DSP - Cyfrowe przetwarzanie sygnałów
Laboratorium 1 – Próbkowanie, aliasing, rekonstrukcja

Jakub Moroń, Paweł Hottowy

Utworzono: Marzec 2022

Ostatnia zmiana: 17 marca 2023

Zdania oznaczone (*) zawierają elementy do zamieszczenia w raporcie.

Spis treści

1 Próbkowanie

1.1 Podstawy próbkowania

Skrypt `Zad1_1_probkowanie_podstawy`.

- Wykonaj skrypt. Przedyskutuj (z prowadzącym i uczestnikami laboratorium) co zostało przedstawione na wykresie?
- Wykonaj drugą część skryptu (usuń `return` z linii 40.). Drugi wykres przedstawi sygnał o dwukrotnie wyższej częstotliwości niż pierwszy, spróbkowany z dwukrotnie wyższą częstotliwością niż pierwszy.
- Przedyskutuj wynik, w szczególności zwracając uwagę na dolne wykresy (czy się różnią, a jeśli nie to dlaczego? Przecież w drugim przypadku częstotliwość sygnału jest dwa razy większa?).

1.2 Próbkowanie dźwięku

Skrypt `Zad1_2_probkowanie_dzwieku`.

- Wykonaj skrypt. Odsłuchaj dźwięk (jest to dźwięk A3 pianina).
- Przekłam częstotliwość próbkowania - podaj fałszywą wartość (2*Fsmp oraz Fsmp/2, możesz też wypróbować inne) jako argument funkcji odtwarzającej dźwięk.
- Przedyskutuj co się stało z tonem (wysokością) dźwięku w obu przypadkach? Wytłumacz dlaczego?
- Przedyskutuj jak długo trwał odtwarzany dźwięk w obu przypadkach? Wytłumacz dlaczego?

2 Aliasing

2.1 Podstawy(*)

Skrypt `Zad2_1_aliasing_podstawy`.

- Wykonaj skrypt. Czy wynik (wykresy i wartości wypisane w Command Window) zgadzają się z oczekiwaniami?
- Uzupełnij podpisy osi na wykresach.

1. Niepewność co do częstotliwości próbkowania sygnału
   a) Wykonaj dodatkowo skrypt dla częstotliwości sygnału Fsig = (0.1Fsmp + Fsmp) oraz Fsig = (0.1Fsmp + 2Fsmp).
   b) Zanotuj częstotliwość sygnału odzyskane z próbek. Zastanów się czy zgadza się to z oczekiwaniami?
2. Aliasing
   a) Wykonaj skrypt dla częstotliwości sygnału Fsig w przedziale 0.6–0.9 Fsmp, z krokiem 0.1 Fsmp.

b) Zanotuj częstotliwości sygnału odzyskane z próbek. Zastanów się czy zgadza się to z oczekiwaniami?
3. Wykonaj skrypt dla Fsig = 0.5 Fsmp. Przedyskutuj co wyszło, a co powinno było wyjść?

Raport

1. Niepewność co do częstotliwości próbkowania sygnału
   a) Zamieść wykres dla jednego, wybranego przypadku.
   b) Przedstaw częstotliwości sygnału odzyskane z próbek dla obu przypadków i pierwszego wykonania skryptu z `FsigRatio = 0.1`. Skomentuj wynik.
   c) Oblicz ile wyniosłaby częstotliwość sygnału odzyskana z próbek gdyby sygnał o częstotliwości 4760 Hz poddać próbkowaniu z częstotliwością 1500 Hz.
2. Aliasing
   a) Przedstaw wykres dla jednej z wybranych wartości `FsigRatio`.
   b) Przedstaw częstotliwość sygnału odzyskana z próbek dla wybranego przypadku i porównaj ją z teorią.

2.2 Aliasing audio

Skrypt `Zad2_2_aliasing_audio`.

- Wykonaj skrypt, odsłuchaj oryginalny dźwięk o częstotliwości 200Hz.
- **Uwaga:** w przeciwieństwie do dźwięku pianina, sztucznie wytworzony dźwięk w postaci pojedynczego sinusa może być nieprzyjemny do odsłuchu. Zacznij od niskiej wartości amplitudy `Amp` i zwiększ ją jedynie, gdy nie będziesz dobrze słyszeć dźwięku.

1. Niepewność co do częstotliwości próbkowania sygnału: Wygeneruj drugi dźwięk o częstotliwości 2200Hz. Odsłuchaj, jeden po drugim, oba dźwięki.
2. Przedyskutuj wnioski z odsłuchu tonów 200 Hz i 2200 Hz – co się stało i dlaczego?
3. Aliasing:
   a) Wytwórz sygnał sinusoidalny, który będzie zmieniał częstotliwość od 100 Hz do 2 kHz (z krokiem 100 Hz) co 2 sekundy.
   b) Stwórz z tego jeden długi sygnał i odsłuchaj. Zaobserwuj zmiany częstotliwości słyszalnego sygnału.
   c) Przedyskutuj wrażenia z odsłuchu – co się stało i dlaczego?

2.3 Prosty aliasing obrazu

Skrypt `Zad2_3_prosty_aliasing_obrazu`.

1. Wykonaj skrypt dla `Rep = 40`. Przedyskutuj:
   - Ile wynosi okres sinusa wyrażony w pikselach? Jak go powiązać z Fsig/Fsmp dla sygnału próbkowanego w czasie?

- Co dokładnie jest na wykresach (2) oraz (4) i dlaczego wydaje się że downsampling nie wpłynął w żaden sposób na sygnał?
- Usuń komentarz z linii 22 - czy teraz widać zmianę częstotliwości?

2. Wykonaj skrypt dla `Rep = 160`. Przedyskutuj wyniki.

2.4 Aliasing obrazu na przykładzie okręgów

Skrypt `Zad2_4_aliasing_obrazu`.

