Exercise 9 TTK4165 Medical Signal Processing

Even Florenes Spring 2016

Contents

Documentation

Purpose:

Script answering tasks given in exercise 9 in the course TTK4165 Medical Signal Processing

Related files:

```
imagelog.m: Image a matrix of ultrasound power in log scale
```

Made by:

```
Even Florenes NTNU 2016
```

Last changes:

```
   2016-03-25 EF: First attempt on part 2,3 and 4

   2016-03-30 EF: First attempt on part 5,6

   2016-03-31 EF: Finished all parts, commented, updated documentation
                  and published
```

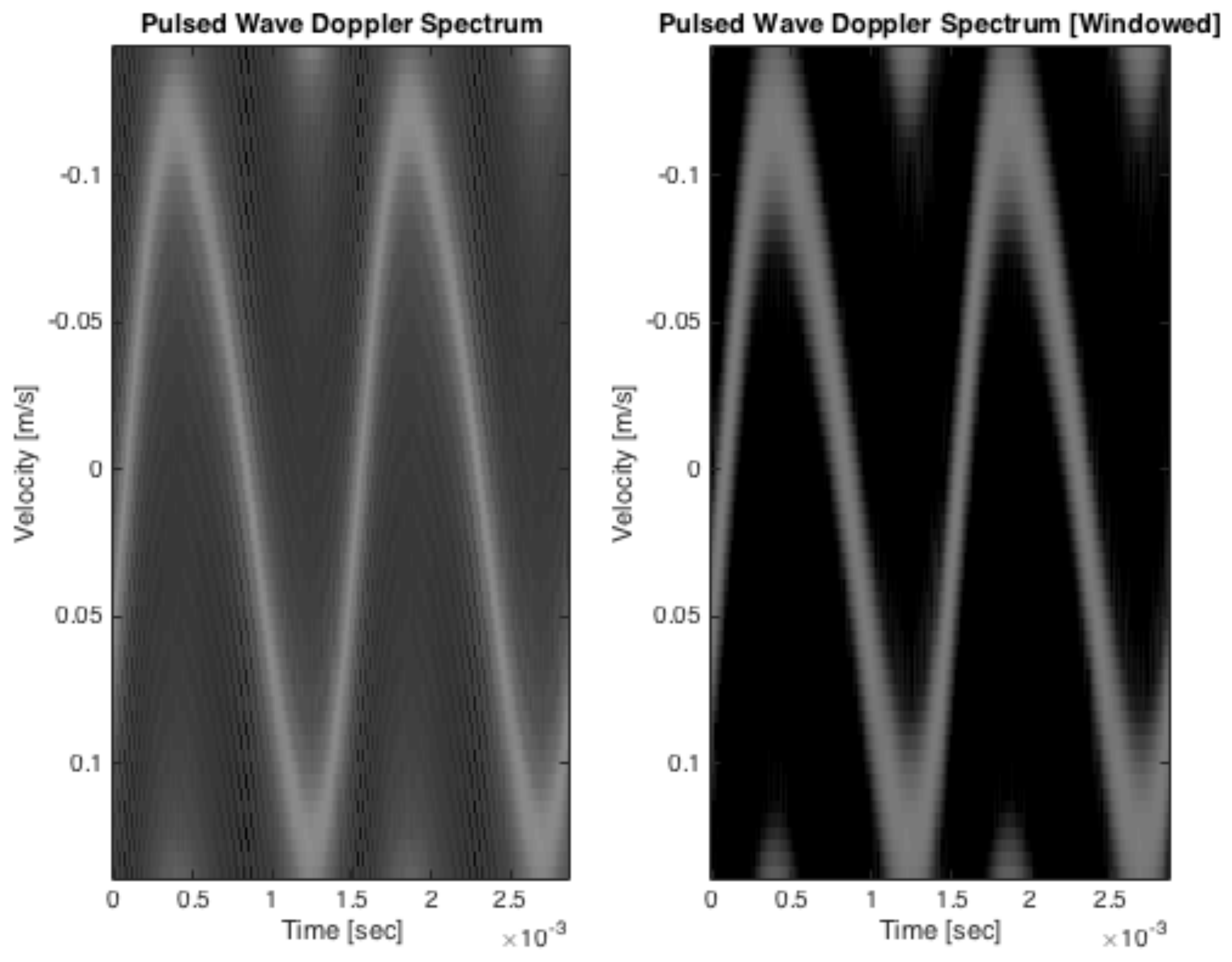
Status:

```
Works
```

Part 2 Pulsed Wave Doppler w/ analytic velocity

```
load slowmotion

% Find middle beam
middleBeamIq = squeeze(iq(:,4,:));

frameRate = s.Framerate_fps; % nFrames/seconds

nFrames = size(middleBeamIq,2); %nFrames

nSamples = size(middleBeamIq,1);
```

