LA TRANSFORMÉE DE FOURIER DISCRÈTE

Christian RONSE, LSIIT UMR 7005 CNRS-ULP, Département d'Informatique de l'ULP

Il s'agit de la transformée de Fourier dans le cas échantillonné périodique. Pour simplifier les choses, on suppose un espace unidimensionnel. Le cas multidimensionnel sera traité plus loin. On a $E = \Delta\mathbb{Z}/N\Delta\mathbb{Z}$, où Δ est le pas d'échantillonnage et $N\Delta$ la période ($N \in \mathbb{N}^*$).

Une fonction f sur E est donnée par ses valeurs $f(0), f(\Delta), \ldots, f\big((N-1)\Delta\big)$. La convolution de deux fonctions f et g est donnée par

$$(f \circledast g)(x\Delta) = \frac{1}{N\Delta} \sum_{t=0}^{N-1} f(t\Delta)g(x\Delta - t\Delta) \qquad (x = 0, \ldots, N-1). \tag{1}$$

Il s'agit ici d'une convolution périodique, dans le sens que $x\Delta - t\Delta$ est calculé modulo $N\Delta$.

La transformée de Fourier $\mathcal{F}(f)$ de f sera aussi échantillonnée et périodique, avec le pas d'échantillonnage $\Psi = \frac{1}{N\Delta}$ et la période $N\Psi = \frac{1}{\Delta}$. Ecrivons F pour $\mathcal{F}(f)$. On a

$$F(u/N\Delta) = \frac{1}{N\Delta} \sum_{x=0}^{N-1} f(x\Delta) \, \exp[-2\pi i \, (u/N\Delta)(x\Delta)] \qquad (u = 0, \ldots, N-1);$$
$$f(x\Delta) = \Delta \sum_{u=0}^{N-1} F(u/N\Delta) \, \exp[+2\pi i \, (u/N\Delta)(x\Delta)] \qquad (x = 0, \ldots, N-1).$$

Ici le produit $(u/N\Delta)(x\Delta)$ dans l'exponentielle imaginaire se simplifie en ux/N. En écrivant f_x pour $f(x\Delta)$ et F_u pour $F(u/N\Delta)$, cela donne :

$$\begin{aligned} F_u &= \frac{1}{N\Delta} \sum_{x=0}^{N-1} f_x \, \exp[-2\pi i \, ux/N] \qquad (u = 0, \ldots, N-1); \\ f_x &= \quad \Delta \sum_{u=0}^{N-1} F_u \, \exp[+2\pi i \, ux/N] \qquad (x = 0, \ldots, N-1). \end{aligned} \tag{2}$$

Généralement on adopte une des conventions suivantes pour la valeur du pas d'échantillonnage Δ :

- $\Delta = 1$ et ainsi $1/N\Delta = 1/N$;
- $\Delta = 1/N$ et ainsi $1/N\Delta = 1$;
- $\Delta = 1/\sqrt{N}$ et ainsi $1/N\Delta = 1/\sqrt{N}$.

La première de ces trois conventions est la plus usitée.

ALGORITHME RAPIDE

La complexité d'un calcul direct par (2) de la transformée de Fourier discrète est en $\Theta(N^2)$; nous allons donner un algorithme en $\Theta(N \log N)$.

Nous faisons ici la convention $\Delta = 1$. Posons

$$W_N = \exp[-2\pi i/N].$$

On a alors

$$F_u = \frac{1}{N} \sum_{x=0}^{N-1} f_x \, W_N^{ux}.$$

Supposons que $N = 2M$. On a $W_M = W_{2M}^2 = W_N^2$. Les $x = 0, \ldots, N-1$ se divisent en ceux qui sont pairs, de la forme $2y$ pour $y = 0, \ldots, M-1$, et ceux qui sont impairs, de la forme $2y+1$ pour $y = 0, \ldots, M-1$. On a donc :

