Exercice 1

Soient $\mathbf{D}$, $\mathbf{L}$ et $\mathbf{U}$ trois matrices inversibles de $\mathcal{M}_n(\mathbb{C})$, respectivement diagonale, triangulaire inférieure et triangulaire supérieure. Soit $b \in \mathbb{C}^n$.

1. *(algo)* Résoudre $\mathbf{D}x = b$ et écrire l'algorithme (fonction $\textsc{ResDiag}$) permettant de résoudre ce système. Calculer le coût (évaluer le nombre d'opérations élémentaires) de cet algorithme.
2. *(algo)* Résoudre $\mathbf{L}x = b$ et écrire l'algorithme (fonction $\textsc{ResTriInf}$) permettant de résoudre ce problème. Calculer le coût de cet algorithme.
3. *(algo)* Résoudre $\mathbf{U}x = b$ et écrire l'algorithme (fonction $\textsc{ResTriSup}$) permettant de résoudre ce problème. Calculer le coût de cet algorithme.
4. Soit $\mathbf{A} = \mathbf{LU}$. Expliquer comment résoudre $\mathbf{A}x = b$ et écrire l'algorithme (fonction $\textsc{ResFactLU}$) permettant de résoudre ce problème. Calculer le coût de cet algorithme.

1.

$$\mathbf{D}x = b \quad \Leftrightarrow \quad (D)_{ii}x_i = b_i \ \ \forall i \in [1:n] \quad \Leftrightarrow \quad x_i = \frac{b_i}{D_{ii}} \quad \forall i \in [1:n].$$

Pour tout $i \in [1:n]$, $D_{ii} \neq 0$ parce que la matrice $\mathbf{D}$ est inversible.

```
Fonction ResDiag(D,b) : x
    [D est une matrice carrée diagonale
    b est un vecteur colonne
    x est un vecteur colonne]

    n ← length(b)
    x ← zeros( length(b),1) ; [On initialise le vecteur x a 0 (vecteur nul de n lignes et 1 colonne)]

    Pour i de 1 à n faire
        x(i) ← b(i)/D(i,i)
    Fin Pour
    Retourner x ;
Fin
```

Algorithme 1: Algorithme pour la résolution du système diagonal $\mathbf{D}x = b$

Le coût de l'algorithme est de n multiplications (ou divisions).

2. Comme L est triangulaire inférieure, on peut calculer facilement x en utilisant la méthode de substitution. En effet, pour tout $i \in [1:n]$, $L_{ij} = 0$ pour $j > i$, si bien que

$$\mathbf{L}x = b \quad \Leftrightarrow \quad \sum_{j=1}^{n} L_{ij}x_j = b_i \quad \Leftrightarrow \quad \sum_{j=1}^{i} L_{ij}x_j = b_i,$$

ou encore, puisque la matrice $\mathbf{L}$ est inversible (et donc que $L_{ii} \neq 0 \ \forall i \in [1:n]$), pour tout $i \in [1:n]$,

$$x_i = \frac{1}{L_{ii}} \left(b_i - \sum_{j=1}^{i-1} L_{ij}x_j \right). \tag{1}$$

Pour calculer x, on procède de la manière suivante : on commence par calculer $x_1 = b_1/L_{11}$. Puis, on substitue x_1 dans la deuxième équation ((2) pour $i = 2$), on obtient $x_2 = \frac{1}{L_{22}}(b_2 - L_{21}x_1)$. Ensuite, on substitue x_1 et x_2 dans la troisième équation ((2) pour $i = 3$), et on obtient x_3. A l'étape $k \leq n$, on substitue x_1, x_2, $\cdots$, x_{k-1} (déjà calculés) dans l'équation (2) pour $i = k$ et on obtient x_k.

```
Fonction ResTriInf(L,b) : x
    [L est une matrice carrée triangulaire inférieure (inversible)
    b est un vecteur colonne
    x est un vecteur colonne]

    n ← length(b)
    x ← zeros( length(b),1) ; [On initialise le vecteur x a 0 (vecteur nul de n lignes et 1 colonne)]

    [boucle pour le calcul de x(i) :]
    Pour i de 1 à n faire
        temp ← 0 [temp est une variable temporaire initialisée à 0]
        [boucle pour le calcul de temp = ∑_{j=1}^{i-1} L(i,j)x(j)]
        Pour j de 1 à i-1 faire
            temp ← temp + L(i,j) x(j)
        Fin Pour
        x(i) = (b(i) - temp)/L(i,i)
    Fin Pour
    Retourner x ;
Fin
```

