Ćwiczenie: splot cyfrowy

1. Wstęp

Splotem cyfrowym nazywamy operację na dwóch dyskretnych wektorach *x(n)* i *y(n)*, która daje w wyniku wektor *y(n)*

$$z(n) = x(n) * y(n) = \sum_{k=-\infty}^{\infty} x(k)y(n-k)$$

Dla transformat Z ciągów obowiązuje twierdzenie o splocie, zgodnie z którym

$$Z(z) = X(z)Y(z)$$

Podstawiając $z = \exp(j\Omega)$ i obliczając w dyskretnych próbkach Ω_n otrzymuje się podobną zależność dla transformat DFT

$$Z(n) = X(n)Y(n)$$

Uwaga: z powyższej zależności wynika, że wymiar wektora X i Y musi być taki sam. Ze względu na wymiar wektora wynikowego, możemy wyróżnić dwa rodzaje splotu:

1. **Splot liniowy**: długość wektora wyjściowego jest równa $N_Z=N_X+N_Y-1$

2. **Splot cykliczny (kołowy)**: długość wektora wyjściowego jest równa $N_Z=\max(N_X,N_Y)$

Wykonanie splotu z definicji (1) jest kosztowne obliczeniowo. Przykładowo, splot dwóch wektorów o długości N wymaga N^2 mnożeń. Dlatego pakiety numeryczne (np. Matlab) obliczają splot przez DFT[1]. W tym celu obliczane jest DFT obu wektorów:

{X(n),Y(n}}=DFT{x(n),y(n)},

Mnoży się jedno przez drugie

Z(n)=X(n)Y(n)

i oblicza transformatę odwrotną

z(n)=IDFT{Z(n)}.

W takim wypadku mamy 2 x (N/2 $\log_2$ N) mnożeń co odpowiada transformacie DFT plus 4N mnożeń co odpowiada mnożeniu zespolonych wektorów o długościach N. Już dla N większych niż kilkanaście jest to spora oszczędność czasu.

[1] conv w pakiecie Matlab liczy splot przez DFT

W celu obliczenia splotu liniowego, oba wektory uzupełnia się zerami do długości $N_Z=N_X+N_Y-1$, a następnie postępuje jak wyżej. W celu obliczenia splotu kołowego wektor o mniejszej ilości próbek uzupełnia się zerami do długości wektora dłuższego i następnie postępuje jak wyżej.

2. Algorytmy sekwencjonowanego splotu

Przypuśćmy, że mamy spleść z wektorem zawierającym współczynniki filtru FIR drugi wektor, zawierający np. próbki z odbiornika software radio. Wektor próbek z kilkunastu sekund rejestracji jest tak długi, że nie można go zmieścić w pamięci operacyjnej procesora. W takim wypadku użyteczne są metody liczenia splotu, w których sygnał wczytywany jest blokami z pliku, następnie splatany i zapisywany z powrotem na dysku. Problem tego podejścia jest taki, że po spleceniu bloku o długości N z filtrem o M próbkach wypadkowy sygnał ma N+M-1 próbek a zatem wynik splotu z jednego bloku będzie zachodzić na blok następny. Dlatego tego rodzaju splot należy wykonać sprytnie, posługując się jedną z dwóch najlepiej znanych metod do tego służących: *overlap add* lub *overlap save* [1]. Metody są szczegółowo opisane w literaturze, nie będą więc tutaj omawiane. W skrócie:

- overlap add dodaje do siebie nadmiarowe próbki z bloku poprzedniego do początkowych próbek bloku natępnego

- overlap save dla każdego bloku oblicza splot cykliczny, odrzucając próbki, które pojawiają się na początku bloku

Proszę zwrócić uwagę, że jeśli długość bloku wynosi N, to krok z którym przesuwany jest bieżący blok wynosi N-M+1 a nie N.

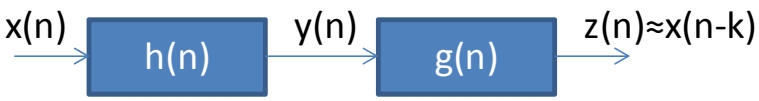
Inne przykłady zastosowania w.w. metod: filtracja cyfrowa przy pomocy OA lub OS jest szybsza niż mnożenie próbek sygnału z próbkami filtru i dodawanie, które trzeba powtarzać w każdym kroku, co odpowiada splotowi liczonemu z definicji. Aby operacja mogła jednak działać w czasie rzeczywistym, tzn. bez opóźnień, konieczny jest podział na bloki.

