Université Claude Bernard Lyon 1

Travail d'initiative personnelle encadré

présenté et réalisé par
Maximilien MAHDHI

Transformée de Fourier discrète et transformée de Fourier rapide

---

sous la direction de Simon MASNOU

Mars 2023

Remerciements

Je tiens tout particulièrement à remercier M.MASNOU pour m'avoir encadrer durant ce travail. Son expertise, sa pédagogie mais aussi son ouverture d'esprit m'ont toujours donné l'envie de comprendre et d'explorer les nouvelles notions auxquelles j'ai été confronté. Je lui suis reconnaissant pour l'expérience qu'il m'a apporté , et qui m'a permis notamment de me détacher des concepts abstraits en but d'étudier leurs applications. La patience qu'il a su m'accorder m'a été très bénéfique, et ses remarques ont régulièrement suscité ma curiosité. Au travers de ce TIPE, je pense avoir acquis de nombreuses compétences et m'être davantage ouvert vers les mathématiques appliquées, pour lesquelles j'éprouve désormais un intérêt plus grand.

Table des matières

1 Introduction

Dans ce mémoire de TIPE, on s'intéresse dans un premier temps à la transformée de Fourier discrète ayant pour but d'approximer les coefficients de Fourier d'un signal périodique et continue par morceaux à partir d'un échantillonnage régulier. Elle constitue l'équivalent discret de la transformée de Fourier et son étude a fortement contribué au développement du monde numérique : on pense notamment à l'imagerie médicale ou plus globalement au domaine audiovisuel.
Dans une seconde partie, on étudie l'algorithme de la transformée de Fourier rapide dont l'objectif est de réduire la complexité numérique de la transformée de Fourier discrète afin d'améliorer considérablement les performances sur ordinateur. En particulier, l'attention de ce mémoire est portée vers l'algorithme de Cooley-Tukey, un algorithme redoutablement célèbre dont le génie et l'efficacité ne sont plus à prouver.
Enfin, il est également question d'aborder une partie importante concernant ces transformées de Fourier par biais de divers tests sur Python, afin d'étudier leurs applications.
Par ailleurs, il est intéressant de noter qu'historiquement ce n'est pas *Joseph Fourier* qui a introduit la transformée de Fourier rapide, mais le mathématicien *Carl Friedrich Gauss* qui avait commencé à développer un algorithme en 1805. Ses travaux ont ensuite été repris par plusieurs mathématiciens, proposant ainsi chacun une nouvelle version de l'algorithme, mais c'est finalement le modèle de deux mathématiciens américains *James Cooley* et *John Tukey* qui a été retenu. C'est d'ailleurs par ce modèle qu'est née la transformée de Fourier discrète.

FIGURE 1 – *Jean Baptiste Joseph Fourier*, 1768 – 1830

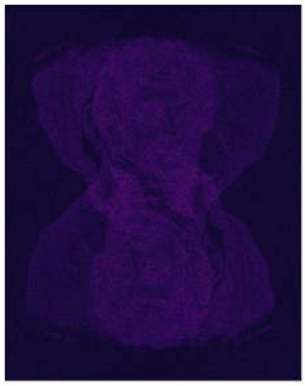


FIGURE 2 – Reconstitution de J. Fourier à partir de la phase de sa transformée de Fourier discrète, avec effet de rotation

2 Transformée de Fourier discrète (DFT)

2.1 Approximation des coefficients de Fourier

On se donne un signal dont le graphe est celui d'une fonction f périodique et continue par morceaux, de période T.
On suppose que ce signal soit échantillonné à des intervalles de temps régulier, et que la taille N de l'échantillon soit fini. On introduit alors $\sigma = (x_0, ..., x_{N-1})$ une subdivision régulière, adaptée à f.
On pose :

$$\forall\, k \in [\![0, N-1]\!], \quad y_k = f(x_k) = f\left(k\frac{T}{N}\right)$$

Dans les hypothèses sur f, on suppose également qu'en tout point t, on ait : $f(t) = \frac{f(t+)+f(t-)}{2}$.
Ainsi, le théorème de Dirichlet assure la convergence de la série de Fourier de f vers f, en tout point de l'intervalle $[0, T]$.
On s'intéresse donc au calcul de N coefficients de Fourier c_n pour $n \in [\![0, N-1]\!]$, où :

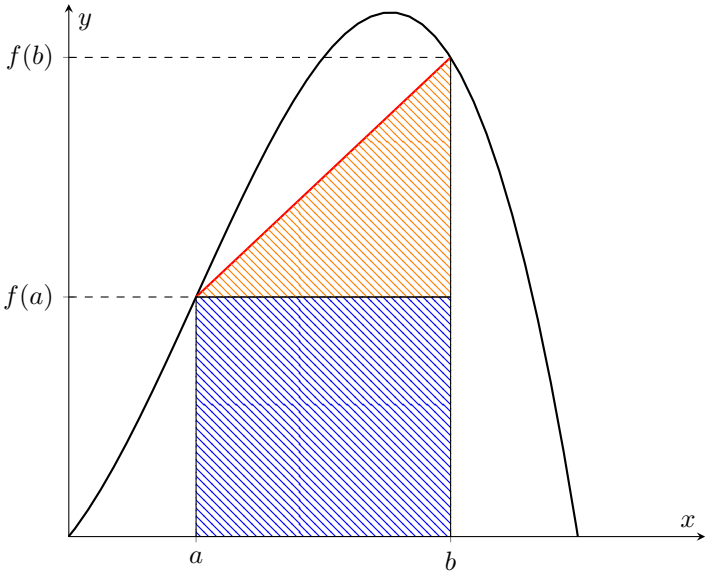
$$c_n = \frac{1}{T}\int_0^T f(t)e^{-2i\pi n\frac{t}{T}}\,dt$$

Le but désormais est d'approcher la valeur de cette intégrale pour obtenir une valeur approximative du coefficient de Fourier c_n.

2.1.1 Formule des trapèzes

Définition 2.1. (Méthode des trapèzes)
Soit $f : [a, b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une fonction continue. On appelle intégration par méthode des trapèzes, l'approximation suivante :

$$\int_a^b f(t)\,dt \approx \frac{f(a)+f(b)}{2}(b-a)$$



Graphique illustrant l'intégration par méthode des trapèzes d'une fonction f continue sur $[a, b]$

En effet, on considère le segment rouge dont les extrémités sont les points $(a, f(a))$ et $(b, f(b))$.
On dit que l'aire sous la courbe sur $[a, b]$ est presque égale à l'aire sous ce segment, soit la somme de l'aire

du triangle rectangle orange, et de celle du rectangle bleu.
Finalement :

$$\mathscr{A}_{approchée} = \mathscr{A}_{triangle} + \mathscr{A}_{rectangle} = \frac{(f(b)-f(a))(b-a)}{2} + f(a)(b-a) = \frac{f(a)+f(b)}{2}(b-a)$$

Remarque : Cette méthode de calcul numérique d'une intégrale repose donc sur une interpolation par un polynôme P de la forme $P(X) = \alpha X + \beta$ sur l'intervalle $[a, b]$ (droite rouge sur le graphique).
Ce type d'interpolation est appelée interpolation linéaire, qui se trouve être la plus simple et naturelle pour approcher une courbe localement.
En effet, elle ne nécessite que 2 points et consiste à approcher le graphe de la fonction localement par un polynôme P de degré 1 (formule de Taylor-Young à l'ordre 1). Par ailleurs, l'appellation de cette méthode est donnée par le trapèze dont on calcule l'aire ici. Ainsi, en notant B et b, resp. la grande et petite base du trapèze, puis h sa hauteur, l'aire du trapèze est donnée par la formule :

$$\mathscr{A}_{trapèze} = \frac{b+B}{2}h = \frac{f(a)+f(b)}{2}(b-a) = \mathscr{A}_{approchée}$$

Appliquons désormais cette méthode à notre problème d'approximation du coefficient c_n.
On pose c'_n comme étant la valeur approchée du coefficient de Fourier c_n. Par relation de Chasles :

$$c_n = \frac{1}{T}\sum_{k=0}^{N-1}\int_{x_k}^{x_{k+1}} f(t)e^{-2i\pi n\frac{t}{T}}\,dt$$

Par intégration des trapèzes :

$$c'_n = \frac{1}{T}\sum_{k=0}^{N-1}(x_{k+1}-x_k)\frac{f(x_k)e^{-2i\pi n\frac{k}{N}} + f(x_{k+1})e^{-2i\pi n\frac{k+1}{N}}}{2}$$

$$= \frac{1}{2N}\left(\sum_{k=0}^{N-1} f(x_k)e^{-2i\pi n\frac{k}{N}} + \sum_{k=0}^{N-1} f(x_{k+1})e^{-2i\pi n\frac{k+1}{N}}\right) = \frac{1}{2N}\left(\sum_{k=0}^{N-1} f(x_k)e^{-2i\pi n\frac{k}{N}} + \sum_{k=1}^{N} f(x_k)e^{-2i\pi n\frac{k}{N}}\right)$$

$$= \frac{1}{2N}\left(\sum_{k=0}^{N-1} f(x_k)e^{-2i\pi n\frac{k}{N}} + \sum_{k=0}^{N-1} f(x_k)e^{-2i\pi n\frac{k}{N}}\right)$$

car $f(x_0)e^{-2i\pi\frac{0}{N}} = f(0) = f(T) = f(x_N)e^{-2i\pi\frac{N}{N}}$ par périodicité de f.

Ainsi, en posant $\omega_N = e^{\frac{2i\pi}{N}}$ on obtient :

$$\boxed{c'_n = \frac{1}{N}\sum_{k=0}^{N-1} y_k\omega_N^{-nk}}$$

2.1.2 Méthode des rectangles

Étonnamment, si on suppose cette fois-ci que f soit en escalier sur $[0, T]$, on arrive à en tirer la même formule que précédemment !
En effet, en considérant que :

$$\forall\, t \in [x_k, x_{k+1}[, \quad g(t) = f(t)e^{-2i\pi n\frac{t}{T}} = g(x_k)$$

on obtient :

$$\begin{aligned} c'_n &= \frac{1}{T}\sum_{k=0}^{N-1}\int_{x_k}^{x_{k+1}} g(t)\,dt \\ &= \frac{1}{T}\sum_{k=0}^{N-1}(x_{k+1}-x_k)g(x_k) \\ c'_n &= \frac{1}{N}\sum_{k=0}^{N-1} y_k\omega_N^{-nk} \end{aligned}$$

où à nouveau $\omega_N = e^{\frac{2i\pi}{N}}$.
Cette méthode est donc plus évidente que la précédente, et elle semble traduire le fait que la vitesse de convergence, de l'une part par rapport à l'autre, ne joue à priori aucun rôle quant à l'approximation des coefficients de Fourier.
Nonobstant, la vitesse de convergence est quadratique ($O(\frac{1}{N^2})$) en ce qui concerne la méthode des trapèzes, ce qui est bien meilleur.
C'est pour cette raison que l'on préférera associer l'approximation à cette première méthode, plutôt qu'à celle des rectangles.

2.1.3 Interpolation par un polynôme trigonométrique

Dans ce sous-paragraphe, nous supposerons que N soit un entier pair pour des raisons de notation. Il est à noter que le cas impair ne pose pas de problèmes, et qu'en pratique on utilise la transformée de Fourier discrète pour N pair. En revanche, nous verrons que ce n'est pas le cas de la transformée de Fourier rapide qui demande à ce que N soit nécessairement une puissance de 2.
Soit f une fonction réelle ou complexe de période T. Soit N un entier pair non-nul.
On cherche un polynôme trigonométrique P de degré $\frac{N}{2}$, de la forme :

$$P(x) = \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{c}_n e^{2i\pi n\frac{x}{T}}$$

qui interpole f en $x_k = k\frac{T}{N}$, pour tout $k \in [\![0, N-1]\!]$.
C'est à dire :

$$P(x_k) = \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{c}_n exp\left(\frac{2i\pi nkT}{TN}\right) = \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{c}_n e^{2i\pi n\frac{k}{N}} = \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{c}_n\omega_N^{nk} = f(x_k) = y_k$$

Par périodicité de f, on peut faire une translation de $\frac{N}{2}$ sur l'intervalle $[\![-\frac{N}{2}, \frac{N}{2}-1]\!]$ afin que n prenne ses valeurs dans $[\![0, N-1]\!]$, et alors on obtient :

$$\boxed{y_k = \sum_{n=0}^{N-1} \tilde{c}_n\omega_N^{nk}}$$

2.1.4 Conclusion

Il est à remarquer que l'on obtient une relation plutôt similaire à celle obtenue avec la formule des trapèzes. Mais finalement, nous allons même prouver qu'elles sont équivalentes et qu'elles sont tout simplement réciproques l'une de l'autre. Pour cela, on va prouver en particulier la proposition suivante.

Proposition 2.2. *Soient c'_n et $\tilde{c_n}$ comme définis précédemment.*
Alors :

$$\boxed{c'_n = \tilde{c_n}}$$

Autrement dit, l'approximation des coefficients de Fourier c_n par méthode des trapèzes revient à interpoler f par un polynôme P trigonométrique (comme défini plus haut), aux points $x_k = k\frac{T}{N}$ pour tout $k \in [\![0, N-1]\!]$.

Preuve. (on prendra garde aux indices)

$$c'_n = \frac{1}{N}\sum_{k=0}^{N-1} y_k \omega_N^{-nk} = \frac{1}{N}\sum_{k=0}^{N-1}\left(\sum_{l=0}^{N-1}\tilde{c}_l\omega_N^{lk}\right)\omega_N^{-nk} = \frac{1}{N}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\tilde{c}_l\omega_N^{k(l-n)} = \frac{1}{N}\sum_{l=0}^{N-1}\tilde{c}_l\sum_{n=0}^{N-1}\omega_N^{k(l-n)}$$

Or,

$$\sum_{n=0}^{N-1}\omega_N^{k(l-n)} = \begin{cases} \frac{1-\omega_N^{N(l-n)}}{1-\omega_N^{k-l}} = 0 \quad si \quad l \neq n \\ N \quad si \quad l = n \end{cases}$$

Finalement :

$$c'_n = \frac{1}{N}\tilde{c_n}N = \tilde{c_n} \qquad \square$$

Notation : Dans la poursuite de cette étude, on fera alors le choix de prendre $c_n^{'}$ pour désigner les coefficients de Fourier approchés.
Convention : Il est possible de poser, pour tout $k \in [\![0, N-1]\!]$, les coefficients x_k tels que : $x_k = \frac{1}{N}y_k$ par associativité, afin d'obtenir une relation plus lisible. Ainsi :

$$\boxed{c'_n = \sum_{k=0}^{N-1} x_k \omega_N^{-nk}}$$

Dans cette étude, on gardera uniquement cette définition pour des raisons de commodité.

2.2 Définition et représentation

D'après le résultat précédent, il en résulte une relation linéaire entre les N valeurs de l'échantillon donné et les N coefficients de Fourier approchés. Ainsi, il est possible de définir la transformée de Fourier discrète comme un endomorphisme de l'espace vectoriel $\mathbb{C}^N$.

Définition 2.3. Soit $N \in \mathbb{N}^*$. La transformée de Fourier discrète d'ordre N est l'application linéaire $\mathscr{F}_N : X = (x_0, ..., x_{N-1}) \in \mathbb{C}^N \longmapsto X' = (X_0, ..., X_{N-1}) \in \mathbb{C}^N$, où :

$$\forall\, k \in [\![0, N-1]\!], \quad X_k = \sum_{n=0}^{N-1} x_n \omega_N^{-nk}, \quad \omega_N = e^{\frac{2i\pi}{N}}$$

Pour simplifier la rédaction, on notera cette relation entre les coordonnées de X, et la $k^{ième}$ coordonnée de X' :

$$(x_n) \xrightarrow{\mathscr{F}_N} (X_k)$$

Proposition 2.4. *(Interpolation du point de vue matriciel)*
Soient $X = (x_0, ..., x_{N-1})$, $X^{'} = (X_0, ..., X_{N-1}) \in \mathbb{C}^N$ où $\mathscr{F}_N(X) = X'$, et soit $\mathcal{B}$ la base canonique de $\mathbb{C}^N$. Alors la matrice V de $\mathscr{F}_N$ dans $\mathcal{B}$ est une matrice de Vandermonde carrée de taille N telle que :

$$X^{'} = VX$$

$$\iff \begin{pmatrix} X_0 \\ \vdots \\ X_{N-1} \end{pmatrix} = \begin{pmatrix} 1 & \alpha_0 & \alpha_0^2 & \dots & \alpha_0^{N-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha_{N-1} & \alpha_{N-1}^2 & \dots & \alpha_{N-1}^{N-1} \end{pmatrix} \begin{pmatrix} x_0 \\ \vdots \\ x_{N-1} \end{pmatrix}, \quad \forall\, k \in [\![0, N-1]\!] \quad \alpha_k = \omega_N^{-k}$$

Remarque : Les coefficients α_k sont les racines N^{ieme} de l'unité.

Preuve. On se donne pour $N \in \mathbb{N}^*$, deux vecteurs $X = (x_0, ..., x_{N-1})$ et $X^{'} = (X_0, ..., X_{N-1})$ de $\mathbb{C}^N$ tels que $\mathscr{F}_N(X) = X'$. Alors d'après la relation de la définition 2.3, on a :

$$\forall\, k \in [\![0, N-1]\!], \quad X_k = \sum_{n=0}^{N-1} x_n \omega_N^{-nk}$$

$$\iff \begin{cases} X_0 & = x_0\omega_N^{-0\times 0} + x_1\omega_N^{-1\times 0} + x_2\omega_N^{-2\times 0} + ... + x_{N-1}\omega_N^{-(N-1)\times 0} \\ X_1 & = x_0\omega_N^{-0\times 1} + x_1\omega_N^{-1\times 1} + x_2\omega_N^{-2\times 1} + ... + x_{N-1}\omega_N^{-(N-1)\times 1} \\ X_2 & = x_0\omega_N^{-0\times 2} + x_1\omega_N^{-1\times 2} + x_2\omega_N^{-2\times 2} + ... + x_{N-1}\omega_N^{-(N-1)\times 2} \\ & \vdots \\ X_{N-1} & = x_0\omega_N^{-0\times (N-1)} + x_1\omega_N^{-1\times (N-1)} + x_2\omega_N^{-2\times (N-1)} + ... + x_{N-1}\omega_N^{-(N-1)^2} \end{cases}$$

$$\iff \begin{pmatrix} X_0 \\ \vdots \\ X_{N-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_N^{-(N-1)} & \omega_N^{-2(N-1)} & \dots & \omega_N^{-(N-1)^2} \end{pmatrix} \begin{pmatrix} x_0 \\ \vdots \\ x_{N-1} \end{pmatrix}$$

En posant $\forall\, k \in [\![0, N-1]\!] \quad \alpha_k = \omega_N^{-k}$, on obtient bien la relation et la matrice V de la proposition 2.4. □

Proposition 2.5. *(Symétrie)*
Soit $\mathcal{B}$ la base canonique de $\mathbb{C}^N$. Pour tout N non nul, la matrice $V = \mathcal{M}_{\mathcal{B}}(\mathscr{F}_N)$, du nom de matrice de Vandermonde-Fourier, est symétrique.

Preuve. Soit $V = (a_{ij})_{1\leq i,j\leq N}$ la matrice de Vandermonde-Fourier à l'ordre N, dans la base $\mathcal{B}$. Alors d'après la proposition 2.4 :

$$a_{ij} = \alpha_{i-1}^{j-1} = \omega_N^{-(j-1)(i-1)} = \omega_N^{-(i-1)(j-1)} = \alpha_{j-1}^{i-1} = a_{ji}$$

ce qui prouve que V est symétrique. □

Le fait que la matrice de Vandermonde-Fourier soit symétrique va permettre de simplifier grandement les calculs, ce qui sera mis à profit dans la partie concernant la transformée de Fourier rapide !

Théorème 2.6. *(Inversibilité)*
Pour tout entier naturel N non-nul, l'application $\mathscr{F}_N$ de la transformée de Fourier discrète est un automorphisme sur $\mathbb{C}^N$. Soit $\mathcal{B}$ la base canonique de $\mathbb{C}^N$ et $V = (v_{lp})_{1\leq l,p\leq N}$, la matrice de $\mathscr{F}_N$ dans $\mathcal{B}$.
Si on a $X = (x_0, ..., x_{N-1})$, $X^{'} = (X_0, ..., X_{N-1}) \in \mathbb{C}^N$ où $\mathscr{F}_N(X) = X'$, alors la transformée de Fourier discrète inverse est l'application $\mathscr{F}_N^{-1}$: $(X_0, ..., X_{N-1}) \in \mathbb{C}^N \longmapsto (x_0, ..., x_{N-1}) \in \mathbb{C}^N$, définie par :

$$(i) \quad \forall\, k \in [\![0, N-1]\!], \quad x_k = \frac{1}{N} \sum_{n=0}^{N-1} X_n \omega_N^{nk}$$

$$(ii) \quad \mathcal{M}_{\mathcal{B}}(\mathscr{F}_N^{-1}) = \frac{1}{N}\overline{V} = \left(\frac{\overline{v_{lp}}}{N}\right)_{1\leq l,p\leq N} \qquad \overline{v_{lp}} = \omega_N^{lp}$$

Preuve. Soit $\mathcal{B}$ la base canonique de $\mathbb{C}^N$. Pour $N \in \mathbb{N}^*$, d'après la proposition 1, on sait que :

$$\mathcal{M}_{\mathcal{B}}(\mathscr{F}_N^{-1}) = \begin{pmatrix} 1 & \alpha_0 & \alpha_0^2 & \dots & \alpha_0^{N-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha_{N-1} & \alpha_{N-1}^2 & \dots & \alpha_{N-1}^{N-1} \end{pmatrix}, \quad \forall\, k \in [\![0, N-1]\!] \quad \alpha_k = \omega_N^{-k}$$

Comme $\mathcal{M}_{\mathcal{B}}(\mathscr{F}_N^{-1})$ est une matrice de Vandermonde de taille N, son déterminant est :

$$\det(V) = \prod_{1\leq l,p\leq N} (\alpha_l - \alpha_p)$$

Donc :

$$\det(V) = 0 \iff \exists (l,p) \in [\![1,N]\!]^2, \quad l \neq p \quad et \quad \alpha_l = \alpha_p$$

Supposons par l'absurde qu'il esiste $(l,p) \in [\![1,N]\!]^2$ tel que $l \neq p$ et $\alpha_l = \alpha_p$. Alors, on aurait :

$$\omega_N^{-l} = \omega_N^{-p}$$
$$Donc \quad e^{-2i\pi \frac{l}{N}} = e^{-2i\pi \frac{p}{N}}$$
$$Donc \quad l = p \quad [2\pi]$$
$$Donc \quad \exists\, k \in \mathbb{Z}^*, \quad l = p + 2k\pi.$$

Or, comme pour tout $k \neq 0$, $2k\pi \notin \mathbb{N}$: on a que $l \notin \mathbb{N}$. (Contradiction)
Donc pour tout $N \in \mathbb{N}^*$, V est inversible soit $\mathscr{F}_N$ est bijective. Comme c'est un endomorphisme de $\mathbb{C}^N$, $\mathscr{F}_N$ est bien un automorphisme de $\mathbb{C}^N$. Vérifions désormais les propriétés (i) et (ii).

$$(i) : \quad S = \frac{1}{N}\sum_{n=0}^{N-1} X_n \omega_N^{nk} = \frac{1}{N}\sum_{n=0}^{N-1}\left(\sum_{l=0}^{N-1} x_l \omega_N^{-nl}\right)\omega_N^{nk} = \frac{1}{N}\sum_{n=0}^{N-1}\sum_{l=0}^{N-1} x_l \omega_N^{n(k-l)} = \frac{1}{N}\sum_{l=0}^{N-1} x_l \sum_{n=0}^{N-1} \omega_N^{n(k-l)}$$

Or,

$$\sum_{n=0}^{N-1} \omega_N^{n(k-l)} = \begin{cases} \frac{1-\omega_N^{N(k-l)}}{1-\omega_N^{k-l}} = 0 & si \quad k \neq l \\ N \quad si \quad k = l \end{cases}$$

Donc, la somme S vaut : $S = \frac{1}{N} x_k N = x_k$, ce qui prouve (i). À noter que cette preuve est celle de la proposition 2.2.
(ii) : Conséquence immédiate de la relation obtenue en (i). En posant $\alpha_k = \overline{\omega_N^{-k}} = \omega_N^k$, la preuve est analogue à celle de la proposition 1. □

La transformée de Fourier discrète est donc une transformation inversible, indépendamment de la taille N de l'échantillon. De plus, le fort intérêt de cette transformation est qu'elle soit linéaire, et que sa matrice soit facilement inversible.
En effet, nous venons de voir qu'il suffit simplement de prendre le conjugué de chaque coefficient de la matrice, puis de les diviser par N pour obtenir la transformation inverse ! C'est là un grand avantage en terme de complexité numérique, qui s'avérera utile dans la transformée de Fourier rapide que nous étudierons.

2.3 Analyse de l'erreur de quadrature

Dans cette partie, nous allons nous intéresser à l'erreur de cette approximation en considérant les cas optimaux et défavorables.
Tout d'abord, il est clair qu'un paramètre majeur sera la taille N de l'échantillon donné. En effet, l'intégration du signal sur sa période repose sur une coupe d'espacement régulier $\frac{T}{N}$ qui devient petit à mesure que N devient grand.

2.3.1 Première approche

En considérant cela, commençons alors par évaluer l'erreur issue de la formule des trapèzes que nous avons utilisé.
Préliminaires : On rappelle que l'on a défini les coefficients de Fourier du signal f sur $[0,T]$, en supposant que f coïncide avec sa série de Fourier en tout point de $[0,T]$, c'est à dire :

$$\forall\, t \in [0,T], \quad f(t) = \sum_{n=-\infty}^{+\infty} c_n(f) e^{2i\pi n \frac{t}{T}} \quad où \quad c_{-n}(f) = \overline{c_n(f)}$$

Donc dans le cadre de cette étude, f n'est pas nécessairement $\mathscr{C}^\infty$ dû aux potentiels points de discontinuité de la fonction.
De plus, il est à noter pour ce qui suit, que la fonction $t \longmapsto e^{-2i\pi n\frac{t}{T}}$ est développable en série entière $\forall\, t \in \mathbb{R}$ (série exponentielle de rayon de convergence infini), qui coïncide avec sa série de Maclaurin. Donc, elle est nécessairement $\mathscr{C}^\infty$ sur $[0,T]$.
Dans ce premier sous paragraphe, nous allons essentiellement nous servir de la formule de Taylor-Lagrange. Seulement, ici les fonctions sont à valeurs complexes, donc pour cela, nous allons identifier le corps complexe $\mathbb{C}$ à l'espace vectoriel $\mathbb{R}^2$.
Cela va donc nous permettre d'utiliser le théorème de Taylor-Lagrange avec reste intégral qui est valable pour des applications à valeurs dans un espace de Banach, c'est à dire un espace vectoriel complet et normé sur un sous-corps $\mathbb{K}$ de $\mathbb{C}$.
En premier lieu, nous allons prendre le soin d'énoncer et de démontrer ce théorème.

