal (Código 3), defino la función en las primeras líneas, para utilizarla a lo largo de los códigos necesarios. Luego defino los intervalos en los cuales se hará la función, y el N calculado con el h definido. La gráfica queda de la siguiente forma:

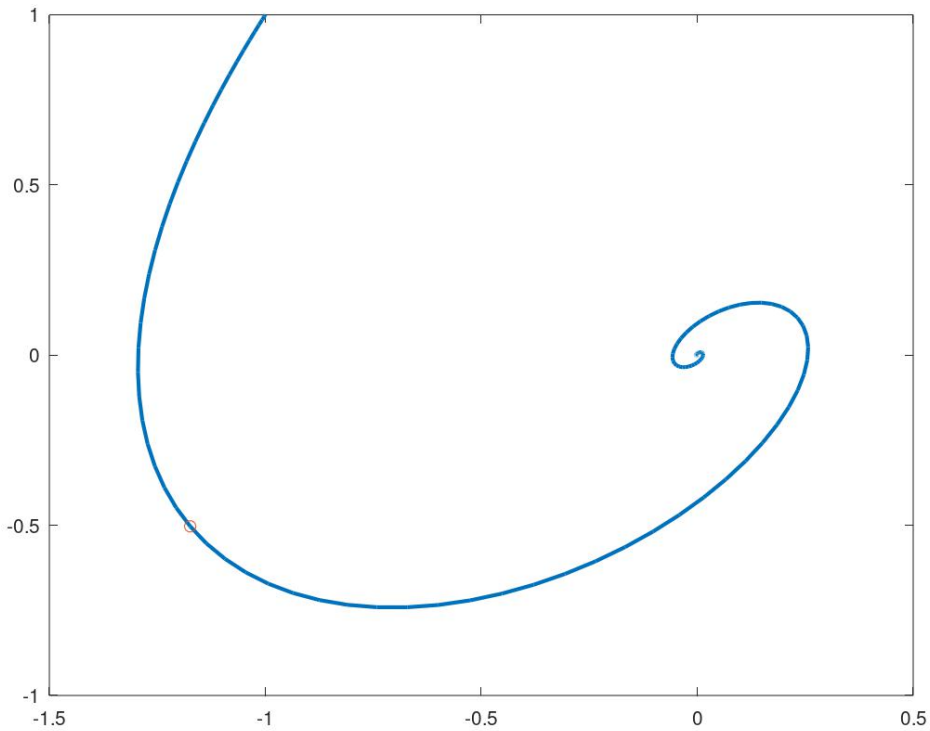


Dicha gráfica es el plot de las dos posiciones a través del tiempo.

POSICIÓN Y RAPIDEZ DE LA PARTÍCULA PARA T=3

Para la determinación de la posición y la rapidez de la partícula cuando t=3 [seg], se usará el método de Runge-Kutta de Orden 4 (Código 2). El mismo tendrá una variación con respecto al de Euler, que la cantidad de subdivisiones será la mitad de la utilizada con anterioridad. Una vez que obtenemos el resultado, se buscará la posición en el vector de tiempo en donde t=3. Una vez obtenido este resultado, graficamos el punto donde esto ocurre, y tendremos la posición.

Para calcular la rapidez, usaremos norm() que es la función de Octave que nos calcula la norma 2 de la variable, que en este caso será el resultado de la función para t=3. Dicha rapidez tendrá un valor de 0.482.



Gráficamente se puede ver el punto donde tenemos la posición de la partícula en t=3.

El resultado de la rapidez tendrá 3 dígitos correctos, esto se determina variando el N (a mayor N, mayor precisión) y, se puede ver que haciendo esto, se obtienen los 3 dígitos definidos con anterioridad.

LONGITUD DE LA TRAYECTORIA DESDE T=0 HASTA T=3.

