Dans ce TP, on présente la transformation bilinéaire qui permet de donner une solution approximative des équations différentielles.

1 Problèmes

On considère un système (par exemple un circuit électronique) recevant $e(t)$ en entrée et produisant $s(t)$ en sortie.

$$e(t) \longrightarrow \boxed{\text{Système}} \longrightarrow s(t)$$

Le système obéit à une certaine équation différentielle.

$$(\mathcal{E}) : \quad \tau \cdot s' + s = e$$

On sait également que $s(0) = 0$.

2 Principe

Passant l'équation différentielle dans le domaine de Laplace on obtient

$$\tau\, p \cdot S(p) + S(p) = E(p) \Rightarrow S(p) = \frac{1}{1 + \tau\, p}\, E(p)$$

On déduit que la fonction de transfert du système est $H(p) = \frac{1}{1+\tau\, p}$.

La transformation bilinéaire consiste à chercher une version en z de $H(p)$. On pourra noter $\hat{H}(z)$ cette fonction. On pose, considérant la période d'échantillonnage T_e :

$$\hat{H}(z) = H\left(\frac{2}{T_e}\,\frac{1 - z^{-1}}{1 + z^{-1}}\right)$$

Dans le cadre d'une simulation sur ordinateur, au lieu de fixer T_e, on choisit la durée T de l'expérience et le nombre N d'échantillons de sorte que $T_e = \frac{T}{N}$.

(1) Exprimez $\hat{H}(z)$ en fonction de T_e, z, τ.

(2) Déduisez-en la récurrence entre l'entrée e_n et la sortie s_n.
Cette récurrence dépend de T_e et τ.

Bien que tout cela devrait conduire à $s_0 \neq 0$, on impose $s_0 = 0$ car l'expérience montre que les résultats sont meilleurs.

3 Tableur

À chaque fois, vous pensez à étendre les formules jusqu'en bas.

(1) Ouvrez le tableur et recopier ce tableau

	A	B	C	D	E	F
1	tau	2	T	10	N	1000
2						
3	n	t0	e(n)	s(n)		
4	0			0		
5	1					

Il faut étendre la colonne A pour que n aille jusque 1000

(2) Écrivez en B4 la formule permettant de calculer le temps t_0 en fonction de n et $T_e = \frac{T}{N}$.

(3) Pour e on prendra $e(t) = \begin{cases} 1 \text{ si } 0 \leqslant t \leqslant 2.5 \\ 0 \text{ sinon} \end{cases}$ Complétez la colonne C.

(4) En D5, écrivez la récurrence trouvée précédemment.

(5) Tracez les courbes de e et s avec t en abscisse. Vous pourrez ensuite modifiez la valeur de τ pour voir l'effet.

4 Code Python

Les numéros à gauche sont les numéros de ligne. Ne pas les recopier.

(1) On commence par importer le module graphique

```
import matplotlib.pyplot as plt # graphique
```

(2) On définit T et N ce qui permet de calculer dt.

```
T = 10
N = 1000
dt = T/N
```

(3) On fournit les paramètres de l'équation : il faut indiquer la valeur de τ et l'expression de $e(t)$. On reprend le même que précédemment.

```
tau = 2
def e(t):
    if 0 <= t <= 2.5:
        return 1
    else:
        return 0
```

(4) On va utiliser des **tableaux**. Ces tableaux sont des listes de nombres que l'on va compléter au fur et à mesure. Les tableaux sont initilisés avec []

```
liste_t = [0]     # liste des temps
liste_e = [e(0)]  # liste des valeurs de e
liste_s = [0]     # liste des valeurs de s
```

(5) Pour ajouter une valeur à une liste, on utilise append. Quand on veut lire le dernier élément d'une liste, on utilise [−1]. Pour parcourir tous les instants de la simulation, on utilise une boucle for.

```
for i in range(N): # répète N fois
    nouveau_s = ...
    nouveau_t = ...
    nouveau_e = ...
    liste_s.append(nouveau_s)
    liste_t.append(nouveau_t)
    liste_e.append(nouveau_e)
```

Complétez les ...

(6) Il ne reste qu'à représenter les courbes.

```
plt.plot(liste_t, liste_e, 'r', label='entrée')
plt.plot(liste_t, liste_s, 'b', label='sortie')
plt.legend()
plt.show()
```

Exécutez pour visualiser les courbes.

Vous pouvez répéter l'opération en modifiant des paramètres comme τ.