DAFTAR PUSTAKA

Aldiano, A. (2014). *Buku Lengkap Belajar Alat Musik. Jakarta: Saufa.*

Banoe, P. (2003). *Kamus Musik. Yogyakarta: Kanisius.*

Binanto, I. (2010). *Multimedia Digital - Dasar Teori + Pengembangannya.* Yogyakarta: Andi Offset.

Depdiknas. (2003). Standar Kompetensi Mata Pelajaran Kesenian. Jakarta: Pusat Kurikulum, Balitbang Depdiknas.

Dewi Awirat, Diah Ayu Retnani, Olivia Kembuan. (n.d.). MIDI (Musical Instrument Digital Interface) . Yogyakarta.

Dhanya M Dhanalaksh, H. P. (2017). Musical Notes to MIDI Conversion.

Djohan. (2016). Psikologi Musik. Yogjakarta: Indonesia Cerdas.

Hardiman, A. F. (2000). *Belajar sendiri Cakewalk Pro Audio versi 8.0* . Jakarta: Elex Media Komputindo.

Jamalus. (1988). Panduan Pengajaran buku Pengajaran musik melalui pengalaman musik. Jakarta. Proyek pengembangan.

MIDI Manufactures Assotiation. (1995). *Tutorial on MIDI and Music Synthesis. California: MIDI Manufactures Assotiation Incorporated.* California.

Wolfe, J. (t.t.). *Note names, MIDI numbers and frequencies*. Retrieved Januari 2020

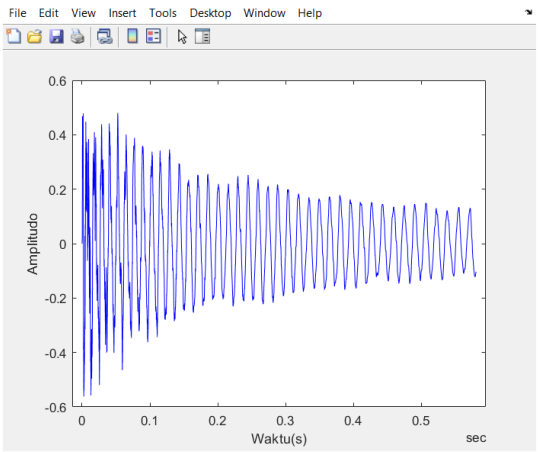
LAMPIRAN

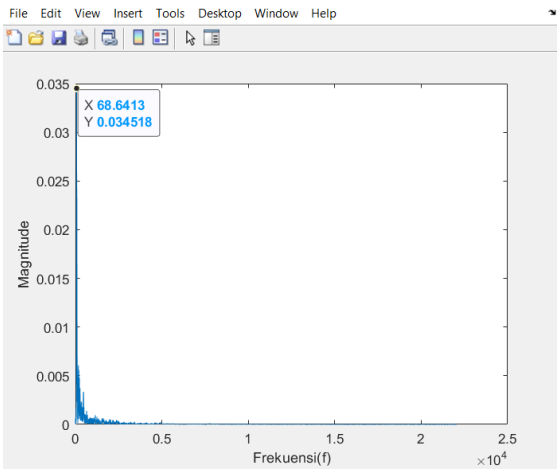
Lampiran 1. Plot Wav, FFT dan Midi Key

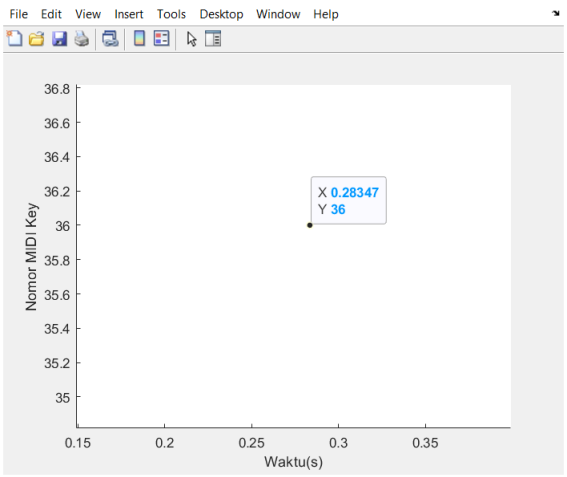
1. Bass Drum (Standar Midi Key : 36)