- Wykonaj skrypt. Początkowo `Rep = 20`. Czy spodziewasz się aliasingu? Czy widzisz aliasing w obrazie po downsamplingu?
- Przeanalizuj wykresy (2) oraz (4). Czy rozumiesz co przedstawiają? Jeśli nie, przedyskutuj to zagadnienie.
- Wykonaj ponownie skrypt ustawiając `Rep = 75`. Spróbuj odpowiedzieć na pytania:
  1. Co się stało z obrazem do downsamplingu? Czy nadal przedstawia koncentryczne okręgi?
  2. Czy potrafisz wytłumaczyć zjawisko analizując wykresy (2) oraz (4), czyli przedstawiające przebieg przecinający obrazek dokładnie w połowie?

  Wykres (4) wskazuje że zaszedł aliasing – widzimy na nim sinusa o częstotliwości mniejszej niż na wykresie (2). Skoro obrazek przedstawia sinus „obrócony” wokół środka obrazu (jak figura obrotowa), to na podstawie wykresu (4) można by przypuszczać że obraz po downsamplingu będzie przedstawiał nadal koncentryczne okręgi, tylko rozłożone rzadziej (z mniejszą częstotliwością). Dlaczego tak nie jest?
- Wykonaj ponownie skrypt dla `Rep = 20`, ale tym razem usuwając (komentując) `return` w linii 47. Dzięki temu skrypt wykona dodatkowy wykres przedstawiający jedną linię obrazu, ale tym razem nie przecinającą środka, zaś położoną w 1/4 wysokości obrazu.
- Przeanalizuj nowy wykres (5) - czy w tej linii nadal mamy sinusa o stałej częstotliwości, jak w przypadku linii przechodzącej przez środek obrazu?
- Wykonaj ponownie cały skrypt dla `Rep = 75`. Zwróć szczególną uwagę na wykres (5). Co się stało? Czy zaszedł aliasing, a jeśli tak to gdzie - dla całej zawartości linii obrazu, czy tylko dla pewnego obszaru? Przedyskutuj wynik i efekt, który za sobą niesie.

2.5 Aliasing obrazu - przykład z wykładu

Skrypt `Zad2_5_aliasing_obrazu_wykład`.

- Wykonaj skrypt i obejrzyj wyniki.
- Przedyskutuj wynik pracy filtru anty-aliasingowego (np. czy pomógł usunąć efekty w pełni?).

2.6 Aliasing obrazu - zdjęcie(*)

Skrypt `Zad2_6_aliasing_zdjecie`.

- Wykonaj skrypt i obejrzyj wyniki.
- Czy widzisz efekt aliasingu? Gdzie i w których miejscach się objawia? Przedyskutuj te zagadnienia.
- Usuń `return` w linii 24. Skrypt wykona downsampling z zastosowaniem filtru anty-aliasingowego.
- Przedyskutuj, czy nadal widzisz efekty aliasingu tam, gdzie występowały poprzednio (bez filtru?). Czy filtr wpłynął w jakiś inny sposób na zdjęcia?
- **Do domu / na następne zajęcia:** znajdź (lub zrób) zdjęcie które pokaże zjawisko aliasingu po downsamplingu (zmniejszeniu rozmiaru).

Raport

1. Umieść w raporcie swoje wybrane zdjęcie – w rozmiarze oryginalnym oraz pomniejszone, w wersji bez oraz z filtrem anty-aliasingowym.
2. Zaznacz obszary na których widać efekt aliasingu. Uzasadnij dlaczego akurat w tych obszarach ten efekt wystąpił.
3. Skomentuj efektywność filtrowania anty-aliasingowego - czy w tych samych obszarach efekt aliasingu nadal występuje?
4. Porównaj oba zdjęcia. Czy poza usunięciem aliasingu widzisz różnice między tymi dwoma obrazami? Jeśli tak, to jakie?

2.7 Downsapling dźwięku - ograniczenie pasma i aliasing(*)

Skrypt `Zad2_7_downsampling_dźwięk`.

- Wykonaj skrypt, odsłuchaj oryginalny dźwięk pianina i wynik downsampling-u. Początkowo zmienna `DS_ratio`, czyli krotność zmniejszenia częstotliwości próbkowania jest ustawionan na 2. Zmniejszamy więc częstotliwość próbkowania i pasmo dwukrotnie. Czy wpłynęło to jakkolwiek na dźwięk? Spróbuj uzasadnić wynik wiedząc, że pasmo odtwarzanego dźwięku kończy się w okolicy 6 kHz.
- Wykonaj skrypt dla `DS_ratio` = 2, 4, 8, 16 oraz 32. Jak zmienia się dźwięk pianina? W szczególności - czy słyszysz coś „dziwnego” dla `DS_ratio` = 16?
- Wykonaj skrypt powtórnie dla „ScottHolmesMusic_EpicCinematic.mp3”. Jest to fragment utworu skomponowanego w stylu „epickiej” muzyki filmowej. Wypróbuj różne wartości DS_ratio, ale zacznij od 2 i 32. Jak wpływają one na oryginalny utwór?

Raport

1. Spróbuj wytłumaczyć efekt dla pliku ”a3.wav” oraz DS_ratio = 16 lub 32 wiedząc, że pasmo odtwarzanego dźwięku kończy się w okolicy 6 kHz.

3 Rekonstrukcja

3.1 Podstawy

Skrypt `Zad3_1_rekonstrukcja_podstawy`.