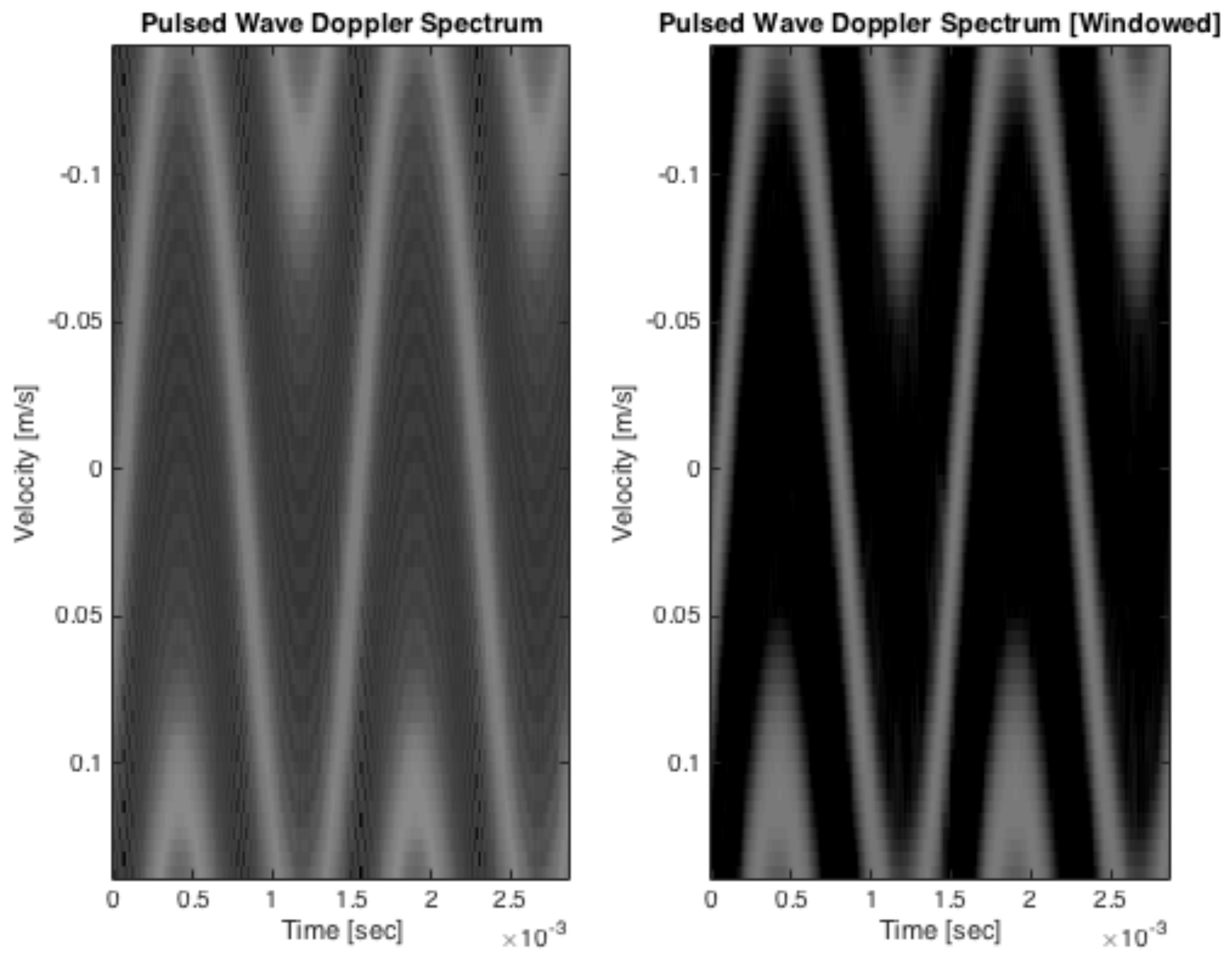
```matlab
nSeconds = nFrames/frameRate;    %seconds = (nFrames/(nFrames/seconds))

time = 0:nSeconds/(nFrames-1):nSeconds;

distanceLength = s.iq.DepthIncrementIQ_m;

distance = 0:distanceLength/(nSamples-1):distanceLength;

% Find analytic velocity
rotationPeriod=0.908;
t0=0.0708;
excenterDistance=0.67;

pistonAngularFrequency = (2*pi*(time-t0))/rotationPeriod;
pistonVelocityAmplitude = -(2*pi*excenterDistance)/rotationPeriod;

pistonVelocity = pistonVelocityAmplitude*sin(pistonAngularFrequency);
pointVelocity = -pistonVelocity;



% Make Pulsed Wave Doppler Spectrum
Nfft=64; %Zeropadding to length 64
crop=16;
depthMiddle = round(size(middleBeamIq,1)/2);
depthindex = depthMiddle-9:depthMiddle+10;
PHamming=zeros(Nfft, nFrames-crop+1);
P=zeros(Nfft, nFrames-crop+1);
for n=1:nFrames-crop+1,
    middleBeamIqFrames=middleBeamIq(depthindex,n+[0:crop-1])';
    P(:,n) = mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
    middleBeamIqFrames=middleBeamIqFrames.*(hamming(crop)*ones(1,length(depthindex)));
    PHamming(:,n)=mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
end;
%Frequency axis
frequencyAxis=distanceLength*(([0:Nfft-1]/Nfft)-0.5)*frameRate;

%Greyscale image of frequency specter in dB
gain = -25;
dynamicRange = 40;

timeAxis = 0:(1/frameRate)/(size(PHamming,2)-1):(1/frameRate);
PHamming=imagelog(PHamming,gain,dynamicRange);
P = imagelog(P,gain,dynamicRange);
figure(1);
% Plot image without windowing
subplot(1,2,1),image(timeAxis,frequencyAxis,P),colormap(gray(64));
hold on
subplot(1,2,1),plot(time,pointVelocity,'w'),title('Pulsed Wave Doppler Spectrum'),xlabel('T
ime [sec]'),...
    ylabel('Velocity [m/s]');
% Plot image with hamming windowing
subplot(1,2,2),image(timeAxis,frequencyAxis,PHamming),colormap(gray(64));
hold on
subplot(1,2,2),plot(time,pointVelocity,'w'),title('Pulsed Wave Doppler Spectrum [Windowed]'
),xlabel('Time [sec]'),...
    ylabel('Velocity [m/s]');

%
% Comments on image:
% Looking at the image you can see that the doppler frequency is
% higher then the maximum allowed by Nyquist criterion. Which leads to
% aliasing. The aliasing is shown in the image by the amplitude of the
% velocity exceeding the window, and the remaining of the amplitude is
% flipped to the opposite side of the window.
%
% Using hamming window leads to less noisy image.
%
```

