Représentation fréquentielle des signaux

Gilles Menez - UCA - UFR Sciences - Dépt. Informatique

February 5, 2020

1 Signaux périodiques

1.1 Séries de Fourier

L'application que l'on va en faire des séries de Fourier s'énonce ainsi :

"Si $s(t)$ est une fonction de t périodique, de période $T_0 = 1/F_0$, elle peut s'écrire sous la forme d'**une somme de fonctions sinusoïdales de fréquence** $f = n * F_0$ **multiples de la fréquence** F_0 **dite fondamentale**"

Avec un formalisme mathématique, on écrit donc :

$$s(t) = a_0 + \sum_{n=1}^{\infty} (a_n \, cos(2\pi n F_0 t) + b_n \, sin(2\pi n F_0 t)) \tag{1}$$

où a_n et b_n sont les **coefficients de la série de Fourier**.

- ➢ **Cette équation traduit bien** qu'un signal $s(t)$ peut être écrit comme une combinaison de fonctions orthogonales **cos** et **sin**.

1.2 Spectre

$S(f)$, le **spectre** en fréquence du signal, est la représentation fréquentielle du signal temporel $s(t)$.

Ce spectre représente les composantes fréquentielles du signal temporel :

- ➢ l'amplitude de la **fondamentale** "$f = F_0$"
- ➢ **et** des différentes **harmoniques** en fonction de la fréquence "$f = nF_0$".

Le spectre $S(f)$ du signal périodique $s(t)$ s'écrit :

$$S(f) = \sum_{n=-\infty}^{\infty} S(nF_0).\delta(f - nF_0) \tag{2}$$

où δ représente un pic de Dirac.

Et avec $S(nF_0)$, un nombre complexe :

$$S(nF_0) = \frac{(a_n - j * b_n)}{2} = |S(nF_0)|e^{-j\phi(nF_0)} \tag{3}$$

Cette représentation spectale bilatérale $S(f)$ (fréquences négatives et positives) est abstraite et présente l'avantage de simplifier les calculs . . . donc **pas de panique** !

➢ **Il est important de remarquer que le spectre d'une fonction périodique (eq 2) est discontinu (à cause des diracs) et donc composé de raies dont l'écart minimum est, sur l'axe des fréquences**, F_0.

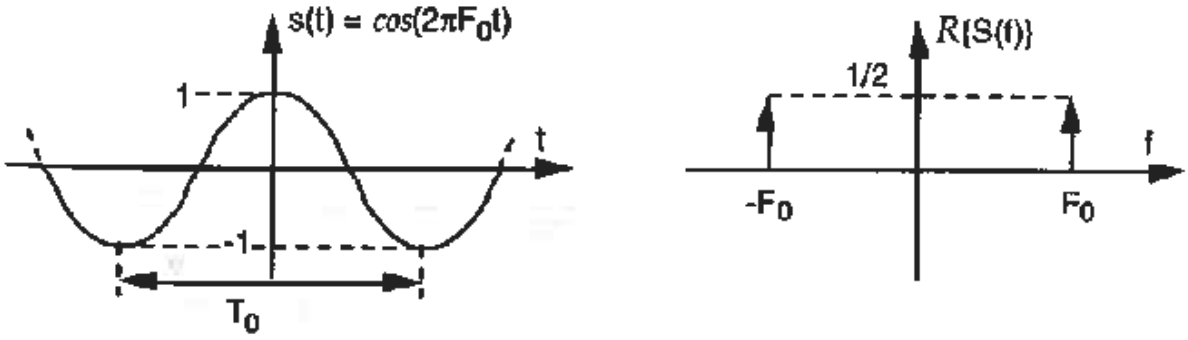
Les $S(nF_0)$ sont les composantes fréquentielles de $s(t)$. Cette grandeur (pour un n donné) est en générale complexe, et a donc pour module :

$$|S(nF_0)| = \frac{1}{2}.\sqrt{a_n^2 + b_n^2} \tag{4}$$

et pour phase :

$$\phi(S(nF_0)) = arctg(-\frac{b_n}{a_n}) \tag{5}$$

2 Exemple : sur un signal cosinusoïdal



① La partie gauche de cette figure représente **le signal temporel**

$$s(t) = cos(2\pi F_0 t) \tag{6}$$

② La partie droite de la figure représente **la partie réelle du spectre (bilatéral) du signal**.

Pour obtenir cette figure, on a identifié dans l'équation (eq 1) les a_n et les b_n qui permettraient d'obtenir $s(t)$ de l'équation (eq 6) :

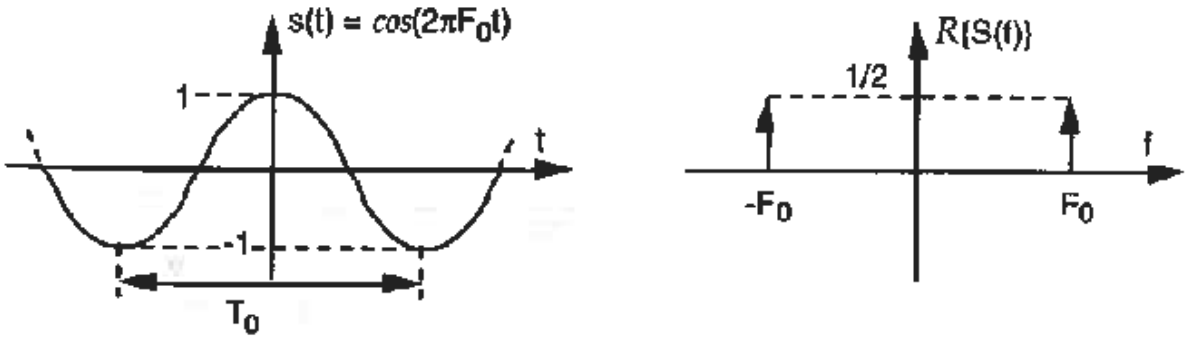
Et donc, les coefficients de Fourier permettant d'obtenir une projection de $s(t)$ dans le domaine des fréquences sont :

Pour les cos :	$a_0 = 0, \quad a_1 = 1, \quad$ et $a_n = 0$	pour $n > 1$
Pour les sin :	$b_n = 0,$	pour $n >= 0$

C'est la décomposition en séries de Fourier de $s(t)$.

Remarques :

- ✔ On a juste cherché les paramètres a_i et b_i permettant de rendre les deux équations (1 et 6) équivalentes.
- ✔ Il s'agit d'un spectre réel puisque les parties imaginaires (les b_n des $S(nF_0) = \frac{1}{2}(a_n - j * b_n)$) sont nulles pour tout n.



La figure est ainsi car d'aprés la décomposition, la seule harmonique non nulle est $n = 1$,

$$S(F_0) = \frac{1}{2}(a_1 - jb_1) = \frac{1}{2}$$
et
$$S(-F_0) = \frac{1}{2}(a_{-1} - jb_{-1}) = \frac{1}{2}$$

donc d'après (eq 2), le spectre s'écrit :

$$S(f) = \frac{1}{2}.\delta(f - F_0) + \frac{1}{2}.\delta(f + F_0)$$

soit, un dirac en F_0 d'amplitude $\frac{1}{2}$ et un autre en $-F_0$.