Na wykładzie przedstawiony został wzór na rekonstrukcję sygnału w postaci splotu próbek i funkcji sinc. W tym wzorze sumowanie przebiega od $-\infty$ do $+\infty$, co dla rzeczywistego, skończonego ciągu próbek o długości N nie jest możliwe do wykonania. Rekonstrukcja dla skończonego ciągu próbek X o długości N, pobranych z okresem próbkowania T_{smp}, dana jest więc wzorem

$$x(t) = \sum_{n=0}^{N-1} X(n \cdot T_{smp}) \cdot \operatorname{sinc}\left(\frac{t - n \cdot T_{smp}}{T_{smp}}\right) \tag{1}$$

- Wykonaj skrypt
  Uwaga Skrypt realizuje „animację” rekonstrukcji. Po wyświetleniu wykresów (1) i (2) skrypt się zatrzyma i będzie oczekiwał na naciśnięcie dowolnego klawisza w „Command Window,. Po naciśnięciu, wykona krok rekonstrukcji (sumy splotu próbek i funkcji sinc) i znów zatrzyma się w oczekiwaniu na naciśnięcie klawisza. Po wykonaniu `pauseTill = 10` kroków (linia 5. skryptu), skrypt będzie automatycznie kontynuował animację obliczając kolejne kroki co sekundę.
  Skrypt w dowolnym momencie można przerwać naciskając CTRL-C w „Command Window”.
- Oblicz błąd rekonstrukcji – różnicę między sygnałem oryginalnym `realSignal` oraz wynikiem rekonstrukcji `xt`. Błąd przedstaw na wykresie.
  Możesz pominąć ”animowanie” rekonstrukcji komentując linie 78–84 w skrypcie.

3.2 Rekonstrukcja – okna(*)

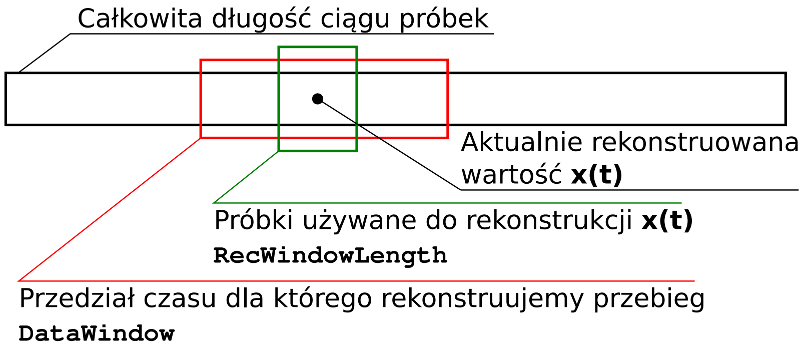
Skrypt `Zad3_2_rekonstrukcja_w_oknie`.

W tym zadaniu, po raz pierwszy, przechodzimy do typowej dla DSP reprezentacji częstotliwości i czasu:

- Ustalamy częstotliwość próbkowania jako $F_{smp} = 1$ (uwaga - bezwymiarowe) oraz, co za tym idzie:
- Okres próbkowania $T_{smp} = 1/F_{smp} = 1 =$ indeks próbki.
- Częstotliwość sygnału podajemy wówczas jako ułamek (też bezwymiarowy), w domyśle - ułamek częstotliwości próbkowania F_{smp}.

Na podstawie poprzedniego przykładu widać, że błąd rekonstrukcji narasta wraz ze zbliżaniem się do końców ciągu próbek. Z tego powodu wykonamy rekonstrukcję używając tylko fragmentu ciągu próbek. Innymi słowy nałożymy na ciąg próbek okno – zobaczymy to, co jest w oknie, a to, co „wystaje” poza, pozostanie ukryte.
Ponadto przetestujemy tutaj wpływ długości ciągu na precyzję rekonstrukcji. W tym celu będziemy manipulować ilością próbek uwzględnianą przy rekonstrukcji, czyli N ze wzoru 1.

Sygnał będziemy rekonstruować jedynie dla przedziału próbek określonego przez `DataWindow`, czyli „z daleka” od brzegów ciągu próbek.
Do rekonstrukcji użyjemy `RecWindowLength` próbek. Dla symetrii zmienimy granice sumowania ze wzoru 1 z $[0, N-1]$ na [`-RecWindowLength/2`, `RecWindowLength/2-1`].
Okna `DataWindow` i `RecWindowLength` pokazano schematycznie, względem ciągu próbek, na rysunku 1.



Rysunek 1: Schematyczne przedstawienie okien używanych do rekonstrukcji

W poprzednim zadaniu, dzięki Matlab'owi, schowaliśmy fakt że $x(t)$ nie jest tak na prawdę ciągłe (a dokładnie - czas nie jest ciągły), a jest reprezentowany quazi-ciągłym wektorem. Dzięki temu obliczyliśmy jawnie (dla każdego n) składniki sumy w wyrażeniu 1 licząc je, niejawnie, dla wszystkich wartości t na raz (gdyż takie podejście umożliwiło animację).
W tym zadaniu odwracamy sytuację - jawnie mówimy że zrekonstruowany sygnał będzie nadal dyskretny w czasie (tylko o `Oversampling` „gęstszy” niż próbki). Innymi słowy $x(t)$ zamieni się w $x(t_j)$, gdzie t_j należy do zbioru `recTime`, czyli wartości z przedziału `DataWindow`, z krokiem 1 / `Oversampling`.
Ponieważ chcemy zastosować okno, wykonamy pętlę jawnie po wszystkich wartościach j, natomiast sumę po n wykonamy funkcjami Matlab'a (oczywiście można by też napisać jawnie pętlę w pętli...).
Uwzględniając wszystko powyższe, wzór na rekonstrukcję będzie dany jako

$$x(t_j) = \sum_{n=-RecWindowLength/2}^{RecWindowLength/2-1} X(n) \cdot \operatorname{sinc}(t_j - n) \tag{2}$$

Uzywamy tutaj tzw. okna prostokątnego. Dokładność rekonstrukcji można znacząco podnieść, używając „lepszego” okna, np. Blackman-a. Ale ponieważ (na razie) o funkcjach okien nic nie wiemy, zapamiętajmy tylko fakt że lepsza funkcja okna dałaby lepsze rezultaty, i wrócimy do tego tematu gdy dowiemy się czym te funkcje okien są...

