TP0
INTRODUCCIÓN A LOS METODOS NUMÉRICOS

Materia: Dinámica de los Sistemas Físicos

Alumno: Ramirez, Fernando (R-3888/1)

Año: 2020

Resumen

En el presente trabajo practico se ejecutaron resoluciones de EDO's mediante distintos métodos de integración numérica utilizando la herramienta MatLab. En particular se utilizaron los métodos de Euler, Heun, ode45(Runge-Kutta orden4y5) y ode15s (método implícito basado en el algoritmo DASSL) y se realizó un análisis de los errores de cada método y de la conveniencia de cada uno para distintos tipos de modelos físicos.

Introducción

Los sistemas continuos habitualmente se representan mediante ecuaciones diferenciales ordinarias (EDO's). Por lo tanto, si se quiere predecir el comportamiento de dichos sistemas ante distintas entradas y/o condiciones iniciales, se deberán resolver las ecuaciones diferenciales en cuestión. Desafortunadamente, a excepción de los casos lineales y algunos casos no lineales triviales, las EDO's carecen de solución analítica. Inclusive en los pocos casos en los que se puede encontrar una solución, el procedimiento suele ser bastante engorroso y el resultado suele ser una expresión por demás de compleja. Por este motivo, se recurre casi siempre al uso de distintos algoritmos numéricos que permiten obtener una solución aproximada de la EDO en distintos instantes de tiempo. Estos algoritmos, denominados métodos de integración de ecuaciones diferenciales se suelen implementar en distintas herramientas de software, algunas de propósito general (como Matlab, Scilab, etc.) o bien para dominios específicos (el PSpice por ejemplo para simular circuitos). En este Trabajo Práctico utilizaremos algunos métodos de integración numérica muy conocidos y exploraremos algunas de sus características. Como herramienta utilizaremos primero Matlab y luego el entorno gráfico de simulación que provee dicho software, llamado Simulink.

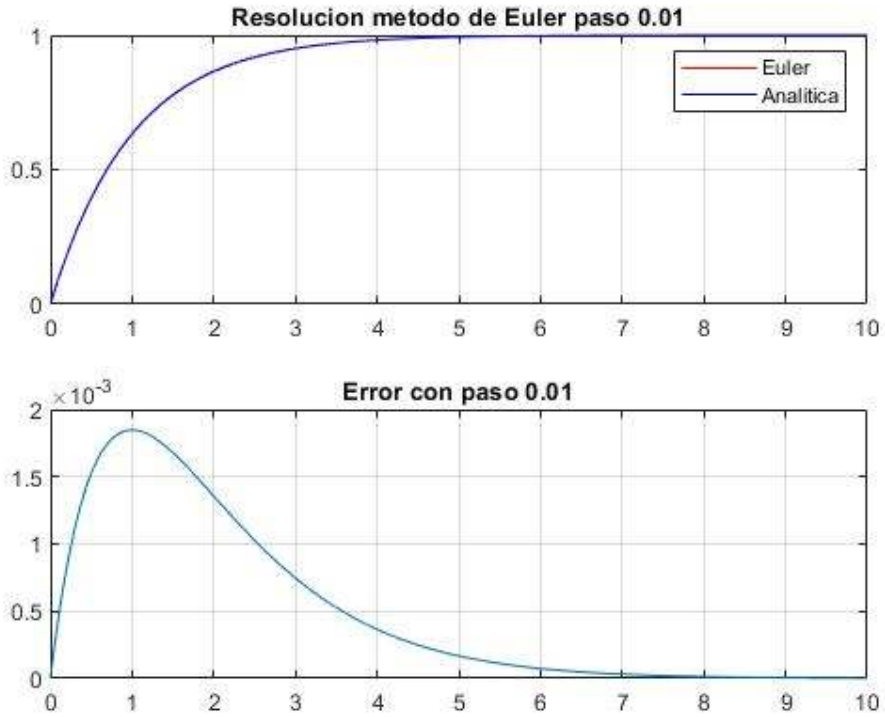
Desarrollo

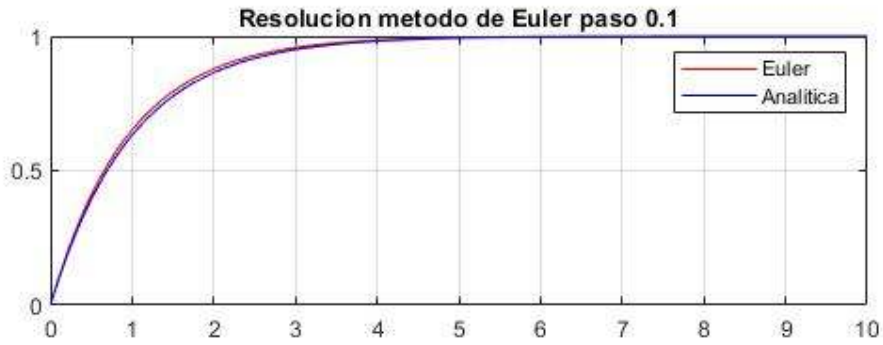
Nota: Todos los scripts de Matlab serán adjuntados al informe, ninguno será copiado al presente informe.

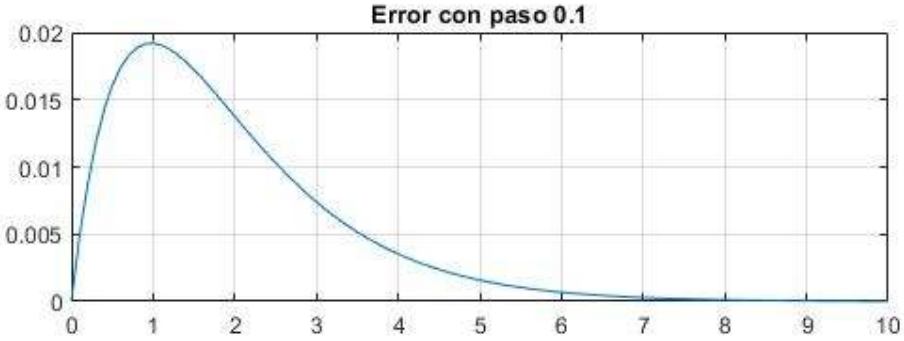
Sistema Elemental de Primer Orden – Euler

Para la resolución de este apartado se crearon dos archivos, la función “Euler.m” y el script “ejercicio2_1.m”. El primero es una función que pasándole como parámetro el tamaño del paso resuelve la EDO solicitada por el método de integración numérica de Euler. El segundo es un script que llama repetidas veces a la función anterior pero cambiándole el tamaño del paso en cada llamado y que luego plotea la resolución otorgada por la función vs la resolución analítica en los mismos instantes de tiempo junto con el error. De esta manera obtenemos una idea del cambio del error con el cambio del paso.

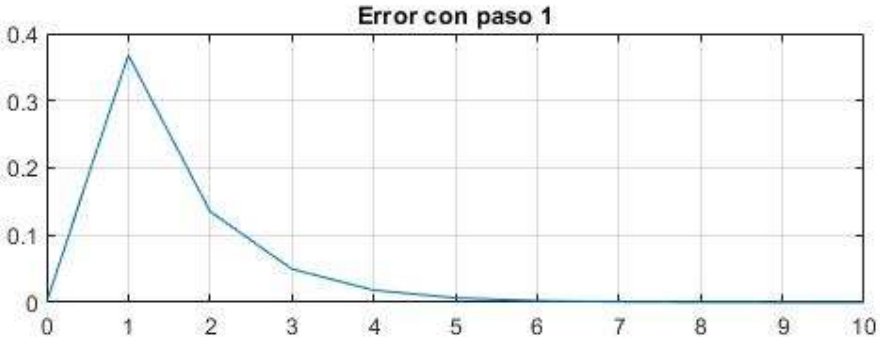
A continuación, se presentan las gráficas obtenidas.

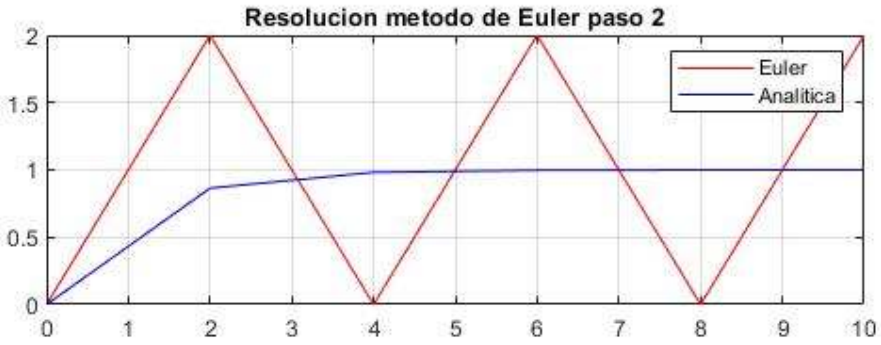


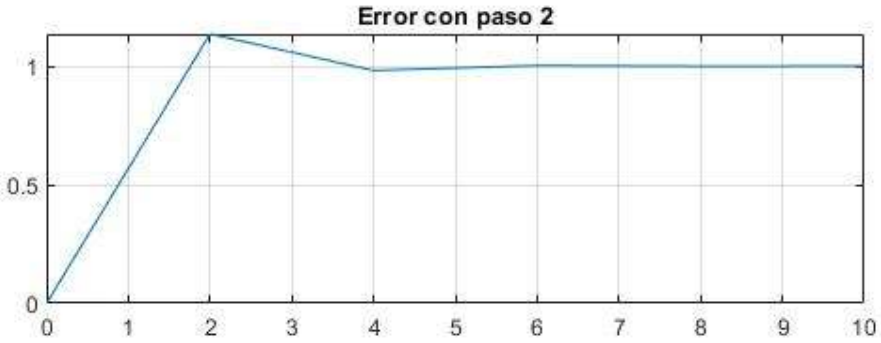

Resolucion metodo de Euler paso 0.1
Euler
Analitica
1
0.5
0
0 1 2 3 4 5 6 7 8 9 10

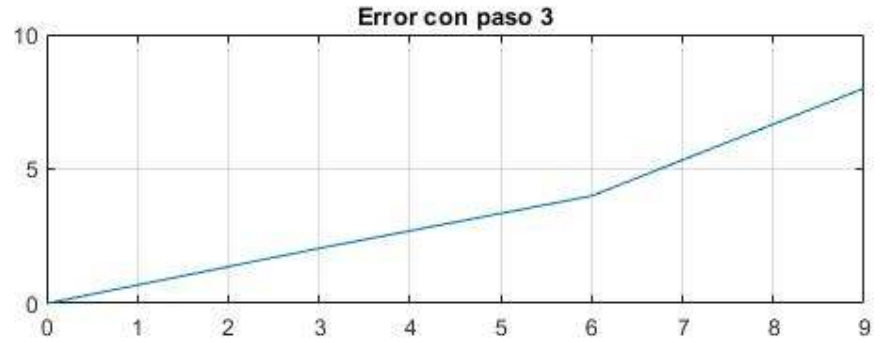

Error con paso 0.1
0.02
0.015
0.01
0.005
0
0 1 2 3 4 5 6 7 8 9 10


Resolucion metodo de Euler paso 1
Euler
Analitica
1
0.5
0
0 1 2 3 4 5 6 7 8 9 10


Error con paso 1
0.4
0.3
0.2
0.1
0
0 1 2 3 4 5 6 7 8 9 10







Cuando analizamos métodos de integración podemos definir dos tipos de errores. El Error Local y el Error Global.

