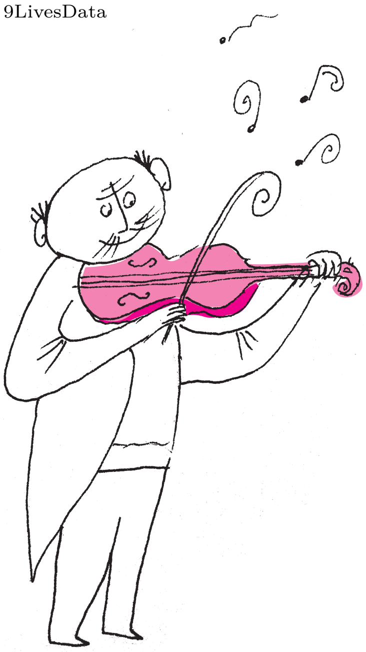
Zawijanie i wycinanie dźwięków

*Radosław KUJAWA**

* 9LivesData



Nagraliśmy ze znajomymi piosenkę. Nie było to profesjonalne przedsięwzięcie: nie wynajęliśmy studia nagraniowego, ale spotkaliśmy się u jednego z nas, wyjęliśmy instrumenty i zagraliśmy kilka razy do porządnego dyktafonu. Niestety, brak zawodowstwa dało się odczuć natychmiast – okazało się, że siedziałem na skrzypiącym krześle, które przy każdym moim ruchu robiło *ziiik*, *ziiiiiik*. Skrzypienie, choć nie permanentne, stanowczo utrudniało percepcję.

Na szczęście reżyserzy dźwięku dysponują narzędziami, które mogą „wyciąć", albo przynajmniej „schować", takie niepożądane odgłosy. Nie trzeba było więc spotykać się jeszcze raz i pamiętać, by siedzieć na innym krześle. Wystarczyło pozbyć się skrzypienia w trakcie obróbki nagrania. Taka operacja wycięcia zbędnego dźwięku jest możliwa dzięki narzędziu, które w skrócie nazwiemy FFT (*Fast Fourier Transform*, czyli Szybka Transformata Fouriera).

Żeby zobaczyć, jak działa FFT, zacznijmy od podstaw. Dźwięk to funkcja zmiany ciśnienia powietrza. Nie jest to jednak dowolna funkcja – ciśnienie podlega niewielkim zmianom, a dźwięki, przynajmniej te w przyjaznej dla ucha postaci, mają (mówiąc muzycznie) swoją wysokość, czyli (ściślej) odpowiadająca im funkcja zmiany ciśnienia powietrza jest funkcją okresową.

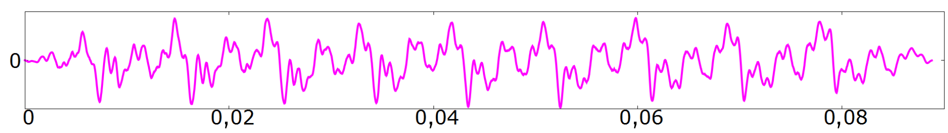
Czytelnik Lubiący Analizę Matematyczną doprecyzuje, że „porządna" znaczy tu „spełniająca warunki Dirichleta".

W funkcjach z okresem $T \neq 2\pi$ trzeba przeskalować x przez $\frac{2\pi}{T}$.

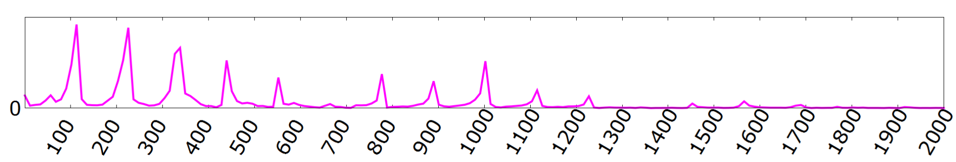
Każdą porządną funkcję okresową można przedstawić za pomocą szeregu Fouriera. W dużym uproszczeniu – jeśli funkcja f ma okres 2π, to da się ją przedstawić jako $f(x) = C + \sum_{n=1}^{\infty} a_n \sin(nx) + b_n \cos(nx)$. Tego mocnego narzędzia teoretycznego nie można tu stosować ściśle, bo na ogół dźwięki zmieniają się w czasie, więc nie mamy do czynienia z funkcjami *par excellence* okresowymi. Wciąż są one jednak „okresowawe", to znaczy zmieniają się na tyle wolno, że można lokalnie sporządzić *widmo* dźwięku – czyli powiedzieć, które częstotliwości bazowe są istotne w danej chwili. Właśnie to *widmo* jest generowane przez FFT.