Funkcja `WindowReconstruction` zwraca liczbę reprezenutjącą dokładność interpolacji, zdefiniowaną jako stosunek wartości średnio-kwadratowej (RMS) tego błędu (`recError`) do wartości średnio-kwadratowej idealnego sygnału (`realSignal`).

- Dla ustalonej częstotliwości `Fsig = 0.1` wykonaj skrypt dla różnych długości okna rekonstrukcji `RecWindowLength`: 20, 64, 200, 640, 2000. Wykonaj wykres błedu rekonstrukcji w funkcji długości okna.
- Dla największej długości okna `RecWindowLength = 2000` powtórz szacowanie dokładności dla różnych częstotliwości sygnału `Fsig`: 0.2, 0.3, 0.4, 0.49. Wykonaj wykres w funkcji `Fsig`.

Raport

1. Umieść wykres błędu rekonstrukcji w funkcji długości okna. Skomentuj wynik.
2. Umieść wykres błędu rekonstrukcji w funkcji częstotliwości sygnału. Skomentuj wynik.
3. *Podpowiedź: możesz odwołać się do wyniku z poprzedniego zadania (3.1)*

4 Bandpass sampling, modulacja

Wstęp

W celu przedstawienia zastosowania bandpass sampling-u w zastosowaniu praktycznym, a nie na teoretycznym, akademickim przykładzie, potrzebny jest sygnał spełniający założenia (patrz wykład), czyli mający relatywnie wąskie pasmo położone wokół (relatywnie) wysokiej częstotliwości środkowej. Taką „dziwną” charakterystykę częstotliwościową mają wszystkie sygnały wykorzystywane w transmisji bezprzewodowej, niezależnie od tego czy rozważymy radio, WiFi czy Bluetooth, przez co technika bandpass sampling-u jest najczęściej wykorzystywana w odbiornikach takich transmisji.

Z transmisja bezprzewodową łączy się nierozerwalnie zagadnienie modulacji (które niestety nie było przedstawione na wykładzie). W najprostszym przypadku – radia – potrzebujemy przesłać sygnał audio o paśmie 20 HZ – 20 kHz. Ze względów praktycznych, o których porozmawiamy na zajęciach, nie da się wprost wypromieniować fali elektromagnetycznej o takich częstotliwościach „w eter” (częstotliwości 20 Hz odpowiada fala o długości blisko 15 000 km...). Ze względu na charakterystykę propagacji fal radiowych (poza zakresem tego przedmiotu) współcześnie używamy częstotliwości rzędu 10–100 MHz (długość fali $\sim$ 30–3 m). Potrzebna jest więc metoda na umieszczenie („zakodowanie”) dźwięku na takiej częstotliwości. (Na marginesie – WiFi używa częstotliwości 2.4 GHz oraz 5 GHz, Bluetooth – 2.4 GHz, telefonia komórkowa – od 450 MHz do 1.9 GHz).

Współczesne radio używa tzw. modulacji częstotliwościowej FM (ang. Frequency Modulation). Telefonia komórkowa, WiFi – jeszcze bardziej zaawansowanych modulacji: np. fazowej, amplitudowo-fazowej QAM (ang. Quadrature Amplitude Modulation), itd. Natomiast najprostszą modulacją jest modulacja amplitudowa AM (ang. Amplitude

Modulation), używana jeszcze kilkadziesiąt lat temu także na potrzeby radia (obecnie prawie zapomniana), oraz wciąż używana w krótkofalarstwie i radiokomunikacji. Modulacja amplitudowa AM jest nie tylko najłatwiejsza do zrozumienia, ale też do demodulacji (czyli odzyskania oryginalnego sygnału), gdyż wystarczy jedynie bandpass sampling bez dodatkowych elementów. Dlatego zagadnienie modulacji omówimy na razie tylko na tym przykładzie. Demodulacja FM wymaga dodatkowo filtracji zdemodulowanego sygnału, więc wrócimy do niej na koniec semestru.

Modulacja amplitudowa AM

Wprowadźmy pojęcie „nośnej" (ang. carrier) – jest to sygnał (typowo – sinus) o docelowej częstotliwości jaką chcemy nadać bezprzewodowo. W przypadku radia może być to, przykładowo, 100 MHz. Sygnał audio, dla uproszczenia, ograniczymy do monotonicznego sinusa o częstotliwości F_{sig}.

Modulacji AM dokonujemy przez przemnożenie obu fal przez siebie:

$$AM = \sin(\omega_{carrier}t) \cdot \sin(\omega_{sig}t) \tag{3}$$

gdzie $\omega = 2\pi f$.
Z tożsamości trygonometrycznych wynika że:

$$AM = \sin(\omega_{carrier}t) \cdot \sin(\omega_{sig}t) = \frac{1}{2}\left\{\cos[(\omega_{carrier} - \omega_{sig})t] - \cos[(\omega_{carrier} + \omega_{sig})t]\right\} \tag{4}$$

Innymi słowy otrzymujemy dwa sygnały - jeden o częstotliwości danej sumą częstotliwości nośnej i częstotliwości sygnału, i drugi, o częstotliwości danej ich różnicą. Sygnały te nazywane są "wstęgami". Mówimy o wstędze lewej (odpowiadającej różnicy częstotliwości), jako położonej w widmie na lewo od częstotliwości nośnej, i wstędze prawej (odpowiadającej sumie częstotliwości).