El error Local es el error cometido tras el primer paso de integración y que en el método de Euler es de orden del paso al cuadrado.

El error Global es la diferencia entre la solución analítica y la solución numérica en un instante de tiempo arbitrario y que en el método de Euler es proporcional al paso de integración.

Dado que el error Global es lo que estamos analizando y teniendo en cuenta que este es proporcional al paso de integración es de esperar que este aumente cuando tomamos pasos de integración mayores. Si observamos las graficas notamos que efectivamente esto es lo que esta sucediendo.

En el caso particular para los pasos de integración h=2 y h=3 el sistema pierde la estabilidad numérica por lo cual la respuesta comienza a oscilar. Siempre que utilicemos el método de Euler, tendremos que obtener el límite de estabilidad del sistema y elegir un paso menor todavía para asegurar la estabilidad de la solución, ya que si elegimos el limite como paso de integración, tenemos un 50% de probabilidades de que sea inestable, lo que nos llevaría a tener que ejecutar, a veces, varias veces el algoritmo para obtener la respuesta adecuada que para sistemas complejos no es deseable por una cuestión de recursos y disponibilidad de ellos.

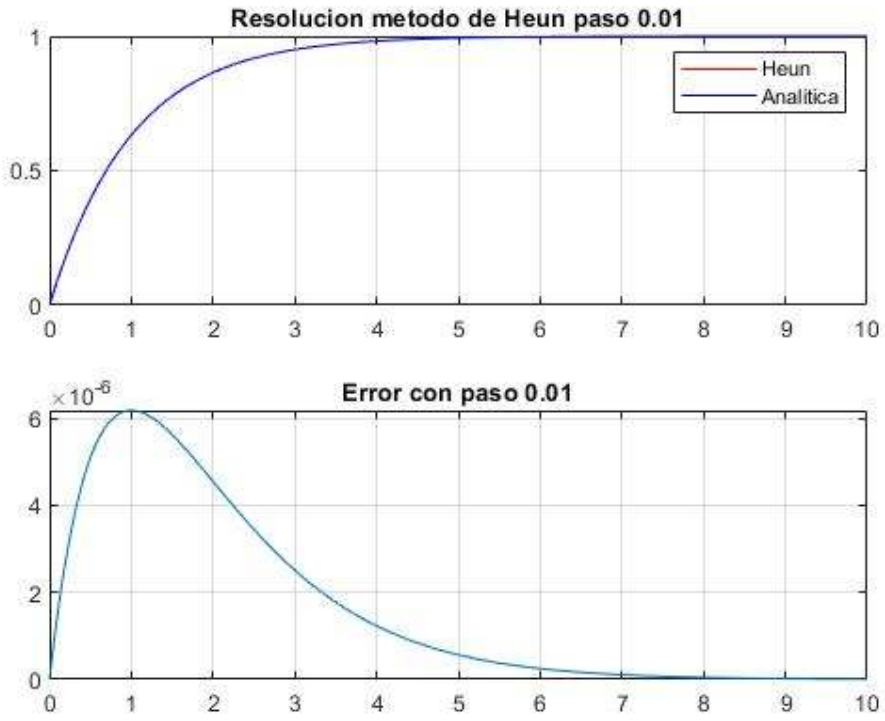
El limite del paso de integración viene dado por una inecuación la cual es la siguiente $|\lambda h+1|<1$ donde lambda son los autovalores de la matriz evolución del sistema. Resolviendo esta inecuación arribamos al valor máximo de paso de integración que podríamos elegir

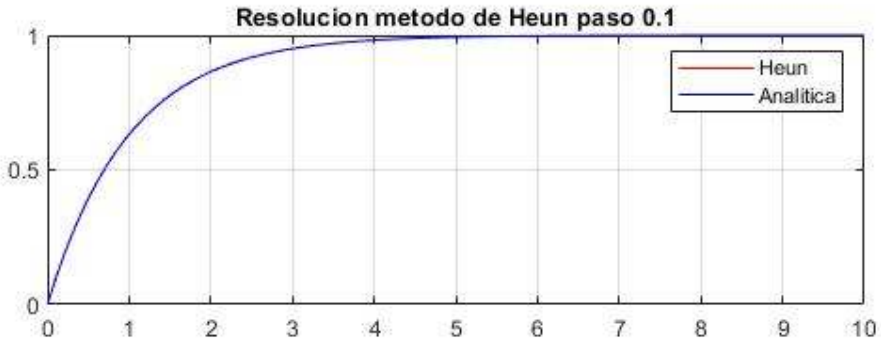
Método de Heun

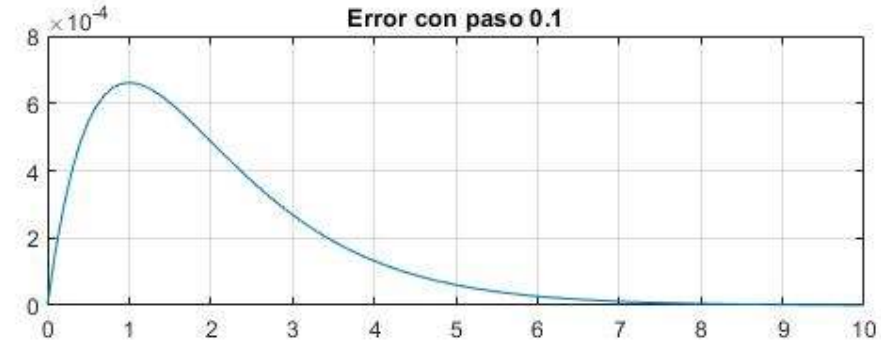
En este apartado se programaron nuevamente una función y un script, “Heun.m” y “ejercicio2_2.m” respectivamente.

La función resuelve el la EDO del sistema mediante el método de Heun, y el script realiza lo mismo que el “ejercicio2_1.m” agregando un ploteo extra donde se muestra la diferencia entre los errores de cada método.

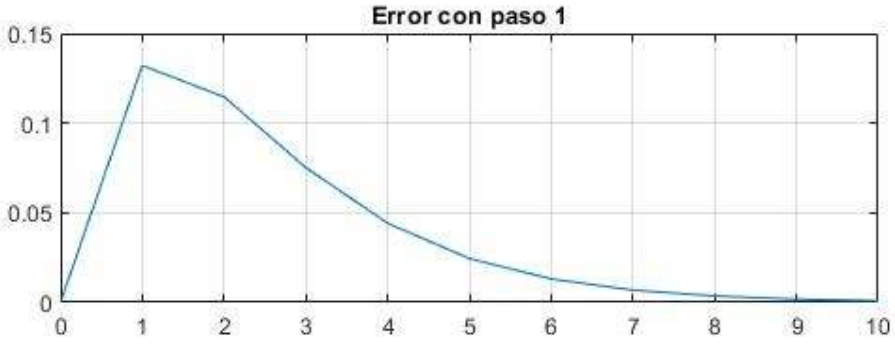
Se presentan las gráficas obtenidas ordenadas de la siguiente manera. Primero las comparativas entre el método de Heun y la analítica con el respectivo error del método, y luego la comparativa de errores entre el método de Heun y el de Euler.