3. Dekonwolucja

Dekonwolucja to ważne zagadnienie, którego celem jest odwrócenie efektów splotu. Przypuśćmy, że sygnał telekomunikacyjny przechodzi przez tor transmisyjny o odpowiedzi impulsowej h(t). W systemie komunikacji cyfrowej, odpowiedź h(t) systemu zapisujemy jako h(nT)=ozn.=h(n). System opisany jest równaniem

$$y(n) = h(n) * x(n) = \sum_{k=0}^{M-1} h(k)x(n-k)$$

W wyniku przejścia przez kanał sygnał zostaje zniekształcony i nie może być odebrany poprawnie przy pomocy prostego układu decyzyjnego. Jeżeli postać funkcji h(n) jest znana, można efekt splotu odwrócić za pomocą dekonwolucji. Jest kilka(naście) metod aby tego dokonać.



Rys. 1. Dekonwolucja przez filtrację

1. Najprościej rozpisać powyższy układ w sposób rekurencyjny, tj.

$$y(0) = h(0)x(0) => x(0) = y(0)/h(0)$$

$$y(1) = h(0)x(1) + h(1)x(0) => x(1) = \frac{y(1) - \frac{h(1)y(0)}{h(0)}}{h(0)}$$

itd. Jest to jednak sposób niewydolny obliczeniowo a w dodatku zniweczyć może go zerowy współczynnik *h(n)*.

2. Innym sposobem jest dekonwolucja w dziedzinie częstotliwości. W tym wypadku szukamy filtru *g(n)* takiego, że $h(n) * g(n) = \delta(n-k)$, gdzie *k* jest arbitralnie przyjętym opóźnieniem. Oczywistym jest, że $G(n) = \frac{1}{H(n)} \exp(-j2\pi nk)$. Współczynniki filtru wynoszą zatem *g(n)*=IDFT(*G(n)*). Ponownie, metoda zawodzi, jeżeli jeden ze współczynników *H(n)* jest równy 0. Można jednak temu do pewnego stopnia zaradzic, uzupełniając *g(n)* zerami.

3. Kolejnym sposób otrzymujemy rozpisując równanie

$$y(n) = h(n) * x(n) = \sum_{k=0}^{M-1} x(k)h(n-k)$$

w postaci macierzowej, tj.

$$\begin{pmatrix} y(0) \\ y(1) \\ \vdots \end{pmatrix} = \begin{pmatrix} h(0) & 0 & \dots \\ h(1) & h(0) & 0 \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} x(0) \\ x(1) \\ \vdots \end{pmatrix}$$

czyli **y**=**Hx**. Szukamy takiej macierzy G, że

z=Gy=GHx≈I

gdzie **I** jest macierzą jednostkową. Stąd wyznaczamy **G**=$\mathbf{H}^{-1}$, i jako wsp. filtru dekonwolucji wybieramy jedną ze środkowych kolumn **G**. Jeżeli (co ma zwykle miejsce) macierz **H** nie jest macierzą kwadratową, korzystamy z metody najmniejszych kwadratów, które określa poszukiwaną macierz zgodnie z poniższym wzorem (pseudoinwersja Moore'a-Penrose'a) [2].

G=$(\mathbf{HH}^T)^{-1}\mathbf{H}^T$

Zadania do przygotowania w domu:

1. Oblicz ile operacji wymaga splot z definicji wektorów o długościach N. Wsk. szereg arytmetyczny.

2. Oblicz splot dwóch wektorów dyskretnych przez transformatę Z

3. Zapoznaj się z funkcją conv matlaba

Literatura:

1. T. Zieliński, Cyfrowe przetwarzanie sygnałów, pkt. 13.6, algorytmy sekwencjonowanego splotu sygnałów dyskretnych, str. 372

2. https://en.wikipedia.org/wiki/Moore%E2%80%93Penrose_pseudoinverse