Para calcular la longitud de la trayectoria, desde 0 hasta 3 se sabe que la misma viene dada por la siguiente función:

$$Longitud = \int_0^T f(t)\,\mathrm{dt} = \int_0^3 \sqrt{x_1'(\mathrm{t})^2 + x_2'(\mathrm{t})^2}\,dt$$

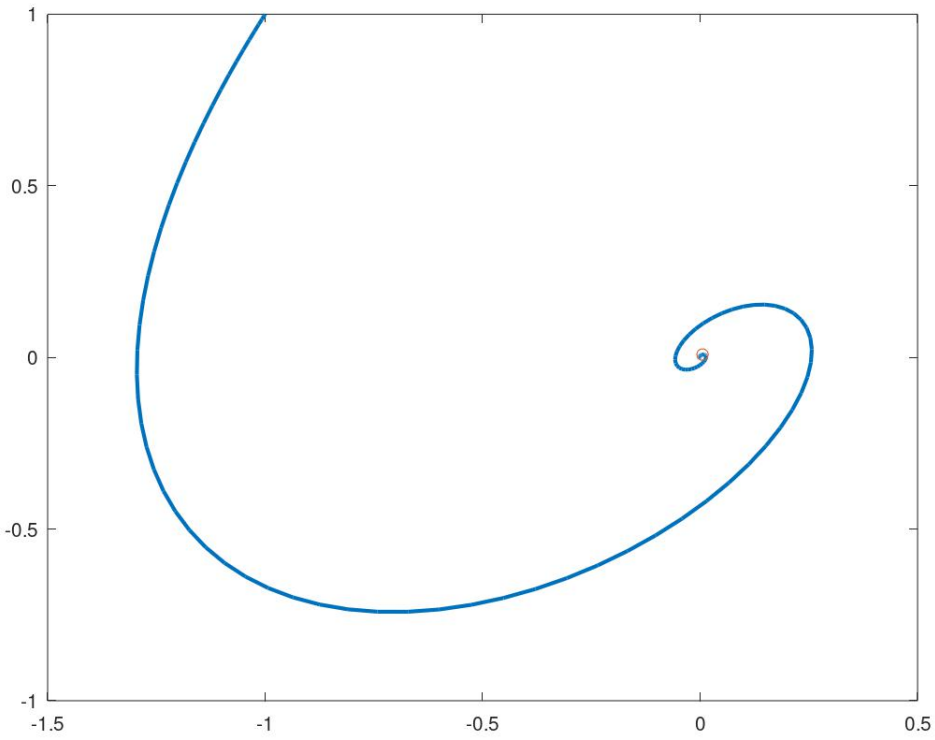
Para resolver dicha función, primero evaluaremos la misma en los puntos que corresponden dentro del intervalo [0,3], que está dentro del Código 3, Inciso 4. Una vez que estos resultados son obtenidos, utilizaré el método de Simpson Compuesto (Código 4), dicho método evalúa la integral de una función y la misma solo tiene como conocido los valores discretos, como ocurre en este caso.

Una vez realizados estos pasos, tenemos como resultado que la trayectoria recorrida por la partícula es de 3.354.

DETERMINACIÓN DE INSTANTE DE TIEMPO BAJO CONDICIÓN.

Se pide determina el momento en el que la partícula empieza a estar siempre a una distancia menor a 0.01 del origen de coordenadas. Para realizar esto, calcularemos la distancia que todos los puntos tienen con respecto al origen, utilizando nuevamente la función norm() de Octave, guardando todos estos datos en un vector. Realizado esto, tendremos entonces todas las distancias entre dos puntos a lo largo del tiempo, solamente nos interesa el punto en el cual el valor absoluto de la distancia menos 0.01 empieza a ocurrir.

Obtenemos dicho momento, el cual será a partir de t=4.95[seg] y lo graficamos para tener una idea del mismo, como se muestra a continuación:



CONCLUSIONES

Podemos concluir en general que la variación del paso, gráficamente, nos genera una gráfica mucho más definida cuando el h es más pequeño, en cambio al agrandar el h, ya tendremos una grafica un poco más "cuadrada" y con puntos poco definidos.

