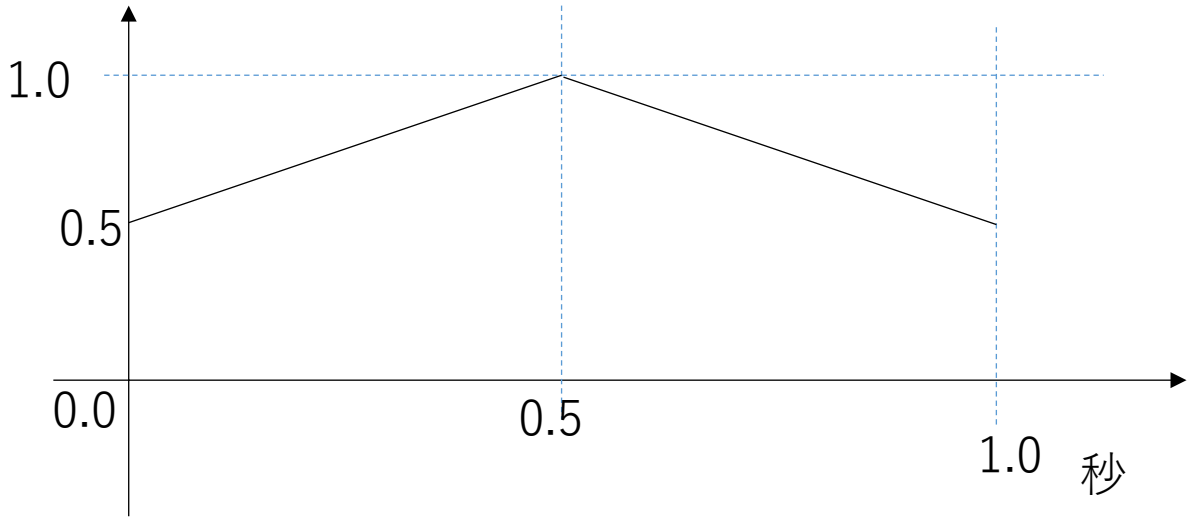
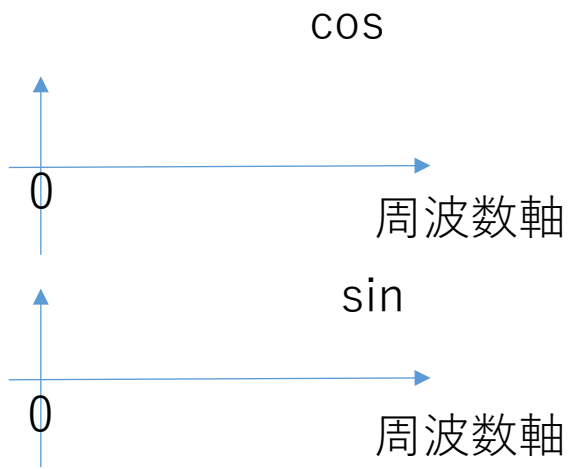
デジタル信号処理の基礎 #5

November 3, 2025

Assignment #4

時間領域関数y(t) = 0.5 + t (0≦t<0.5) , y(t)=1.5 – t(0.5≦t<1.0)をfft関数を用いずに周波数領域信号に変換し、cos成分とsin成分を横軸を周波数としたグラフに重ね書き(hold on)やsubplotを利用して別々に表示するmスクリプトを提出せよ。



