TP n°3: Systèmes d'équations
Méthode pour matrices tridiagonales

S2 Maths pour l'Informatique Saint-Étienne 2019-2020

1 Position du problème

On considère le système d'équations

$$M \cdot X = Y \tag{1}$$

où Y est une colonne donnée

$$Y = \begin{pmatrix} y_1 \\ \vdots \\ y_j \\ \vdots \\ y_n \end{pmatrix}$$

appelée **second membre**, X est une colonne d'inconnues

$$X = \begin{pmatrix} x_1 \\ \vdots \\ x_j \\ \vdots \\ x_n \end{pmatrix}.$$

On suppose de plus que M est une matrice carrée d'ordre n qu'on suppose inversible, et qui a la forme particulière suivante :

$$M = \begin{pmatrix} b_1 & c_1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ldots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & a_n & b_n \end{pmatrix}.$$

L'objectif de ce TP est d'aboutir à un algorithme spécifique pour ce type de matrices.

2 Stockage de M sous forme de vecteurs colonnes

On pose par convention :

$$a_1 = c_n = 0, \tag{2}$$

de sorte que pour avoir la matrice M il suffit d'avoir 3 colonnes A, B et C de dimension n, qui contiennent les coefficients non nuls de la matrice M. L'extraction de ces vecteurs colonnes à partir de M peut se faire avec la fonction `diag` de MATLAB.

Vous commencerez donc par écrire une petite fonction :

```
function [A,B,C] = extract_coefs_tridiag(M)
```

qui réalise cette opération et qui sera utilisée par la fonction que vous programmerez à l'étape suivante.

3 Méthode de Richtmeyer ou des "e_i, f_i"

Soit les coefficients (e_i) et (f_i) définis par la relation récurrente

$$\begin{cases} e_1 = b_1, & f_1 = \dfrac{c_1}{b_1}, \\ ei_= b_i - a_i \cdot f_{i-1}, & f_i = \dfrac{c_i}{b_i}, \quad i = 2 \dots n. \end{cases}$$

Notons que ces termes sont tous bien définis grâce à (2). Vérifier alors que M s'écrit comme le produit de deux matrices triangulaires $M = L \cdot U$, avec

$$L = \begin{pmatrix} e_1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ a_2 & e_2 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_{n-1} & e_{n-1} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & a_n & e_n \end{pmatrix}, \qquad U = \begin{pmatrix} 1 & f_1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & f_2 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & f_{n-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix}.$$

Déduisez de ce qui précède la programmation d'une fonction :

```
function [L,U] = fact_richtmeyer(M)
```

qui construit les matrices L et U.

4 Résolution du système

A partir des résultats précédents, montrez que le système (1) est équivalent à deux systèmes triangulaires que vous expliciterez. Vous programmerez ensuite une fonction

```
function [X,L,U] = resol_tridiag(M,Y)
```

et qui rend aussi les matrices L et U car elles sont réutilisables pour d'autres résolutions si on modifie le second membre Y.

Vous trouverez dans l'ENT un fichier `construc_mat.m` qui contient une fonction :

```
function [M] = construc_mat(n)
```

permettant de construire une matrice d'ordre n qui vous servira d'exemple.