



**Année universitaire**  
**2025 - 2026**

---

**Signaux analogiques et  
numériques — 3ETI**  
**Séance pratique 4**

**Transformée de Fourier discrète**

---

**Serge Mazaauric**  
**Eric Van Reeth**

## Présentation de la séance

### Contexte et objectifs

Ce travail pratique a pour objectif de mettre en application les outils vus en cours pour la mise en œuvre d'une analyse spectrale de signaux discrets. Cette deuxième séance utilisera des signaux synthétiques pour prendre en main les concepts associés au cours. Nous les appliquerons par la suite sur des signaux réels.

### Compétences associées

À l'issue de cette séance vous serez capables :

- de maîtriser une librairie de calcul scientifique sous Python pour l'analyse spectrale d'un signal
- d'afficher correctement le spectre d'un signal discret, en fonction d'un axe fréquentiel cohérent
- d'identifier les composantes fréquentielles dominantes d'un signal discret
- de comprendre et réduire les effets de l'échantillonnage temporel sur l'analyse spectrale
- de comprendre l'intérêt du fenêtrage temporel pour l'analyse spectrale

### Évaluation

Une évaluation orale unique aura lieu en fin de séance (15 minutes par binôme) **à partir du prochain TP**. La séance pendant laquelle chaque binôme est évalué est choisie aléatoirement. De ce fait, **il est impératif de garder les mêmes binômes pour toutes les séances**.

### Import des librairies nécessaires

```
import numpy as np
import scipy.signal as signal
import plotly.express as px
import pandas as pd # pour les dataframes
import sounddevice as sd # pour jouer les sons associés aux signaux
```

## 1 Fenêtrage temporel

1. Décrire le fonctionnement de la fonction `np.fft.fftshift()`, et son intérêt pour l'analyse spectrale. Déterminer la transformation à appliquer sur l'axe fréquentiel pour afficher de manière cohérente le spectre obtenu après l'application de la fonction `np.fft.fftshift()`.
2. Grâce aux fonctions `signal.windows.hann()` et `signal.windows.boxcar()`, créer deux séquences `wH` et `wB` composées de  $K = 32$  points correspondant respectivement aux fenêtres de Hann et rectangulaire.  
Afficher ces deux fenêtres en fonction du temps.
3. Calculer la valeur moyenne des 2 fenêtres précédentes (`np.mean()`). En déduire  $G$ , le facteur d'amplitude à appliquer sur `wH` pour que les deux fenêtres aient la même moyenne.
4. Affichez sur un même graphe en fonction des fréquences réduites, les spectres **centrés** des deux fenêtres de même moyenne calculés sur  $N = 4K$  points. Retrouver à partir de ces tracés les caractéristiques de chacune de ces deux fenêtres (cf. TD Signaux continus Ex. 10). En déduire leurs avantages/inconvénients pour l'analyse spectrale (amplitude des rebonds et résolution fréquentielle).
5. Créer maintenant la séquence suivante :

$$s_B[k] = A_1 \cos(2\pi\nu_1 t) + A_2 \cos(2\pi\nu_2 t)$$

avec  $\nu_e = 1$  kHz,  $A_1 = 2$ ,  $\nu_1 = 82.35$  Hz,  $A_2 = 0.1$ ,  $\nu_2 = 100.15$  Hz, et  $K = 256$  points.

Puis créer la séquence fenêtrée avec Hann :

$$s_H[k] = s_B[k] \cdot G \cdot w_H[k]$$

6. Afficher sur un même graphe les deux signaux temporels en fonction du temps, puis sur un autre graphe les spectres **centrés** de  $s_W[k]$  et  $s_B[k]$ . Notez le nombre de pics d'amplitude pour chaque courbe, et relevez les fréquences auxquelles ces pics apparaissent. Commentez.
7. Refaire la même étude avec les paramètres suivants :  
 $\nu_e = 1$  kHz,  $A_1 = 2$ ,  $\nu_1 = 88.65$  Hz,  $A_2 = 2$ ,  $\nu_2 = 100.15$  Hz, et  $K = 64$  points. Notez le nombre de pics d'amplitude pour chaque courbe, et relevez les fréquences auxquelles ces pics apparaissent. Commentez.