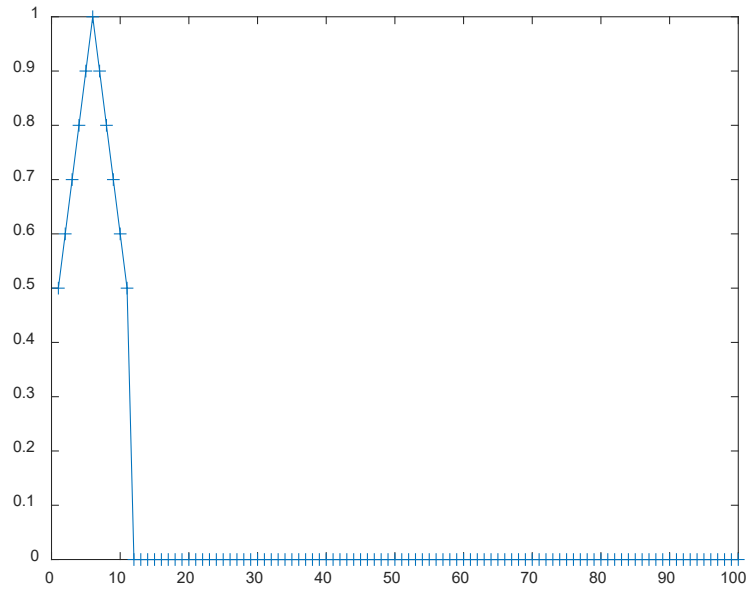


as4

```
% 佐藤 弘基, 72443869
Fs = 100;
L = 1.0; % length in second
res = 1/L; % frequency resolution
numF = Fs/res; % frequency components number

t = 0.0:1/Fs:L-1/Fs;
y = zeros(size(t)); % まずゼロで初期化
% 前半 (0 <= t < 0.5)
idx1 = t < 0.5;
y(idx1) = 0.5 + t(idx1);
% 後半 (0.5 <= t <= 1.0)
idx2 = t >= 0.5;
y(idx2) = 1.5 - t(idx2);
% 結果を入れる
fc = zeros(numF, L*Fs);
fs = zeros(numF, L*Fs);
a = zeros(numF,1);
b = zeros(numF,1);
f = (0:1/L:Fs-1/L);
size(f)
for i=1:numF
fc(i, :) = cos(2*pi*f(i)*t);
fs(i,:) = sin(2*pi*f(i)*t);
end

for i=1:numF
if i==1
k=1;
else
k=2;
end
a(i) = k*fc(i,:)*y'/L/numF;
b(i) = k*fs(i,:)*y'/L/numF;
end

% 再構成するサンプル数
Nrec = 50; % 例えば50次まで使う
y_rec = zeros(size(t));
for i = 1:min(Nrec, numF)
y_rec = y_rec + a(i)*cos(2*pi*f(i)*t) + b(i)*sin(2*pi*f(i)*t);
end
% プロット
figure;
plot(t, y, 'r', 'LineWidth', 2); hold on;
plot(t, y_rec, 'b--');
legend('original','reconstructed');
hold off

figure;
plot(f, a);
hold on
plot(f, b);
legend("cos", "sin");
hold off
```

