COMPUTACIÓN II

PRÁCTICA 4 evaluable (clase 9)

Método de factorización LU en sistemas tridiagonales

Dada la ecuación matricial ***A·x=f,*** donde la matriz **A** es tridiagonal, y que está expresada por:

$$\begin{bmatrix} 4 & -1 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 4 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 & \vdots \\ 0 & \vdots & \ddots & \ddots & 4 & -1 & 0 \\ 0 & 0 & \ddots & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} 100 \\ 200 \\ 200 \\ \vdots \\ 200 \\ 200 \\ 100 \end{bmatrix}$$

a) Dado un n, escribe una función que genere el fichero de coeficientes y términos independientes siguiendo el formato dado arriba. Genera un sistema con n=1000.
b) Genera una función que compruebe si una matriz es tridiagonal.
c) Programa una función con el algoritmo LU específico para sistemas tridiagonales. La función debe trabajar sólo con los elementos no nulos de las 3 diagonales de la matriz. Pásalos en forma de 3 vectores o de matriz de n*3 elementos.
d) Realiza un programa principal para resolver el sistema de ecuaciones que has generado, guardando tu resultado en un fichero externo.

EXTRA: Compara los tiempos de ejecución utilizando el algoritmo LU general y el específico para matrices tridiagonales. Puedes utilizar la librería “chrono”:

```
#include <chrono>
using namespace chrono;
```

```
…
time_point<system_clock> start, end;
duration<double> elapsed_seconds;
…
start = system_clock::now();
...
end = system_clock::now();
elapsed_seconds = end - start;
cout<<"* Tiempo LU (s)=" << elapsed_seconds.count() << endl;
```