Computación Numérica
Segunda Práctica de Fortran. Curso 2009 – 2010
Sistemas de ecuaciones lineales. Ecuaciones no lineales.

En esta práctica se propone la programación de dos métodos numéricos para resolver sistemas de ecuaciones lineales, **LU**, variante para matrices tridiagonales, y **Cholesky**, y del método de **Newton** para aproximar una raíz compleja de una ecuación.

Parte 1. Sistemas de ecuaciones lineales

Método LU para sistemas de matriz tridiagonal

- Almacena la matriz tridiagonal en tres vectores, uno por cada diagonal, y crea una subrutina que lea cada uno de los vectores que constituyen la matriz, así como el vector término independiente.
- Implementa una subrutina que realice la factorización y otra subrutina que realice la resolución del sistema lineal.
- Mediante otra subrutina, escribe en un fichero las matrices L y U y por pantalla la solución obtenida.

Método de Cholesky

- Crea una subrutina que lea la matriz de coeficientes y el término independiente.
- Implementa una subrutina que realice la factorización y otra subrutina que resuelva los sistemas triangulares correspondientes.
- Mediante otra subrutina, escribe en un fichero la matriz L, tal que $A = LL^t$, y por pantalla la solución obtenida.

Cuestiones generales

1. El programa principal dispondrá de un menú de selección del método de resolución: LU para matrices tridiagonales o Cholesky.
2. Las subrutinas pueden ser externas o bien ir incluidas en módulos.
3. Verifica el funcionamiento de los algoritmos con ejemplos de solución conocida.

Aplicación:
La ecuación de Poisson

$$-T'' = f(x)$$

describe la distribución de la temperatura T en una barra unidimensional de longitud L, donde f es una función que define una fuente de calor a lo largo de la barra, y además los extremos de la barra se mantienen a temperaturas fijas: $T(0) = T_1$ y $T(L) = T_2$.

Para resolver la ecuación anterior se subdivide el intervalo $[0, L]$ en $n+1$ subintervalos de igual longitud,

$$\Delta x = \frac{L}{n+1}, \qquad x_j = j * \Delta x \text{ con } j = 0, \ldots, n+1$$

El objetivo es buscar una solución aproximada en los nodos x_j, $T_j \approx T(x_j)$, $j = 1 \ldots, n$. Para ello, hay que resolver un sistema de ecuaciones lineales $Au = b$, donde

$$A = \begin{pmatrix} 2 & -1 & 0 & \ldots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \vdots & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \ldots & 0 & -1 & 2 \end{pmatrix} \quad u = \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_{n-1} \\ T_n \end{pmatrix} \quad b = \Delta x^2 \begin{pmatrix} f(x_1) + \frac{T_0}{\Delta x^2} \\ f(x_2) \\ \ldots \\ f(x_{n-1}) \\ f(x_n) + \frac{T_{n+1}}{\Delta x^2} \end{pmatrix}$$

Problema:

Aproxima la temperatura a lo largo de una barra de 10 cm con las siguientes condiciones fronteras, $T(0) = 40$ y $T(10) = 200$ y una fuente uniforme de calor $f(x) = 10$.

- Se pide resolver el sistema resultante (independientemente del número de nodos empleados) mediante los dos métodos implementados en el apartado anterior.
- Pedir por teclado la longitud del intervalo, la temperatura en los extremos de la barra ($T(0)$ y $T(L)$), el número de nodos ($n+2$) que se quieren emplear y el método que se quiere utilizar para resolver el correspondiente sistema.
- En este caso particular se conoce la solución exactaa de la ecuación:

$$T(x) = -5x^2 + 66x + 40$$

- Calcular el error relativo que se comete para: $n = 5$, $n = 50$ y $n = 500$. Escribir los resultados obtenidos en un fichero: número de nodos, error cometido y vector solución.
- Representar en un gráfica la solución exacta y la solución aproximada obtenida en cada uno de los casos.

Parte 2. Ecuaciones no lineales. Método de Newton

Para aproximar una raíz compleja de una ecuación no lineal, emplearemos el algoritmo de Newton. Para ello, introduciremos en el código la función que define la ecuación y su derivada.

En primer lugar programamos el algoritmo para aproximar raíces reales y, una vez probado su buen funcionamiento, creamos la subrutina **Newtoncomplejo** para la aproximación a una raíz compleja partiendo de un número complejo, no real, inicial z_0.

- Se pedirán por teclado: la aproximación inicial, el número máximo de iteraciones y la tolerancia de error relativo.
- Mediante una subrutina escribe en un fichero los datos empleados: aproximación inicial, el número máximo de iteraciones y la tolerancia. Además, en el mismo fichero, se mostrará para cada iteración, la aproximación obtenida y el error relativo asociado.

Ejecución: La ecuación $z^3 - 1 = 0$ tiene tres soluciones en los números complejos. Aproxima los tres valores partiendo de aproximaciones iniciales distintas con un error relativo inferior a 10^{-9}.

Memoria
Consta de los siguientes apartados, debidamente comentados:

1. Código fuente: Listado de los programas elaborados.
2. Tests, donde figuren los diferentes tests realizados para validar los algoritmos y las soluciones exactas.
3. Tabla resumen de la ejecución de la aplicación mediante todos los métodos implementados.

TIEMPO DE REALIZACIÓN: 5 semanas