   - Data 1-5

     Nomor Midi : 36



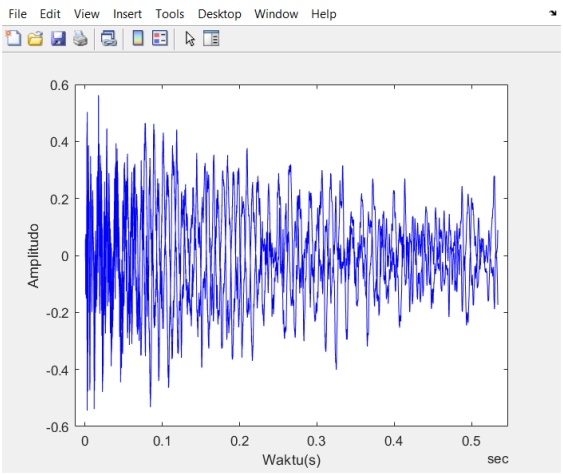


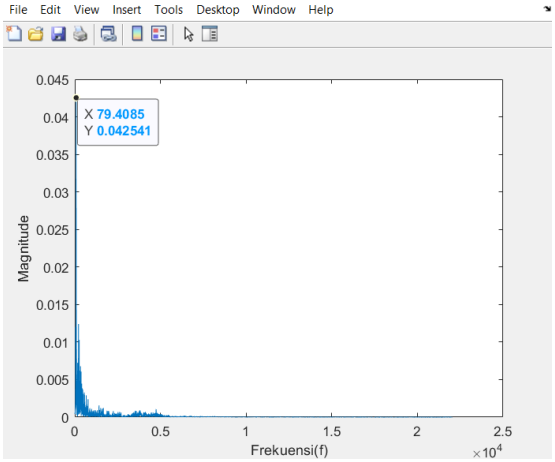


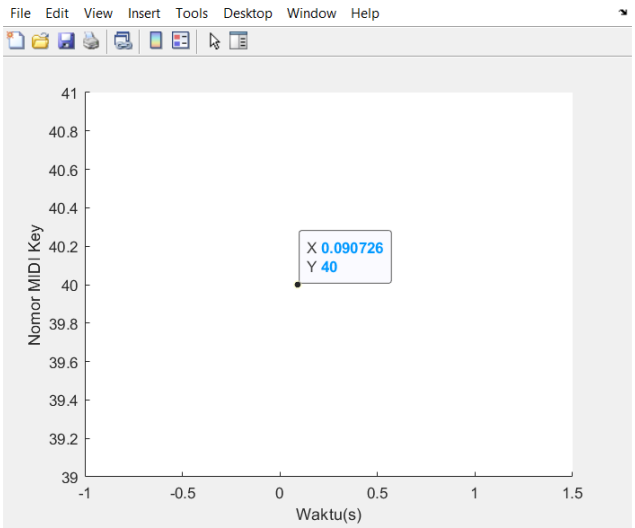
2. Snare Drum (Standar Midi Key : 40)