Widać że obie wstęgi (suma i różnica częstotliwości) niosą tą samą informację, czesto więc wycina się jedną z nich aby nie nadawać nadmiarowej informacji. Taką modulację nazywa się modulacją SSB (ang. Single Side Band) – jednowstęgową modulacją amplitudową. Sygnał niosący wyłącznie wstęgę prawą będzie więc dany równaniem

$$SSB = -\frac{1}{2}\cos[(\omega_{carrier} + \omega_{sig})t] \tag{5}$$

Ze względów praktycznych modulacji AM dokonuje się w sposób nieco bardziej złożony. Z powyższych wzorów widać, że gdy nie ma sygnału (przerwa w piosence lub między zdaniami w mowie), cały zmodulowany sygnał AM lub SSB znika (jest równy zero). Co więcej w zmodulowanym sygnale nie ma w ogóle częstotliwości nośnej, do której normalnie stroimy radio. Jest to bardzo niedogodne dla odbiornika, gdyż traci on możliwość śledzenia częstotliwości fali nośnej (a robi to obecnie praktycznie każdy odbiornik, w szczególności samochodowy). Dlatego modulacji AM dokonujemy tak, aby zawsze w sygnale znajdowałą się oryginalna częstotliwość nośna, niezależnie od tego co aktulanie dzieje się z sygnałem audio. Wprowadźmy dwa parametry – współczynnik podtrzymania fali nośniej C i współczynnik głebokości modulacji M. Wówczas:

$$AM = \sin(\omega_{carrier} t) \cdot [C + M \sin(\omega_{sig} t)] \tag{6}$$

Z tożsamości trygonometrycznych otrzymujemy wówczas

$$AM = C \sin(\omega_{carrier} t) + \frac{M}{2} \{\cos[(\omega_{carrier} - \omega_{sig})t] - \cos[(\omega_{carrier} + \omega_{sig})t]\} \tag{7}$$

oraz, dla modulacji SSB (bez wytłumionej fali nośnej)

$$SSB = C \sin(\omega_{carrier} t) - \frac{M}{2} \cos[(\omega_{carrier} + \omega_{sig})t] \tag{8}$$

Demodulacja AM

Klasycznie demodulacji AM dokonujemy „analogowo” poprzez ponowne przemnożenie zmodulowanego sygnału przez czestotliwość nośną:

$$Demod = SSB \cdot \sin(\omega_{carrier} t) \tag{9}$$

skąd

$$\begin{aligned} Demod = {} & C \sin(\omega_{carrier} t) \sin(\omega_{carrier} t) - \frac{M}{2} \cos[(\omega_{carrier} + \omega_{sig})t] \sin(\omega_{carrier} t) = \qquad (10) \\ & = \frac{1}{2} \{\cos[(\omega_{carrier} - \omega_{carrier})t] - \cos[(\omega_{carrier} + \omega_{carrier})t]\} - \\ & - \frac{M}{2} \frac{1}{2} \{\sin[(\omega_{carrier} + \omega_{sig} + \omega_{carrier})t] + \sin[(\omega_{carrier} - \omega_{sig} - \omega_{carrier})t]\} = \\ & = \frac{1}{2} [1 - \cos(2\omega_{carrier} t)] - \frac{M}{4} \sin[(2\omega_{carrier} + \omega_{sig})t] - \frac{M}{4} \sin(-\omega_{sig} t) = \\ & = \frac{1}{2} + \frac{M}{4} \sin(\omega_{sig} t) - \frac{1}{2} \cos(2\omega_{carrier} t) - \frac{M}{4} \sin[(2\omega_{carrier} + \omega_{sig})t] \end{aligned}$$

Dwa ostatnie składniki zdemodulowanego sygnału są następnie odfiltrowywane (bardzo łatwo, gdyż ich częstotliwości są rzędu dwóch częstotliwości nośnej, np. ∼200 MHz, „bardzo daleko” od pasma dźwięku), dając finalnie oryginalny sygnał.

Demodulacja AM poprzez downsampling

Procedura modulacji przesuwa efektywnie sygnał, którego oryginalne częstotliwości zawierały się w paśmie

$$f_{sig}^{org} \in [0, f_{bandwidth}] \tag{11}$$

w pasmo określone częstotliwością nośną $f_{carrier}$, czyli (dla modulacji SSB z prawą wstęgą)

$$f_{sig}^{mod} \in [f_{carrier}, f_{carrier} + f_{bandwidth}] \tag{12}$$

Demodulacja polega na przesunięciu sygnału odebranego z anteny o $f_{carrier}$ „w lewo”, czyli do oryginalnego pasma.

Możemy tu użyć własności niepewności względem częstotliwości próbkowania – jeżeli dobierzemy częstotliwość próbkowania $f_{smp} < f_{carrier}$ tak, aby

$$f_{carrier} \% f_{smp} = 0 \tag{13}$$

uzyskamy dzięki temu przesunięcie pasma zmodulowanego sygnału do jego oryginalnego zakresu $f_{sig}^{org} \in [0, f_{bandwidth}]$, czyli dokonamy demodulacji. Oczywiście pasmo sygnału musi spełniać warunek Nyquist'a, czyli

$$f_{smp} \geqslant 2 \cdot f_{bandwidth} \tag{14}$$

Warunek 13 można zapisać jako

$$\frac{f_{carrier}}{f_{smp}} = k \qquad k \in \mathbb{N} \tag{15}$$

co w połączeniu z 14 daje warunek

$$k \leqslant \frac{f_{carrier}}{2 \cdot f_{bandwidth}} \tag{16}$$

Ze względów praktycznych chcemy wybrać najniższą możliwą częstotliwość próbkowania f_{smp}, czyli wybrać największą możliwą całkowitą wartość k spełniającą relację 16. Ponadto chcemy najczęściej by częstotliwość próbkowania f_{smp} była w miarę „równa”, czyli wynosiła np. 10.5 kHz, a nie 9 i 1/3 kHz. W przypadku laboratorium, ze względu na to jak jest implementowany w Matlab'ie proces próbkowania, będziemy starać się wybrać „równy” okres próbkowania $T_{smp} = 1/f_{smp}$.