- ✔ Le signal n'est composé que d'une fréquence fondamentale ! . . . ce qui est normal . . . non ?

3 Voyons si c'est compris . . .

3.1 Retrouver le signal ?

Il s'agit dans cette question de retrouver le signal temporel représenté par son développement en série de Fourier (donc fréquentiel).

➟ On réalise alors la synthèse "additive" du signal à partir de sa série de Fourier.

3.1.1 Signal T ...

On vous donne la série de Fourier qui représente le signal $s(t)$ recherché :

$$s(t) = \frac{8 * A}{\pi^2} \sum_{n=0}^{+\infty} \frac{cos(2\pi(2n+1)F_0 t)}{(2n+1)^2} \tag{7}$$

(a) Pour ce signal, identifier/déduire les expressions de a_n et b_n présentes dans l'équation 1,

(b) Faire une (ou deux) fonction qui calcule ces a_n et b_n,

On rappelle que ces valeurs permettraient de déduire les raies du spectre $S(nF_0) = \frac{1}{2}(a_n - j * b_n)$.

(c) Dans l'hypothèse où le signal $s(t)$ est discrétisé temporellement :

- ➢ c'est à dire avec $t = k * T_e$
- ➢ et T_e la période d'échantillonnage

synthétiser ce signal $s(t)$ à partir de l'équation 1.

On choisira :

- ➢ $n \leq n_{max}$ et $n_{max} = 2 * 7$ pour commencer ...

Dit autrement, chaque échantillon $s(i * T_e)$ est obtenu en sommant les valeurs de plusieurs cosinus pondérés par les a_n que vous venez de déduire.

Je vous aide en proposant quelques lignes incomplètes qui permettent de se lancer !

```
for i in range(N) :   # pour chaque sample i
    t = i*te          # instant correspondant
    sig_t[i] = t
    n = 0          # calcul de la valeur du sample
    while n<nmax: # pour chaque harmonique
          sig_s[i] += an(k,A) * math.cos(2*math.pi*k*Fo*t) + ...
...
```

A est l'amplitude et Fo est la fréquence (fondamentale) du signal $s(t)$ recherché.

- ✔ Ces deux valeurs sont données.

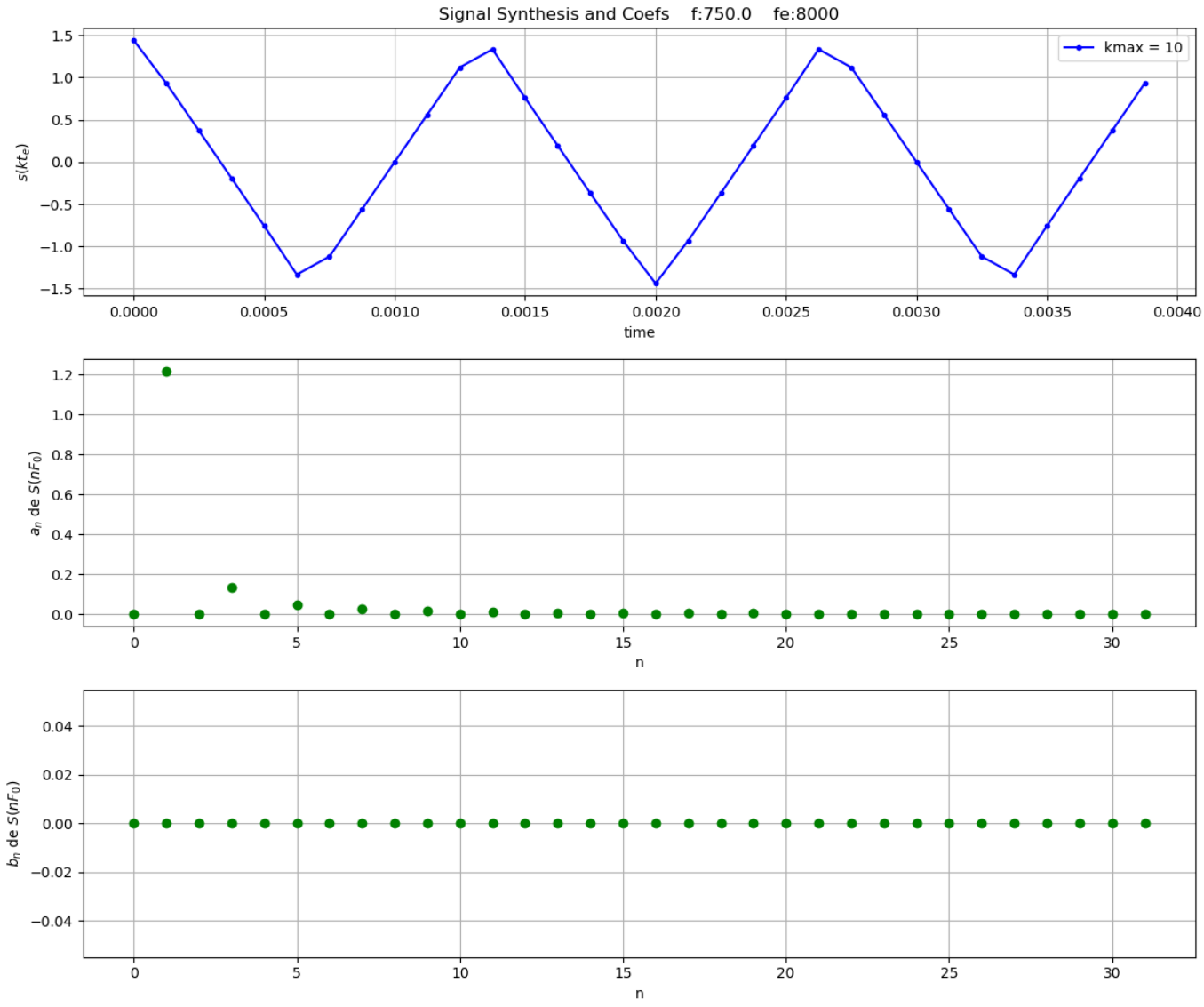
Par exemple :

- ➢ $A = 1.5$ V
- ➢ $f = 750.0$ Hertz
- ➢ $fe = 8000$ Hertz
- ➢ nmax = 32
- ➢ d = $3 * \frac{1}{f}$ secondes

Au niveau de l'obtention de la figure, en vous aidant de :

`http://matplotlib.sourceforge.net/examples/pylab_examples/cohere_demo.html`

Vous devriez obtenir une figure comme celle qui suit :



