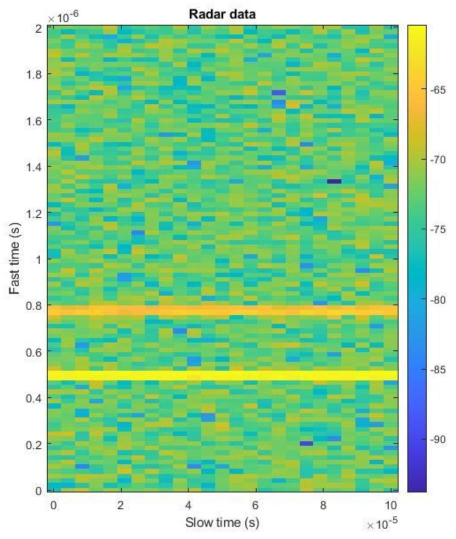
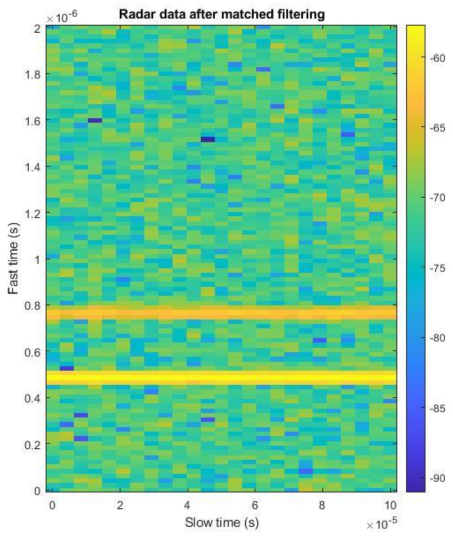
Part A

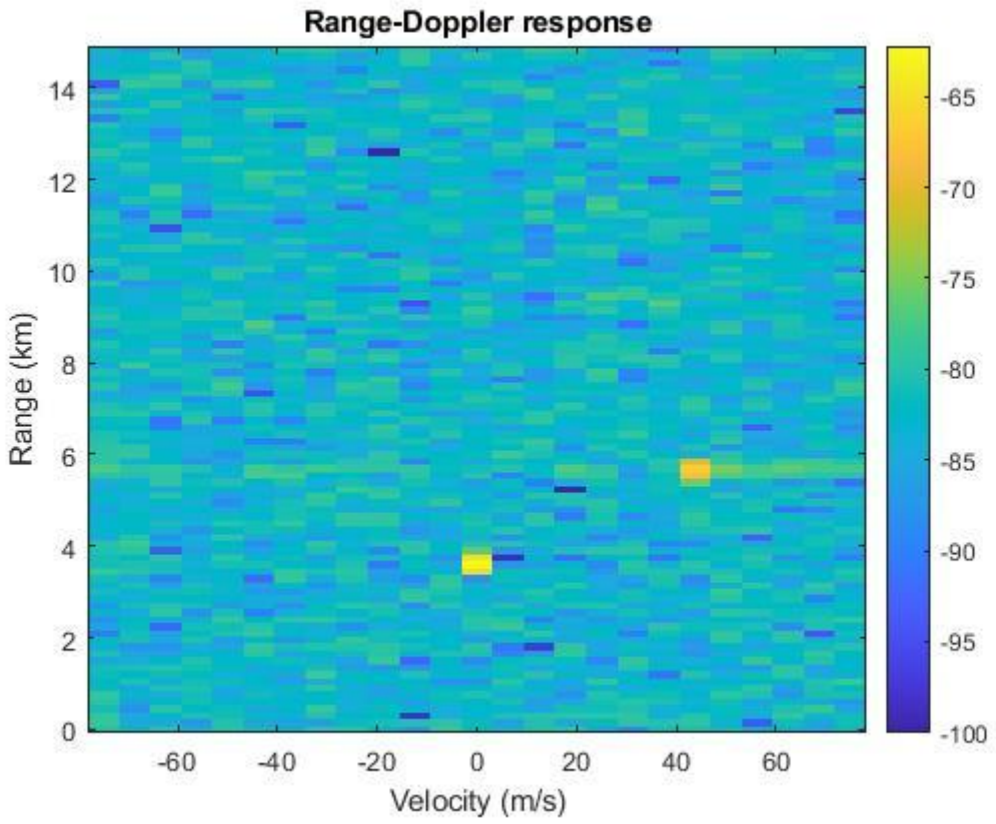
Implement fast-time matched filter processing, and plot the obtained radar data matrix