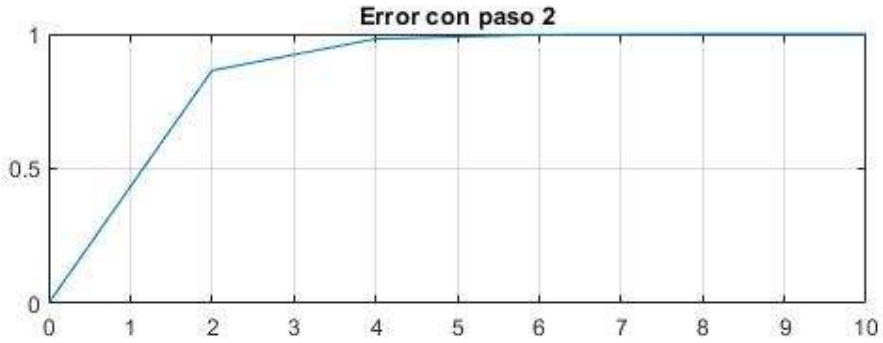

Resolucion metodo de Heun paso 0.1
Heun
Analitica
0
0.5
1
0
1
2
3
4
5
6
7
8
9
10


Error con paso 0.1
×10^-4
0
2
4
6
8
0
1
2
3
4
5
6
7
8
9
10


Resolucion metodo de Heun paso 1
Heun
Analitica
0
0.5
1
0
1
2
3
4
5
6
7
8
9
10


Error con paso 1
0
0.05
0.1
0.15
0
1
2
3
4
5
6
7
8
9
10

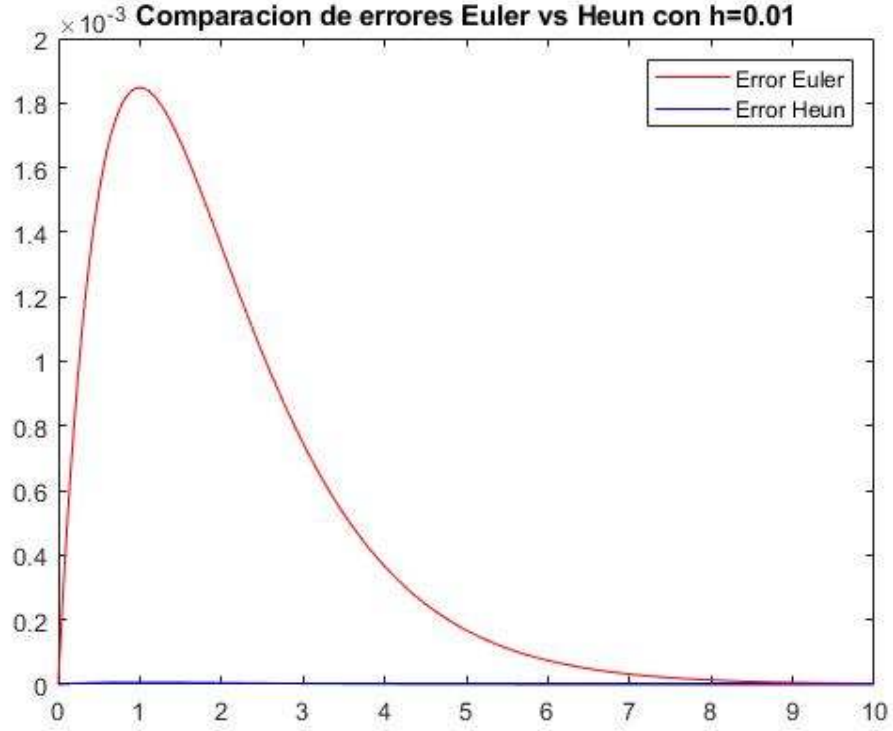

Resolucion metodo de Heun paso 2
Heun
Analitica
1
0.5
0
0
1
2
3
4
5
6
7
8
9
10

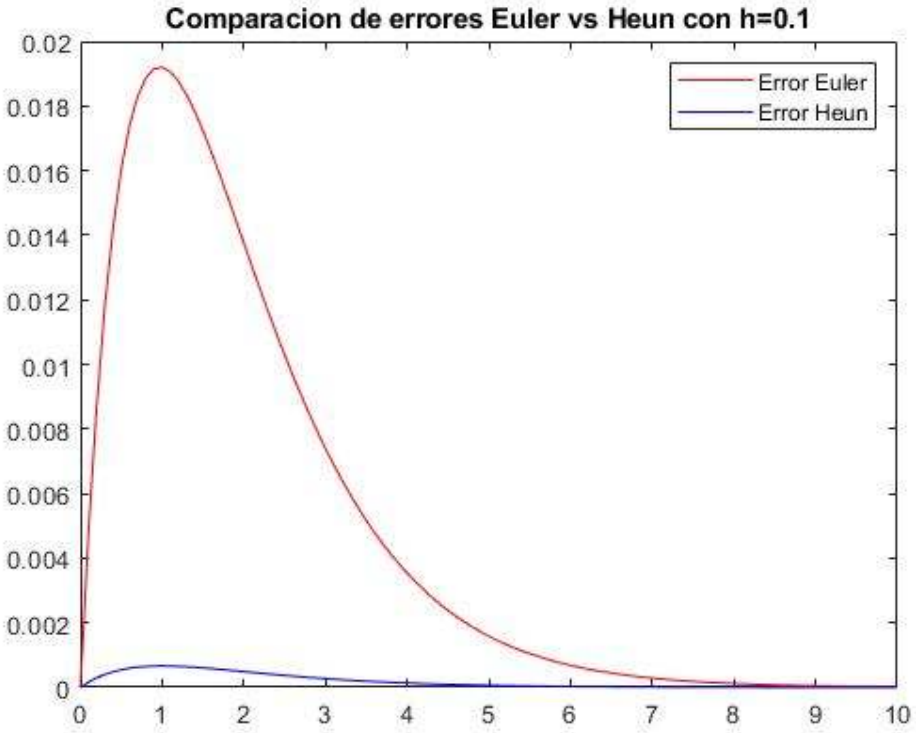

Error con paso 2
1
0.5
0
0
1
2
3
4
5
6
7
8
9
10

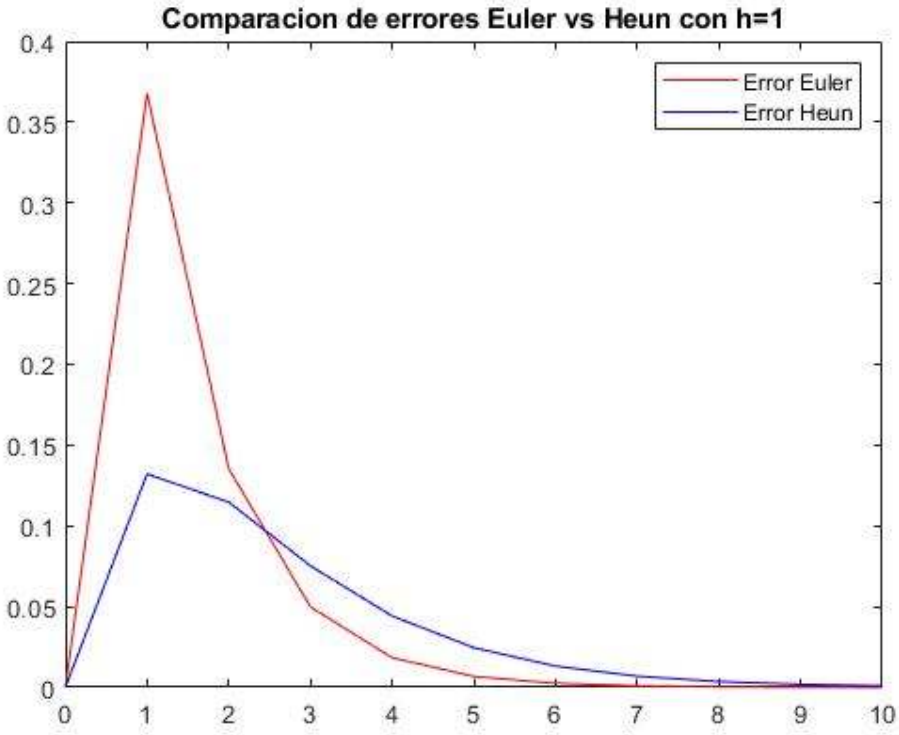

Resolucion metodo de Heun paso 3
Heun
Analitica
0
-5
-10
-15
0
1
2
3
4
5
6
7
8
9

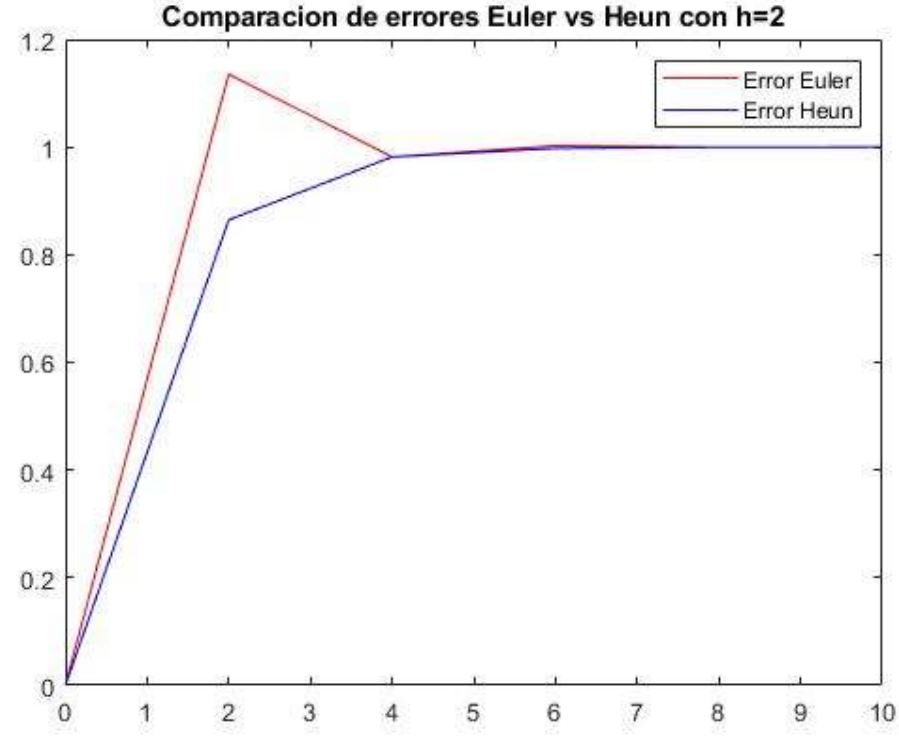

Error con paso 3
15
10
5
0
0
1
2
3
4
5
6
7
8
9

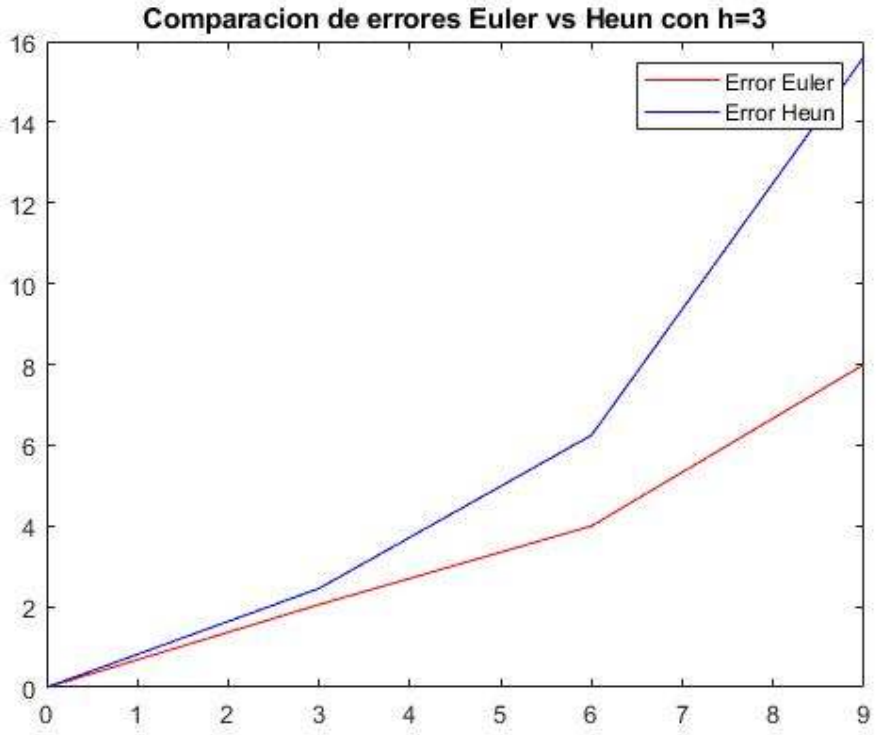
Comparativa de errores de ambos métodos






Comparacion de errores Euler vs Heun con h=1
Error Euler
Error Heun
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0 1 2 3 4 5 6 7 8 9 10


Comparacion de errores Euler vs Heun con h=2
Error Euler
Error Heun
1.2
1
0.8
0.6
0.4
0.2
0
0 1 2 3 4 5 6 7 8 9 10



Comparando los errores del método de Heun con el de Euler, podemos notar que el error global del método de Heun es menor, esto se debe básicamente a que el algoritmo de Heun es de orden dos con lo cual el error Global cometido esta dado por h al cuadrado y a medida que elegimos un paso más pequeño, este también se vuelve más pequeño.