- Data 1-5

  Nomor Midi : 40



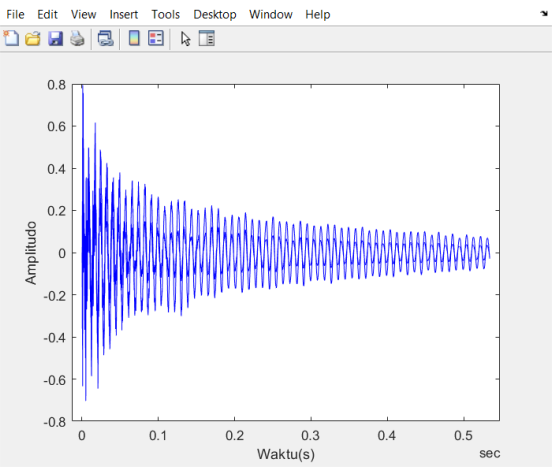


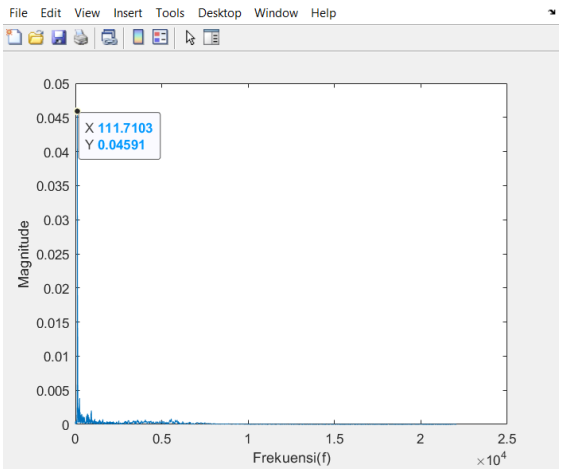


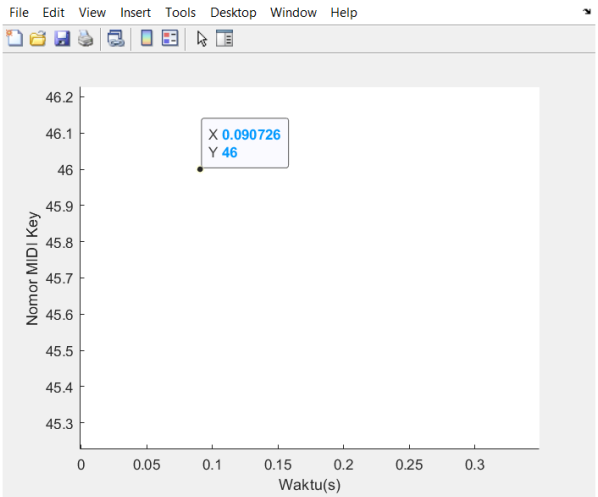
3. Hi-Hat (Standar Midi Key : 46)

- Data 1-2

  Nomor Midi : 46



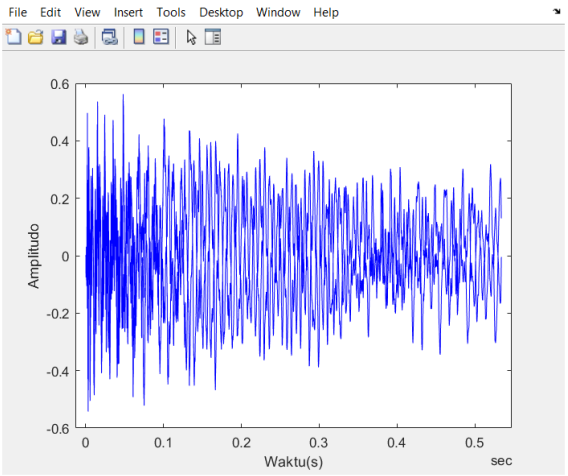


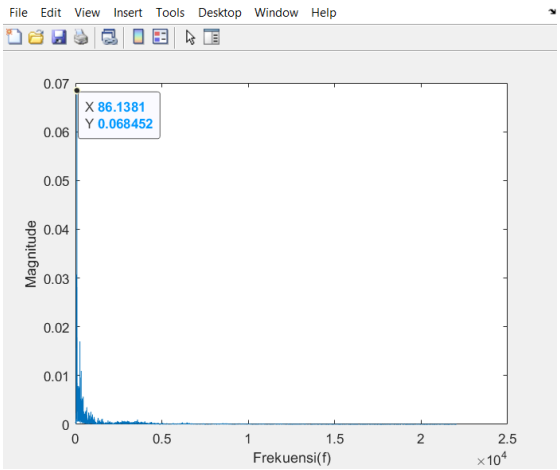


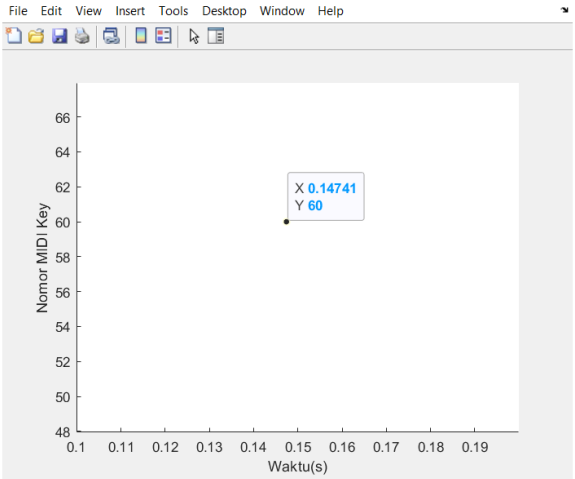
4. Ride Cymbal (Standar Midi Key : 59)

