Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio

Máster Universitario en Sistemas Espaciales

Milestone 4: Problemas lineales. Regiones de estabilidad absoluta

Ampliación de Matemáticas I:
Cálculo numérico

02 de diciembre de 2022

Autor:

- Daniel Cerón Granda

ÍNDICE

PÁGINA

1.Introducción

Hay tres objetivos principales en la realización de este trabajo. En primer lugar, la obtención de las soluciones numéricas para distintos esquemas para el problema del oscilador lineal. En segundo lugar, la obtención de las regiones de estabilidad absoluta de diferentes esquemas de integración. Finalmente, se realizará el análisis de las soluciones del problema del oscilador lineal a través de las regiones de estabilidad absoluta que se obtuvieron previamente. Además, se incluye un esquema que muestra las relaciones funcionales en el programa utilizado para la consecución de estos objetivos.

2.Problema del oscilador lineal

El oscilador lineal armónico es aquel que, cuando se deja en libertad fuera de su posición de equilibrio, vuelve hacia ella realizando oscilaciones armónicas (periódicas) fuera de su posición de equilibrio. Si hay amortiguamiento, es un problema asintóticamente estable (tiende a la posición de equilibrio pasado suficiente tiempo) y si no lo hay, es estable (no tiende a la posición de equilibrio, pero sí que la distancia a la posición de equilibrio está acotada cuando el tiempo tiende a infinito). Se analizará el caso de oscilador lineal sin amortiguamiento por una mayor simplicidad. La ecuación diferencial que representa este problema es la siguiente:

$$\ddot{x} + x = 0$$

Ecuación 1: Ecuación diferencial del oscilador armónico

Que se integrará para las condiciones iniciales de velocidad nula y posición unidad positiva: x (0) = 1. Para estas condiciones iniciales, la solución analítica es la función cosenoidal:

$$x = \cos(t)$$

Ecuación 2: Solución analítica del problema del oscilador armónico para las condiciones iniciales elegidas

Cuya representación gráfica para un tiempo de 200 segundos es:

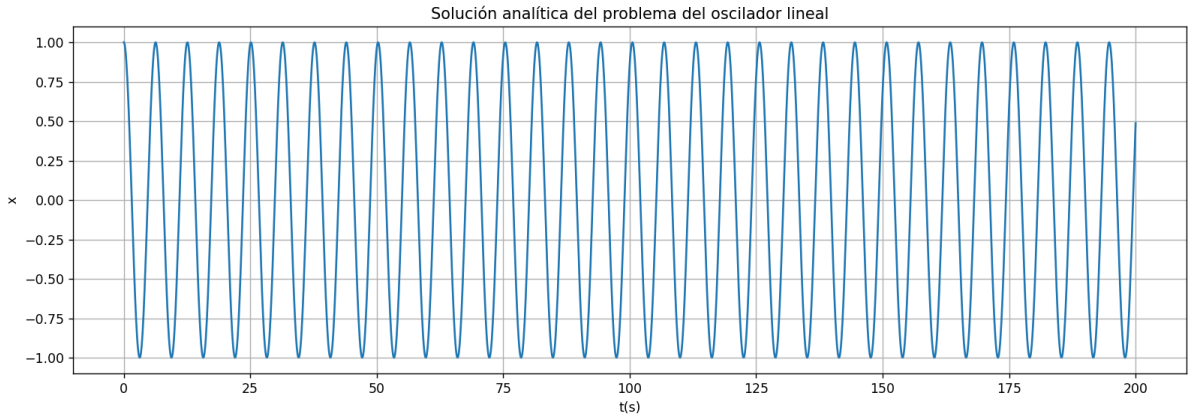

Figura 1: Representación gráfica de la solución analítica del problema del oscilador lineal

Para su resolución numérica, esta ecuación diferencial de orden 2 se transforma en un sistema de dos ecuaciones diferenciales ordinarias definiendo las variables y_1 como igual a x e y_2 como igual a la derivada temporal de x, resultando en el siguiente sistema:

$$\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\end{pmatrix} = \begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix} * \begin{pmatrix}y_1\\ y_2\end{pmatrix}$$

Ecuación 3: Sistema de ecuaciones diferenciales ordinarias resultante a resolver numéricamente

Los esquemas numéricos utilizados en el siguiente apartado deben resolver este sistema de ecuaciones diferenciales ordinarias.

3. Integración del oscilador lineal utilizando distintos esquemas

En el código, el sistema de ecuaciones diferenciales ordinarias de la ecuación 3 se ha implementado mediante la siguiente función, que se almacena en el módulo de "Funciones_a_integrar" y se importa y posteriormente utiliza en el programa principal:

```
13  def Linear_oscillator(U,t):
14      x = U[1]
15      y = -U[0]
16      return array([x,y])
17
```

Figura 2: Función del oscilador lineal utilizada en el programa

Que al utilizarla como F en la función del problema de Cauchy vista en los milestones anteriores integra numéricamente el problema del oscilador lineal.

En el mismo módulo, se incluye una función "wrapped" para la representación y edición de los ejes y título de la solución del oscilador lineal para cualquier esquema:

```
19  def Wrapped_Repr_Linear_Oscillator(t, x, Scheme):
20      plt.plot(t,x)
21      plt.xlabel("t")
22      plt.ylabel("x")
23      i = "Oscilador lineal con esquema: "+Scheme.__name__
24      plt.title(i)
25      plt.show()
26      return i
```

Figura 3: Función para la representación del oscilador lineal

Introduciendo un vector de tiempos y un vector de posiciones para esos tiempos, además del esquema numérico para el que se ha obtenido la solución numérica, devuelve la gráfica del problema del oscilador lineal.

En el programa principal, en primer lugar, se definen las condiciones iniciales para las que se va a integrar el problema:

```
#APARTAD0 1: Integrar el oscilador lineal con varios esquemas
#Definición de condiciones iniciales
n = 2000 # Número de particiones
deltat = 0.1 #Intervalo de tiempo entre iteraciones
U0 = array([1,0]) #Condiciones iniciales
matrizU = array(zeros([n,4])) #Matriz que contendrá en sus columnas los valores de las variables, cada fila es un paso de tiempo
F0 = array([0,-1])#Valor de F en el instante inicial
U = zeros(2)
x = zeros(n)
Esquemas = [Euler, Euler_inverso, CN, RK4]
```

