Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

KKY/MS1
Druhá semestrální práce

Yauheni Petrachenka
14. května 2024

1 Zadání

Zadání 2. semestrální práce z předmětu MS1

modelování v diagnostice - zpracování signálu

1. Načtěte signál ze souboru signal.mat do Matlabu

2. Zobrazte časový vývoj signálu (vzorkovací frekvence je 80kHz)

3. Určete časové parametry signálu - střední hodnotu signálu, energii signálu a efektivní hodnotu

3. Určete frekvenční parametry - zobrazte spektrum signálu. Které frekvence jsou v signálu dominantní?

4. Implementujte metodu krátkodobé Fourierovy transformace v Matlabu.

5. Ověřte princip neurčitosti - zvolte krátkou (např. 256 vzorků) a dlouhou (např. 4096 a více vzorků) okénkovou funkci a výsledky zobrazte formou spektrogramu. Jaký je rozdíl mezi oběma spektrogramy? V čem spočívá princip neurčitosti při časo-frekvenčním zpracování signálů?

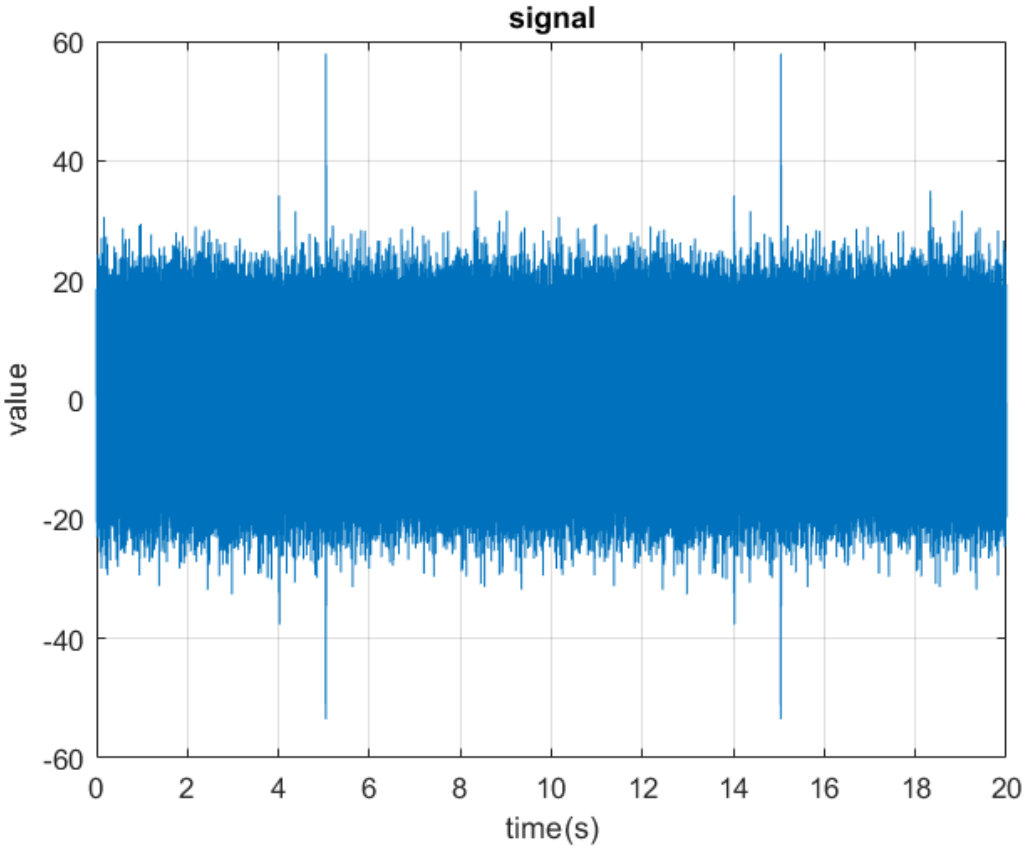
6. Nalezněte časo-frekvenční událost v datech, kolik událostí se v signálu nachází a v jakém čase nastaly?