Sporządzanie widma funkcji f to nic innego, jak znajdowanie takich n, że wyrażenie $a_n \sin(nx) + b_n \cos(nx)$ jest istotnym składnikiem funkcji f.

Spójrzmy na dwie postacie pierwszego dźwięku wspomnianej wcześniej piosenki – wykres drgania oraz odpowiadającego mu widma.



Rys. 1. Funkcja zmiany ciśnienia w czasie dla dźwięku struny gitarowej. Można w niej dostrzec pewną regularność, lecz trudno ją precyzyjnie opisać



Rys. 2. Widmo dźwięku z rysunku 1. Na osi X jest częstotliwość w Hz, na osi Y – amplituda drgania w danej częstotliwości

Czytelnik Obeznany z Reżyserią Dźwięku, gdy spojrzy na rysunek 2, powie z łatwością, że przedstawiony jest na nim dźwięk A z oktawy wielkiej (czyli dźwięk o bazowej częstotliwości 110 Hz).

Rysunek 2 przedstawia udział danej częstotliwości w dźwięku z rysunku 1. Ten dźwięk nie jest funkcją okresową – każdy puls ma *nieco* inny kształt – jednak mimo to można spróbować skonstruować jego szereg Fouriera. Istotną rolę będą odgrywać w nim jedynie amplitudy a_{110}, b_{110}, a_{220}, b_{220}, a_{330}, b_{330} itd.

Razem z FFT pojawia się nowy punkt widzenia, niebywale przydatny w analizie dźwięku. Z rysunku 1 można od biedy wydedukować częstotliwość bazową, ale cała subtelność kształtu widma jest już nie do wychwycenia. Na rysunku 2 zupełnie jasne jest, że szósta składowa (ząbek o $f = 660\,\mathrm{Hz}$)

niemal nie istnieje, za to dziewiąta ($f = 990\,\mathrm{Hz}$) jest tak duża jak czwarta ($f = 440\,\mathrm{Hz}$). To, że takie fakty są w ogóle możliwe do zaobserwowania, jest kluczowe chociażby w analizie mowy.

Konstrukcja widma dyskretnego sygnału to Dyskretna Transformata Fouriera (w skrócie DFT – *Discrete Fourier Transform*).

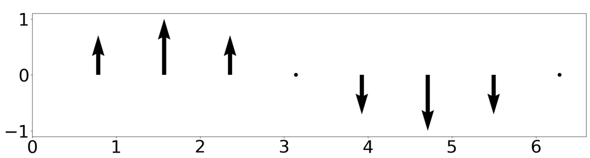
Jak skonstruować takie widmo? Załóżmy, że funkcja opisująca dźwięk jest dyskretna, a więc zadana jakimś ciągiem próbek $a_0, \ldots, a_{N-1}$. Wtedy widmo również jest dyskretne, nazwijmy jego wartości $A_0, \ldots, A_{N-1}$. Wartość A_k powie, jak istotną składową dźwięku jest drganie o częstotliwości k.

Czytelnik Oswojony z Liczbami Zespolonymi może dla $k = 0, \ldots, N-1$ oraz dla próbek $a_0, \ldots, a_{N-1}$ obliczyć k-ty element widma zgodnie ze wzorem:

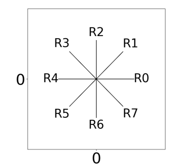
$$A_k = \left|\sum_{n=0}^{N-1} a_n e^{i\frac{2\pi kn}{N}}\right|.$$

Uwaga: w DFT moduł nie występuje – A_k jest wtedy liczbą zespoloną, która niesie ze sobą dodatkową informację o fazie drgania.