Figura 4: Condiciones iniciales para el problema del oscilador lineal

Se definen los valores iniciales del intervalo de tiempo, número de particiones en las que se va a iterar (se obtiene un tiempo final de 200s) y las condiciones iniciales de U y F para la integración del problema de Cauchy. Posteriormente, se realiza la representación gráfica de la solución analítica del problema del oscilador lineal, que ya se vio en la figura 1, con el fin de poder comparar más fácilmente con las soluciones de los diferentes esquemas:

```
#Solución analítica
t = linspace(0, n*deltat, n)
x = array(zeros(len(t)))
for i in range (len(t)):
    x[i] = math.cos(t[i])
    continue
plt.plot(t,x)
plt.xlabel("t(s)")
plt.ylabel("x")
plt.title("Solución analítica del problema del oscilador lineal")
plt.grid()
plt.show()
```

Figura 5: Programación de la representación de la solución analítica del problema del oscilador lineal

Posteriormente, para los esquemas de Euler, Euler inverso, Crank-Nicholson y Range-Kutta de orden 4, se realiza la integración del esquema y la representación gráfica:

```
for j in Esquemas:
    matrizU = Cauchy_problem(j, U0, n, deltat, Linear_oscillator, F0, 2)
    x = matrizU[:,0]
    Wrapped_Repr_Linear_Oscillator(t, x, j)
    continue
```

Figura 6: Integración y representación de la solución del problema del oscilador lineal para los diferentes esquemas

Se puede observar que la utilización de la función del problema de Cauchy y la función “wrapped” vista en la figura 3 permite que, con un simple bucle, se realicen todas las integraciones y representaciones gráficas para los 4 esquemas numéricos en tan solo 5 líneas, ahorrando una gran cantidad de líneas de código en el programa principal.

Finalmente, como el esquema Leap Frog tiene unas particularidades propias (que se comentarán en la sección e de este punto), la función de integración del problema de Cauchy es ligeramente diferente del resto de esquemas, por lo que hay que llamarla aparte:

```
matriz_U1 = Cauchy_problem(Euler, U0, 10, deltat, Linear_oscillator, F0, 2)
U1 = matriz_U1[0,:]
matrizU = Cauchy_problem_Leap_Frog(Leap_Frog, U0, n, deltat, Linear_oscillator, U1, 2)
x = matrizU[:,0]
Wrapped_Repr_Linear_Oscillator(t, x, Leap_Frog)

```

Figura 7: Integración y representación de la solución del problema del oscilador lineal para el esquema Leap Frog

Por otro lado, la representación se realiza de igual forma con la función “wrapped”, reduciendo una vez más la cantidad de líneas de código del programa.

a) Esquema Euler

La solución del problema del oscilador lineal utilizando el esquema de Euler es la siguiente:

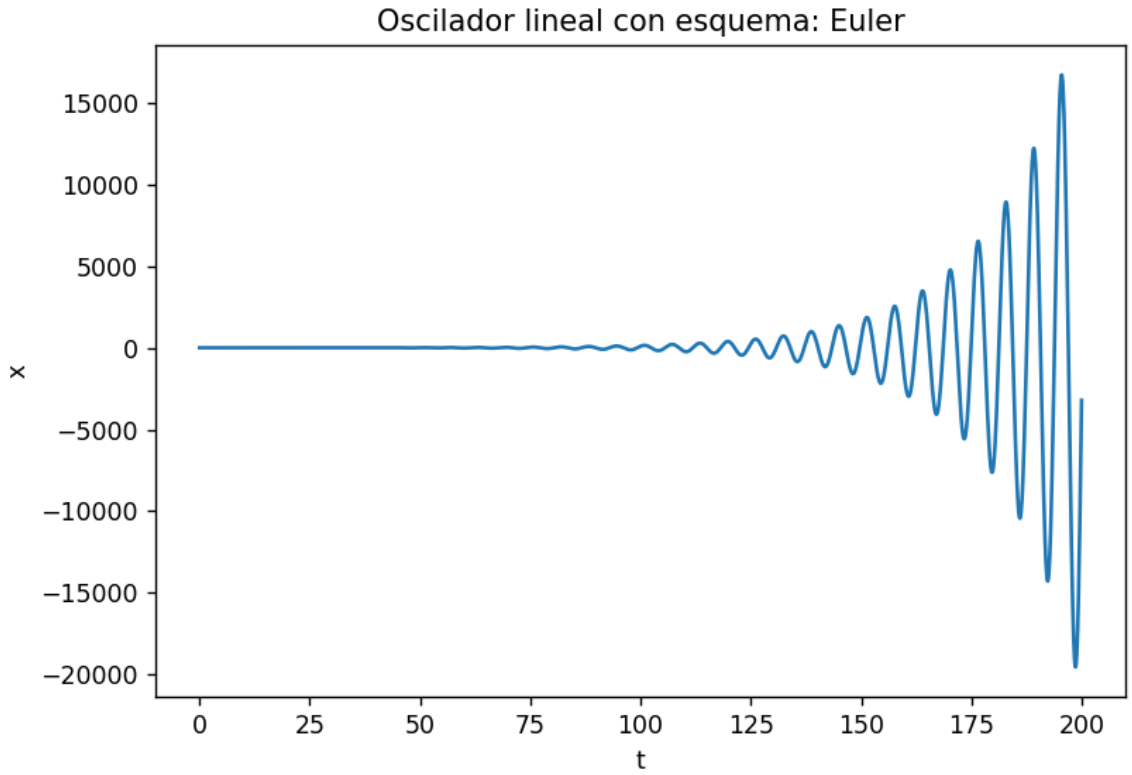

Figura 8: Representación del oscilador lineal integrado con el esquema Euler

Donde se ve que el esquema diverge con el tiempo rápidamente, alcanzando unas desviaciones muy grandes en poco más de 3 minutos de integración. Es obvio que este esquema no es nada adecuado, en ninguna circunstancia, para la integración del oscilador lineal.

b) Euler inverso

La solución del problema del oscilador lineal utilizando el esquema Euler inverso es la siguiente:

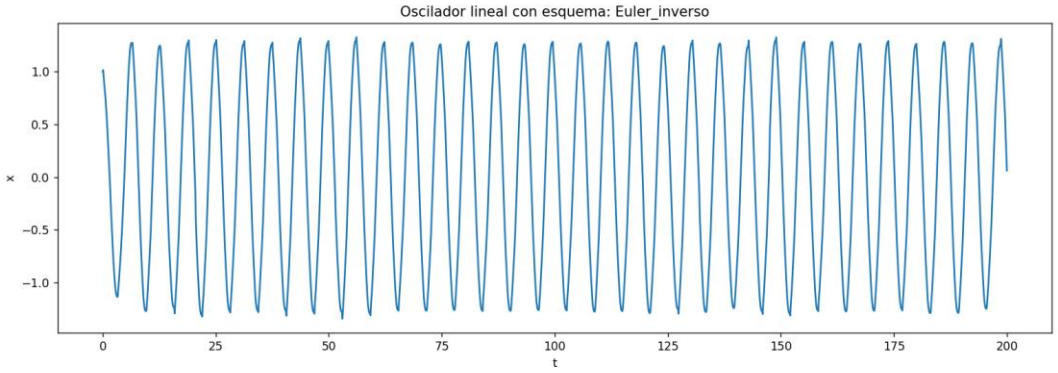

Figura 9: Representación del oscilador lineal integrado con el esquema Euler inverso