W wyniku demodulacji poprzez bandpass sampling otrzymamy:

$$\begin{aligned} Demod_{bs} = C \sin(\omega_{carrier} \% \omega_{smp} \cdot t) - \frac{M}{2} \cos[(\omega_{carrier} + \omega_{sig}) \% \omega_{smp} \cdot t] = \\ = |\omega_{carrier} \% \omega_{smp} = 0; \qquad \omega_{sig} \% \omega_{smp} = \omega_{sig}| = \\ = C \sin(0) - \frac{M}{2} \cos(\omega_{sig} t) = -\frac{M}{2} \sin\left(\frac{\pi}{2} - \omega_{sig} t\right) = \frac{M}{2} \sin\left(\omega_{sig} t - \frac{\pi}{2}\right) \end{aligned} \tag{17}$$

gdyż $\cos(x) = \sin\left(\frac{\pi}{2} - x\right)$. Czyli w wyniku demodulacji z użyciem bandpass sampling otrzymamy oryginalny sygnał przesunięty w fazie o $\frac{\pi}{2}$. Szczęśliwie dla nas dźwięk jest niewrażliwy na przesunięcie fazowe (więcej o tym w dalszym ciągu semestru), więc nie wpłynie to na jakość sygnału audio zdemodulowanego tą metodą. W ogólności trzeba jednak pamiętać o tym przesunięciu fazowym.

4.1 Podstawy modulacji amplitudowej

Skrypt `Zad4_1_modulacja_am_podstawy`.

Skrypt ma cztery wersje, wybierane zmienną `Wersja` (linia 7.).

- Wykonaj skrypt w wersji `Wersja = 1`, przedyskutuj jego działanie i zawartość wykresów tworzonych przez skrypt.
- Zmień parametry modulacji na rzeczywisty przypadek - ustaw:
  - `ModulationDepth = 0.3`
  - `CarrierHold = 1`

  Wykonaj skrypt i przedyskutuj wyniki.
- Wykonaj skrypt w wersji `Wersja = 2`, gdzie sygnał jest sumą dwóch sinusów o częstotliwościach `Fsig` oraz `Fsig2`. Przedyskutuj wyniki.
- Wykonaj skrypt w wersji `Wersja = 3`. Skrypt wykona dodatkowo obcięcie lewej wstęgi, czyli przetworzy sygnał o modulacji AM na modulację SSB. Przedyskutuj zawartość ostatniego wykresu (w porównaniu do widma modulacji AM).
- Na koniec ustaw wersję `Wersja = 4` co przygotuje skrypt dla następnego zadania.

4.2 Bandpass sampling - demodulacja sygnału modulowanego amplitudowo(*)

Skrypt `Zad4_2_bandpass_sampling`.

Uwaga – zadbaj by w skrypcie `Zad4_1_modulacja_am_podstawy` zmienna `Wersja` była ustawiona na 4.

- Na postawie zależności 16 oraz 15 dobierz optymalną częstotliwość próbkowania dla bandpass samplingu i wpisz ją do zmiennej `Fsmp`. Usuń poprzedzający ją return.
- Wykonaj skrypt, przedyskutuj wyniki, czyli wykresy widma sygnału po demodulacji i jego postaci czasowej w porównaniu z oryginalnym sygnalem.
- Zmień wersję skryptu `Zad4_1_modulacja_am_podstawy` na `Wersja = 4` . Wykonaj ten skrypt ponownie i sprawdź jak tym razem wygląda przebieg czasowy sygnału po demodulaji. Przeduskutuj wyniki.

Raport

1. Przedstaw wybraną wartość Fsmp i uzasadnij ten wybór.
2. Zamieść wykres przebiegu czasowego sygnału po demodulacji. Skomentuj wynik: zgodność sygnału po demodulacji w odniesieniu do oryginalnego sygnału.

4.3 Zastosowanie bandpass sampling'u

Skrypt `Zad4_3_zastosowanie_bandpass_sampling`.

Skrypt wczytuje plik `media/ssb_signal.mat` zawierający zmienne:

- `ssb_signal` – tablica pseudo-ciągłego sygnału modulacji SSB z nośną i prawą wstęgą – reprezentuje sygnał „odebrany z anteny";
- `Tstep` – krok pseudo-ciągłego czasu dla sygnału.

- Wykonaj skrypt. Przeanalizuj i przedyskutuj wykres widma i spróbuj odczytać z niego wartości:
  - częstotliwości nośnej `Fcarrier`
  - pasma sygnału `Bandwidth`

  Ustaw wartości w odpowiednich zmiennych (linie 30. i 31.)
- Na postawie zależności 16 oraz 15 dobierz optymalną częstotliwość próbkowania dla bandpass samplingu i wpisz ją do zmiennej `Fsmp`.
- Usuń return z linii 29. Wykonaj skrypt, obejrzyj i przedyskutuj wykresy sygnału po demodulacji.
- Odkomentuj funkcję `PlaySound` na końcu sktyptu. Co zostało ukryte w zmodulowanym sygnale?

5 Szum kwantyzacji

5.1 Szum kwantyzacji w matematycznym modelu ADC

Skrypt `Zad5_1_szum_kwantyzacji_modelowe_ADC`.

Informacje teoretyczne znajdziesz w wykładze W3, slajdy 15–16
Wykonaj skrypt. Następnie:

- Oblicz ile kodów (wartości) zwraca przetwornik? Czy ta liczba zgadza się z rozdzielczością przetwornika (zmienna `Bits`)?
- Odczytaj z wykresu (albo wypisz za pomocą funkcji `min`, `max`, minimalną i maksymalną wartość błedu kwantyzacji wyrażonego w LSB. Czy te wartości zgadzaja się z teorią?
- Uzupełnij obliczenia w sekcji „Stosunek sygnału do szumu", czyli oblicz teoretyczną wartość RMS błędu kwantyzacji, stosunek sygnału do szumu SNR i wynikającą z niego efektywną liczbę bitów przetwornika ENOB
- Porównaj teoretyczną i zmierzoną wartość RMS błędu kwantyzacji. Porównaj teoretyczną rozdzielczość przetwornika (zmienna Bits) z wynikiem obliczenia ENOB.