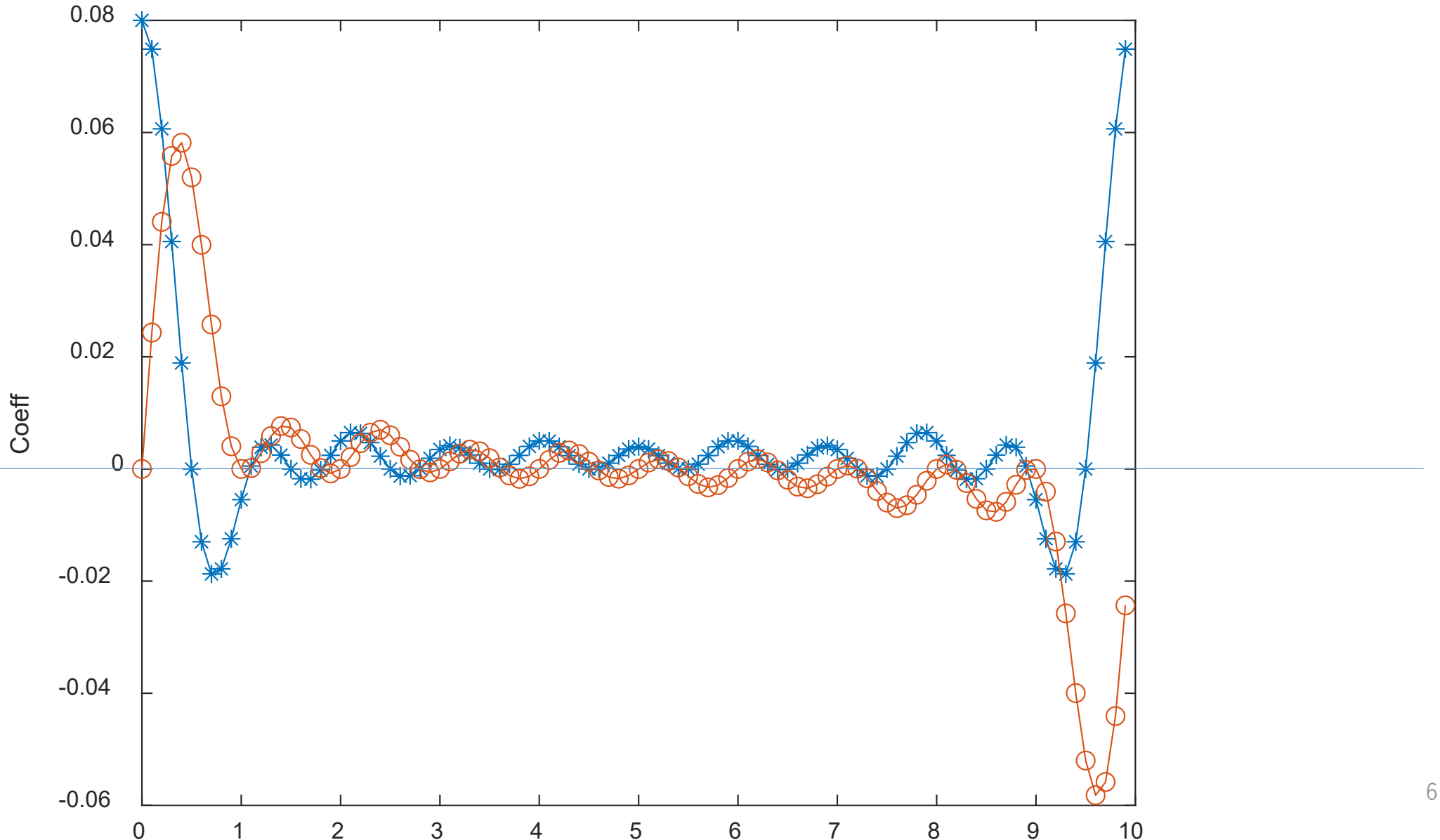
```
%古宮祥太 72470019
%4秒で1振動の0.25 hz の波形に近いので、低い周波数も見やすくするためサンプリングレートを小さくした
clf; % clear figure
Fs = 100; %sampling rate 1秒間に標本化する回数
L = 1; % 1秒
t = (0:1/Fs:L-1/Fs);% 0から L-1/Fs まで1/Fs 刻み サンプル数は L*Fs
res = 1/L; % 分解能 1hz
numF = Fs/res; % 成分数
y = (t<0.5).*(0.5 + t) + (t>=0.5).*(1.5 - t);
% disp(y);
fc = zeros(numF, L*Fs);% zeros(m,n) m*nのゼロ行列
fs = zeros(numF, L*Fs);
a = zeros(numF,1);
b = zeros(numF,1);
f = (0:1/L:Fs-1/L);% 0~7999 Hz 1Hz刻みの周波数軸ベクトル
for i=1:numF
fc(i,:) = cos(2*pi*f(i)*t);
fs(i,:) = sin(2*pi*f(i)*t);
end
for i=1:numF
if i==1% kは0Hz の場合のみ1 他は2
k=1; else
k=2; end
a(i) = k*fc(i,:)*y'/L/numF;% ' は転置 行ベクトル* 列ベクトル スカラー値
b(i) = k*fs(i,:)*y'/L/numF;
end
hold off; % overwrite suppressed
plot(f, a);
hold on; % overwrite enabled
plot(f, b);
xlabel('frequency(Hz)');
ylabel('intensity');
```

sin (2π/2 t)の成分があるはずでは？
Resolutionを高めるにはDurationが長くないといけない



Zero padding

```
Fs = 10;
duration = 10;
nsample = int16(duration*Fs);
t = (0:1/Fs:duration - 1/Fs);
f = (0:1/duration:Fs-1/duration);
y = zeros(nsample, 1);
%y = [0.5 0.6 0.7 0.8 0.9 1.0 0.9 0.8 0.7 0.6]';
nn = uint16(0.5*Fs);
for i=1:nn
y(i) = 0.5 + t(i);
y(nn+i) = 1.5 - t(i+nn);
end
y(1+2*nn) = 0.5;
[cs, sn] = discFT(y);
hold off;
plot(f, cs);
hold on
plot(f, sn);
```


0.08
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
Coeff
0
1
2
3
4
5
6
7
8
9
10

Assignment # 5

Exercise 3.4 *(threeTonesFft) If we produce a harmony A with the following script, we obtain frequency domain signal where the level of the three tones are different. If we generate three A tones such that* A_{220} *= 220 Hz,* A_{440} *= 440 Hz and* A_{880} *= 880 Hz, these three tones produce exactly the same level frequency components. Explain why the three levels are different only in the case of harmony A.*

```
%%  fft of harmony A.m
%
%   generate a 440Hz tone
Fs = 8000; % sampling rate
A = 440; %tone frequency
period = 0.02; %signal length in second
t = (0:1/Fs:period-1/Fs);
f = (1/period:1/period:Fs);
sep = power(2, 1/12);
Cs  = A*power(sep, 4)
E = Cs*power(sep, 3)
  % we reduce the amplitude by 0.2 to avoid distortion.
y = 0.2*sin(2*pi*(A)*t) + 0.2*sin(2*pi*Cs*t) + 0.2*sin(2*pi*E*t);
sound(y, Fs);
fy = fft(y);
plot(f, abs(fy)/length(t));
xlabel('frequency (Hz)')
ylabel('gain')
```