En temas de precisión, cuando variamos la cantidad de iteraciones, es decir, nuevamente el paso, tendremos una mayor precisión para pasos mas pequeños que para pasos más grandes.

BIBLIOGRAFÍA

- *Notas/Apuntes de clase.*
- *Libro Análisis Numérico - Richard L. Burden 10a Edición.*

ANEXO DE FUNCIONES

(1) - FUNCIÓN DEL MÉTODO DE EULER.

```
function [x,w] = euler_adelante_multi(f,a,b,y0,N)
  h = (b-a)/N;
  x = [a:h:b];
  w = zeros(length(y0), N+1);
  w(:,1) = y0;

  for n = 1:N
    w(:,n+1) = w(:,n) + h*f(x(n),w(:,n));
  endfor
endfunction
```

(2) - FUNCIÓN DEL MÉTODO DE RUNGE-KUTTA DE ORDEN 4 .

```
function [t,y]=rk4(f, inter, y0, L)
t = linspace(inter(1),inter(2),L+1)';
h = (inter(2)-inter(1))/L;

y = zeros( length(y0), L+1 );

y(:,1) = y0;

for n=1:L
  k1 = h * f(t(n)  , y(:,n));
  k2 = h * f(t(n)+h/2, y(:,n)+k1/2);
  k3 = h * f(t(n)+h/2, y(:,n)+k2/2);
  k4 = h * f(t(n+1), y(:,n)+k3);

  y(:,n+1) = y(:,n) + (k1+2*k2+2*k3+k4)/6;
end

y=y';

endfunction
```

(3) - FUNCIÓN PRINCIPAL.

```
f = @(t,x) ([...
            - t * x(2);...
            t * x(1) - t*x(2)...
            ]);

t0 = 0; %Tiempo Inicial.
tF = 20; %Tiempo Final.

x0 = [-1;1];

h = 0.05;
N = (tF - t0)/h;

[t,x] = euler_adelante_multi(f,t0,tF,x0,N);
```

```
#Inciso 1.
figure(1);
plot(x(1,:), x(2,:), "linewidth", 2);

#Inciso 2.
[tRK, xRK] = rk4(f,[t0 tF],x0,N/2);

for i = 1:length(tRK)
  if (tRK(i) == 3)
    posT = i;
  endif
endfor

hold on;
plot(x(1,posT),x(2,posT),"o"); #Posicion de la particula en 3 segundos.
rap = norm(f(tRK(posT),[xRK(posT,1) xRK(posT,2)])); #Rapidez de la partícula
en 3 segundos.

#Inciso 4
compTray = sqrt ( (-tRK.*xRK(:,2)).^2 + (tRK.*xRK(:,1)-tRK.*xRK(:,2)).^2 );
Tray = SimpsonCompDatos(tRK(1:31),compTray(1:31)); %31 es el valor para el
cual t=3

#Inciso 5
d = norm(x,2,"cols");
[~,p] = min(abs(d - 0.01));
hold on;
plot(x(1,p),x(2,p),"o");
```

(4) - Función Simpson Compuesto para un conjunto de datos.

```
function S = SimpsonCompDatos(x,y)
  %Regla de Simpson Compuesto dado los datos x,y
  N = length(x);                %Se supone impar
  h = (x(N) - x(1))/(N-1);      %Calculamos el h

  S0 = y(1) + y(N);             %f(a) + f(b)
  S1 = 0;                       %Parte par
  S2 = 0;                       %Parte impar

  for i = 2:(N-1)               %Desde 2 hasta n-1
    r = mod(i,2);               %Chequeo si la iteración es par o impar
    if (r == 0)
      S1 += 4*y(i);             %Si es par, la sumo a S1
    else
      S2 += 2*y(i);             %Si es impar, la sumo a S2
    endif
  endfor

  S = h*(S0 + S1 + S2)/3;       %Calculo el S final.

endfunction
```