Universidad de Alicante

Facultad de Ciencias

Bloque 1

Física Computacional

Gabriel Simón López

2023/2024

Índice

1. Introducción

En este documento presentaré:

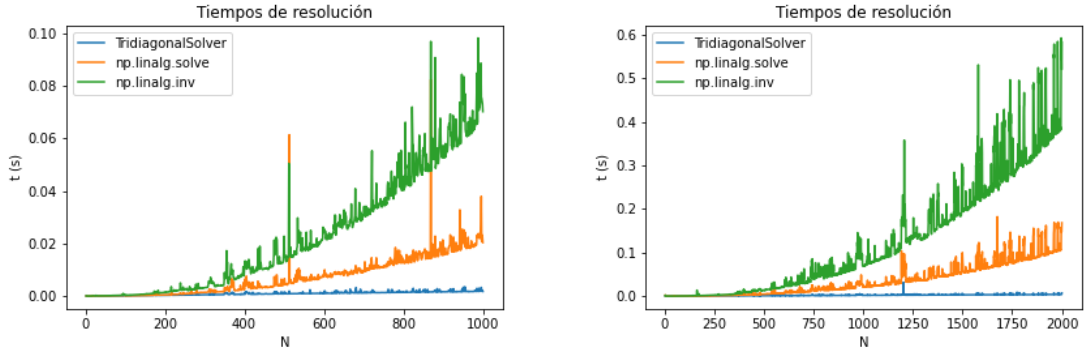
- Los programas de cada ejercicio.
- Un breve informe explicando qué partes hemos hecho de cada ejercicio, resaltando las partes más importantes de cada programa, mostrando los resultados más importantes como comparación de métodos, convergencia, etc. Menos en las prácticas en las que se piden animaciones, estas se encontrarán en los programas.
- Las conclusiones sacadas de los programas o extras.

2. Práctica 1

La primera parte del programa se basa en la implementación de una función, la cual llamaremos TridiagonalSolver, en la que le pasaremos una matriz tridiagonal random y nos devolverá un array de sus soluciones.

Una vez creada, usaremos la función de python time.time(), la cual funcionará como nuestro cronómetro de solución de la matriz pasada, ya que la llamaremos tanto antes como después de la resolución de la matriz.

Esto lo realizaremos para los tres métodos distintos posibles y para distintos tamaños del sistema. Los cuales graficaremos y compararemos.



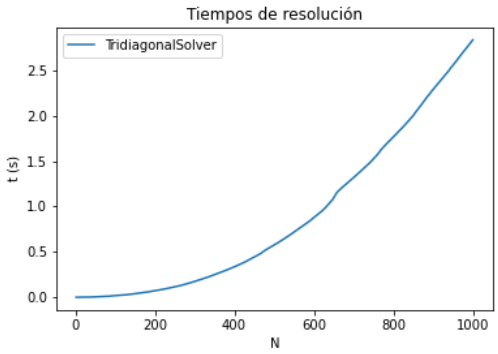
(a) *Para N = 1000* **(b)** *N = 2000*

Figura 1: *Gráficas de tiempo*

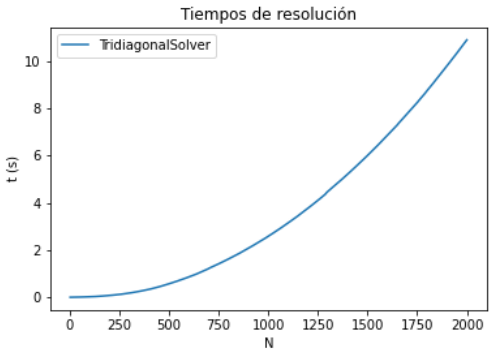
En donde observamos que el tiempo para resolver la matriz pasada es muchísimo menor con el método implementado con la TridiagonalSolver y que presenta ciertos picos a lo largo de las resoluciones, esto puede ser debido a que al tratarse de matrices aleatorias algunas serán más sencillas de resolver que otras.