- Data 1-5

  Nomor Midi : 59



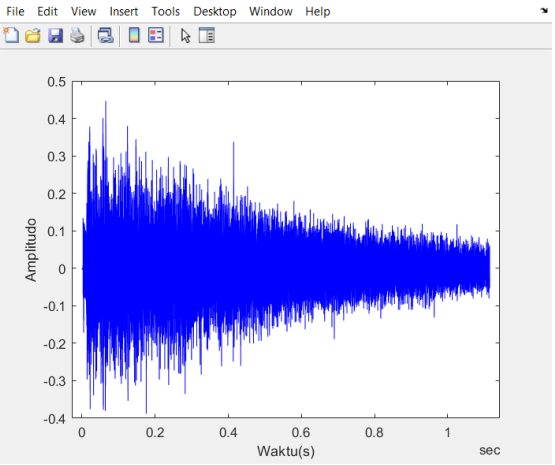


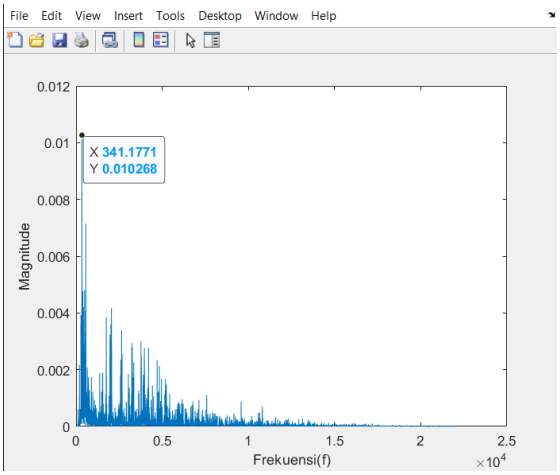


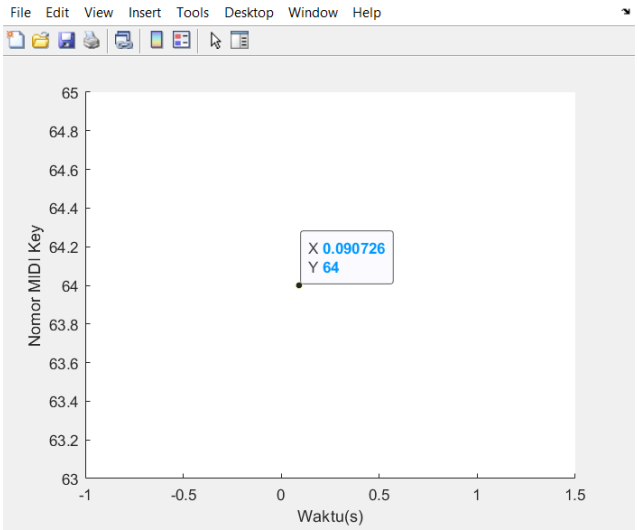
5. Crash Cymbal (Standar Midi Key : 57)

