Assignments #3 and #4.

Mathematical Tools for ITS (11MAI)

Mathematical tools, 2020

Jan Přikryl

11MAI, lecture 9

Monday, November 30, 2020

version: 2020-12-07 10:18

Department of Applied Mathematics, CTU FTS

Lectue Contents

Assignment 3.1 — What's the frequency, Kenneth?

a) Review the Matlab Session 4.3 (slides for Lecture 4, pages 48–55).

b) Download both recorder recordings from the lecture website:

```
[x1 sr1] = audioread('recorder1.wav');
[x2 sr2] = audioread('recorder2.wav');
```

c) Your task is related to `spectrogram` usage in non-visual mode,

```
[s,w,t] = spectrogram(...);
```

where `s` is a matrix of FFT coefficients, `w` vector of normalised frequencies, and `t` vector of time stamps where particular FFTs have been computed.

Assignment 3.1 — What's the frequency, Kenneth?

d) Both recording contain an octave played on soprano recorder (staccato, legato). Your task is to analyse `s` and use `w` to find out *particular tones* of the recording.

e) For each recording do the following:

   i. Identify particular base frequencies for each tone
   ii. Identify significant harmonics
   iii. Estimate which tone is being played, i.e. C_2, Fis_3 etc.
   iv. Plot the spectrogram

Lectue Contents

Assignment 4.1 — Playing an octave

Use your knowledge of harmonic signal generation and windowing in Matlab to write a script that generates a `.wav` file playing eight tones of the selected music scale.

How do I generate tone frequencies?

- Every octave has 12 half-tones, 8 main tones
- Every octave doubles the frequency, i.e. $f(\mathrm{C}_5) = 2f(\mathrm{C}_4)$
- Half-tone frequencies form a geometric series

tone	C_4	D_4	E_4	F_4	G_4	A_4	B_4	C_5
f[Hz]	261.63	293.66	329.63	349.23	392.00	440.00	493.88	523.25

Assignment 4.1 — Music scale generator

```
% Define note indexes for C major and G major
c4=40; cis4=41; d4=42; dis4=43; e4=44; f4=45; fis4=46; g4=47; gis4=48;
a4=49; ais4=50; h4=51; c5=52; cis5=53; d5=54; dis5=55; e5=56; f5=57;
fis5=58; g5=59;
% Octaves
c_maj = [ c4 d4 e4 f4 g4 a4 h4 c5 ];
g_maj = [ g4 a4 h4 c5 d5 e5 fis5 g5 ];
```

Example (Frequency of C_4 and C_5 based on A_4)

The tone C_4 is nine half-tones below A_4, and the tone C_5 is three half-tones above A_4, therefore

$$f(C_4) = f(A_4) \cdot 2^{(-9/12)} = 440 \cdot 2^{-0.75} = 261.63\,\text{Hz}$$

$$f(C_5) = f(A_4) \cdot 2^{(3/12)} = 440 \cdot 2^{0.25} = 523.25\,\text{Hz}$$

Assignment 4.1 — Music scale generator

Using the c_maj defined above, generate one octave of C-major scale:

```
Fs = 8000; % sampling frequency
t0 = 0.2; % tone duration
tt = 0:(1/Fs):t0; % sample times
yc = []; % output
for key = c_maj % go through all keys in c_maj
  % compute the tone frequency, freq(a4)=440 Hz
  freq = 440 * (2 .^((key - a4)/12));
  % concatenate the new tone to the previous ones
  yc = [ yc, sin(2*pi*freq*tt) ]; % or cos(2*pi*freq*tt - phi)
end
sound(yc, Fs); % play it with the given sampling frequency
```

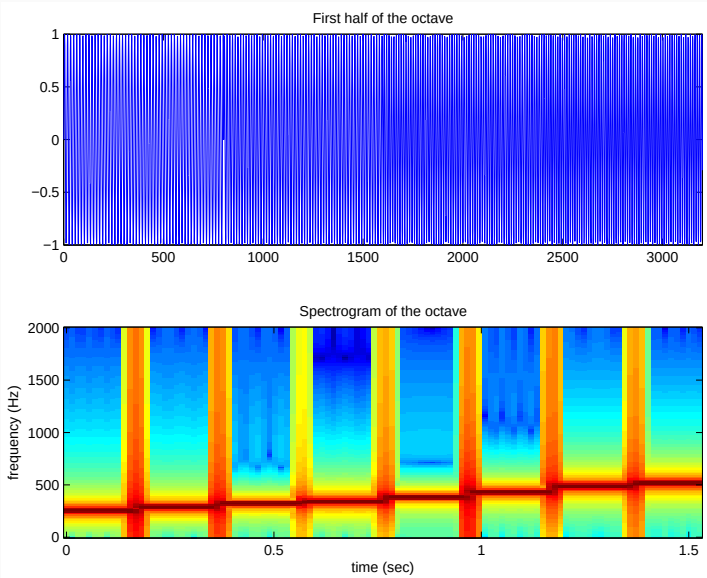
Assignment 4.1 — Music scale generator issues

The tone transitions are quite disturbing, let's check with spectrogram:

```
Fmax = Fs/4; % maximum frequency
Nfft = 256; % number of FFT coefficients
Nover = 200; % number of overlapping samples
% Display the spectrogram
figure(1);
spectrogram(yc, Nfft, Fmax, [], Nover);
title('Spectrogram of the C-major without windowing');
```

The distortion ("clicking") occuring at tone transition point is due to an abrupt change of the waveform amplitude. This can be mitigated e.g. by applying a window of suitable size to the tone sample.