7. Vytvořte zprávu shrnující získané výsledky formou zobrazení a vysvětlujícího textu. V závěru zprávy uveďte kód z Matlabu, který jste použili pro získání výsledků.

Základní funkce v Matlabu doporučené pro zpracování semestrální práce (podrobnější informace viz dokumentace/help Matlabu):

- load, size, length, for cyklus, ...
- plot, xlabel, ylabel, grid, title,...
- fft, abs, mean, sqrt, hanning, ...
- imagesc, waterfall, caxis, colorbar, ...
- ...

2 načtení a zobrazení signálu



Obrázek 1: Zobrazení signálu

Signál musí být zobrazen s frekvencí $f = 80kHz$. Pro tento úkol musíme vypočítat čas který bude trvat signál $\frac{N}{f} = \frac{N}{\frac{N}{T}} = T$, kde N je počet vzorků. Pomocí těchto výpočtů dostaneme že signál proběhne za dobu 20s. teď lze vykreslit signál který je znázorněn na obrázku 1.

3 Časové parametry signálu

Střední hodnotu signálu lze určit pomocí následujícího vztahu:

$$E = \frac{1}{N} \sum_{i=N}^{i=0} x_i \tag{1}$$

V matlabu tento vztah je realizován pomocí funkce mean(). Po provedení této operace pro daný signál bude obdržen výsledek 'Avarage value of a signal: -0.00030025'. Energie

signálu lze určit pomocí následujícího vztahu:

$$En = \sum_{i=0}^{i=N} x_i^2 \tag{2}$$

Při výpočtech musíme respektovat počet vzorků kvůli tomu musíme vynásobit náš výraz:

$$En = \frac{1}{f} \sum_{i=0}^{i=N} x_i^2 \tag{3}$$

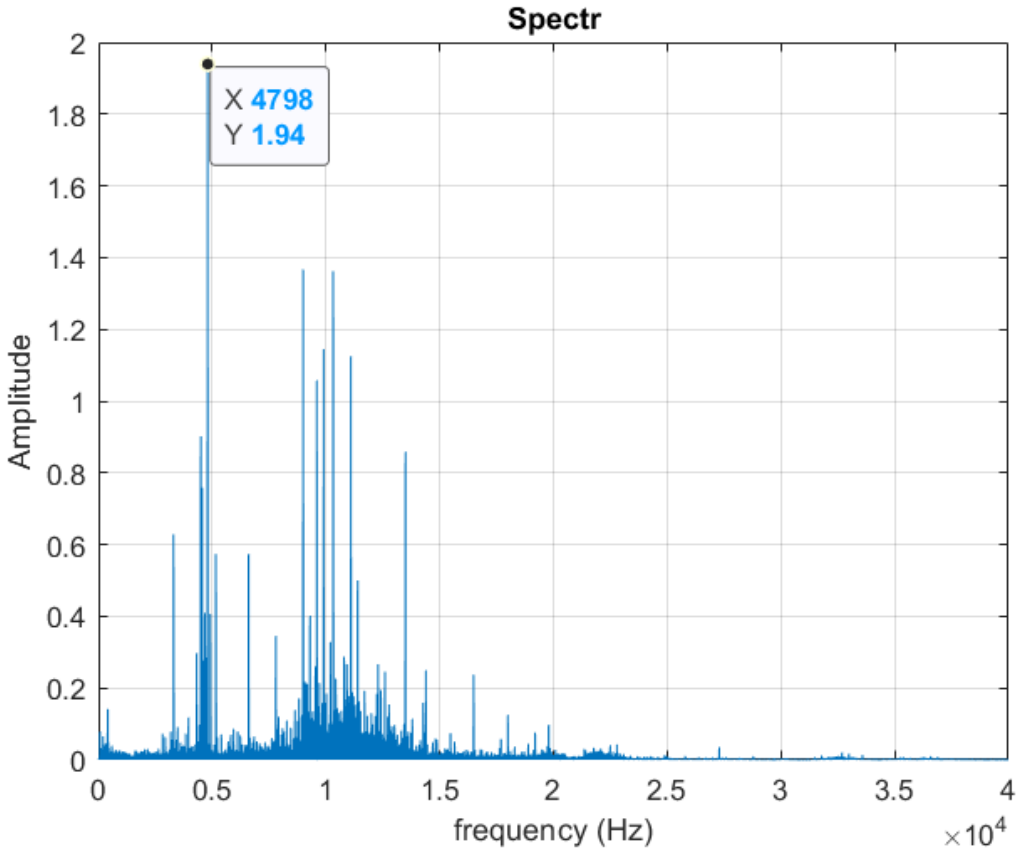
Po provedení této operace pro daný signál bude obdržen výsledek: 'Energy of a signal: 1025.6469'. Efektivní hodnotu signálu pro diskrétní signál lze vypočítat pomocí vzorce:

$$x_{ef} = \sqrt{\frac{1}{N} \cdot \sum_{i=0}^{i=N} x_i^2} \tag{4}$$

Realizujeme tento vztah v matlabu a dostaneme výsledek: 'Effective value:7.1612'.

4 Frekvenční parametry

Pomocí Fourierovy transformace obdržíme následující graf:



Obrázek 2: Zobrazení signálu

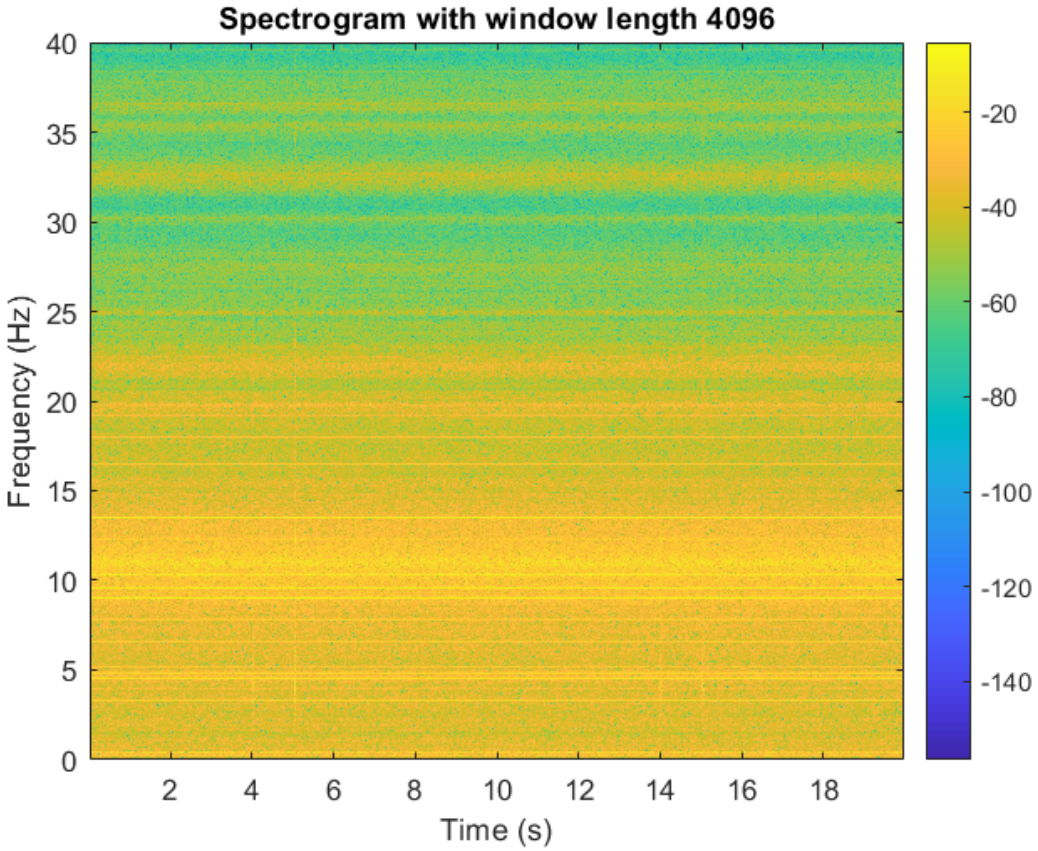
Na grafu je znázorněno že dominantní frekvence je 4798 Hz.