$$\begin{aligned}
F_u = \frac{1}{N}\sum_{x=0}^{N-1} f_x\, W_{2M}^{ux} &= \frac{1}{2}\Big[\frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_{2M}^{u(2y)} + \frac{1}{M}\sum_{y=0}^{M-1} f_{2y+1} W_{2M}^{u(2y+1)}\Big] \\
&= \frac{1}{2}\Big[\frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_{2M}^{2uy} + \frac{1}{M}\sum_{y=0}^{M-1} f_{2y+1} W_{2M}^{2uy} W_{2M}^{u}\Big] \\
&= \frac{1}{2}\Big[\frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_{M}^{uy} + \frac{W_N^u}{M}\sum_{y=0}^{M-1} f_{2y+1} W_{M}^{uy}\Big].
\end{aligned}$$

Comme $W_M^M = 1$ et $W_N^M = -1$, on obtient aussi :

$$\begin{aligned}
F_{u+M} &= \frac{1}{2}\Big[\frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_{M}^{(u+M)y} + \frac{W_N^{(u+M)}}{M}\sum_{y=0}^{M-1} f_{2y+1} W_{M}^{(u+M)y}\Big] \\
&= \frac{1}{2}\Big[\frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_{M}^{uy} W_M^{My} + \frac{W_N^u W_N^M}{M}\sum_{y=0}^{M-1} f_{2y+1} W_{M}^{uy} W_M^{My}\Big] \\
&= \frac{1}{2}\Big[\frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_{M}^{uy} - \frac{W_N^u}{M}\sum_{y=0}^{M-1} f_{2y+1} W_{M}^{uy}\Big].
\end{aligned}$$

Posant pour $u = 0, \ldots, M-1$

$$\begin{aligned}
F_u^P &= \frac{1}{M}\sum_{y=0}^{M-1} f_{2y} W_M^{uy} \\
\text{et} \qquad F_u^I &= \frac{1}{M}\sum_{y=0}^{M-1} f_{2y+1} W_M^{uy},
\end{aligned} \tag{3}$$

on voit que F^P est la transformée de Fourier discrète de la fonction à M échantillons $f_0, f_2, \ldots, f_{2M-2}$, tandis que F^I est la transformée de Fourier discrète de la fonction à M échantillons $f_1, f_3, \ldots, f_{2M-1}$. On a aussi pour $u = 0, \ldots, M-1$:

$$\begin{aligned}
F_u &= \frac{1}{2}\big[F_u^P + W_N^u F_u^I\big] \\
\text{et} \qquad F_{u+M} &= \frac{1}{2}\big[F_u^P - W_N^u F_u^I\big].
\end{aligned} \tag{4}$$

On voit donc que la transformée de Fourier discrète sur $N = 2M$ échantillons se calcule par combinaison de deux transformées de Fourier discrètes sur M échantillons.

Supposons que $N = 2^n$. On obtient ainsi un algorithme récursif pour la transformée de Fourier discrète. On calcule au départ les $W_{2^n}^u$ $(u = 0, \ldots, 2^{n-1}-1)$, et ceux-ci donneront les $W_{2^k}^v$ pour $k < n$, car $W_{2^k}^v = W_{2^n}^{2^{n-k}v}$. On applique récursivement la décomposition (4) pour les transformées sur 2^{n-1} échantillons données en (3). La récursion s'arrête pour $n = 1$, c.à.d. pour 1 échantillon, où la transformée de Fourier discrète se réduit à l'identité ($F_0 = f_0$ pour 1 échantillon). Nous illustrons ci-dessous la décomposition récursive pair/impair pour $N = 8$:

$$\begin{array}{cccccccc}
\multicolumn{8}{c}{0,1,2,3,4,5,6,7} \\
\multicolumn{4}{c}{0,2,4,6} & \multicolumn{4}{c}{1,3,5,7} \\
\multicolumn{2}{c}{0,4} & \multicolumn{2}{c}{2,6} & \multicolumn{2}{c}{1,5} & \multicolumn{2}{c}{3,7} \\
0 & 4 & 2 & 6 & 1 & 5 & 3 & 7
\end{array}$$

On voit que l'ordre des échantillons à la fin de la décomposition récursive en pair/impair correspond à l'ordre des nombres où on a inversé l'ordre des bits : l'inversion de bits des nombres transforme la suite $0,1,2,3,4,5,6,7$ en la suite $0,4,2,6,1,5,3,7$.