Assignment 4.1 — C-major spectrogram with transients



Assignment 4.1

a) Repeat the steps outlined above for G-major scale, with samples stored in `yg` vector.

b) Save your sound using `audiowrite('g_major_orig', yg, Fs)`.

c) It sounds rather artificially, can you find the way to improve the generated record?

d) Bonus: Repeat the steps (a) and (b) for windowed tone samples.

e) Deliver the code that generates the `.wav` format(s) as a ZIP file by December 23, 2020. No report is necessary.

Assignment 4.1

a) Repeat the steps outlined above for G-major scale, with samples stored in `yg` vector.

b) Save your sound using `audiowrite('g_major_orig', yg, Fs)`.

c) It sounds rather artificially, can you find the way to improve the generated record? Hint: Use a window function to overlap the tones!

d) Bonus: Repeat the steps (a) and (b) for windowed tone samples.

e) Deliver the code that generates the `.wav` format(s) as a ZIP file by December 23, 2020. No report is necessary.

Assignment 4.1

a) Repeat the steps outlined above for G-major scale, with samples stored in `yg` vector.

b) Save your sound using `audiowrite('g_major_orig', yg, Fs)`.

c) It sounds rather artificially, can you find the way to improve the generated record? Hint: Use a window function to overlap the tones! Hint: Add harmonics!

d) Bonus: Repeat the steps (a) and (b) for windowed tone samples.

e) Deliver the code that generates the `.wav` format(s) as a ZIP file by December 23, 2020. No report is necessary.

Assignment 4.1

a) Repeat the steps outlined above for G-major scale, with samples stored in `yg` vector.

b) Save your sound using `audiowrite('g_major_orig', yg, Fs)`.

c) It sounds rather artificially, can you find the way to improve the generated record? Hint: Use a window function to overlap the tones! Hint: Add harmonics! If you want to be even cooler, look at additive, subtractive or FM synthesis.

d) Bonus: Repeat the steps (a) and (b) for windowed tone samples.

e) Deliver the code that generates the `.wav` format(s) as a ZIP file by December 23, 2020. No report is necessary.

Assignment 4.1

a) Repeat the steps outlined above for G-major scale, with samples stored in `yg` vector.

b) Save your sound using `audiowrite('g_major_orig', yg, Fs)`.

c) It sounds rather artificially, can you find the way to improve the generated record? Hint: Use a window function to overlap the tones! Hint: Add harmonics! If you want to be even cooler, look at additive, subtractive or FM synthesis.

d) Bonus: Repeat the steps (a) and (b) for windowed tone samples.

e) Deliver the code that generates the `.wav` format(s) as a ZIP file by December 23, 2020. No report is necessary.

Lectue Contents

Assignment 5.1 — Christmas Carol

a) Extend the approach from Assignment 4.1 to take into account the length of the tones — probably the best solution would be to use an array of tone lengths:

```
jednadve_t = [a4 a4 f4 a4 a4 f4 a4 a4 h4 a4 a4 g4 ]; % tones
jednadve_d = [ 8 8 4 8 8 4 8 8 8 8 4 4 ]; % tone lengths 1/8 and 1/4 of t0
```

and modify the generator of sample times `tt` to end at `t0/tdiv`:

```
for i = 1:length(jednadve_t) % go through all keys indexes
  key = jednadve_t(i); % tone number
  tdiv = jednadve_d(i); % length divisor for the tone
  tt = 0:(1/Fs):(t0/tdiv); % t0 is now the bar length
  freq = 440*(2.^((key-a4)/12));
  sample = sin(2*pi*tt); % can be modified by windowing
  y = ...
end
```

Assignment 5.1 — Christmas Carol

b) Select a Christmas Carol *that has not been composed in C-major scale*.

c) Compose the Carol using Matlab commands.
Generate the sampled tones using the non-windowed approach from Assignment 4.1 taking into account different tone lengths as suggested in (a) of this list. Save the result to a `.wav` file.

d) Bonus: Compose a Christmas Carol with a tone scale improved by windowing, harmonics or some other approach mentioned in Assignment 4.1

e) Deliver the code that generates the `.wav` format as a ZIP file by December 23, 2020. No report is necessary.