Wyobraźmy sobie płaszczyznę z punktem $(0,0)$, z którego wychodzi N promieni (zob. rys. 3b). Potraktujmy wartości próbek jako długości wektorów (zob. rys. 3a). Każdy wektor umieszczamy na odpowiednim promieniu, czyli zaczepiamy go w punkcie $(0,0)$ i nadajemy mu kierunek tegoż promienia. Dla danego k zaczynamy od k-tego promienia i z następną próbką skaczemy do promienia k kroków wprzód. Liczba A_k to długość sumy wektorów-próbek nawiniętych podczas k-tego nawijania.



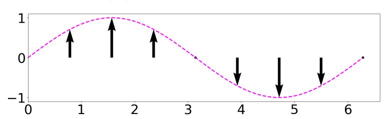
(a) 8 próbek funkcji sinus



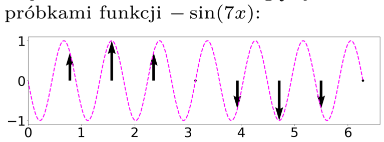
(b) 8 promieni z $(0,0)$

Rys. 3. Weźmy $N = 8$ próbek, by na rysunku 4 nawinąć je wokół punktu $(0,0)$

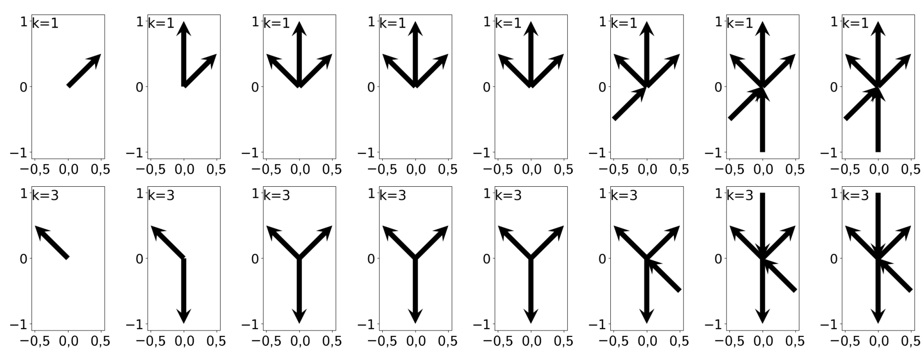
Czytelnik Wnikliwy zauważy, że oba nawinięcia, które nie sumują się do 0, są równoważne – nawijanie co 1 promień musi być symetryczne do nawijania co $N-1$ promień! Co więcej, nawijanie co $N-1$ promień ma swój własny sens. Bo choć, patrząc na próbki z rysunku 3a, spodziewamy się, że przybliżają one funkcję $\sin(x)$:



to jednak równie dobrze mogą być próbkami funkcji $-\sin(7x)$:



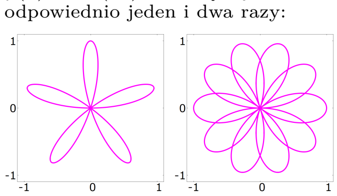
Niepożądane zjawisko, które bierze się z niejednoznacznej reprezentacji sygnału okresowego przez jego wartości chwilowe i może zaistnieć przy nieostrożnym przetwarzaniu sygnałów, nosi nazwę *aliasingu*. Polega na tym, że po obróbce sygnału pojawiają się w nim częstotliwości, których na początku nie było. Aby tego uniknąć, trzeba brać na poważnie tylko połowę nawinięć (druga połowa jest symetryczna). Zatem dla N próbek zbadamy efektywnie tylko pierwszych $N/2$ częstotliwości bazowych.



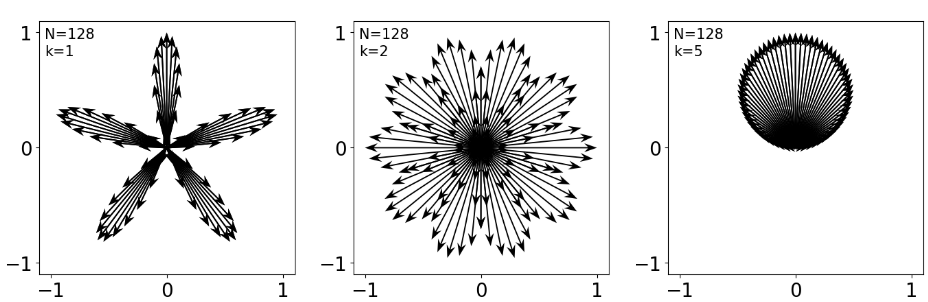
Rys. 4. Nawijanie, przykłady dla $k = 1, 3$. Wektory zsumują się do zera dla $k = 0, 2, 3, 4, 5, 6$ (dla $k = 1$ zsumują się do $(0,4)$, więc drganie o częstotliwości 1 jest drganiem bazowym badanej funkcji; natomiast $k = 7$ to patologia – zob. margines)