Para ver cuanto tiempo tarda cada método con los distintos N que le podemos pasar, hemos cambiado la posición del time.time() y en vez de reiniciarse cada vez, lo hemos colocado fuera del for.

Podemos observar como el tiempo que tarda en calcularse crece exponencialmente.



(a) *Para $N = 1000$*



(b) *$N = 2000$*

Figura 2: *Crecimiento exponencial*

En las cuales observamos que para $N = 1000$ tarda unos $2,6$ segundos, mientras que para $N = 2000$ tarda al rededor de 11 segundos.

3. Práctica 2

En esta práctica implementaremos un programa que resuelva la ecuación de Laplace para el problema dado. Lo resolveremos con distintas condiciones de contorno: Dirichlet y Neumann.

3.1. Introducción

La ecuación de Poisson-Laplace es un buen ejemplo de ecuación parcial elíptica. En esta parte del informe, resolveremos esta ecuación en concreto para encontrar potenciales electromagnéticos de distintas formas.

El ejemplo que resolveremos es el de una malla cuadrada con potencial 0 en todos los puntos excepto en uno de sus lados, el cual tendrá potencial 100V.

3.2. Parte 1: Resolvemos con un método directo y condiciones de Dirichlet

Resolveremos la Ecuación de Laplace para el problema dado con Condiciones de Dirichlet, en donde dado el valor de ϕ en la frontera de la region de interes, encontraremos su valor en cualquier otro punto.

Obtendremos la siguiente gráfica, es importante comentar que en nuestro caso el potencial se encuentra en la parte inferior, esto es simplemente por como se ha definido la matriz en el programa.

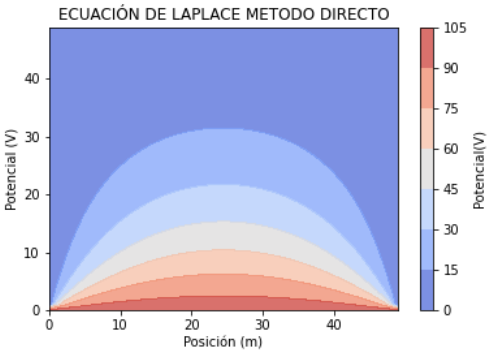


Figura 3: *Ecuación de Laplace método directo*

Partes de interés del código:

- **Línea 20 a la 32:** Definimos las matrices que vamos a usar.
- **Línea 37:** Obtenemos la matriz deseada con un producto tensorial. La cual contiene las condiciones de Dirichlet

3.3. Parte 2: Resolvemos con el método iterativo de Jacobi y condiciones de Dirichlet

Resolveremos el mismo problema con el metodo iterativo de Jacobi.