Se observa que la solución se desvía respecto de la analítica ligeramente al principio, alcanzando valores ligeramente superiores, para luego permanecer relativamente estable cuando el tiempo crece en torno a este valor máximo que estará en el entorno de 1,2-1,3; variando en cada máximo y mínimo ligeramente. Es un esquema claramente mejor que el de Euler para el análisis de este problema, pero es preferible el uso de otros esquemas que no den un error del orden del 10% respecto a la solución analítica. Un aspecto destacable de esta solución es el error de fase que se produce, ya que en t=200s se puede ver que la gráfica está decreciendo y cerca de 0, mientras que en la solución analítica debería estar en el máximo. En Euler el error de fase era menor, pero evidentemente no compensa para nada el grandísimo error en el módulo que se produce.

c) Crank-Nicholson

La solución del problema del oscilador lineal utilizando el esquema Crank-Nicholson es la siguiente:

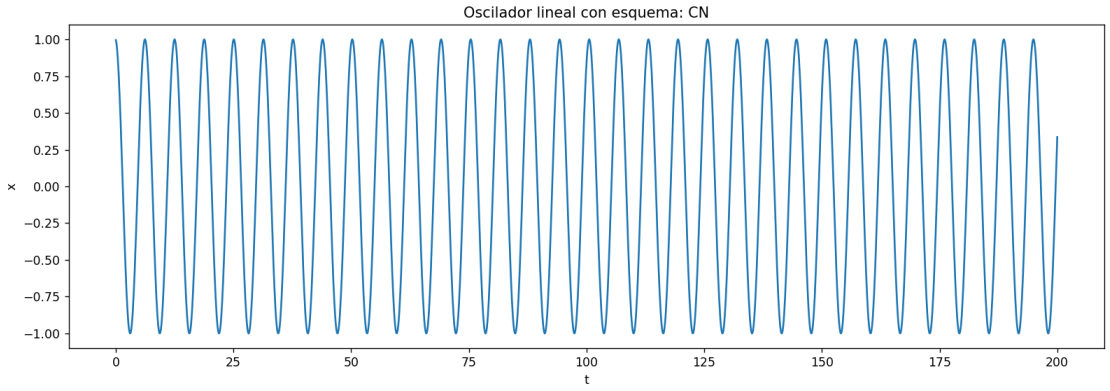

Figura 10: Representación del oscilador lineal integrado con el esquema Crank-Nicholson

Este esquema sí es adecuado para la integración de este problema, puesto que la solución ofrece un error de módulo que es prácticamente nulo, siendo el máximo de las oscilaciones 1 o un valor muy cercano. En cuanto al error de fase, se observa que, aunque esté presente, es mucho menor que el error de fase cometido en el Euler inverso, ya que, en t=200s, aunque la oscilación no esté llegando al máximo, sí que está en el entorno de 0,25; no habiendo sufrido un gran retraso respecto a la solución analítica, que está en 0,5. Por tanto, este es un esquema muy atractivo para realizar la integración numérica de este problema, ya que no es tan complejo como otros esquemas de órdenes superiores y ofrece unos resultados bastante aceptables.

d) Range-Kutta de orden 4

La solución del problema del oscilador lineal utilizando el esquema Range-Kutta de orden 4 es la siguiente:

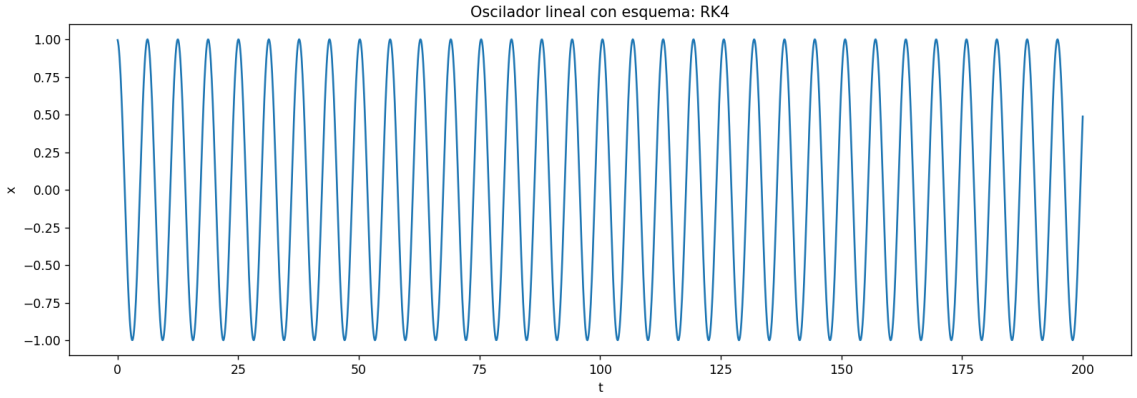

Figura 11: Representación del oscilador lineal integrado con el esquema RK4

Se puede observar que la solución obtenida con este esquema es muy buena, para el tiempo de integración escogido es prácticamente igual que la analítica, sin presentar apenas error ni en módulo ni en fase. Si el criterio más importante para realizar la integración es el nivel de precisión, este esquema sería el esquema elegido sin lugar a duda.

e) Leap Frog

Este esquema presenta la siguiente estructura:

$$U_{n+1} = U_{n-1} + 2 * \Delta t * F_n$$

Ecuación 4: Expresión de un paso del esquema Leap Frog

Siendo un esquema de orden 2. Es reseñable el hecho de que este esquema, además de la condición inicial U_0 necesita una segunda condición inicial, ya que, por la expresión que tiene, no puede calcular U_1. Esta segunda condición inicial se obtendrá aplicando el método de Euler, ya que su error local de truncación es de orden 2 (ya que siempre es uno más que el orden del esquema, el esquema pierde un orden de precisión precisamente por la acumulación a lo largo del tiempo del error local de truncación), y al ser el esquema de orden 2 mantiene la precisión del esquema (se podría poner una condición inicial U_1 más precisa con un RK4, pero su efecto tras un paso de tiempo grande acabará siendo prácticamente nulo, por lo que se opta por una mayor sencillez aplicando el esquema Euler).