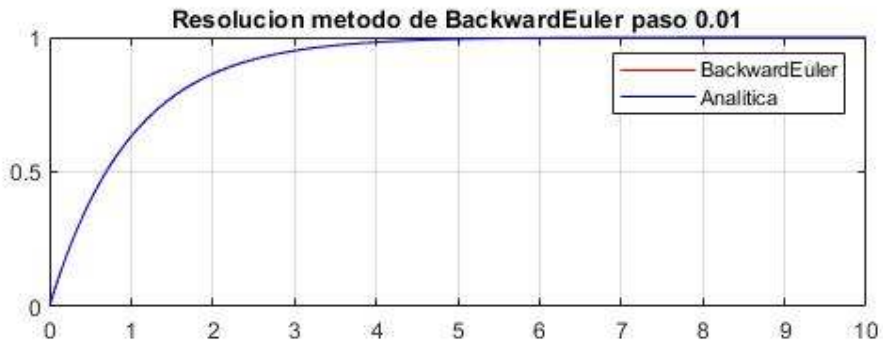
Al igual que en caso anterior, tenemos que analizar la estabilidad numérica para evitar que la solución se inestable. Operando con los autovalores de la matriz evolución podemos arribar a la ecuación de estabilidad del método y de esta manera elegir el tamaño de paso adecuado que conlleve a una solución estable. Como podemos observar en las gráficas, un tamaño de paso de 2 y de 3 conlleva a una solución inestable.

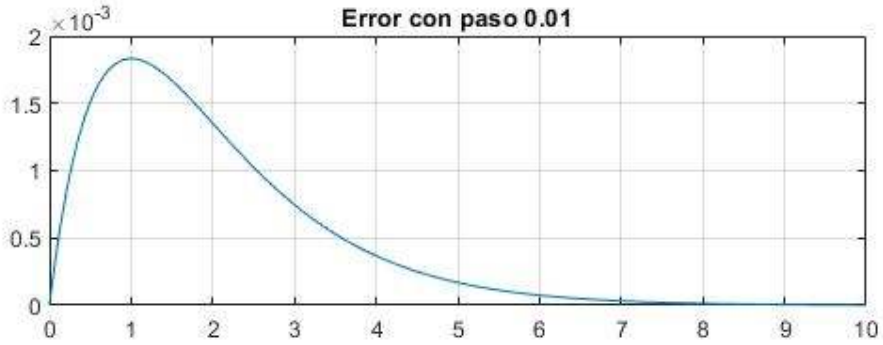
Método de Backward Euler

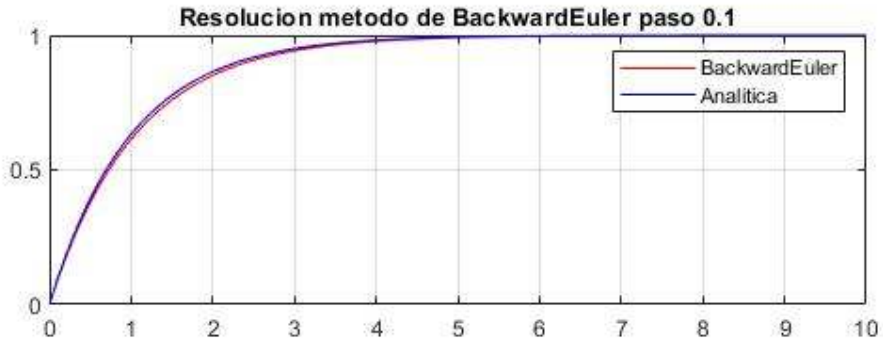
En este apartado se programaron nuevamente una función y un script, “BackwardEuler.m” y “ejercicio2_3.m” respectivamente.

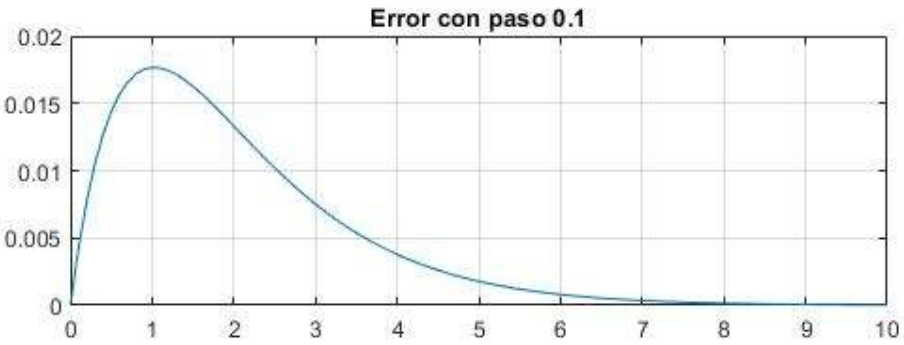
La función resuelve el la EDO del sistema mediante el método de Backward Euler, y el script realiza lo mismo que el “ejercicio2_1.m” agregando un ploteo extra donde se muestra la diferencia entre los errores de cada método utilizado anteriormente.

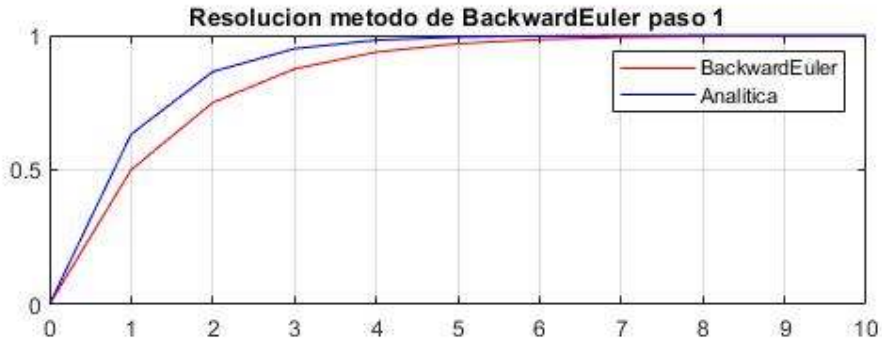
Las imágenes a continuación están ordenadas de la siguiente manera: primero las que comparan el resultado del método con la analítica y grafican el error relativo. Segundo, las que grafican la comparativa de los errores de todos los métodos utilizados anteriormente.

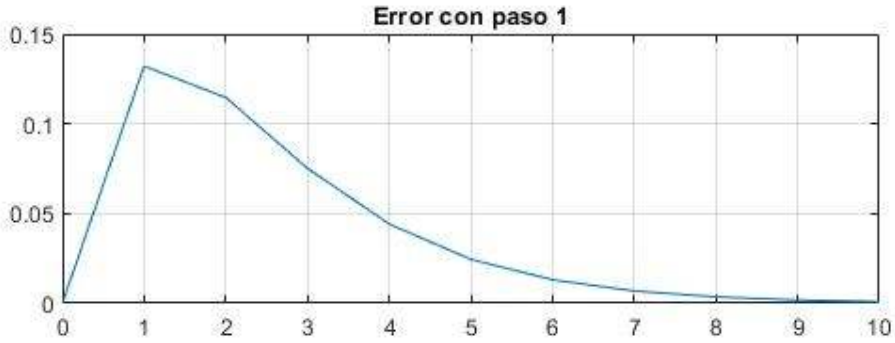

Resolucion metodo de BackwardEuler paso 0.01
BackwardEuler
Analitica
0
0.5
1
0
1
2
3
4
5
6
7
8
9
10

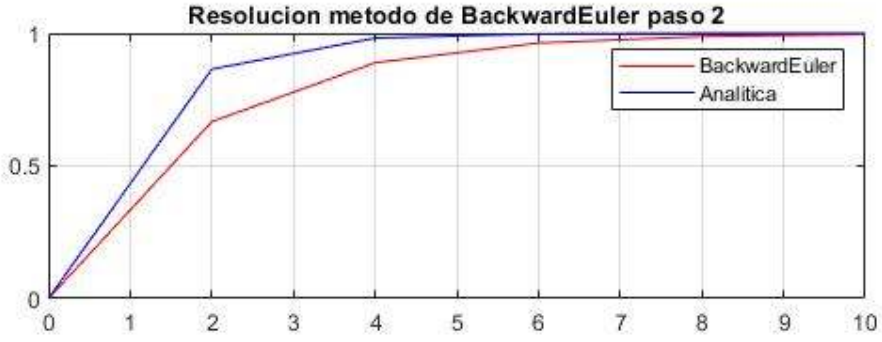

Error con paso 0.01
×10^-3
0
0.5
1
1.5
2
0
1
2
3
4
5
6
7
8
9
10

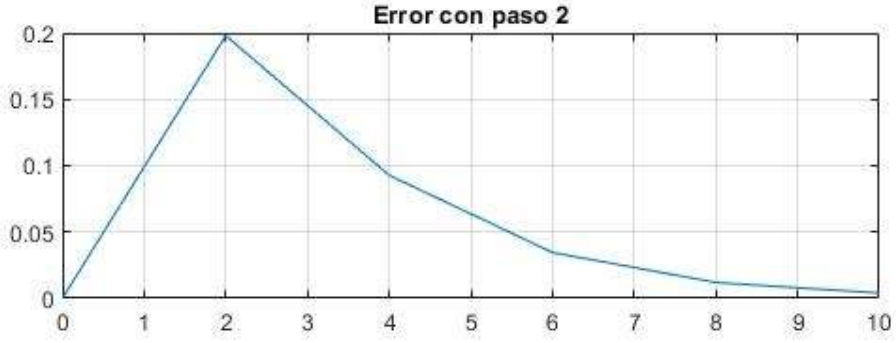

Resolucion metodo de BackwardEuler paso 0.1
BackwardEuler
Analitica
0
0.5
1
0
1
2
3
4
5
6
7
8
9
10