Pour obtenir la transformée de Fourier discrète inverse, l'algorithme est semblable ; on a une décomposition comme en (4), mais sans division par 2, et avec W^{-1} au lieu de W. Sinon, on peut remarquer à partir de la définition (2), que $\overline{f_x}$ $(x=0,\ldots,N-1)$ est à un facteur constant près la transformée de Fourier discrète de $\overline{F_u}$ $(u=0,\ldots,N-1)$.

La complexité en calcul de l'algorithme est de $m(n)$ multiplications et $a(n)$ additions. On vérifie que $a(0)=m(0)=0$, tandis que pour $n>0$ la récursion (4) donne

$$\begin{aligned} & m(n) = 2m(n-1) + 2^{n-1} \\ \text{et} \qquad & a(n) = 2a(n-1) + 2^n; \end{aligned}$$

(la division par 2 n'est pas comptée comme une multiplication). Cela donne donc

$$\begin{aligned} & m(n) = 2^{n-1}n = \frac{1}{2}N\log_2(N) \\ \text{et} \qquad & a(n) = 2^n n = N\log_2(N). \end{aligned}$$

Donc la complexité de cet algorithme est en $\Theta(N\log N)$, comparée à celle d'un calcul direct (2), qui est en $\Theta(N^2)$ (en fait N^2 multiplications et $N(N-1)$ additions). Cet algorithme rapide pour la transformée de Fourier discrète est appelé *transformée de Fourier rapide*.

La complexité de la convolution par calcul direct (1) est aussi en $\Theta(N^2)$, donc il est plus économique de réaliser la convolution de f par g comme suit: (a) calculer les transformées de Fourier discrètes F et G de f et g par l'algorithme rapide ; (b) multiplier F par G (la complexité est en $\Theta(N)$) ; faire la transformée de Fourier discrète inverse de FG. La complexité totale est donc en $\Theta(N\log N)$.

SIGNAUX DISCRETS NON PERIODIQUES A SUPPORT BORNÉ

Considérons les signaux échantillonés non-périodiques avec pas d'échantillonnage Δ, c.à.d. les fonctions définies sur $\Delta\mathbb{Z}$. La convolution de deux signaux f et g est définie par

$$(f*g)(x\Delta) = \sum_{t=0}^{N-1} f(t\Delta)g(x\Delta - t\Delta) \qquad (x=0,\ldots,N-1). \tag{5}$$

Si nous comparons cela avec la convolution périodique définie en (1), outre l'absence du facteur $1/N\Delta$, la principale différence est qu'il s'agit ici d'une convolution non-périodique, dans le sens que $x\Delta - t\Delta$ est calculé normalement (alors qu'il l'était modulo $N\Delta$ pour (1)).

Le *support* $\mathrm{supp}(f)$ d'une fonction f est l'ensemble des positions $x\Delta$ telles que $f(x\Delta)\neq 0$. Si f est à support borné, on a un entier m_f tel que $\mathrm{supp}(f)\subseteq[-m_f\Delta, m_f\Delta]$. Pour deux fonctions f et g à support borné, leur convolution $f*g$ sera aussi à support borné :

$$\begin{aligned} & \Big(\mathrm{supp}(f)\subseteq[-m_f\Delta, m_f\Delta] \quad \text{et} \quad \mathrm{supp}(g)\subseteq[-m_g\Delta, m_g\Delta]\Big) \\ \Longrightarrow \qquad & \mathrm{supp}(f*g)\subseteq[-(m_f+m_g)\Delta, (m_f+m_g)\Delta]. \end{aligned}$$

Prenons un entier N “suffisamment grand”. Pour toute fonction h définie sur $\Delta\mathbb{Z}$, la périodisation de h modulo $N\Delta$ est la fonction $\pi_{N\Delta}(h)$ définie sur $\Delta\mathbb{Z}/N\Delta\mathbb{Z}$ par

$$\pi_{N\Delta}(h)(x\Delta) = \sum_{z\in\mathbb{Z}} f(x\Delta - zN\Delta). \tag{6}$$