Para el código, el empleo de esta segunda condición inicial hace que se haya optado por la creación de una función especial para aplicar el problema de Cauchy a este esquema:

```
#Usar exclusivamente para el esquema de Leap Frog
def Cauchy_problem_Leap_Frog(Temporal_scheme, U0, n, deltat, F, U1, d): #d = dimensión de U
    t=0
    c = int(d)
    matrizU = array(zeros([n,c]))
    for i in range (n):
        t = t+deltat
        U = Temporal_scheme(U0, deltat, t, U1, F)
        matrizU[i,:] = U
        U0 = U1
        U1 = U
        continue
    return (matrizU)
```

Figura 12: Función para la aplicación del problema de Cauchy al esquema Leap Frog

Es una función muy similar a la del problema de Cauchy, pero adaptada a la existencia de dos condiciones iniciales y dos pasos diferentes para U_{n-1} y para F_n.

La solución del problema del oscilador lineal utilizando el esquema Leap Frog es la siguiente:

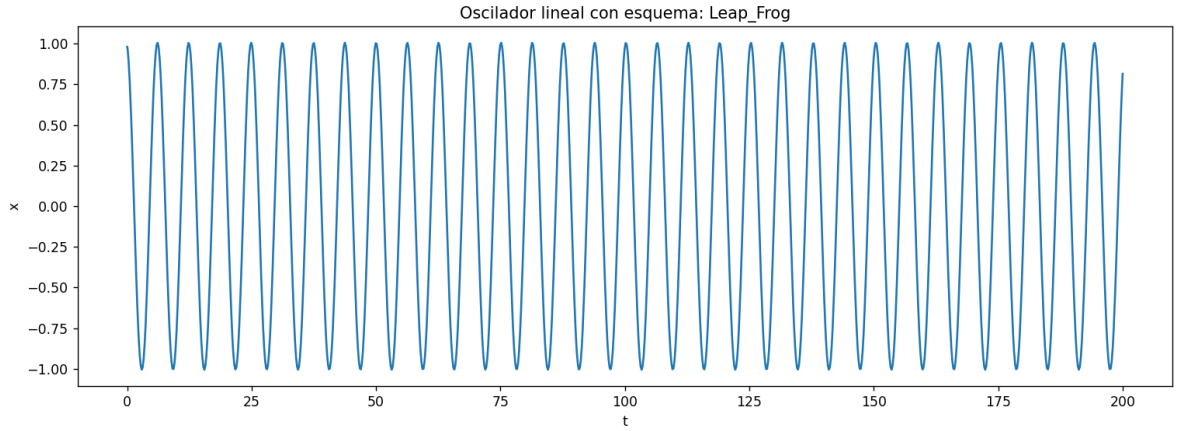

Figura 13: Representación del oscilador lineal integrado con el esquema Leap Frog

Se observa que hay un ligero crecimiento del máximo de la función, de forma similar a lo que ocurría en el esquema Euler inverso, pero con una escala mucho menor, por lo que los resultados pueden considerarse aceptables. Hay un error de fase del mismo orden que en el esquema Euler inverso, pero en esta ocasión la solución numérica va más adelantada respecto a la analítica, al contrario de lo que ocurría en los esquemas previos que presentaban error de fase. En conclusión, es un esquema aceptable para realizar la integración de este problema, pero el Leap Frog, un esquema del mismo orden y seguramente un tiempo de computación similar, presenta mejores resultados que él, y si lo que importa es la precisión, el RK4 es muy superior también, por lo que, si las otras opciones son posibles, las elegiría antes que este esquema.

4.Regiones de estabilidad absoluta

Las regiones de estabilidad absoluta son las zonas del plano complejo en las que el polinomio característico de un esquema tiene módulo menor que la unidad. El polinomio característico de estabilidad es un polinomio particular de cada esquema numérico que permite, a través del teorema

espectral, pasar de los autovalores de un polinomio a los autovalores de un polinomio de matrices. Se define un parámetro r, que es el módulo de los autovalores del sistema numérico estudiado. Si todas las raíces del polinomio característico son menores que la unidad, la ganancia en cada iteración es menor que la unidad y el esquema es estable para el problema estudiado. Llamando w = λ*Δt, se puede controlar el valor de los autovalores a través del paso de tiempo para asegurar que el esquema se encuentre en la región de estabilidad.

La obtención de las regiones de estabilidad absoluta se consiguen a través del siguiente código:

```
48  #APARTADO 2: Regiones de estabilidad absoluta
49  Esquemas = [Euler, Euler_inverso, CN, RK4]
50  for i in Esquemas:
51      rho, Re, Im  = Stability_Region(i, n/10, -4, 4, -4, 4)
52      Wrapped_Stability_Zones(rho, Re, Im, i)
53      continue
54
55  rho, Re, Im  = Stability_Region(Leap_Frog, n/10, -1, 1, -1, 1)
56  Wrapped_Stability_Zones(rho, Re, Im, Leap_Frog)
```

Figura 14: Obtención de las regiones de estabilidad en el programa

Donde se ha definido una función para la definición de la región de estabilidad y otra función wrapped para la representación de las regiones de estabilidad. Este uso de funciones permite que la representación de la región de estabilidad se realice en menos de 10 líneas de código, ganando mucha eficiencia en la realización de este programa. Estas funciones se definen en un módulo aparte llamado Stability.

La función de la región de estabilidad es la siguiente:

```
7   #Región de estabilidad
8   def Stability_Region(Esquema, N, x0, xf, y0, yf):
9
10      M = int(N)
11      Re = linspace(x0, xf, M)
12      Im = linspace(y0, yf, M)
13      rho = array(zeros([M, M]))
14
15      for i in range(M):
16          for j in range(M):
17
18              w = complex(Re[i], Im[j])
19              r = Esquema( 1., 1., 0., w,lambda u, t:w*u)
20              rho[i, j] = abs(r)
21
22      return rho, Re, Im
```

Figura 15: Función de definición de la región de estabilidad de un esquema

En esta función se debe incluir un número de regiones que representar en la función de estabilidad, valores iniciales y finales de los ejes real e imaginario en los que representar la región de estabilidad y el esquema que se quiere estudiar. En la función, se definen los linspace de los valores inicial y final de los ejes real e imaginario con un número de particiones igual al número de regiones que se quiere representar. Después, para esos linspace se da a w, una variable compleja creada, los valores de esos linspace y se calcula el valor de r aplicando el esquema elegido a una ecuación de un solo paso (las condiciones iniciales se ponen 1, pero no tienen relevancia) y se sustituyen los valores de F_n por w, calculando el valor de r para cada iteración y obteniendo después su módulo. Así, se obtienen los valores de rho (módulo de r) para ese intervalo, y los valores del eje real e imaginario asociados a cada valor de rho.