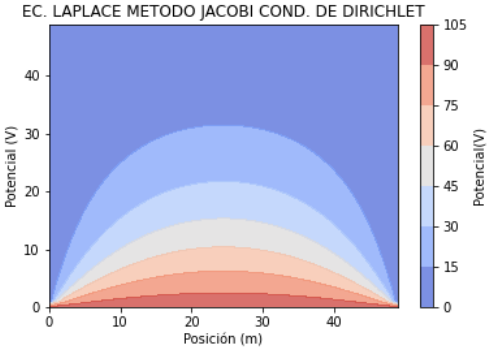


Figura 4: *Ecuación de Laplace método Jacobi y condiciones de Dirichlet*

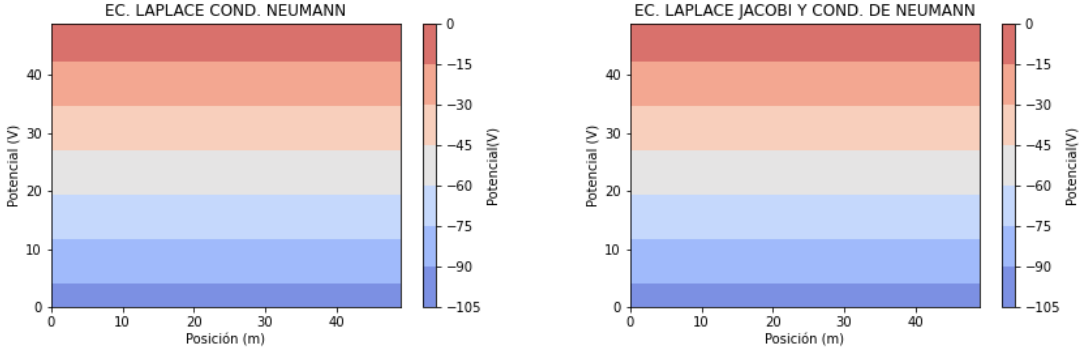
En donde observamos que obtenemos la misma gráfica que en 3.

Parte de interés del código:

- **Línea 75 a 89:** Definimos la función del método iterativo de Jacobi. En el cual le quitaremos la diagonal a la matriz introducida (matriz B), y con el producto de la inversa de su diagonal(InvD) con el obtenido del producto de la mtriz B con la interación anterior, llegaremos a la solución con una toleracia dada.

3.4. Parte 3: Resolvemos con el metodo directo y jacobi y condiciones de Neumann.

Con las condiciones de Neumann dado el valor de $\nabla\phi$ en la frontera de la region de interes, queremos encontrar su valor en cualquier otro punto. En este caso la diferencia principal se encuentra en las matrices A y B, las cuales deberán llevar las condiciones de contorno de Neumann.



(a) *Ec. Laplace método directo* **(b)** *Ec. Laplace método Jacobi*

Figura 5: *Distintos métodos con condiciones de Neumann*

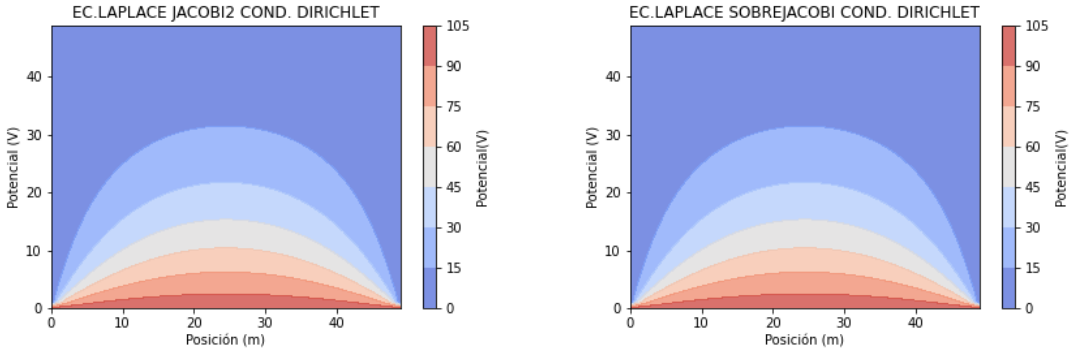
Observamos que las gráficas 5a y 4 son distintas debido a que al imponer condiciones de Dirichlet estamos fijando los valores laterales y superior, en cambio en las condiciones de Neumman en los laterales de la matriz no existe un valor fijo, por lo que la gráfica será distinta. Por esta razón también observamos que con Neumann tarda más que Dirichlet, porque tiene que resolver una ecuación más.

Parte de interés del código:

- **Línea 124 a 133:** Creamos las matrices A y B con las condiciones de Neumann.
- **Línea 136:** Aplicamos la solución del método directo a la matriz con condiciones de Neumann
- **Línea 151:** Aplicamos el método iterativo de Jacobi a las matrices con condiciones de Neumann.