Théorème 2.7. *(Formule de Taylor-Lagrange avec reste intégral)*
Soit E un espace de Banach. Soit $f : [a,x] \to E$ une application de classe $\mathscr{C}^{n+1}([a,x])$, alors :

$$f(x) = f(a) + f'(a)(x-a) + ... + \frac{f^{(n)}(a)}{n!}(x-a)^n + \int_a^x \frac{(x-t)^n}{n!} f^{(n+1)}(t)\, dt$$

Preuve. Montrons-le par récurrence. On pose le prédicat (P_n) = "f vérifie la formule du théorème, sous réserve qu'elle vérifie bien les hypothèses énoncées".
Initialisation : Par le théorème fondamental de l'analyse, on a que :

$$f(x) = f(a) + \int_a^x f'(t)\, dt$$

ce qui prouve (P_0).
Hérédité : Supposons que (P_n) soit vrai pour un rang n fixé. Montrons que (P_{n+1}) est vrai également.
Dans un premier temps, comme on a supposé (P_n) vrai, on a :

$$f(x) = f(a) + f'(a)(x-a) + ... + \frac{f^{(n)}(a)}{n!}(x-a)^n + \int_a^x \frac{(x-t)^n}{n!} f^{(n+1)}(t)\, dt$$

Donc si l'application f est supposée $\mathscr{C}^{n+2}([a,x])$, par intégration par parties on obtient que :

$$\begin{aligned}\int_a^x \frac{(x-t)^n}{n!} f^{(n+1)}(t)\, dt &= \left[-\frac{(x-t)^{n+1}}{(n+1)!} f^{(n+1)}(t)\right]_a^x + \int_a^x \frac{(x-t)^{n+1}}{(n+1)!} f^{(n+2)}(t)\, dt \\ &= \frac{(x-a)^{n+1}}{(n+1)!} f^{(n+1)}(a) + \int_a^x \frac{(x-t)^{n+1}}{(n+1)!} f^{(n+2)}(t)\, dt\end{aligned}$$

ce qui prouve (P_{n+1}) et donc l'hérédité. □

Proposition 2.8. *Supposons f une fonction T-périodique de classe $\mathscr{C}^2$ sur $[0,T]$. Soient $c_n(f)$ ses coefficients de Fourier exacts, et $c'_n(f)$ ses coefficients de Fourier approchés par méthode des trapèzes sur $[0,T]$.*
Alors :

$$\exists\, M \geq 0, \quad |c_n(f) - c'_n(f)| \leq \frac{5T^3}{12N^2} M$$

Preuve. Posons $g : t \in [0,T] \subset \mathbb{R} \longmapsto f(t)e^{-2i\pi n\frac{t}{T}} \in \mathbb{C}$ de classe $\mathscr{C}^2$, comme produit de fonctions $\mathscr{C}^2$.
Tout d'abord, on cherche à évaluer l'erreur ε_k commise pour chaque sous-intervalle $[x_k, x_{k+1}]$, soit :

$$\forall\, k \in [\![0, N-1]\!], \quad \varepsilon_k = \int_{x_k}^{x_{k+1}} g(t)\, dt - \frac{T}{2N}(g(x_k) + g(x_{k+1}))$$

Comme pour tout $t \in [x_k, x_{k+1}]$, $g \in \mathscr{C}^2([x_k, t])$, on peut utiliser la formule de Taylor-Lagrange avec reste intégral (*cf. Théorème 2.7*) à l'ordre 2 :

$$g(t) = g(x_k) + g'(x_k)(t - x_k) + \int_{x_k}^t (t-u) g''(u)\, du$$

Or, comme g est $\mathscr{C}^2$ sur $[0,T]$, g'' est alors bornée sur cet intervalle. Donc par croissance de l'intégrale :

$$\exists\, M \geq 0, \quad \forall\, k \in [\![0, N-1]\!], \quad |g''(x_k)| \leq M \implies \left| \int_{x_k}^{t} (t-u) g''(u)\, du \right| \leq \left| -\frac{1}{2} M \left[(t-u)^2 \right]_{x_k}^{t} \right| = \frac{1}{2}(t-x_k)^2 M$$

Puis, on obtient par linéarité :

$$\int_{x_k}^{x_{k+1}} g(t)\, dt \leq h g(x_k) + \frac{1}{2} h^2 g'(x_k) + \frac{1}{6} h^3 M \quad \text{où} \quad h = x_{k+1} - x_k = \frac{T}{N}$$

On a aussi, en évaluant le développement de g en $t = x_{k+1}$:

$$\frac{T}{2N}(g(x_k) + g(x_{k+1})) = h g(x_k) + \frac{1}{2} h^2 g'(x_k) + \frac{1}{2} h \int_{x_k}^{x_{k+1}} (x_{k+1} - u) g''(u)\, du$$

Donc :

$$\varepsilon_k \leq \frac{1}{6} h^3 M - \frac{1}{2} h \int_{x_k}^{x_{k+1}} (x_{k+1} - u) g''(u)\, du$$

À nouveau, par hypothèse sur g :

$$\forall\, k \in [\![0, N-1]\!], \quad |g''(x_k)| \leq M \implies |\varepsilon_k| \leq \frac{1}{6} h^3 M + \frac{1}{4} h \left| \left[(x_{k+1} - u)^2 \right]_{x_k}^{x_{k+1}} \right| = \left(\frac{1}{6} + \frac{1}{4} \right) h^3 M = \frac{5T^3}{12N^3} M$$

Finalement, en prenant ε comme étant la somme des erreurs ε_k commises dans chaque sous-intervalle :

$$|\varepsilon| = |c_n(f) - c_n'(f)| = \left| \sum_{k=0}^{N-1} \varepsilon_k \right| \leq \sum_{k=0}^{N-1} |\varepsilon_k| \leq \sum_{k=0}^{N-1} \frac{5T^3}{12N^3} M = \frac{5T^3}{12N^2} M \qquad \square$$

Remarque : On constate donc que l'erreur ε commise est de l'ordre de $O(\frac{1}{N^2})$.

2.3.2 Relation linéaire entre c_n et c_n'

Dans cette deuxième approche, nous allons tenter de déterminer une relation explicite entre les coefficients de Fourier exacts c_n, et ses coefficients approchés $c_n^{'}$. Pour cela, il est nécessaire de revenir à la série de Fourier de f sous sa forme exponentielle :

$$f(t) = \sum_{n=-\infty}^{+\infty} c_n e^{2i\pi n \frac{t}{T}}$$

Ici, nous allons supposer que la série soit absolument convergente, c'est à dire que :

$$\sum_{n=-\infty}^{+\infty} |c_n| < +\infty$$

A partir de cette hypothèse, l'idée est de regrouper astucieusement les coefficients c_n par indice.
Avant cela, il est cependant judicieux d'introduire brièvement la notion de familles sommables.

Définition 2.9. (Ensemble dénombrable)
Un ensemble E est dit *dénombrable* s'il existe une bijection $\sigma : E \to \mathbb{N}$.
Autrement dit, un ensemble est qualifié de dénombrable si on peut énumérer ses éléments.

Définition 2.10. (famille réelle)
Soient I un ensemble dénombrable, et $(a_i)_{i\in I}$ une famille de réels positifs. Alors, on dit que $(a_i)_{i\in I}$ est une famille sommable, si et seulement si :

$$A = \left\{ \sum_{i\in F} a_i \mid F \text{ partie finie de } I \right\}$$

est majoré.
Dans ce cas, on appelle $\sup(A)$ la somme de la famille $(a_i)_{i\in I}$. Si la famille n'est pas sommable, on dit que sa somme vaut $+\infty$.

On dispose également de la définition pour une famille de nombre complexes (utile dans notre cas de figure).

Définition 2.11. (famille complexe)
Soient I un ensemble dénombrable, et $(a_i)_{i\in I}$ une famille de réels complexes. Alors, $(a_i)_{i\in I}$ est une famille sommable, si et seulement si :

(i) La famille $(|a_i|)_{i\in I}$ est une famille sommable.

(ii) $(\mathfrak{Re}(a_i))_{i\in I}$, $(\mathfrak{Im}(a_i))_{i\in I}$ sont des familles sommables. Dans ce cas, on a :

$$\sum_{i\in I} a_i = \sum_{i\in I} \mathfrak{Re}(a_i) + i\sum_{i\in I} \mathfrak{Im}(a_i)$$

Proposition 2.12. *Soit $(a_n)_{n\in\mathbb{N}}$ une famille de réels positifs ou de complexes, indexée par $\mathbb{N}$.*

(i) Si $(a_n)_{n\in\mathbb{N}}$ est une famille de réels positifs, alors elle est sommable, si et seulement si :

$$\sum_{n\in\mathbb{N}} a_n < +\infty$$

(ii) Si $(a_n)_{n\in\mathbb{N}}$ est une famille de complexes, alors elle est sommable, si et seulement si :

$$\sum_{n\in\mathbb{N}} |a_n| < +\infty$$

Preuve. (i) $\implies$) Supposons que la famille $(a_i)_{i\in\mathbb{N}}$ soit sommable.
Posons $S(a)$, la somme de cette famille, et posons $I_n = \{0, ..., n\}$. Alors pour tout $n \in \mathbb{N}$, I_n est une partie finie de $\mathbb{N}$, et donc :

$$\forall\, n \in \mathbb{N}, \quad \sum_{k=0}^{n} a_k = \sum_{i\in I_n} a_i \leq S(a)$$

ce qui prouve que la série $\sum_{n\geq 0} a_n$ converge.
$\impliedby$) Supposons que le série $\sum_{n\geq 0} a_n$ converge. Soit $J \subset \mathbb{N}$, fini non-vide, alors J admet un plus grand élément n.
Puisque $J \subset I_n$ et que la suite $(a_i)_{i\in\mathbb{N}}$ est positive, on a :

$$\sum_{k\in J} a_k \leq \sum_{k=0}^{n} a_k \leq \sum_{n=0}^{+\infty} a_n$$

Donc, il vient que l'ensemble A, comme défini dans la définition 2.10, est majoré par $\sum_{n\geq 0} a_n$.
Donc la famille $(a_i)_{i\in\mathbb{N}}$ est sommable.
(ii) D'après le point (i) de la définition 2.11 avec $I = \mathbb{N}$, $(a_n)_{n\in\mathbb{N}}$ est sommable, si et seulement si, $(|a_n|)_{n\in\mathbb{N}}$ est sommable.
Or, d'après le point (i) que nous venons démontrer, cela équivaut à ce que la série $\sum_{n\geq 0}|a_n|$ converge. □

Dans notre cas particulier, la série de Fourier de f est une famille complexe $(c_n)_{n\in\mathbb{Z}}$ composée des coefficients de Fourier.
Or, d'après la proposition 2.12, c'est une famille sommable, si et seulement si, la série de Fourier de f converge absolument. On a donc la proposition suivante :

Proposition 2.13. *Pour tout $t \in [0, T]$, la série de Fourier de f :*

$$\sum_{n=-\infty}^{+\infty} c_n e^{2i\pi n \frac{t}{T}}$$

est une famille sommable, si et seulement si,

$$\sum_{n=-\infty}^{+\infty} |c_n| < +\infty$$

Preuve. On applique le point (ii) de la proposition 2.12 sur la famille $(c_n e^{2i\pi n \frac{t}{T}})_{n\in\mathbb{Z}}$, avec $I = \mathbb{Z}$. $\square$

Donc cette proposition 2.13 nous dit que l'on peut grouper les indices par paquets. On va ici faire le choix particulier de les grouper selon leur reste dans la division euclidienne par N.
Ainsi, on pose $\mathcal{P}$ comme étant la partition de $\mathbb{Z}$ définie par :

$$\mathcal{P} = \bigcup_{i\in[\![0,N-1]\!]} A_i\,, \quad A_i = \{n \in \mathbb{Z} \mid n \equiv i\,[N]\}$$

On peut alors écrire, en posant $n = m + qN$ et $\omega_N = e^{\frac{2i\pi}{N}}$:

$$f\left(k\frac{T}{N}\right) = y_k = \sum_{n=-\infty}^{+\infty} c_n \omega_N^{nk} = \sum_{m=0}^{N-1} \sum_{n\in A_m} c_n \omega_N^{nk}$$

$$= \sum_{m=0}^{N-1} \sum_{q=-\infty}^{+\infty} (c_{m+qN}) \omega_N^{(m+qN)k}$$

Or, $\omega_N^{(m+qN)k} = \omega_N^{mk+qNk} = \omega_N^{mk}$.
Donc :

$$y_k = \sum_{m=0}^{N-1} \left(\sum_{q=-\infty}^{+\infty} c_{m+qN} \right) \omega_N^{mk}$$

Donc, par la définition de la transformée de Fourier discrète inverse, on reconnaît que :

$$\boxed{c'_m = \sum_{q=-\infty}^{+\infty} c_{m+qN}}$$

Remarque : Cette relation est à la fois sublime et fascinante, car elle traduit une relation exacte entre les coefficients de Fourier c_n et leurs valeurs approchées c'_n !
Finalement, on en déduit le théorème suivant.

Théorème 2.14. *Soient c_n et c'_n comme définis précédemment. Alors :*

$$(i) \quad c'_n - c_n = \sum_{\substack{q=-\infty \\ q\neq 0}}^{+\infty} c_{n+qN}$$

$$(ii) \quad \exists\, M \geq 0, \quad \left| \sum_{\substack{q=-\infty \\ q\neq 0}}^{+\infty} c_{n+qN} \right| \leq \frac{5T^3}{12N^2} M$$

Preuve. (i) : Se déduit à partir du résultat précédent (avec $m = n$) en soustrayant par c_n de par et d'autre de l'égalité, ce qui revient à retirer le terme d'indice $q = 0$ ($n + 0 \times N = n$) de la somme.
(ii) : D'après la proposition 2.8 et le point (i) ci-dessus :

$$\exists\, M \geq 0, \quad \left| \sum_{\substack{q=-\infty \\ q\neq 0}}^{+\infty} c_{n+qN} \right| = |c'_n - c_n| \leq \frac{5T^3}{12N^2} M \quad \square$$

Ce théorème montre que l'erreur ε commise par la transformation de Fourier discrète est somme des coefficients de Fourier exacts, issus de fréquences $n + qN$ (avec $q \neq 0$) plus hautes. Ceci se traduit par un phénomène nommé "aliasing" ou "repliement de spectre" qui affecte la qualité des images.
L'introduction de ce phénomène est présentée dans le sous-paragraphe 4.2.3 de cette étude.
Le théorème va également nous permettre d'établir un lien entre la régularité du signal f et l'erreur ε.

2.3.3 Qualité de l'approximation

En effet, il nous est possible d'étudier des cas favorables et défavorables de notre approximation des coefficients de Fourier. Dans cette idée, on s'intéresse au comportement asymptotique des coefficients de Fourier de f. Pour cela, on dispose du lemme de Riemann-Lebesgue qui stipule les éléments suivants.

Lemme 2.15. *(Riemann-Lebesgue)*
Soit I un intervalle de $\mathbb{R}$. On suppose un signal $f : I \to \mathbb{C}$ intégrable et de classe $\mathscr{C}^k(I)$ avec $k \geq 0$.
Alors :

$$\lim_{x\to\pm\infty} \int_I f(t)e^{-2i\pi xt}\,dt = 0$$

En particulier, la convergence de cette intégrale au voisinage de $\pm\infty$ est de l'ordre de $O\left(\frac{1}{|x|^k}\right)$.

Preuve. Sachant que la preuve pour $k = 0$ ne nous intéresse pas et qu'elle est moins immédiate, nous décidons de la laisser au lecteur.
Pour $k \geq 1$: Soit f un signal vérifiant les hypothèses énoncées. On pose $I = [a, b]$.
On procède par intégration par parties :

$$\int_I f(t)e^{-2i\pi xt}\,dt = -\frac{1}{2i\pi x}\left[f(t)e^{-2i\pi xt}\right]_a^b + \frac{1}{2i\pi x}\int_I f'(t)e^{-2i\pi xt}\,dt$$

Or,

$$\left|-\frac{1}{2i\pi x}\left[f(t)e^{-2i\pi xt}\right]_a^b\right| \leq \frac{1}{2\pi|x|}(|f(a)| + |f(b)|) \xrightarrow[x\to\pm\infty]{} 0$$

Donc :

$$\int_I f(t)e^{-2i\pi xt}\,dt \underset{x\to\pm\infty}{\sim} \frac{1}{2i\pi x}\int_I f'(t)e^{-2i\pi xt}\,dt$$

Or, de proche en proche, on a :

$$\frac{1}{2i\pi x}\int_I f'(t)e^{-2i\pi xt}\,dt \underset{x\to\pm\infty}{\sim} \frac{1}{(2i\pi x)^k}\int_I f^{(k)}(t)e^{-2i\pi xt}\,dt$$

Donc par transitivité :

$$\int_I f(t)e^{-2i\pi xt}\,dt \underset{x\to\pm\infty}{\sim} \frac{1}{(2i\pi x)^k}\int_I f^{(k)}(t)e^{-2i\pi xt}\,dt$$

Donc pour $|x|$ assez grand, on a :

$$\left|\int_I f(t)e^{-2i\pi xt}\,dt\right| \leq \frac{1}{2\pi|x|^k}\int_I |f^{(k)}(t)|\,dt$$

Or, comme f est $\mathscr{C}^k$ sur I, $|f^{(k)}(t)| \leq M \in \mathbb{R}_+$.
Donc :

$$\left|\int_I f(t)e^{-2i\pi xt}\,dt\right| \leq \frac{(b-a)M}{2\pi|x|^k} \xrightarrow[x\to\pm\infty]{} 0$$

ce qui prouve que l'intégrale sur I converge vers 0 en $\pm\infty$ et en $O\left(\frac{1}{|x|^k}\right)$ (pour $k \geq 1$). □

Grâce au lemme 2.15, on comprend que la transformation de Fourier discrète est particulièrement efficace à mesure que l'on considère des signaux de plus en plus réguliers.
En effet, en appliquant le lemme de Riemann-Lebesgue pour $I = [0, T]$ (notre intervalle discrétisé) et un signal f de classe $\mathscr{C}^k(I)$, on obtient que :

$$\boxed{c_n(f) = \frac{1}{T}\int_0^T f(t)e^{-2i\pi n\frac{t}{T}}\,dt \xrightarrow[n\to\pm\infty]{} 0 \quad en \quad O\left(\frac{1}{|n|^k}\right)}$$

Donc, par le théorème 2.14, on en déduit plus précisément que l'erreur ε valant la série :

$$\sum_{\substack{q=-\infty \\ q\neq 0}}^{+\infty} c_{n+qN}$$

aura une convergence vers 0 plus rapide si f est assez régulière (k assez grand), car le terme général c_{n+qN} tendra plus vite vers 0 d'après le lemme 2.15.
Par ailleurs, on se rappelle que l'approximation $c_n \approx c'_n$ est issue d'une interpolation par un polynôme P trigonométrique de degré $\frac{N}{2} - 1$, valable pour

$$-\frac{N}{2} \leq n \leq \frac{N}{2} - 1$$

à N fixé.
Ainsi, on voit bien que n est limité dans les grandes valeurs qu'il peut prendre. En ce sens, on comprend qu'il est nécessaire qu'il y ait une convergence suffisamment véloce des coefficients de Fourier vers 0, pour pouvoir prétendre à une approximation assez satisfaisante.
Donc à nouveau, si le signal est suffisamment régulier, la transformée Fourier discrète approchera de manière plus précise les coefficients de Fourier.
Le cas optimal serait, par ailleurs, que le signal considéré soit $\mathscr{C}^\infty$.
Un cas défavorable, au contraire, serait que le signal soit discontinue sur $[0, T]$ car la vitesse de convergence des coefficients de Fourier serait médiocre. Nous pouvons donc établir le théorème qui suit.

Théorème 2.16. *Plus le signal f est supposé régulier sur $[0, T]$, plus l'approximation des coefficients de Fourier c_n est satisfaisante. Autrement dit, la transformée de Fourier discrète est efficace à mesure que les signaux soient supposés réguliers sur l'intervalle discrétisé.*

Preuve. Découle immédiatement du raisonnement précédent.

2.4 Quelques propriétés abstraites

Pour cette sous-partie, on définit E comme un espace vectoriel de $\mathbb{C}^N$, et $(E, \langle . \mid . \rangle)$ un espace hermitien de dimension N muni du produit hermitien canonique $\langle . \mid . \rangle$. Notons $\|.\|$ la norme associée au produit hermitien.

Notation : Pour ce qui suit, on pose pour tout $n \in [\![0, N-1]\!]$, $x_{-n} = x_{N-n}$. Autrement dit, par périodicité du signal f sur $[0, T]$, on est en mesure de discrétiser l'intervalle dans l'ordre opposé.

Lemme 2.17. *Soient $X = (x_0, ..., x_{N-1}), X' = (X_0, ..., X_{N-1}) \in E$ tels que $\mathscr{F}_N(X) = X'$. Alors :*

$$\begin{array}{ll} (i) & (x_{-n}) \xrightarrow{\mathscr{F}_N} (X_{-k}) \\ (ii) & (\overline{x}_n) \xrightarrow{\mathscr{F}_N} (\overline{X}_{-k}) \\ (iii) & (\overline{x}_{-n}) \xrightarrow{\mathscr{F}_N} (\overline{X}_k) \end{array}$$

De plus, la réciproque de chacune de ces 3 relations est vérifiée par la TFD inverse $\mathscr{F}_N^{-1}$.

Preuve. On rappelle que pour tout $p \in \mathbb{Z}$: $\overline{\omega_N^p} = \overline{(e^{\frac{2i\pi}{N}})^p} = \omega_N^{-p}$, et $\forall (z_1, z_2) \in \mathbb{C}^2$, $\overline{z_1 + z_2} = \overline{z_1} + \overline{z_2}$.

$$(i) : \quad \sum_{n=0}^{N-1} x_{-n}\omega_N^{-kn} = \sum_{n=1-N}^{0} x_n\omega_N^{kn} = X_{-k}$$

$$(ii) : \quad \sum_{n=0}^{N-1} \overline{x}_n\omega_N^{-kn} = \sum_{n=0}^{N-1} \overline{x_n\omega_N^{kn}} = \overline{\sum_{n=0}^{N-1} x_n\omega_N^{kn}} = \overline{X}_{-k}$$

$$(iii) : \quad \sum_{n=0}^{N-1} \overline{x}_{-n}\omega_N^{-kn} = \sum_{n=1-N}^{0} \overline{x}_n\omega_N^{kn} = \sum_{n=1-N}^{0} \overline{x_n\omega_N^{-kn}} = \overline{\sum_{n=1-N}^{0} x_n\omega_N^{-kn}} = \overline{X}_k$$

Enfin, la réciproque de ces 3 relations est vrai en raison de l'inversibilité de la TFD $\mathscr{F}_N$ à tout ordre.
La preuve se démontre de la même façon. □

Théorème 2.18. *(Egalité de Parseval)*
Soient $X = (x_0, ..., x_{N-1}), X' = (X_0, ..., X_{N-1}) \in E$ *tels que* $\mathscr{F}_N(X) = X'$. *Alors, on dispose de l'égalité suivante :*

$$\sum_{n=0}^{N-1} |x_n|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |X_n|^2$$

Preuve.

$$\sum_{n=0}^{N-1} |x_n|^2 = \sum_{n=0}^{N-1} x_n \overline{x}_n$$

Or, d'après la réciproque de la relation (*iii*) du lemme 2.17, on a :

$$\overline{x}_{-n} = \frac{1}{N} \sum_{l=0}^{N-1} \overline{X}_l \omega_N^{nl}$$

soit :

$$\overline{x}_n = \frac{1}{N} \sum_{l=0}^{N-1} \overline{X}_l \omega_N^{-nl}$$

Donc :

$$\sum_{n=0}^{N-1} |x_n|^2 = \sum_{n=0}^{N-1} \left(\frac{1}{N} \sum_{k=0}^{N-1} X_k \omega_N^{nk} \right) \left(\frac{1}{N} \sum_{l=0}^{N-1} \overline{X}_l \omega_N^{-nl} \right)$$

$$= \frac{1}{N^2} \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} X_k \overline{X}_l \omega_N^{n(k-l)} = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} X_k \overline{X}_l \left(\sum_{n=0}^{N-1} \omega_N^{n(k-l)} \right)$$

Or,

$$\sum_{n=0}^{N-1} \omega_N^{n(k-l)} = \begin{cases} \frac{1-\omega_N^{N(k-l)}}{1-\omega_N^{k-l}} = 0 & si \quad k \neq l \\ N \quad si \quad k = l \end{cases}$$

Finalement :

$$\sum_{n=0}^{N-1} |x_n|^2 = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} X_k \overline{X}_k = \frac{1}{N^2} \sum_{k=0}^{N-1} N |X_k|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X_k|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |X_n|^2. \quad \square$$

Corollaire 2.19. *Soient* $X = (x_0, ..., x_{N-1}), X' = (X_0, ..., X_{N-1}) \in E$ *tels que* $\mathscr{F}_N(X) = X'$.
Alors :

$$\|X'\| = \sqrt{N} \|X\|$$

Preuve. D'après la proposition 2.18, la preuve est immédiate. En effet, on a :

$$\|X'\|^2 = \langle X' \mid X' \rangle = \sum_{n=0}^{N-1} \overline{X_n} X_n = \sum_{n=0}^{N-1} |X_n|^2 = N \sum_{n=0}^{N-1} |x_n|^2 = N \|X\|^2$$

Donc :

$$\|X'\| = \sqrt{N} \|X\| \qquad \square$$

Remarque : On constate donc que la transformée de Fourier discrète est une isométrie à un facteur près qui dépend de l'ordre N (pour le produit scalaire et la norme choisis).
Si $N = 1$, c'est une isométrie mais qui ne porte aucun intérêt. On peut également en déduire que la matrice V (comme définie dans la définition 2.3) est orthogonale, si et seulement si, $N = 1$.
A nouveau, elle l'est à un facteur près qui dépend de N.