A continuación, se define la función wrapped que contiene la representación de estas regiones de estabilidad:

```
25  #Función de representación de regiones de estabilidad
26  def Wrapped_Stability_Zones(rho, Re, Im, Scheme):
27      plt.contour( Re, Im, transpose(rho), linspace(0, 1, 10))
28      plt.xlabel("Re(|r|)")
29      plt.ylabel("Im(|r|)")
30      j = "Estabilidad "+Scheme.__name__
31      plt.title(j)
32      plt.axis('equal')
33      plt.grid()
34      plt.show()
35      return j
```

Figura 16: Función de representación de las regiones de estabilidad

Donde los argumentos de entrada son los mismos de salida de la función anterior más el esquema de la región de estabilidad elegida, donde se dibuja el contorno de la región en los que el inverso de rho tiene los valores desde 1 hasta 10. Habrá dos zonas en el dibujo resultante, una interior al contorno de valor del inverso de rho 1(la frontera de la región de estabilidad, será la línea amarilla en los gráficos de los esquemas) y otra exterior al mismo. La zona en la que estén el resto de líneas además de la unidad será la región de estabilidad, y la zona en blanco la región en la que el esquema es inestable.

A continuación, se definen las regiones de estabilidad de los diferentes esquemas utilizados en este programa.

a) Euler

La región de estabilidad del esquema Euler es la siguiente:

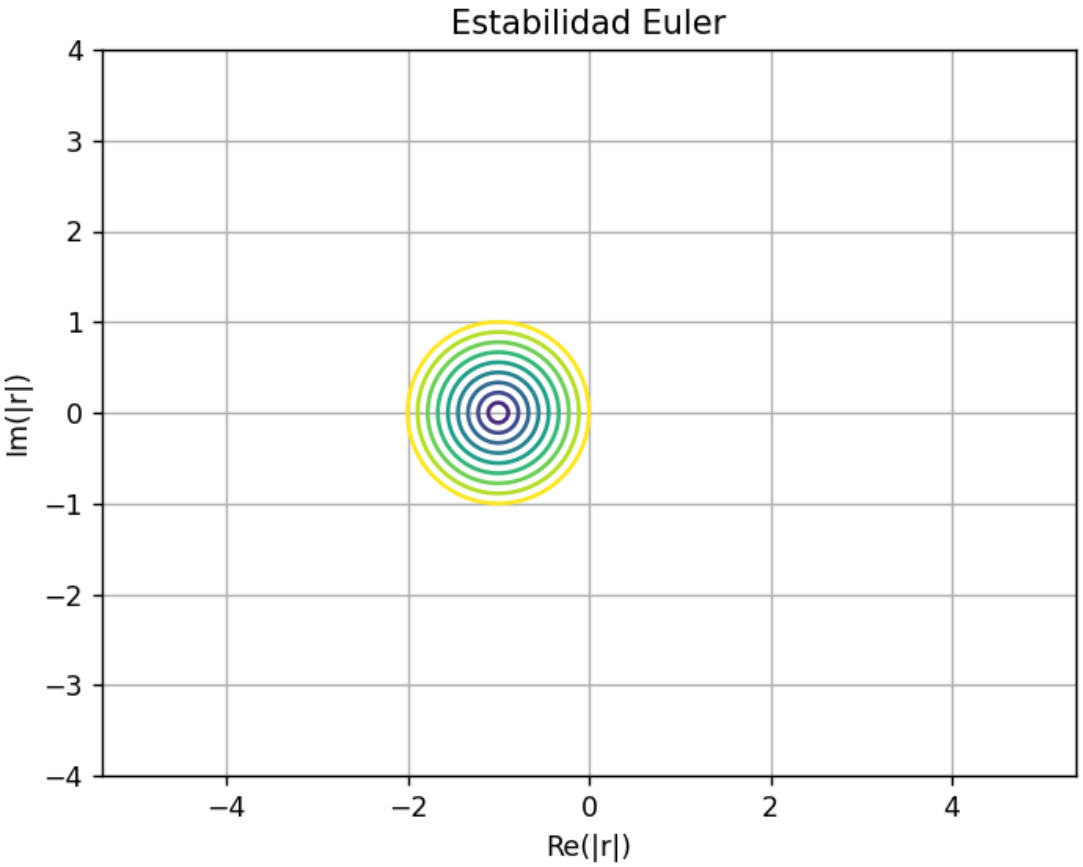

Figura 17: Región de estabilidad del esquema Euler

Se observa que la región de estabilidad es un círculo de radio unidad y de centro el (-1,0) en el plano complejo.

b) Euler inverso

La región de estabilidad del esquema Euler inverso es la siguiente:

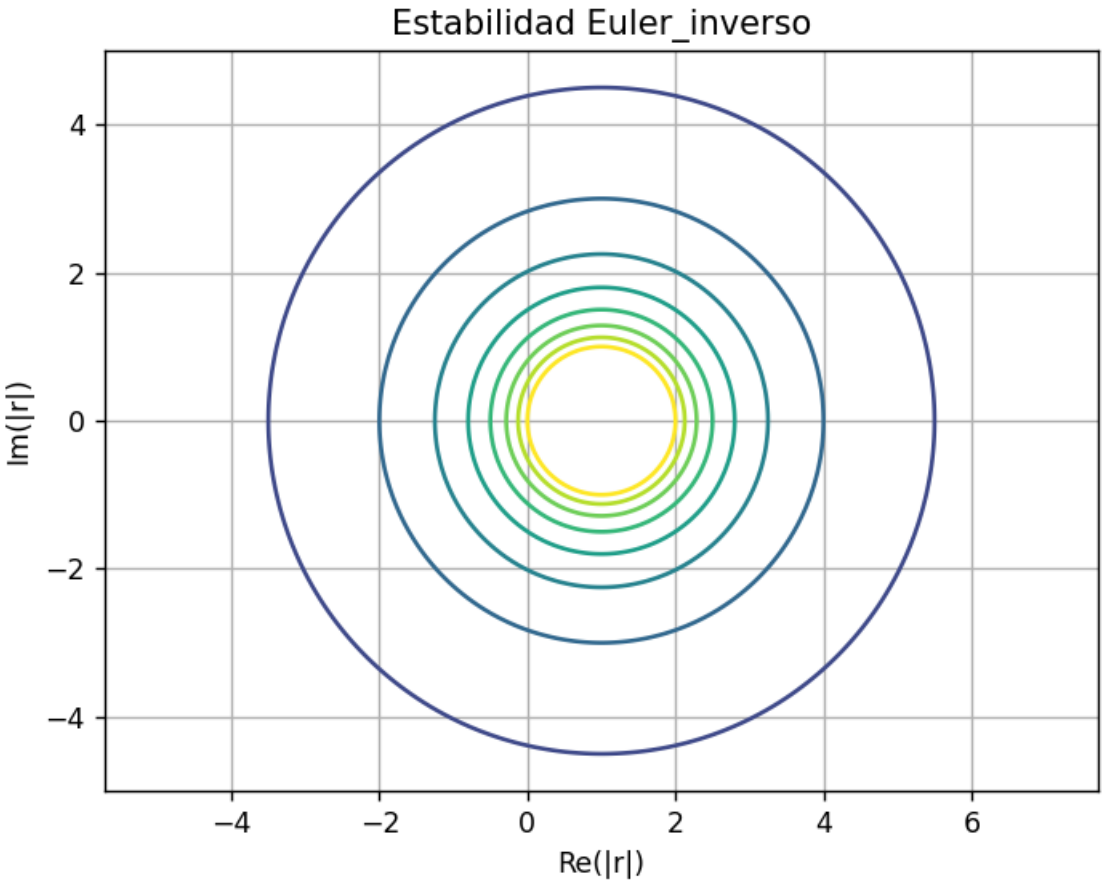

Figura 18: Región de estabilidad del esquema Euler inverso

La región de estabilidad resultante es la contraria al esquema de Euler, es todo el mapa complejo a excepción de un círculo de radio unidad y centro en el (-1,0). Este resultado es lógico teniendo en cuenta la forma de los esquemas de Euler y Euler inverso.

c) Crank-Nicholson

La región de estabilidad del esquema Crank-Nicholson es la siguiente:

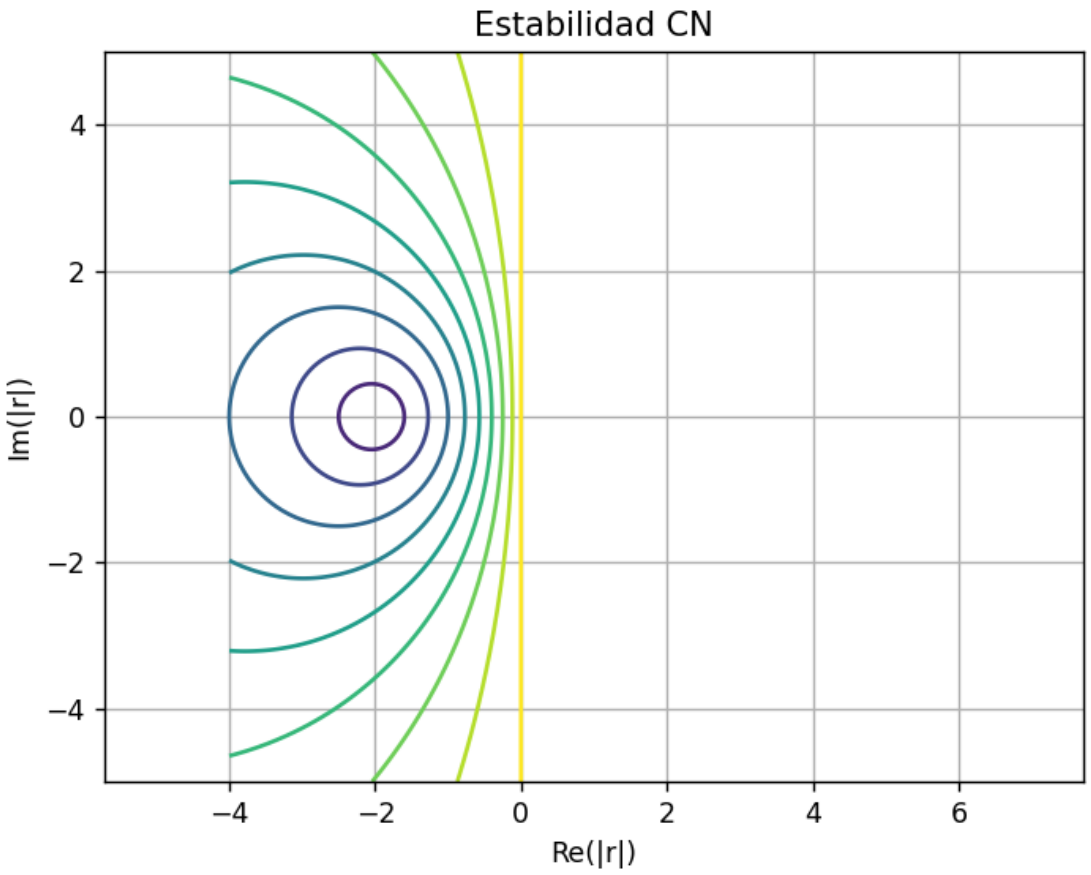

Figura 19: Región de estabilidad del esquema Crank-Nicholson

Donde se puede observar que la frontera de la región de estabilidad es el eje imaginario, por lo que toda la región de estabilidad es el plano complejo con parte real negativa.

d) Range-Kutta de orden 4

La región de estabilidad del esquema RK4 es la siguiente:

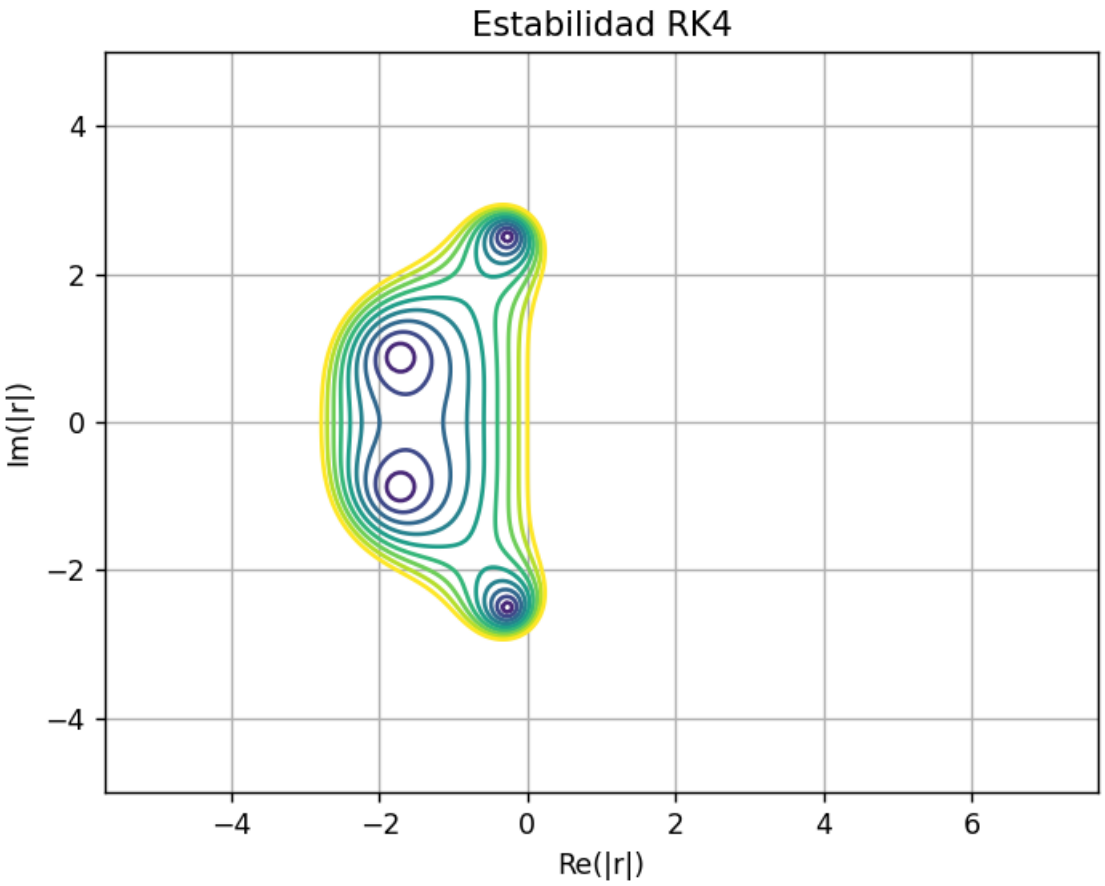

Figura 20. Región de estabilidad del esquema RK4

Donde la mayor complejidad del esquema numérico provoca que la región de estabilidad sea más complicada que el resto de los esquemas estudiados.

e) Leap-Frog

El programa devuelve una región de estabilidad del esquema Leap-Frog:

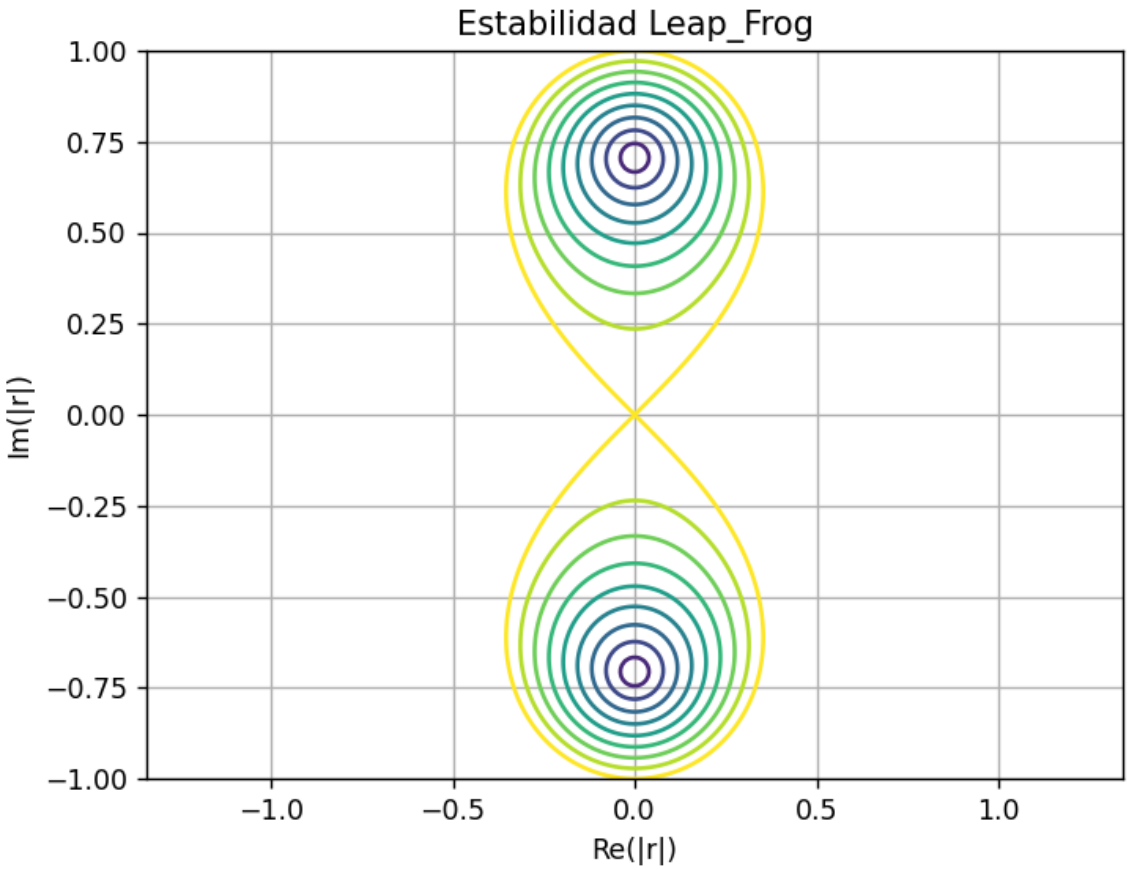

Figura 21: Región de estabilidad obtenida para el esquema Leap-Frog

Que es parecida a la región de estabilidad real, un segmento entre el -1 y el 1 en el eje imaginario, pero no exactamente igual. Esto es así porque la aparición de un paso de tiempo adicional respecto a los otros esquemas hace que la función para obtener esta región de estabilidad no sea válida, aunque el resultado que ofrece no se aleja demasiado de la realidad.

5.Explicación de las soluciones del oscilador lineal con las regiones de estabilidad absoluta

En esta sección se pretende analizar los resultados obtenidos para la integración del oscilador lineal y los errores que se cometen con cada uno de los esquemas aplicando los resultados de la región de estabilidad absoluta del esquema. Se observará si los autovalores del polinomio característico del problema del oscilador lineal caen dentro de la región de estabilidad absoluta de cada esquema (especialmente si caen en la frontera, con ganancia unidad, que sería la situación deseada) y se comparará con los resultados obtenidos en el apartado 3, para comprobar si hay coherencia con ellos.