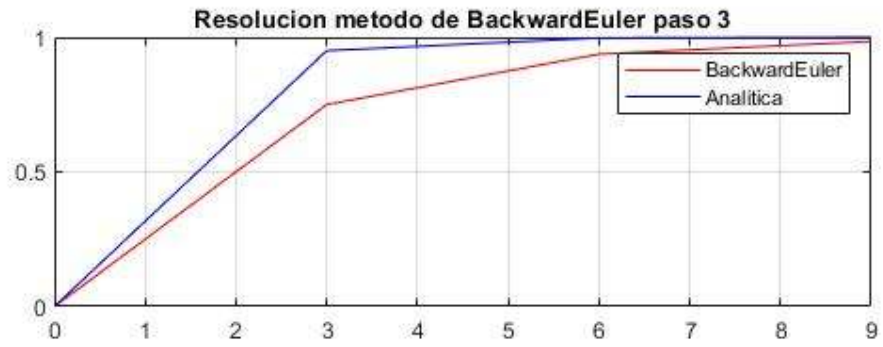

Error con paso 0.1
0
0.005
0.01
0.015
0.02
0
1
2
3
4
5
6
7
8
9
10

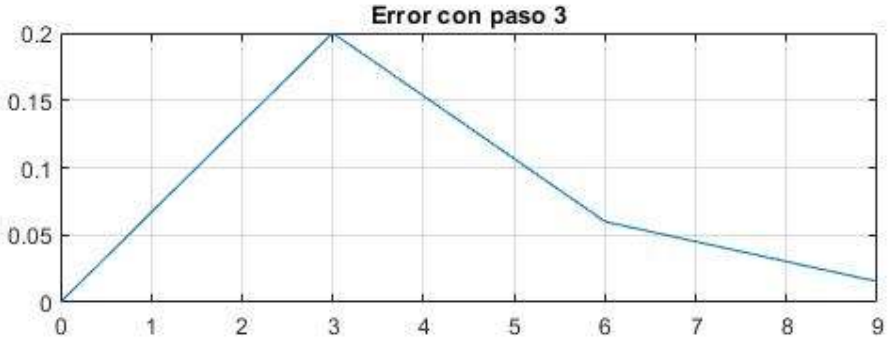

Resolucion metodo de BackwardEuler paso 1
BackwardEuler
Analitica
0
0.5
1
0 1 2 3 4 5 6 7 8 9 10


Error con paso 1
0
0.05
0.1
0.15
0 1 2 3 4 5 6 7 8 9 10

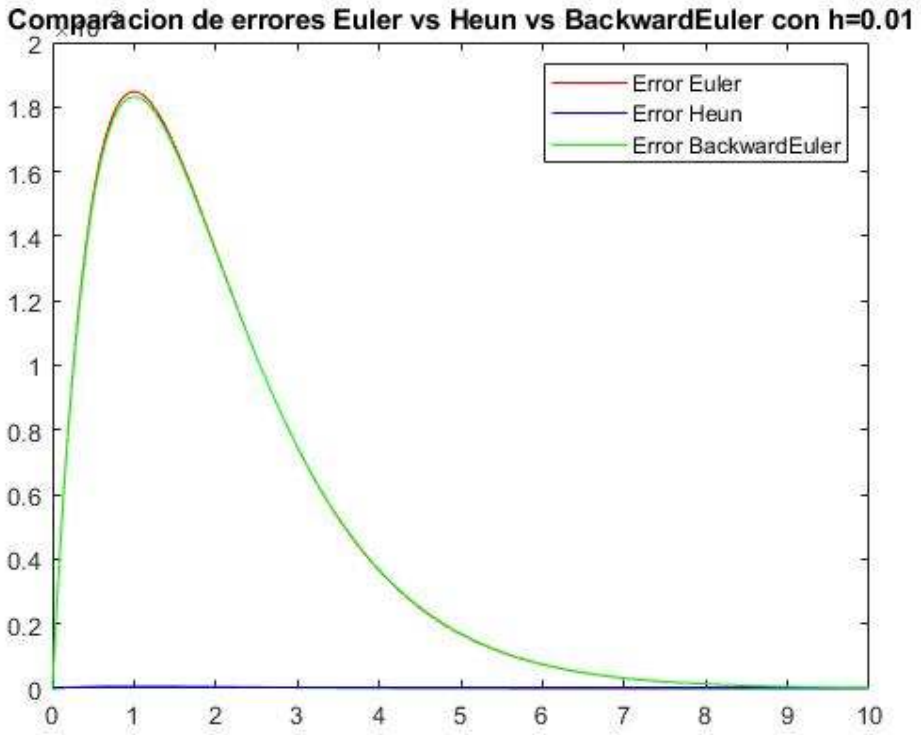

Resolucion metodo de BackwardEuler paso 2
BackwardEuler
Analitica
0
0.5
1
0 1 2 3 4 5 6 7 8 9 10