- Data 1-5

  Nomor Midi : 57



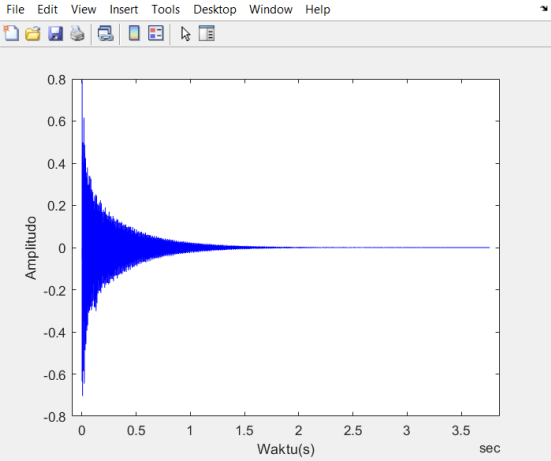


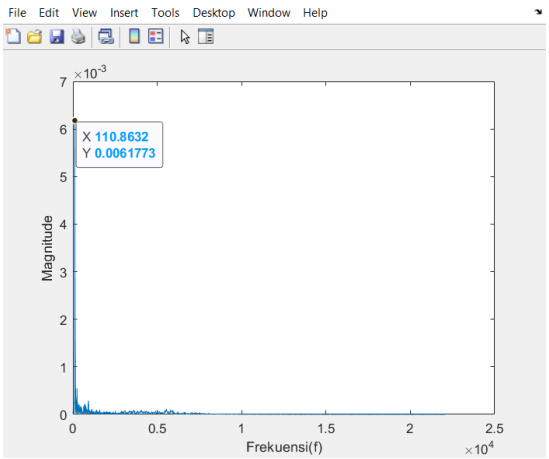


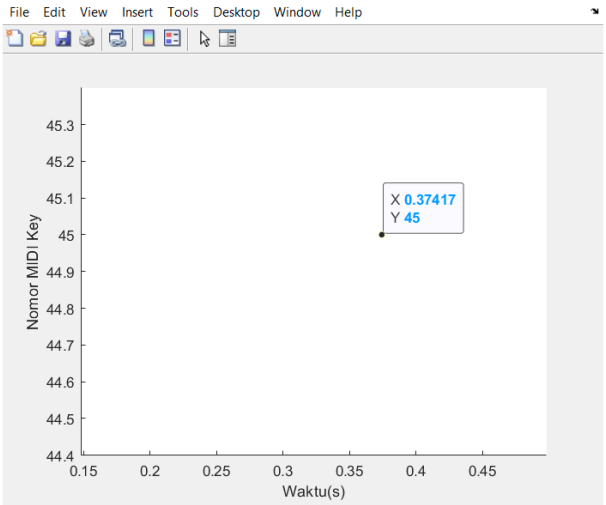
6. Low Tom (Standar Midi Key : 45)

- Data 1-5

  Nomor Midi : 45



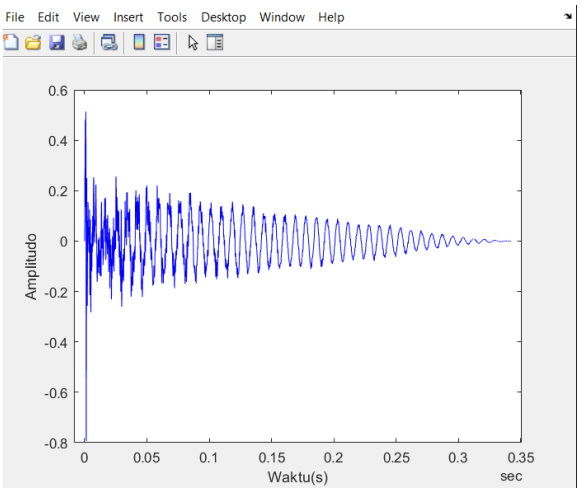


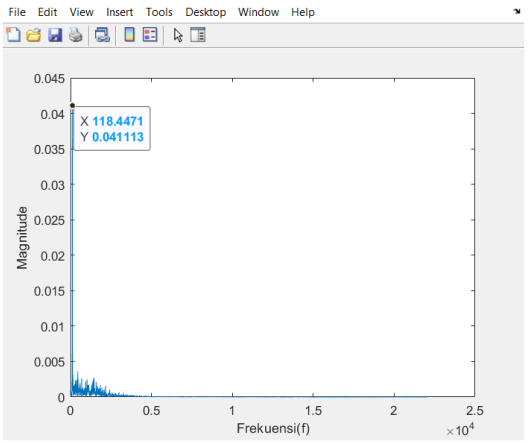


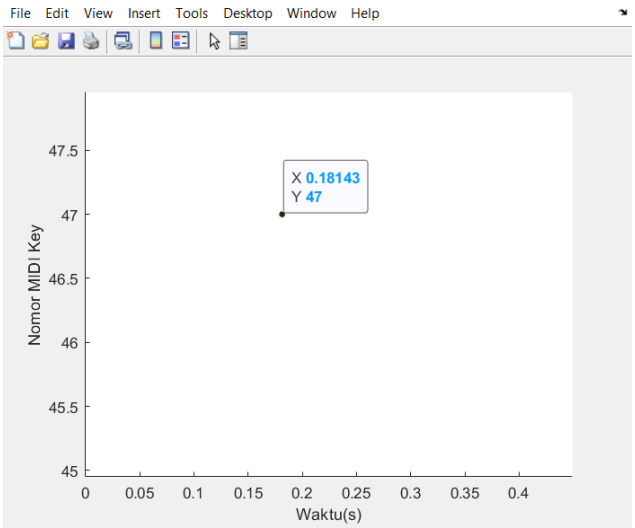
7. Mid Tom (Standar Midi Key : 47)

