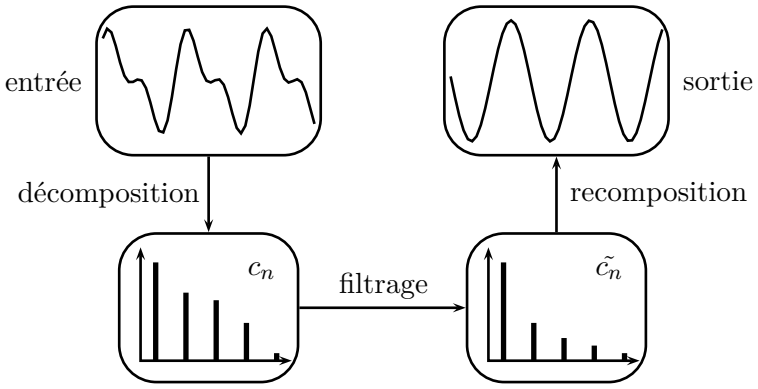
Nous allons étudier une application des séries de fonctions que vous allez étudier cette année.

Courant du XVIIIèmeS l'étude des cordes vibrantes lance la résolution de problème à l'aide d'une superposition de fonctions sinusoïdales. Peu après, entre 1 807 et 1 822, Fourier proposa d'utiliser des séries de fonctions pour résoudre l'équation de la chaleur. Néanmoins, il n'avait pas cherché les hypothèses nécessaires à la validité de sa théorie. En 1 829, Dirichlet délimita les conditions d'utilisation des séries de Fourier et en 1 848 Wilbraham découvrit le phénomène de Gibbs (qui sera redécouvert 51 ans plus tard par Gibbs).

Il est parfois difficile d'appliquer une opération (*e.g.* un filtrage) sur un signal d'entrée dans sa globalité. Une stratégie est alors de le décomposer comme une "somme de signaux", appliquer la-dite opération et reconstruire le signal qui devient le signal de sortie.

Schématiquement cela donne :

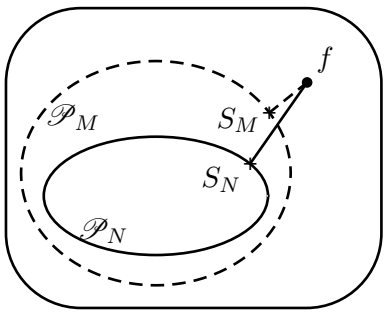


Le but du TP est :

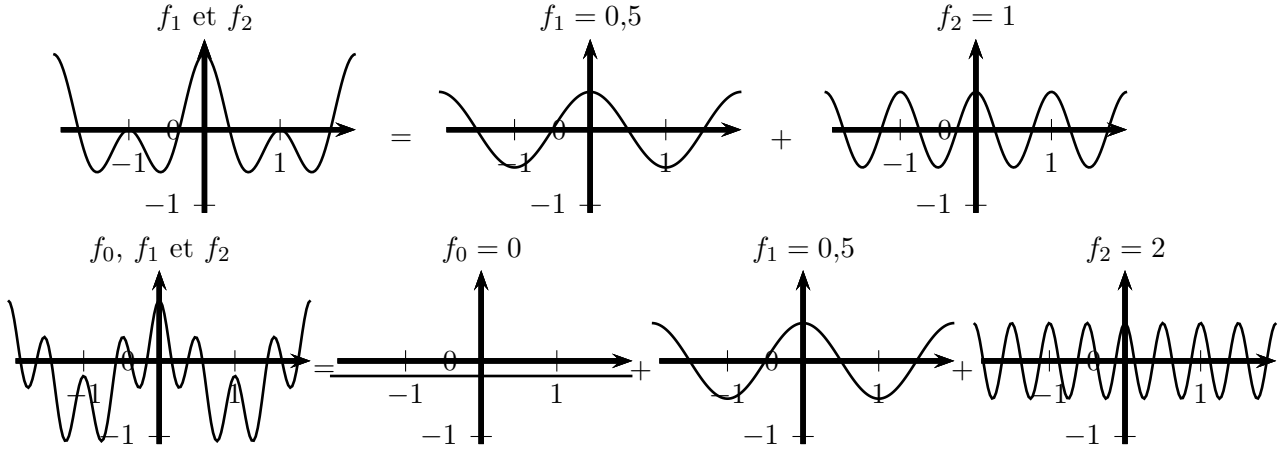
- ◇ appréhender la décomposition et reconstitution d'un signal
- ◇ appréhender le phénomène de Gibbs
- ◇ essayer un filtre passe-bas et appréhender ses conséquences sur le signal