Lemme 2.20. *(Colinéarité)*
Il existe une infinité de vecteurs $X = (x_0, ..., x_{N-1}), X' = (X_0, ..., X_{N-1}) \in E$ *colinéaires vérifiant :*

$$\mathscr{F}_N(X) = X'$$

Preuve. Comme $\mathscr{F}_N$ est un endomorphisme de $\mathbb{C}^N$, le problème revient à prouver l'existence d'espaces propres E_λ, avec λ réel ou complexe.
Or, dans le corps $\mathbb{C}$, toute matrice d'endomorphisme est diagonalisable et admet des valeurs propres, en particulier $\mathcal{M}_{\mathcal{B}}(\mathscr{F}_N)$ dans toute base $\mathcal{B}$ de E.
Par conséquent, $\mathscr{F}_N$ admet un espace propre E_λ pour chaque valeur propre λ. □

Corollaire 2.21. *(Orthogonalité)*
Soient $X = (x_0, ..., x_{N-1}), X^{'} = (X_0, ..., X_{N-1}) \in E$ *tels que :* $\mathscr{F}_N(X) = X'$.

$$\langle X \mid X^{'} \rangle = 0 \iff X = X' = 0_{\mathbb{C}^N}$$

Preuve. $\Longleftarrow$) Evident car : $\langle 0_{\mathbb{C}^N} \mid 0_{\mathbb{C}^N} \rangle = 0$.
$\Longrightarrow$) Supposons que $\langle X \mid X' \rangle = 0$.
Utilisons l'inégalité de Cauchy-Schwarz, qui est bien définie dans un espace hermitien. Alors, en utilisant le résultat du corollaire 2.19, on a :

$$|\langle X \mid X' \rangle| \leq \|X\|\|X'\| = \sqrt{N}\|X\|^2 = \frac{1}{\sqrt{N}}\|X'\|^2$$

Or, d'après le lemme 2.20, comme il peut y avoir cas d'égalité (dans le cas où les vecteurs sont colinéaires), il vient que :

$$\sup_{N \in \mathbb{N}^*} |\langle X \mid X' \rangle| = \sqrt{N}\|X\|^2 = \frac{1}{\sqrt{N}}\|X'\|^2$$

Cependant, on a supposé que $\langle X \mid X' \rangle = 0$, donc nécessairement $\sup_{N \in \mathbb{N}^*} |\langle X \mid X' \rangle| = 0$.
Donc, comme $\sqrt{N} \neq 0$ et $\frac{1}{\sqrt{N}} \neq 0$, on en déduit que $\|X\| = \|X'\| = 0$.
Donc $X = X' = 0_{\mathbb{C}^N}$ par propriété de séparation de la norme. □

Remarque : En pratique, comme il n'y a aucun intérêt à utiliser un échantillon ne contenant que des valeurs nulles, on peut alors affirmer sans problèmes que les vecteurs X et X' ne seront *jamais* orthogonaux.

Proposition 2.22. *Pour tout N entier non nul, $\mathscr{F}_N$ est un endomorphisme normal et son adjoint (pour le produit hermitien canonique) est l'application* $\mathscr{F}_N^* : (X_0, ..., X_{N-1}) \in \mathbb{C}^N \longmapsto (x_0, ..., x_{N-1}) \in \mathbb{C}^N$ *où :*

$$\forall\, k \in [\![0, N-1]\!], \quad x_k = \sum_{n=0}^{N-1} X_n \omega_N^{kn}$$

Preuve. Déterminons l'adjoint de $\mathscr{F}_N$. On le fait matriciellement en posant V, V^* et A comme étant resp. les matrices de $\mathscr{F}_N$, $\mathscr{F}_N^*$ et celle du produit hermitien canonique.
Donc $A = I_2$ et alors pour tout $X, Y \in \mathbb{C}^N$:

$$\begin{aligned} &\langle \mathscr{F}_N(X) \mid X' \rangle = \langle X \mid \mathscr{F}_N^* \rangle \\ \iff &(\overline{V}\overline{X})^t AY = (\overline{X})^t AV^*Y \\ \iff &(\overline{X})^t (\overline{V})^t Y = (\overline{X})^t V^* Y \end{aligned}$$

Donc, il faut que $(\overline{V})^t = V^*$. Or, comme V est symétrique (cf. proposition 2.5), $\overline{V}$ l'est également, donc :

$$V^* = \overline{V}$$

Par ailleurs, par le théorème 2.6, on a $\overline{V} = NV^{-1}$. Donc :

$$\begin{aligned} &VV^{-1} = V^{-1}V = I_2 \\ \iff &V\frac{1}{N}\overline{V} = \frac{1}{N}\overline{V}V \\ \iff &V\overline{V} = \overline{V}V \end{aligned}$$

ce qui prouve que $\mathscr{F}_N$ est normal.
De plus, $V^{-1}X = \frac{1}{N}\overline{V}X$ donc $V^*X = NV^{-1}X$.
Donc pour tout entier N non nul, $\mathscr{F}_N^* = N\mathscr{F}_N^{-1}$, et donc par le point (i) du théorème 2.6, on en déduit la relation définie pour $\mathscr{F}_N^*$. □

3 Transformée de Fourier rapide (FFT)

3.1 Principe et essence de la FFT

3.1.1 Complexité d'une multiplication de deux polynômes

Pour arriver à comprendre le fonctionnement de la transformée de Fourier rapide, intéressons-nous à la complexité numérique d'un calcul de produit de deux polynômes P et Q quelconques de même degré d.
Posons :

$$P(x) = \sum_{k=0}^{d} a_k x^k \quad Q(x) = \sum_{k=0}^{d} b_k x^k$$

Ainsi, si on souhaite les multiplier pour obtenir l'unique polynôme $K(x) = P(x)Q(x)$, l'algorithme classique est de procéder par distributivité jusqu'à obtenir la forme développée de K :

$$\begin{aligned} K(x) = &\quad (a_0 + a_1 x + ... + a_d x^d)(b_0 + b_1 x + ... + b_d x^d) &(1)\\ = &\quad a_0(b_0 + b_1 x + ... + b_d x^d) + a_1 x(b_0 + b_1 x + ... + b_d x^d) + ... + a_d x^d(b_0 + b_1 x + ... + b_d x^d) &(2)\end{aligned}$$

Ce que l'on constate c'est que pour effectuer ce calcul, il faut réaliser $(d+1)^2$ multiplications, car dans (2) on multiplie chacun des $d+1$ coefficients de P par chacun des $d+1$ coefficients de Q.
Enfin, toujours dans l'égalité (2), on effectue une somme de $d+1$ termes dans chacune des $d+1$ parenthèses, ce qui donne $d(d+1) + d = d(d+2)$ additions.
Donc, le calcul comporte un nombre total de $(d+1)^2 + d(d+2)$ opérations, ce qui donne une complexité en $O(d^2)$. En pratique, la complexité est davantage portée par le nombre de multiplications, les additions étant relativement peu coûteuses pour un ordinateur.

Question : Serait-ce possible de réduire cette complexité pour pouvoir avoir moins d'opérations ?

Pour cela, essayons de représenter les polynômes différemment.
Prenons l'exemple d'un polynôme $C(x) = c_0 + c_1 X$ de degré 1. Si on le représente dans le plan $\mathbb{R}^2$, le graphe de C est une droite, qui peut être entièrement caractérisée par deux points quelconques du graphe.
En effet, on dispose d'un axiome de géométrie euclidienne important, qui nous dit qu'il existe toujours une unique droite passant par deux points du plan.
Cette idée de représentation des polynômes dans le plan par un nombre fini de points se généralise à n'importe quel degré d, grâce au théorème suivant.

Théorème 3.1. *Tout polynôme de degré d peut être caractérisé dans le plan par au moins $d+1$ points de son graphe. Autrement dit, pour $d+1$ points quelconques du plan, il existe toujours un unique polynôme de degré d, dont le graphe passe par ces points.*

Preuve. Soit $\{(x_0, y_0); (x_1, y_1); ...; (x_d, y_d)\}$, un ensemble de $d+1$ points quelconques du plan. Alors, on cherche l'ensemble des polynômes $P(x) = a_0 + a_1 x + ... + a_d x^d$ de degré d vérifiant :

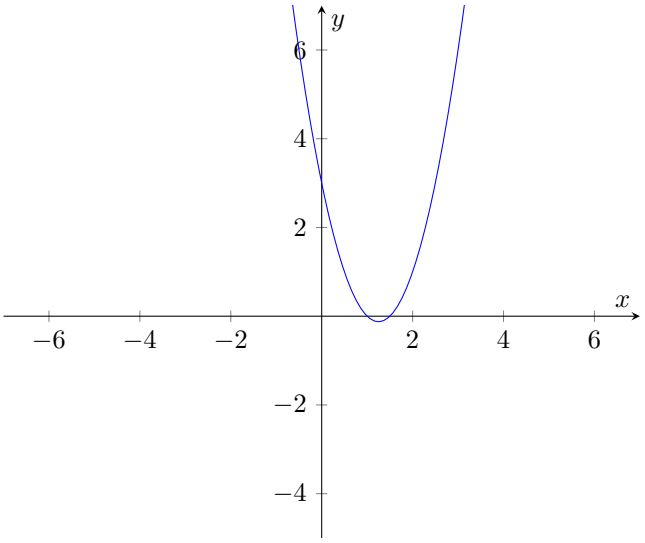
$$\begin{cases} y_0 = & a_0 + a_1 x_0 + ... + a_d x_0^d \\ y_1 = & a_0 + a_1 x_1 + ... + a_d x_1^d \\ & \vdots \\ y_d = & a_0 + a_1 x_d + ... + a_d x_d^d \end{cases} \iff \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_d \end{pmatrix} = \begin{pmatrix} 1 & x_0 & \dots & x_0^d \\ 1 & x_1 & \dots & x_1^d \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_d & \dots & x_d^d \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_d \end{pmatrix}$$

Or, comme on réalise une interpolation, on reconnait une matrice de Vandermonde, dont le déterminant est non nul ($\forall i, j \in [\![0, d]\!]$, $i \neq j$, $x_i \neq x_j$). On en déduit que le système est inversible, et possède donc une unique solution.
On en conclut qu'il existe un unique polynôme P de degré d qui convient. □

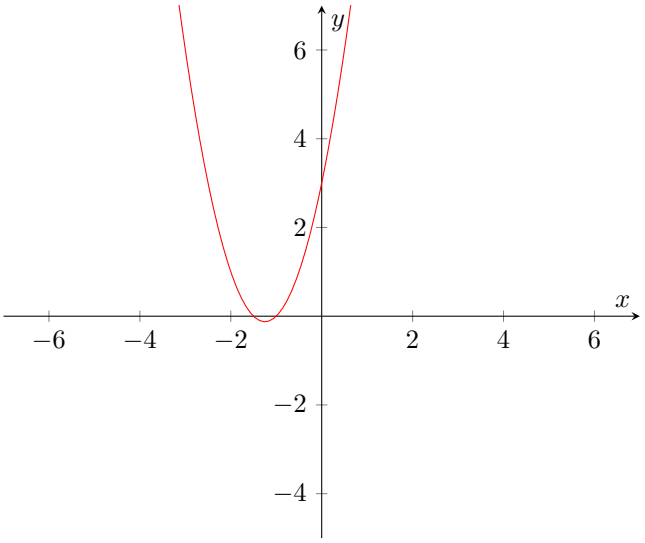
Ainsi, à l'issue de ce théorème, on s'assure que notre représentation des polynômes a un sens.
Regardons désormais ce qu'il se passe si l'on essaye de déterminer le produit de deux polynômes de même

degré sous cette représentation. Prenons à titre d'exemples les polynômes :

$P(x) = 2x^2 - 5x + 3$

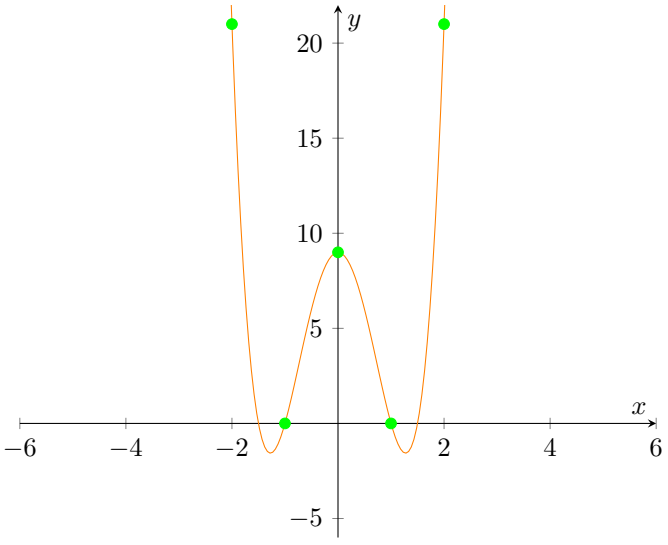


$Q(x) = 2x^2 + 5x + 3$



Le polynôme $K(x) = P(x)Q(x)$ sera alors de degré 4. Donc, par le théorème 3.1, K peut être caractérisé par 5 points du plan. Il est donc suffisant de prendre un échantillon de 5 points pour chaque graphe.
On prend par exemple pour P l'échantillon $\{(-2, 21); (-1, 10); (0, 3); (1, 0); (2, 1)\}$, et pour Q l'ensemble de points $\{(-2, 1); (-1, 0); (0, 3); (1, 10); (2, 21)\}$.
En découle alors, en multipliant deux à deux les images ayant même antécédent, l'ensemble de points $\{(-2, 21); (-1, 0); (0, 9); (1, 0); (2, 21)\}$ caractérisant K.

Graphe du polynôme K passant par ses cinq points caractéristiques



Remarque : Bien entendu, il existe une infinité d'ensembles à cinq points du plan qui caractérisent K.

En somme, sous cette autre représentation, nous sommes en mesure de retrouver un produit de polynôme en effectuant un calcul moins coûteux. En effet, pour obtenir les cinq points, nous avons effectué seulement 4 multiplications, dû au fait que le polynôme K produit a un degré deux fois plus important.
Donc, en général pour une multiplication de deux polynômes de degré d, on aurait réalisé d multiplications, ce qui traduit une complexité en $O(d)$.

Ainsi, on est parvenu à réduire la complexité de $O(d^2)$ à $O(d)$ avec cette représentation !
Note : Pour la suite de l'étude, on appellera cette représentation, la "représentation par points".

3.1.2 Évaluation d'un polynôme de degré d en $d+1$ valeurs

Si on considère un polynôme P de degré d, alors une autre manière de le représenter est sous forme d'un vecteur dont les coordonnées $(a_0, ..., a_d)$ sont les coefficients de P. Appellons cette représentation, la "représentation par coordonnées".

Question : Comment pourrait-on basculer aisément entre la représentation par points, et la représentation par coordonnées avec une complexité minimale ? Autrement dit, quel procédé nous permettrait de retrouver les coefficients d'un polynôme de degré d à partir de $d+1$ de ses points (et inversément), avec une faible charge de calculs ?

En effet, la question se pose car précédemment, nous avions pu déterminer la représentation par points du produit de deux polynômes d'un même degré, avec une complexité en $O(d)$.
Cependant, à partir d'un nombre fini de points, il n'est souvent pas évident de trouver l'unique polynôme qui convient sans alourdir les calculs.
De même, si à l'inverse on part d'un polynôme de degré d, et que l'on souhaite déterminer sa représentation par points, cela reviendrait à vouloir l'évaluer en au moins $d+1$ valeurs, d'après le théorème 3.1.
Ceci reviendrait donc à une complexité en $O(d^2)$, ce qui nous ramènerait au point de départ.

La réponse à cette question est un algorithme qui porte le nom de transformée de Fourier rapide (ou FFT pour *Fast Fourier Transform*).

L'idée de départ est la suivante. Existerait-il des polynômes particuliers que lorsqu'on les évalue en une valeur, on pourrait en déduire plusieurs points ?
Si je considère P un polynôme pair, et I un polynôme impair, et que je les évalue en une valeur x quelconque, alors on sait que :

$$P(x) = P(-x) \quad I(x) = -I(-x)$$

Donc, en évaluant seulement une fois en une valeur x quelconque, on arrive à déterminer :
$(-x, P(x)), (x, P(x)) \in \mathscr{C}_P$ et $(-x, -I(x)), (x, I(x)) \in \mathscr{C}_I$, soit deux points pour chaque polynôme P et I.
On en déduit que pour ce type de polynôme, si on souhaite un nombre N pair de points, alors il suffit d'évaluer en :

$$x_1, x_2, \dots, x_{\frac{N}{2}}$$

pour en déduire immédiatement les points évalués en $-x_1, -x_2, \dots, -x_{\frac{N}{2}}$ sans efforts.
L'idéal est donc de considérer des polynômes pairs et impairs.
Or, naturellement tout polynôme K de degré impair $N-1$ peut s'écrire :

$$K(x) = \sum_{n=0}^{N} k_n x^n = \underbrace{\sum_{n=0}^{\frac{N}{2}-1} k_{2n} x^{2n}}_{P(x)} + \underbrace{\sum_{n=0}^{\frac{N}{2}-1} k_{2n+1} x^{2n+1}}_{I(x)}$$

où P et I sont resp. pair et impair. On peut également exprimer K de la manière suivante :

$$K(x) = \sum_{n=0}^{\frac{N}{2}-1} k_{2n} x^{2n} + x \sum_{n=0}^{\frac{N}{2}-1} k_{2n+1} x^{2n} = P(x^2) + x\tilde{P}(x^2)$$

où P et $\tilde{P}$ sont cette fois-ci des polynômes pairs de degré $\frac{N}{2} - 1$!

Si maintenant on évalue K en N valeurs $\{\pm x_1, \pm x_2, \dots, \pm x_{\frac{N}{2}}\}$ ± appariées, on obtient :

$$\forall\, j \in [\![1, \frac{N}{2}]\!], \quad \begin{cases} K(x_j) & = P(x_j^2) + x_j \tilde{P}(x_j^2) \\ K(-x_j) & = P(x_j^2) - x_j \tilde{P}(x_j^2) \end{cases} \implies P(x_j^2) = \frac{K(x_j) + K(-x_j)}{2}$$

Donc l'évaluation du polynôme K en x_j et $-x_j$ se déduit directement à partir de l'évaluation de P en x_j^2.
Or, comme $P(x_j^2)$ et $\tilde{P}(x_j^2)$ sont tous deux de degré $\frac{N}{2}-1$, on se retrouve avec une évaluation à $\frac{N}{2}$ points qui sont : $\{x_1^2, x_2^2, ..., x_{\frac{N}{2}}^2\}$. On a donc réussi à diviser la complexité par 2 !

Cet algorithme semble très efficace, et on souhaiterait le réitérer sur les polynômes P et $\tilde{P}$.
Seulement, on rencontre un problème important.
En effet, la récursion repose sur le fait que les valeurs soient $\pm$ appariées à chaque itération.
Or, ici les $\frac{N}{2}$ valeurs x_j^2 pour P et $\tilde{P}$ sont toutes positives.
Par conséquent, elles ne sont pas $\pm$ appariées, ce qui empêche la réitération.

Remarque importante : Comme à chaque itération on souhaite diviser N par 2, il est nécessaire que N soit une k-ième puissance de 2, afin que l'on puisse effectuer jusqu'à k itérations ($\forall\, j \in [\![0,k]\!]$, $\frac{N}{2^k}$ doit être entier).

3.1.3 Évaluation sur le cercle unité

Il est clair que le seul moyen pour que les valeurs x_j^2 soient négatives (et donc potentiellement $\pm$ appariées), est qu'elles soient complexes. Cependant, il faudrait également s'assurer que ce soit le cas à chaque itération de l'algorithme.

Question : Quels sont donc les complexes qui conviendraient ?

On considère un polynôme P de degré d, que l'on souhaite évaluer en $N \geq d+1$ points : $\{\pm x_0, \pm x_1, ..., \pm x_{\frac{N}{2}-1}\}$ (où $N = 2^k$). Pour que la récursion fonctionne, on a vu qu'il fallait que les points $\{x_0^2, x_1^2, ..., x_{\frac{N}{2}-1}^2\}$ soient $\pm$ appariées de la façon suivante :

$$\begin{cases} x_1^2 & = -x_0^2 \\ x_3^2 & = -x_2^2 \\ & \vdots \\ x_{\frac{N}{2}-1}^2 & = -x_{\frac{N}{2}-2}^2 \end{cases}$$

Maintenant, si on fait le choix de poser $x_1 = 1$, on constate qu'à la k-ième itération il nous restera plus qu'un point : $x_0^{2^k} = x_0^N$ qui vaudra donc nécessairement 1. Par ailleurs, pour tout ensemble $\{x_0^{2^j}, x_1^{2^j}, ..., x_{\frac{N}{2^j}-1}\}$ à $\frac{N}{2^j}$ points ($j \in [\![1,k]\!]$) obtenu à la j-ième itération, on a :

$$\begin{cases} x_1^{2^j} & = -x_0^{2^j} \\ x_3^{2^j} & = -x_2^{2^j} \\ & \vdots \\ x_{\frac{N}{2^j}-1}^2 & = -x_{\frac{N}{2^j}-2}^2 \end{cases}$$

De proche en proche, en sachant que $x_0 = 1$, on arrive à voir avec un peu d'efforts, que le système a pour unique solution :

$$(\pm x_0, \pm x_1, ..., \pm x_{\frac{N}{2}-1}) = (\pm\omega_N^0, \pm\omega_N^1, ..., \pm\omega_N^{\frac{N}{2}-1}), \quad \omega_N = e^{\frac{2i\pi}{N}}$$

Comme tous les points sont distincts deux à deux et solutions de $x_0^N = 1$, on en déduit que les points complexes qui conviennent sont exactement les N racines $N^{ième}$ de l'unité.

3.1.4 FFT : l'algorithme de Cooley-Tukey (1965)

Donc d'après la sous partie précédente, on a pour intérêt d'évaluer le polynôme P en les racines $N^{ième}$ de l'unité, en posant $N = d + 1$, ce qui donne le système suivant :

$$\begin{cases} P(\omega_N^0) & = a_0 + a_1 + a_2 + ... + a_{N-1} \\ P(\omega_N) & = a_0 + a_1\omega_N + a_2\omega_N^2 + ... + a_{N-1}\omega_N^{N-1} \\ P(\omega_N^2) & = a_0 + a_1\omega_N^2 + a_2\omega_N^4 + ... + a_{N-1}\omega_N^{2(N-1)} \\ & \vdots \\ P(\omega_N^{N-1}) & = a_0 + a_1\omega_N^{N-1} + a_2\omega_N^{2(N-1)} + ... + a_{N-1}\omega_N^{(N-1)^2} \end{cases}$$

$$\iff \begin{pmatrix} P(\omega_N^0) \\ P(\omega_N) \\ P(\omega_N^2) \\ \vdots \\ P(\omega_N^{N-1}) \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N & \omega_N^2 & \dots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \dots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_N^{-(N-1)} & \omega_N^{-2(N-1)} & \dots & \omega_N^{-(N-1)^2} \end{pmatrix}}_{\mathcal{M}(\mathscr{F}_N)} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{N-1} \end{pmatrix}$$

On reconnait alors la matrice Vandermonde-Fourier de la transformée de Fourier discrète à l'ordre N !
Ainsi, la transformée de Fourier rapide se sert de la transformée de Fourier discrète pour évaluer un polynôme en N points complexes (les racines $N^{ième}$ de l'unité).
Cela signifie que l'on est en mesure de réutiliser, en supplément, toutes les propriétés apportées par la DFT ! En particulier, nous avions notamment étudié dans la partie 2.2, les propriétés de symétrie et d'inversibilité de la DFT.

Donc par ailleurs, si l'on s'intéresse à effectuer le processus inverse de l'évaluation par la FFT, qui est l'interpolation par P, la tâche devient plus simple qu'il n'y parait. En effet, la matrice V de la DFT est facilement inversible à tout ordre N et son inverse vaut $\frac{1}{N}\overline{V}$ (cf. théorème 2.6), ce qui traduit un gain de performances considérable, ainsi qu'un allègement supplémentaire dans le code de la FFT !

Puis, si nous revenons aux polynômes $K, P, \tilde{P}$ définis dans la sous partie 3.1.2, nous avions finalement déterminé la relation fondamentale qui caractérise l'algorithme de la FFT.
Désormais, si on évalue en les racines $N^{ième}$ de l'unité ($x_j = \omega_N^j$), on a :

$$\forall\, j \in [\![0, \frac{N}{2} - 1]\!], \quad \begin{cases} K(\omega_N^j) & = P(\omega_N^{2j}) + \omega_N^j \tilde{P}(\omega_N^{2j}) \\ K(-\omega_N^j) & = P(\omega_N^{2j}) - \omega_N^j \tilde{P}(\omega_N^{2j}) \end{cases}$$

Or, $-\omega_N^j = e^{i\pi} e^{\frac{2ij\pi}{N}} = e^{\frac{2i\pi}{N}(j + \frac{N}{2})} = \omega_N^{j+\frac{N}{2}}$. Donc, en remplaçant on obtient :

$$\forall\, j \in [\![0, \frac{N}{2} - 1]\!], \quad \begin{cases} K(\omega_N^j) & = P(\omega_N^{2j}) + \omega_N^j \tilde{P}(\omega_N^{2j}) \\ K(\omega_N^{j+\frac{N}{2}}) & = P(\omega_N^{2j}) - \omega_N^j \tilde{P}(\omega_N^{2j}) \end{cases}$$

Enfin, si on considère les vecteurs :

$$u = (u_0, u_1, ..., u_{\frac{N}{2}-1}) = (P(\omega_N^0), P(\omega_N^2), \dots, P(\omega_N^{N-2})) \quad v = (v_0, v_1, ..., v_{\frac{N}{2}-1}) = (\tilde{P}(\omega_N^0), \tilde{P}(\omega_N^2), \dots, \tilde{P}(\omega_N^{N-2}))$$

on peut écrire :

$$\boxed{\forall\, j \in [\![0, \frac{N}{2} - 1]\!], \quad \begin{cases} K(\omega_N^j) & = u_j + \omega_N^j v_j \\ K(\omega_N^{j+\frac{N}{2}}) & = u_j - \omega_N^j v_j \end{cases}}$$

Remarque : Les ω_N sont également appelés *facteurs rotatifs* (ou *twiddle factors*).
L'algorithme de *James Cooley* et de *John Tukey* voit le jour en 1965, et s'inscrit comme l'un des algorithmes les plus prodigieux, pour ses performances absolument remarquables en calculs numériques et du fait qu'il soit facile à inverser et à coder. Cependant, d'après l'article wikipédia, cet algorithme aurait déja été inventé par le mathématicien *Carl Friedrich Gauss* en 1805, qui l'aurait utilisé afin d'interpoler les trajectoires de deux astéroïdes du nom de *Pallas* et *Juno*.