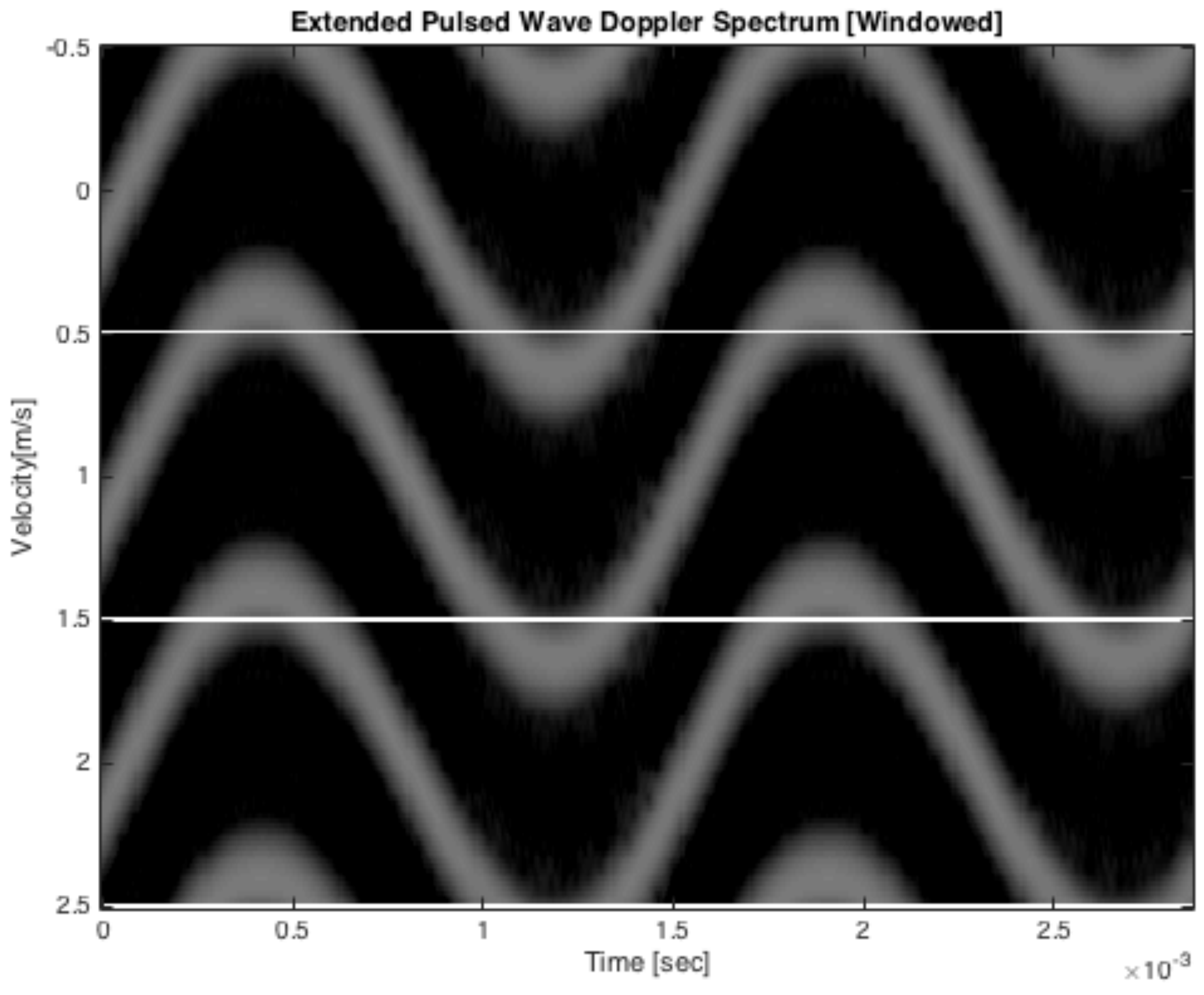


Part 3 - Doppler shift and aliasing

```
load fastmotion.mat

% Find middle beam iq
middleBeamIq = squeeze(iq(:,4,:));

frameRate = s.Framerate_fps; % nFrames/seconds

nFrames = size(middleBeamIq,2); %nFrames

nSamples = size(middleBeamIq,1);

nSeconds = nFrames/frameRate;   %seconds = (nFrames/(nFrames/seconds))

time = 0:nSeconds/(nFrames-1):nSeconds;

distanceLength = s.iq.DepthIncrementIQ_m;

distance = 0:distanceLength/(nSamples-1):distanceLength;

% From suggested solution exercise 8
x=[0.0419;0.356;0.675];y=[2.75;4.12;2.78];
excenterDistance=(y(2)-y(1))/2;
```