3.5. Parte Extra: Comparación tiempo de solución de Jacobi y de Sobrejacobi

Como extra crearemos de nuevo la función Jacobi de otra forma enseñada en métodos numéricos de primero de carrera e implementaremos el método de sobrerelajación, Sobrejacobi. Comparando así los tiempos de resolución y viendo como el método de sobrerelajación es mas rápido que el de relajación. Realizaremos el mismo cálculo pero con un método de resolución iterativo de sobrerelajación, en donde tendremos que ver como tarda menos ya que podremos acelerar el proceso si aumentamos un poco esta variación, controlando el parámetro con un nuevo parámetro w. El método elegido es el de Sobrejacobi, usando como base el método de Jacobi desarrollado en apartados anteriores.



(a) *Ec. Laplace Jacobi2 cond. Dirichlet* **(b)** *Ec. Laplace SobreJacobi cond. Dirichlet*

Figura 6: *Solución con Jacobi y SobreJacobi*

En donde el programa nos imprime:

El tiempo que tarda en resolver Jacobi con Dirichlet es: **16.868 segundos**

El tiempo que tarda en resolver SobreJacobi con Dirichlet es: **8.872 segundos**

De tal manera, que vemos como con el método de sobrerelajación tardaremos en torno a la mitad de tiempo que con el de relajación.

Parte de interés del código:

- **Línea 184 a 198:** Creamos la segunda función de Jacobi.
- **Línea 200 a 215:** Creamos la función de sobrerelajación SobreJacobi.
- **Línea 217 a 230:** Comparamos tiempos de resolución entre ambos métodos.

4. Práctica 3

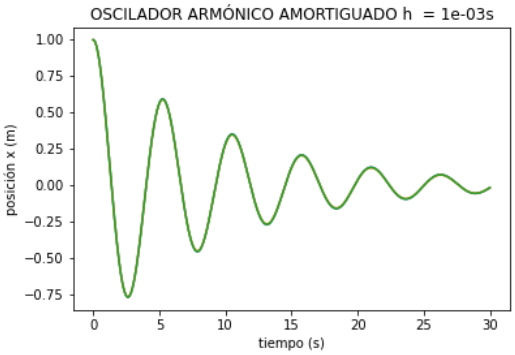
En esta práctica implementaremos un programa el cual resuelva la siguiente ecuación diferencial para el oscilador armónico amortiguado: $\ddot{x} = -w^2 x - \alpha \dot{x}$, usando métodos distintos. Además, compararemos los resultados obtenidos con la solución exacta y veremos como varía el error.

Primero resolveremos la ecuación diferencial para el oscilador armónico con el método explícito de Euler-forward. Este método consiste en discretizar la parte espacial de la ecuación mediante sus diferencias finitas centrales y la parte temporal por sus diferencias hacia adelante.

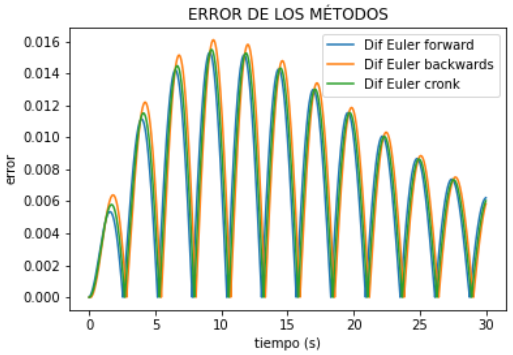
Para el método de Euler implícito resolvemos la ecuación diferencial en función de la derivada en el siguiente paso, es mas lento(requiere resolver una ecuación no lineal en cada paso). En este caso usaremos x_{i+1} en función del paso i+1.

Una vez estudias la estabilidad de los métodos parece que la sencillez del método FTCS tiene sus limitaciones, que desde un punto de vista físico la falta de estabilidad se puede interpretar como la imposibilidad de que el cálculo numérico avance en el tiempo más deprisa que la perturbación. Para evitar esto el método de Crank-Nicolson propone utilizar las diferencias temporales hacia atrás (backward) y promediar con las diferencias forward.