Théorème 3.2. *Soit N une puissance de 2. Soient $X = (x_0, x_1, \dots x_{N-1}), X' = (X_0, X_1, \dots X_{N-1}) \in \mathbb{C}^N$, tels que $X' = \mathscr{F}_N(X)$.*
Alors :

$$\forall\, k \in [\![0, N-1]\!], \quad X_k = \left(\mathscr{F}_{\frac{N}{2}}(X)\right)_{2k} + \omega_N^{-k}\left(\mathscr{F}_{\frac{N}{2}}(X)\right)_{2k+1}$$

Il s'agit de l'algorithme de Cooley-Tukey, qui traduit une complexité en $O(Nlog(N))$.

Preuve. Par la définition 2.2, on a :

$$X_k = \sum_{n=0}^{N-1} x_n \omega_N^{-nk}$$

Donc, comme N est une puissance de 2, on peut séparer la somme de sorte à regrouper les termes pairs et impairs :

$$\begin{aligned} X_k &= \sum_{n=0}^{\frac{N}{2}-1} x_{2n}\omega_N^{-2nk} + \sum_{n=0}^{\frac{N}{2}-1} x_{2n+1}\omega_N^{-2(n+1)k} \\ &= \sum_{n=0}^{\frac{N}{2}-1} x_{2n}\omega_N^{-2nk} + \omega_N^{-k}\sum_{n=0}^{\frac{N}{2}-1} x_{2n+1}\omega_N^{-(2n+1)k} \end{aligned}$$

On reconnaît donc aisément les transformées de Fourier discrètes d'ordre $\frac{N}{2}$ appliquées aux termes pairs et impairs, d'où la formule.
Si on pose $p \in \mathbb{N}$ tel que $N = 2^p$, alors l'algorithme aura p niveaux de récursion. Comme on effectue $\frac{N}{2} + \frac{N}{2} = N$ opérations pour chaque récursion, la complexité de l'algorithme est en $O(Np)$.
Or, comme $p = log_2(N) = log_2(10)log(N)$, on en déduit que la complexité est en $O(Nlog(N))$. □

L'algorithme de Cooley-Tukey arrive donc à réduire un problème d'interpolation ou d'évaluation à N points, qui est de l'ordre de N^2, à une complexité d'ordre $Nlog(N)$.
On en conclut que la transformée de Fourier rapide est bien au delà de l'efficacité de la méthode basique !

Proposition 3.3. *Si on considère deux signaux réels $X = (x_0, \dots, x_{N-1}), Y = (y_0, \dots, y_{N-1})$, alors en posant le signal complexe $W = X + iY = (w_0, \dots, w_{N-1})$ regroupant les deux, on a :*

$$X_k = \frac{W_k + \overline{W_{-k}}}{2} \quad Y_k = \frac{W_k - \overline{W_{-k}}}{2i}$$

où $X' = (X_0, \dots, X_{N-1}), Y' = (Y_0, \dots, Y_{N-1}), W' = (W_0, \dots, W_{N-1})$ sont les DFT de X, Y et W.

Preuve. Il suffit d'appliquer le point (ii) du lemme 2.17 pour $\overline{W_{-k}}$:

$$\overline{W_{-k}} = \sum_{n=0}^{N-1} (\overline{w_n})\omega_N^{-kn} = \sum_{n=0}^{N-1} (\overline{x_n + iy_n})\omega_N^{-kn} = \sum_{n=0}^{N-1} (x_n - iy_n)\omega_N^{-kn}$$

car les signaux sont réels.
Donc, par linéarité, on en déduit aisément :

$$\frac{W_k + \overline{W_{-k}}}{2} = \sum_{n=0}^{N-1} x_n\omega_N^{-kn} = X_k \quad \frac{W_k - \overline{W_{-k}}}{2i} = \sum_{n=0}^{N-1} y_n\omega_N^{-kn} = Y_k \quad \square$$

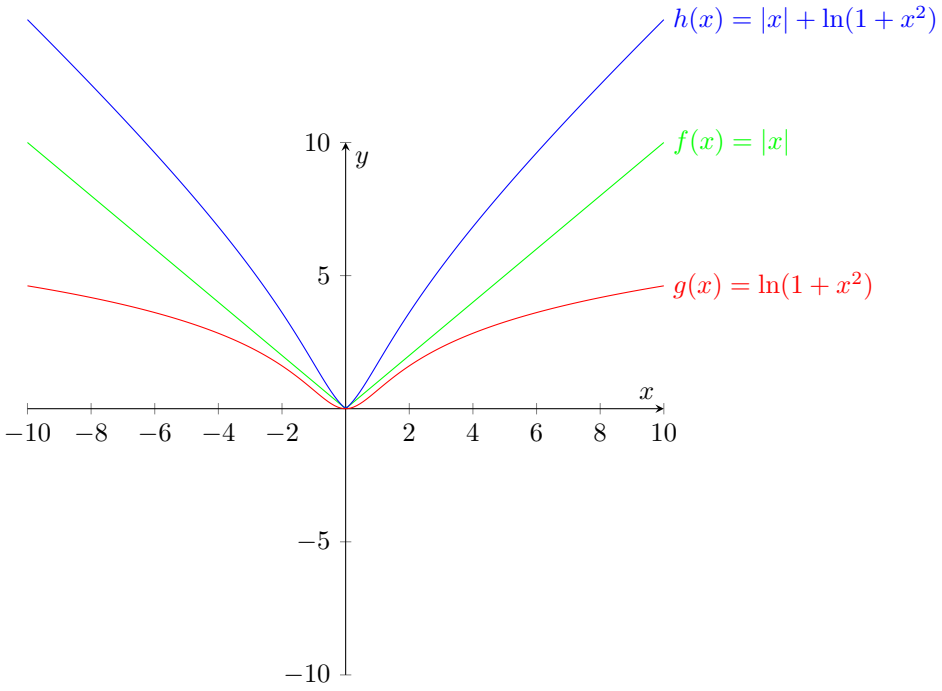
Donc dans ce cas particulier où l'on traite des signaux réels, on est en mesure de diviser la charge de calcul par 2, et ainsi obtenir une complexité en $O(\frac{N}{2})$. En effet, en calculant la transformée de Fourier discrète du signal W, on en déduit facilement celles des signaux réels X et Y.
Remarque : Les signaux discrets X' et Y' ne sont pas nécessairement réels !

3.2 FFT appliquée à la convolution circulaire

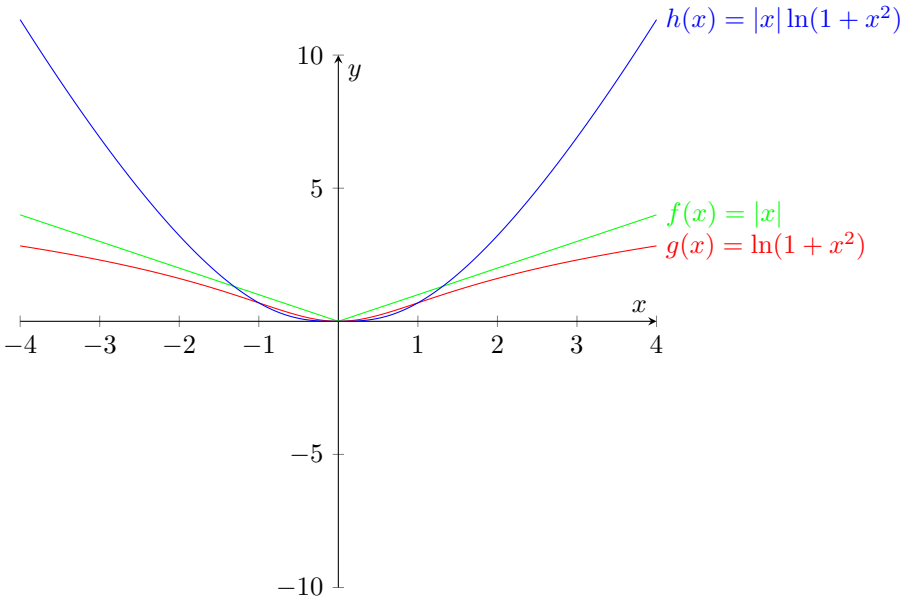
3.2.1 Produit de convolution

Prenons f et g, deux signaux analogiques (dont le graphe est continue), et définis sur $\mathbb{R}$.
Il existe plusieurs manières de considérer une combinaison de ces deux signaux pour obtenir un signal h.

- Sommer les deux signaux f et g :



- Multiplier les deux signaux f et g :



Cependant, il en existe une qui est fondamentale pour l'analyse de Fourier : le produit de convolution.

Définition 3.4. (Espaces de Lebesgue)
On dit qu'un signal analogique f défini sur un intervalle Ω est dans $L^p(\Omega)$ (avec $p \geq 1$ entier), si :

$$\int_\Omega |f|^p < +\infty$$

C'est un espace de Banach dont la norme est :

$$\|f\|_p = \left(\int_\Omega |f|^p \right)^{1/p}$$

Définition 3.5. Soient f et g deux signaux dans $L^1(\mathbb{R})$. Alors on appelle produit de convolution de f et g, le signal h défini par :

$$\forall\, x \in \mathbb{R}, \quad h(x) = (f * g)(x) = \int_{-\infty}^{+\infty} f(x-t)g(t)\, dt$$

Le produit de convolution est donc une opération entre deux fonctions, et son opérateur est noté "$*$".

Théorème 3.6. *En supposant f et g dans $L^1(\mathbb{R})$, leur convolée $f * g$ est bien définie grâce au contrôle de sa norme L^1 :*

$$\|f * g\|_1 \leq \|f\|_1 \|g\|_1$$

Preuve. En effet, par inégalité triangulaire et en posant le changement de variable $y = x - t$ on a :

$$\begin{aligned} \forall\, x \in \mathbb{R}, \quad \|f * g\|_1 &\leq \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} |f(x-t)||g(t)|\, dt\, dx \\ &= \left(\int_{-\infty}^{+\infty} |g(t)|\, dt \right) \left(\int_{-\infty}^{+\infty} |f(t)|\, dy \right) \\ &= \|f\|_1 \|g\|_1 \\ &< +\infty \end{aligned}$$

car f et g sont dans $L^1(\mathbb{R})$, ce qui prouve l'inégalité des normes L^1. Enfin, comme la norme L^1 de la convolée $f * g$ est finie, on en déduit que :

$$\forall\, x \in \mathbb{R}, \quad (f * g)(x) = \int_{-\infty}^{+\infty} f(x-t)g(t)\, dt < +\infty$$

ce qui prouve l'existence de $f * g$. $\square$

Proposition 3.7. *Le produit de convolution est un opérateur bilinéaire, associatif et commutatif, c'est à dire :*

$$(i) \quad \forall\, \lambda \in \mathbb{R}, \quad (\lambda f + g) * h = \lambda(f * h) + g * h \qquad f * (\lambda g + h) = \lambda(f * g) + f * h$$

$$(ii) \quad (f * g) * h = f * (g * h)$$

$$(iii) \quad f * g = g * f$$

Preuve. (i) Par linéarité de l'intégrale :

- $$\begin{aligned} [(\lambda f + g) * h](x) &= \int_{-\infty}^{+\infty} [(\lambda f + g)(x-t)]\, h(t) = \lambda \int_{-\infty}^{+\infty} f(x-t)h(t)\, dt + \int_{-\infty}^{+\infty} g(x-t)h(t)\, dt \\ &= \lambda(f * h)(x) + (g * h)(x) \end{aligned}$$
- $$\begin{aligned} [f * (\lambda g + h)](x) &= \int_{-\infty}^{+\infty} f(t)\, [(\lambda g + h)(x-t)] = \lambda \int_{-\infty}^{+\infty} g(x-t)f(t)\, dt + \int_{-\infty}^{+\infty} h(x-t)f(t)\, dt \\ &= \lambda(g * f)(x) + (h * f)(x) \\ &= \lambda(f * g)(x) + (f * h)(x) \end{aligned}$$

par commutativité du produit de convolution (prouvée au point (iii)).

$$(ii) \quad [(f*g)*h](x) = \int_{-\infty}^{+\infty} [(f*g)(x-u)]\, h(u)\, du$$
$$= \int_{-\infty}^{+\infty} \left(\int_{-\infty}^{+\infty} f(x-(u+t))g(t)\, dt \right) h(u)\, du$$

On pose le changement de variable $y = u + t$:

$$[(f*g)*h](x) = \int_{-\infty}^{+\infty} \left(\int_{-\infty}^{+\infty} f(x-y)g(y-u)\, dy \right) h(u)\, du$$
$$= \int_{-\infty}^{+\infty} f(x-y) \left(\int_{-\infty}^{+\infty} g(y-u)\, h(u)\, du \right) dy$$
$$= [f*(g*h)](x)$$

(iii) On pose le changement de variable $u = x - t$:

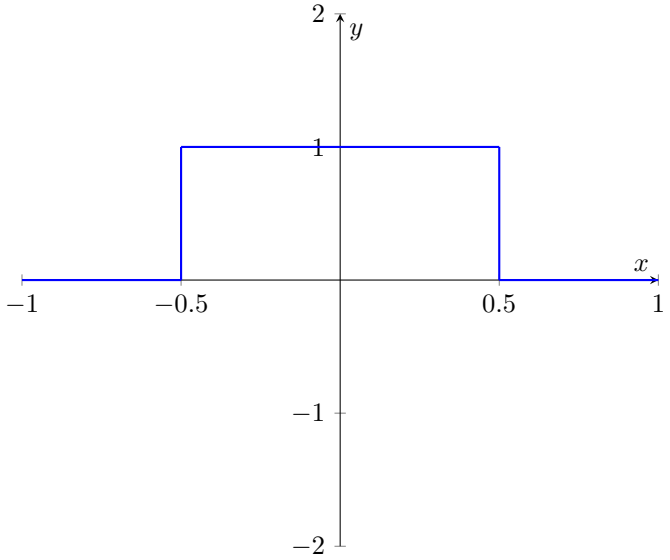
$$(f*g)(x) = \int_{-\infty}^{+\infty} f(x-t)g(t)\, dt = -\int_{+\infty}^{-\infty} f(u)g(x-u)\, du = \int_{-\infty}^{+\infty} g(x-u)f(u)\, du = (g*f)(x) \quad \square$$

Ainsi défini, le produit de convolution traduit concrètement ce que l'on pourrait qualifier de "moyenne glissante". Pour le comprendre, on l'illustre au moyen d'un exemple.

On s'intéresse à la "fonction porte" Π définie par :

$$\forall\, x \in \mathbb{R}, \quad \Pi(x) = \begin{cases} 1 & x \in [-\frac{1}{2}, \frac{1}{2}] \\ 0 & x \notin [-\frac{1}{2}, \frac{1}{2}] \end{cases}$$

Graphe de la fonction porte Π



Remarque : En particulier, il s'agit de la fonction indicatrice sur l'intervalle $[-\frac{1}{2}, \frac{1}{2}]$.

Pour comprendre comment fonctionne le produit de convolution, déterminons le signal Π convolué à lui-même, soit $\Pi * \Pi$.
Tout d'abord, il est évident que $\Pi \in L^1(\mathbb{R})$ car :

$$\int_{-\infty}^{+\infty} |\Pi(x)|\, dx = \int_{-\frac{1}{2}}^{\frac{1}{2}} |\Pi(x)|\, dx = \int_{-\frac{1}{2}}^{\frac{1}{2}} dx = 1 < +\infty$$

ce qui assure l'existence du signal $\Pi * \Pi$ par le théorème 3.6. Par ailleurs, on a :

$$\forall\, x \in \mathbb{R}, \quad (\Pi * \Pi)(x) = \int_{-\infty}^{+\infty} \Pi(x-t)\Pi(t)\, dt = \int_{-\frac{1}{2}}^{\frac{1}{2}} \Pi(x-t)\Pi(t)\, dt = \int_{-\frac{1}{2}}^{\frac{1}{2}} \Pi(x-t)\, dt$$

car $\forall\, t \in [-\frac{1}{2}, \frac{1}{2}]$, $\Pi(t) = 1$. On peut effectuer le changement de variable judicieux $u = t - x$, ce qui donne :

$$(\Pi * \Pi)(x) = \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} \Pi(u)\, du$$

Ainsi, on en déduit que $\Pi * \Pi$ est nul, si et seulement si :

$$\begin{aligned}\left[x - \frac{1}{2}, x + \frac{1}{2}\right] \cap \left[-\frac{1}{2}, \frac{1}{2}\right] = \emptyset &\iff x - \frac{1}{2} \geq \frac{1}{2} \quad ou \quad x + \frac{1}{2} \leq -\frac{1}{2} \\ &\iff x \in\,]-\infty, -1] \cup [1, +\infty[\end{aligned}$$

Traitons les cas où $\left[x - \frac{1}{2}, x + \frac{1}{2}\right] \cap \left[-\frac{1}{2}, \frac{1}{2}\right] \neq \emptyset$:

- $\underline{1^{er} \text{ cas :}}$ $\begin{cases} x - \frac{1}{2} \leq -\frac{1}{2} \\ -\frac{1}{2} \leq x + \frac{1}{2} \leq \frac{1}{2} \end{cases} \iff -1 \leq x \leq 0$

$$(\Pi * \Pi)(x) = \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} \Pi(u)\, du = \int_{-\frac{1}{2}}^{x+\frac{1}{2}} du = x + 1$$

- $\underline{2^{ème} \text{ cas :}}$ $\begin{cases} -\frac{1}{2} \leq x - \frac{1}{2} \leq \frac{1}{2} \\ \frac{1}{2} \leq x + \frac{1}{2} \end{cases} \iff 0 \leq x \leq 1$

$$(\Pi * \Pi)(x) = \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} \Pi(u)\, du = \int_{x-\frac{1}{2}}^{\frac{1}{2}} du = -x + 1$$

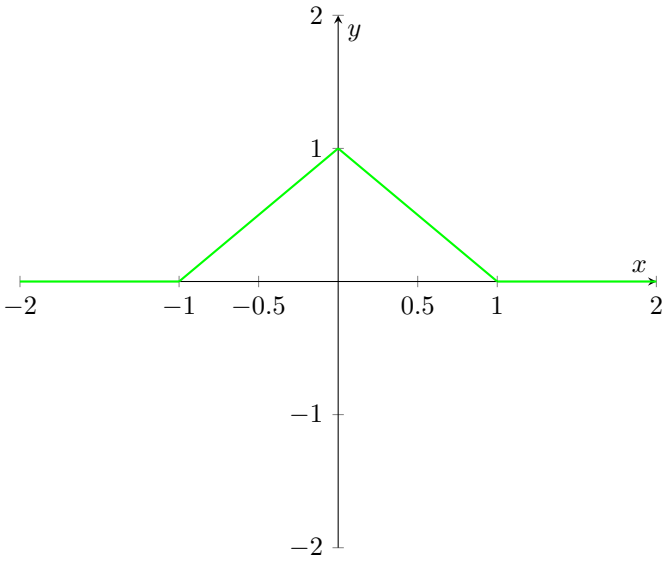
- $\underline{3^{ème} \text{ cas :}}$ $x = 0$

$$(\Pi * \Pi)(0) = \int_{0-\frac{1}{2}}^{0+\frac{1}{2}} \Pi(u)\, du = \int_{-\frac{1}{2}}^{\frac{1}{2}} du = 1$$

Donc, après calcul, la convolution de la fonction porte par elle-même est définie par :

$$\forall\, x \in \mathbb{R}, \quad (\Pi * \Pi)(x) = \begin{cases} 0 & x \leq -1 \\ x + 1 & -1 \leq x \leq 0 \\ -x + 1 & 0 \leq x \leq 1 \\ 0 & 1 \leq x \end{cases}$$

Graphe de la fonction $\Pi * \Pi$

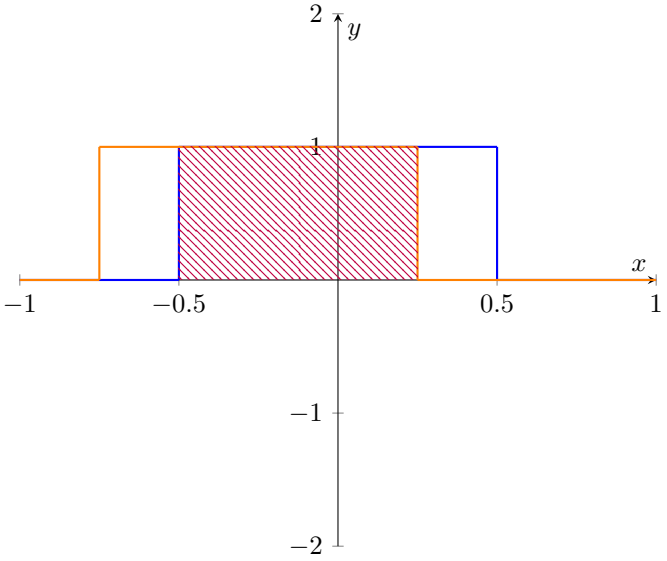


Interprétation et phénomène graphique :

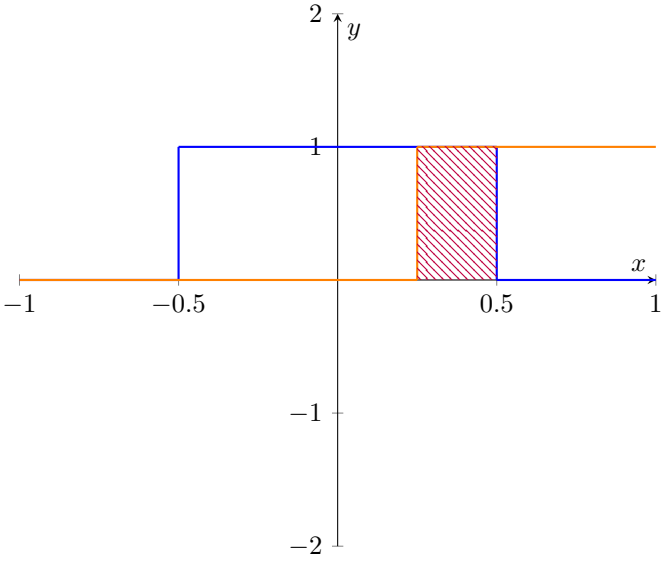
Cependant, dans cet exemple, le produit de convolution peut s'obtenir de manière graphique.
En effet, le produit de convolution $f * g$ consiste à translater (ou "faire glisser") le graphe de f le long de l'axe des abscisses (d'où l'intervalle d'intégration $]-\infty, +\infty[$), tandis que le graphe de g est fixé.
À chaque déplacement du graphe de f sur l'axe Ox, le produit de convolution renvoie l'aire qui coïncide avec le graphe de f et de g, exprimée en fonction de x. Ici, on a pris le cas particulier : $f = g = \Pi$.
Afin d'effectuer le déplacement du graphe de f le long de l'axe des abscisses, on réalise une paramétrisation par x de l'intervalle $[-\frac{1}{2}, \frac{1}{2}]$.

Pour effectuer ce déplacement, on considère donc f comme la fonction porte sur l'intervalle $[x - \frac{1}{2}, x + \frac{1}{2}]$, avec $x \in \mathbb{R}$, tandis que g est la fonction porte sur $[-\frac{1}{2}, \frac{1}{2}]$.

A - Graphe de f centré en $x = -\frac{1}{4}$



B - Graphe de f centré en $x = \frac{3}{4}$



Note : f et g sont respectivement représentées par la courbe en orange, et la courbe en bleue.

Dans le graphe A, l'aire recherchée est hachurée en rouge, et comme $x = -\frac{1}{4}$, elle vaut $(\Pi * \Pi)(-\frac{1}{4})$.
De même pour le graphe B, on évalue $(\Pi * \Pi)(\frac{3}{4})$ comme étant l'aire hachurée en rouge.
En effet, d'après les précédents résultats sur la fonction $\Pi * \Pi$:

$$\mathscr{A}_A = (0.25 - (-0.5)) \times 1 = \frac{3}{4} = (\Pi * \Pi)\left(-\frac{1}{4}\right) \qquad \mathscr{A}_B = (0.5 - 0.25) \times 1 = \frac{1}{4} = (\Pi * \Pi)\left(\frac{3}{4}\right)$$

En poursuivant la même disjonction de cas que pour le premier calcul, en calculant l'aire des rectangles hachurés en fonction de x, on retrouve aisément la même définition pour $(\Pi * \Pi)$, soit à nouveau :

$$\forall\, x \in \mathbb{R}, \quad (\Pi * \Pi)(x) = \begin{cases} 0 & x \leq -1 \\ x + 1 & -1 \leq x \leq 0 \\ -x + 1 & 0 \leq x \leq 1 \\ 0 & 1 \leq x \end{cases}$$

Remarques :

- Cette interprétation graphique est valable pour tout produit de convolution entre deux signaux f et g (dont le produit de convolution est défini). Nous avons pris l'exemple $f = g = \Pi$, car il nous permet de mieux visualiser l'aspect graphique. En général, les produits de convolution sont souvent difficiles à calculer par le biais de la méthode graphique !
- En reprenant la définition 3.5, on reconnaît que la fonction évaluée en $x - t$ dans l'intégrale, est celle dont le graphe est translaté. Or, comme le produit de convolution est commutatif (cf. proposition 3.7), on est libre de choisir quel graphe l'on soit souhaite translater : le résultat sera identique.
- Le produit de convolution est très utile et a un lien profond avec la transformée de Fourier continue $\mathscr{F}$ définie par :

  $$\mathscr{F}(f)(x) = \int_{-\infty}^{+\infty} f(t) e^{-2i\pi x t}\, dt$$

  dont l'ensemble de définition $\mathscr{D}_{\mathscr{F}}$ contient $L^1(\mathbb{R})$. En effet :

  $$\mathscr{F}(f * g)(x) = \mathscr{F}(f)(x)\mathscr{F}(g)(x)$$

  La transformée de Fourier continue transforme le produit de convolution en un produit de deux fonctions. Comme la transformée de Fourier continue n'est pas l'objet direct de cette étude, on se contentera de la preuve du théorème 3.11 qui constitue l'équivalent dans le cas discret.

3.2.2 Cas discret : convolution circulaire

Dans le cas précédent, nous venons de voir que le produit de convolution était défini pour des signaux continus.
Ici, on souhaiterait étendre la notion de convolution pour des signaux discrets (dans l'intérêt de notre étude sur la transformée de Fourier discrète). Pour cela, on ne considère plus des fonctions mais des échantillons de valeurs de même taille.

