1 Cyfrowe Przetwarzanie Sygnałów

Próbkowanie sygnałów

prof. dr hab. inż. Tomasz Zieliński, dr inż. Jarosław Bułat

2022.03.05

1. Próbkowanie sygnałów analogowych (2+0.5 pkt)

Przyjmijmy, że funkcja `sin(...)` ze środowiska Matlab jest funkcją ciągłą, tj. możemy ją spróbkować w dowolnym miejscu w dziedzinie czasu.

A. Wygeneruj 0.1 sekundy sinusoidy o amplitudzie A=230 V i częstotliwości f=50 Hz (napięcie sieci energetycznej) stosując częstotliwość próbkowania:

1) f_{s1} =10 kHz (pseudo analog) (niebieska linia ciągła 'b-')
2) f_{s2} =500 Hz (czerwone kółko 'r-o')
3) f_{s3} =200 Hz (czarny krzyżyk 'k-x')

Wyświetl wszystkie przebiegi na jednym wykresie z wyskalowaniem osi czasu w sekundach [s].

B. Następnie wygeneruj 1 sekundę sinusoidy f=50 Hz, próbkując:

1) f_{s1} =10 kHz (pseudo analog) ('b-')
2) f_{s2} =51 Hz ('g-o')
3) f_{s3} =50 Hz ('r-o')
4) f_{s4} =49 Hz ('k-o')

Wyświetl te cztery sygnały na jednym wykresie z zachowaniem skali osi czasu w sekundach [s]. Powtórz ostatni rysunek dla f_{sx}=26, 25, 24 Hz, zamiast 51, 50, 49 Hz.

C. Przyjmij f_s=100 Hz. W pętli generuj i wyświetlaj (wyskaluj oś czasu) 1 sekundę sinusoidy, zmieniając jej częstotliwość od 0 Hz co 5 Hz do 300 Hz (61 obiegów pętli, wyświetlaj numer obiegu i zadaną wartość częstotliwości). Potem na jednym wykresie porównaj ze sobą sinusoidy o częstotliwościach 5 Hz, 105 Hz i 205 Hz, następnie 95, 195 i 295 Hz, a na końcu 95 Hz i 105 Hz. Powtórz ten eksperyment dla kosinusoidy zamiast sinusoidy.

D. (**opcjonalnie** +0.5 pkt) Wygeneruj 1 sekundę sygnału sinusoidalnego z sinusoidalną modulacją częstotliwości SFM (częstotliwość próbkowania f_s=10 kHz, częstotliwość nośna f_n=50 Hz, częstotliwość modulująca f_m=1 Hz, głębokość modulacji d_f=5 Hz). Następnie:

1. Wyświetl sygnał zmodulowany oraz modulujący na jednym wykresie.
2. Spróbkuj (f_s=25 Hz) sygnał zmodulowany. Porównaj go z sygnałem ,,analogowym'' na jednym wykresie. Narysuj w osi czasu błędy spowodowane próbkowaniem.
3. Wygeneruj i wyświetl widma gęstości mocy sygnału przed próbkowaniem i po próbkowaniu. Użyj funkcji `spectrum(...)`.