Algorithme 2: Algorithme pour la résolution du système triangulaire inférieur $\mathbf{L}x = b$

Calculons le coût de l'algorithme. Pour chaque $i \in [1:n]$, le calcul de la variable temp (boucle intérieure) nécessite $i-1$ multiplications et $i-1$ additions. Le calcul de $x(i)$ requiert donc i multiplications et i additions. Finalement, le nombre total de multiplications n^*_{inf} est donc de

$$N^*_{\mathsf{inf}} = \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \sim \frac{n^2}{2}.$$

De manière similaire, le nombre total d'additions N^+_{inf} est donc de

$$N^+_{\mathsf{inf}} = \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \sim \frac{n^2}{2}.$$

3. Comme U est triangulaire supérieure, on peut calculer facilement x en utilisant la méthode de substitution, mais en commençant cette fois par calculer x_n. En effet, pour tout $i \in [1:n]$, $U_{ij} = 0$ pour $j < i$, si bien que

$$\mathbf{L}x = b \quad \Leftrightarrow \quad \sum_{j=i}^{n} U_{ij} x_j = b_i,$$

ou encore, puisque $\mathbf{U}$ est inversible ($\forall i \in [1:n], U_{ii} \neq 0$) pour tout $i \in [1:n]$,

$$x_i = \frac{1}{U_{ii}} \left(b_i - \sum_{j=i+1}^{n} U_{ij} x_j \right) \tag{2}$$

Pour calculer x, on procède de la manière suivante : on commence par calculer $x_n = b_n / U_{nn}$. Puis, on substitue x_n dans l'equation (2) pour $i = n-1$, on obtient $x_{n-1} = \frac{1}{U_{n-1,n-1}} (b_{n-1} - U_{n-1,n} x_n)$. Ensuite, on substitue x_n et x_{n-1} dans l'équation (2) pour $i = n-2$, et on obtient x_{n-2}. À l'étape $k \leq n$, on substitue x_n, x_{n-1}, $\cdots$, x_{n-k+2} (déjà calculés) dans l'équation (2) pour $i = n-k+1$ et on obtient x_{n-k+1}.

```
Fonction ResTriSup(U,b) : x
    [U est une matrice carrée triangulaire supérieure (inversible)
    b est un vecteur colonne
    x est un vecteur colonne]

    n ← length(b)
    x ← zeros( length(b),1) ; [On initialise le vecteur x a 0 (vecteur nul de n lignes et 1 colonne)]

    [boucle pour le calcul de x(n − i + 1)]
    Pour i de 1 à n faire
        temp ← 0 [temp est une variable temporaire initialisée à 0]
        [boucle pour le calcul de temp = ∑_{j=n−i+2}^{n} U(i, j)x(j) :]
        Pour j de n-i+2 à n faire
            temp ← temp + U(i,j) x(j)
        Fin Pour
        x(n-i+1) = (b(n-i+1) - temp)/U(n-i+1,n-i+1)
    Fin Pour
    Retourner x ;
Fin
```

Algorithme 3: Algorithme pour la résolution du système triangulaire supérieur $\mathbf{U}x = b$

Tout comme dans le cas de la résolution d'un système triangulaire inférieur, le nombre total de multiplications N^{*}_{sup} et le nombre total d'additions N^{+}_{sup} sont de l'ordre de n^2.

4.

$$\mathbf{A}x = b \quad \Leftrightarrow \quad \mathbf{LU}x = b \quad \Leftrightarrow \quad (\mathbf{L}y = b \text{ et } \mathbf{L}x = y)$$

Donc, pour résoudre $\mathbf{A}x = b$, il suffit de résoudre d'abord le système triangulaire inférieur $\mathbf{L}y = b$ (pour obtenir le vecteur y) puis le système triangulaire supérieur $\mathbf{U}x = y$. Le nombre total d'additions et de multiplications sont donc de l'ordre de n^2.

```
Fonction ResTriSup(L,U,b) : x
    [L est une matrice carrée triangulaire inférieure (inversible)
    U est une matrice carrée triangulaire supérieure (inversible)
    b est un vecteur colonne
    x est un vecteur colonne
    cet algorithme calcul x solution du systeme LUx = b]

    y ← ResTriInf(L, b) [Resolution du système triangulaire inférieur Ly = b]

    x ← ResTriInf(U, y) [Resolution du système triangulaire supérieur Ux = y]

    Retourner x ;
Fin
```

Algorithme 4: Algorithme pour la résolution du système $\mathbf{A}x = \mathbf{LU}x = b$