Définition 3.8. Soient $A = (a_0, \dots a_{N-1})$ et $B = (b_0, \dots b_{N-1})$ deux échantillons de taille N. Le produit de convolution circulaire $A * B$ est défini par le signal discret $C = (c_0, \dots c_{2N-2})$, où :

$$\forall\, n \in [\![0, 2N-2]\!], \quad c_n = \sum_{i=0}^{N-1} a_i b_{n-i}$$

Remarques :

- Les signaux discrets étudiés sont des échantillons de signaux continus et T-périodiques.
  Comme l'échantillonnage est régulier (rappel : les valeurs sont espacées de $\frac{N}{T}$), N est à la fois la taille de l'échantillon mais aussi la période du signal discret considéré.
  En effet, en prenant $X = (x_0, \dots, x_{N-1})$ un tel signal, on a :

$$\forall\,(k,q) \in [\![0, N-1]\!] \times \mathbb{Z}, \quad x_k = x_{k+qN}$$

  d'où la terminologie "circulaire" pour le produit de convolution (égalité modulo N).
  On parle aussi de produit de convolution périodique.
- Le produit de Cauchy de deux séries est un produit de convolution discret !
  En effet, en considérant les séries $\sum_{n\geq 0} a_n$ et $\sum_{n\geq 0} b_n$, on a :

$$\left(\sum_{n=0}^{+\infty} a_n\right)\left(\sum_{n=0}^{+\infty} b_n\right) = \sum_{n=0}^{+\infty} c_n\,, \quad c_n = \sum_{k=0}^{n} a_k b_{n-k}$$

Proposition 3.9. *Le produit de convolution circulaire vérifie les propriétés de la proposition 3.7.*

Preuve. Il s'agit de la même preuve que pour la proposition 3.7, en sachant que la somme et l'intégrale vérifient globalement les mêmes propriétés, à savoir la linéarité par exemple. Les changements de variables sont également les mêmes, et s'appliquent de la même façon. □

Afin de mieux illustrer la définition 3.8, nous allons reprendre l'exemple de la très bonne vidéo, produite par *3Blue1Brown* sur le produit de convolution.

On se place dans une situation de probabilités, où l'on lance deux dés à six faces. Afin de généraliser, on n'exclut pas la possibilité que les dés soient truqués. Ainsi, on définit $A = (a_0, \dots, a_5)$ et $B = (b_0, \dots, b_5)$ les ensembles des probabilités resp. de chaque dé, tels que a_k et b_k sont resp. les probabilités d'obtenir le chiffre $k+1$ sur le dé A et le dé B ($k \in [\![0,5]\!]$).
Posons les événements : $A_k = \{a_k\}$ ("obtenir le chiffre $k+1$ avec le dé A") et $B_k = \{b_k\}$ ("obtenir le chiffre $k+1$ avec le dé B").
Alors, comme il s'agit d'un lancé de dés, les événements A_i et B_j sont indépendants ($\forall\, i,j \in [\![0,5]\!]$).

Donc :

$$\forall\, i,j \in [\![0,5]\!]\,(i \neq j), \quad \mathcal{P}(A_i \cap B_j) = \mathcal{P}(A_i) \times \mathcal{P}(B_j) = a_i b_j$$

Par ailleurs, $P(A_i \cap B_j)$ traduit la probabilité d'avoir le nombre $i+j+2$, en ayant eu le chiffre $i+1$ avec le dé A, et le chiffre $j+1$ avec le dé B. Donc si, pour $n \in [\![0,10]\!]$, on note $\mathcal{P}(n)$, la probabilité d'obtenir le nombre $n+2$, alors :

$$\boxed{\mathcal{P}(n) = \sum_{\substack{i,j \\ i+j=n}} \mathcal{P}(A_i \cap B_j) = \sum_{\substack{i,j \\ i+j=n}} a_i b_j = \sum_{i=0}^{5} a_i b_{n-i}}$$

On retrouve alors la relation de la définition 3.8 avec $N = 6$ et $c_n = \mathcal{P}(n)$.

3.2.3 Optimisation par transformée de Fourier rapide

On se place à nouveau dans le cadre d'un produit de deux polynômes P et Q de degré n, définis par :

$$P(x) = \sum_{i=0}^{n} a_i x^i \quad Q(x) = \sum_{j=0}^{n} b_j x^j$$

Si on pose $K(x) = P(x)Q(x)$, on observe une relation entre ses coefficients et un produit de convolution circulaire bien choisi.

Théorème 3.10. *Soient $A = (a_0, \dots, a_n)$, $B = (b_0, \dots, b_n)$.*
Alors :

$$K(x) = \sum_{k=0}^{2n} (A * B)_k x^k = \sum_{k=0}^{2n} \left(\sum_{i=0}^{n} a_i b_{k-i} \right) x^k$$

Preuve.

$$\begin{aligned} K(x) &= P(x)Q(x) \\ &= \left(\sum_{i=0}^{n} a_i x^i \right) \left(\sum_{j=0}^{n} b_j x^j \right) \\ &= \sum_{j=0}^{n} \left(\sum_{i=0}^{n} a_i x^i \right) b_j x^j \\ &= \sum_{0 \le i,j \le 2n} a_i b_j x^{i+j} \end{aligned}$$

Soient $(c_0, \dots, c_{2n})$, les coefficients de K. On souhaite déterminer le coefficient c_k pour $k \in [\![0, 2n]\!]$.
Comme ce dernier est un terme en x^k, cela implique donc que :

$$c_k x^k = \sum_{\substack{i,j \\ i+j=k}} a_i b_j x^{i+j} = \sum_{i=0}^{n} a_i b_{k-i} x^k = \left(\sum_{i=0}^{n} a_i b_{k-i} \right) x^k = (A * B)_k x^k \quad \square$$

Donc ce théorème nous dit qu'un produit de convolution circulaire revient à multiplier deux polynômes de même degré, et en récupérer les coefficients !
Exemple : Soient $f(x) = 5x^2 - 3$ et $g(x) = -2x^2 + x + 7$ deux fonctions polynômiales, dont on souhaiterait calculer $f * g$. Il est clair qu'utiliser la formule :

$$(f * g)(x) = \int_{-\infty}^{+\infty} f(x - t) g(t)\, dt$$

va nous mener à un calcul fastidieux.
Or, on peut basculer aisément au produit de convolution circulaire, car les échantillons $(-3, 0, 5)$ et $(7, 1, -2)$ contiennent l'entièreté des informations des signaux continus f et g, en plus d'être de même taille.
Donc, par la définition 3.8, on a :

$$(-3, 0, 5) * (7, 1, -2) = (-3 \times 7, -3 \times 1 + 0 \times 7, 3 \times 2 + 0 \times 1 + 5 \times 7, 0 \times 2 + 5 \times 1 - 5 \times 2) = (-21, -3, 41, 5, -10)$$

Donc, d'après le théorème 3.10, $h(x) = f(x)g(x) = -21 - 3x + 41x^2 + 5x^3 - 10x^4$.
De plus, on en déduit que $f * g = h$.

Théorème 3.11. *Soient $\mathscr{F}_N(A) = (X_0, \dots, X_{N-1}), \mathscr{F}_N(B) = (Y_0, \dots, Y_{N-1})$ les transformées de Fourier discrètes des deux signaux discrets A et B, de taille N. Soit $\mathscr{F}_N(A * B) = (U_0, \dots U_{2N-2})$.*
Alors :

$$\forall\, k \in [\![0, 2N-1]\!], \quad U_k = X_k Y_k$$

Note : Comme les signaux A et B sont N-périodiques (cf. remarque - définition 3.8), alors $\mathscr{F}_N(A)$ et $\mathscr{F}_N(B)$ sont également N-périodiques, ce qui donne un sens à X_k et Y_k pour $k \ge N$.

Preuve. Soient $A = (x_0, \dots, x_{N-1}), B = (y_0, \dots, y_{N-1})$, et $A * B = (u_0, \dots, u_{2N-2})$.
Par la définition 3.8, on a :

$$\forall\, n \in [\![0, 2N-2]\!], \quad u_n = \sum_{i=0}^{N-1} x_i y_{n-i}$$

Donc :

$$\forall\, k \in [\![0, 2N-2]\!], \quad U_k = \sum_{n=0}^{N-1} u_n \omega_N^{-kn} = \sum_{n=0}^{N-1} \left(\sum_{i=0}^{N-1} x_i y_{n-i} \right) \omega_N^{-kn}$$

D'autre part :

$$\forall\, k \in [\![0, 2N-2]\!], \quad X_k Y_k = \left(\sum_{l=0}^{N-1} x_l \omega_N^{-kl}\right)\left(\sum_{p=0}^{N-1} y_p \omega_N^{-kp}\right)$$

Il suffit de considérer ce produit comme un produit de deux polynômes en la variable $t = \omega_N^{-k}$, ce qui par le théorème 3.10 nous donne :

$$X_k Y_k = \sum_{n=0}^{N-1} u_n t^n = \sum_{n=0}^{N-1} \left(\sum_{i=0}^{N-1} x_i y_{n-i}\right) \omega_N^{-kn} = U_k \quad \square$$

But : Optimiser les calculs du produit de convolution

Nous venons de voir que le cas discret du produit de convolution s'étend vers le cas continu, lorsque l'on considère que pour une taille d'échantillon N assez grande, on ait suffisamment d'informations sur les signaux continus f et g. En particulier, si f et g sont polynômiales, $N > \max(\deg(f), \deg(g))$ convient.
Par ailleurs, si $\deg(f) \neq \deg(g)$, alors on considérera les coefficients nuls au delà du degré minimal.

Or, on peut s'apercevoir, d'après la définition 3.8, que le produit de convolution circulaire est un calcul en $O(N^2)$. Cependant, on vient de voir grâce au théorème 3.10, qu'il s'agit d'un produit de deux polynômes de degré N.
Donc, on est en mesure d'utiliser l'algorithme de Cooley-Tukey en s'assurant que N soit une puissance de 2 !

Conclusion : Le signal obtenu comme le produit de convolution continu de deux signaux analogiques f et g peut être approché, avec une charge de calcul optimale par la transformée de Fourier rapide.
En effet, si on les échantillonne en un nombre suffisamment grand (une puissance de 2), alors l'algorithme de Cooley-Tukey permet de retrouver une bonne partie des informations contenues dans le signal $f * g$, avec une complexité en $O(N \log(N))$ (cf. théorème 3.2).

4 Analyse spectrale des signaux par transformée de Fourier rapide

La transformée de Fourier rapide, se basant sur la transformée de Fourier discrète, peut s'implémenter facilement sur *Python* de part la simplicité de son code. Cependant, pour les tests applicatifs sur *Python*, nous allons utiliser le package *numpy.fft* qui contient déja un code optimisé avec l'ajout de la fonction *fft*(). Pour pouvoir étudier la transformée de Fourier d'un signal, il est en effet indispensable de passer par l'usage de la transformée de Fourier discrète et de la transformée de Fourier rapide, d'où leur grande importance. Afin de pouvoir réaliser les calculs sur ordinateur, il est nécessaire de passer par le cas discret d'où à nouveau l'intérêt de l'algorithme de la FFT qui en plus réduit considérablement les temps de traitement.

Note : Tous les signaux utilisés lors des expériences (fichiers audio, images) proviennent de données personnelles. Les fichiers audio seront par ailleurs au format *.wav* (sans compression) afin de négliger aucune information présente dans le signal d'origine.

4.1 Analyse fréquentielle d'un signal unidimensionnel

Objectif : Comprendre le rôle de la transformée de Fourier d'un signal.

4.1.1 Exemple d'un signal périodique

Considérons le signal $f(t) = 5\cos(3t) - 7\sin^2(2t)$, qui est 2π-périodique et continue par morceaux.
Afin de pouvoir étudier sa transformée de Fourier, on procède d'abord par un échantillonnage régulier en $N = 40$ points :

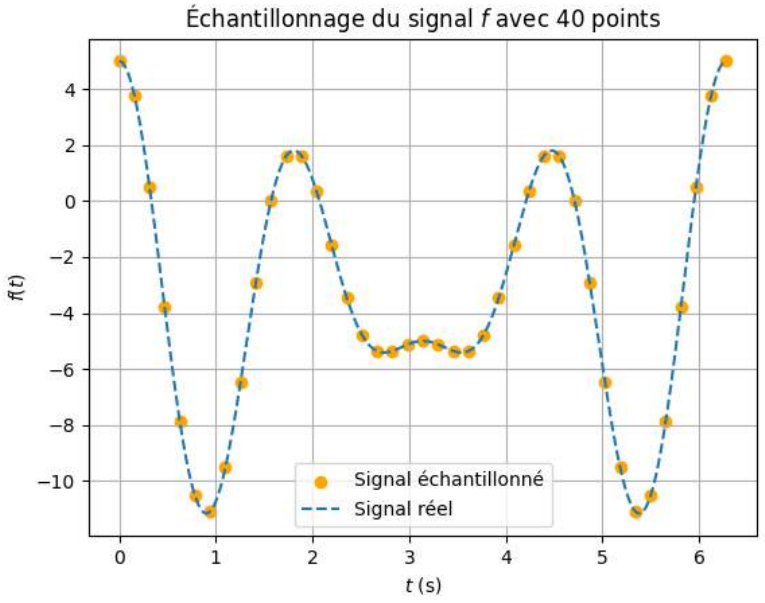


FIGURE 3 – Sur le graphe, il y a 41 points. Or, par périodicité du signal f, le dernier point vaut le premier ce qui nous ramène à un échantillonnage à 40 points.

En sachant que la période d'échantillonnage T_e et la fréquence d'échantillonnage F_e du signal sont :

$$T_e = \frac{2\pi}{40} = \frac{\pi}{20} \quad F_e = \frac{1}{T_e} = \frac{20}{\pi}$$

on applique l'algorithme de la transformée de Fourier rapide en prenant en compte ces deux valeurs.
Par ailleurs, la fréquence d'échantillonnage F_e respecte le critère de Shannon (cf. théorème 4.1), ce qui est

indispensable si l'on souhaite éviter un sous-échantillonnage, et donc une perte importante d'informations sur le signal.
Après application de l'algorithme de la FFT sur un vecteur X contenant l'échantillon du signal, on obtient un nouveau signal $\hat{f}$ continue par morceaux, construit à partir des fréquences renvoyées par la DFT :

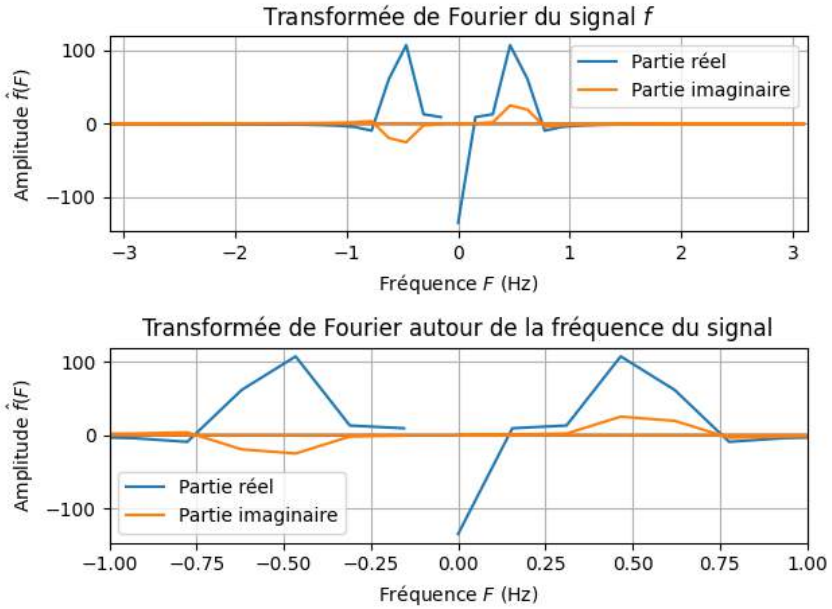


FIGURE 4 – Signal $\hat{f}$ obtenu par FFT

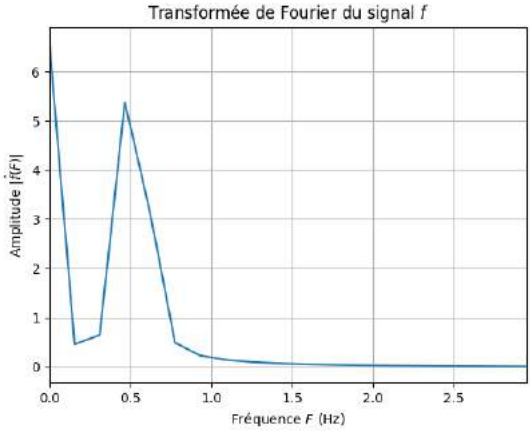


FIGURE 5 – Graphe du signal $|\hat{f}|$ centrée sur la plage utile

On constate sans surprise que $\hat{f}$ est un signal complexe (ou une fonction holomorphe) d'où l'intérêt de représenter séparément la partie réelle et imaginaire.
Également, on s'aperçoit que la partie réelle et imaginaire du signal $\hat{f}$ sont caractérisées essentiellement par 2 fréquences proches de -0.5 Hz et de 0.5 Hz. Par ailleurs, on remarque que le graphe de la partie réelle est quasiment symétrique par rapport à l'axe des ordonnées, et que la partie "positive" du graphe de la partie imaginaire est obtenu par rotation d'angle π et de centre $(0, 0)$, de la partie négative du graphe.
Cela est dû évidemment au fait que l'on ait :

$$\forall\, n \in \mathbb{N}, \quad c_{-n}(f) = \overline{c_n(f)}$$

où $c_n(f)$ est le coefficient de Fourier de f associé à la fréquence n.
Cependant, on souhaiterait étudier un seul signal afin de mieux étudier $\hat{f}$.
Ainsi pour mieux mettre en forme le résultat obtenu, on a décidé de représenter le module de $\hat{f}$ (voir figure 5).

On conclut que le signal obtenu est entièrement caractérisé par deux fréquences dominantes, proches quasiment de 0 Hz et de 0.5 Hz, sachant que les autres coefficients de Fourier associés aux plus hautes fréquences sont quasi-nuls.
En effet, cela est dû à l'extrême régularité du signal prit en exemple : f est une combinaison de cosinus et de sinus, ce qui lui donne la particularité d'être une fonction lisse, donc de classe $\mathscr{C}^\infty$.
Par conséquent, la vitesse de convergence des coefficients de Fourier est maximale (cf preuve - théorème 2.16 en faisant tendre k vers $+\infty$), ce qui explique que les informations pertinentes du signal se situent très très près de 0 Hz.

Remarque : On en déduit que le signal f d'origine correspond plutôt à un son extrêmement grave.
Cependant, ce dernier serait inaudible car ses fréquences prépondérantes sont largement en dessous du spectre auditif humain qui est de 20 Hz - 20000 Hz. Un tel signal est appelé un infrason.

La transformée de Fourier est donc en mesure de nous livrer le spectre d'un signal analogique afin de mieux comprendre sa composition. En particulier, elle permet de voir si un son enregistré est plutôt grave ou aigu.

Code Python utilisé pour les tests :

```python
import numpy as np
from numpy.fft import fft, fftfreq
import matplotlib.pyplot as plt

def f(t):
    # Calcul du signal f(t) = 5cos(3t) - 7sin^2(2t)
    return 5*np.cos(3*t)-7*np.sin(2*t)**2


# Échantillonnage du signal
Durée = 2*np.pi  # Durée du signal en secondes
Te = 0.2/4*np.pi # Période d'échantillonnage en secondes

N = int(Durée/Te) + 1  # Nombre de points du signal échantillonné

te = np.linspace(0, Durée, N)  # Temps des échantillons
t = np.linspace(0, Durée, 2000)  # Temps pour le signal non échantillonné

f_e = f(te)  # Calcul de l'échantillonnage

# Tracé du signal
plt.scatter(te, f_e, color='orange', label="Signal échantillonné")
plt.plot(t, f(t), '--', label="Signal réel")
plt.grid()
plt.xlabel(r"$t$ (s)")
plt.ylabel(r"$f(t)$")
```

```
plt.title(r"Échantillonnage du signal $f$ avec 40 points")
plt.legend()
plt.show()

# Calcul FFT
X = fft(f_e)  # Transformée de fourier
frequence = fftfreq(f_e.size, d=Te)  # Fréquences de la transformée de Fourier

plt.subplot(2,1,1)
plt.plot(frequence, X.real, label="Partie réel")
plt.plot(frequence, X.imag, label="Partie imaginaire")
plt.grid()
plt.xlim(-3.13,3.13)
plt.legend()
plt.xlabel(r"Fréquence $F$ (Hz)")
plt.ylabel(r"Amplitude $\hat{f}(F)$")
plt.title("Transformée de Fourier du signal $f$")

plt.subplot(2,1,2)
plt.plot(frequence, X.real, label="Partie réel")
plt.plot(frequence, X.imag, label="Partie imaginaire")
plt.xlim(-1,1) # Limite autour de la fréquence du signal
plt.grid()
plt.legend()
plt.xlabel(r"Fréquence $F$ (Hz)")
plt.ylabel(r"Amplitude $\hat{f}(F)$")
plt.title("Transformée de Fourier autour de la fréquence du signal")
plt.tight_layout()
plt.show()

# On prend la valeur absolue de l'amplitude uniquement pour les fréquences positives
X_abs = np.abs(X[:N//2])
# Normalisation de l'amplitude
X_norm = X_abs*2.0/N
# On garde uniquement les fréquences positives
freq_pos = frequence[:N//2]

plt.plot(freq_pos, X_norm, label="Amplitude absolue")
plt.xlim(0, 2.95)  # On réduit la plage des fréquences à la zone utile
plt.grid()
plt.xlabel(r"Fréquence $F$ (Hz)")
plt.ylabel(r"Amplitude $|\hat{f}(F)|$")
plt.title("Transformée de Fourier du signal $f$")
plt.show()
```

4.1.2 Cas d'un fichier audio : signal non périodique

Dans le but de comprendre davantage l'utilité de la transformée de Fourier, on va considérer un signal non périodique.
Pour cela, on a réalisé deux enregistrements au piano :

- la note Sol joué sur la $4^{ième}$ octave
- l'accord Mi mineur, comprenant les trois notes Mi - Sol - Si jouées en même temps sur la $4^{ième}$ octave

dont on souhaiterait étudier le spectre.
On définit alors les signaux f et g comme étant ceux des deux enregistrements respectifs.

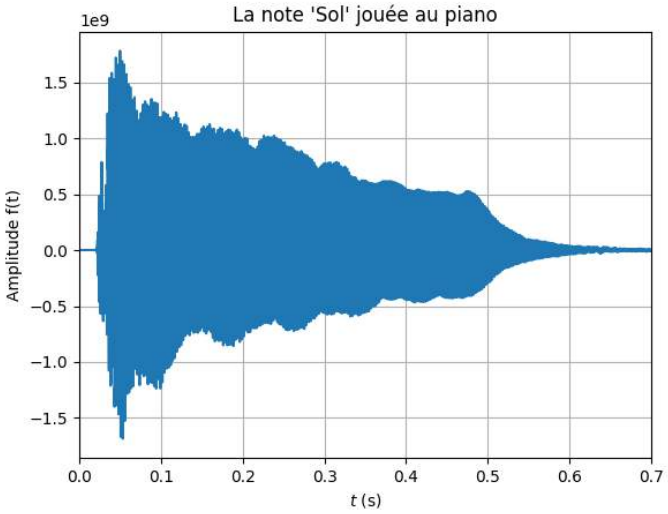


FIGURE 6 – Enregistrement du signal f (note Sol)

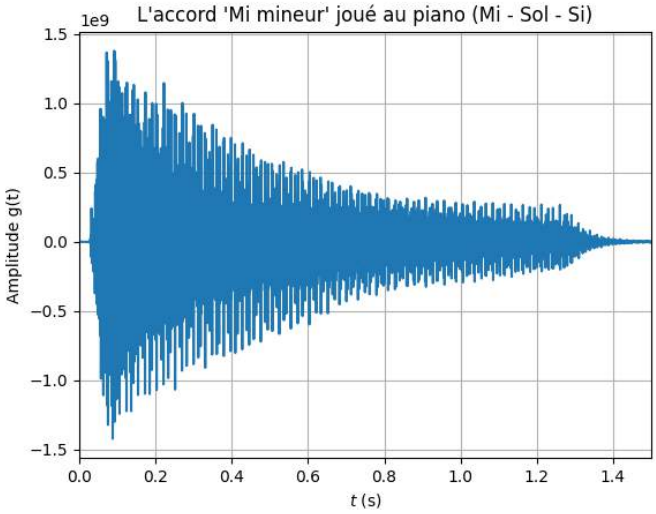


FIGURE 7 – Enregistrement du signal g (accord Mi mineur)

Ce sont donc tous deux des signaux réels et analogiques. Nonobstant, comme ce sont des enregistrements

audio de durée finie et que leur amplitude est majorée, on en déduit qu'ils sont dans $L^1(\mathbb{R})$ ce qui prouve l'existence de leur transformée de Fourier $\hat{f}$ et $\hat{g}$.
Or cependant, il est à noter que l'on connaît précisément la fréquence de ces notes sur la $4^{ième}$ octave :

- Mi3 : 330 Hz
- Sol3 : 392 Hz
- Si3 : 494 Hz

Après application de la FFT sur ces deux signaux, on obtient :

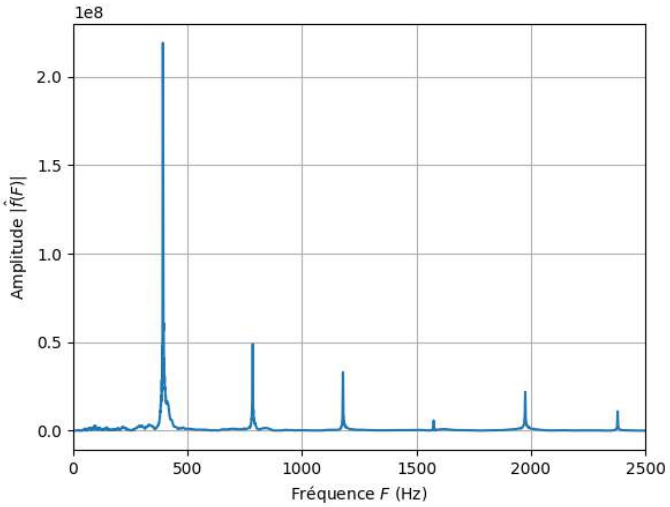


FIGURE 8 – Transformée de Fourier de f (note Sol)

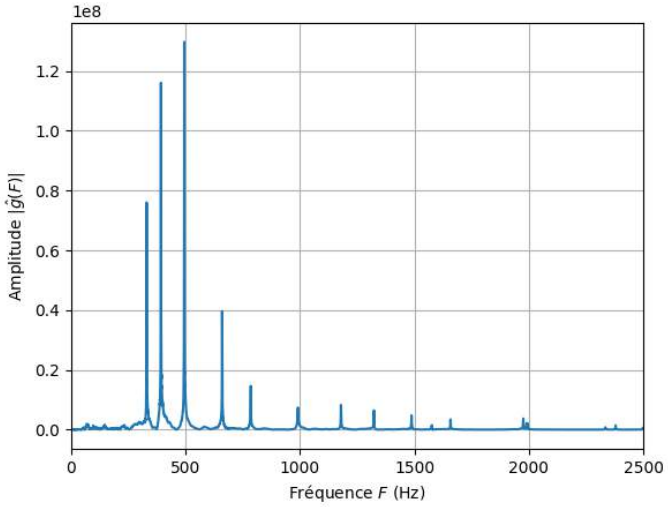


FIGURE 9 – Transformée de Fourier de g (accord Mi mineur)

On observe que dans chaque graphe, se trouvent des pics dominants représentant les fréquences qui caractérisent ces deux signaux.
Sur la figure 8, on observe un pic important en dessous de 500 Hz, qui correspond en réalité à une fréquence

de 392 Hz (visualisé sur Python) : on reconnaît alors la note Sol ayant été jouée ! (voir fréquences plus haut) Par ailleurs, cette fréquence est appelée la fondamentale tandis que les autres (plus faibles) visibles sont des harmoniques.
On observe le même phénomène dans le spectre de la figure 9, à l'exception du nombre de fréquences dominantes observées qui s'élève à trois. À nouveau, grâce aux fréquences citées plus haut, on arrive à reconnaître les trois notes de l'accord Mi mineur, à savoir : Mi, Sol et Si. Sachant que les notes sont rangées dans l'ordre croissant des fréquences, on pouvait deviner que le premier pic correspondrait à la note Mi, la deuxième à la note Sol, et enfin la dernière à la note Si (la plus aigu).