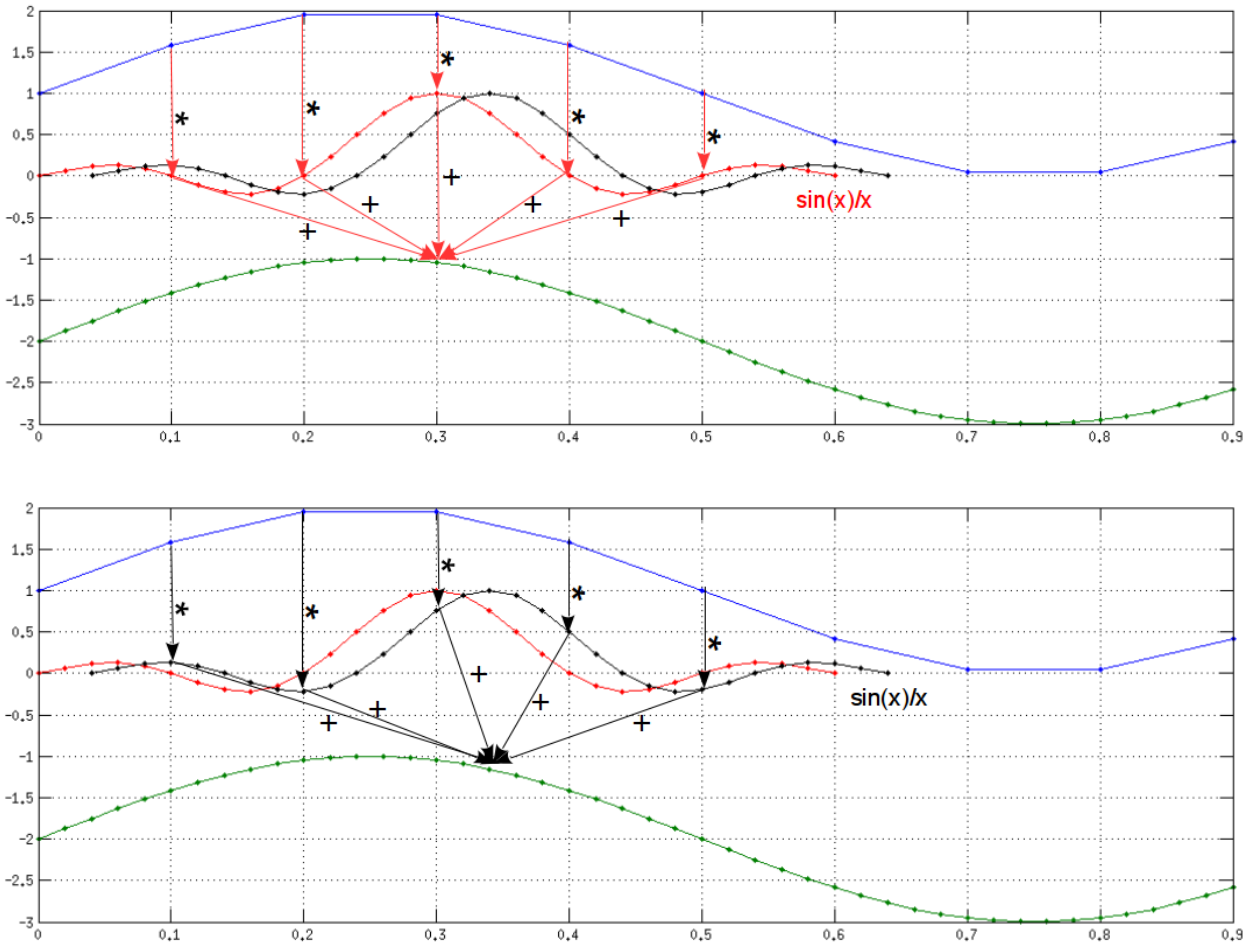
2. Rekonstrukcja sygnału analogowego (2 pkt)

Wykonaj rekonstrukcję sygnału z punktu 1.A za pomocą splotu z funkcją $sin(x)/x$. Sygnał spróbkowany f_{s3}=200 Hz odtwórz w chwilach czasowych odpowiadających częstotliwości próbkowania f_s=10 kHz. Porównaj zrekonstruowany sygnał i ,,pseudo analogowy'' (różnymi kolorami na jednym wykresie). Wyświetl błędy rekonstrukcji.

Uwaga! Podobny przykład był na przedmiocie TOwNiT.

Ilustrację graficzną rekonstrukcji za pomocą splotu z sygnałem $sin(x)/x$ przedstawiono na rysunku 2.1. Oryginalny sygnał zaznaczono niebieskim kolorem, jest on zdefiniowany tylko w chwili t=0, 0.1, 0.2, …, 0.9. Sygnał zrekonstruowany przedstawiono zielonym kolorem. Rekonstrukcję sygnału wykonano dla t=0, 0.01, 0.02, …, 0.1, ..., 0.9 a więc odtwarzając sygnał dla częstotliwości próbkowania pięć razy większej niż w sygnale wejściowym. Poprawnie spróbkowany sygnał wejściowy można odtworzyć w dowolnej chwili t.

Na rysunkach przedstawiono jakie operacje należy wykonać aby uzyskać dwie próbki sygnału wynikowego, w chwili t=0.3 (górny rysunek) i t=0.34 (dolny rysunek). Należy zwrócić uwagę, że jest to operacja splotu oraz na przesunięcie sygnału $sin(x)/x$ względem sygnału wejściowego.



Rys 2.1. Rekonstrukcja sygnału za pomocą splotu z sygnałem $sin(x)/x$

Dla przypomnienia: niech $x(t)$ będzie sygnałem analogowym, natomiast $x(nT)$ - sygnałem $x(t)$ po operacji próbkowania w dziedzinie czasu (niebieskie kropki na Rys 2.1) w chwilach $t=nT$, gdzie $T=1/f_{pr}$ o znacza okres próbkowania, a n numer próbki. Rekonstrukcję sygnału analogowego (zielone kropki na Rys 2.1) przeprowadza się na podstawie wzoru:

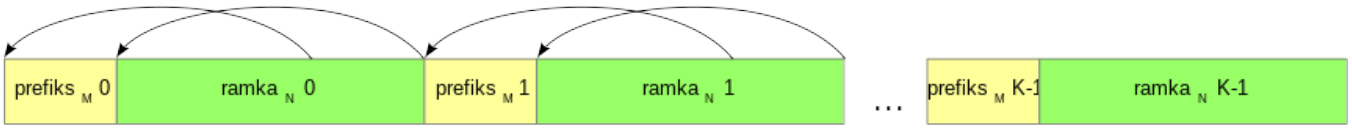
$$x^{(t)}=\sum_{n=-\infty}^{\infty} x(nT)\frac{\sin\left(\frac{\pi}{T}(t-nT)\right)}{\frac{\pi}{T}(t-nT)}=\sum_{n=-\infty}^{\infty} x(nT)\,sinc\left(\frac{\pi}{T}(t-nT)\right)\text{gdzie } sinc(y)=\{\frac{\sin(y)}{y}\,y\neq 0\,1\,y=0$$