Para ello estudiaremos tres casos de pasos de h, para ver como se comporta cada método:

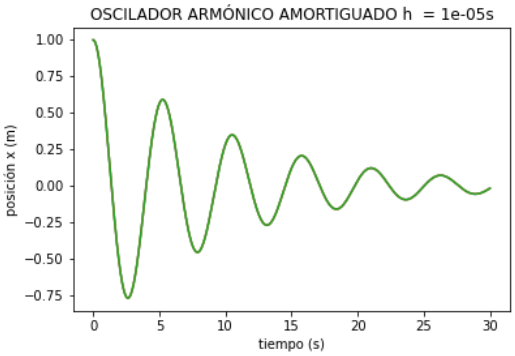


(a) *Oscilador armónico amortiguado*

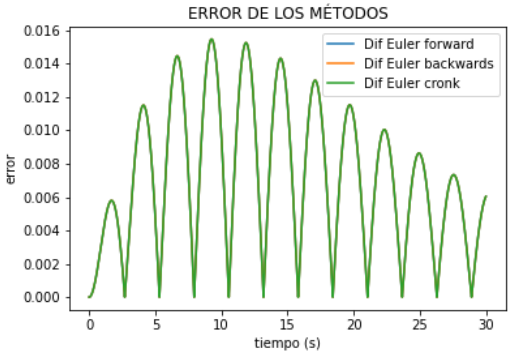


(b) *Errores de los métodos comparados con la solución exacta*

Figura 7: *Comparativa h = 1e-3 s*

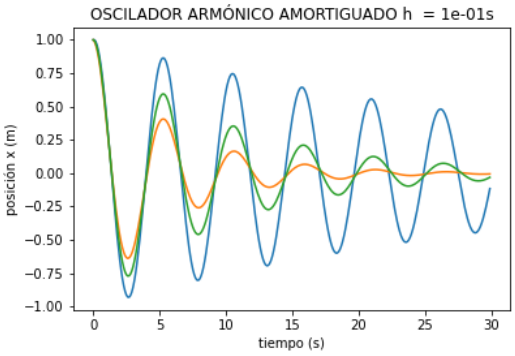


(a) *Oscilador armónico amortiguado*

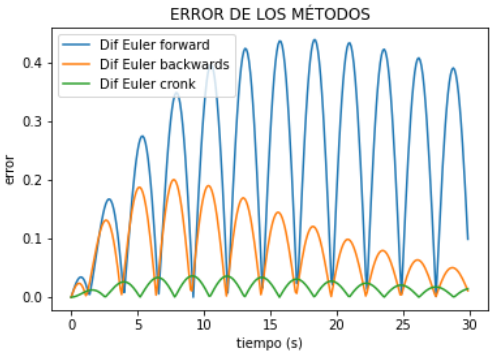


(b) *Errores de los métodos comparados con la solución exacta*

Figura 8: *Comparativa h = 1e-5 s*



(a) *Oscilador armónico amortiguado*



(b) *Errores de los métodos comparados con la solución exacta*

Figura 9: *Comparativa $h = 0.1$ s*

Conclusiones:

- Euler explícito es más fácil de programar que Euler implícito
- No es recomendable usar Euler explícito si no conocemos la región de estabilidad, ya que una mala elección de h (paso de tiempo) puede hacer que se inestabilice.
- Euler implícito es más costoso computacionalmente ya que involucra la resolución de una ecuación lineal o no lineal. Sobre todo en este último caso de no linealidad, en donde tendremos que resolver una ecuación algebraica (o incluso sistema de ecuaciones) en cada iteración.
- Ambos son métodos de primer orden, es decir, el error de la solución exacta y la numérica en el tiempo es a orden de h.
- El método de Crank-Nicolson se trata de un método implícito incondicionalmente estable de segundo orden, esto quiere decir que mejorará la precisión de los métodos de Euler, haciendo el error más pequeño. Aunque sigue presentando la misma desventaja que el método implícito.