Remarques :

- On constate en comparant les deux spectres que les pics de même fréquence n'ont pas forcément la même amplitude. En réalité, elle ne fait que traduire l'intensité à laquelle les notes correspondantes ont été jouées. Seules les fréquences caractérisent les informations contenues dans le signal.
- Il ne peut y avoir plusieurs fondamentales dans un spectre, et en l'occurrence, la fréquence dominante la plus basse est toujours désignée comme la fondamentale. En effet, c'est pour cela qu'en harmonie musicale, on dit que la note Mi est la fondamentale de l'accord Mi mineur (non renversé).

Dans le but de mieux percevoir l'évolution du spectre d'un signal au cours du temps, on peut notamment réaliser un spectrogramme pour chaque fichier audio :

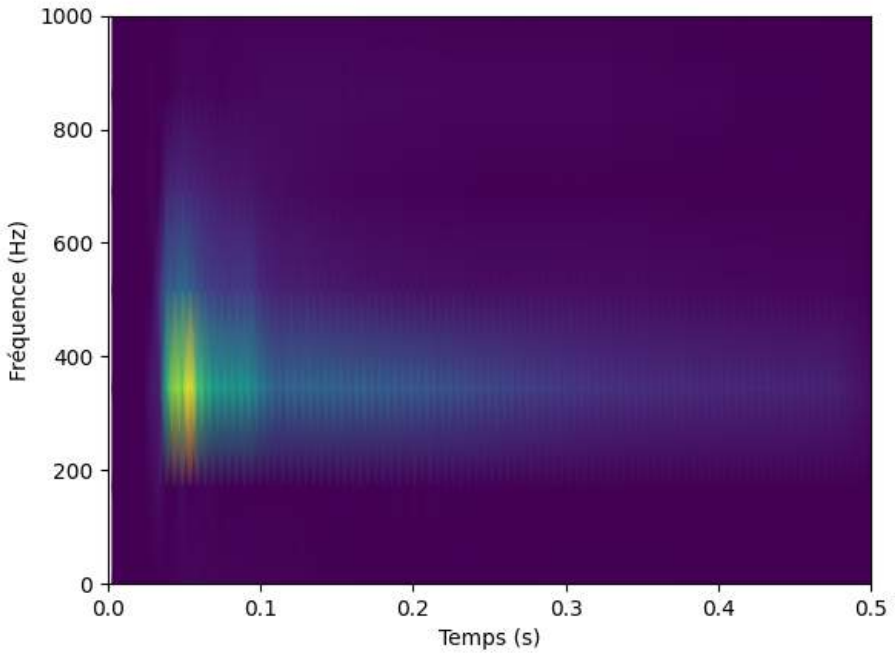


FIGURE 10 – Spectrogramme du signal g (note Sol)

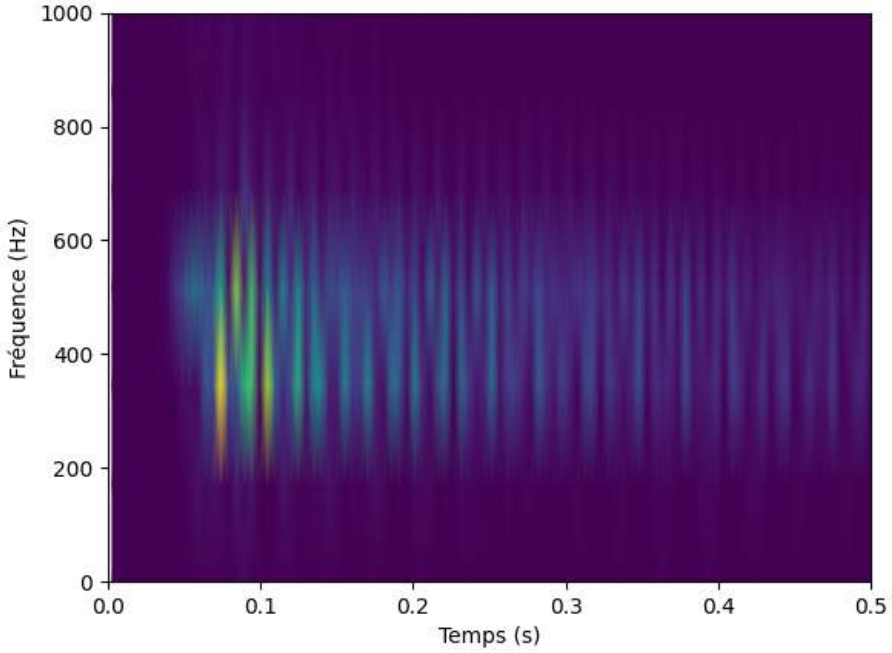


FIGURE 11 – Spectrogramme du signal g (accord Mi mineur)

Code Python utilisé pour les tests :

```
import scipy.io.wavfile as wavfile
import matplotlib.pyplot as plt
import numpy as np
from numpy.fft import fft, fftfreq
import scipy.signal as signal

# Lecture des deux fichiers
rate1, data1 = wavfile.read(r"C:\Users\Anwender\Desktop\Note Sol5.wav")
rate2, data2 = wavfile.read(r"C:\Users\Anwender\Desktop\Mi Mineur.wav")

# Sélection du canal 1
x1 = data1[:, 0] # fichier audio 1
x2 = data2[:, 0] # fichier audio 2

# Création de instants d'échantillons
t1 = np.linspace(0, data1.shape[0]/rate1, data1.shape[0]) # fichier audio 1
t2 = np.linspace(0, data2.shape[0]/rate2, data2.shape[0]) # fichier audio 2

# Tracé du signal

```

```
# fichier audio 1
plt.plot(t1, x1, label="Sol échantillonné")
plt.grid()
plt.xlim(0,0.7)  # fenêtre centrée sur l'intervalle [0,0.7]
plt.xlabel(r"$t$ (s)")
plt.ylabel(r"Amplitude f(t)")
plt.title(r"La note 'Sol' jouée au piano")
plt.show()

# fichier audio 2
plt.plot(t2, x2, label="Mi mineur échantillonné")
plt.grid()
plt.xlim(0,1.5)  # fenêtre centrée sur l'intervalle [0,1.5]
plt.xlabel(r"$t$ (s)")
plt.ylabel(r"Amplitude g(t)")
plt.title(r"L'accord 'Mi mineur' joué au piano (Mi - Sol - Si)")
plt.show()

# Calcul FFT
X = fft(x1)  # Transformée de fourier
freq = fftfreq(x1.size, d=1/rate1)  # Fréquences de la transformée de Fourier

# Calcul du nombre d'échantillon
N = x1.size

# On prend la valeur absolue de l'amplitude uniquement pour les fréquences positives
et normalisation
X_abs = np.abs(X[:N//2])*2.0/N
# On garde uniquement les fréquences positives
freq_pos = freq[:N//2]

plt.plot(freq_pos, X_abs, label="Amplitude absolue")
plt.xlim(0, 2500)  # On réduit la plage des fréquences à la zone utile
plt.grid()
plt.xlabel(r"Fréquence $F$ (Hz)")
plt.ylabel(r"Amplitude $|\hat{f}(F)|$")
plt.show()

# Calcul FFT
X = fft(x2)  # Transformée de fourier
freq = fftfreq(x2.size, d=1/rate2)  # Fréquences de la transformée de Fourier

# Calcul du nombre d'échantillon
N = x2.size

```

```
# On prend la valeur absolue de l'amplitude uniquement pour les fréquences positives
et on normalise
X_abs = np.abs(X[:N//2])*2.0/N
# On garde uniquement les fréquences positives
freq_pos = freq[:N//2]

plt.plot(freq_pos, X_abs, label="Amplitude absolue")
plt.xlim(0, 2500)  # On réduit la plage des fréquences à la zone utile
plt.grid()
plt.xlabel(r"Fréquence $F$ (Hz)")
plt.ylabel(r"Amplitude $|\hat{g}(F)|$")
plt.show()

# Spectrogramme de la note 'Sol'

x = data1[:, 0]  # Sélection du canal 1
# Création d'instants d'échantillons
# t = np.linspace(0, data1.shape[0]/rate1, data1.shape[0])

# Calcul du spectrogramme
f, t, Sxx = signal.spectrogram(x, rate1)
# On limite aux fréquences présentes
Sxx_red = Sxx[np.where(f<6000)]
f_red = f[np.where(f<6000)]

# Affichage du spectrogramme
plt.pcolormesh(t, f_red, Sxx_red, shading='gouraud')
plt.ylabel('Fréquence (Hz)')
plt.xlabel('Temps (s)')
plt.xlim(0,0.5)
plt.ylim(0,1000)
plt.show()

# Spectrogramme de l'accord 'Mi mineur'

x = data2[:, 0]

f, t, Sxx = signal.spectrogram(x, rate2)
Sxx_red = Sxx[np.where(f<6000)]
f_red = f[np.where(f<6000)]

plt.pcolormesh(t, f_red, Sxx_red, shading='gouraud')
plt.ylabel('Fréquence (Hz)')
plt.xlabel('Temps (s)')
plt.xlim(0,0.5)
```

```
plt.ylim(0,1000)
plt.show()
```

4.1.3 Conclusion

Le domaine de Fourier est un espace où la visualisation de signaux périodiques ou non périodiques devient immensément plus simple. En effet, on est mesure de comprendre quelles sont les informations importantes qui caractérisent le signal considéré. De plus, comme la transformée de Fourier rapide est inversible à tout ordre, il est possible de modifier ou de supprimer les fréquences indésirables (voir parasites) mises en évidence par la transformée de Fourier, pour ensuite retrouver un signal d'origine ayant subit ces mêmes modifications. Ce principe est notamment utilisé par les cartes son pour améliorer la qualité audio, mais aussi lors des compressions d'images (jpeg). Les fréquences parasites ou jugées inutiles sont retirées du signal par la transformée de Fourier afin d'alléger le volume de l'image sans pour autant sacrifier la qualité. Dans le cas d'une interface audio ou d'une technologie de réduction de bruit, les fréquences parasites sont visées puis retirées afin de clarifier et améliorer considérablement la qualité du son.

Fort heureusement, tout ceci s'étend en dimension 2 avec les images, ce qui constitue le sujet de la partie 4.3.

4.2 Théorie de l'échantillonnage et applications

Pour rappel, l'échantillonnage d'un signal analogique consiste à prélever un nombre fini de ses valeurs, appelées échantillons. Ces derniers caractérisent le signal et forment un signal discret. L'opération d'échantillonnage intervient notamment dans la numérisation d'un signal analogique (sons, enregistrements) et permet le passage entre le continu et le discret.

Objectif : Comprendre comment un signal continu peut être correctement restitué ou reconstruit par la transformée de Fourier discrète, et étudier les mauvais phénomènes pouvant intervenir.

4.2.1 Phénomène de Gibbs

Soit la fonction porte

$$\forall x \in \mathbb{R}, \quad \Pi(x) = \begin{cases} 1 & x \in [-\frac{1}{2}, \frac{1}{2}] \\ 0 & x \notin [-\frac{1}{2}, \frac{1}{2}] \end{cases}$$

que l'on décide de périodiser en un signal δ 2-périodique défini par :

$$\forall x \in [0, 2[, \quad \delta(x) = \begin{cases} 1 & x \in [0, \frac{1}{2}] \cup [\frac{3}{2}, 2[\\ 0 & x \in]\frac{1}{2}, \frac{3}{2}[\end{cases}$$

dont la série de Fourier est :

$$\forall t \in \mathbb{R}, \quad S_\delta(t) = \sum_{n=-\infty}^{+\infty} c_n(\delta) e^{i\pi nt}$$

où les $c_n(\delta)$ désignent les coefficients de Fourier de δ. Notons $S_N(\delta)$ la somme partielle d'ordre N de cette série telle que :

$$\forall t \in \mathbb{R}, \quad S_N(\delta)(t) = \sum_{n=-N}^{N} \delta(t) e^{i\pi nt}$$

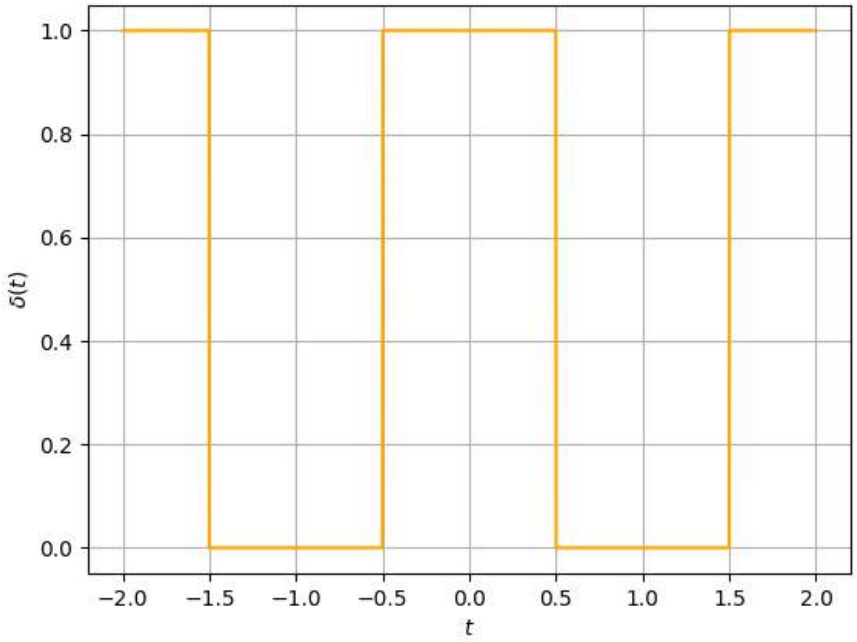


FIGURE 12 – Graphe de la fonction δ

En plus d'être périodique, le signal δ est continu par morceaux sur $[0, 2[$, mais présente cependant une discontinuité aux points $t = \frac{1}{2}$ et $t = \frac{3}{2}$. En effet, on a par exemple en $t = \frac{1}{2}$:

$$\delta(t) = 0 \xrightarrow[\substack{t \to \frac{1}{2} \\ t \neq \frac{1}{2}}]{} 0 \neq 1 = \delta\left(\frac{1}{2}\right)$$

ce qui prouve que δ n'est pas continu en ce point. En conséquence, le théorème de Dirichlet n'est pas applicable puisque le signal n'est pas $\mathscr{C}^1$ en ces points, et donc la convergence ponctuelle de sa série de Fourier en ces points de discontinuité reste ambiguë.
Il s'agit donc d'observer ce qu'il se passe aux "bords" de ces points.
Pour cela, on effectue un premier échantillonnage régulier du signal δ à $N = 20$ valeurs ce qui nous ramène à un calcul de 20 coefficients de Fourier c_n. Or, δ est un signal créneau dont les coefficients de Fourier peuvent être facilement calculés :

$$\begin{aligned} c_n(\delta) &= \frac{1}{2}\int_0^2 \delta(t) e^{-2i\pi n \frac{t}{2}}\, dt = \frac{1}{2}\int_0^2 \delta(t) e^{-i\pi n t}\, dt \\ &= \frac{1}{2}\int_0^{\frac{1}{2}} e^{-i\pi n t}\, dt + \frac{1}{2}\int_{\frac{3}{2}}^2 e^{-i\pi n t}\, dt \end{aligned}$$

- Si $n = 0$:

$$c_0(\delta) = \frac{1}{2}\left(\frac{1}{2} - 0 + 2 - \frac{3}{2}\right) = \frac{1}{2}$$

- Si $n \neq 0$:

$$\begin{aligned} c_n(\delta) &= -\frac{1}{2i\pi n}\left(\left[e^{-i\pi nt}\right]_0^{1/2} + \left[e^{-i\pi nt}\right]_{3/2}^{2}\right) \\ &= -\frac{1}{\pi n}\left(\frac{e^{-i\frac{\pi}{2}nt} - e^{-i\frac{3}{2}\pi nt}}{2i}\right) \\ &= \frac{1}{\pi n}\left(\frac{e^{i\frac{\pi}{2}nt} - e^{-i\frac{\pi}{2}nt}}{2i}\right) \\ &= \frac{\sin(\frac{\pi}{2}n)}{\pi n} \\ c_n(\delta) &= \frac{\operatorname{sinc}(n/2)}{2} \end{aligned}$$

On remarque que les coefficients de Fourier de δ associés aux fréquences $n \in \{2k\pi \mid k \neq 0\}$ sont tous nuls, en raison du facteur $\frac{1}{2}$ dans le sinus cardinal.

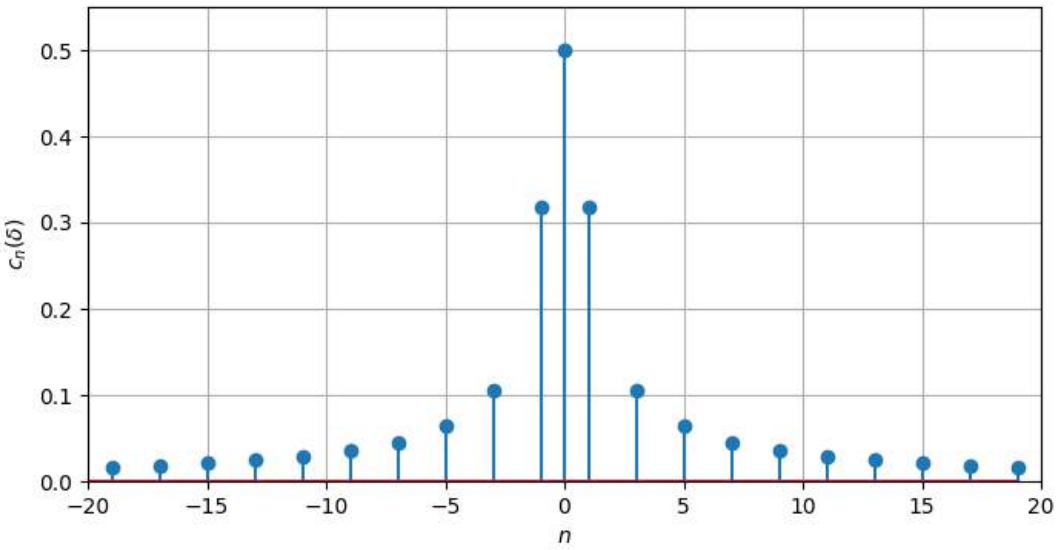


FIGURE 13 – Spectre du signal $S_{20}(\delta)$ construit à partir des coefficients de Fourier

Par ailleurs, on constate que le spectre est symétrique ce qui est évidemment dû au fait que :

$$\forall\, n \in \mathbb{N}, \quad c_{-n}(\delta) = \overline{c_n(\delta)} = c_n(\delta)$$

puisque les coefficients de Fourier de δ sont tous réels.
Donc, on en déduit la somme partielle de δ d'ordre N :

$$S_N(\delta)(t) = \sum_{n=-N}^{N} c_n(\delta) e^{i\pi nt}$$

$$S_N(\delta)(t) = \frac{1}{2} + \frac{1}{2} \sum_{\substack{n=-N \\ n\neq 0}}^{N} \operatorname{sinc}\left(\frac{n}{2}\right) e^{i\pi nt}$$

Puis enfin, par symétrie du spectre on obtient :

$$\boxed{\forall\, t \in \mathbb{R}, \quad S_N(\delta)(t) = \frac{1}{2} + \sum_{n=1}^{N} \operatorname{sinc}\left(\frac{n}{2}\right) e^{i\pi nt}}$$

Essayons tout d'abord de visualiser le signal δ reconstruit à partir de la somme partielle pour $N = 20$:

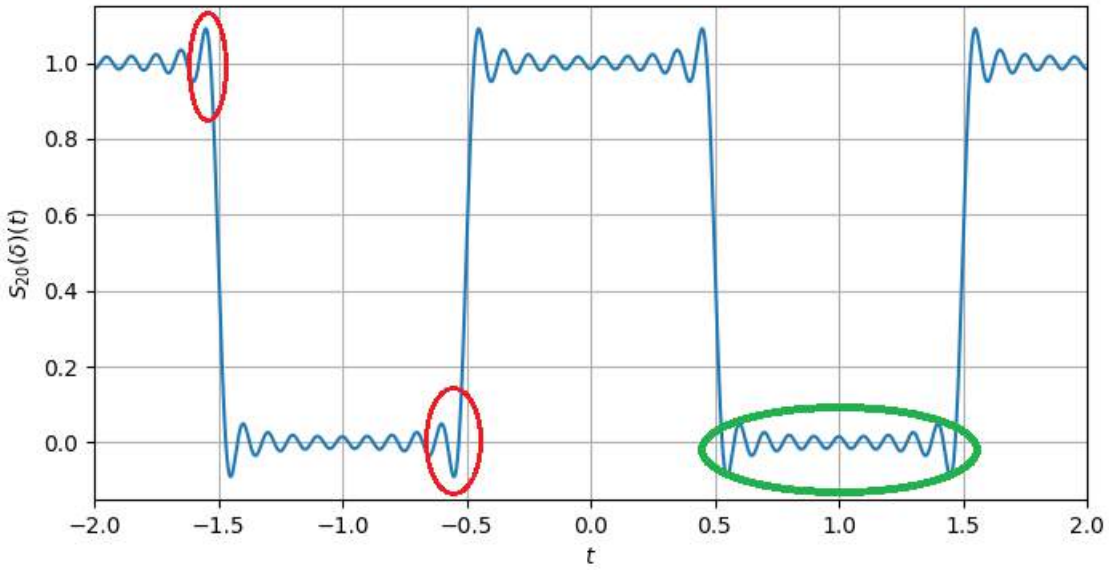


FIGURE 14 – Graphe de la somme partielle $S_{20}(\delta)$

On observe de fortes oscillations dans les zones semblables à celle entourée en vert. Ces oscillations s'intensifient au fur et à mesure que l'on se rapproche des points de discontinuité (zones entourées en rouge).
Il s'agit du phénomène de Gibbs.
Ce dernier traduit une vitesse de convergence médiocre des coefficients de Fourier $c_n(\delta)$ dû à la pauvre régularité du signal δ (cf. théorème 2.16), causée par les points discontinus.
Par conséquent, la plus grande fréquence n_{max} telle que $c_{n_{max}} \neq 0$ sera d'autant plus élevée à mesure que l'on considère des signaux peu réguliers.