- Data 1-5

  Nomor Midi : 47



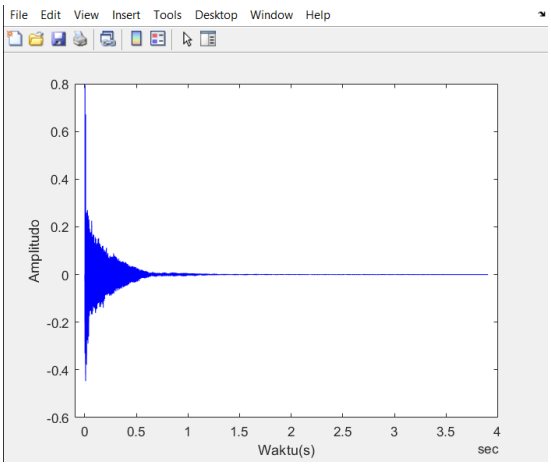


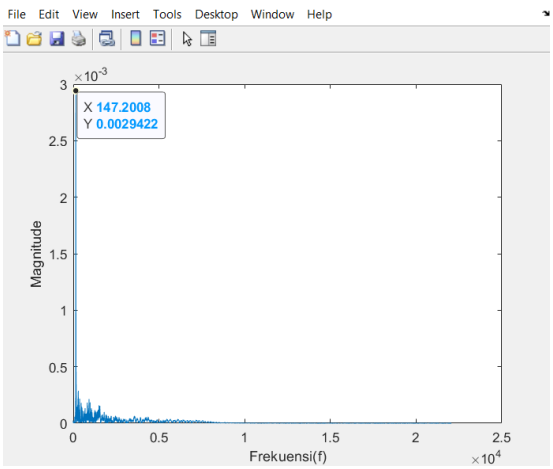


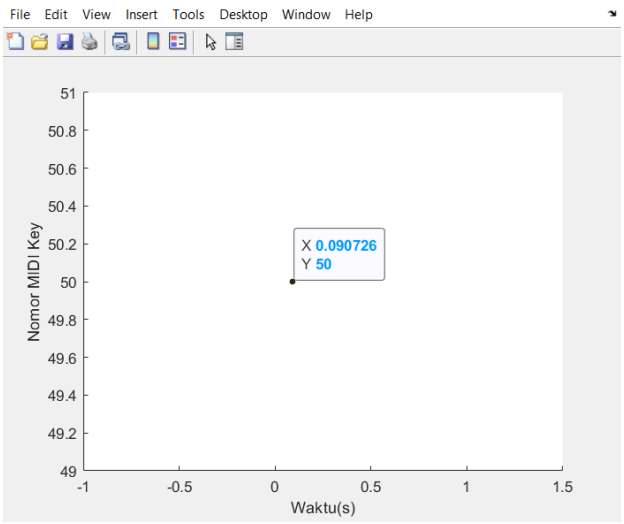
8. High Tom (Standar Midi Key : 50)