## 2 Analyse temps-fréquence

Nous nous intéresserons dans cette partie à l'analyse temps-fréquence d'un signal non stationnaire, i.e. dont le contenu fréquentiel varie dans le temps. Nous utiliserons pour cela la classe `ShortTimeFFT` de la librairie `scipy.signal`.

### 2.1 Création du signal

La séquence étudiée dans cette partie sera composée de 3 séquences mises bout à bout :

- Une séquence monochromatique, d'expression :

$$s_1(t) = \cos(2\pi\nu_1 t)$$

- Une séquence polychromatique, d'expression :

$$s_2(t) = \cos(2\pi\nu_2 t) + \cos(2\pi\nu_3 t)$$

- Une séquence contenant un « *chirp* linéaire », d'expression :

$$s_3(t) = \cos(2\pi\nu_4 t^2)$$

1. Créer les 3 séquences `s1`, `s2` et `s3` avec les paramètres suivants :

- Durée de chaque séquence : 1 s
- Fréquence d'échantillonnage :  $\nu_e = 1$  kHz
- Fréquences des signaux :  $\nu_1 = 100$  Hz,  $\nu_2 = 50$  Hz,  $\nu_3 = 200$  Hz et  $\nu_4 = 200$  Hz

2. Grâce à la commande `s = np.concatenate((s1, s2, s3))`, créer la séquence finale et afficher-la en fonction du temps.

## 2.2 Spectrogramme

1. Tracer le spectre centré de la séquence `s` en fonction des fréquences réelles et analyser son contenu. Commenter.
2. Rappeler le principe de construction d'un spectrogramme, et les caractéristiques du signal qu'il permet de visualiser.
3. À partir de l'aide de la classe `ShortTimeFFT` (cliquer [ici](#)), identifier la signification des paramètres d'entrée suivants : `win`, `hop`, `fs`, et `mfft`.
4. À partir maintenant de la méthode `extent` et des attributs `delta_t`, et `f`, déterminer comment obtenir les vecteurs temporel (`t_spec`) et fréquentiel (`nu_spec`) associés au spectrogramme.
5. Créer deux fenêtres (porte `wB`, et Hann `wH`), sur  $K = 64$  points grâce aux fonctions vues dans la partie précédente. Créer ensuite une instance de la classe `ShortTimeFFT`, puis calculer le spectrogramme grâce aux commandes suivantes :

```
# Paramétrisation des 2 spectrogrammes (1 par fenêtre)
SFT_B = signal.ShortTimeFFT(win=wB, hop=K, fs=nue, mfft=K)
SFT_H = signal.ShortTimeFFT(win=wH, hop=K, fs=nue, mfft=K)
# Calcul des spectrogrammes de s
spectrogram_B = SFT_B.stft(s)
spectrogram_H = SFT_H.stft(s)
```

Afficher les 2 spectrogrammes obtenus en fonction du temps et des fréquences à partir des vecteurs déterminés à la question 4, et la commande suivante :

```
# Affichage d'un spectrogramme
px.imshow(np.abs(spectrogram_B), x = t_spec, y = nu_spec,
↪ origin='lower', aspect='auto', labels={'x': 'Temps (s)', 'y':
↪ 'Fréquence (Hz)', 'color': 'Magnitude'}, title='Spectrogramme de
↪ s').show()
```

6. Comparer l'allure des spectrogrammes obtenus à celle du spectre obtenu en début d'exercice, puis entre-eux.
7. Quels sont les 2 paramètres à modifier pour améliorer les **finesses d'analyse** temporelle et fréquentielle ? Appliquer ces changements jusqu'à trouver des valeurs satisfaisantes.

8. Quel paramètre influence la **résolution fréquentielle et temporelle** du spectrogramme ? Discuter l'influence de ce paramètre sur la résolution.
9. Tracer enfin le spectrogramme d'une séquence identique à la précédente, en changeant simplement la formule de  $s_3$  (mêmes signaux  $s_1$  et  $s_2$ ) :

$$s_3(t) = \cos(2\pi\nu_4 t^4)$$

Décrire le phénomène qui apparaît sur le spectrogramme. Vous pourrez utiliser l'option `fft_mode='centered'` lors de la paramétrisation du spectrogramme pour une visualisation plus complète.