Remarque : Dans la définition :

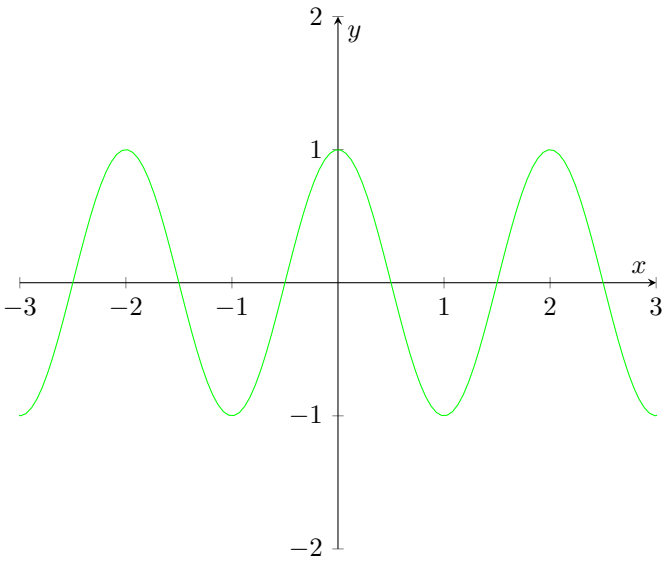
$$c_n = \frac{1}{T}\int_0^T f(t)e^{-2i\pi n\frac{t}{T}}\,dt$$

pour les coefficients de Fourier, n désigne alors une fréquence et la quantité $\omega = \frac{2\pi}{T}$ est la pulsation du signal. Donc la base de Fourier se constitue des fonctions de la forme :

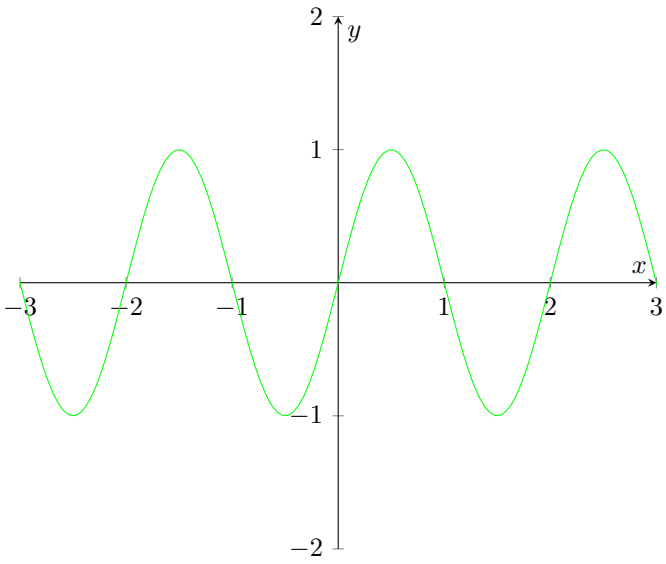
$$t \longmapsto \cos(n\omega t) \quad t \longmapsto \sin(n\omega t)$$

où $n \in \mathbb{N}$, et $\omega = \frac{2\pi}{T}$ désignant à nouveau la pulsation. Par ailleurs, ce sont des signaux $\frac{T}{n}$-périodiques pour tout $n \neq 0$ ou constants si $n = 0$. Chacun de ces signaux a donc une fréquence de $\frac{n}{T}$ où T est la période de f.
Pour mieux comprendre, on décide de reprendre le signal δ 2-périodique et de tracer le graphe de $\cos(n\pi t)$ et de $\sin(n\pi t)$ ($\omega = \pi$) pour des valeurs de n très différentes :

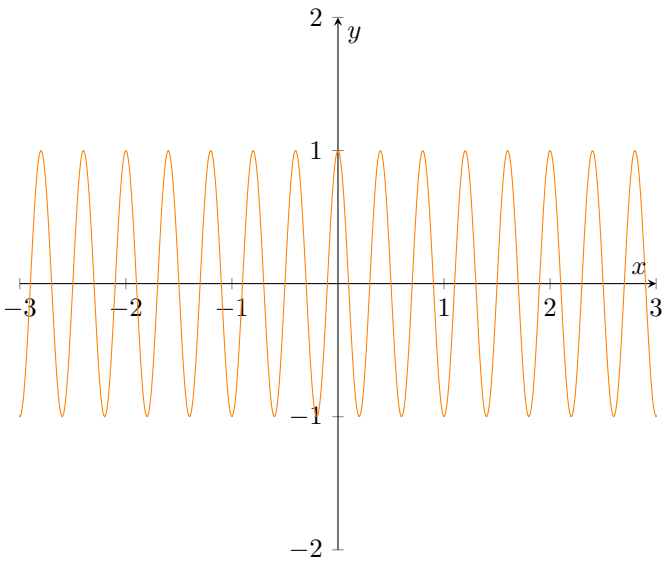
$x \longmapsto \cos(\pi x) \quad (n = 1)$



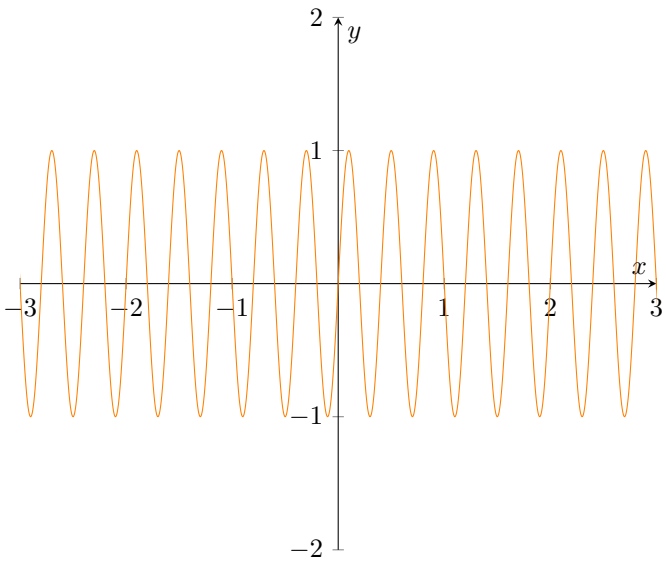
$x \longmapsto \sin(\pi x) \quad (n = 1)$

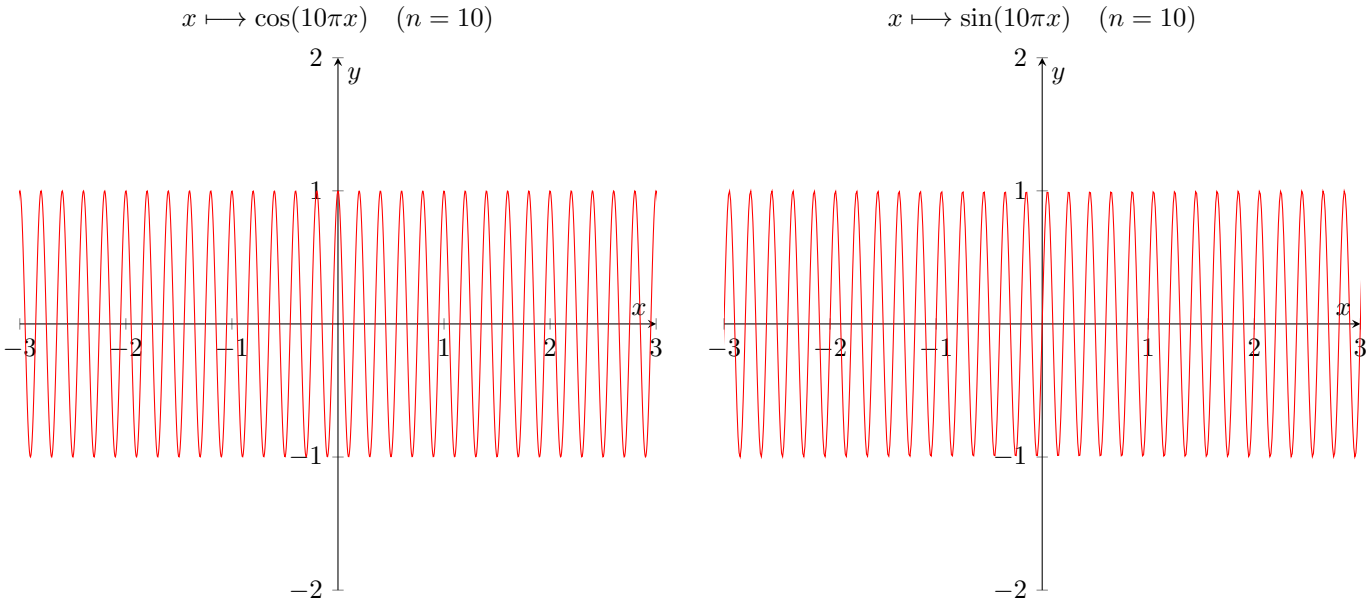


$x \longmapsto \cos(5\pi x) \quad (n = 5)$



$x \longmapsto \sin(5\pi x) \quad (n = 5)$





Donc, on en conclut que les zones de discontinuité d'un signal sont construits à partir de très hautes fréquences dont la valeur maximale dépend de la régularité de la fonction. Ainsi, pour la fonction δ par exemple (voir figure 14) les "chutes" brutales/rapides du graphe aux points de discontinuité sont restituées par des fréquences de plus en plus élevées à mesure que l'on se rapproche de ces points, ce qui explique le phénomène.
Enfin regardons l'allure de la somme partielle pour $N = 100$ et $N = 1000$:

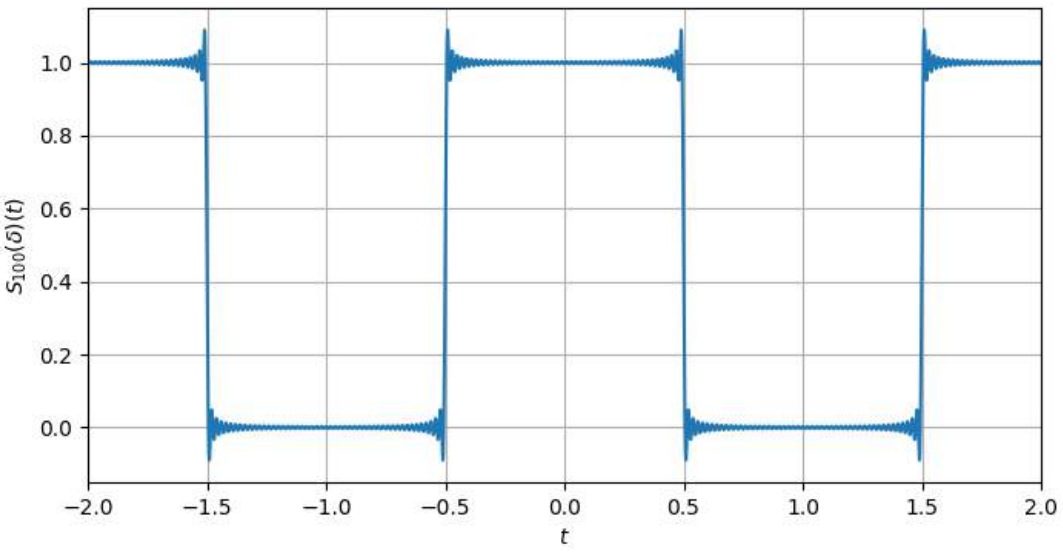


FIGURE 15 – Graphe de la somme partielle $S_{100}(\delta)$

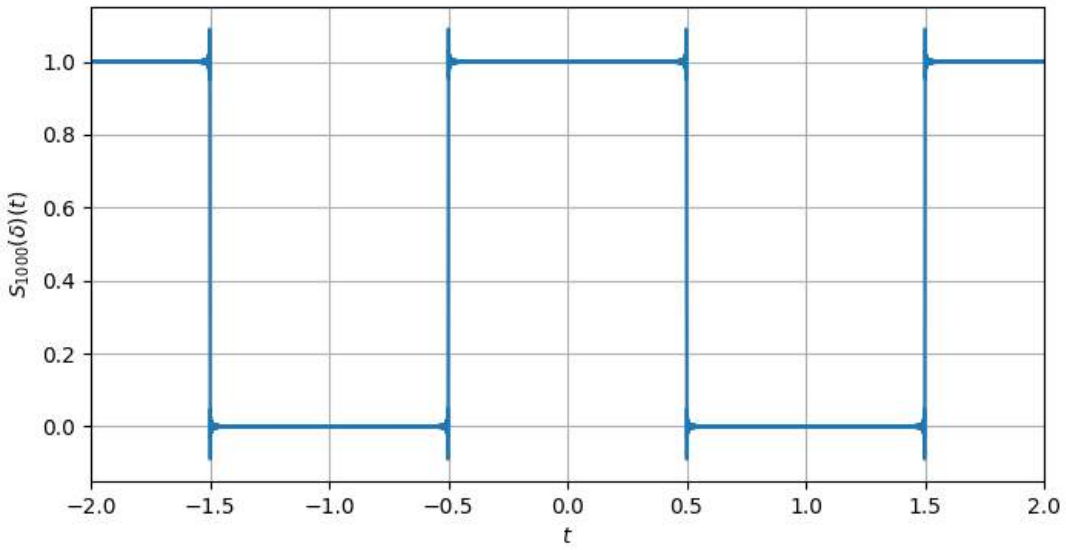


FIGURE 16 – Graphe de la somme partielle $S_{1000}(\delta)$

Bien que l'on ait considérablement augmentée la fréquence d'échantillonnage F_e du signal δ et donc le nombre de coefficients de Fourier calculés, le phénomène de Gibbs reste plutôt perceptible avec des "pics" qui se redressent au niveau des points de discontinuité. Néanmoins, les effets d'oscillation bien visibles sur la figure 14 (zone entourée en vert) n'apparaissent plus sur la figure 16.
Il est cependant possible de restituer le signal δ d'origine en appliquant la transformée de Fourier discrète sur le spectre de la figure 16 :

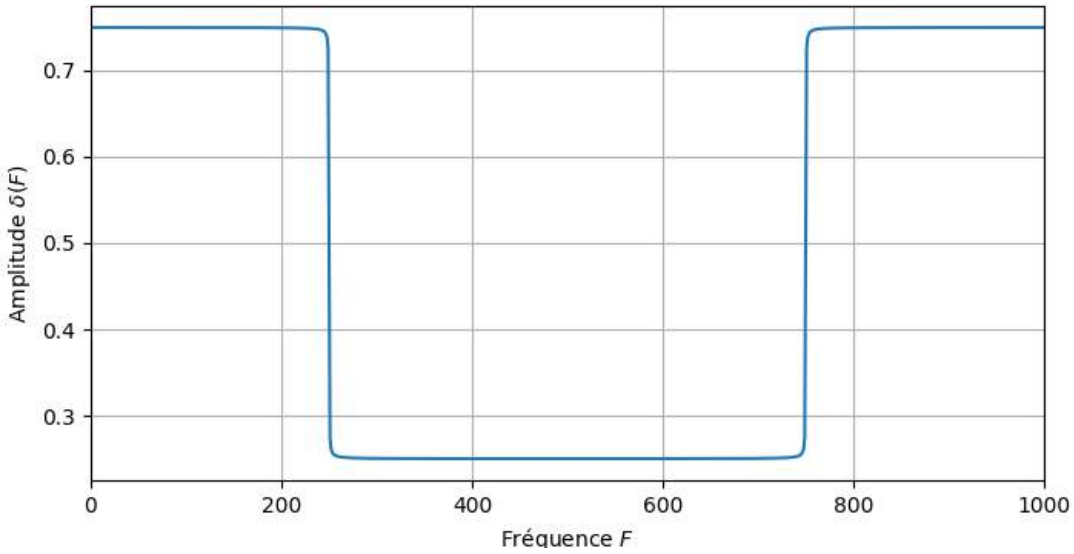


FIGURE 17 – Graphe du signal δ reconstruit à partir du spectre de $S_{1000}(\delta)$

On remarque que les fréquences maximales du signal d'origine ont été atténuées : on n'atteint plus la valeur 1. Cela a pour effet de contrebalancer et donc de faire disparaître le phénomène de Gibbs.

Code Python utilisé pour les tests :

```python
import math,cmath
import numpy as np
import numpy.fft
from matplotlib.pyplot import *
import matplotlib.pyplot as plt

# Signal créneau (delta)
def delta(x) :
    n=x%2  # On ramène la valeur x dans l'intervalle [0,2]
    if (0.5<n and n<1.5) :
        return 0
    else :
        return 1

# On stocke les images de d dans un tableau
def images(tab) :
    val = len(tab) * [0]
    for i in range(len(tab)) :
        val[i] = delta(tab[i])
    return val

# Affichage du graphe de delta sur [0,20]
fig, ax = plt.subplots()
x = np.linspace(-2,2,2000)
y = images(x)
plt.plot(x,y, color='orange')
plt.grid()
xlabel('$t$')
ylabel('$\delta(t)$')

# Calcul des coefficients c_n à partir de la formule
def calcul_cn(n):
    if (n==0):
        return 0.5
    else:
        return (np.sin(np.pi*n/2))/(np.pi*n)

# Représentation du spectre avec les coefficients c_n
def plotSpectre(nmax):
    if (nmax % 2 == 0) :
        ind = np.arange(start=-nmax-1,stop=nmax,step=2)
    else :
        ind = np.arange(start=-nmax,stop=nmax,step=2)
```

```
    indices = np.append(ind, 0)
    n = len(indices)
    spectre = np.zeros(n)
    for k in range(0,n):
        spectre[k] = abs(calcul_cn(indices[k]))
    stem(indices,spectre)
    xlabel('$n$')
    ylabel('$c_n(\delta)$')
    grid()
figure(figsize=(8,4))
plotSpectre(20)
plt.xlim(-20,20)
plt.ylim(0,0.55)

def serie(nmax,t):
    s = 0.5  # initialisation à c_0
    for k in range(1,nmax):
            z = calcul_cn(k)*cmath.exp(1j*math.pi*k*t)
            s += 2*z.real # On utilise l'argument de symétrie du spectre
    return s

# Signal S_20(delta)
def Somme_partielle(nmax,step):
    t = np.arange(start=-2,stop=2,step=step)
    n = t.size
    u = np.zeros(n)
    for k in range(n):
        u[k] = serie(nmax,t[k])
    plot(t,u)
    xlabel('$t$')
    ylabel("$S_{1000}(\delta)(t)$")
    grid()
figure(figsize=(8,4))
Somme_partielle(1000,0.001) # N = 1000 (ou 20, 100)
plt.xlim(-2,2)
plt.ylim(-0.15,1.15)

# Transformée de Fourier discrète inverse appliquée au spectre
def calculerSignal(nmax):
    indices = np.arange(start=0,stop=nmax,step=1)
    n = len(indices)
    spectre = np.zeros(n,dtype=numpy.complex_)
    for k in range(n):
        spectre[k] = calcul_cn(indices[k])
    u = numpy.fft.ifft(spectre)*nmax
```

```
    return u
u = calculerSignal(1000)
figure(figsize=(8,4))
plot(u)
xlabel('Fréquence $F$')
ylabel('Amplitude $\delta(F)$')
xlim(0,1000)
grid()

plt.show()
```

4.2.2 Critère de Shannon

Lorsque l'on souhaite échantillonner un signal, on s'attend normalement à perdre de l'information au cours du processus. Cependant, en 1949, Claude Shannon qui était à la fois ingénieur en génie électrique et un mathématicien américain, publia la démonstration de son théorème qui montre un résultat étonnant. Aujourd'hui, il porte le nom de Nyquist-Shannon en raison de la grande contribution du physicien Harry Nyquist à la théorie de l'échantillonnage dès 1928, qui aurait permis à Shannon de la démontrer. Néanmoins, il existe plusieurs appellations de ce théorème comme le critère de Shannon, la condition de Nyquist-Shannon, la théorie de Shannon ou bien encore la théorie de l'échantillonnage.

Théorème 4.1. *(Nyquist-Shannon) Un signal analogique f peut être entièrement restitué à partir d'un échantillonnage régulier, si et seulement si, sa fréquence d'échantillonnage F_e vérifie :*

$$\boxed{F_e > 2F_{max}}$$

où F_{max} désigne la fréquence maximale du signal échantillonné. Cette condition se définit également sous la forme :

$$\boxed{F_N > F_{max}}$$

où $F_N = F_e/2$ est la fréquence dite de Nyquist.

Preuve. Le résultat se déduit à partir d'une propriété d'isométrie reposant sur un échantillonnage particulier. Comme la preuve formelle et rigoureuse de ce théorème n'est pas immédiate et nécessite des compléments sur des notions telles que les bases hilbertiennes ainsi que divers résultats, nous décidons de l'admettre.
Le lecteur, si intéressé, est incité à lire la référence *Analyse harmonique réelle, Willem* à la page 126 traitant l'échantillonnage de Shannon, ou bien la reprise écrite par *Léo Gayral* .

Ce théorème est surprenant et fascinant à la fois car l'idée de pouvoir reconstituer entièrement un signal continu à partir d'un échantillonnage, impliquant seulement un nombre fini de valeurs, est impressionnant ! La puissance et l'utilité de ce théorème dans des cas de conversion analogique/numérique n'est donc pas à prouver.

On peut, par ailleurs, réutiliser ce résultat pour établir un lien avec la précision de la transformée de Fourier discrète quant à l'approximation des coefficients de Fourier d'un polynôme trigonométrique.

Corollaire 4.2. *Soit P un polynôme trigonométrique de degré d. Soient $c_n(P)$ ses coefficients de Fourier et $c'_n(P)$ ses coefficients de Fourier approchés par DFT d'ordre N.*
Alors :

$$\forall N \geq 2d+1, \quad \forall n \in [\![0, N-1]\!], \quad c'_n(P) = c_n(P)$$

Preuve. On pose donc :

$$\forall t \in \mathbb{R}, \quad P(t) = \sum_{n=-d}^{d} c_n(P) e^{-2i\pi n \frac{t}{T}}$$

le polynôme trigonométrique sur lequel on souhaite pratiquer un échantillonnage régulier. Notons N la taille de l'échantillon utilisé et F_e la fréquence d'échantillonnage associée. Afin de retrouver les coefficients de Fourier avec une précision totale, on a tout intérêt à ce que la fréquence d'échantillonnage vérifie le critère de Shannon :

$$F_e > 2F_{max}$$

Or, $F_e = \frac{N}{T}$ et dans le cas d'un polynôme trigonométrique, il est clair que la fréquence maximale du signal vaudra toujours $F_{max} = \frac{d}{T}$. Et donc, en remplaçant dans l'inégalité, on obtient le résultat voulu :

$$\begin{aligned} &\frac{N}{T} > 2 \times \frac{d}{T} \\ \iff &N \geq 2d + 1 \quad \square \end{aligned}$$

Ce résultat a d'autant plus de sens étant donné que l'on cherche les coefficients de Fourier d'un polynôme trigonométrique de degré d fini. En effet, on peut faire l'analogie avec le développement limité d'un polynôme de degré fini qui au bout d'un certain ordre devient exact. Ici, notre signal est une combinaison linéaire finie de fonctions $\cos(nwt)$ et $\sin(nwt)$ ce qui veut dire qu'à partir d'un certain ordre, le calcul des coefficients de Fourier par transformée de Fourier discrète est exact, en l'occurrence à partir de l'ordre $N = 2d + 1$.

4.2.3 Sous-échantillonnage : repliement spectral

On parle de sous-échantillonnage lorsque le critère de Shannon n'est pas respecté.
Pour le visualiser, nous allons prendre comme exemple le signal $f(t) = \cos(40000\pi(1-t^2))$, qui est périodique de période $T = 0,1$. On décide tout d'abord de l'échantillonner raisonnablement, c'est à dire de sorte à ce que le nombre N d'échantillons soit suffisamment grand, de sorte que la fréquence d'échantillonnage F_e vérifie bien le théorème de Nyquist-Shannon. On choisit alors $N = 2000$ et pour représenter graphiquement les échantillons, on décide de les relier par des segments.

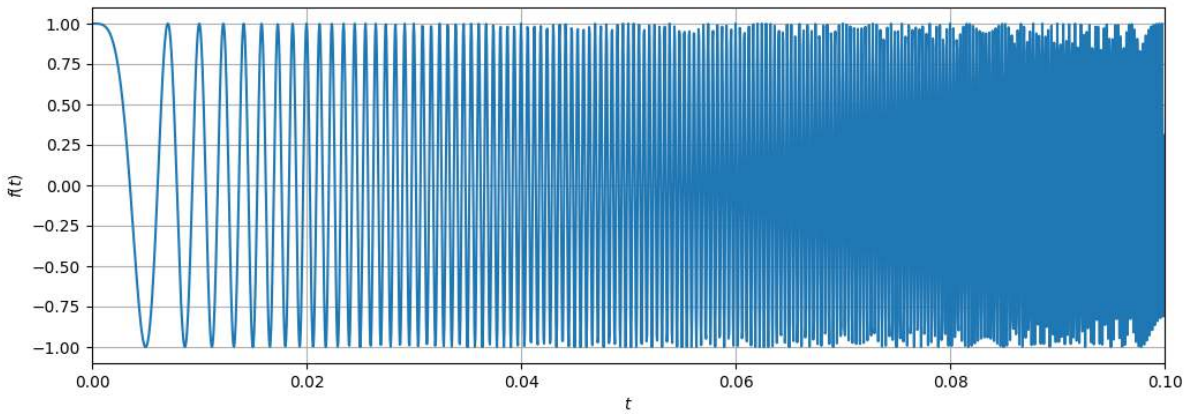


FIGURE 18 – Graphe du signal f centré près de l'origine

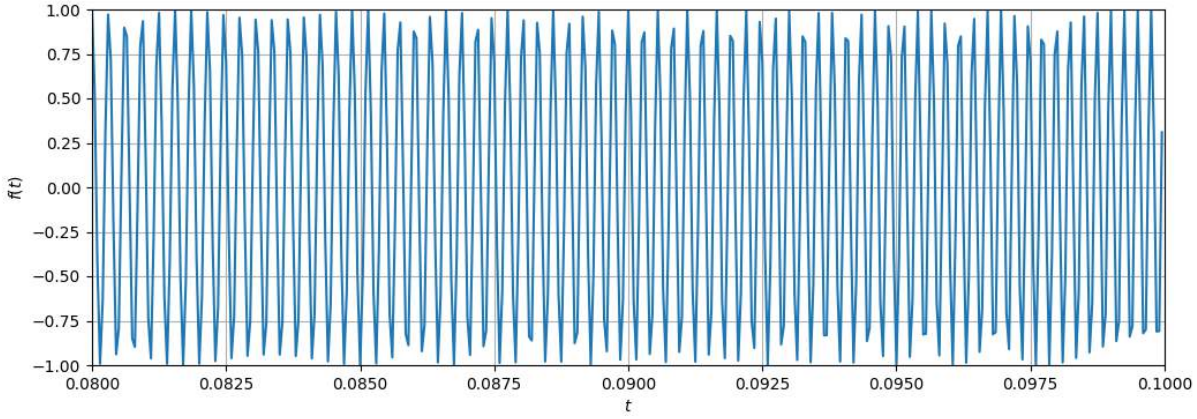


FIGURE 19 – Zoom sur la partie hautes fréquences du graphe de f : zone foncé de la figure 18

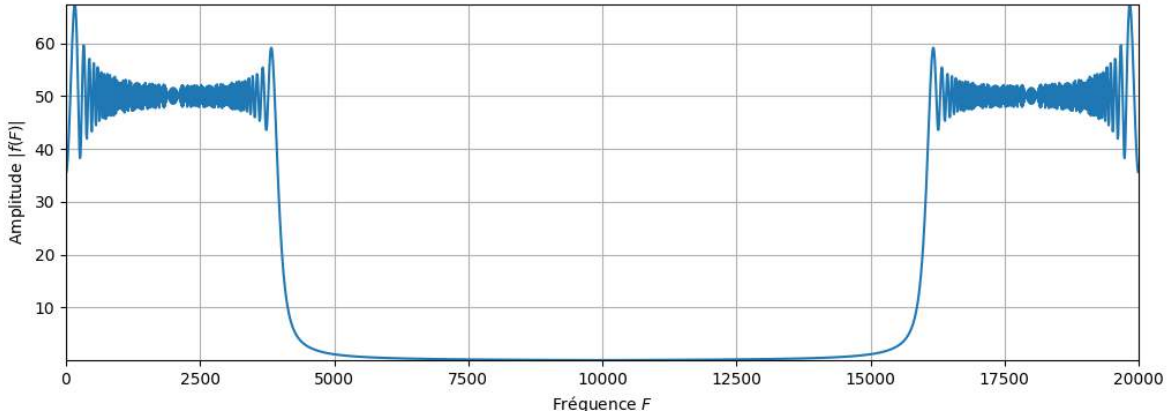


FIGURE 20 – Spectre du signal f

On voit dans la figure 18 que la fréquence de f augmente très rapidement au cours du temps t.
On observe alors que le signal contient à la fois des basses fréquences et des hautes fréquences.
Appliquons désormais la transformée de Fourier rapide pour avoir une représentation du spectre de f.
Le spectre obtenu (figure 20) est sans surprise symétrique par rapport à la fréquence de Nyquist $F_N = 10000$.
On s'aperçoit que $F_{max} \approx 6000 < F_N$, ce qui signifie que notre fréquence d'échantillonnage F_e et donc notre nombre N d'échantillons sont suffisamment grands.