Parte de interés del código:

- **Línea 8 a 24:** Definimos la función Eulerforward, en la cual le introducimos los parámetros x0(posición inicial),v0(velocidad inicial),h(pasos de tiempo),w(frec. angular),alpha(coef. amortiguamiento),tsim(tiempo de simulación) y nos devuelve unos arrays de: t(tiempo),xa(posición calculada con el método) y v(velocidad calculada con el método).
- **Línea 28 a 46:** Definimos la función Eulerbackward, en la cual con respecto a la anterior cambia la matriz a resolver.
- **Línea 48 a 66:** Definimos Crank-Nicolson
- **Línea 71 a 75:** Definimos la solución exacta de la ecuación con la cual compararemos los demás métodos.

5. Práctica 4

En esta práctica tenemos que resolver la ecuación de difusión a ciertas condiciones dadas en cada apartado. La ecuación de difusión es una EDP que describe fluctuaciones de densidad en un material que se difunde. También se usa para describir procesos que exhiben un comportamiento de difusión. Resolveremos por tanto, una EDP parabólica.

Tendremos que tener en cuenta que tenemos que cumplir la condición de estabilidad $\Delta t \leq \frac{\Delta x^2}{2D}$.

En el primer apartado resolvemos la ecuación con método explícito, teniendo en cuenta que la barra presenta una temperatura inicial de $T(x) = 100$ en todo su dominio menos en los extremos cuya temperatura es 0. Por tanto, a lo largo de la animación tenemos que ver como la barra se va enfriando a lo largo del tiempo, del tal manera que al final la barra tendrá que estar en 0 (o casi 0).

En el segundo apartado resolveremos con el método de Crank-Nicolson el mismo problema, sólo que la barra en uno de sus extremos se encuentra a 0 y en otro a 50. Esperamos en este caso un resultado similar, en el que la barra se va enfriando a lo largo del tiempo, hasta llegar al equilibrio. En este caso, al presentar un extremo en 0 y otro en 50 tardará más tiempo en llegar al equilibrio con respecto al apartado anterior.

Por último, en el tercer apartado, planteamos la misma barra problema que en segundo apartado pero una región de la barra tiene un coeficiente de difusividad menor. Por lo que cabe esperar, es que tarde más en llegar al equilibrio y que no lo haga de manera uniforme.

Todos estos resultados se podrán ver en las tres animaciones del código del as que podemos sacar las siguientes conclusiones:

- Observamos correctamente la dependencia de la transmisión de temperatura de un material en función del coeficiente de transmisión, tal que a menor coeficiente mayor es el tiempo de alcanzar el equilibrio.
- De los dos primeros apartados comprobamos como el método de Cranck-Nicolson es más estable que el método FTCS, el cual es un método explícito y tiene que cumplir una condición de estabilidad
- Al comparar los dos últimos apartados comprobamos como al tener un coeficiente de difusión variable, no solo cambia el tiempo para llegar al equilibrio, sino que a este no se llega uniformemente.

6. Práctica 5

En esta práctica vamos a resolver la ecuación de Schrödinger dependiente del tiempo en 1D con el método de Crank-Nicholson para un paquete de ondas de una partícula libre en un pozo infinito.

De tal manera que necesitaremos definir la función de onda de una partícula libre, imponer que se encuentre en un pozo infinito y resolverlo con el método de Crank-Nicolson.

Parte de interés del código:

- **Línea 36 a 52:** Creamos la matriz que vamos a necesitar para Crank-Nicolson, en la línea 52 imponemos las condiciones de contorno que nos permitira´n encontrarnos en un pozo infinito.
- **Línea 29 y 30:** Creamos la función que nos devuelve la función de onda de una partícula libre.
- **Línea 70 a 81:** Animamos para el módulo al cuadrado de la función de onda de la partícula libre. La multiplicación de Psil por su conjugado produce el módulo al cuadrado, que en el contexto de la mecánica cuántica corresponde a la probabilidad de encontrar la partícula en una posición determinada.
- **Línea 86 a 97:** Animamos la parte real de la función de onda
- **Línea 103 a 114:** Animamos la parte imaginaria de la función de onda

En las animaciones vemos una representación de un paquete de ondas unidimensional: la parte real, parte imaginaria y la densidad de probabilidad de un paquete de ondas desplazándose hacia la derecha. Al representar un paquete de ondas de una partícula libre, al contrario que las ondas planas, es de cuadrado integrable y, por tanto, se puede normalizar tal y como vemos en la primera animación.