By lepiej wyobrazić sobie mechanizm nawijania, weźmy funkcję $\sin(5x)$ oraz ogromne N, np. 128 (zob. rys. 5). Nawinięte wektory wykreślą fantazyjne, choć zarazem bardzo regularne, kształty. Dość łatwo można wyrobić sobie intuicję, że w każdym przypadku, z wyjątkiem $k = 5$, wektory zsumują się do $(0,0)$.

Można na nawijanie spojrzeć z innej strony – jeśli wiemy, jaką funkcję nawijamy, nie musimy jej próbkować, możemy ją nawinąć w sposób ciągły. Na poniższych ilustracjach funkcja $f(x) = \sin(5x)$ nawinięta jest odpowiednio jeden i dwa razy:



Oba kształty nieprzypadkowo wyglądają bardzo podobnie do tych z rysunku 5 dla $k = 1$ i $k = 2$. Takie nawinięcie nieskończenie wielu próbek odbywa się w Ciągłej Transformacie Fouriera.

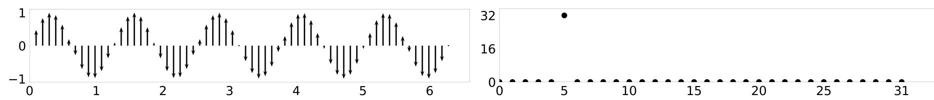


Rys. 5. Nawijanie $N = 128$ próbek funkcji $\sin(5x)$ dla $k = 1, 2, 5$

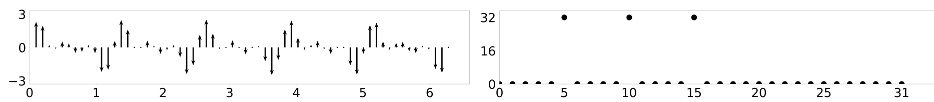
To, że nawijane próbki funkcji $\sin(lx)$ zsumują się do $(0,0)$ dla każdego $k \neq l$, to tylko połowa sukcesu. Kluczową własnością jest, że algorytm wciąż będzie działać, gdy badana funkcja ma więcej niż jedno drganie bazowe. Mówiąc bardziej formalnie, jeśli nawiniemy próbki funkcji $f(x) = \sum_{l=1}^{n} a_l \sin(lx) + b_l \cos(lx)$, to zsumują się one do $(0, 0)$ zawsze, gdy dla danego l mamy $a_l = 0$ i $b_l = 0$. Czytelnik Spostrzegawczy od razu zauważy, że wzór funkcji f jest już bardzo podobny do wzoru na szereg Fouriera…

Zgodnie z uwagą o aliasingu, dla 64 próbek policzymy 32 wartości widma.

Spójrzmy na zestawione parami $N = 64$ próbki różnych funkcji z ich widmami wyliczonymi zgodnie z algorytmem nawijania.

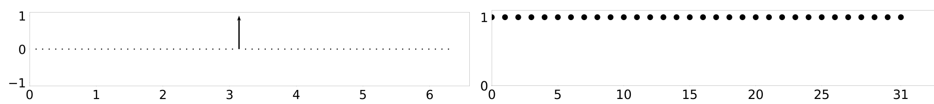


Rys. 6. Próbki $f(x) = \sin(5x)$ oraz ich widmo – jest zerem wszędzie poza $k = 5$



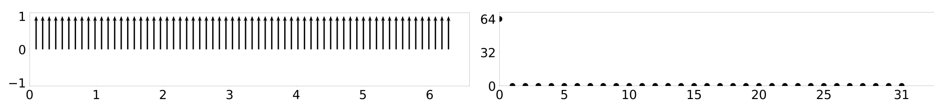
Rys. 7. Dla $f(x) = \sin(5x) + \sin(10x) + \sin(15x)$ widmo wskaże 5, 10 i 15

Przykład z punktowym skokiem (rys. 8) ma intuicyjny sens fizyczny, w praktyce bowiem takim skokiem jest każdy stuk, trzask itd. Widmo zaś elegancko tłumaczy, czemu jeśli klaśnie się przy pianinie z nietłumionymi strunami, to wszystkie mniej więcej jednakowo mocno zadźwięczą.