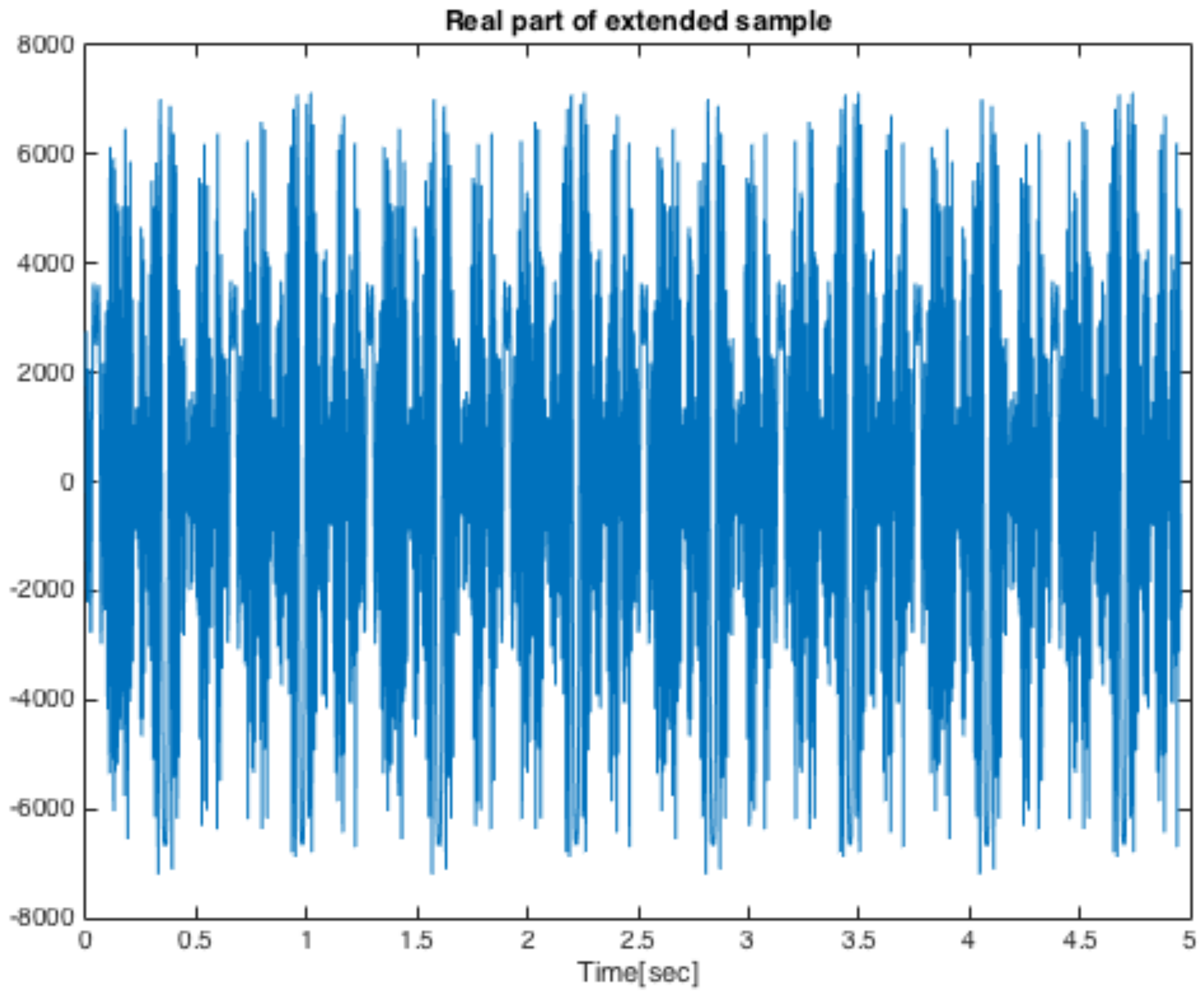
```
rotationPeriod=x(3)-x(1);
t0=x(1);%R in cm, T and t0 in seconds

pistonAngularFrequency = (2*pi*(time-t0))/rotationPeriod;
pistonVelocityAmplitude = -(2*pi*excenterDistance)/rotationPeriod;

pistonVelocity = pistonVelocityAmplitude*sin(pistonAngularFrequency);
pointVelocity = -pistonVelocity;

% Make Pulsed Wave Doppler Spectrum
Nfft=64; %Zeropadding to length 64
crop=16;
depthMiddle = round(size(middleBeamIq,1)/2);
depthindex = depthMiddle-9:depthMiddle+10;
PHamming=zeros(Nfft, nFrames-crop+1);
P=zeros(Nfft, nFrames-crop+1);
for n=1:nFrames-crop+1,
    middleBeamIqFrames=middleBeamIq(depthindex,n+[0:crop-1])';
    P(:,n) = mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
    middleBeamIqFrames=middleBeamIqFrames.*(hamming(crop)*ones(1,length(depthindex)));
    PHamming(:,n)=mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
end;
%Frequency axis
frequencyAxis=distanceLength*(([0:Nfft-1]/Nfft)-0.5)*frameRate;

%Greyscale image of frequency specter in dB
gain = -25;
dynamicRange = 40;

timeAxis = 0:(1/frameRate)/(size(PHamming,2)-1):(1/frameRate);
PHamming=imagelog(PHamming,gain,dynamicRange);
P = imagelog(P,gain,dynamicRange);
figure(2);
% Plot image without windowing
subplot(1,2,1),image(timeAxis,frequencyAxis,P),colormap(gray(64));
hold on
subplot(1,2,1),plot(time,pointVelocity,'w'),title('Pulsed Wave Doppler Spectrum'),xlabel('T
ime [sec]'),...
    ylabel('Velocity [m/s]');
% Plot image with hamming windowing
subplot(1,2,2),image(timeAxis,frequencyAxis,PHamming),colormap(gray(64));
hold on
subplot(1,2,2),plot(time,pointVelocity,'w'),title('Pulsed Wave Doppler Spectrum [Windowed]'
),xlabel('Time [sec]'),...
    ylabel('Velocity [m/s]');

PhammingExtendable = PHamming;
PhammingExtendable(end,:) = 64;
PExtended = [PhammingExtendable;PhammingExtendable;PhammingExtendable];
frequencyAxis=distanceLength*([0:Nfft-1]/Nfft)-0.5;
frequencyAxisStacked=[frequencyAxis,-2*min(frequencyAxis)+frequencyAxis+1,-4*min(frequencyA
xis)+frequencyAxis+1];
figure(3);
image(timeAxis,frequencyAxisStacked,PExtended),colormap(gray(64));
hold on
plot(time,pointVelocity,'w'),title('Extended Pulsed Wave Doppler Spectrum [Windowed]'),xlab
el('Time [sec]'),...
    ylabel('Velocity[m/s]');

% Comments on image:
% The Nyquist limit in the extended image is shown as a white line in the
% image.
%
```