Los autovalores del problema del oscilador lineal son los autovalores de la matriz que se obtuvo en la ecuación 3, y son los siguientes.

$$|A - \lambda I| = 0 \rightarrow \begin{vmatrix} -\lambda & 1 \\ -1 & -\lambda \end{vmatrix} = 0 \rightarrow \lambda^2 + 1 = 0 \rightarrow \lambda = \pm i$$

Ecuación 5: Obtención de los autovalores del problema del oscilador lineal

Por lo que los autovalores de este problema caen en el eje imaginario, en 1 y -1. Para los distintos esquemas, debería cumplirse que si su región de estabilidad contiene estos puntos, el esquema debería ser estable y no dar grandes errores de módulo, mientras que si cae fuera de esta región, el esquema será inestable y se producirán grandes errores que crecerán rápidamente.

a) Euler

En la figura 8 se observa que, aunque mantiene la forma de oscilaciones, estas crecen exponencialmente de tamaño. En la figura 17 se observa que la región de estabilidad del esquema Euler no contiene los puntos de los autovalores del problema del oscilador lineal. Estos resultados son coherentes, al caer los autovalores fuera de la región de estabilidad, el esquema es inestable para este problema, aumentando rápidamente el error cometido y alejándose de la solución real en muy poco tiempo, haciendo que el esquema no sea adecuado para la integración del problema.

b) Euler inverso

En la figura 9 se observa una estabilidad adecuada del esquema, creciendo ligeramente en la primera oscilación y después manteniéndose estable alrededor de ese esquema. Esta desviación quizás pueda provenir del error inicial del esquema. En la figura 18 se observa que la región de estabilidad del esquema sí incluye los puntos asociados a los autovalores del problema del oscilador lineal (aunque no en la frontera), por lo que el esquema es estable. También se observa coherencia entre estos dos resultados, ya que el resultado del oscilador lineal es estable, pese a la desviación inicial, haciendo que el esquema pueda ser adecuado para la integración del esquema (aunque por su error inicial esto no es demasiado aconsejable).

c) Crank-Nicholson

En la figura 10 se ve que apenas hay desviación en el error de módulo cometido con este esquema, mientras que en la figura 19 se observa que, en este esquema, los autovalores caen en la frontera de la región de estabilidad, por lo que el esquema es estable. Estos resultados también presentan coherencia, pues apenas hay desviación en el error de módulo cometido tras un paso de tiempo muy grande. La principal diferencia con el esquema de Euler inverso es que aquí los autovalores caen en la frontera, por lo que no se observan esas oscilaciones en los máximos y mínimos que se producen en el Euler inverso alrededor de un valor de equilibrio.

d) Range-Kutta de orden 4

En la figura 11 se ve que, como en el Crank-Nicholson, apenas se produce desviación en el módulo del error cometido, mientras que en la figura 20 se observa que los autovalores caen en la frontera de la región de estabilidad del esquema, por lo que ambos resultados vuelven a ser coherentes una vez más.

e) Leap-Frog

Los autovalores de este esquema caen justo en los límites del segmento que es la región de estabilidad del esquema, por lo que cabe esperar los resultados de la figura 13, en donde se ve que el esquema sigue siendo estable, de forma similar a como lo era el Crank-Nicholson (lógico, por ser esquemas del mismo orden y en los que los autovalores caen en la misma zona de la región de estabilidad). Como curiosidad, con la región de estabilidad obtenida en la figura 21 los resultados de este esquema serían perfectamente válidos, ya que los autovalores seguirían cayendo dentro de la frontera de la región de estabilidad.

Como conclusión final, se puede llegar a que mediante el análisis de la región de estabilidad y los autovalores del sistema asociado al problema estudiado, pueden explicarse los resultados diferentes entre esquemas numéricos del mismo orden, y poder elegir un esquema apropiado para la integración del problema elegido.

6.Relaciones funcionales del programa utilizado

En el siguiente esquema se pueden ver el esquema de relaciones funcionales utilizado en este programa:

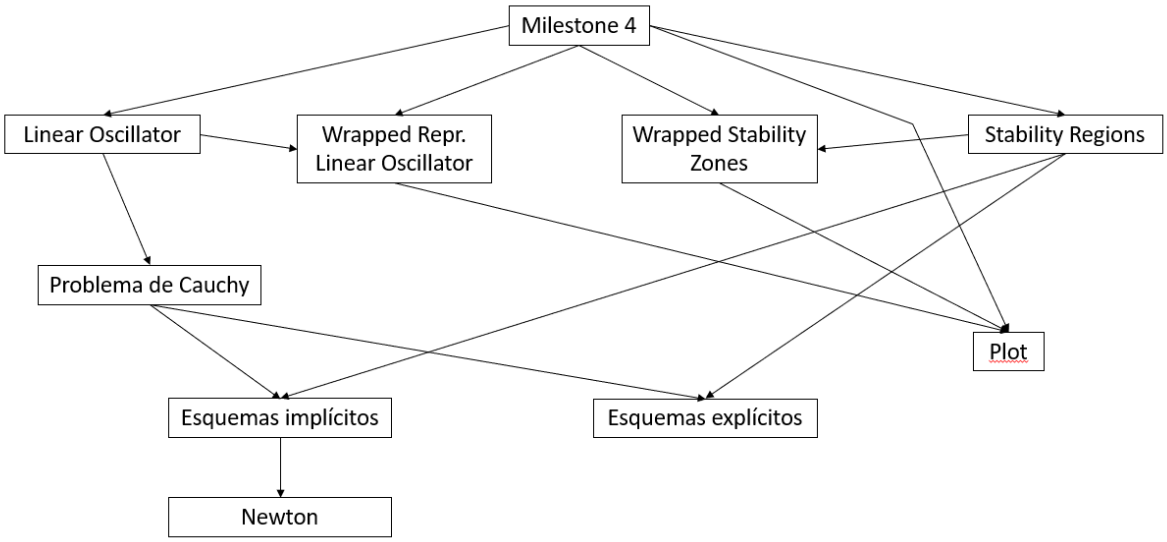

Figura 22: Esquema de relaciones funcionales del programa

7.Crítica del código

En mi opinión, creo que se ha producido una evidente mejora desde los Milestones anteriores a este. Se ha producido una reducción considerable de las líneas de código del programa, gracias al uso de las funciones wrapped para la obtención y representación tanto de las soluciones del problema del oscilador lineal como de las zonas de estabilidad absoluta de los distintos esquemas. El método Leap Frog se ha podido implementar sin problemas en el mismo módulo que los esquemas anteriores, aunque por las características de este esquema (principalmente la necesidad de dos condiciones

iniciales) se ha debido crear una nueva función del problema de Cauchy adaptada a este esquema. Quizás podría haberse incluido dentro de la función del problema de Cauchy principal, pero me ha resultado más sencillo realizar esta separación entre estas dos funciones. La principal desventaja es que añade más líneas de código al programa, pero se ha optado por pagar ese precio teniendo en cuenta que no el incremento no ha sido una cantidad muy elevada de líneas.

Si se quisiera ahorrar líneas de código, podría optarse por la eliminación de la solución analítica del sistema, pero se ha decidido añadir esta representación para facilitar una mayor comprensión de las características y adecuación de las soluciones los esquemas analizados. Otro punto potencial de mejora es poder adaptar la función de obtención de las soluciones lineales para poder representar la región del esquema Leap Frog, pero es cierto que por las características de esta región (un segmento recto) puede ser complicado hacerlo.

En conclusión, aunque la mejora es evidente respecto a los primeros hitos, aún hay margen para la mejora, que espero que siga produciéndose durante los siguientes Milestones.