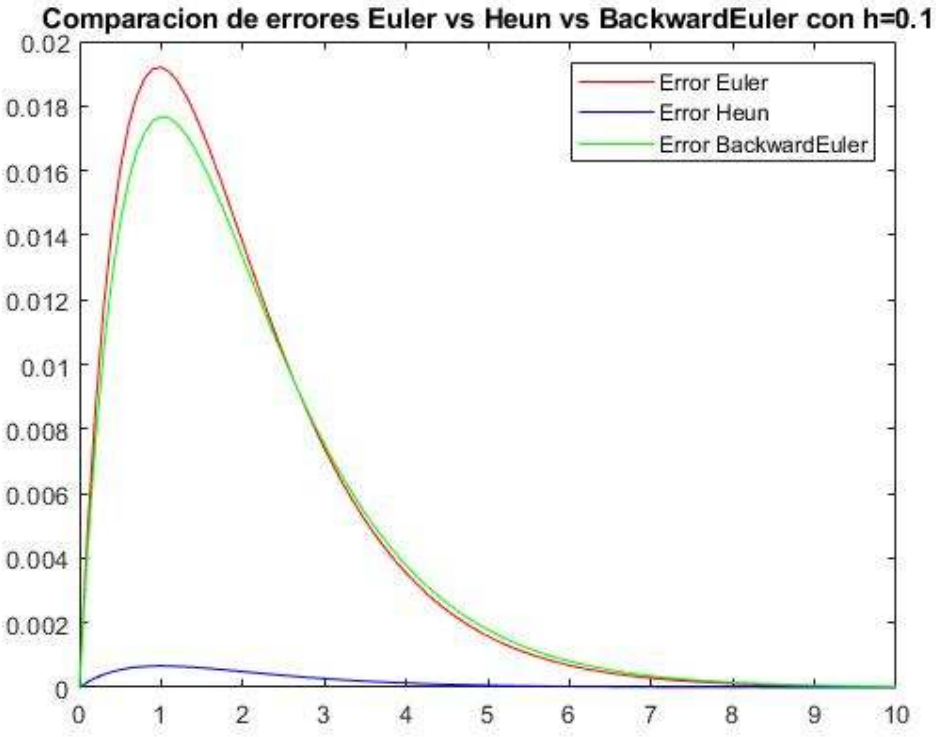

Error con paso 2
0
0.05
0.1
0.15
0.2
0 1 2 3 4 5 6 7 8 9 10

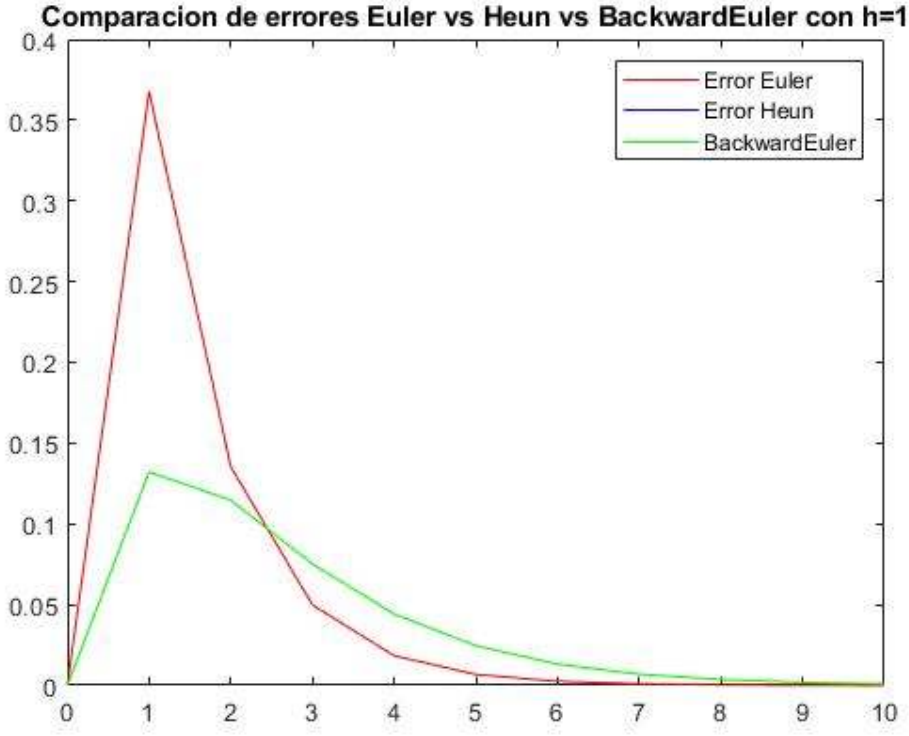


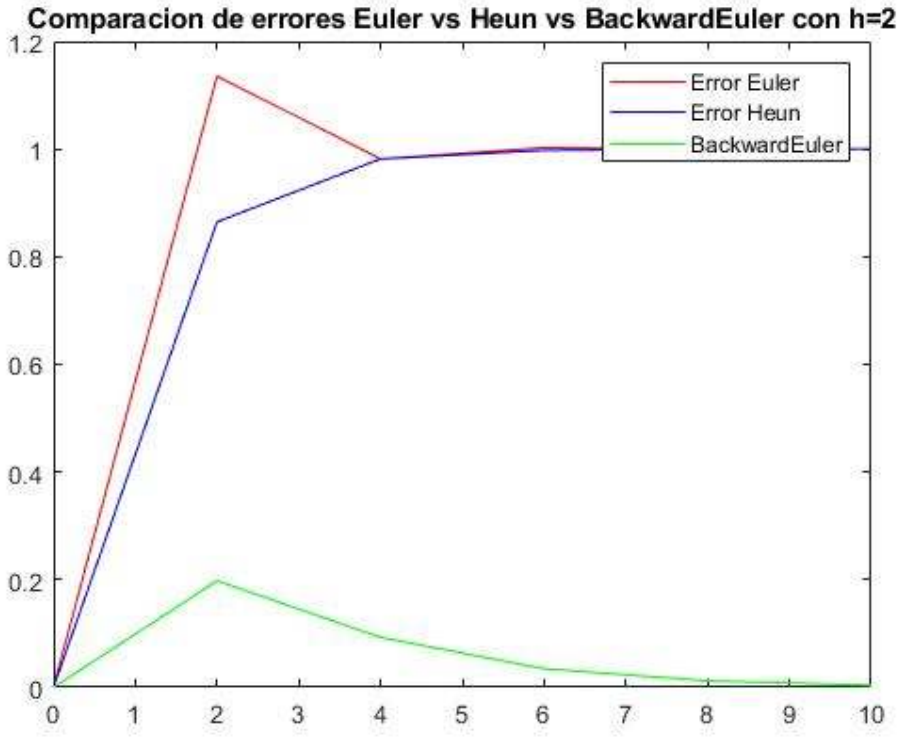


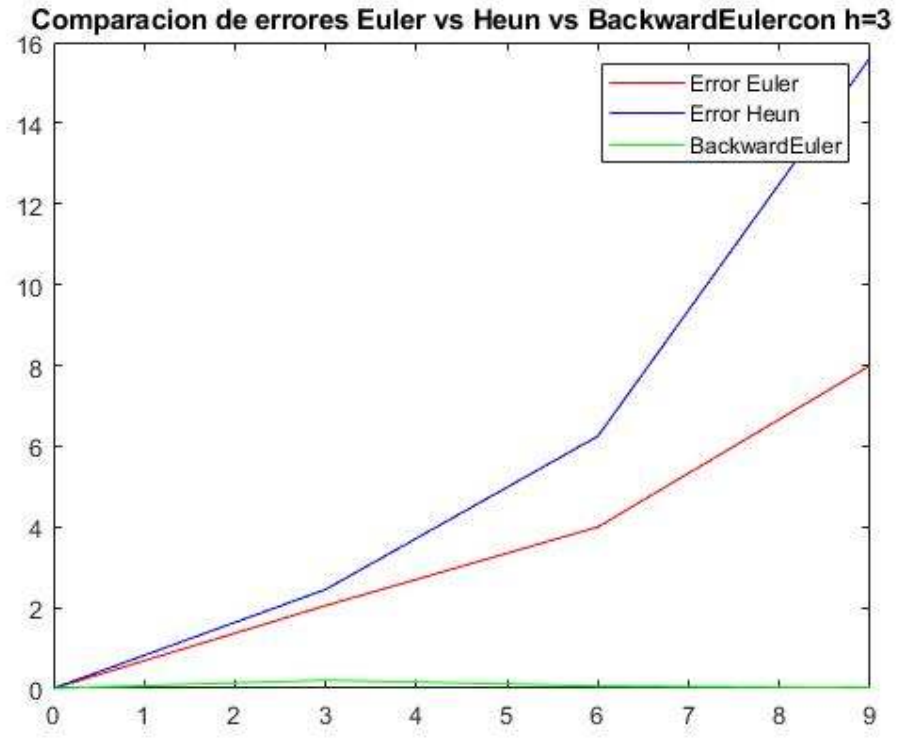
Comparativa de errores de todos los métodos utilizados




Comparacion de errores Euler vs Heun vs BackwardEuler con h=0.1
Error Euler
Error Heun
Error BackwardEuler
0.02
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
0 1 2 3 4 5 6 7 8 9 10


Comparacion de errores Euler vs Heun vs BackwardEuler con h=1
Error Euler
Error Heun
BackwardEuler
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0 1 2 3 4 5 6 7 8 9 10





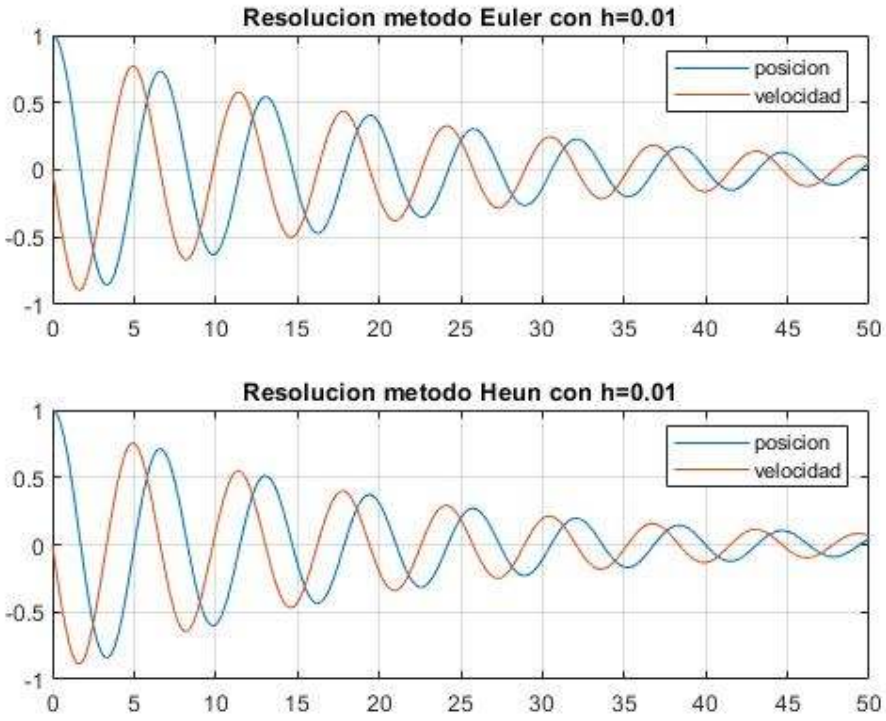
El algoritmo de Backward Euler es un algoritmo implícito de orden 1, con lo cual el error cometido es proporcional al paso h elegido, como podemos apreciar en las gráficas, el error cometido en por el método de Euler y el método de Backward Euler son muy parecidos y superiores al cometido por el método de Heun siempre que el

paso se mantenga en valores pequeños. La principal diferencia se aprecia cuando el paso asume valores muy grandes, esto se debe a que el algoritmo de Backward Euler es estable para cualquier valor de paso. Sin embargo, esto a veces no es deseable ya que, bajo las condiciones adecuadas puede llevar a una solución estable de un sistema que es inestable.

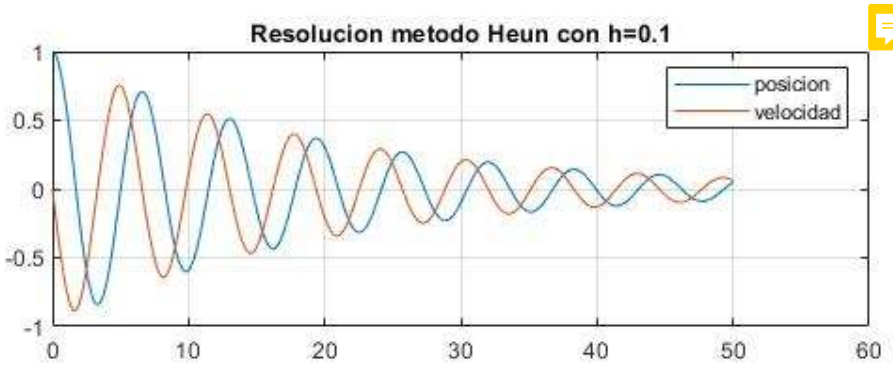
Métodos de Integración Para EDOs Generales

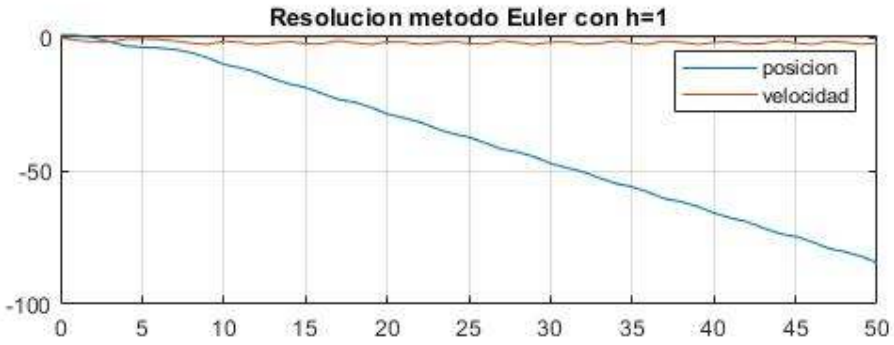
En este apartado se programaron tres funciones, “euler1.m”, “heun2.m” y “pendulo.m”, y un script, “ejercicio2_4.m”. Las funciones “euler1.m” y “heun2.m” implementan el método de Euler y de Heun de manera general, mientras que la función “pendulo.m” implementa la ecuación que representa al sistema, de manera que llamando a la función euler1 o heun2 y pasándole con como argumento la función péndulo.m, el tamaño del paso de integración, las condiciones iniciales y el tiempo de integración podemos obtener la solución de nuestro sistema. Esto se realizó, ya que normalmente es la manera de proceder en el análisis de sistemas.