7. Práctica 6

Antes de resolver lo que la práctica nos pide, como extra primero realizaré un programa para la resolución de la EDP hiperbólica con el método implícito y explícito y después, para una onda amortiguada.

En este caso para la solución de la EDP hiperbólica con el método FDTD usaremos la siguiente expresión:

$$u_i^{k+1} = \frac{c^2\delta t^2}{\delta x^2} * ((u_{i+1}^k + u_{i-1}^k - 2u_i^k) + (2u_i^k - u_i^{k-1})) \tag{1}$$

Esta expresión la hemos deducido aplicando diferencias centradas en ambas partes de la ecuación hipérbolica y despejando el término u_i^{k+1}.

En cambio para el método implícito, tras promediar las partes espaciales y temporales llegaremos a la expresión:

$$u_i^{k+1} + u_i^{k-1} - 2u_i^k = \frac{c^2\delta t^2}{\delta x^2} * (u_{i+1}^{k+1} + u_{i-1}^{k+1} - 2u_i^{k+1} + (u_{i-1}^{k-1} - 2u_i^{k-1})) \tag{2}$$

En este caso y en la práctica, lo hemos resuelto el caso de una onda a la cual le aplicamos un pulso sinusoidal y la soltamos en reposo. Conocemos las condiciones iniciales, es decir, sabemos cómo soltamos la cuerda. Además, como inicialmente se encuentra en reposo, podemos aplicar Taylor y quedarnos en el primer orden. Con ello deducimos que la posición en el primer instante será la misma que en el 0 aproximadamente, y podemos inicializar la primera parte del programa del ejercicio 6.

Los resultados se podrán ver en las animaciones del programa adjuntado con la solución del problema.

En la primera parte resolveremos la ecuación en derivadas parciales hiperbólicas a la que le hemos añadido un termino de una fuerza de rozamiento producida por la viscosidad del fluido.Este termino depende de dt a primer orden y en la función se llamara **tro**.

En la segunda parte resolveremos el mismo caso pero con un método implícito, es importante destacar que en este método necesitaremos una matriz,M1, la cual refleja el paso actual y otra,M2, la cual reflejará el paso anterior.

Por último, en el tercer apartado resolveremos la ecuación del telégrafo, en la que metemos un término fuente adicional a la ecuación. Lo resolveremos aplicando el método FDTD a la ecuación planteada, siguiendo con la metodología de la parte 1. En este caso tendremos 3 términos y trabajaremos con 2 pasos anteriores.

Como extra, además de que el pulso sea sinusoidal, también he añadido la posibilidad de imponer un pulso triangular, uno cuadrado o incluso una combinación de ambos.

Parte de interés del código:

- **Línea 21 a 47:** Definimos distintos tipos de pulsos de ondas que podemos usar en vez del seno, como extra.
- **Línea 101 a 108:** Definimos la función explícita para resolver la edp hiperbólica con el término añadido.
- **Línea 152 a 168:** Definimos la función implícita para resolver el mismo problema anterior.
- **Línea 230 a 237:** Definimos la función que resuelve la ecuación del telégrafo.

Conclusiones y análisis de resultados:

- En las animaciones 1 y 3, además de la vibración principal, se obtienen unas pequeñas ondas en la propia cuerda que corresponden a una vibración de alta frecuencia debida a que la primera derivada no está bien ajustada ya que estos métodos solo tienen en cuenta el paso anterior. Que la segunda animación no los presente es debido a que los métodos implícitos son más precisos ya que tienen en cuenta el ajuste de las derivadas futuras.
- En todos los apartados con los distintos pulsos posibles hemos visto la amortiguación de la onda.

