Practical-1 (dft)

BSP PRACTICAL 1

clc;

clear all;

Fs = 1000; % sampling frequency 1 kHz

t = 0 : 1/Fs : 0.296; % time scale

f = 200; % Hz, embedded dominant frequency

x = cos(2\*pi\*f\*t) + randn(size(t)); % time series

plot(t,x), axis('tight'), grid('on'), title('Time series'), figure

nfft = 512; % next larger power of 2

y = fft(x,nfft); % Fast Fourier Transform

y = abs(y.^2); % raw power spectrum density

y = y(1:1+nfft/2); % half-spectrum

[v,k] = max(y); % find maximum

fprintf('Maximum value is: %f\n', v);

f\_scale = (0:nfft/2)\* Fs/nfft; % frequency scale

plot(f\_scale, y),axis('tight'),grid('on'),title('Dominant Frequency')

f\_est = f\_scale(k); % dominant frequency estimate

fprintf('Dominant freq.: true %f Hz, Estima

practical-2 (PP interval & RR interval)

BSP Practical 2

CODE:

h=60/72;

for a=0:h:2h

ECGdur=0:0.001:2.5;

Pint=0.11;

PRint=0.17;

STseg=0.10;

QTint=0.40;

QRSint=0.09;

pPeak=0.25;

rPeak=1.6;

qPek=.25\*rPeak;

sPeak=.35\*rPeak;

tPeak=0.3;

%genertion of p Wave

x=0:0.001:Pint;

y=pPeak\*sin(x\*pi/Pint)

plot (a+x,y);

grid on

hold on

xlabel('time(a)');

ylabel('voltage(mv');

title('ECG with 72bpm');

%genertion of pr segment

x=Pint:0.001:PRint;

plot(a+x,0);

%qrs Complex

q1=PRint+0.015;

r1=PRint+0.045;

s1=PRint+0.075;

s2=PRint+QRSint;

x=PRint:0.001:q1;

y=-26.667\*x+4.53333;

plot(a+x,y)

x=q1:0.001:r1;

y=66.667\*x-12.733;

plot(a+x,y)

x=r1:0.001:s1;

y=-72\*x+17.08;

plot(a+x,y)

x=s1:0.001:s2;

y=37.333\*x-9.706;

plot(a+x,y)

%st segment

t1=PRint+QRSint+STseg;

x=s2:0.0001:t1;

plot(a+x,0)

%Twave

t2=PRint+QTint;

x=t1:0.001:t2;

y=tPeak\*sin((x-t1)\*pi/(t2-t1));

plot(a+x,y)

x=t2:0.0001:h;

plot(a+x,0)

xlabel('time(s)');

ylabel('voltage(mV');

title('ECG wave');

end;

% Calculate R-R interval

RR\_interval = r\_locs (2) r\_locs(1);

% Display intervals on the plot

text(mean([p\_locs (1), p\_locs (2)]), pPeak + 0.1, [num2str(PP\_interval ), 's'], 'Color', 'b', 'FontSize', 12); text(mean([r\_locs (1), r\_locs (2)]), rPeak + 0.2, [num2str(RR\_interval ), 's'], 'Color', 'r', 'FontSize', 12);

% Draw P-P interval line

plot([p\_locs(1), p\_locs (2)], [pPeak, pPeak], 'k-', 'LineWidth', 0.5);

% Draw R-R interval line

plot([r\_locs(1), r\_locs (2)], [rPeak, rPeak], 'k-', 'LineWidth', 0.5);

BSP PRACTICAL 3 (import EMG signal)

CODE

clc;

clear all;

close all;

fid = fopen('emg\_healthy.txt','r');

emg = fscanf(fid,'%f');

fclose(fid);

fs=1000;

Y=fft(emg);

N=length(Y);

f=linspace(0,fs,N);

Y(1)=0;

[~,idx]=max(abs(Y));

freq=f(idx);

fprintf('Dominant frequency:%f Hz\n', freq)

Practical 4 (import ECG signal)

t = linspace(0, 1, 500); % Time Vector

EEG = rand(10, 500)\*0.005; % Simulate EEG (mV)

ofst = [1:size(EEG,1)]\*0.005 + 0.001; % ‘Offset’ Vector

EEGp = bsxfun(@plus, EEG', ofst)'; % Add ‘Offset’ To Each Row

figure(1)

plot(t, EEGp) % Plot EEG

axis([xlim 0 0.055]) % Set Axis Limits

ChC = regexp(sprintf('Ch-%02d ', [1:size(EEG,1)]), ' ', 'split'); % Y-Tick Labels

yt = ofst+0.0025; % Y-Yick Positions

set(gca, 'YTick',yt, 'YTickLabel',ChC(1:end-1))

