Introduction

La transformée de Fourier est un outil mathématique qui permet de représenter une fonction (dans le temps ou l'espace par exemple) par une autre fonction qui contient la même information totale, mais qui l'exprime sous la forme d'une somme de composantes harmoniques (en fréquences temporelles ou spatiales). On peut parler de série temporelle et de son spectre par exemple de façon interchangeable, les deux contenant la même information mais la présentant différemment. Le document présente les notions pratiques de base pour pouvoir adéquatement analyser un signal et obtenir son spectre de Fourier, étalonné correctement.

La connaissance et la maitrise de cette transformation en science est importante: nous prenons des mesures dans le temps et l'espace, mais l'origine des phénomènes a souvent une composante oscillatoire qui est facilement décrite en fréquence (par exemple, autour d'un point d'équilibre, le comportement de rappel est souvent une oscillation harmonique, ou encore les émissions provenant des atomes sont des ondes à des fréquences spécifiques, etc...). Ainsi, la description dans l'espace de Fourier est souvent un élément clef pour mieux comprendre, ou pour mesurer ou analyser plus facilement un phénomène.

Définitions

Il y a plusieurs définitions de la transformée de Fourier. Une des définitions pour une fonction $f(x)$ est donnée par:

$$F(k) = \int_{-\infty}^{+\infty} f(x) e^{ikx} dx, \tag{1}$$

et sa transformée inverse est ainsi:

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(k) e^{-ikx} dk. \tag{2}$$

La variable k est la variable conjuguée de x et correspond à une fréquence angulaire $k = 2\pi\nu$, la fonction $F(k)$ est souvent appelée le spectre de $f(x)$. Notons que la variable x peut-être une variable d'espace (m) ou de temps (s). Ainsi, la variable conjuguée k aura respectivement des unités de $\mathrm{rad} \cdot m^{-1}$ ou $\mathrm{rad} \cdot s^{-1}$, et la variable ν aura des unités de m^{-1} ou s^{-1}. Pour calculer cette transformation, la fonction $f(x)$ doit être connue sur l'intervalle complet de $-\infty$ à $+\infty$.

> Peut-être vous demandez-vous pourquoi l'équation (1) est complexe? Voir note ci-dessous.

Au laboratoire, on peut mesurer des signaux par échantillonnage discret: plutôt que d'avoir la représentation continue du signal, on ne connait la valeur de la fonction qu'à certains points discrets $x_j = j\Delta x$. On écrira parfois simplement $f[x_i]$, pour rappeler la syntaxe informatique pour l'accès d'un tableau. On définit la transformée de Fourier discrète comme:

$$F[\nu_i] = \sum_{j=0}^{N-1} f[x_j] e^{i2\pi\nu_i x_j} \tag{3}$$

et sa transformée inverse:

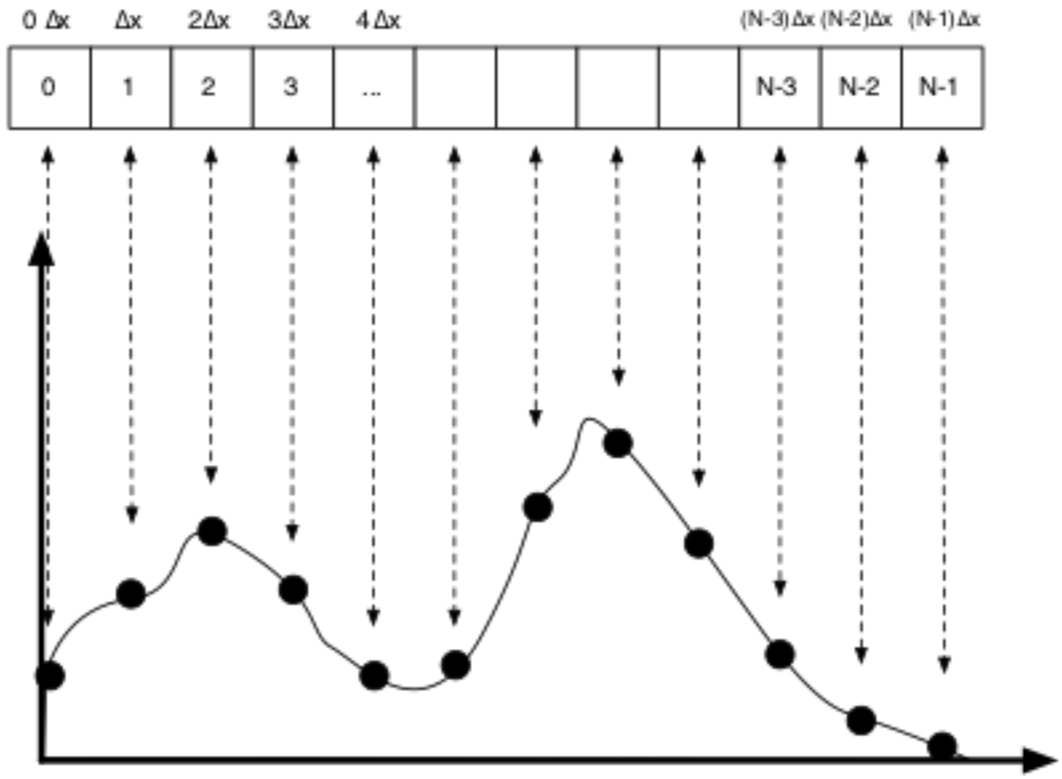
$$f[x_i] = \frac{1}{N} \sum_{j=0}^{N-1} F[\nu_j] e^{-i2\pi\nu_j x_i}. \tag{4}$$

