DS 03 - Numérique - Correction

1 Représentation des nombres

EXERCICE 1 — *Sur n bits, on peut représenter $m = 2^n$ valeurs différentes.*

— *Il y a 26 lettres dans l'alphabet, 10 chiffres et 201 années (de 1900 à 2100 inclus), soit au total $m = 26^2 \times 10^3 \times 26^2 \times 201 = 91852176000$ possibiltés. Il faut donc $n = \log_2(m) \approx 36.42$ bits. On divise par 8 pour obtenir le nombre d'octets nécessaires (partie entière supérieure) : 5 octets.*

2 Equations différentielles

EXERCICE 2 — *Le schéma d'Euler explicite est :*

$$y_{k+1} = y_k + (t_{k+1} - t_k) \times F(y_k, t_k)$$

— *Pour une équation différentielle d'ordre supérieur, on se ramène à une équation du premier ordre en vectorialisant l'équation différentielle (ou le système d'équations) : On pose $Y = \begin{pmatrix} y \\ y' \\ y'' \end{pmatrix}$ (par exemple pour une équation différentielle d'ordre 3), puis on exprime Y' en fonction de Y et t. On a alors le schéma d'Euler :*

$$Y_{k+1} = Y_k + (t_{k+1} - t_k) \times F(Y_k, t_k)$$

EXERCICE 3 — *En utilisant l'énoncé, on a $\frac{d(S+I+R)}{dt} = -\beta IS + \beta IS - \frac{1}{\lambda} I + \frac{1}{\lambda} I = 0$, donc la quantité $S(t) + I(t) + R(t)$ est constante au cours du temps.*

— *$Y = \begin{pmatrix} S \\ I \\ R \end{pmatrix}$. On a directement : $F(Y,t) = Y' = \begin{pmatrix} S' \\ I' \\ R' \end{pmatrix} = \begin{pmatrix} -\beta IS \\ \beta IS - \frac{I}{\lambda} \\ \frac{I}{\lambda} \end{pmatrix}$. On a le code python suivant :*

```
def F(Y, t):
    S, I, R = Y
    return (-beta*I*S, beta*I*S-I/lam, I/lam)
```

EXERCICE 4 *La fonction* `euler(F, y0, temps)` *est supposée connue :*

```
temps = np.linspace(0, 1, 300)   # On a pris un nombre de points arbitraire (ici 300)
y0 = (999, 1, 0)                 # 999 individus sains au départ, et un malade
solution = euler(F, y0, temps)
plt.plot(temps, solution)        # trace les 3 courbes en fonction du temps
```

3 Intégration

Soit f une fonction de $\mathbb{R}$ dans $\mathbb{R}$. On travaille sur un intervalle $[a;b]$ où f est intégrable.

EXERCICE 5 — *Mathématiquement, on a tout simplement $m = \frac{1}{b-a} \times \int_a^b f(t).dt$*

— *Les subdivisions σ_k sont définies par $\sigma_k = a + k \times \frac{b-a}{n}$. On a $\sigma_0 = a$ et $\sigma_n = b$*

EXERCICE 6 *On calcule la somme de Rieman à gauche, associée à la sudivision $(\sigma_k)_{k \in [\![0;n-1]\!]}$:*

$$R_n = \frac{b-a}{n} \sum_{k=0}^{n-1} f(\sigma_k)$$

```
def trapezes(f, a, b, n):
    somme, h = 0, (b-a)/n
    for k in range(0, n):  # n exclu
        sigma = a + k*h
        somme += f(sigma)
    return h*somme
```

4 Sécante

La méthode de la sécante permet de trouver une solution approchée de l'équation $f(x) = 0$ en partant de deux réels a et b "pas trop éloignés" d'une solution x_0. On définit la suite $(u_n)_{n\in\mathbb{N}}$ par $u_0 = a$, $u_1 = b$ et pour $n \geq 1$, u_{n+1} est l'abscisse de l'intersection entre l'axe des x et la sécante au graphe de f passant par les points d'abscisses u_{n-1} et u_n.

