

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

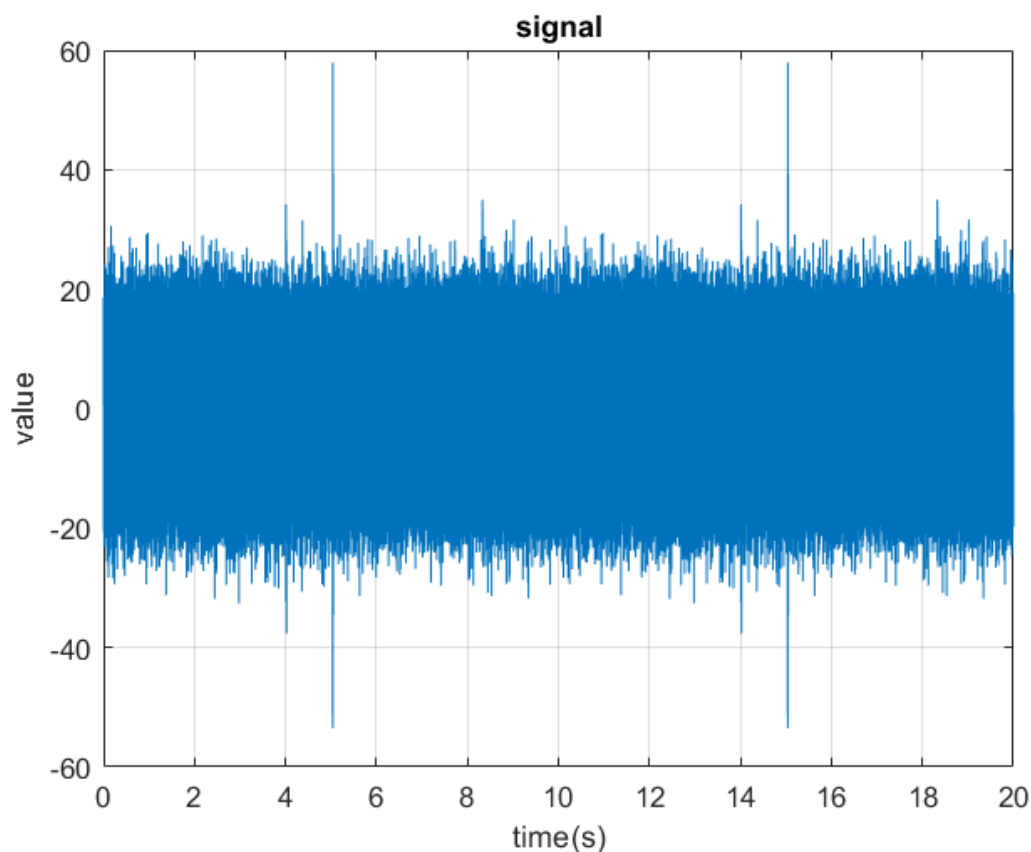
### modelování v diagnostice - zpracování signálu

1. Načtete 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  $f = 80\text{kHz}$ . Pro tento úkol musíme vypočítat čas který bude trvat signál  $\frac{N}{f} = \frac{N}{T} = T$ , 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:

$$E = \frac{1}{N} \sum_{i=0}^{i=N} x_i \quad (1)$$

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

signálu lze určit pomocí následujícího vztahu:

$$En = \sum_{i=0}^{i=N} x_i^2 \quad (2)$$

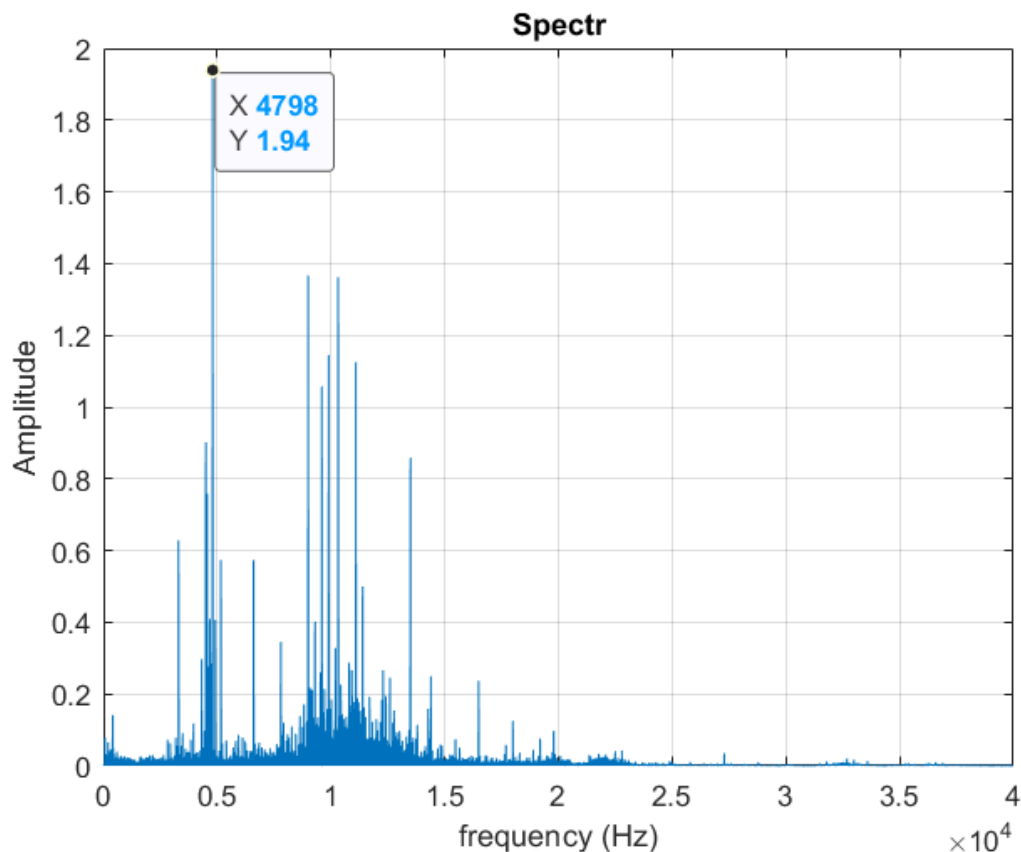
Po provedení této operace pro daný signál bude obdržen výsledek: 'Energy of a signal: 82051749.8568'. Efektivní hodnotu signálu pro diskrétní signál lze vypočítat pomocí vzorce:

$$x_{ef} = \sqrt{\frac{1}{N} \cdot \sum_{i=0}^{i=N} x_i^2} \quad (3)$$

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

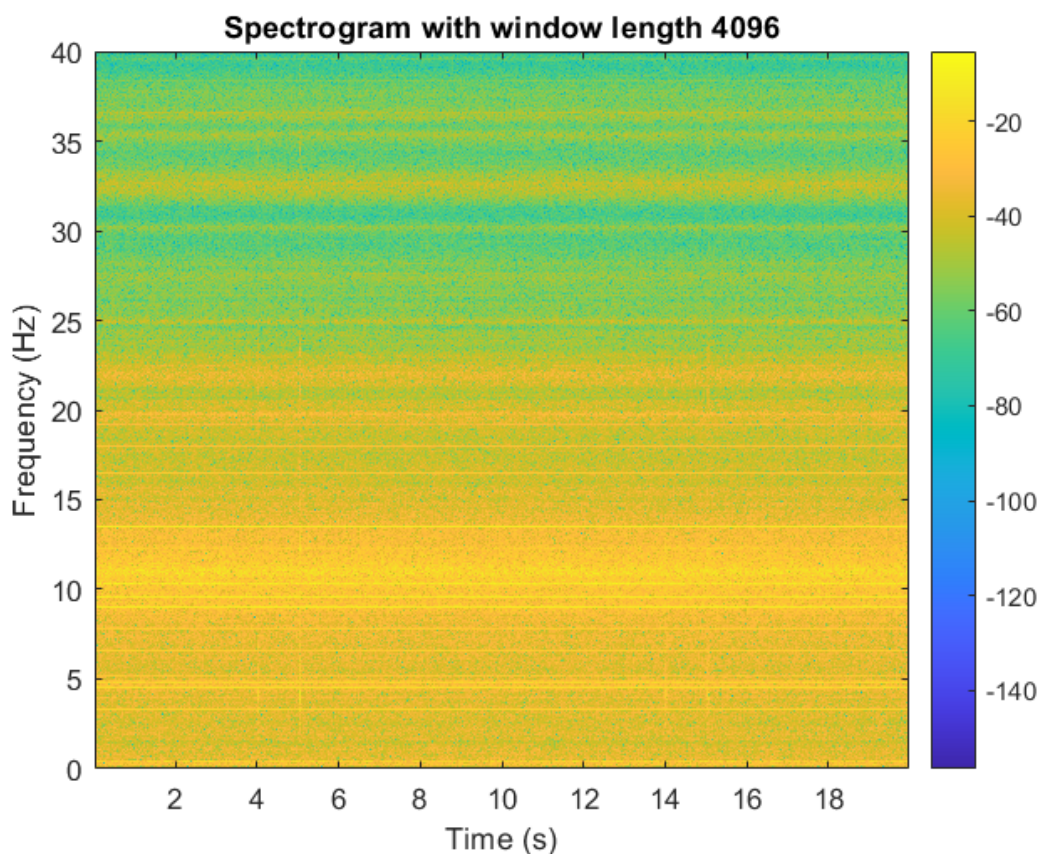
Na grafu je znázorněno že dominantní frekvence je 4798 Hz.

## 5 Ověření principu neurčitosti

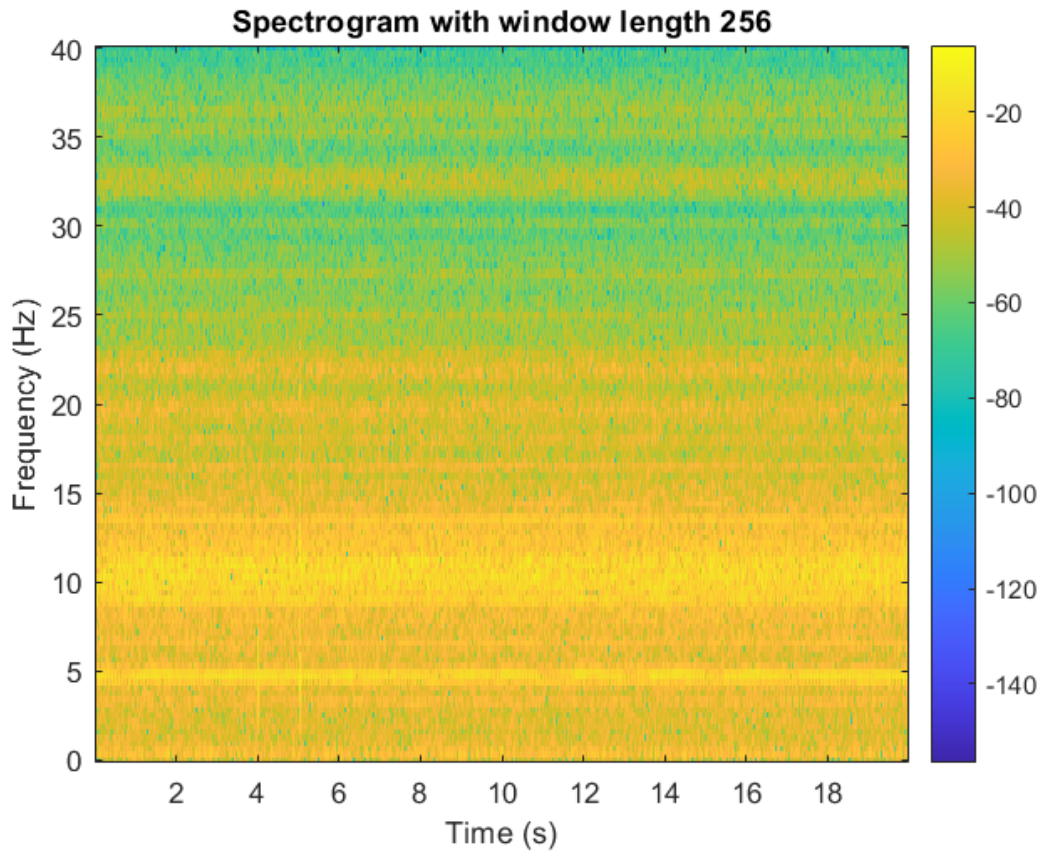
Princip neurčitosti ilustruje, že zpřesněním znalosti jedné charakteristiky (v tomto případě signálu) obětuje přesnost v jiné oblasti. Konkrétně, zvětšení velikosti okna během analýzy nám umožňuje lépe identifikovat události v čase na úkor ztráty detailů ve frekvenčních charakteristikách. Spektrogram 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: Spectrogram with window length 4096