Remarque : Il s'agit donc bien du signal f, entièrement reconstruit à partir des échantillons grâce à la transformée de Fourier discrète d'ordre $N = 2000$ vérifiant le critère de Shannon.

Regardons ce qu'il se passe lors d'un sous-échantillonnage, c'est à dire lorsque N et donc F_e sont trop petits pour que la condition de Nyquist-Shannon soit respectée. Faisons le choix de prendre $N = 300$, ce qui divise la fréquence d'échantillonnage précédente par 7 environ. La fréquence de Nyquist se retrouve à $F_N \approx 1428$, ce qui est très inférieur à la fréquence max. du signal qui est à 6000. Quand l'écart est aussi important, on parle de sous-échantillonnage abusif et cette différence est proportionnelle aux dégâts causés au signal en sortie. Notons $\hat{f}$ le signal obtenu par ce sous-échantillonnage de f.

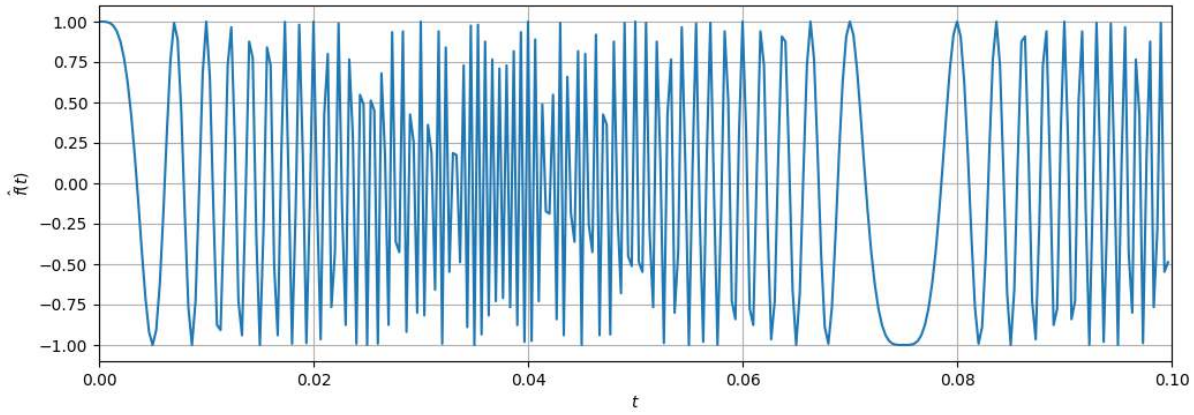


FIGURE 21 – Graphe de $\hat{f}$

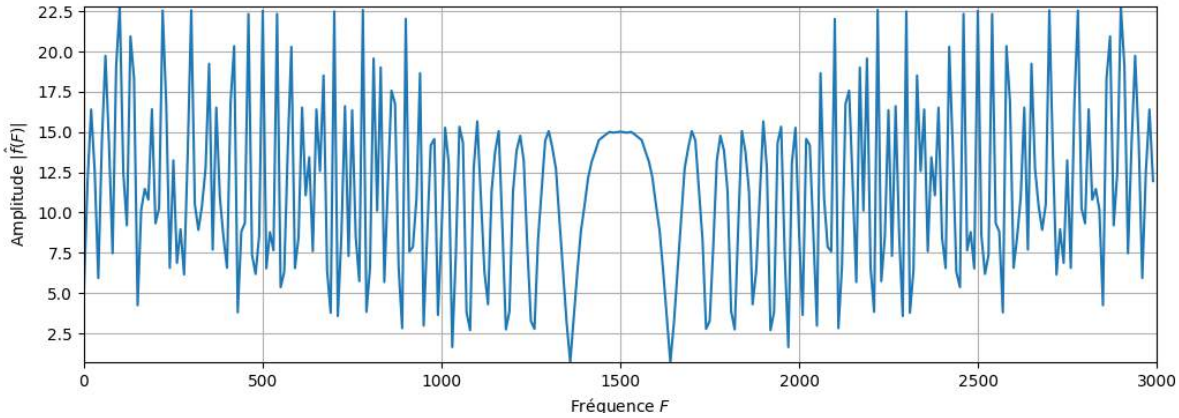


FIGURE 22 – Spectre de $\hat{f}$

Sur la figure 21, on observe que le signal n'a plus rien à voir avec celui de f, et qu'un peu en dessous de $t = 0.08$ s, des "nouvelles" fréquences bien plus faibles se sont glissées dans la zone où les fréquences sont censées être très élevées. Cependant, si on regarde bien, ces fréquences ne sont pas nouvelles mais sont des répliques ou des "alias" des basses fréquences présentes dans le signal.
On appelle ce phénomène le repliement de spectre ou l'aliasing (terme anglais). À noter que les fréquences répliquées sont propres au signal.
Par ailleurs, sur le spectre de $\hat{f}$ (voir figure 22), on constate que les deux parties symétriques se sont entremêlées. Par conséquent, le spectre de f est très méconnaissable : celui de $\hat{f}$ n'est en aucun cas ressemblant.
On appelle cela un repliement de bande en raison des bandes fréquentielles qui se sont retrouvées de l'autre côté du signal. Il s'agit d'ailleurs du même phénomène que le repliement de spectre, mais d'un point de vue fréquentielle et non temporel !

Remarque : Ces phénomènes de repliement s'expliquent par le point (i) du théorème 2.14 qui stipule que :

$$\boxed{c'_n(f) = \sum_{q=-\infty}^{+\infty} c_{n+qN}}$$

ce qui montre l'ajout de coefficients de Fourier "parasites" et donc d'autres fréquences ($\neq n$), par le calcul de la transformée de Fourier du signal dans le cas d'un mauvais échantillonnage. Nonobstant, en notant F_n et F_{n+qN} les fréquences respectives associées aux coefficients de Fourier c_n et c_{n+qN}, on a :

$$F_{n+qN} = \frac{n+qN}{T} = \frac{n}{T} + q \times \frac{N}{T} = F_n + qF_e = F_n + 2qF_N \quad q \in \mathbb{Z}^*$$

Code Python utilisé pour les tests :

```
import math
import numpy
from numpy.fft import fft
from matplotlib.pyplot import *


def f(t):
    return math.cos(40000*math.pi*(1-t*t))

# Graphe de f
T=0.1
N=2000
Te=T/N
fe=1.0/Te
t = numpy.zeros(N)
echant = numpy.zeros(N)
for k in range(N):
    t[k] = k*Te
    echant[k] = f(t[k])

# Fenêtre placée sur l'intervalle [0,0.1]
figure(figsize=(12,4))
plot(t,echant)
xlabel('$t$')
ylabel('$f(t)$')
xlim(0,0.1)
grid()

# Fenêtre placée sur l'intervalle [0.08,0.1]
figure(figsize=(12,4))
plot(t,echant)
xlabel('$t$')
ylabel('$f(t)$')
grid()
axis([0.08,0.1,-1,1])

# Calcul du spectre de f par DFT
tfd = fft(echant)
```

```
spectre = numpy.absolute(tfd)
freq = numpy.zeros(N)
for k in range(N):
    freq[k] = k*1.0/T
figure(figsize=(12,4))
plot(freq,spectre)
axis([0,fe,spectre.min(),spectre.max()])
xlabel('Fréquence $F$')
ylabel('Amplitude $|f(F)|$')
grid()

# Sous-echantillonnage de f avec N = 300
N=300
Te=T/N
fe=1.0/Te
t = numpy.zeros(N)
echant = numpy.zeros(N)
for k in range(N):
    t[k] = k*Te
    echant[k] = f(t[k])
figure(figsize=(12,4))
plot(t,echant)
xlabel('$t$')
ylabel('$\hat{f}(t)$')
xlim(0,0.1)
grid()

# Spectre du signal sous-echantillonné
tfd = fft(echant)
spectre = numpy.absolute(tfd)
freq = numpy.zeros(N)
for k in range(N):
    freq[k] = k*1.0/T
figure(figsize=(12,4))
plot(freq,spectre)
axis([0,fe,spectre.min(),spectre.max()])
xlabel('Fréquence $F$')
ylabel('Amplitude $|\hat{f}(F)|$')
grid()


show()
```

4.3 Traitement des images : étude des signaux bidimensionnels

4.3.1 Transformée de Fourier discrète en dimension 2 (DFT2)

On considère cette fois-ci un signal f discret de dimension 2. Dans ce cas de figure, le signal f est une certaine image de dimension $M \times N$, qui peut se représenter intuitivement sous forme d'une matrice :

$$U = \begin{pmatrix} u_{1,1} & u_{1,2} & \dots & u_{1,N} \\ u_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ u_{M,1} & \dots & \dots & u_{M,N} \end{pmatrix}$$

de taille $M \times N$, où le coefficient $u_{m,n}$ de U définit la valeur du pixel situé au point $(m,n) \in [\![0, M-1]\!] \times [\![0, N-1]\!]$ du plan de l'image.
$M \times N$ constitue alors la définition de l'image, avec M et N désignant resp. la longueur et la largeur de l'image en nombre de pixels.

Proposition 4.3. *La transformée de Fourier discrète s'étend en dimension 2, et associe à une image f (comme définie plus haut), une image $\hat{f}$ représentée par une matrice $\hat{U}$ de taille $M \times N$ dont les coefficients sont définis par :*

$$\forall (k,l) \in [\![0, M-1]\!] \times [\![0, N-1]\!], \quad \hat{u}_{k,l} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} u_{m,n} \omega_M^{-mk} \omega_N^{-nl}$$

C'est un automorphisme de $\mathcal{M}_{M,N}(\mathbb{C})$, et les coefficients de U s'obtiennent par la relation :

$$\forall (k,l) \in [\![0, M-1]\!] \times [\![0, N-1]\!], \quad u_{k,l} = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \hat{u}_{m,n} \omega_M^{mk} \omega_N^{nl}$$

qui définit la transformée de Fourier discrète inverse en dimension 2.

Preuve. La preuve est la même qu'en dimension 1. En effet, elle repose toujours sur l'interpolation de f par un polynôme trigonométrique P, dont le domaine spectral est cependant :

$$\Omega_{M,N} = \left\{ -\frac{M}{2}, \dots, \frac{M}{2} - 1 \right\} \times \left\{ -\frac{N}{2}, \dots, \frac{N}{2} - 1 \right\}$$

À nouveau, comme on évalue en les racines $M^{ième}$ et $N^{ième}$ de l'unité (ω_M et ω_N) qui sont toutes distinctes, le polynôme P est unique. Par conséquent, la transformée de Fourier discrète reste inversible en dimension 2. La démonstration pour obtenir la DFT2 inverse suit le même raisonnement qu'en dimension 1. □

Remarque : La transformée de Fourier discrète reste valable en dimension quelconque.
Cependant, ce mémoire se restreint à la dimension 2.

Donc, comme la transformée de Fourier discrète en dimension 2 hérite des mêmes propriétés qu'en dimension 1, l'algorithme de la transformée de Fourier rapide reste également applicable sur des images.

Justement pour notre étude en dimension 2, nous allons prendre une photo personnelle du château de Chambord :

FIGURE 23 – Photo du château de Chambord (format .jpg)

Cette image sera notre exemple sur lequel nous allons travailler. La définition de l'image étant 1600×777, elle peut donc être représentée par une matrice U de même taille, soient $M = 1600$ et $N = 777$.

4.3.2 Phase, module et reconstruction d'une image numérique

Dans cette partie, on s'intéresse plus particulièrement au contenu fréquentiel d'une image obtenue par transformée de Fourier discrète. Ce sujet aurait pu être abordé en dimension 1, cependant les test et résultats sont un peu plus parlant sur des images d'où ce choix de le traiter en dimension 2.

Objectif : Déterminer la ou les parties du signal contenant les données importantes du signal échantillonné.

On souhaite pour cela calculer la transformée de Fourier discrète de U avec $M = 1600$ et $N = 777$ (voir figure 23) et étudier son spectre. Seulement l'image $\hat{U}$ obtenue sera généralement complexe, et il est naturel de se demander comment la représenter en dimension 2.
La réponse est que l'on a pour habitude de calculer le module $|.|$ et la phase Φ de l'image, en représentant le module. On calcule donc resp. les valeurs $|\hat{u}_{m,n}|$ et $\Phi(\hat{u}_{m,n})$ et on établit les deux spectres :

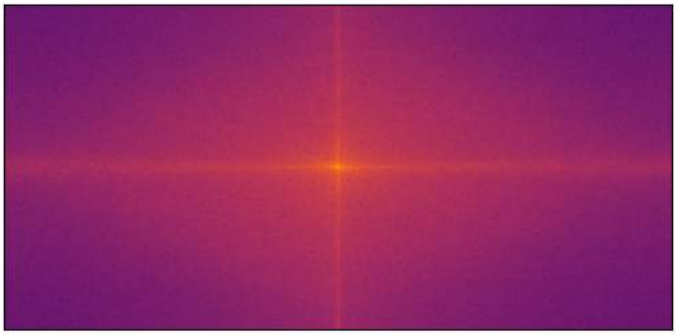


FIGURE 24 – Spectre du module de $\hat{U}$

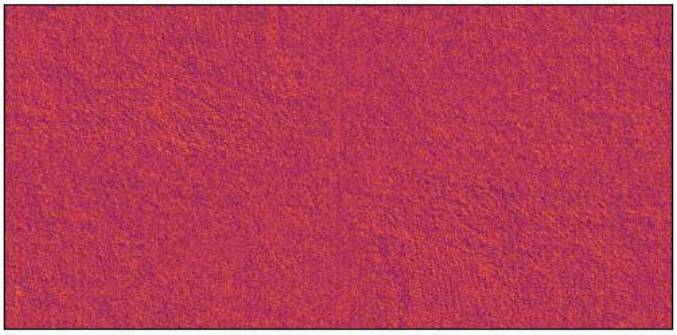


FIGURE 25 – Spectre de la phase de $\hat{U}$

Sur la figure 24, on aperçoit un point lumineux au centre qui correspond à la fréquence nulle du spectre. En effet, on a conventionnellement placé cette fréquence au centre de chacun des spectres.
Le spectre de la phase de $\hat{U}$ (voir figure 25) est encore moins visuel que celui du module. C'est pour cela que l'on décide d'effectuer la transformation de Fourier discrète inverse sur chacun de ces spectre pour mieux comprendre les informations qu'ils contiennent sur l'image 23 d'origine.
Malheureusement, comme il a été difficile de l'effectuer sur le module en particulier, nous allons utilisé un exemple tiré d'une page du site *StackExchange*.

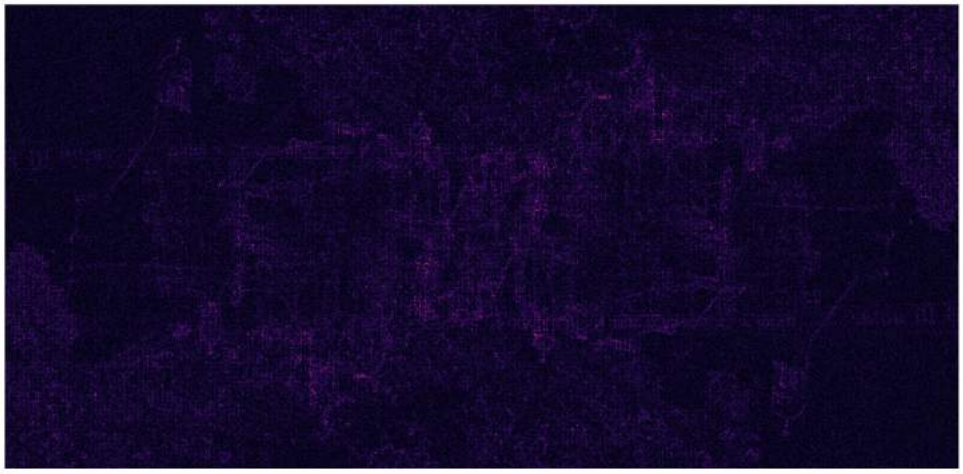


FIGURE 26 – Château de Chambord reconstitué uniquement à partir de la phase

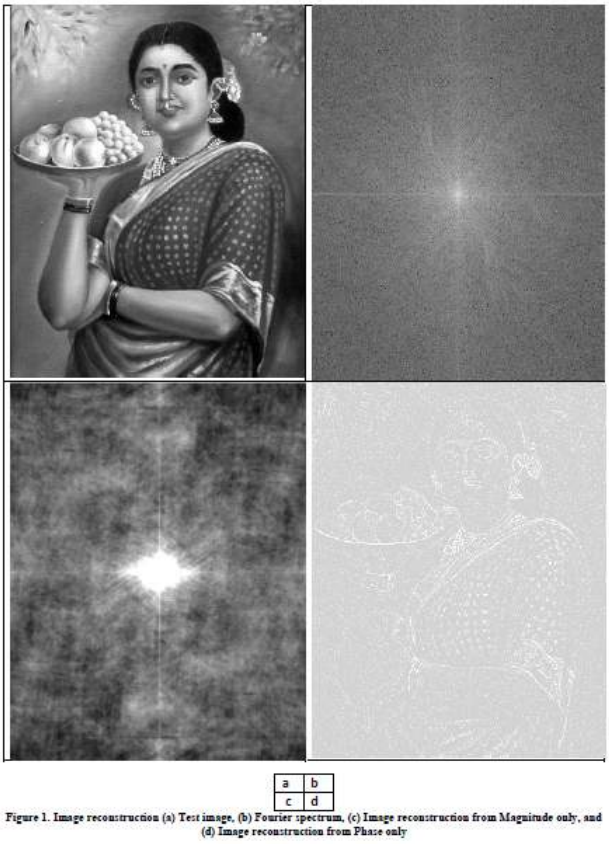
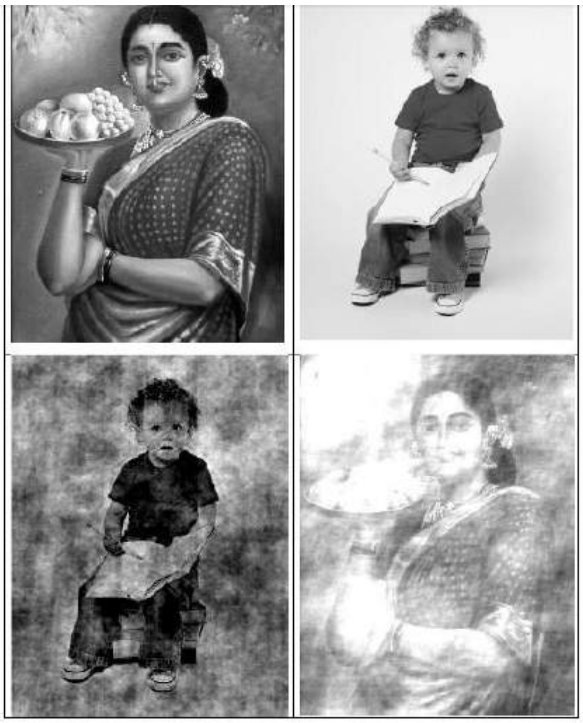


FIGURE 27 – (*a*) Image test (*b*) Spectre de Fourier (*c*) Image reconstituée à partir du module (*d*) Image reconstitutée à partir de la phase

Nous remarquons, grâce à la figure 27 , que la reconstitution de l'image par le module ne donne rien de ressemblant : la dame visible sur l'image d'origine est introuvable et aucun élément ne ressort !
Ce qui est néanmoins très surprenant c'est que l'on s'attend désormais à ce qu'il en soit de même pour la phase. Or, on est surpris de constater que sur la figure 26 par exemple, on arrive à reconnaître les bords qui dessinent le château. Pour plus de lisibilité, on peut également se tourner vers l'image présente en bas à droite de la figure 27, où on perçoit mieux les contours de l'image d'origine.
Également, c'est ce qui est observé sur l'image 2 de *Joseph Fourier* mise en introduction du mémoire (excepté l'effet de rotation).
On en conclut que la phase renferme les fréquences les plus élevées du signal, car ces dernières servent justement à décrire les contours sur les images. À l'inverse, le module va plutôt contenir des informations à propos d'éléments non mis en avant sur l'image. On va plutôt y trouver des basses fréquences.
On peut par ailleurs tenter de reconstruire l'image du Château en utilisant essentiellement le module :

FIGURE 28 – Château de Chambord reconstruit principalement par le module de sa TFD

On constate que l'image est sévèrement dégradée par la manifestation de basses fréquences, et du phénomène d'aliasing qui apparaît près des contours.
Enfin, il est également possible de prendre deux images et de les reconstituer en échangeant leur phase et leur module, ce qui permet davantage de comprendre leurs rôles respectifs :



a	b
c	d

(a) test image1 (b) test image2 (c) Image Reconstruction from Magnitude of Image1 and Phase of Image2 (d) Image Reconstruction from Magnitude of Image2 and Phase of Image1

FIGURE 29 – Reconstruction en échangeant module et phase

Code Python utilisé pour les tests :

```python
import math
import cv2
from pylab import *
import numpy as np
from numpy.fft import fft2, ifft2, fftshift, ifftshift
from matplotlib import pyplot as plt

# Spectre de la phase

img = cv2.imread(r"C:\Users\Anwender\Desktop\Chambord.jpg")
img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
dft = np.fft.fft2(img)
dft_shift = np.fft.fftshift(dft)
phase_spectre = np.angle(dft_shift) # On calcule la phase de la DFT

plt.imshow(phase_spectre, cmap='inferno')
plt.xticks([]), plt.yticks([])
```

```
18
19  plt.show()
20
21  # Module et reconstruction
22
23  # Lire l'image d'entrée
24  img = cv2.imread(r"C:\Users\Anwender\Desktop\Chambord.jpg",0)
25
26  # Calcul de la transformée de Fourier discrète de l'image
27  dft = cv2.dft(np.float32(img),flags = cv2.DFT_COMPLEX_OUTPUT)
28
29  # La fréquence nulle est placée au centre du spectre
30  dft_shift = np.fft.fftshift(dft)
31  magnitude_spectrum = 20*np.log(cv2.magnitude(
32        dft_shift[:,:,0],
33        dft_shift[:,:,1])
34     )
35  # Graphe du spectre obtenu
36  plt.plot(122),plt.imshow(magnitude_spectrum, cmap = 'inferno')
37  plt.xticks([]), plt.yticks([])
38
39  plt.show()
40
41  # Reconstruction du château de Chambord par TFD
42
43  dft_shift = np.fft.fftshift(dft)
44  rows, cols = img.shape
45  crow,ccol = rows//2 , cols//2
46  mask = np.zeros((rows,cols,2),np.uint8)
47  mask[crow-30:crow+30, ccol-30:ccol+30] = 1
48
49  # Application du masque et de la DFT inverse
50
51  fshift = dft_shift*mask
52  f_ishift = np.fft.ifftshift(fshift)
53  img_back = cv2.idft(f_ishift)
54  img_back = cv2.magnitude(img_back[:,:,0],img_back[:,:,1])
55
56  # Image d'origine reconstruite par DFT
57  plt.plot(122),plt.imshow(img_back, cmap = 'gray')
58  plt.xticks([]), plt.yticks([])
59
60  plt.show()
61
62  imgI = cv2.imread(r"C:\Users\Anwender\Desktop\Chambord.jpg")
```

```
63
64  # Reconstruction de l'image à partir de la phase de la DFT
65
66  img = cv2.cvtColor(imgI,cv2.COLOR_BGR2GRAY)
67  dft = np.fft.fft2(img)
68  dft_shift = np.fft.fftshift(dft)
69  phase_spectre = np.angle(dft_shift) # On calcule la phase de la DFT
70  dft_shift = np.fft.fftshift(phase_spectre)  # Recentrer la fréquence nulle
71  img = np.abs(np.fft.ifft2(dft_shift))
72
73  plt.imshow(img, cmap='inferno')
74  plt.xticks([]), plt.yticks([])
75
76  plt.show()
```

[10] [4] [8] [9] [6] [5] [7] [1] [3] [2]

Références

[1] Université Aix-Marseille. Traitement et analyse d'images numériques partie 5 : Transformée de fourier pour l'imagerie numérique. https://www.fresnel.fr/perso/marot/Documents/Enseignements/ATI/CoursImNum4_TF2D.pdf.

[2] G. BAUDOIN et J.-F. BERCHER. Transformée de fourier discrète. École Supérieure d'Ingénieurs en Électrotechnique et Électronique, version 0.1 - Novembre 2001, https://perso.esiee.fr/~bercherj/New/polys/poly_tfd.pdf.

[3] B. Galerne. Transformée de fourier discrète 1d et 2d. https://www.idpoisson.fr/galerne/m2_tours/cours_tfd.pdf.

[4] S. Ladjal. Flou et quantification dans les images numériques. PhD thesis, Ecole Normale Supérieure de Cachan, 2005. Chapitre A.

[5] F. Legrand. Série de fourier et filtrage d'un signal en créneaux. https://www.f-legrand.fr/scidoc/docmml/sciphys/electro/creneau2/creneau2.html.

[6] F. Legrand. Transformée de fourier d'une image. https://www.f-legrand.fr/scidoc/docmml/numerique/tfd/tfdimage/tfdimage.html, Janvier 2016.

[7] Auteur non spécifié. Analyse fréquentielle d'un signal par transformée de fourier. https://cpge.frama.io/fiches-cpge/Python/Analyse%20fr%C3%A9quentielle%20%28FFT%29/01%20-%20Transform%C3%A9e%20de%20Fourier%201D/.

[8] 3Blue1Brown (pseudonyme). But what is convolution ? https://www.youtube.com/watch?v=KuXjwB4LzSA, Novembre 2022.

[9] Reducible (pseudonyme). The fast fourier transform (fft) : Most ingenious algorithm ever ? https://www.youtube.com/watch?v=h7apO7q16V0, Novembre 2020.

[10] C. Gasquet P. Witomski. Analyse de Fourier et applications : filtrage, calcul numérique et ondelettes. Dunod, Malakoff, France, 2000.