Practical 5 (import EMG signal calculate AVR value to EMG signal)

code:-

clc;

close all;

clear all;

%problem 11.1

%%import the data in the fil\tsclient\E\BIOM480A3\Hw5\p11\_1.xls'e and plot the signal

problem11\_1 = load('C:\Users\admin\Downloads\b2\b2\emg\_healthy.txt');

t = problem11\_1(:, 1);

y1 = problem11\_1(:, 2);

N = length(y1);% find the length of the data per second

ls = size(y1); %% size

f = 1/N;% find the sampling rate or frequency

fs = 3000;

T = 1/fs % period between each sample

t1 = (0 : N-1) \*T;%t = (0:1:length(y1)-1)/fs; % sampling period

Nyquist = fs/2; figure;

subplot (3,1,1), plot(t,y1,'b');

title ('EMG signal of single muscle 40 month old patient ');

xlabel ('time (sec)');

ylabel ('Amplitute (V)');

grid on;

Y= abs(fft(y1)); Y(1) = [];

power = abs(Y(1:N/2)).^2;

nyquist = 1/(2\*0.001); freq = (1:N/2)/(N/2)\*nyquist;

subplot(212), plot(freq,power), grid on xlabel('Sample number (in Frequency)')

ylabel('Power spectrumen');

title({'Single-sided Power spectrum' ...

' (Frequency in shown on a log scale)'});

axis tight

%%% RMS of the signal

rms\_y1 = sqrt(mean(y1.^2));

msgbox(strcat('RMS of EMG signal is = ',mat2str(rms\_y1), ''));

rms\_emg = rms (y1);

%%%%%AVR of the signal

arv\_y1 = abs(mean(y1));

msgbox(strcat('ARV of EMG signal is = ',mat2str(arv\_y1), ''));

Practical 6 (import EMG signal , determine frequency pectrum or power spectrum )

clc;

close all;

clear all;

% Import the CSV file correctly

problem11\_1 = readtable('/MATLAB Drive/BSP/EMG-data.csv');

% Extract time and EMG signal data

t = problem11\_1.time;

y1 = problem11\_1.channel1;

% Find the length of the data per second

N = length(y1);

ls = size(y1); % Size of y1

f = 1/N; % Sampling rate or frequency

fs = 3000;

T = 1/fs; % Period between each sample

t1 = (0 : N-1) \* T; % Time vector

Nyquist = fs / 2;

% Plot EMG Signal

figure;

subplot(3,1,1), plot(t, y1, 'b');

title('EMG signal');

xlabel('time (sec)');

ylabel('Amplitude (V)');

grid on;

% Compute Power Spectrum

Y = abs(fft(y1));

Y(1) = [];

power = abs(Y(1:N/2)).^2;

nyquist = 1 / (2 \* 0.001);

freq = (1:N/2) / (N/2) \* nyquist;

% Plot Power Spectrum

subplot(212), plot(freq, power), grid on;

xlabel('Sample number (in Frequency)')

ylabel('Power spectrum');

title({'Single-sided Power Spectrum' ...

' (Frequency shown on a log scale)'});

axis tight;

practical – 7 (design a filter to remove noise)

CODE:

x = ecg(5theta\*theta) ;

ysgolayfilt(x,0,5);

[M,N] = size(y);

Fs = 1000

TS = timescope('SampleRate', Fs,...

'TimeSpanSource', 'Property',...

'TimeSpan', 1.5,...

'ShowGrid', true,

NumInputPorts', 2,...

'LayoutDimensions', [2 1]);

TS.ActiveDisplay = 1;

TS. YLimits = [- 1, 1] ;

TS. Title 'Noisy Signal';

TS.ActiveDisplay = 2 ;

TS. YLimits = [- 1, 1] ;

TS. Title Filtered Signal';

Fpass = 200;

Fstop = 400 ;

Dpass 0.05;

Dstop 0.0001;

F=[0 Fpass Fstop Fs/2]/(Fs/2);

A = [[1, 1, 0]]

D [Dpass Dstop);

b firgr('minorder', F,A,D);

LP dsp. FIRFilter('Numerator',b);

Fstop = 200;

Fpass = 400;

Dstop 0.0001;

Dpass = 0.05;

F = [0 Fstop Fpass Fs/2]/(Fs/2); % Frequency vector

A = [0011]; % Amplitude vector

D = [Dstop Dpass]; % Deviation (ripple) vector

b = ('minord', F, A,D);

HP = dsp.FIRFilter('Numerator',

while toc < 30

x = 0.1 randn(M,N);

highFreqNoise = HP(X)

noisySignal = y + highFreqNoise;

filteredSignal LP(noisySignal);

TS(noisySignal, filteredSignal);

end