5.2 Szum kwantyzacji w rzeczywistym ADC(*)

Skrypt `Zad5_2_szum_kwantyzacji_rzeczywiste_ADC`.

- Uzupełnij sekcję „Stosunek sygnału do szumu" tak samo jak w zadaniu 5.1.
- Wykonaj skrypt, następnie:

  - Zwróć uwagę na przesunięcie sygnału na dodatnie wartości napięć w stosunku do sygnału z zadania 5.1.
  - Oblicz ile kodów (wartości) zwraca przetwornik? Czy ta liczba zgadza się z tym razem rozdzielczością przetwornika (zmienna `Bits`)?
  - Porównaj wykresy błędu kwantyzacji w stosunku do zadania 5.1. Czy widzisz istotne zmiany?
- Przestaw zmienne `showPlots` i `printValues` na zero.
- Zamknij sekcje „Oblicznie błędu kwantyzacji" i „Stosunek sygnału do szumu" w pętlę tak, aby zebrać następujące dane:
  - Zmierz zależność ENOB od Vref. Ustaw Vref = [0.001 0.1 1 10 100]. Czy ENOB zależy od VREF?
  - Zmierz zależność ENOB, błędu kwantyzacji w [V], minimalnej i maksymalnej wartości błędu kwantyzacji w [LSB] od rozdzielczości przetwornika (zmienna Bits) w zakresie 1:16 (dla stałego Vref=1). Zrób wykres pokazujący ENOB w funkcji rozdzielczości. Dodaj do wykresu teoretyczną wartość rozdzielczości przetwornika (czyli prostą y=x).
  - Sprawdż czy "Zakres szumu kwantyzacji" wyrażony w LSB zależy od rozdzielczości?

Raport

1. Napisz czy ENOB zależy od Vref i skomentuj wynik.
2. Przedstaw wykres zależności ENOB od rozdzielczości przetwornika i skomentuj zgodność z krzywą teoretyczną.
3. Napisz czy szum kwantyzacji wyrażony w LSB zależy od rozdzielczości przetwornika i skomentuj.

6 Korelacja szumu kwantyzacji, dithering(*)

Skrypt `Zad6_korelacja_szumu_dithering`.

- Wykonaj skrypt. Przedyskutuj wykresy i działanie ditherignu.
- Napisz wyrażenia obliczające:
  - SFDR, czyli stosunek amplitudy piku sygnału (zmienna `signalPeak`) do drugiego najwyższego piku w widmie bez i z ditheringiem (zmienne `maxPeakOrg` i `maxPeakDither`)
  - SNHR, czyli stosunek amplitudy piku sygnału (zmienna `signalPeak`) do wartości średniej poziomu szumów z pominięciem harmonicznych (zmienne `meanNoiseOrg`, `meanNoiseDither`)

Raport

1. Przedstaw (i porównaj) wartości SFDR, SNHR i RMS szumu kwantyzacji w przypadkach bez i z ditheringiem. Skomentuj wpływ ditheringu na RMS szumu kwantyzacji i widmo sygnału po kwantyzacji (wartości SNHR i SFDR).

7 Kwantyzacja w praktycznych przykładach

7.1 Kwantyzacja muzyki(*)

Skrypt `Zad7_1_kwantyzacja_muzyki`.

- Wykonaj skrypt. Odtworzy on fragment pliku '2L-092_01_44kHz_32b_end.mp3' zawierającego samą końcówkę utworu (niewielka głośność). Oryginalny plik zawiera 32b próbki. Następnie skrypt zmniejsza głębokość bitową (zmienna `BitDepth`), czyli wykonuje downsampling, ale w dziedzinie wartości. Czy słyszysz różnicę? Rozumiesz skąd wzięła się nazwa „szum" kwantyzacji?
- Zakomentuj linię 8. (odtwarzanie oryginalnego dźwięku). Wykonaj skrypt stopniowo zmniejszając głębokość bitową `BitDepth` od 16 co 1 w dół. Znajdź próg przy którym zaczynasz słyszeć szumy.
- Przetestuj wartości `BitDepth` mniejsze od progu słyszalności szumów. Zaobserwuj zwiększanie się szumów, a potem (dla małych rozdzielczości) degradację dźwięku.
- Wróć do wartości znalezionej w punkcie 2. (próg słyszalności szumów), ale tym razem odsłuchaj plik 'media/2L-092_01_44kHz_32b.mp3' zawierający fragment ze środka utworu (większa głośność i dynamika). Czy nadal słyszysz szum? Jeśli nie, to zmniejszaj głębokość bitową do momentu pojawienia się szumu kwantyzacji.
- *Zadanie dodatkowe do domu:* Jeżeli masz możliwość, powtórz zadanie w domu na lepszej karcie dźwiękowej/słuchawkach. Spróbuj znaleźć próg słyszalności szumu w takich warunkach.

Raport

1. Przedstaw wartości największej głębokości bitowej dla której słyszysz już szumy w obu utworach (progi słyszalności szumów).

7.2 Kwantyzacja kolorów

Skrypt `Zad7_2_kwantyzacja_kolory`.

- Wykonaj skrypt. Czy Twoim zdaniem dolne paski przedstawiają prostokąty o jednolitym kolorze wewnątrz każdego z prostokątów?
- Usuń return z linii 51. Skrypt wyświetli drugi obrazek, bez górnych połówek pasków prezentujących ciągłą zmianę koloru. Czy paski na tym obrazku przedstawiają prostokąty o jednolitym kolorze wewnątrz każdego z prostokątów?
- Porównaj odczucia wzrokowe z wykresem przedstawiającym wartości jednej z linii paska dla koloru czerwonego.
- Zmniejszaj głębokość bitową BitDepth, zaczynając od 8 co 1 w dół. Przy jakiej wartości zaczynasz widzieć segmentację pasków? Czy dla wszystkich kolorów i ich jasności efekt jest tak samo widoczny (szczególnie bez wpatrywania się

w paski z nosem w monitorze, ale patrząc na niego z normalnej pozycji przed komputerem)?