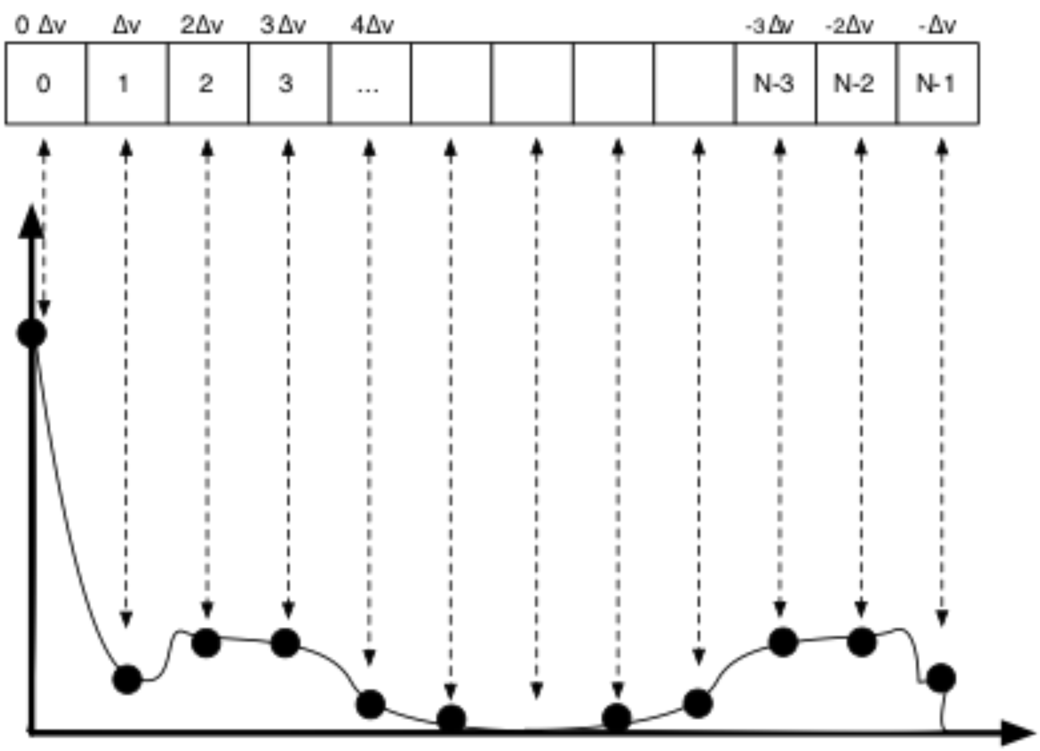
Ainsi, avec un tableau de données f contenant N points, on peut obtenir un second tableau F contenant aussi N points. Si l'espacement entre les points de f est de ΔX, l'étendue du tableau est de $N\Delta X$. Après transformation de Fourier, le tableau en F sera défini de $]-\nu_{\max}, \nu_{\max}]$ avec :

$$\begin{aligned} \Delta\nu &= \frac{1}{N\Delta X} \\ \nu_{\max} &= \frac{1}{2\Delta X} \end{aligned}$$

Notez:

1. La transformée de Fourier discrète est définie sur les fréquences négatives et positives, de $-\nu_{\max}$ à $\nu_{\max}$.
2. Avec une fonction réelle $f(x)$, les composantes négatives et positives des fréquences seront identiques et réelles.
3. La composante $F[0]$ est la composante continue, ou DC, ou dit autrement la moyenne du signal $f[x]$.
4. La grande majorité des programmes (MATLAB, Mathematica, etc...) retourne le tableau en commençant par les composantes positives suivi par les composantes négatives.
5. La grande majorité des programmes (MATLAB, Mathematica, etc...) travaille dans l'espace de Fourier avec ν, et non $k = 2\pi\nu$. Ceci correspond mieux à la réalité (i.e. les fréquences sont des vraies fréquences, comme par exemple 60 Hz est 60 Hz, pas 376 rad Hz).
6. Les composantes $\nu_{\max}$ et $-\nu_{\max}$ sont identiques et sont au $N/2$-ième point du tableau.
7. La première composante du tableau est celle correspondant à $0\Delta\nu$, suivies de $\Delta\nu$, $2\Delta\nu$, etc... jusqu'à $(N/2)\Delta\nu$. Le dernier point du tableau est $-\Delta\nu$, l'avant dernier est $-2\Delta\nu$, précédé par $-3\Delta\nu$. Vous pourrez vérifier que le dernier point du tableau est égal au deuxième, etc...





