Sampling and quantization

May 21, 2019

Basic definitions:

1. *sampling rate* - the number of samples per second
2. *analog signal frequency* - the number of signal's periods per second (Hz)
3. *digital frequency* - phase angle per sample (rad/s) or (πrad/s)

If a function $x(t)$ contains no frequencies higher than B hertz, it is completely determined by giving its ordinates at a series of points spaced $1/(2B)$ seconds apart.

1 Discrete-time sinusoidal signal

Discrete time sinusoidal signal can be written as

$$s(n) = \sin(\Omega n) \ , \tag{1}$$

where Ω is digital frequency expressed in radians per sample. Generate the following arrays

```
n=1:10;
sin(0.2*pi*n)
sin((0.2*pi+2*pi)*n)
sin((0.2*pi-2*pi)*n)
```

What are the digital frequencies of these signals? Why there is no difference between these signals?

Task 1

Generate sampled signal $f(t) = sin(2\pi 1000t)$ (excerpt with 512 samples) with the following sampling rates

- 1500 Hz
- 2000 Hz

- 2500 Hz

Plot spectra for the above cases. Comment these results in your report. You can use an example below.

```
f = 1000;% analog signal frequency
fs = 1500; % sampling rate
n = 1:512; % indexes of samples
x = sin(2*pi*f/fs*n); % f - analog signal frequency
[h,cz] = freqz(x,1,512,fs);
plot(cz,abs(h));
% wykres
xlabel(Frequency);
```

Task 2

Use function `sound` to play signals with the provided sampling rates(`sound(x, fs)`). Can you play signal `x` correctly for all sampling rates?

Task 3

Read `obraz.jpg` using function `imread`. What is the size of the array that represents this signal? What is the meaning of dimensions of the array?

If each pixel is represented by 8-bit number, how much disk space is needed to steore this image without compression? How many megapixels could have the camera which was used to take this picture?

Task 4

Use function `down_samp` to change spatial sampling

```
[new_image] = down_samp(image, level)
```

of 2, 4, 5, 16, 32 levels. Describe how the function works? For which level the quality of the picture is not sufficient?

Task 5

Use function `quantization.m` to decrease quantization of image of 2, 8, 32, 128 levels:

```
[new_image] = quantization(image, level)
```

Task 6

Generate signal $r[n] = n/1000$ for $n = 1000, \ldots, 1000$. Perform quantization with the following instructions

```
levels=8;
step=(max(r)-min(r))/levels;
partition=min(r):step:max(r);
partition=partition(2:end-1);
rq=quantiz(r, partition);
plot(n, rq, .)
```

Task 7

Fit the range of signal rq to range of the original signal. Calculate quantization noise.

Task 8

Generate 1000 samples of the Gaussian noise. Prepare the following scripts

```
function s=snr(s, noisy)
s = s(:)
noisy = noisy(:)
s = 10*log10(s*s/((s-noisy)*(s-noisy)))

function y=snrq(r, levels)
partition=min(r):(max(r)-min(r))/levels:max(r)
partition=partition(2:end-1)
rq = quantiz(r, partition);
step=(max(r)-min(r))/levels;
A=
B=
ra = rq*A+B;
y=snr(r, ra);
```

Compute SNR values for the quantizers with 4, 8, 10, and 12 bits. Compare these results to the theoretical values. Discuss possible causes of differences between empirical and theoretial SNRs.