Kod programu powinien wyglądać mniej więcej tak jak poniżej:

```
ts=0:0.02:1;                % wektor wszystkich miejsc (t) w których chcemy odtworzyć sygnał
for idx = 1:length(ts)
    t = ts(idx);
    xhat(idx) = 0;
    for n = -3:3                    % zakres sinc()
        xhat(idx) = xhat(idx) + ….      % we wzorze należy wykorzystać „n” i „t”
    end
end
```

3. Korelacja sygnałów (1+0.5 pkt)

Wyszukiwanie prefiksu w sygnale czasowym odbiornika ADSL. Dostarczony sygnał zaprezentowany na rysunku 3.1 ma następującą strukturę: K=4 razy po (M+N)-próbek, gdzie początkowych M=32 próbek (tzw. prefiks) stanowi powtórzenie ostatnich M próbek każdego bloku N=512 próbek.



Rys 3.1. Fragment struktury sygnału ADSL

Wczytaj sygnał w pliku adsl_x.mat, wyznacz początek pierwszej próbki każdego prefiksu. Używaj funkcji `xcorr()` Matlaba.

(**Opcjonalnie** +0.5 pkt) napisz własną funkcję obliczającą korelację wzajemną.

4. Transmisja bitów (opcjonalne, dla dociekliwych, +1 pkt)

(+0.5) Przyjmij częstotliwość próbkowania f_{pr} =16 kHz oraz T =100 milisekund.

Wygeneruj sygnał transmitujący bity kodów znaków ASCII Twojego imienia, np. Janek. Jeśli bit jest równy "0" to czasie T jest transmitowana sinusoida o częstotliwość f_c= 500 Hz, natomiast kiedy "1" — to minus sinusoida.

Narysuj ten sygnał oraz odsłuchaj go, przyjmując w funkcji `soundsc()` wartości f_{pr} równe 8, 16, 24, 32 i 48 kHz. Czy byłbyś wstanie napisać program odczytujący bity z tego sygnału?

(+0.5) Co zrobić, aby szybciej przesłać te same bity? Transmitować krótsze fragmenty sygnału z pojedynczym bitem? A może transmitować kilka bitów jednocześnie? Jak? Np. dodatkowo zmieniać amplitudę sinusoidy i jej przesunięcie kątowe, czyli fazę? Czy umiałbyś to zrobić? Czy umiałbyś zdekodować taki sygnał?

2 Oznaczenia – dotyczy wszystkich konspektów

W instrukcjach bardzo często będzie mieszany kod źródłowy w językach Matlab lub C/C++ oraz wzory matematyczne. Będą one zapisane w różny sposób aby uniknąć pomyłek. Poniżej przykłady:

Kod źródłowy w postaci ramki (żółte tło, ramka na całą szerokość strony):

```
clear all; close all
w1 = [ 0 0 1 0 0 0 0 0 ];              % Wektor 1
w2 = [ 0 0 0 0 1 0 0 0 ];              % Wektor 2
w12 = w1 .* w2;                        % Iloczyn odpowiadających sobie próbek
prod1 = sum(w12)                       % ,,0'' oznacza że wektory są ortogonalne
prod2 = dot( w1, w2 )          % w przestrzeni Euklidesowej
prod3 = w1*w2'                         % bezpośrednie obliczenie (mnożenie wektorowe)
```

Kod źródłowy w tekście. Dotyczy np. nazwy zmiennej, funkcji, nazwy pliku, ścieżki dostępu, etc... (żółte tło, czcionka courier):

Tekst zawierający przykładowe wywołanie funkcji sin(2*pi*t*f+phi) w języku Matlab. Nazwy zmiennych nie zawierają indeksów, np. częstotliwość próbkowania będzie zapisana jako fs.

Wzór (wycentrowany, czcionka szeryfowa – Times New Roman, kursywa, tło białe):

$$w_k(n) = s_k * \cos\left(\frac{\pi * k}{N} * (n+0.5)\right)$$

Wzór w tekście (czcionka szeryfowa – Times New Roman, kursywa, tło białe):

Tekst opisujący powyższy wzór, gdzie $N=128$, $k=8$.

Uwaga! We wzorach mogą wstąpić indeksy, np. częstotliwość próbkowania będzie oznaczona jako f_s.

Zmienne są pisane kursywą np.: N, k, xf, t

Macierze i wektory to litery, proste, pogrubione np.: $\mathbf{A}, \mathbf{n}, \mathbf{v}$