12 Fourierova transformace

Fourierova transformace je jednou z nejčastěji používaných metod ke zpracování signálu. Spočívá v převodu časově závislé funkce $h(t)$ na funkci spektra harmonických frekvencí $H(\omega)$. S Fourierovou transformací analytických funkcí se seznámíte v matematické analýze a v teorii distribucí. My se zde budeme zabývat diskrétní verzí této transformace, která se používá pro numerické zpracování naměřených signálů.

12.1 Diskrétní Fourierova transformace

Transformace a případná zpětná transformace $h_j \leftrightarrow H_k$ se provádějí předpisem

$$\begin{aligned} H_k &= \frac{1}{N} \sum_{j=0}^{N-1} h_j \, \mathrm{e}^{-\frac{2\pi i j k}{N}}, \\ h_j &= \sum_{k=0}^{N-1} H_k \, \mathrm{e}^{+\frac{2\pi i j k}{N}}, \end{aligned} \tag{107}$$

kde

- h_j, $j = 0, 1, \ldots, N-1$ je *vstupní signál (časová řada)* délky N naměřený v ekvidistantních okamžicích odpovídajících časům
$$t_j = \frac{j}{f_s}. \tag{108}$$
- f_s je *vzorkovací frekvence* a odpovídá počtu měření za sekundu.
- H_k, $k = -\frac{N-1}{2}, \ldots, 0, \frac{N-1}{2}$ (pro N liché) je spektrální rozklad vstupního signálu, kde
$$f_k = |k| \, \frac{f_s}{N-1} \tag{109}$$
je frekvence odpovídajícího příspěvku. Níže ukážeme, že k jedné frekvenci existují dvě komponenty H_k a H_{-k}, které se liší jen fází. Jelikož díky periodičnosti platí $H_k = H_{k+N}$, je při programování praktičtější uvažovat index $k = 0, \ldots, N-1$. Pak H_k a H_{N-k} odpovídají stejné frekvenci.
- Nejvyšší frekvence, kterou jsme schopni ze vstupního signálu určit, nezávisí na délce signálu N a je určena pouze vzorkovací frekvencí: $f_{\max} = f_s/2$. Nazývá se *Nyquistova frekvence.*
- Nejnižší frekvence, kterou ze signálu vyextrahujeme, je $f_{\min} = \frac{f_s}{N-1}$. Pro signály velmi nízkých frekvencí potřebujeme tedy dlouhé vstupní časové řady.
- Frekvence f_0 odpovídá tzv. *stejnosměrnému příspěvku* a odpovídající komponenta H_0 udává střední hodnotu signálu.

Přímá a zpětná transformace se liší jen znaménkem a normalizačním faktorem $1/N$, jehož umístění do zpětné Fourierovy transformace je věcí konvence.[54]

Úkol 12.1: *Dokažte, že provedením Fourierovy transformace a zpětné Fourierovy transformace dostanete původní časovou řadu.*

[54] Jiná často používaná konvence je tento faktor rozdělit symetricky mezi obě transformace: pak bude před oběma sumami ve vztazích (107) stát $\frac{1}{\sqrt{N}}$ a Fourierova transformace bude unitární transformací mezi vektory $\boldsymbol{h} = (h_j)$ a $\boldsymbol{H} = (H_k)$ (unitární transformace zachovává délku komplexního vektoru).

Z předpisu (107) vyplývá, že výsledek Fourierovy transformace je obecně sekvence *komplexních* čísel. Rozepsáním komplexní exponenciály dostaneme

$$H_k^R = \frac{1}{N}\sum_{j=0}^{N-1}\left(h_j^R \cos\frac{2\pi jk}{N} + h_j^I \sin\frac{2\pi jk}{N}\right),$$
$$H_k^I = \frac{\mathrm{i}}{N}\sum_{j=0}^{N-1}\left(-h_j \sin\frac{2\pi jk}{N} + h_j^I \cos\frac{2\pi jk}{N}\right), \tag{110}$$

kde

$$h_j = h_j^R + \mathrm{i}h_j^I,$$
$$H_k = H_k^R + \mathrm{i}H_k^I \tag{111}$$

je rozklad na reálnou a imaginární část. Analogický vztah bychom získali pro zpětnou Fourierovu transformaci.