- Data 1-5

  Nomor Midi : 50







Lampiran 2. ***Source Code***

```
clear global; clear;

%Mendefinisikan variable
Fs=44100;            % sample rate
T=1/Fs;              % sample time
chunksize=8000;      % size of samples considered per FFT
stepsize=500;        % size of step - some overlap
namew='E:\Aplikasi Skripsi\Baru\High Tom 4.wav';   %
name of .wav file
namem='E:\Aplikasi Skripsi\Baru\High Tom 4.mid';   %
name of midi file
tol=10;              % tolerance of individual note pick
cl=chunksize/Fs;     % chunklength (secs)

%% Membaca data suara WAVE
y=audioread(namew);
info = audioinfo(namew); %
t = 0:seconds(1/Fs):seconds(info.Duration);
t = t(1:end-1);
figure('Name','File WAVE');
plot(t,y,'b')
xlabel('Waktu(s)')
ylabel('Amplitudo')
simpanpng1 = strcat(namew,'_FILEWAV.png');
saveas(gcf,simpanpng1)
% sound(y, Fs);            % plays sound
ns=size(y,1);            % number of samples


%% spectrum of whole .wav ------------------------------
--------------------
NFFT = 2^nextpow2(ns);               % next power of 2
from length of y
Y=fft(y(:,1), NFFT)/ns;              % Fast Fourier
Transform
```

```
f = Fs/2*linspace(0,1,NFFT/2);    % calculate frequency
%
figure('name', 'Magnitude')
plot (f, abs(Y(1:NFFT/2)));    % plot single sided
spectrum of whole.wav
xlabel('Frekuensi(f)')
ylabel('Magnitude')
% axis([0 20000 0 0.1]);



% split into overlapping chunks of samples and find freq
spectrum of each -
% ------------------------------------------------------
------------------
% noteinfo = [ frequency(hz), magnitude, starttime(sec),
stoptime(sec) ] --
% ------------------------------------------------------
------------------
noteinfo = zeros(1,4);

%y = y(:,1);
%dt = 1/Fs;
%t = 0:dt:(length(y)*dt)-dt;
%plot(t,y);
y2=0;
y3=[];
for i = 1:stepsize:ns-chunksize
  %sampling
    y2(:, end+1 ) = y(i,1);
    dt = 1/Fs;
    t = 0:dt:(length(y2)*dt)-dt;
  % Fast Fourier Transform
  NFFT = 2^nextpow2(chunksize);
  Y = fft(y(i:i+chunksize,1), NFFT)/(chunksize);
  f = Fs/2*linspace(0,1,NFFT/2+1);
  freqmag(:,1) = f;
  freqmag(:,2) = abs(Y(1:NFFT/2+1));

  % work out start and finish time of chunk
  chunkstart = i/Fs;    chunkend = (i + chunksize)/Fs;
  stepend = (i + stepsize)/Fs;              % start of
next chunk
```

```
%   %grafik hasil fft
%     y3(:, end+1 ) = Y(1:NFFT/2+1 ,1);
%     dt2 = 1/Fs;
%     t2 = 0:dt2:(length(y3)*dt2)-dt2;

 % if i==5001
 %   NFFT = 2^nextpow2(chunksize);              % next
power of 2 from length of y
 %   Y=fft(y(i:i+chunksize,1), NFFT)/chunksize;
% Fast Fourier Transform
 %   f = Fs/2*linspace(0,1,NFFT/2+1);    % calculate
frequency

 %   subplot(1,1,1)
  %plot (f, (Y(1:NFFT/2+1 ,1)));    % plot single sided
spectrum of whole.wav
 %   axis([0 20000 0 0.1]);
%  end


  if max(freqmag(:,2)>0.1) % only look for notes if
there is a frequency spike
    [row, col] = find(freqmag(:,2)>0.1);  % limit of mag
detection
    limitfreqmag = freqmag(row, :);

    sfm = zeros(1,2);                          % ##make sure
doesn't block any freq##
    for k=1:size(row,1)
      [row2, col2] =
find(limitfreqmag(:,2)==max(limitfreqmag(:,2)));   %
find max
      [row5, col5] =
find(sfm(:,1)<limitfreqmag(row2,1)+(tol*5) &
sfm(:,1)>limitfreqmag(row2,1)-(tol*5));
      if isempty(row5)==0;                    % if sfm
already has a similar freq
        limitfreqmag(row2,:) = [];            % delete
      else                                    % if its far
away
        sfm = [sfm; limitfreqmag(row2,:)];    % save
        limitfreqmag(row2,:) = [];            % delete
      end
    end
```

```
    sfm=[sfm(2:size(sfm,1),:)];              % ignore 1st