5 Ověření principu neurčitosti

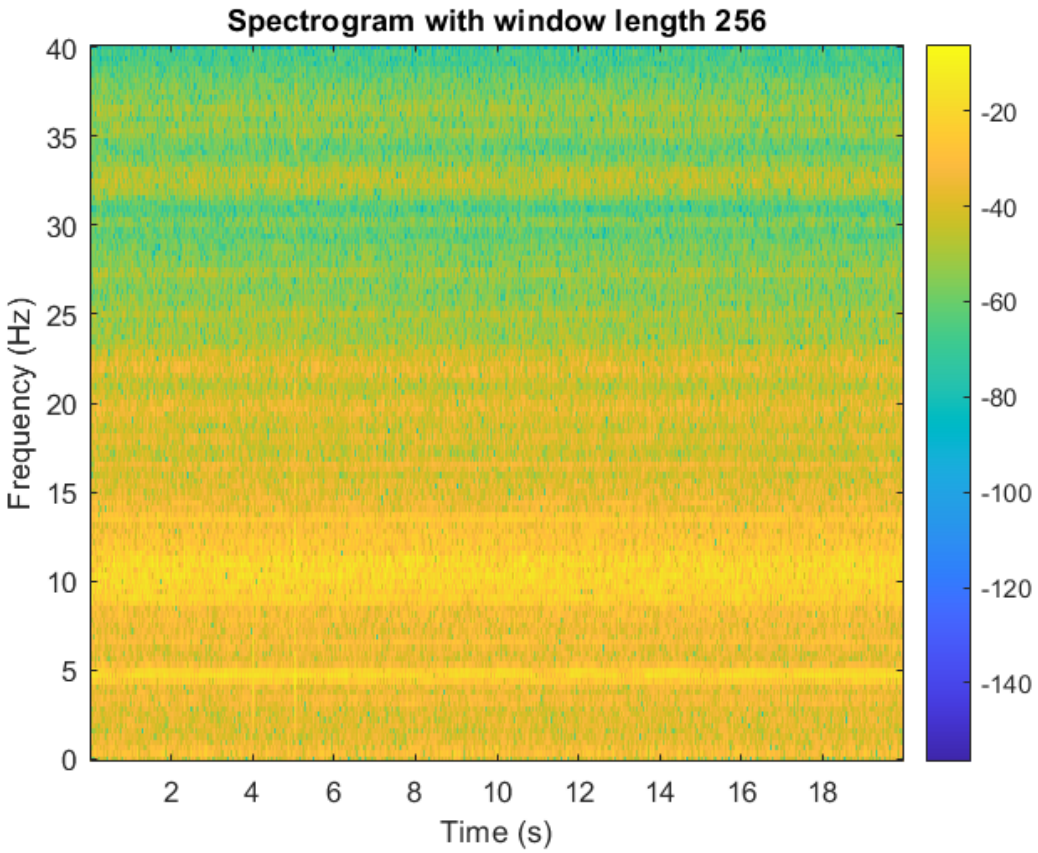
Princip neurčitosti ilustruje, že zpřesněním znalosti jedné charakteristiky (v tomto případě signálu) obětujeme přesnost v jiné oblasti. Konkrétně, zvětšení velikosti okna během analýzy nám umožňuje lépe identifikovat události v čase na úkor ztráty detailů ve frekvenčních charakteristikách. Spektrogram se snaží zachytit stacionární stavy signálu během krátkých intervalů, které jsou definovány právě velikostí použitého okna.