I Décomposition en série de Fourier et reconstitution

On considère une fonction f à valeur de $\mathbb{R}$ dans $\mathbb{R}$, T-périodique et continue par morceaux ; *i.e.* $f \in \mathscr{CM}_T(\mathbb{R},\mathbb{R})$.
Le but est d'approximer la fonction f à l'aide d'une somme de fonctions trigonométriques de fréquence de plus en plus élevée.
L'idée est de projeter orthogonalement la fonction f sur un espace de fonctions trigonométriques simples $\mathscr{P}_N$.
Si l'espace $\mathscr{P}_M$ est plus grand que $\mathscr{P}_N$, le projeté orthogonal S_M est plus proche de f que ne l'est S_N.
On muni $\mathscr{CM}_T(\mathbb{R},\mathbb{R})$ du produit scalaire $< f,g >= \frac{1}{T}\int_a^{a+T} f(t)g(t)dt$. Par périodicité de f et g, on peut prendre n'importe quel a.



1 Exemples graphiques



2 Coefficients trigonométriques de Fourier

Sous certaines hypothèses de continuité de f, nous avons : $f(t) = \frac{a_0}{2} + \sum_{n\geqslant 1} a_n cos\left(\frac{2\pi n}{T}t\right) + \sum_{n\geqslant 1} b_n sin\left(\frac{2\pi n}{T}t\right)$

où $\begin{cases} a_n = \frac{2}{T}\int_a^{a+T} f(t) \times cos\left(\frac{2\pi n}{T}t\right)dt \\ b_n = \frac{2}{T}\int_a^{a+T} f(t) \times sin\left(\frac{2\pi n}{T}t\right)dt \end{cases}$

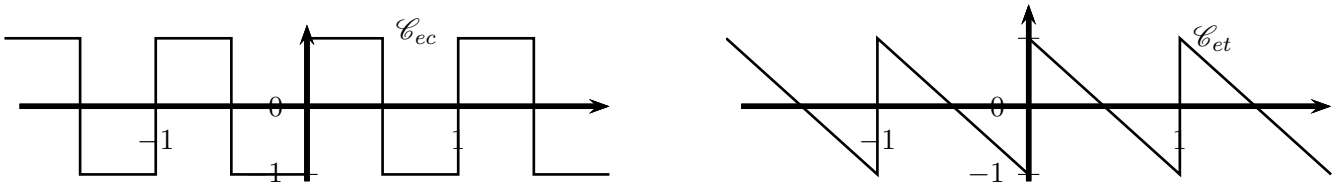
Propriété 1

$\frac{a_0}{2}$ est la valeur moyenne de f sur une période T,
Si f est paire alors $\forall n \in \mathbb{N}^*$, $b_n = 0$ et si f est impaire alors $\forall n \in \mathbb{N}^*$, $a_n = 0$.

Démonstration :

□

Dans ce TP, nous allons étudier les deux signaux d'entrées périodiques ci-dessous.



1. Définir les fonctions entrée carrée *ec* et entrée triangulaire *et*, donner leur parité et leur valeur moyenne,
2. calculer leurs coefficients trigonométriques de Fourier,
3. représenter leur spectre d'amplitude pour $f = \frac{1}{T} < 10$.

rq : on représente le spectre d'amplitude du signal graphiquement en représentant les amplitudes des fonctions trigonométriques selon leur fréquence.

3 Implémentation

Vous pourrez vous inspirer du fichier `CPGE-Info-Filtrage-TP-eleve.py`.
En préambule, on n'oubliera pas d'importer les bibliothèques utiles, à savoir :

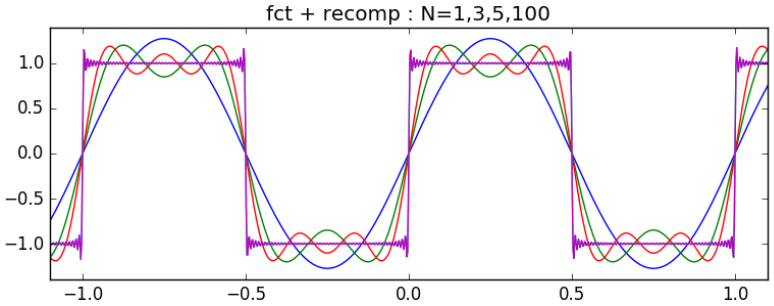
```
import numpy as np
import matplotlib.pyplot as plt
from math import cos, sin, pi
from scipy.integrate import quad
from cmath import phase
```

On travaillera sur l'intervalle $[-1{,}2; 1{,}2]$ avec 1 000 points de discrétisation.