row

    for p=1:size(sfm,1)                      % now check
if all potential saves cropped up before
      [row3, col3] = find(noteinfo(:,1)<sfm(p,1)+tol &
noteinfo(:,1)>sfm(p,1)-tol);
      if isempty(row3)==0                          % if had
pitch before (so row3 is not empty)
          [row6,col6] = find( noteinfo(row3, 4) ==
max(noteinfo(row3, 4)) );    % find most recent note of
that freq
          if noteinfo(row3(row6),4) < chunkstart
% if old pitch is finished -> new note
            noteinfo = [noteinfo; sfm(p,:), chunkend-
(chunksize/(2*Fs)), chunkend-(chunksize/(2*Fs))];
          else                                       % if
old pitch is not finished -> update
            noteinfo(row3(row6),4) = chunkend-
(chunksize/(2*Fs));           % update end time
          end
      else                                         % if
completely new pitch -> new note
        noteinfo = [noteinfo; sfm(p,:), chunkend-
(chunksize/(2*Fs)), chunkend-(chunksize/(2*Fs))];
      end
    end
  end
end
noteinfo=[noteinfo(2:size(noteinfo,1),:)];  % ignore 1st
row of zero's

%plot sampling
figure('Name','File Sampling');
plot (t, y2);
xlabel('waktu(s)')
ylabel('Audio Signal')
simpanpngsampling = strcat(namew,'_FILESAMPLING.png');
saveas(gcf,simpanpngsampling)

%plot sampling
%figure('Name','File FFT');
%plot (t2, y3);
```

```
%xlabel('Time')
%ylabel('FFT')
%simpanpngsampling = strcat(namew,'_FILESAMPLING.png');
%saveas(gcf,simpanpngsampling)

T1 = array2table(noteinfo);
T1.Properties.VariableNames(1:4) =
{'Frequency','Magnitude','Start_time','End_time'}
% delete any notes with dur < 0.1 secs
%noteinfo(:,5) = noteinfo(:,4) - noteinfo(:,3);
[r,c] = find(noteinfo(:,4) - noteinfo(:,3) < 0.1);
%noteinfo(r,:) = [];


% midi matrix ----------------------------------------
------------------
% ----------------------------------------------------
------------------
Mmini = freq2midi2([noteinfo(:,1)]); % calculate midi
pitch and bend mag.
N = size(noteinfo,1);                % number of notes
M = zeros(N,6);
M(:,1) = 1;                          % all in track 1
M(:,2) = 1;                          % all in channel 1
M(:,3) = Mmini(:,1);                 % nearest whole
midi note
M(:,4) = 90;                         % velocity   %
###this will be noteinfo(:,2)### with scaling fac (max
vel is 127?)
M(:,5) = noteinfo(:,3);              % time note on
(secs)
M(:,6) = noteinfo(:,4);              % time note
off(secs)
M(:,7) = Mmini(:,2);        % extra column-how much bend
note MSB
M(:,8) = Mmini(:,3);        % extra column-how much bend
note LSB

% save midi file  ------------------------------------
------------------
% ----------------------------------------------------
------------------
midi_new = matrix2midi2(M);
```

```
writemidi(midi_new, namem);    % save calculated midi
figure('Name','Frekuensi WAVE');
bar(noteinfo(:,3),noteinfo(:,1));
xlabel('Waktu(s)')
ylabel('Frekuensi(f)')
simpanpng2 = strcat(namew,'_FILEWAVFFT.png');
saveas(gcf,simpanpng2)
figure('Name','Nomor MIDI Key');
scatter(noteinfo(:,3),M(:,3),'filled');
xlabel('Waktu(s)')
ylabel('Nomor MIDI Key')
simpanpng3 = strcat(namew,'_FILEMIDI.png');
saveas(gcf,simpanpng3)
figure('Name','Frekuensi WAV dan MIDI');
a=(2.^((M(:,3)-69)/12)*440);
bar(noteinfo(:,3),[noteinfo(:,1),a]);
xlabel('Waktu(s)')
ylabel('Frekuensi(f)')
simpanpng4 = strcat(namew,'_FILEFREKWAVMIDI.png');
saveas(gcf,simpanpng4)
N = size(noteinfo,1);
WAVMID = zeros(N,4);
WAVMID(:,1) = noteinfo(:,3);
WAVMID(:,2) = noteinfo(:,1);
WAVMID(:,3) = M(:,3);

T2 = array2table(WAVMID);
T2.Properties.VariableNames(1:4) =
{'Start_Time','Frequency','Midi_Key','Note'};
```