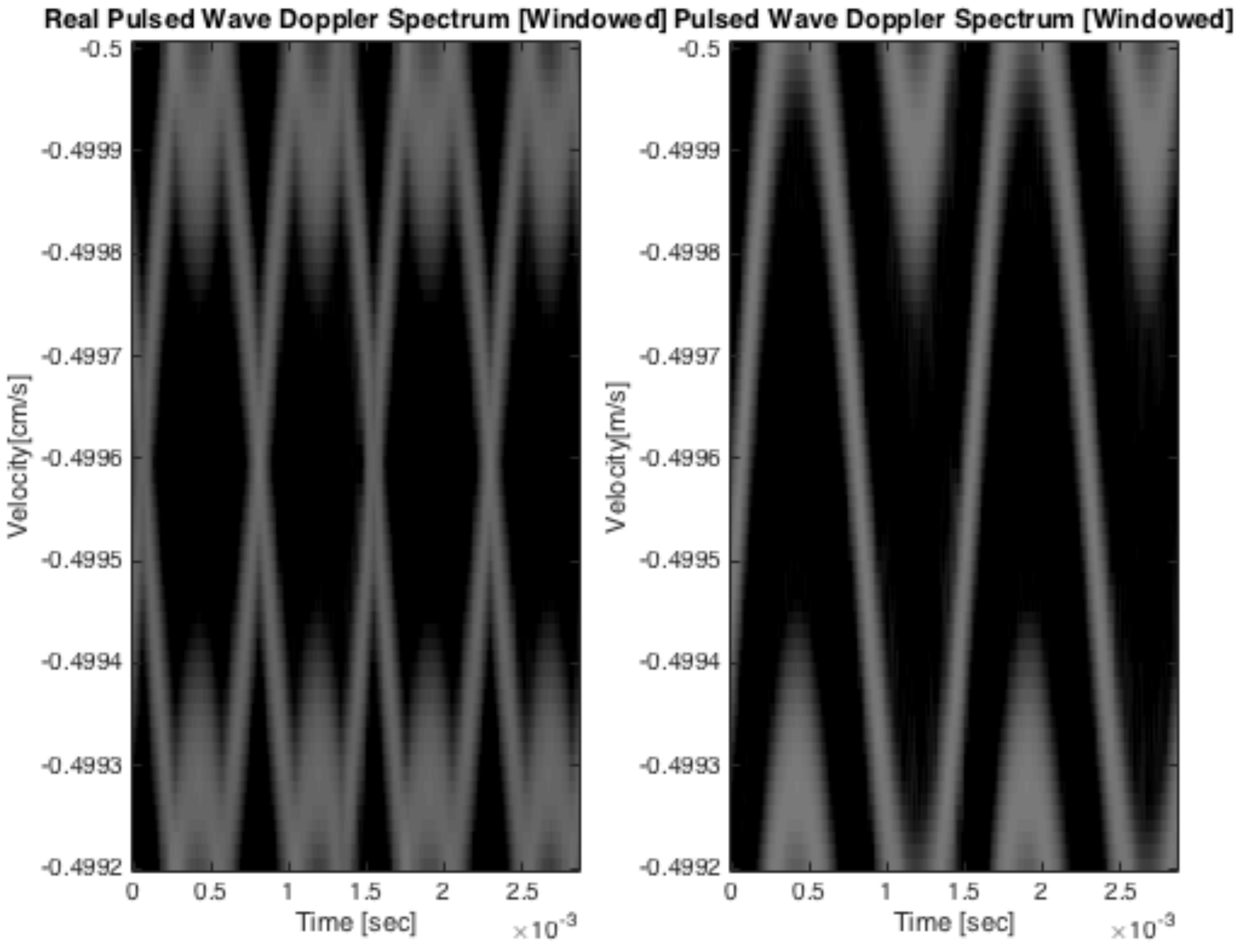




Part 4 - Doppler sound

```matlab
% Find middle sample
iqSample = middleBeamIq(round(size(middleBeamIq,1)/2),:);

% Extend sample, find real part, resample and rescale
iqSampleExtended = [iqSample,iqSample,iqSample,iqSample];
realSample = real(iqSampleExtended);
realSampleResampled = resample(realSample,8192,round(frameRate));
realSampleScaled = realSampleResampled/max(abs(realSampleResampled));

% Play hearable doppler frequency
soundsc(realSampleScaled,8192,8);

% Plot changes in time domain
timeExtended = 0:time(end)*4/(size(realSample,2)-1):time(end)*4;
figure(4);
plot(timeExtended,realSample),xlabel('Time[sec]'),...
    title('Real part of extended sample');

% Image FFT of real sample
PHammingReal=zeros(Nfft, nFrames-crop+1);
for n=1:nFrames-crop+1,
    middleBeamIqFrames=middleBeamIq(depthindex,n+[0:crop-1])';
    middleBeamIqFrames=middleBeamIqFrames.*(hamming(crop)*ones(1,length(depthindex)));
    PHammingReal(:,n)=mean(abs(fftshift(fft(real(middleBeamIqFrames),Nfft))),2);
end
timeAxis = 0:(1/frameRate)/(size(PHammingReal,2)-1):(1/frameRate);
PHammingReal=imagelog(PHammingReal,gain,dynamicRange);
figure(5),subplot(1,2,1),image(timeAxis,frequencyAxis,PHammingReal),colormap(gray(64));
title('Real Pulsed Wave Doppler Spectrum [Windowed]'),xlabel('Time [sec]'),...
    ylabel('Velocity[cm/s]');
subplot(1,2,2),image(timeAxis,frequencyAxis,PHamming),colormap(gray(64)),...
title('Pulsed Wave Doppler Spectrum [Windowed]'),xlabel('Time [sec]'),...
    ylabel('Velocity[m/s]');

% Comments on time plot:
% The time plot show that the extended sample looks many sinc-functions
% with varying amplitude and frequency.
%
%
%
% Comments on image:
%
% FFT of the real part of the signal produces an image where the velocity
% varies simultaneous as a negative and positive velocity. The two
% variations are perfectly matched.
%
%
```