4. Définir un vecteur `vt` prenant mille points sur notre intervalle d'étude.
5. définir la fonction `carre(t)` sous python qui renvoie le signal carré d'amplitude 1 et de période 1,
6. définir une fonction `vectImage_fct(vt, fct)` qui renvoie sous forme de vecteur toutes les images de `vt` par `fct`,
7. représenter graphiquement la fonction `carree(t)` sur $[-1{,}2; 1{,}2]$.
8. définir une fonction `sin_fct(t, n, T, fct)` qui renvoie $\frac{2}{T} \times fct(t) \times sin\left(\frac{2\pi n}{T}t\right)$ puis une fonction `coeff_Fourier_bn (T, n, fct)` qui renvoie $b_n(f)$ à l'aide de `quad`,

La fonction `quad(f,a,b,args=())` du module `scipy.integrate` permet d'intégrer numériquement une fonction f sur $[a;b]$. Si la fonction f nécessite des arguments supplémentaires, on peut les lui fournir à l'aide du champ `args`. `quad` renvoie un couple de réels $[val, err]$ où val est la valeur numérique de l'intégrale et err une estimation de l'erreur commise.
Ainsi `quad(sin_fct, -T/2, T/2, args=(n,T,fct))[0]` renverra la valeur de b_n.
Pour plus d'information [https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.quad.html].

9. définir un vecteur `vn` prenant les valeurs entières jusqu'à 100,
10. à l'aide de `plot` représenter le spectre d'amplitude comprenant les 10 premières fréquences et comparer à votre résultat théorique. Que met-on en abscisse ? en ordonnée ? On pourra utiliser `plt.bar(left=x, height=y, width=1)` [http://matplotlib.org/api/pyplot_summary.html].
11. définir une fonction `fct_reconstituee_bn(t, coeff)` qui renvoie le développement de la série de Fourier $S_N(f)$ de coefficients `coeff` évaluée en `t`.
12. représenter graphiquement `fct_reconstituee_bn(t, coeff)` $(S_N(f))$ pour $N \in \{1,3,5,100\}$. Évaluer numériquement le phénomène de Gibbs (on pourra calculer une valeur moyenne et maximale). Comparez avec la valeur limite théorique du maximum $\approx 1{,}178979$.
Vous devriez obtenir quelque chose de semblable à cela :

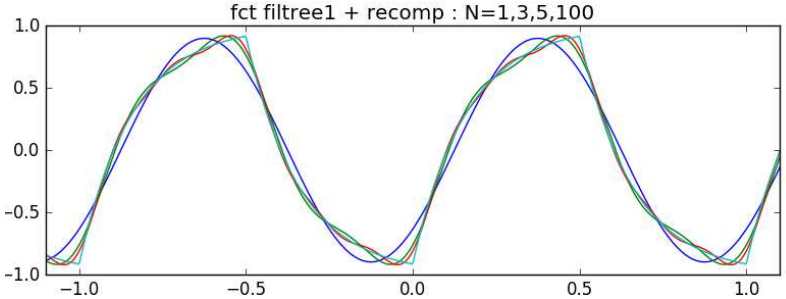


II Filtrage et reconstitution

1 Fonction de transfert et implémentation

On rappelle qu'un filtre passe-bas du 1ère ordre de gain K et de fréquence de coupure f_c a pour fonction de transfert $H(f) = \dfrac{K}{1 + j\frac{f}{fc}}$.

13. définir la fonction de transfert `Hpb(f,fc)` d'un filtre passe-bas d'ordre 1 et de gain unitaire (rq : `j`↔`1j`).
14. définir une fonction `coeff_filtres(coeff, fctH, fc)` qui renvoie les coefficients et les décalages de phase après filtrage par `fctH` (on prendra $f_c = 1$). Représenter le spectre d'amplitude du signal de sortie.
15. définir une fonction `fct_filtree_reconstituee_bn(t, modul, faz, N)` qui reconstitue la sortie tronquée à l'ordre N à partir des coefficients b_n et des phases (*i.e.* renvoie $S_N(\tilde{f})(t)$ si $a_n = 0$). Représenter la fonction filtrée et reconstituée pour différentes valeurs de troncature $N \in \{1,3,5,100\}$.
16. Faites de même pour différentes fréquences de coupure $f_c \in \{\frac{1}{10},1,10\}$. Comment semble se comporter un filtre passe-bas lorsque la f_c est faible ? est élevée ?
Avec Hpb(f,1), vous devriez obtenir quelque chose de semblable à cela :



2 Suite...

En adaptant le code déjà écrit, faire la même implémentation pour la fonction triangulaire et/ou un filtre passe-haut du 1er ordre de gain unitaire $\left(\text{pour rappel, } Hph(f,fc) = \dfrac{Kj\frac{f}{fc}}{1 + j\frac{f}{fc}}\right)$.