Se programo un script, “ejercicio2_4.m”, que ploteo las respuestas con cada uno de los métodos, para distintos tamaños de paso, comparándolas entre sí, y los resultados obtenidos son los siguientes.




Resolucion metodo Euler con h=0.1
posicion
velocidad
1
0.5
0
-0.5
-1
0
10
20
30
40
50
60


Resolucion metodo Heun con h=0.1
posicion
velocidad
1
0.5
0
-0.5
-1
0
10
20
30
40
50
60


Resolucion metodo Euler con h=1
posicion
velocidad
0
-50
-100
0
5
10
15
20
25
30
35
40
45
50

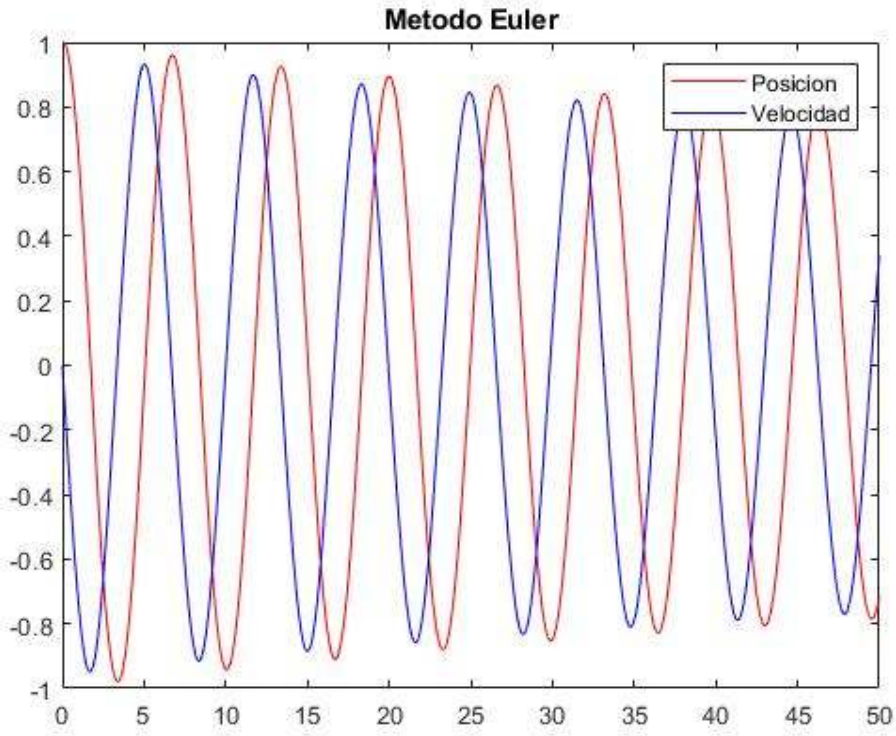

Resolucion metodo Heun con h=1
posicion
velocidad
2
1
0
-1
-2
0
5
10
15
20
25
30
35
40
45
50

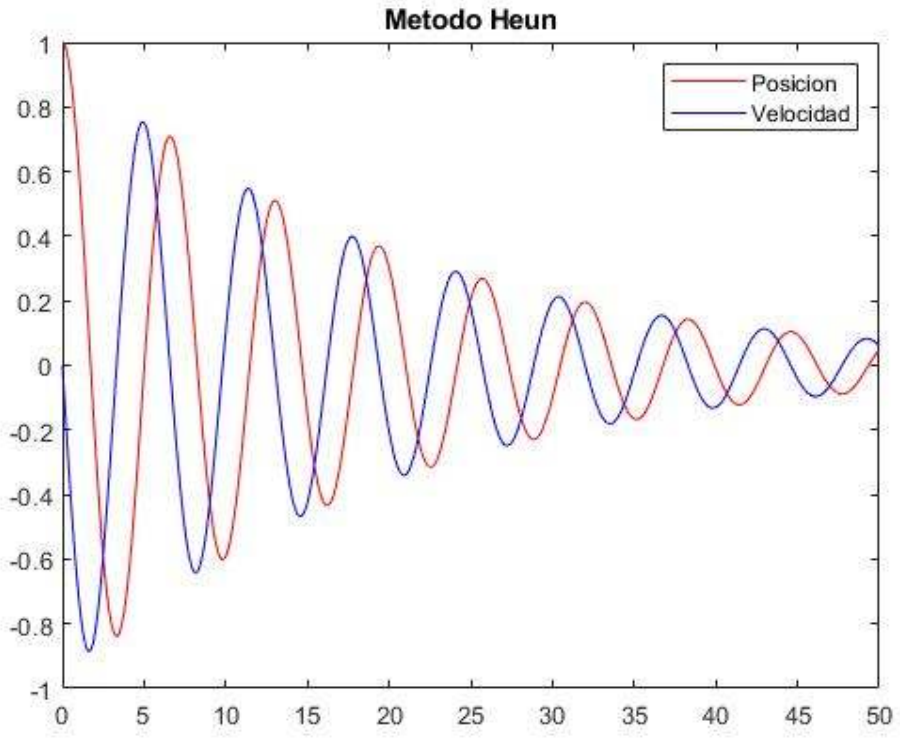
Se puede apreciar que para pasos chicos(h=0.01), la solución con el método de Euler y con el método de Heun son aproximadamente iguales, sin embargo cuando el paso aumenta a h=0.1 el método de Euler presenta una amortiguación menor que el método de Heun, y una tendencia mas marcada a la oscilación. Cuando el paso es 1 el de Euler se desestabiliza mientras que el de Heun comienza a oscilar llevando a una solución incorrecta carente de sentido en un sistema físico pendular con amortiguamiento como el que se está analizando. Claramente, para este caso el método de Euler no es conveniente de utilizar, salvo que usemos pasos de integración muy pequeños, debido a sus características propias de amortiguamiento bajo y por tanto tendencia a oscilar. El método de Heun debe solo utilizarse con un paso de integración pequeño.

Métodos de Paso Fijo

Para este apartado se realizó en simulink el Diagrama en Bloques del péndulo del ejercicio anterior y se simuló dicho sistema con un paso de integración de 0.1, un tiempo de simulación de 50seg y utilizando el método de Heun(ode2) y el método de Euler(ode1).

Se programó el script “ejercicio3_1.m” y se creo el modelo en simulink “pendulosimulink.slx”, los resultados obtenidos son los siguientes.





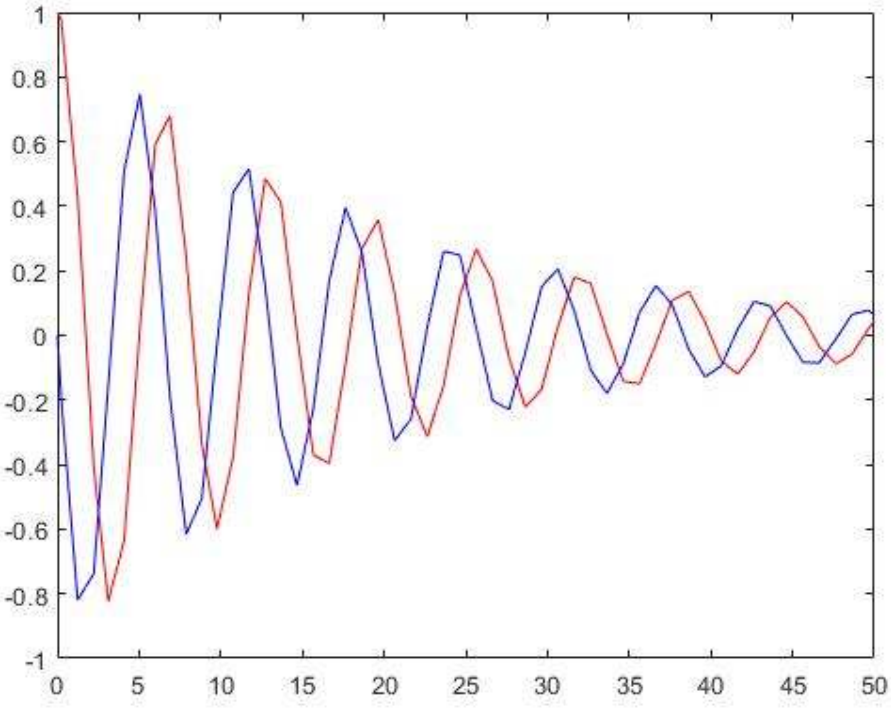
Se puede observar que se obtienen los mismos resultados, con un paso de 0.1 el método de Euler no es apto para resolver este sistema.

Métodos de Paso Variable

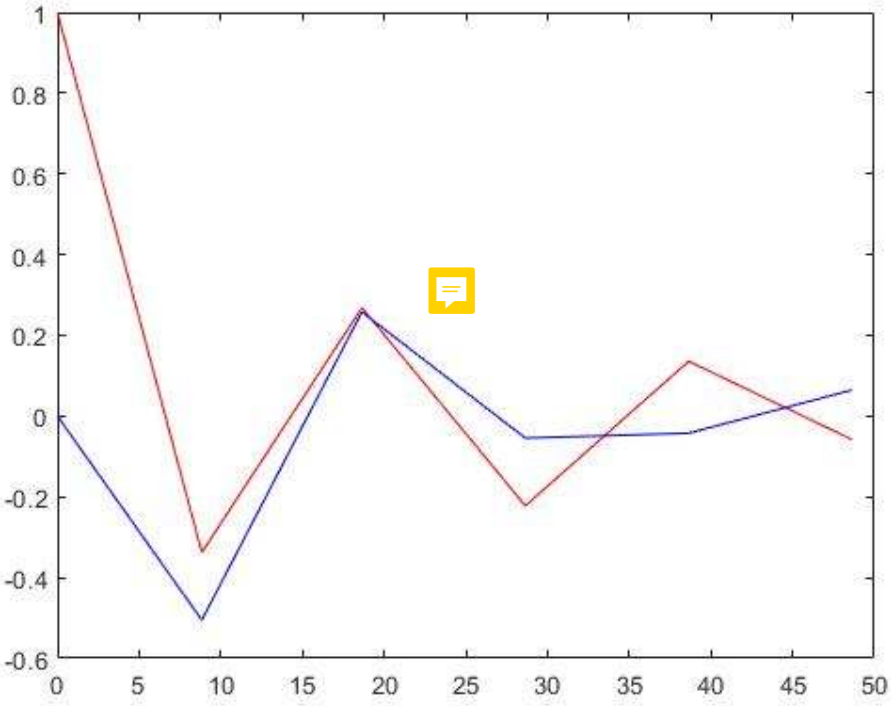
En esta sección se implementaron métodos de paso variable simulando nuevamente el sistema del péndulo esta vez con 2 métodos, ode45 (método explícito) y ode15s (método implícito) utilizando tolerancias relativas y absolutas de 0.001.