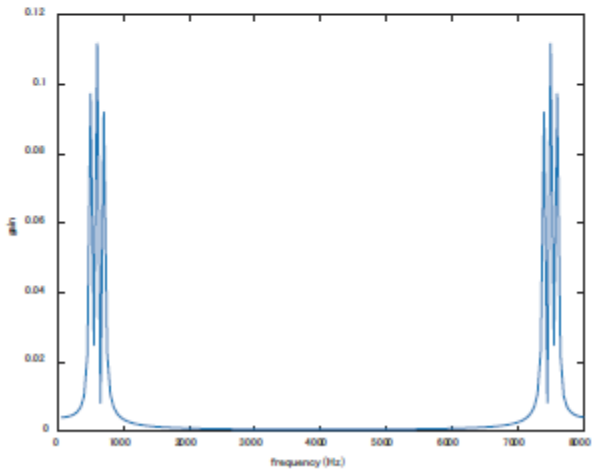
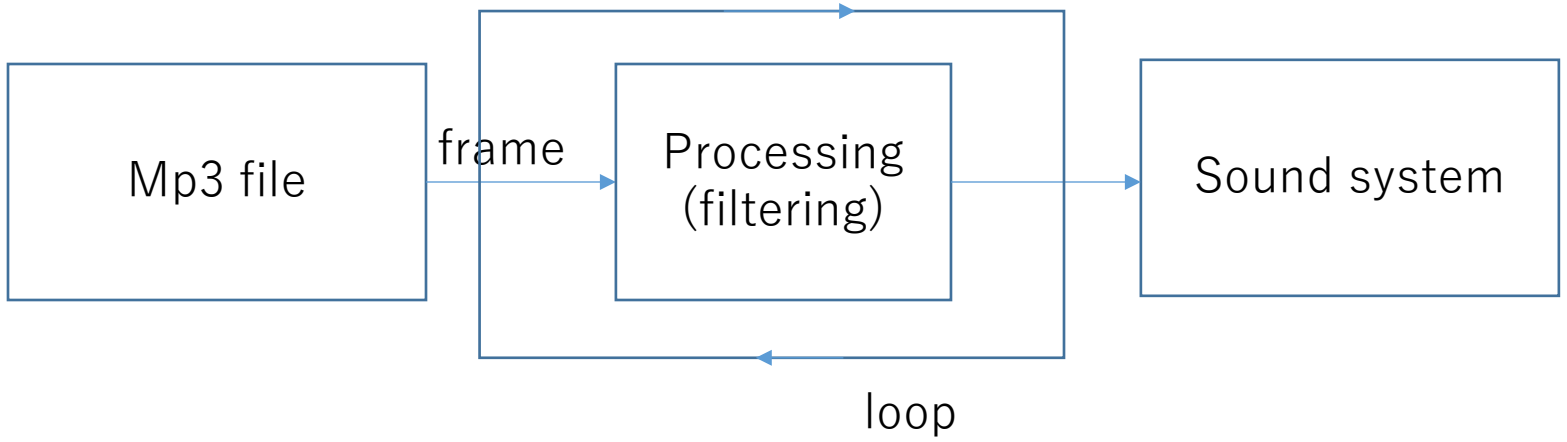


Figure 3.15: Frequency domain signal of harmony A for 0.02 second

musicplaydsp_base

Read mp3 file for a frame length, process the frame and output to the sound system (speaker).



Simple music player (musicdsp_base)

```
%
%  simple music play
%
clear;
frameLength = 1024; % length of a frame
mfile = uigetfile('*.mp3'); %mp3 file
fileReader = dsp.AudioFileReader(...
    mfile,...
    'SamplesPerFrame',frameLength);
Fs = fileReader.SampleRate;
deviceWriter = audioDeviceWriter(...
    'SampleRate',Fs);% audio system define

while ~isDone(fileReader)
    signal = fileReader(); % read frameLength samples from the mp3 file
    deviceWriter(signal); % write to audio device
end
release(fileReader);
release(deviceWriter);
```