Obrázek 4: Spectrogram with window length 256

## 6 Matlab

```

1 %% load data and parameters
2 clc;
3 clear all;
4 close all;
5 data = load('C:\Users\User\Desktop\signal.mat');
6 frequency = 80000;
7 %% vizualization of a signal
8 duration = length(data.signal)/frequency;
9 time = linspace(0, duration ,length(data.signal));
10 plot(time, data.signal)
11 grid on;
12 xlabel('time(s)')
13 ylabel('value')
14 title('signal')
15 %% time parameters
16 average_value = mean(data.signal);
17 disp(['Avarage value of a signal: ', num2str(average_value)]);
18 energy = sum(data.signal.^2);
19 disp(['Energy of a signal: ', num2str(energy)]);

```

```

20 effective_value = sqrt(mean(data.signal.^2));
21 disp(['Effective value:', num2str(effective_value)]);
22 %% frekvencni parametry
23 N = length(data.signal);
24 Y = fft(data.signal);
25
26 % Create frequency vektor
27 Fs = 80000;
28 f = Fs*(0:(N/2))/N;
29
30 % Create spektrum of capacity
31 P = abs(Y/N);
32 P = P(1:N/2+1);
33 P(2:end-1) = 2*P(2:end-1);
34
35 % Graph
36 plot(f, P)
37 grid on;
38 title('Spectr')
39 xlabel('frequency (Hz)')
40 ylabel('Amplitude')
41 %% princip neurcitosti
42 windowLengths = [256, 4096];
43 for i = 1:length(windowLengths)
44     window = hamming(windowLengths(i));
45     noverlap = round(0.9 * windowLengths(i));
46     nfft = max(256, 2^nextpow2(windowLengths(i)));
47
48
49     [S, F, T] = my_stft(data.signal, window, noverlap, nfft, frequency
50 );
51
52     figure;
53     spectrogram(data.signal, window, noverlap, nfft, frequency, 'yaxis
54 ');
55     title(['Spectrogram with window length ', num2str(windowLengths(i)
56 )]);
57     xlabel('Time (s)');
58     ylabel('Frequency (Hz)');
59     colorbar;
60 end
61
62 function [S, f, t] = my_stft(signal, window, overlap, Nfft, Fs)
63     %UNTITLED Summary of this function goes here
64     signal_length = length(signal);
65     window_length = length(window);
66
67     step = floor(window_length * (1 - overlap));
68
69     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

```