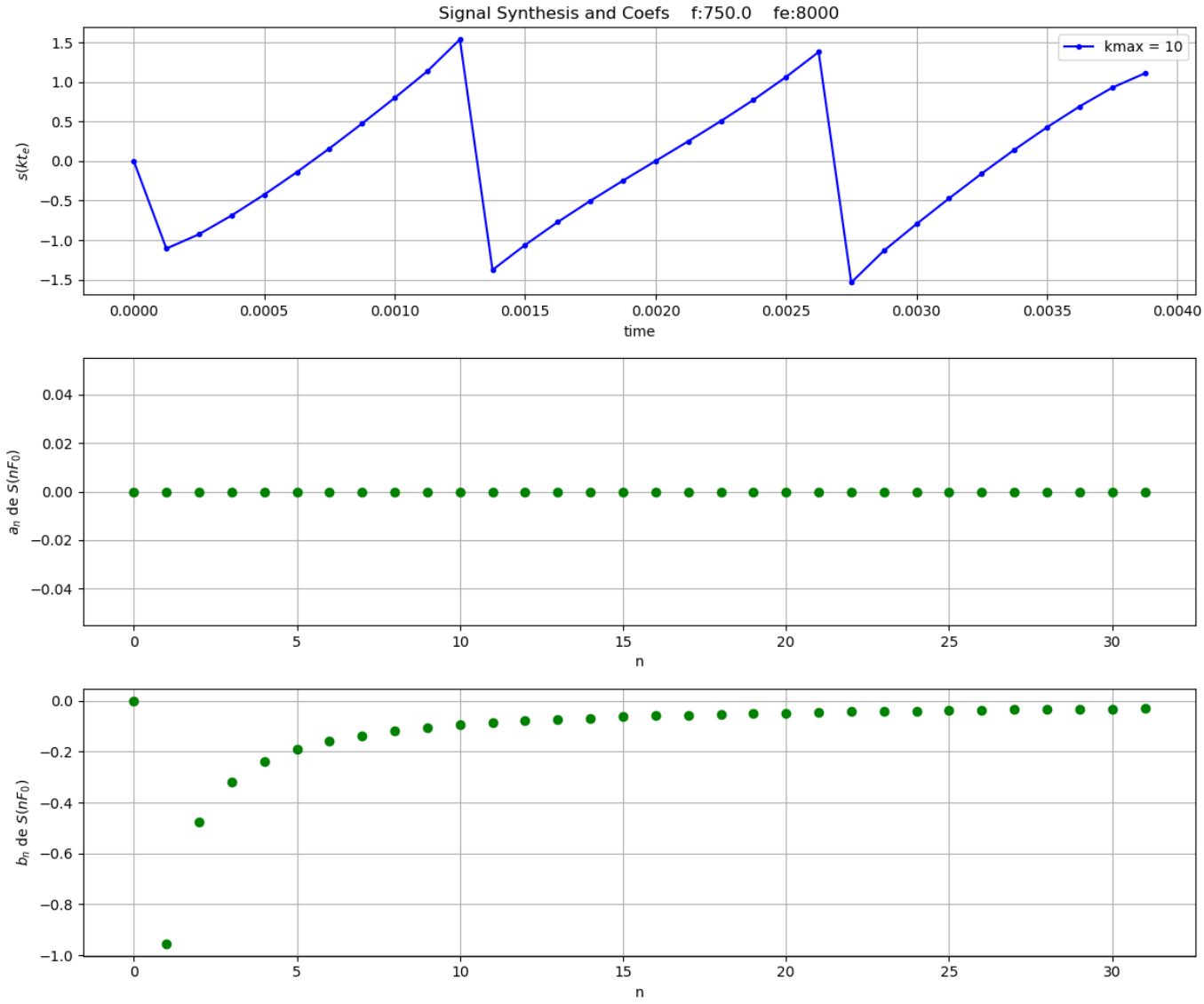
Cela pourrait être sympa de visualiser la "construction" du signal au fur et à mesure que l'on ajoute des harmoniques dans sa composition.

3.1.2 Signal R . . . à partir de la série de Fourier

Mêmes questions avec :

$$s(t) = -\frac{2 * A}{\pi} \sum_{n=1}^{+\infty} \frac{\sin(2\pi n F_0 t)}{n} \tag{8}$$

Vous devriez obtenir une figure comme celle qui suit :



3.1.3 Signal C . . . à partir du spectre

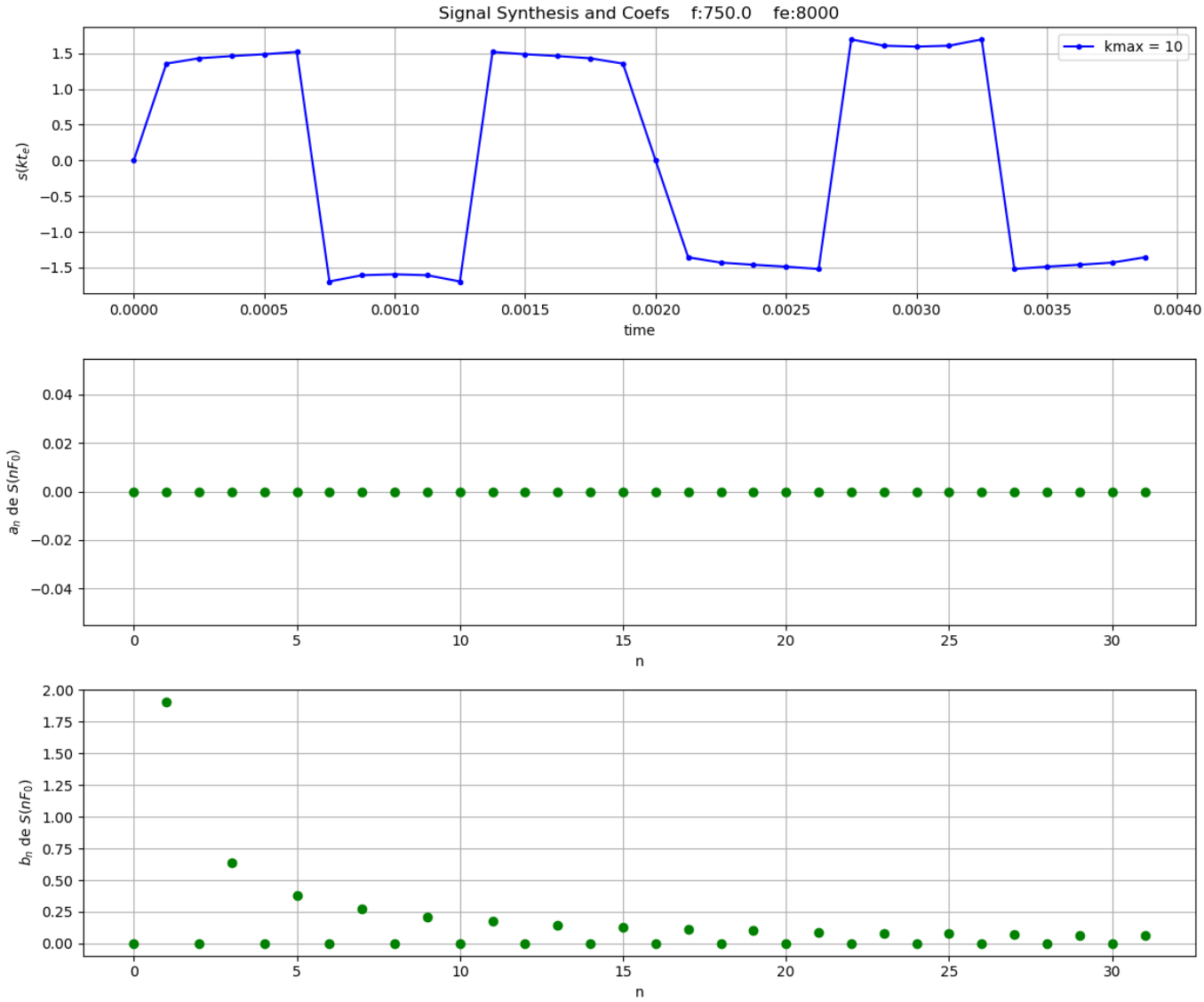
Il s'agit du "même" exercice que précédemment mais ici on va chercher les a_n et b_n dans l'expression du spectre :

$$S(f) = \frac{2 * A}{\pi} . \sum_{n=-\infty}^{+\infty} \frac{-j}{2n+1} \delta(f - (2n+1)F_0) \tag{9}$$

(a) Des équations 9, 2 et 3, déduire la série : c'est à dire les a_n et b_n.
Vous devriez constater qu'il n'y a que des harmoniques impaires et imaginaires.