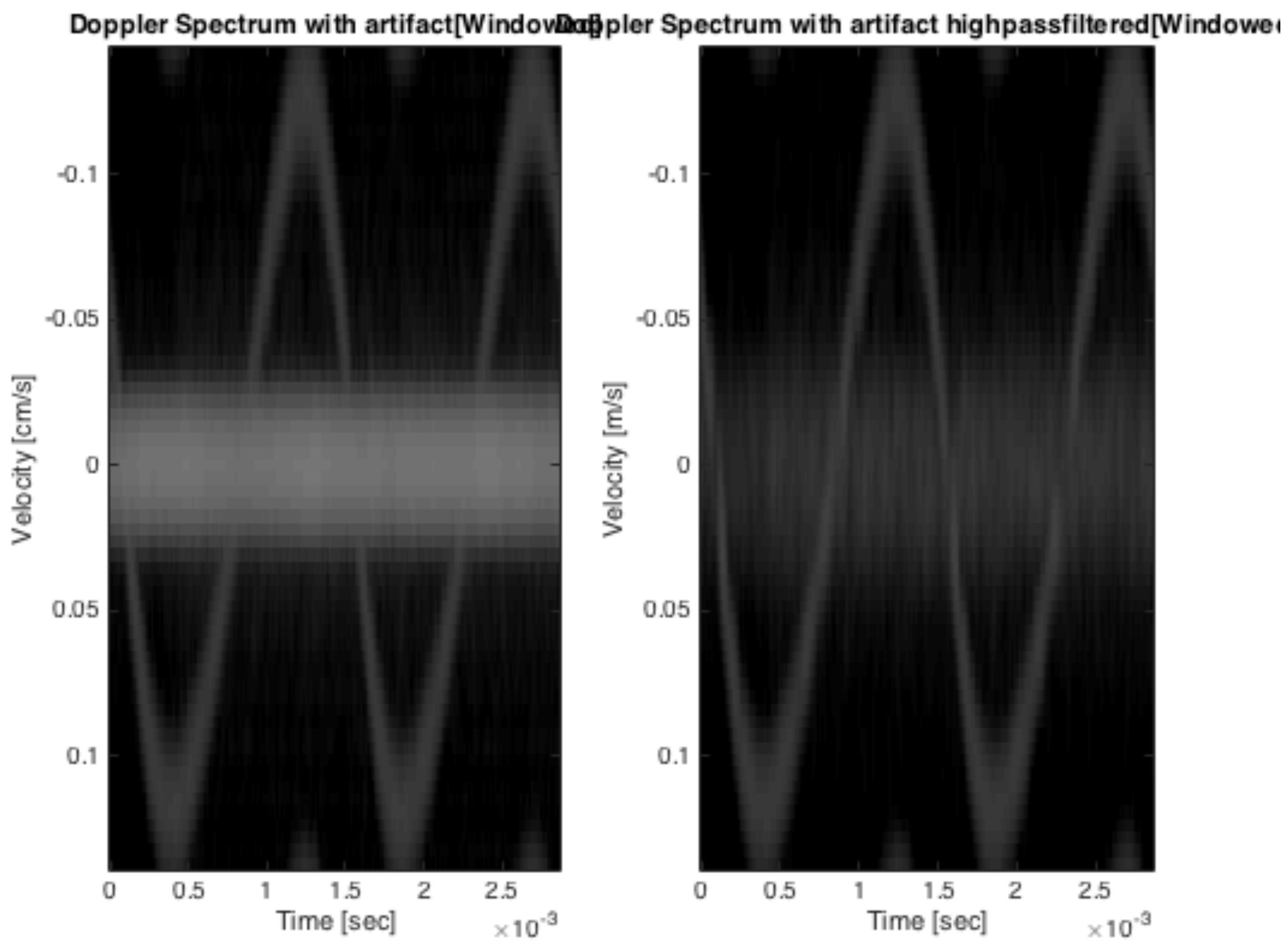


Part 5 - Clutter and clutter filter

```
load slowmotion_clutter

% Find middle beam
```

```
middleBeamIq = squeeze(iq(:,4,:));

frameRate = s.Framerate_fps; % nFrames/seconds

nFrames = size(middleBeamIq,2); %nFrames

nSamples = size(middleBeamIq,1);

nSeconds = nFrames/frameRate;   %seconds = (nFrames/(nFrames/seconds))

time = 0:nSeconds/(nFrames-1):nSeconds;

distanceLength = s.iq.DepthIncrementIQ_m;

distance = 0:distanceLength/(nSamples-1):distanceLength;

% Find analytic velocity
rotationPeriod=0.908;
t0=0.0708;
excenterDistance=0.67;

pistonAngularFrequency = (2*pi*(time-t0))/rotationPeriod;
pistonVelocityAmplitude = -(2*pi*excenterDistance)/rotationPeriod;

pistonVelocity = pistonVelocityAmplitude*sin(pistonAngularFrequency);
pointVelocity = -pistonVelocity;



% Make Pulsed Wave Doppler Spectrum
Nfft=64; %Zeropadding to length 64
crop=16;
depthMiddle = round(size(middleBeamIq,1)/2);
depthindex = depthMiddle-9:depthMiddle+10;
PHamming=zeros(Nfft, nFrames-crop+1);
for n=1:nFrames-crop+1,
    middleBeamIqFrames=middleBeamIq(depthindex,n+[0:crop-1])';
    middleBeamIqFrames=middleBeamIqFrames.*(hamming(crop)*ones(1,length(depthindex)));
    PHamming(:,n)=mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
end;
%Frequency axis
frequencyAxis=distanceLength*(([0:Nfft-1]/Nfft)-0.5)*frameRate;

%Greyscale image of frequency specter in dB
gain = -25;
dynamicRange = 40;

timeAxis = 0:(1/frameRate)/(size(PHamming,2)-1):(1/frameRate);
PHamming=imagelog(PHamming,gain,dynamicRange);
figure(6);
% Plot image without windowing
subplot(1,2,1),image(timeAxis,frequencyAxis,PHamming),colormap(gray(64));
hold on
subplot(1,2,1),plot(time,pointVelocity,'w'),title('Doppler Spectrum with artifact[Windowed]
'),xlabel('Time [sec]'),...
    ylabel('Velocity [cm/s]');

% Low pass filter
nFilterCoefficients = 8;
filterCoefficents=ones(1,nFilterCoefficients); %=boxcar(N). May also use hamming(N), hannin
g(N), ....
```

```
filterCoefficents=filterCoefficents/sum(filterCoefficents); %Normalization of filter coeffi
cients
iqLowPassFiltered=filter(filterCoefficents,1,middleBeamIq,[],2); %Filter along rows
iqHighPassFiltered=middleBeamIq-iqLowPassFiltered; %Subtract low pass component:

% Make Pulsed Wave Doppler Spectrum
Nfft=64; %Zeropadding to length 64
crop=16;
depthMiddle = round(size(middleBeamIq,1)/2);
depthindex = depthMiddle-9:depthMiddle+10;
PHamming=zeros(Nfft, nFrames-crop+1);
for n=1:nFrames-crop+1,
    middleBeamIqFrames=iqHighPassFiltered(depthindex,n+[0:crop-1])';
    middleBeamIqFrames=middleBeamIqFrames.*(hamming(crop)*ones(1,length(depthindex)));
    PHamming(:,n)=mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
end;
%Frequency axis
frequencyAxis=distanceLength*(([0:Nfft-1]/Nfft)-0.5)*frameRate;

%Greyscale image of frequency specter in dB
gain = -25;
dynamicRange = 40;

timeAxis = 0:(1/frameRate)/(size(PHamming,2)-1):(1/frameRate);
PHamming=imagelog(PHamming,gain,dynamicRange);
% Plot image without windowing
figure(6);
subplot(1,2,2),image(timeAxis,frequencyAxis,PHamming),colormap(gray(64));
hold on
subplot(1,2,2),plot(time,pointVelocity,'w'),title('Doppler Spectrum with artifact highpassf
iltered[Windowed]'),xlabel('Time [sec]'),...
    ylabel('Velocity [m/s]');

% Make sound of filtered and unfiltered

% Find middle sample
iqSample = middleBeamIq(round(size(middleBeamIq,1)/2),:);
iqSampleFiltered = iqHighPassFiltered(round(size(iqHighPassFiltered,1)/2),:);
% Extend sample, find real part, resample and rescale
iqSampleExtended = [iqSample,iqSample,iqSample,iqSample];
iqSampleExtendedFiltered = [iqSampleFiltered,iqSampleFiltered,iqSampleFiltered ,...
    iqSampleFiltered];
iqSamples = [iqSampleExtended;iqSampleExtendedFiltered];
for i = 1:size(iqSamples,1)
    realSample = real(iqSamples(i,:));
    realSampleResampled = resample(realSample,8192,round(frameRate));
    realSampleScaled = realSampleResampled/max(abs(realSampleResampled));

    % Play hearable doppler frequency
    soundsc(realSampleScaled,8192,8);
    pause(10);
end % for i

% Comments on clutter image:
% The slowmotion_clutter.mat gives a more noisy image. The image shows that
% there are lot of small movements in the imaged area, which leads to the
% quality of the large movement in the image is reduced, compared with
% slowmotion.mat.

% Comments on sound test:
% In the unfiltered sample the clutter movements leads to the sound of the
% blood flow being contaminated.
%
% The filtering removes some of the noise due to clutter movements, which
% makes the sound of the blood flow more hearable
%
```