Dans la suite, on supposera qu'on est bien dans un cas où $f(u_n) \neq f(u_{n-1})$ et $u_n \neq u_{n-1}$ pour tout n.

EXERCICE 7 *[Dessin...]*

- *La sécante a pour équation de droite $y = mx + p$, et passe par $(x_0, y_0) = (u_n, f(u_n))$*
- *Le coefficient directeur de la secante est $m = \frac{f(u_n)-f(u_{n-1})}{u_n-u_{n-1}}$*
- *On cherche l'intersection de cette droite avec l'axe des abscisses : soit $y = 0$ (la sécante passe par le point $(u_{n+1}, 0)$)*
- *Il suffit de conclure*

EXERCICE 8 *Pour $f(x) = x^2 - 2$, $a = 1$ et $b = 2$, calculer les valeurs de u_2 et u_3*

Appliquer la formule donnée : obtient $u_2 = \frac{4}{3}$ et $u_3 = \frac{7}{5}$

EXERCICE 9 *On peut s'arrêter quand $|u_n - u_{n-1} \leq \varepsilon$.*

```
def secantes(f, u0, u1, epsilon):
    while abs(u0 - u1) > epsilon:
        u0, u1 = u1, u1 - (u1-u0) / (f(u1)-f(u0)) + f(u1)
    return (u0 + u1)/2
```

5 Pivot de Gauss - questions faciles

EXERCICE 10

- *La formule mathématique utilisée est bien la bonne, mais la matrice A vient d'être modifiée à la ligne du dessus!*

```
mu = -A[i][j]/A[j][j]     # calculé une fois pour toute
transvection(A, i, j, mu)
transvection(B, i, j, mu) # c'est bien la même formule / la même valeur
```

- *Pour obtenir l'inverse d'une matrice carrée inversible A, il suffit de calculer* `gauss(A, I)` *où* `I` *est la matrice identité de même ordre que A. On obtiendra comme résultat le couple (I, B) avec $B = A^{-1}$*

EXERCICE 11 *Attention : dans ce genre de question, il faut vraiment appliquer le code donné dans l'énoncé, et pas seulement la méthode vue en cours de maths. On obtient successivement les matrices :*

- $j = 0$
  - *Échange : $L_0 \leftrightarrow L_1$*
  - $A = \begin{pmatrix} 3 & 4 \\ 1 & 2 \end{pmatrix}$; $B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
  - *Dilatation : $L_0 \leftarrow 0.3333333333333333 \times L_0$ (Évidement, vous pouvez arrondir un peu hein)*
  - *Transvection : $L_1 \leftarrow L_1 + -1.0 \times L_0$*
  - $A = \begin{pmatrix} 1.0 & 1.3333333333333333 \\ 0.0 & 0.6666666666666667 \end{pmatrix}$; $B = \begin{pmatrix} 0.0 & 0.3333333333333333 \\ 1.0 & -0.3333333333333333 \end{pmatrix}$
- $j = 1$
  - *Échange : $L_1 \leftrightarrow L_1$*
  - $A = \begin{pmatrix} 1.0 & 1.3333333333333333 \\ 0.0 & 0.6666666666666667 \end{pmatrix}$; $B = \begin{pmatrix} 0.0 & 0.3333333333333333 \\ 1.0 & -0.3333333333333333 \end{pmatrix}$
  - *Transvection : $L_0 \leftarrow L_0 + -1.9999999999999998 \times L_1$*
  - *Dilatation : $L_1 \leftarrow 1.4999999999999998 \times L_1$*
  - $A = \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{pmatrix}$; $B = \begin{pmatrix} -1.9999999999999998 & 0.9999999999999998 \\ 1.4999999999999998 & -0.4999999999999999 \end{pmatrix}$

EXERCICE 12 *Écrire la fonction* `pivot_partiel(A, j)`.

```
def pivot_partiel(A, j):
    res = j
    lignes, _ = dimensions(A)
    for i in range(j, lignes):
        if abs(A[i][j]) > abs(A[res][j]):
            res = i
    return res
```