Z obrázků je zřejmé, že s rostoucí velikostí okna jsme sice lépe viděli některé časové události, avšak za cenu ztráty přesnosti ve frekvenčním rozlišení.

S použitím většího okna 2^{12} se dokonce podařilo zmírnit výskyt jedné specifické události kolem 4. sekundy. Abych ověřil časové události, využil jsem spektrogram vytvořený s menším oknem 2^8 , přičemž agregace dat z jednotlivých časových okének odhalila, že signál obsahuje čtyři důležité časo-frekvenční body ve 4., 5., 14. a 15. sekundě.



Obrázek 3: Spectogram with window length 4096



Obrázek 4: Spectogram with window length 256

6 Matlab

```
%% load data and parameters
clc;
clear all;
close all;
data = load('C:\Users\User\Desktop\signal.mat');
frequency = 80000;
%% vizualization of a signal
duration = length(data.signal)/frequency;
time = linspace(0, duration ,length(data.signal));
plot(time, data.signal)
grid on;
xlabel('time(s)')
ylabel('value')
title('signal')
%% time parameters
average_value = mean(data.signal);
disp(['Avarage value of a signal: ', num2str(average_value)]);
energy = 1/frequency*sum(data.signal.^2);
disp(['Energy of a signal: ', num2str(energy)]);
```

```
effective_value = sqrt(mean(data.signal.^2));
disp(['Effective value:', num2str(effective_value)]);
%% frekvencni parametry
N = length(data.signal);
Y = fft(data.signal);

% Create frequency vektor
Fs = 80000;
f = Fs*(0:(N/2))/N;

% Create spektrum of capacity
P = abs(Y/N);
P = P(1:N/2+1);
P(2:end-1) = 2*P(2:end-1);

% Graph
plot(f, P)
grid on;
title('Spectr')
xlabel('frequency (Hz)')
ylabel('Amplitude')
%% princip neurcitosti
windowLengths = [256, 4096];
for i = 1:length(windowLengths)
    window = hamming(windowLengths(i));
    noverlap = round(0.9 * windowLengths(i));
    nfft = max(256, 2^nextpow2(windowLengths(i)));


    [S, F, T] = my_stft(data.signal, window, noverlap, nfft, frequency
    );


    figure;
    spectrogram(data.signal, window, noverlap, nfft, frequency, 'yaxis
    ');
    title(['Spectrogram with window length ', num2str(windowLengths(i)
    )]);
    xlabel('Time (s)');
    ylabel('Frequency (Hz)');
    colorbar;
end


function [S, f, t] = my_stft(signal, window, overlap, Nfft, Fs)
    %UNTITLED Summary of this function goes here
    signal_length = length(signal);
    window_length = length(window);

    step = floor(window_length * (1 - overlap));

    num_segments = floor((signal_length - window_length) / step) + 1;
```

```
70
71      S = zeros(Nfft/2 + 1, num_segments);
72      t = zeros(1, num_segments);
73
74      for i = 1:num_segments
75          start_index = (i - 1) * step + 1;
76          end_index = start_index + window_length - 1;
77
78          segment = signal(start_index:end_index);
79
80          windowed_segment = segment .* window;
81
82          Y = fft(windowed_segment, Nfft);
83
84          P = abs(Y/Nfft);
85          P = P(1:Nfft/2 + 1);
86          P(2:end-1) = 2*P(2:end-1);
87
88          S(:, i) = P;
89
90          t(i) = start_index / Fs;
91      end
92
93      f = linspace(0, Fs/2, Nfft/2 + 1);
94  end
```