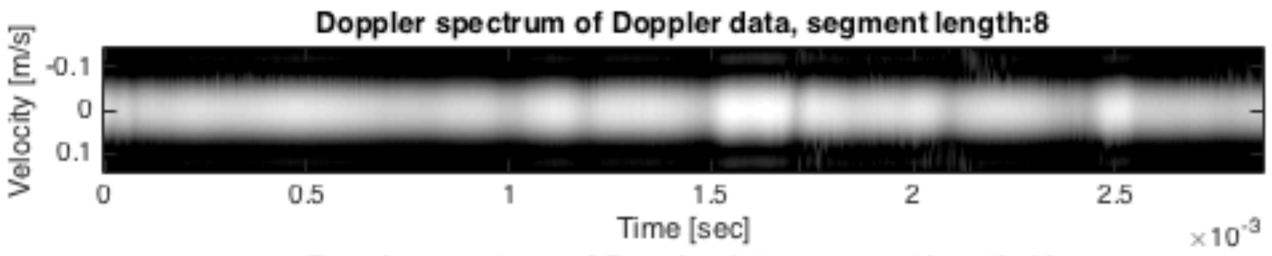


Part 6 - Blood flow measurement using Doppler

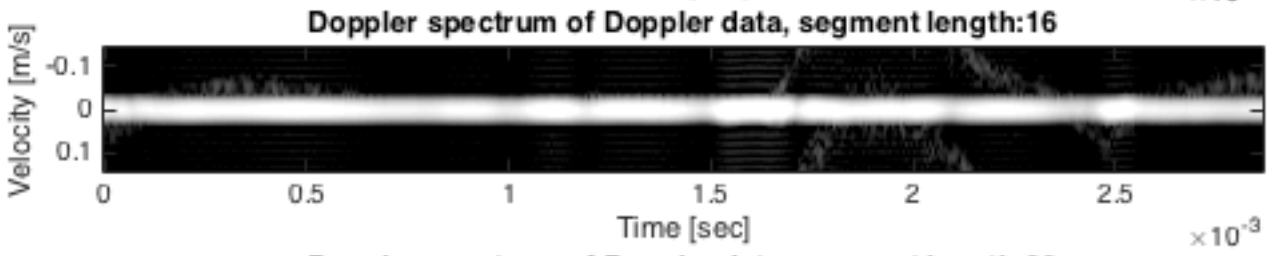
```
load Dopplerdata

% Find middle beam
middleBeamIq = iq;

frameRate = s.Framerate_fps; % nFrames/seconds

nFrames = size(middleBeamIq,2); %nFrames

nSamples = size(middleBeamIq,1);

nSeconds = nFrames/frameRate;   %seconds = (nFrames/(nFrames/seconds))

time = 0:nSeconds/(nFrames-1):nSeconds;

distanceLength = s.iq.DepthIncrementIQ_m;

distance = 0:distanceLength/(nSamples-1):distanceLength;


% Make Pulsed Wave Doppler Spectrum
Nfft=256; %Zeropadding to length 64
crops=[8,16,32,64];
for i = 1:length(crops)
```

```
    crop = crops(i);
    depthindex = [70:80];
    PHamming=zeros(Nfft, nFrames-crop+1);
    for n=1:nFrames-crop+1,
        middleBeamIqFrames=middleBeamIq(depthindex,n+[0:crop-1])';
        middleBeamIqFrames=middleBeamIqFrames.*(hamming(crop)*ones(1,length(depthindex)));
        PHamming(:,n)=mean(abs(fftshift(fft(middleBeamIqFrames,Nfft))),2);
    end;
    %Frequency axis
    frequencyAxis=distanceLength*(([0:Nfft-1]/Nfft)-0.5)*frameRate;

    %Greyscale image of frequency specter in dB
    gain = -20;
    dynamicRange = 20;

    timeAxis = 0:(1/frameRate)/(size(PHamming,2)-1):(1/frameRate);
    PHamming=imagelog(PHamming,gain,dynamicRange);
    figure(7);
    plotTitle = ['Doppler spectrum of Doppler data, segment length:' num2str(crop)];
    % Plot image without windowing
    subplot(length(crops),1,i),image(timeAxis,frequencyAxis,PHamming),colormap(gray(64)),..
.
        title(plotTitle),xlabel('Time [sec]'),...
        ylabel('Velocity [m/s]');
end
PHammingStakable = PHamming;
PHammingStakable(1,:) = 64;
PHammingStakable(end,:) = 64;
PHammingStacked = [PHammingStakable;PHammingStakable;PHammingStakable];
frequencyAxisStacked=[frequencyAxis,-2*min(frequencyAxis)+frequencyAxis+1,-4*min(frequencyA
xis)+frequencyAxis+1];
figure(8),image(timeAxis,frequencyAxisStacked,PHammingStacked),colormap(gray(64)),...
    title('Stacked Doppler spectrum of Doppler data, segment length: 64'),xlabel('Time[sec]
'),...
    ylabel('Velocity [m/s]');


centerFrequency = f0;
speedSound = 1540*100; %cm/s
pulseRepetition = prf;
nyquistSpeed = (speedSound*pulseRepetition)/(4*centerFrequency);
% Measured Nyquist limit:
% Nyquist limit: 33.5 cm/s
maxVelocity = 80.1;
pulseRepetitionForMaxVelocity = (4*centerFrequency*maxVelocity)/speedSound;
fprintf('Nyquist speed with given PRF: %g cm/s\n',nyquistSpeed);
fprintf('Measured maximum velocity: %g cm/s\n',maxVelocity);
fprintf('PRF needed to avoid aliasing: %.0f Hz\n',pulseRepetitionForMaxVelocity);
```

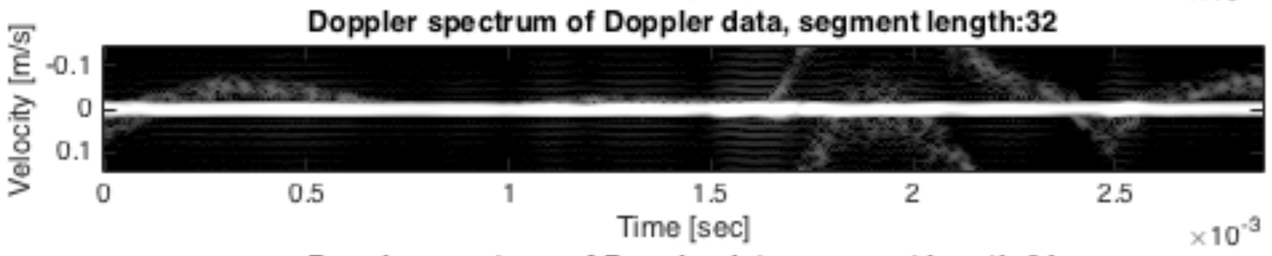
```
Nyquist speed with given PRF: 38.5 cm/s
Measured maximum velocity: 80.1 cm/s
PRF needed to avoid aliasing: 5201 Hz
```