(b) Faire un programme qui permettra de représenter le signal $s(t)$ (ou du moins sa forme discrétisée) à partir de l'équation 1.

Vous devriez obtenir une figure comme celle qui suit :



4 Conclusion

Sur ces quelques questions, nous avons utilisé la synthèse additive pour construire un signal périodique à partir d'une somme de sinus/cosinus pondérées par des a_n et b_n.

La première des conclusions est que l'on peut synthétiser un signal soit par sa représentation dans le domaine temporel, soit par sa représentation correspondante dans le domaine fréquentiel.

La deuxième conclusion est qu'en terme de télécommunications et de façon "ultra simpliste",

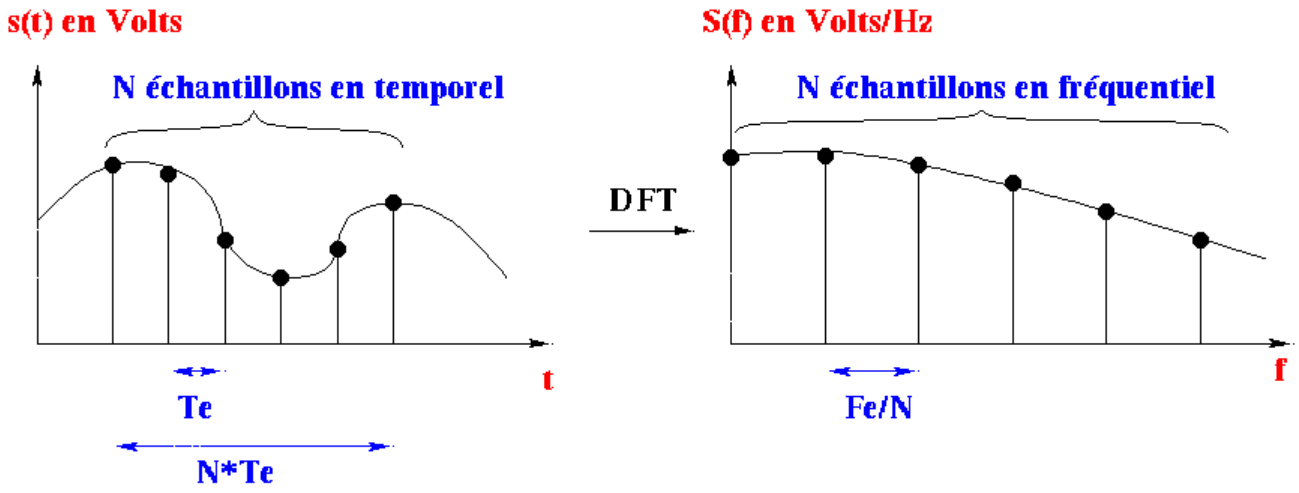
Un signal temporel correspond à un ensemble de "fréquences"

5 Transformée de Fourier des fonctions non périodiques

La transformation de Fourier est un analogue de la théorie des séries de Fourier pour les fonctions **non périodiques**, et permet de leur associer un spectre en fréquences.

- ➢ Dans le but de calculer la transformée de Fourier d'un signal $s(t)$ à l'aide d'un ordinateur, **on a obligation de travailler sur une discrétisation du signal $s(t)$ dont on aura aussi (et forcément) tronqué le support temporel** (car t n'est pas infini !).
- ➢ De plus le signal $s(t)$ étant discret, la transformé de Fourier devient **discrète** : la TF devient TFD ou DFT pour "Discrete Fourier Transform".

Le traitement opéré par la DFT réalise **la correspondance entre deux suites de N termes** :



La "**précision fréquentielle**" entre les points de la représentation fréquentielle est :

$$\Delta f = F_e/N \tag{10}$$

Elle dépend donc :

(a) De la période d'échantillonnage temporelle T_e,

(b) **ET** du nombre de points fournis à la DFT : N

5.1 Mise en oeuvre

La bibliothèque `numpy` propose une fonction de calcul de la FFT (ou "Fast Fourier Transfom").

Comme son nom l'indique il s'agit d'une version "rapide" de la DFT qui impose une petite contrainte :

- ➢ **le nombre de points doit être une puissance de 2** (afin d'exploiter une symétrie dans les calculs).