Za předpokladu, že vstupní časová řada h_j je reálná (obecně reálná být nemusí, ale v praxi obvykle bývá), můžeme pro reálnou a imaginární složku spektrálního rozkladu H_k psát

$$H_k = \frac{1}{N}\sum_{j=0}^{N-1} h_j \cos\frac{2\pi jk}{N} - \frac{\mathrm{i}}{N}\sum_{j=0}^{N-1} h_j \sin\frac{2\pi jk}{N}. \tag{112}$$

Ze sudosti cosinu a lichosti sinu plyne, že $H_k^R = H_{-k}^R$ a $H_k^I = -H_{-k}^I$.

Komponenty H_k lze také rozepsat pomocí jiných dvou reálných čísel: velikosti $|H_k|$ a fáze ϕ_k,

$$H_k = |H_k|\,\mathrm{e}^{i\phi_k}, \tag{113}$$
$$|H_k| = \sqrt{\left(H_k^R\right)^2 + \left(H_k^I\right)^2},$$
$$\phi_k = \arctan\frac{H_k^I}{H_k^R}. \tag{114}$$

V praxi je nejdůležitější údaj velikost příspěvku k-té frekvence

$$A_k = |H_k| + |H_{-k}| = 2\,|H_k|\,, \qquad k = 1, \ldots, \frac{N-1}{2}, \tag{115}$$

který udává amplitudu odpovídající harmonické vlny. Místo amplitudy se také často používá kvadrát

$$S_k \equiv |A_k|^2\,, \tag{116}$$

který se nazývá *výkonové spektrum* a udává, jak již název napovídá, výkon, který do signálu vnáší k-tá harmonická vlna.

Fourierova transformace tedy vyjadřuje signál jako součet prostých harmonických vln (sinů a cosinů) s frekvencemi f_k a amplitudami A_k. Hudební terminologií se jedná o rozklad souzvuku několika tónů na jednotlivé jednoduché tóny.

12.2 Použití Fourierovy transformace

- Získání dominantních frekvencí neznámého signálu (například délky slunečního cyklu, vlastních frekvencí kmitů složitého objektu atd.).
- Rozpoznávání hlasu.
- Filtrování signálu, odstranění šumu (signál očekáváme na určitých frekvencích, šumu se zbavíme, pokud komponenty H_k od ostatních frekvencí vynulujeme).

- Komprese signálu: uchováme jen složky signálu s několika největšími příspěvky H_k. Toho se využívá v kompresi zvukového signálu (například MP3) i obrazového signálu (formát JPEG).

My si vyzkoušíme Fourierovu transformaci na zvukovém souboru. Práce se zvukovými soubory v Pythonu je popsána v sekci 2.2.7

Úkol 12.2: *Naprogramujte Fourierovu transformaci a zpětnou Fourierovu transformaci podle vztahů* (107). *Můžete počítat buď nezávisle reálnou a imaginární část, nebo využít toho, že funkce knihovny* `numpy` *rozumějí komplexním číslům. Imaginární jednotka v Pythonu je* `1j`*, komplexní číslo* $3 + 4\mathrm{i}$ *se tedy zapíše ve tvaru* `3 + 4j`*.*

Úkol 12.3: *Vytvořte časovou řadu délky* $N = 2000$ *se vzorkovací frekvencí* $f_s = 2000\,Hz$ *(signál tedy bude trvat přesně* $1\,s$*) danou součtem tří harmonických funkcí:*

$$h_j = \sum_{n=1}^{3} a_n \sin\left(2\pi F_n t_j\right), \tag{117}$$

kde

$$a_1 = 0.1, \qquad a_2 = 0.2 \qquad a_3 = 0.3,$$
$$F_1 = 440\,Hz \qquad F_2 = \frac{5}{4}F_1 = 550\,Hz \qquad F_3 = \frac{3}{2}F_1 = 660\,Hz$$

a časy t_j *jsou dány vztahem* (108). *Přehrajte si ji pomocí funkcí knihovny* `sounddevice`.[55] *Spočítejte Fourierovu transformaci a vykreslete do grafu amplitudy* A_k *na frekvencích* f_k*. Přesvědčte se, že graf bude mít tři vrcholy s výškami odpovídajícími zadaným amplitudám* a_n *a středy v místech zadaných frekvencí* F_n*.*