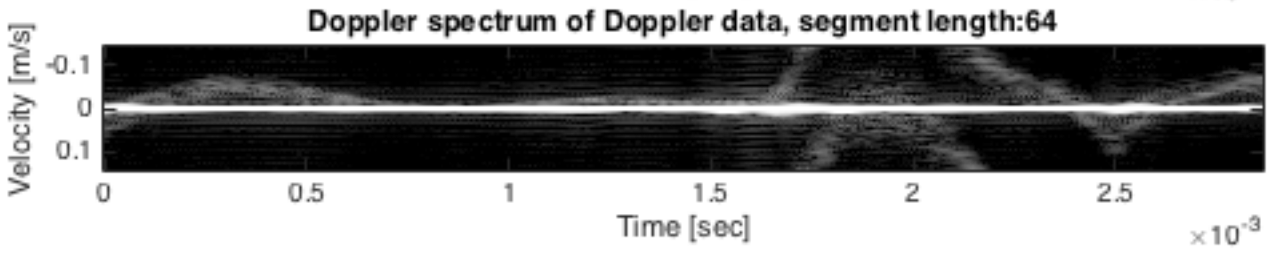
Doppler spectrum of Doppler data, segment length:8
Velocity [m/s]
-0.1
0
0.1
0
0.5
1
1.5
2
2.5
Time [sec]
×10^-3



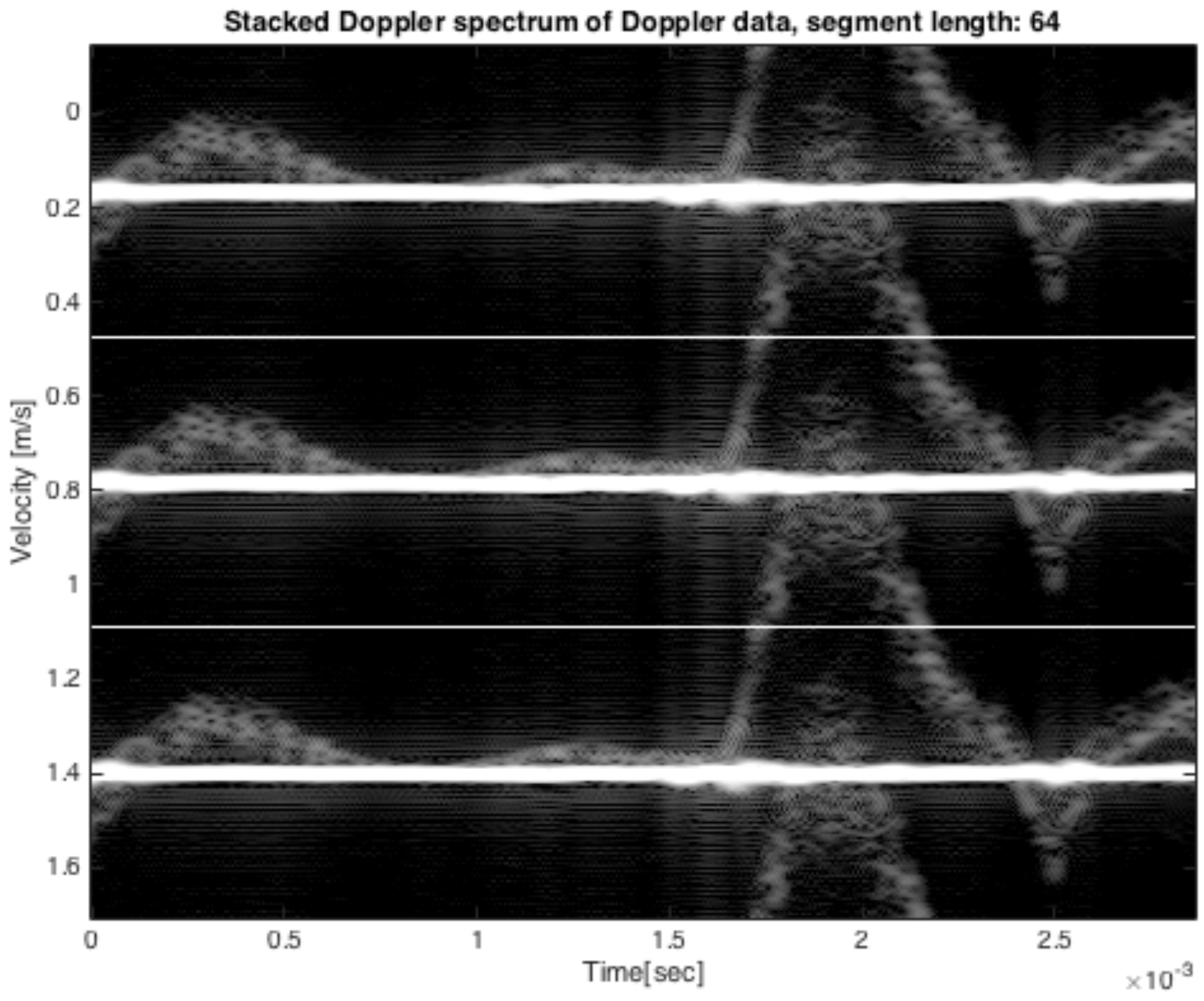
Doppler spectrum of Doppler data, segment length:16
Velocity [m/s]
-0.1
0
0.1
0
0.5
1
1.5
2
2.5
Time [sec]
×10^-3



Doppler spectrum of Doppler data, segment length:32
Velocity [m/s]
-0.1
0
0.1
0
0.5
1
1.5
2
2.5
Time [sec]
×10^-3



Doppler spectrum of Doppler data, segment length:64
Velocity [m/s]
-0.1
0
0.1
0
0.5
1
1.5
2
2.5
Time [sec]
×10^-3



Stacked Doppler spectrum of Doppler data, segment length: 64
Velocity [m/s]
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
0
0.5
1
1.5
2
2.5
Time[sec]
×10^-3