5.1.1 Analyse d'un code existant

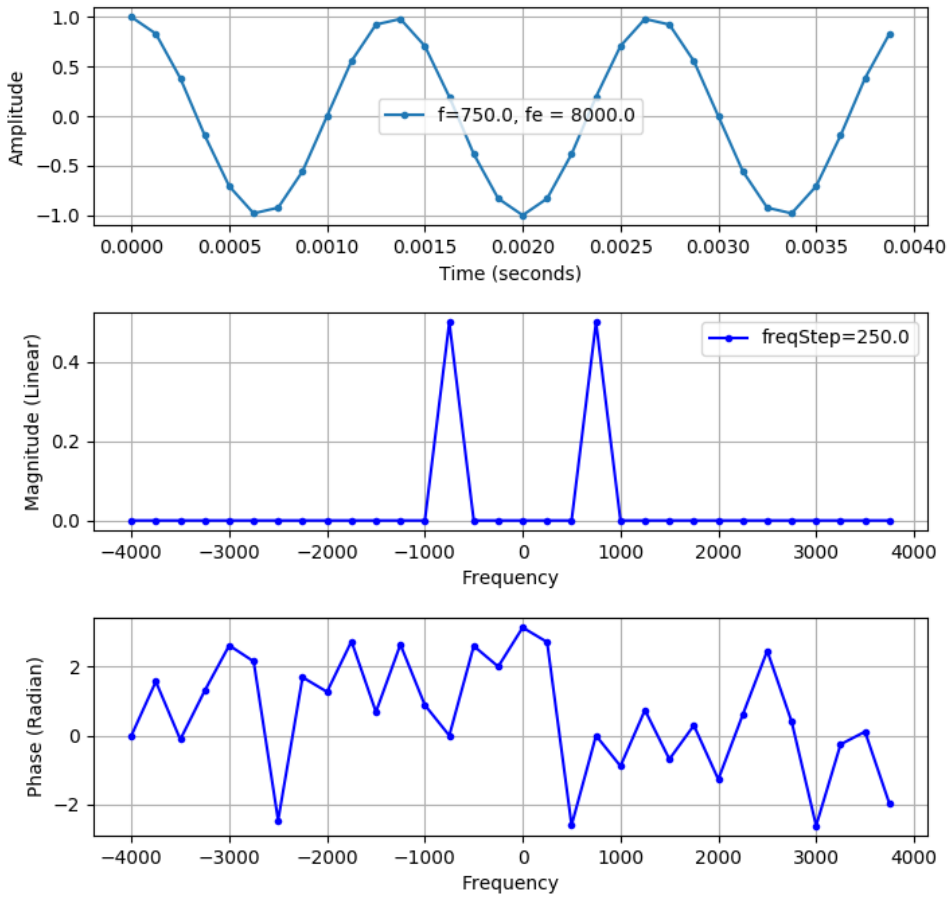
Ce code (dispo sur le site Web) permet d'illustrer l'utilisation de la Transformée de Fourier pour obtenir la décomposition fréquentielle d'un signal :

```
#!/usr/bin/env python
"""
http://yukuan.blogspot.com/2006/12/fft-in-python.html
This example demonstrates the FFT of a simple sine wave and displays its
bilateral spectrum.

Since the frequency of the sine wave is folded by
whole number freqStep, the bilateral spectrum will display two non-zero point.

"""
import math
import numpy as np
import matplotlib.pyplot as plt


def plot_fft(N, freqStep, Fe, t, s_t, s_f):
    """
    s_f is the fft of s_t(t) on N samples

    """
    #==== Symetrisation et normalisation du spectre du signal
    #cf https://dsp.stackexchange.com/questions/4825/why-is-the-fft-mirrored
    s_f = np.fft.fftshift(s_f)      # middles the zero-point's axis
    s_f = s_f/N     # Normalization => ainsi le module ne dependra
                    # de la longueur du signal ou de sa fe
    freq = freqStep * np.arange(-N/2, N/2)  # ticks in frequency domain

    #=== Affichage console des valeurs des raies
    for i,r in enumerate(list(s_f)):
        print("Raie {} \t= \t{:.5g}".format(freq[i],r))

    #==== Plot des spectres
    plt.figure(figsize=(8,8))
    plt.subplots_adjust(hspace=.6)
    # Plot time data ------------------------------------
    plt.subplot(3,1,1)
    plt.plot(t, s_t, '.-', label="N={}, fe={}".format(N,Fe))
    plt.grid(True)
    plt.legend()
    plt.xlabel('Time (seconds)')
    plt.ylabel('Amplitude')
    plt.title('Signal')
    plt.axis('tight')
    # Plot spectral magnitude ---------------------------
    plt.subplot(3,1,2)
    plt.plot(freq, np.abs(s_f), '.-b', label="freqStep={}".format(freqStep))
    plt.grid(True)
    plt.legend()
    plt.xlabel('Frequency')
    plt.ylabel('S(F) Magnitude (Linear)')
    # Plot phase ----------------------------------------
    plt.subplot(3,1,3)
    plt.plot(freq, np.angle(s_f), '.-b')
    plt.grid(True)
    plt.xlabel('Frequency')
    plt.ylabel('S(F) Phase (Radian)')

#================================================

if __name__ == '__main__':
    N = 32  # the number of points in signal s(n*te) et S(n*Fe)
            # Power of 2 !
    Fe = 8000.        # the sampling rate
    Te = 1./Fe        # the sampling period
    freqStep = Fe/N  # resolution of the frequency IN FREQUENCY DOMAIN
    f = 3*freqStep # frequency of the sine wave
                   # On choisit un multiple de freqstep
    T = 1.0/f         # periode de la sinusoide
    a = 255

    #====  Generation d'un signal via numpy !
    t = np.arange(N)*Te          # N ticks in time domain, t = n*Te
                                 # t vector covers 3 periods of cos.
    s_t = a*np.cos(2*math.pi*f*t) # cos
    s_t = a*(4*(abs(t*f-np.floor(t*f+0.5)))-1.0) # triangle

    #==== Calcul du spectre du signal
    s_f = np.fft.fft(s_t)        # Spectrum
    print("fft result is a {} of len {} of type {}\n".format(type(s_f),len(s_f),type(s_f[0])))

    # Plot
    plot_fft(N, freqStep,  Fe, t, s_t, s_f)

    plt.savefig("fft_example{}.png".format(N))
    plt.show()
```

(a) Récupérer le code `gui_fft.py`

(b) Tester et comprendre au moins le principe . . . il y a des explications dans ce qui suit!

Vous devriez obtenir une figure comme celle qui suit :



Plusieurs étapes sont présentes dans ce code :

① La "génération du signal" : Il s'agit d'une "tranche temporelle" de signal cos.

➢ C'est à dire $N = 32$ échantillons "distants temporellement" de $T_e = \frac{1}{F_e}$ secondes.

➢ F_e est la fréquence d'échantillonnage

➢ $freqStep = \frac{F_e}{N}$ représente la résolution fréquentielle de la DFT.

✔ C'est à dire l'écart fréquentiel entre deux raies du spectre.

✔ Une forme de "précision fréquentielle" avec laquelle on peut analyser le signal.

✔ Il dépend inversement de N et donc plus N est grand, plus on aura une "bonne" résolution.

➢ pour ce qui est de la fréquence du cos, dans la mesure où $f = 3 * freqStep$, les N échantillons représenteront 3 périodes de signal temporel.

✔ Petite démo :

$$\begin{aligned} f &= 3 * freqStep \\ &= 3 * \frac{F_e}{N} \end{aligned}$$

donc en passant de fréquence en période :

$$\begin{aligned} \frac{1}{T} &= 3 * \frac{1}{N * T_e} \\ N * T_e &= 3 * T \end{aligned}$$

ce qui peut s'énoncer ainsi :

"N périodes d'échantillonnage sont égales à 3 périodes du signal".

② Ensuite on réalise le "calcul de la FFT" :

Le résultat de cette fonction est un tableau de nombre complexes : Y

Là encore ... quelques petites choses à savoir :

- ➢ Il y avait N échantillons dans le domaine temporel : $s(t) = s(k.T_e)$ pour $k \in [0, N-1]$
- ➢ Il y a N échantillons dans le domaine fréquentiel : $S(f) = S(k.\Delta f)$ pour $k \in [0, N-1]$

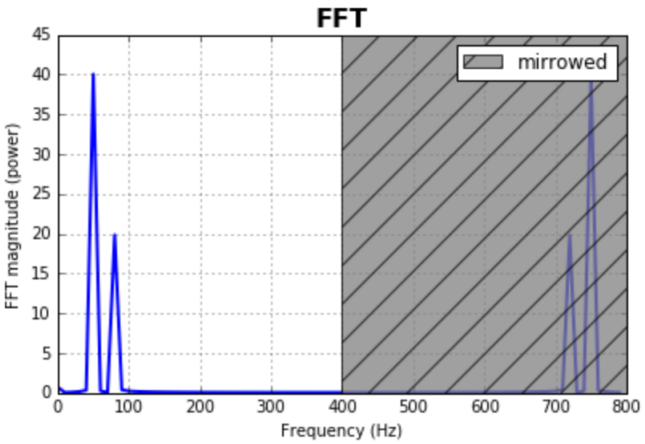
Chaque élément du tableau Y correspond à une "harmonique" du signal $s(t)$:

- ➢ $Y[0]$ est la composante continue de $s(t)$,
- ➢ $Y[1]$ est l'harmonique (i.e. le cosinus) de fréquence f du signal $s(t)$,
  C'est la **fondamentale !**
- ➢ $Y[2]$ est l'harmonique (i.e. le cosinus) de fréquence $2 * f$ du signal $s(t)$,
- ➢ ...
- ➢ $Y[N-1]$ est l'harmonique (i.e. le cosinus) de fréquence $(N-1) * f$ du signal $s(t)$.
  C'est pratiquement $Fe.$ (à freqStep ($\equiv \Delta f$) près)

Souvenez vous que le principe de la série de Fourier est que ces harmoniques de fréquences différentes (mais multiples) s'additionnent pour former $s(t)$.