Phaseurs et autres

1. La notation phaseur n'a rien de sorcier: elle est simplement plus facile à manipuler: l'intégrale de $e^{ix} dx$ est plus facile et plus courte que $(\cos(x) + i \sin(x))dx$. Inversement, plutôt que de travailler avec des sinus et des cosinus, on ré-écrit en notation phaseur.

2. Elle est tellement plus facile que c'est la norme, et c'est tellement la norme qu'on n'écrit même plus le complexe conjugué dans les équations en ne faisant l'integrale que sur les fréquences positives. En effet, on devrait vraiment toujours écrire:
$$f(x) = \frac{1}{2\pi}\int_0^{+\infty} F(k)e^{-ikx}dk + \text{c.c.} \tag{5}$$
Cependant, on ne le fait pas: on fait l'intégrale de $-\infty$ à $+\infty$ et on dit que $F(k)$ est hermétienne, c'est à dire $F(-k) = F^*(k)$. C'est une supposition implicite qui est répandue et plus personne ne le répète après le deuxième cours de physique mathématique à la première session.
3. Sachant cela, on se retrouve avec des fréquences positives et négatives pour décrire une oscillation sinusoidale car si on prend l'exemple le plus simple d'une oscillation harmonique à une fréquence spécifique constante $k_\circ$, on obtient :
$$F(k) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \cos(k_\circ x)e^{ikx}dx, \tag{10}$$
$$F(k) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \frac{e^{ik_\circ x} + e^{-ik_\circ x}}{2}e^{ikx}dx, \tag{7}$$
$$F(k) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \frac{e^{i(k_\circ + k)x}}{2}dx + \frac{1}{2\pi}\int_{-\infty}^{+\infty} \frac{e^{-i(k_\circ - k)x}}{2}dx, \tag{8}$$
$$F(k) = \delta(k + k_\circ) + \delta(k - k_\circ), \tag{9}$$
On dit qu'il y a une composante à $+k_\circ$ et $-k_\circ$. Le sens profond n'est rien de plus que la somme des composantes exponentielles doit toujours donner une fonction réelle, donc on se doit d'avoir les deux composantes, et leurs amplitudes doivent être complexes conjuguées.