Luego se programó un script, “ejercicio3_2.m” que calcula la cantidad de pasos utilizados en la simulación, para este caso arrojo un número de 53 pasos, y grafica los resultados con el método ode45, luego, modificando el método de integración en el sistema del péndulo, y corriendo el script, se obtiene la gráfica de la simulación con ode15s.

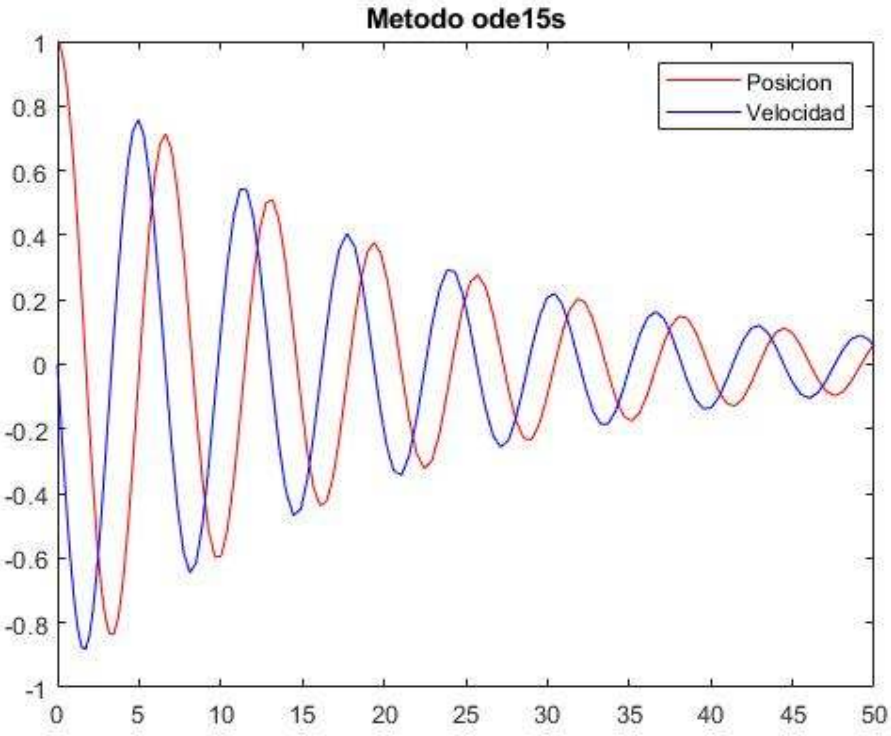
Las gráficas obtenidas se muestran a continuación.



En la siguiente grafica se aplica una dezimación de 10 en la salida del bloque “To Workspace” del modelo en simulink.



La siguiente Gráfica es con el método ode15s

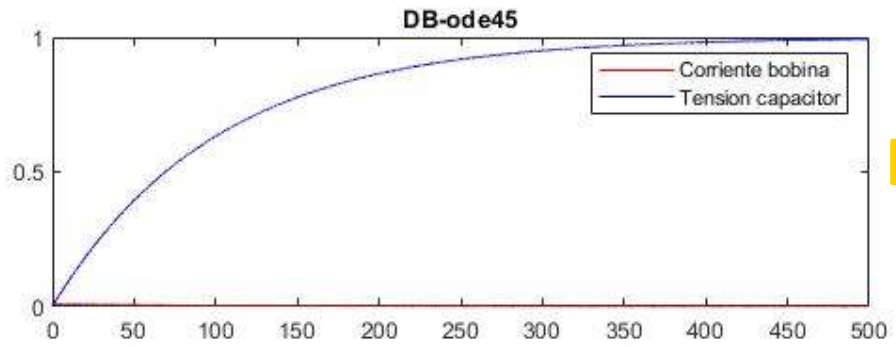


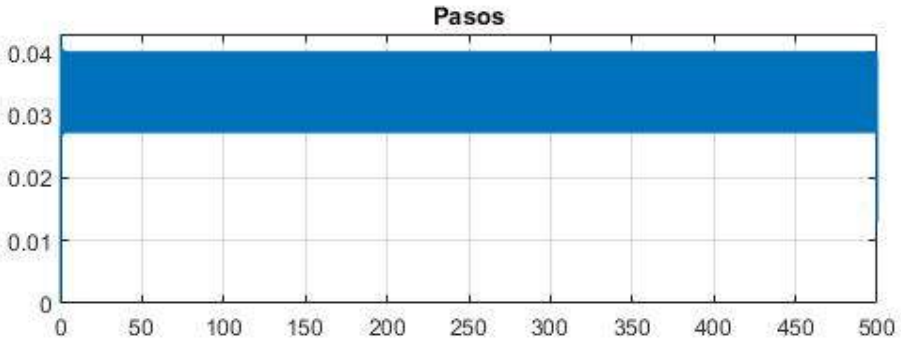
La diferencia entre ambos métodos radica en la cantidad de pasos realizados por cada uno. El ode45 realizo un total de 53 pasos mientras que el ode 15s realizo un total de 125, mas del doble, con lo cual se obtiene una grafica mas suave y mas precisa. Para este caso en particular, dado que el sistema es oscilante, conviene utilizar el ode 15s ya que entrega una respuesta mas detallada de la dinámica del sistema.

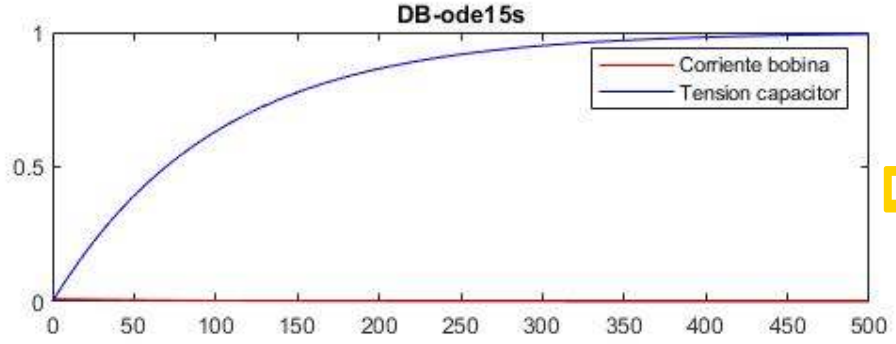
Sistemas Stiff

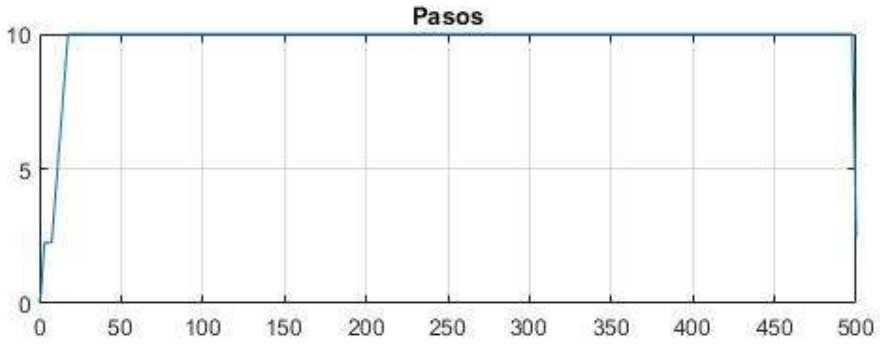
En este apartado se armo en simulink el DB de un circuito RLC (“ejercicio3_3_2_modelo.slx” y “ejercicio3_3_3_modelo.slx”), para lo cual se utilizo el DB obtenido en clases, luego, se programo un script(“ejercicio3_3_1.m”) que calcula los autovalores de la matriz de evolución y luego calcula el máximo paso de integración que puede utilizarse para el método de Euler y que arroje un resultado estable. Luego, se ejecuta la simulación del modelo con un ode45 primero y con un ode15s posteriormente y se grafican las respuestas y la evolución de los pasos, ya que ambos métodos son de paso variable.

Las gráficas se corresponden primero con la resolución con el ode45 y luego con el ode15.


DB-ode45
Corriente bobina
Tension capacitor
1
0.5
0
0
50
100
150
200
250
300
350
400
450
500


Pasos
0.04
0.03
0.02
0.01
0
0
50
100
150
200
250
300
350
400
450
500


DB-ode15s
Corriente bobina
Tension capacitor
1
0.5
0
0
50
100
150
200
250
300
350
400
450
500


Pasos
10
5
0
0
50
100
150
200
250
300
350
400
450
500

Este sistema presenta dos dinámicas muy marcadas, en concreto, posee un polo rápido y un polo lento, se puede observar claramente que le polo lento aparece en la tensión del capacitor, mientras que el polo rápido en la corriente de la bobina.

Calculando los autovalores de la matriz evolución obtenemos que estos son λ=-0.01 y λ=-99.99 lo cual indica que hay in polo rápido en aproximadamente -100.

Este polo rápido impide que el método ode45 eleve el paso por encima de 0.04, ya que tornaría al sistema inestable, con lo cual el algoritmo obtiene un error grande que hace achicar nuevamente el paso.

Como podemos ver, utilizando el ode15s (método implícito), que es un método que no se desestabiliza, la cantidad de pasos se reduce drásticamente y podemos apreciar que la longitud del paso aumenta rápidamente luego de que la dinámica rápida desaparece.

Si analizamos el tamaño del paso de los dos algoritmos podemos observar que el ode45 nunca supero los 0.05 de tamaño de paso, es más, siempre oscilo entre valores inferiores a 0.03 y apenas por encima de 0.04, mientras que el ode15s rápidamente salta a pasos de longitud 10.