③ Il est de coutume de représenter les harmoniques entre $[-F_e/2, +F_e/2[$.

- ➢ Pour ce faire, on translate (fonction fftshift) de $-N$ l'ensemble des harmoniques comprises entre $[N/2, N-1]$.

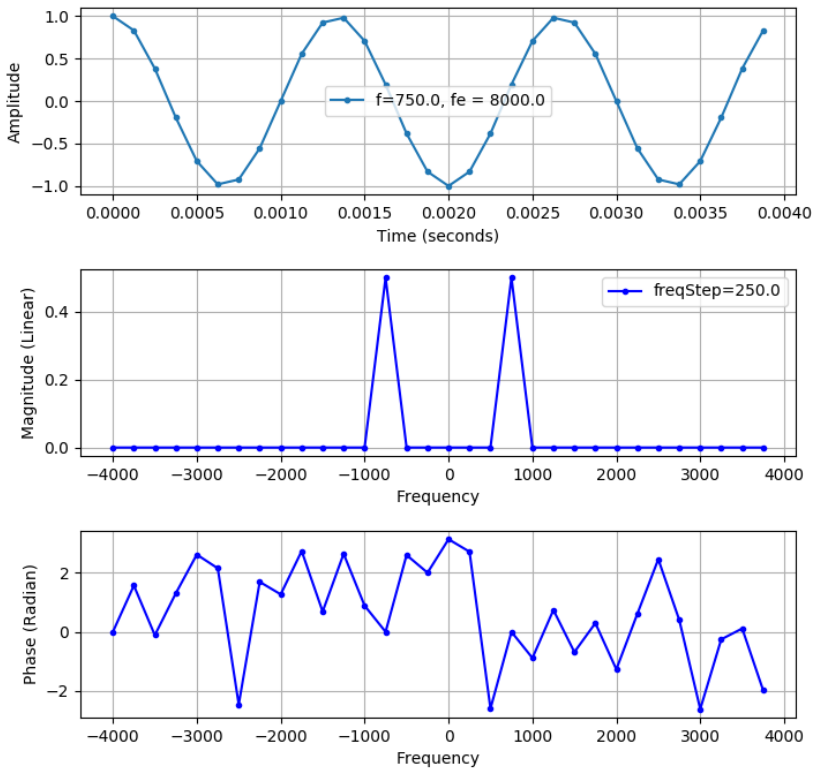


④ Ensuite c'est l'affichage sur 3 plots:

- ➢ Un pour la représentation temporelle,
- ➢ Un pour la représentation de l'amplitude des harmoniques de la représentation fréquentielle . . . donc valeur absolue,
- ➢ Un pour la représentation de l'argument des harmoniques de la représentation fréquentielle.

5.1.2 Levons un peu la tête . . .

Que disent ces courbes . . . et surtout celle d'amplitude ?



L'harmonique remarquable par son amplitude, et aussi la première harmonique (donc fondamentale), est celle située en graduation 3.

- ➢ on peut donc en déduire que le signal $s(t)$ est composé "exclusivement" d'un cosinus (pas de partie imaginaire) de fréquence :

$$3 * \Delta f = 3 * F_e/N = 3 * 8000/32 = 750Hz$$

Ce qui correspond à la fréquence du signal temporel.

5.1.3 Et les autres harmoniques . . .

Si on regarde (sur la console) le tableau Y de plus près, on va constater que les autres harmoniques sont nulles.

5.1.4 Phénomène de fenêtrage

Vous modifiez le programme Python pour choisir une fréquence f qui n'est pas un multiple entier de freqStep.

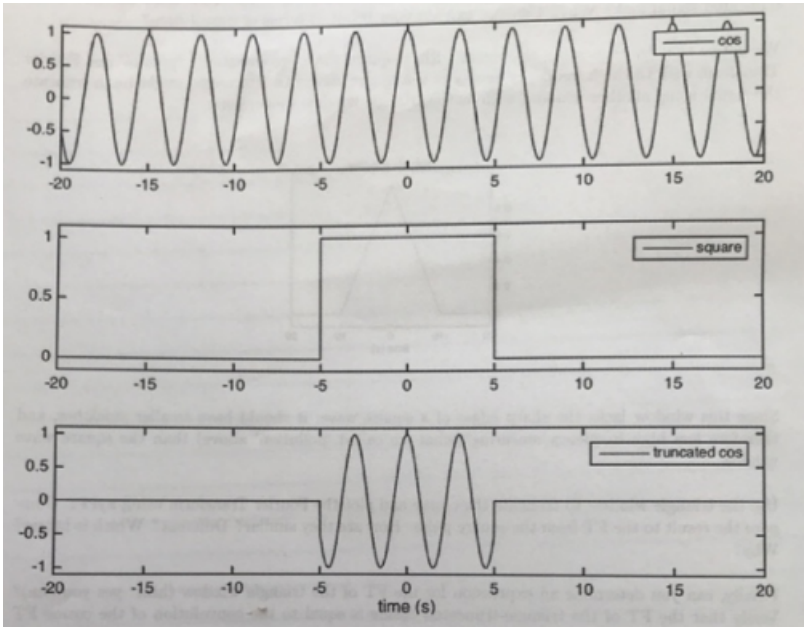
➢ Prenez par exemple : f= 3.2*freqStep

Que constatez vous sur les harmoniques ?