Zopakujte výpočet pro $N = 1000$ *a* $f_s = 1000\,Hz$*. V tomto případě jsou již frekvence* F_2, F_3 *větší než Nyquistova frekvence* f_{max}*. Zvuk bude zkreslený a vrcholy ve frekvenčním diagramu po provedení Fourierovy transformace se posunou.*

Úkol 12.4: *Ve složce sounds v repozitáři jsou soubory* `a.wav`*,* `e.wav`*,* `i.wav`*,* `o.wav`*,* `u.wav`*, ve kterých jsou nahrané krátké vzorky samohlásek.*[56] *Vzorkovací frekvence nahrávek je* $f_s = 16000\,Hz$*, nahrávky mají délku okolo* $N \approx 8000$*. Soubory načtěte pomocí knihovny* `soundfile`*, přehrajte si je, vyřízněte z nich kvůli rychlosti okno délky* $N = 2000$ *(vezměte například prostřední část příkazem* `sound[3000:5000]`*), vypočítejte na něm Fourierovu transformaci a zakreslete získané amplitudy* A_k *do jednoho grafu pro všech pět samohlásek. Uvidíte, že každá samohláska má zcela jinak vysoké dominantní frekvence. Této skutečnosti se využívá pro strojovou analýzu hlasu.*[57]

Úkol 12.5: *Ve složce sounds v repozitáři je soubor* `BlackHolesCollision.wav`*, ve kterém je nasimulovaný průběh gravitačních vln těsně před srážkou dvou černých děr.*[58] *Načtěte tento soubor, přehrajte si ho (uslyšíte charakteristický tzv.* chirp sound*). Pozor, soubor má dva kanály, pro následující analýzu vyberte pouze jeden z nich příkazem* `sound[:,0]`*. Rozdělte časovou řadu na časová okna délky* $N_W = 2000$ *bodů a pro každé okno spočítejte Fourierovu transformaci a amplitudy* A_k*. Následně vykreslete konturový graf (spektrogram), kde na ose* x *bude čas (začátku nebo středu použitého časového okna), na ose* y *frekvence a na ose* z *(barevný kód) amplituda. Frekvence omezte pomocí příkazu* `plt.ylim(0, 500)` *na hodnoty* $\langle 0\,Hz, 500\,Hz\rangle$*.*

[55] Uvedené frekvence odpovídají po řadě tónům a^1, $c\#^2$, e^2 v přirozeném ladění. Zaznít by tedy měl durový kvintakord.

[56] Nahrávky jsou staženy z https://homepages.wmich.edu/~hillenbr/voweldata.html.

[57] Pomocí funkce `rec` si můžete nahrát a zanalyzovat vlastní hlas.

[58] Soubor pochází z http://web.mit.edu/sahughes/www/sounds.html.

12.3 Rychlá Fourierova transformace

Fourierova transformace naprogramovaná pomocí definičních vztahů (107) má časovou složitost $\mathcal{O}(N^2)$, pro delší časové řady tedy doba výpočtu prudce roste (již pro $N = 10000$ si počkáte několik minut). Existuje mnohem efektivnější algoritmus s časovou složitostí $\mathcal{O}(N \log N)$. V Pythonu naleznete funkci počítající rychlou Fourierovu transformaci ze zadané časové řady v balíku **`numpy`**. Jmenuje se fft.fft.

Úkol 12.6: *Spočítejte Fourierovu transformaci (i) pomocí funkce naprogramované v úkolu 12.2 a (ii) pomocí knihovní funkce* `numpy.fft.fft` *buď pro signál z úkolu 12.3, nebo pro jednu z hlásek z úkolu 12.4. V grafu porovnejte získané amplitudy. Měly by být stejné.*