7.3 Przejścia tonalne(*)

Skrypt `Zad7_3_przejscia_tonalne`.

- Wykonaj skrypt zmniejszając głebokość bitową `BitDepth` co 1 od wartości 8 w dół. Przy jakiej głebokości bitowej zaczynasz dostrzegać kwantyzację na gradiencie kolorów? Czy widzisz kwantyzację na paskach przy tej głębokości?
- Zmniejszaj głębokość dalej, aż do zauważenia wyraźnego efektu na paskach.

Raport

1. Podaj głębokości bitowe przy których:
   - zaczynasz zauważać kwantyzację na gradiencie barw
   - widzisz wyraźnie kwantyzację zielonych pasków.

   Skomentuj (uzasadnij) wyniki. Możesz oczywiście umieścić wykresy przedstawiające przekrój przez obrazki, jeżeli ułatwi to wyjaśnienie efektu.

7.4 Kwantyzacja na przykładzie zdjęć

Skrypt `Zad3_4_kwantyzacja_zdjec`.

- Dla pliku 'hills.jpg' wykonaj kwantowanie z głębokością bitową (zmienna `BitDepth`) zmieniającą się od 8 do 4. Jak poprzednio, znajdź wartość przy której widzisz zmiany w stosunku do oryginalnego obrazu. Zwróć uwagę gdzie widzisz największe (albo jakiekolwiek) zmiany, a gdzie nie.
- Wyeksportuj obraz o tej głębokości bitowej z użyciem funkcji `imwrite`.
- Powtórz zadanie dla pozostałych dwóch plików.

Dodatek

Stosunek sygnału do szumu dla ADC jest wyznaczony (wykład W3, slajd 16.) przy założeniu, że wartość średnia sygnału („środek” sinusa) jest równa zero, tak jak środek przedziału kwantowania rozciągającego się od $-\frac{V_{ref}}{2}$ do $+\frac{V_{ref}}{2}$.

Wówczas RMS sygnału dany jest powszechnie znanym wzorem

$$RMS[A \cdot \sin(\omega t)] = \frac{A}{\sqrt{2}} \tag{18}$$

gdzie, dla modelowego ADC, $A = \frac{V_{ref}}{2}$.

Typowo spotykamy się jednak z sytuacją przetwornika o przedziale kwantowania rozciągającym się 0 do V_{ref}. Wówczas wartość średnia sygnału wynosić musi $\frac{V_{ref}}{2}$.

Ponadto, aby zapewnić że sygnał pokryje dokładnie $2^N - 1$ przedziałów kwantyzacji (rzeczywisty, N-bitowy przetwornik może zwrócić 2^N kodów, czyli dzieli sygnał na $2^N - 1$ przedziałów), amplituda sygnału powinna zostać ograniczona do $\frac{V_{ref}-V_{LSB}}{2}$, gdzie $V_{LSB} = \frac{V_{ref}}{2^N}$.

Powoduje to że sygnał jest opisany równaniem

$$S_{real}(t) = \frac{V_{ref} - V_{LSB}}{2} \cdot \sin(\omega t) + \frac{V_{ref} - V_{LSB}}{2} = A_r \cdot \sin(\omega t) + A_r \tag{19}$$

gdzie

$$A_r = \frac{V_{ref} - V_{LSB}}{2} \tag{20}$$

RMS sygnału $S_{real}(t)$ dany jest więc jako

$$\begin{aligned} RMS[S_{real}(t)] = \sqrt{\frac{1}{T}\int_0^T S_{real}^2(t)dt} = \sqrt{\frac{1}{T}\int_0^T \{A_r[\sin(\omega t) + 1]\}^2 dt} = \\ = A_r\sqrt{\frac{1}{T}\int_0^T [\sin^2(\omega t) + 2 \cdot \sin(\omega t) + 1]dt} = \\ = \frac{A_r}{\sqrt{T}}\sqrt{\int_0^T \sin^2(\omega t)dt + 2\int_0^T \sin(\omega t)dt + \int_0^T dt} = \\ = \frac{A_r}{\sqrt{T}}\sqrt{\frac{T}{2} + 2 \cdot 0 + T} = A_r\sqrt{\frac{3}{2}} \end{aligned} \tag{21}$$

Należy zauważyć że dla procesu kwantyzacji istotna jest różnica pomiędzy wartością średnią sygnału a połową przedziału kwantyzacji. Jeżeli obie te wartości są sobie równe, czyli różnica wynosi zero, to dla procesu kwantyzacji RMS sygnału należy obliczyć tak, jakby jego wartość średnia wynosiła zero. Średni błąd kwantyzacji (i jego RMS) będą bowiem liczone również wokół zera (błąd kwantyzacji rozciąga się zawsze od -0.5 do +0.5 LSB!).

Stąd też, licząc w Matlab'ie RMS sygnału opisanego równaniem 19, należy go znormalizować do wartości nominalnej, danej 18.

$$\begin{aligned} RMS[S_{real}(t)] \cdot \alpha &= RMS[S_{ideal}(t)] \\ \alpha \cdot \frac{V_{ref} - V_{LSB}}{2}\sqrt{\frac{3}{2}} &= \frac{V_{ref}}{2\sqrt{2}} \\ \alpha &= \frac{1}{\sqrt{3}} \cdot \frac{V_{ref}}{V_{ref} - V_{LSB}} \end{aligned} \tag{22}$$