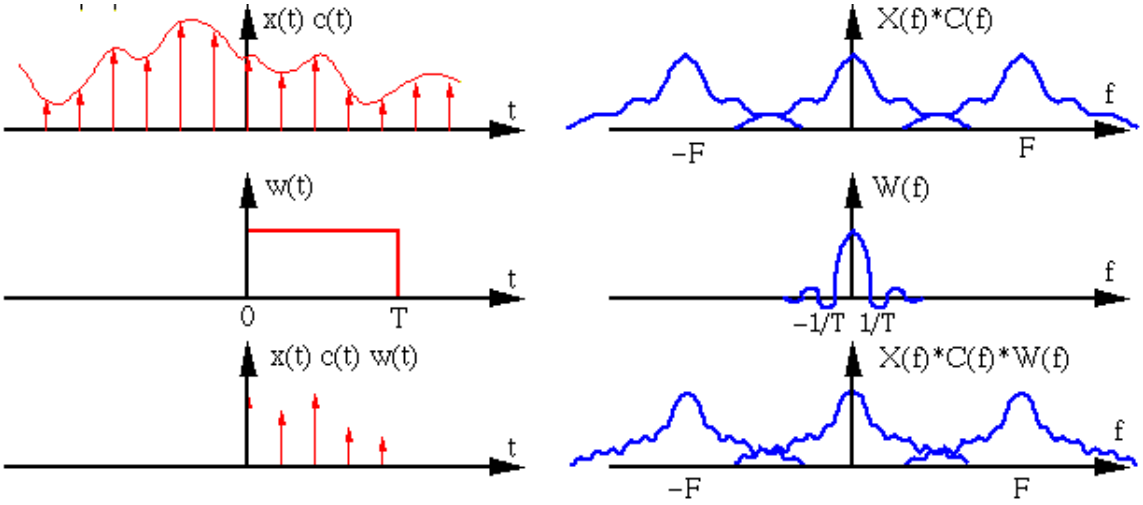
Et pourtant ... elles devraient être nulles puisque $s(t)$ est toujours un cosinus. En plus sa fréquence n'a pas changé !

Il s'agit du problème de fenêtrage du signal $s(t)$ (le fait de ne prendre qu'un morceau de signal) qui n'apparaît pas si on donne un nombre de période entier du signal à la FFT.

➢ En effet, dans tous les cas la fft n'opère que sur un signal tronqué, un morceau, "une tranche" de N échantillons.



Cette troncature correspond à la multiplication dans le domaine temporel par un signal "porte" et donc à la convolution par un sinus cardinal dans le domaine des fréquences :



Ce phénomène n'apparaît pas si on donne à la FFT un nombre de périodes entier du signal . . . encore faut-il que le signal soit périodique !

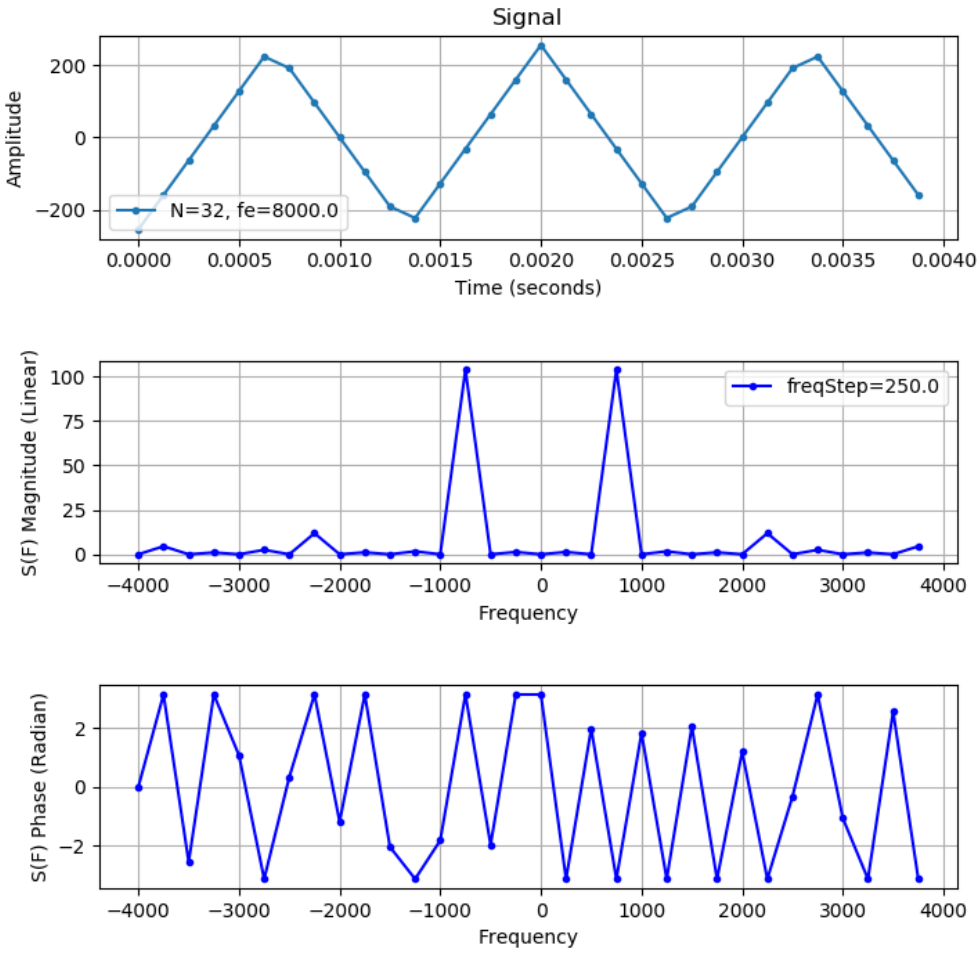
On abandonne là compte tenu du temps imparti !

6 TODO

6.1 Avec les signaux du TP1

(a) Essayer d'appliquer la FFT et d'interpréter les résultats sur les signaux générés au TP1 (donc avec leur équation temporelle) : carré, rampe, triangle . . .

On ne donne ici que la dent de scie (N=32), à vous de faire les autres . . .



(b) Quel lien faîtes vous entre cette figure et celle concernant la synthèse par les séries de Fourier de la dent de scie ?

Retrouver les coefficients et les fréquences.

6.2 Avec des signaux réels

```
import math
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile
import gui_fft
#================================================

if __name__ == '__main__':
    filename = "La3piano.wav"
    filename = "La3guitare.wav"
    filename = "La3diapason.wav"
    filename = "La3guitare.wav"

    Fe,s_t = wavfile.read(filename, mmap=False)

    # on ne fait la FFT que des N premiers samples
    N = 1024
    freqStep = Fe/N
    t = np.arange(N)/Fe
    s_t = s_t[0:N]
    s_f = np.fft.fft(s_t)   # Spectrum

    gui_fft.plot_fft(N, freqStep, Fe, t, s_t, s_f)
    plt.show()
```

Utiliser ce code pour différentes version de la note "La 440" sur les fichiers fournis :