Des exemples

Pour manipuler des transformées de Fourier numériquement, utilisez le code Python suivant disponible sur GitHub. Vous y trouverez l'essentiel pour lire ou générer des données et explorer leurs spectres.

```
import numpy as np
from numpy.random import *
from numpy.fft import *
import matplotlib.pyplot as plt

""" Ce script genere des interferogrammes tels qu'obtenus avec un interferometre
de Michelson dans le but d'etudier la transformée de Fourier et de comprendre
comment la resolution spectrale est déterminée.
"""


def readVectorsFromFile(filename):
  x = np.loadtxt(filename, usecols=(0))
```

```
    y = np.loadtxt(filename, usecols=(1))
    return (x,y)

def generateHeNeInterferogram(xMin, xMax, N):
    """ Genere un tableau de N valeurs equidistantes enntre xMin et xMax.
    Ensuite, genere un tableau de N valeurs qui representent un interferogramme
    d'un laser He-Ne a 0.6328 microns. On ajoute du bruit pour rendre le tout
    plus realiste.
    """
    dx = (xMax - xMin)/N
    x = np.linspace(xMin, xMax, N)
    noise = random(len(x))*0.05
    y = 1+np.cos(2 * np.pi / 0.6328 * x)+noise
    return (x,y)

def generateWhiteLightInterferogram(xMin, xMax, N):
    """ Genere un tableau de N valeurs equidistantes enntre xMin et xMax.
    Ensuite, genere un tableau de N valeurs qui representent un interferogramme
    d'une source blanche visible. On ajoute du bruit pour rendre le tout
    plus realiste.
    """
    dx = (xMax - xMin)/N
    x = np.linspace(xMin, xMax, N)
    noise = random(len(x))*0.05
    k1 = 1/0.4
    k2 = 1/0.8
    y = 1+np.exp(-x*x/4)*(np.sin(2 * np.pi * (k1+k2)*x/2)/x * np.sin(2 * np.pi *
(k1-k2)*x/2)+ noise)
    return (x,y)

def fourierTransformInterferogram(x,y):
    """ A partir du tableau de valeurs Y correspondant a l'abscisse X,
    la transformée de Fourier est calculée et l'axes des fréquences (f en
    μm^-1) et des wavelengths (1/f en microns) est retournée.

    Le spectre est un ensemble de valeurs complexes pour lesquelles l'amplitude
    et la phase sont pertinentes: l'ordre des valeurs commence par la valeur DC (0)
    et monte jusqu'a f_max=1/2/Δx par resolution de Δf = 1/N/Δx. A partir de la
    (N/2) ieme valeur, la frequence est negative jusqu'a -Δf dans la N-1 case.
    Voir
    https://github.com/dccote/Enseignement/blob/master/HOWTO/HOWTO-
Transformes%20de%20Fourier%20discretes.pdf
    """
    spectrum = fft(y)
    dx = x[1]-x[0] # on obtient dx, on suppose equidistant
    N = len(x)     # on obtient N directement des données
```

```
  frequencies = fftfreq(N, dx) # Cette fonction est fournie par numpy
  wavelengths = 1/frequencies  # Les fréquences en µm^-1 sont moins utiles que
lambda en µm
  return (wavelengths, frequencies, spectrum)

def plotCombinedFigures(x, y, w, s, title="", left=400, right=800):
  """
  On met l'interferogramme et le spectre sur la meme page.
  """
  fig, (axes, axesFFT) = plt.subplots(2,1,figsize=(10, 7))
  axes.plot(x, y, '-')
  axes.set_title("Interferogramme")
  axesFFT.plot(w*1000, abs(s), 'o-')
  axesFFT.set_xlim(left=left, right=right)
  axesFFT.set_xlabel("Longueur d'onde [nm]")
  axesFFT.set_title(title)
  plt.show()


# Basse resolution
(x,y) = generateHeNeInterferogram(xMin=-10, xMax=10, N=200) # en microns
(w, f, s)  = fourierTransformInterferogram(x,y)
df = f[1]-f[0]
dl = 0.6328*0.6328*df*1000 # x 1000 pour nm
plotCombinedFigures(x,y,w,s,left=632.8-5*dl, right=632.8+5*dl, title="Spectre He-
Ne basse resolution {0:0.2f} nm".format(dl))

# Haute resolution
# Resolution Δf = 1/(200 µm * 2000)
# Resolution @ 632.8 nm : Δlambda = 632.8^2 * Δf car Δlambda/lambda = Δf/f.
(x,y) = generateHeNeInterferogram(xMin=-100, xMax=100, N=2000) # en microns
(w, f, s) = fourierTransformInterferogram(x,y)
df = f[1]-f[0]
dl = 0.6328*0.6328*df*1000
plotCombinedFigures(x,y,w,s,left=632.8-5*dl, right=632.8+5*dl, title="Spectre He-
Ne haute resolution {0:0.2f} nm".format(dl))


# Tres haute resolution
# Resolution Δf = 1/(2000 µm * 2000)
# Resolution @ 632.8 nm : Δlambda = 632.8^2 * Δf
(x,y) = generateHeNeInterferogram(xMin=-1000, xMax=1000, N=20000) # en microns
(w, f, s) = fourierTransformInterferogram(x,y)
df = f[1]-f[0]
dl = 0.6328*0.6328*df*1000
```

```
plotCombinedFigures(x,y,w,s,left=632.8-5*dl, right=632.8+5*dl, title="Spectre He-Ne tres haute resolution {0:0.2f} nm".format(dl))


# Hyper haute resolution
# Resolution Δf = 1/(20000 μm * 20000)
# Resolution @ 632.8 nm : Δlambda = 632.8^2 * Δf
(x,y) = generateHeNeInterferogram(xMin=-10000, xMax=10000, N=200000) # en microns
(w, f, s) = fourierTransformInterferogram(x,y)
df = f[1]-f[0]
dl = 0.6328*0.6328*df*1000
plotCombinedFigures(x,y,w,s,left=632.8-5*dl, right=632.8+5*dl, title="Spectre He-Ne hyper haute resolution {0:0.2f} nm".format(dl))


# Spectre de lumiere blanche
# Resolution Δf = 1/(20000 μm * 20000)
# Resolution @ 500 nm : Δlambda = 500^2 * Δf
(x,y) = generateWhiteLightInterferogram(xMin=-100, xMax=100, N=20000) # en microns
(w, f, s)  = fourierTransformInterferogram(x,y)
df = f[1]-f[0]
dl = 0.500*0.500*df*1000 # resolution autour de 0.500 μm en nm
plotCombinedFigures(x,y,w,s,left=0, right=2000, title="Spectre lumiere blanche, resolution {0:0.2f} nm".format(dl))
```