Si $\mathrm{supp}(h) \subseteq]-N\Delta/2, N\Delta/2[$, on a alors

$$\pi_{N\Delta}(h)(x\Delta \ mod \ N\Delta) = h(x\Delta) \qquad \text{pour} \quad x = -\frac{N}{2}, \ldots, \frac{N}{2} - 1.$$

Donc h peut être retrouvée à partir de sa périodisation. De plus si $\mathrm{supp}(f) \subseteq [-m_f\Delta, m_f\Delta]$ et $\mathrm{supp}(g) \subseteq [-m_g\Delta, m_g\Delta]$ pour $m_f + m_g < N/2$, alors $\mathrm{supp}(f * g) \subseteq]-N\Delta/2, N\Delta/2[$, donc $f * g$ peut être obtenu à partir de $\pi_{N\Delta}(f * g)$, et on a de plus

$$\pi_{N\Delta}(f) \circledast \pi_{N\Delta}(g) = \frac{1}{N\Delta}\pi_{N\Delta}(f * g). \tag{7}$$

Donc on aura avantage à réaliser $f * g$ en faisant la convolution périodique $\pi_{N\Delta}(f) \circledast \pi_{N\Delta}(g)$ obtenue au moyen de l'algorithme de transformée de Fourier rapide.

CAS MULTIDIMENSIONNEL

On a un espace à n dimensions, avec pour $i = 1, \ldots, n$ un pas d'échantillonnage Δ_i et une période $N_i\Delta_i$ selon l'axe i ($N_i \in \mathbb{N}^*$). On pose $\Delta = \prod_{i=1}^n \Delta_i$ et $N = \prod_{i=1}^n N_i$. La convolution devient ici :

$$(f \circledast g)(x_1\Delta_1, \ldots, x_n\Delta_n) = \frac{1}{N\Delta} \sum_{t=0}^{N-1} f(t_1\Delta_1, \ldots, t_n\Delta_n) g(x_1\Delta_1 - t_1\Delta_1, \ldots, x_n\Delta_n - t_n\Delta_n).$$

La transformée de Fourier sera échantillonnée et périodique, avec le pas d'échantillonnage $1/N_i\Delta_i$ et la période $1/\Delta_i$ selon l'axe i. La transformée de Fourier discrète directe et inverse seront données par les formules :

$$F(u_1/N\Delta_1, \ldots, u_n/N\Delta_n) = \frac{1}{N\Delta} \sum_{x_1=0}^{N_1-1} \cdots \sum_{x_n=0}^{N_n-1} f(x_1\Delta_1, \ldots, x_n\Delta_n) \exp\Big[-2\pi i \, \frac{u_1x_1 + \cdots + u_nx_n}{N}\Big];$$

$$f(x_1\Delta_1, \ldots, (x_n\Delta_n) = \Delta \sum_{u_1=0}^{N_1-1} \cdots \sum_{u_n=0}^{N_n-1} F(u_1/N\Delta_1, \ldots, u_n/N\Delta_n) \exp\Big[+2\pi i \, \frac{u_1x_1 + \cdots + u_nx_n}{N}\Big].$$

On peut obtenir la transformée de Fourier discrète à n dimensions en opérant la transformée à une dimension (2) sur chacune des coordonnées en succession : pour $i = 1, \ldots, n$, on fait pour chacun des N/N_i choix de coordonnées x_j fixées ($j \neq i$) la transformée de Fourier discrète sur la fonction f où seule varie la coordonnée x_i. La complexité au moyen de l'algorithme de transformée de Fourier rapide sera alors proportionnelle à

$$\sum_{i=1}^{n} (N_i \log N_i)(N/N_i) = \sum_{i=1}^{n} N \log N_i = N \log N.$$

Donc la complexité de l'algorithme de transformée de Fourier rapide ne dépend pas de la dimension.