(a) Analyser le diapason

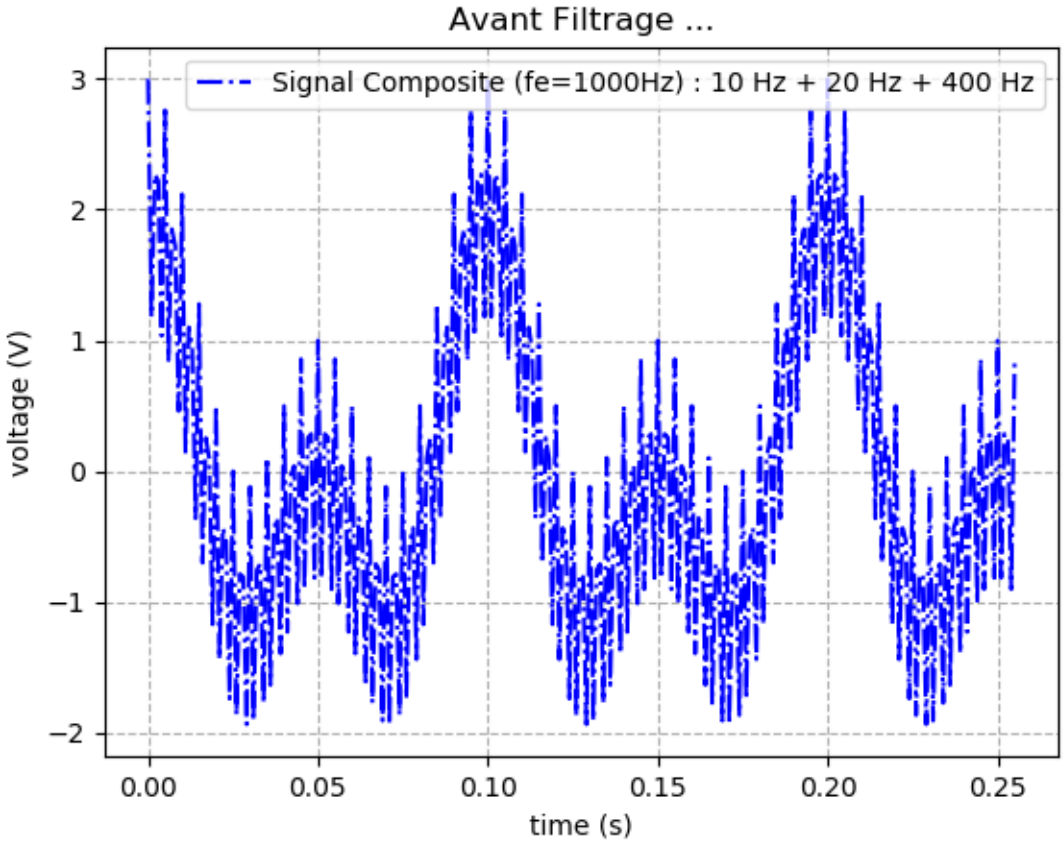
(b) Analyser la guitare

(c) Analyser le piano

Conclure sur ces spectres.

6.3 Le traitement des signaux numériques

Pour illustrer ce thème, vous allez commencer par générer un signal (fe=1000Hz et $N = 256$) composite composé de trois cosinus de même amplitude (1 Volts) mais de fréquences : 10, 20 et 400 Hertz.



On peut voir ça comme l'agrégation d'informations différentes, ou alors comme un signal venant en bruiter deux autres.

(a) Représenter le signal et son spectre !

Combien de raies/harmoniques ?

L'opération de **filtrage** que vous devez ensuite programmer vise à séparer les informations du signal composite en supprimant par exemple une des composantes.

6.3.1 Convolution

La formule qui suit est celle d'une **convolution du signal x par un filtre h afin d'obtenir le signal filtré résultant y** :

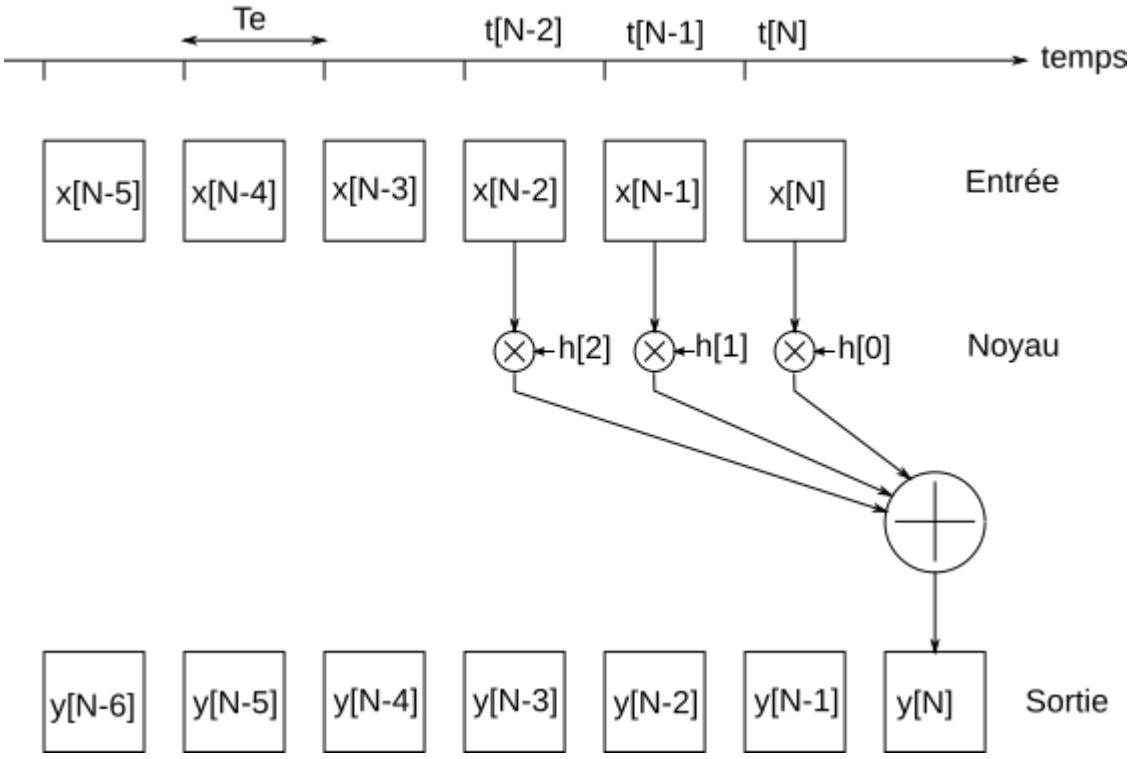
$$y[n] = \sum_{k=0}^{L-1} h[k] * x[n-k] \tag{11}$$

avec $L = len(h) = 35$ et

```
h = [-6.849167e-003, 1.949014e-003, 1.309874e-002,1.100677e-002,
-6.661435e-003,-1.321869e-002, 6.819504e-003, 2.292400e-002,7.732160e-004,
-3.153488e-002,-1.384843e-002,4.054618e-002,3.841148e-002,-4.790497e-002,
-8.973017e-002, 5.285565e-002,3.126515e-001, 4.454146e-001,3.126515e-001,
5.285565e-002,-8.973017e-002,-4.790497e-002, 3.841148e-002, 4.054618e-002,
-1.384843e-002,-3.153488e-002, 7.732160e-004,2.292400e-002,6.819504e-003,
-1.321869e-002,-6.661435e-003, 1.100677e-002,1.309874e-002,1.949014e-003,
-6.849167e-003]
```

On prendra $x[n-k] = 0$ si $(n-k) < 0$

Cette formule ne fait qu'exprimer que chaque échantillon du signal y se calcule à partir d'une combinaison linéaire d'un noyau (fourni par h) et d'échantillons passés de x :



http://www.f-legrand.fr/scidoc/docimg/numerique/filtre/convolution/convolution.html

(a) Montrer le signal filtré et son spectre !

Alors que concluez vous ?