- En la ecuación del telégrafo observamos que la amplitud disminuye mucho más rápido que en el método explícito.
- Si variamos el número de nodos del seno vemos que obtenemos los modos normales de vibración del seno y disminuye el periodo de las ondas que graficamos.

8. Práctica 7

En esta práctica vamos a resolver la ecuación de advección con tres distintos métodos explícitos, el llamado upwind, downwind y el de diferencias centradas. El método upwind se llama así debido a que va a la misma dirección de la velocidad y el downwind se llama así por lo contrario, por tanto la dirección de la velocidad es importante, ya que uno cambiara al otro dependiendo del signo de esta.

Será muy importante tener en cuenta el criterio de estabilidad de cada método, estando este determinado por el **Número de Courant-Friedrichs-Levy:** $u = \frac{c\Delta t}{\Delta x}$. Con u el número de Courant y c la velocidad. Para los parámetros introducidos $\Delta t = \Delta x = 0{,}1; c = 1$, nuestro límite de estabilidad estará cuando $u = 1$.

En las animaciones vemos como, para los parámetros comentados anteriormente, el método explícito de upwind funciona, mientras que el de downwind y el de diferencias centrales explota. También vemos que al cambiar de signo la velocidad estamos camiando la dirección de propagación de esta y por lo tanto, no solo la dirección de la onda en la animación es la contraria, sino que también los métodos upwind y downwind cambian entre ellos, por la naturaleza de cada método. Dado que el método de las diferencias centradas también explota para el límite de estabilidad, vamos a programar el método de Lax-Wendroff el cual es una extensión que mejora la precisión en la captura de ondas más afiladas y rápidas.

El método explícito de Lax-Wendroff, este método es de gran interés por sus propiedades de orden y estabilidad. Se obtiene, usando el desarrollo de Taylor de u(x, t) respecto de la variable t en (x, t). De la cual obtendremos la siguiente ecuación: $u_i^{k+1} = u_i^k - \alpha(u_{i+1}^k - u_{i-1}^k) + \beta(u_{i+1}^k + u_{i-1}^k - 2u_i^k)$. Con $\alpha = c\Delta t/2\Delta x$ y $\beta = c^2\Delta t^2/2\Delta x^2$. Vemos en la animación como con los mismos parámetros, este no explota y grafica satisfactoriamente la solución de la ecuación, como cabría esperar. De nuevo, sería de interés usar un método implícito, por lo menos para evitar la necesidad de cumplir el criterio de estabilidad.

Parte de interés del código:

- **Línea 17 a 19:** Estudiamos la estabilidad del método para las condiciones dadas.
- **Línea 32 a 50:** Definimos los tres métodos explícitos que vamos a usar.
- **Línea 84 a 86:** Definimos el método explícito de Lax-Wendroff como extra.

9. Practica 8

En esta práctica se nos pide programar la solución de la ecuación de Burgers no viscosa, ya que su término de difusión es nulo. Este ejercicio sigue la misma estructura que el anterior, asi que solo explicaré las cosas que sean distintas.

Parte de interés del código:

- **Línea 34 a 43:** Implementamos el método de Upwind y Diferencias centradas con las condiciones periódicas.
- **Línea 75 a 79:** Implementamos el método extra Downwind impniendo también las condiciones periódicas

En las animaciones vemos que el método Upwind si que funciona para resolver la ecuación de Burgers, pero que, el metodo de la diferencias centrales falla su estabilidad y explota. Podemos minimizarlo si definimos un beta muy pequeño, pero al menos, en mi programa siguen existiendo pequeñas variaciones en la onda, que no se observan en la animación del upwind. Lo que nos deja claro que será mejor el uso de métodos implícitos, ya que, aunque estos son más costosos computacionalmente hablando, nos aseguraremos de no tener problemas con la condición de estabilidad.

En el extra con la gráfica del Downwind, llegamos a la misma conclusión de la práctica anterior.