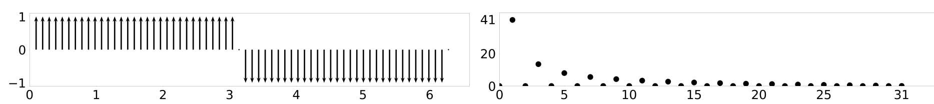
Rys. 8. Funkcja punktowo różna od 0 ma widmo jednolicie różne od 0

Tu zbliżamy się jeszcze bardziej do szeregu Fouriera (rys. 9), bo wartość widma dla $k = 0$ odpowiada stałej C we wzorze $f(x) = C + \sum_{l=1}^{n} a_l \sin(lx) + b_l \cos(lx)$.



Rys. 9. Funkcja stała ma widmo różne od 0 tylko w $k = 0$

Oczywiście nie tylko syntetyczna fala prostokątna ma w widmie same nieparzyste składowe. W naturze taką falą jest chociażby dźwięk tworzony przez drgającą kolumnę powietrza jednostronnie zamkniętą, czyli np. przez klarnet. Zwłaszcza niski rejestr tego instrumentu ma miękką, ciepłą barwę, która bierze się właśnie z wyeliminowania parzystych składowych.



Rys. 10. Próbki tzw. fali prostokątnej, czyli $f(x) = \mathrm{sgn}(\sin(x))$, dadzą widmo, w którym zapalone są składowe dla k nieparzystych

Czytelnika Uważnego zaniepokoi, że we wszystkich przykładach okresem badanej funkcji było sprytnie dobrane 2π, co więcej, po cichu założyliśmy, że ten okres jest znany. Takiego założenia w praktyce jednak na ogół nie ma – i to jest realny problem podczas przetwarzania sygnału. Rozwiązuje się go, „ściszając” krańce badanego przedziału, by zniwelować skok na granicy okresów. Mówiąc ściślej, przemnaża się próbki przez tzw. funkcję okna. Dlatego też drganie przedstawione na rysunku 1 ma małą amplitudę na swoich końcach.

Wzmianka o szybkim generowaniu widma jest doskonałą okazją, by wspomnieć, od czego *tak naprawdę* pochodzi skrót FFT: Fast Fourier Transform to nazwa algorytmu, który w czasie $O(n \log n)$ oblicza widmo zadanego sygnału. Jest szybki, bo w elegancki sposób korzysta z pewnej dość prostej cechy liczb zespolonych, aby dwukrotnie zmniejszyć rozmiar danych wejściowych

Wiemy zatem, że widmo dźwięku można wygenerować, i mniej więcej wiemy jak. Okazuje się, że po pierwsze: umiemy je wygenerować szybko, a po drugie: umiemy to widmo z powrotem odwinąć i wrócić do jego pierwotnej postaci (i to też umiemy zrobić szybko!). Mogę więc bez problemu użyć FFT, by przeanalizować widmo nagrania ze skrzypiącym krzesłem, zlokalizować częstotliwość dźwięku skrzypu i ją ordynarnie wyciąć (przy odrobinie szczęścia uda mi się nie zrobić dużej krzywdy właściwym dźwiękom piosenki). Następnie mogę z powrotem odwinąć widmo i uzyskać pierwotne nagranie, tylko już bez krzesła.

Naturalnie transformata Fouriera jest czymś dużo więcej niż tylko korektorem niedoskonałości offowych piosenek. Jest wszechobecna w cyfrowym przetwarzaniu sygnałów, zarówno elektrycznych, jak i akustycznych, jej wersja ciągła jest nieodzowna w mechanice kwantowej, pod nieco zmienioną postacią odpowiada za kompresję MP3, a nawet jest biologicznie zaimplementowana w ludzkim uchu. Jednak już sam jeden przykład wycinania skrzypiącego krzesła wystarcza, bym wciąż trwał w zachwycie, że „tak się da”.