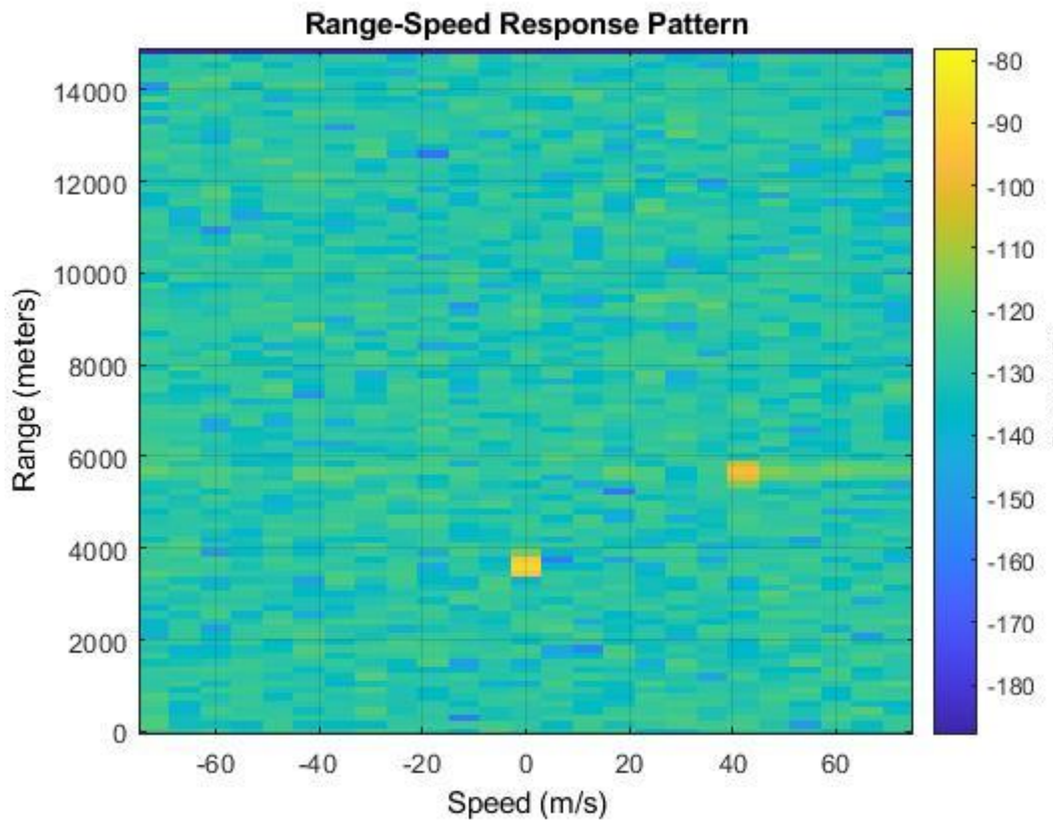




Part C

Implement slow-time Doppler processing, and plot the obtained radar data matrix





Part D

Determine maximum unambiguous range and velocity as well as range and velocity resolution for the setup.

- Max velocity = 74.9481 m/s
- Range resolution = 299.79m
- Velocity resolution = 5.9958 m/s
- Max unambiguous range = 14.99 kms

Part E

What happens if one target has a velocity/range higher than the maximum unambiguous velocity/range and explain.

The maximum unambiguous range is the longest range to which a transmitted pulse can travel out to and back again between consecutive transmitted pulses. Pulse Repetition Frequency give us number of pulses that are transmitted per second by radar system. Suppose the radar emits a pulse that strikes a target and returns to the radar in round trip time t and T= 1/PRF

If t > T then the return signal arrives after the next pulse has been emitted and there is an ambiguity, the radar cannot tell whether the return signal has come from the first or second pulse since target range is greater than maximum unambiguous range It will not be possible to determine from what pulse the echo resulted.

Matlab Code

```
clear all,
close all,
clc

antenna = phased.IsotropicAntennaElement('FrequencyRange',[5e9 15e9]);
transmitter = phased.Transmitter('Gain',20,'InUseOutputPort',true);
fc = 10e9;
T = 1/1e4;
txloc = [0;0;0];
rcs = [5 5];
tgtloc= [2000 4000; 3000 4000; 0 0];
tgtvelo = [0 -35; 0 -25; 0 0];

target =
phased.RadarTarget('Model','Nonfluctuating','MeanRCS',rcs,'OperatingFrequency',fc)
;
antennaplatform = phased.Platform('InitialPosition',txloc);
targetplatform = phased.Platform('InitialPosition',tgtloc, 'Velocity',tgtvelo);
[tgtrng,tgtang] =
rangeangle(targetplatform.InitialPosition,antennaplatform.InitialPosition);

waveform = phased.RectangularWaveform('PulseWidth',2e-
6,'OutputFormat','Pulses','PRF',1e4,'NumPulses',1);
c = physconst('LightSpeed');
maxrange = c/(2*waveform.PRF);
SNR = npwgnthresh(1e-6,25,'noncoherent');
lambda = c/target.OperatingFrequency;
maxrange = c/(2*waveform.PRF);
tau = waveform.PulseWidth;
Ts = 290;
dbterm = db2pow(SNR - 2*transmitter.Gain);
Pt = 1e4;

radiator =
phased.Radiator('PropagationSpeed',c,'OperatingFrequency',fc,'Sensor',antenna);
channel =
phased.FreeSpace('PropagationSpeed',c,'OperatingFrequency',fc,'TwoWayPropagation',
true);
collector =
phased.Collector('PropagationSpeed',c,'OperatingFrequency',fc,'Sensor',antenna);
receiver =
phased.ReceiverPreamp('NoiseFigure',0,'EnableInputPort',true,'SeedSource','Propert
y','Seed',2e3);

numPulses = 25;
rx_puls = zeros(100,numPulses);
filter =
phased.MatchedFilter('Coefficients',getMatchedFilter(waveform),'GainOutputPort',tr
ue);
response = phased.RangeDopplerResponse('DopplerFFTLengthSource',
'Property','DopplerOutput','Speed', 'OperatingFrequency',fc);
for n = 1:numPulses
    wf = waveform();
    [wf,txstatus] = transmitter(wf);
     wf = radiator(wf,tgtang);
     wf = channel(wf,txloc,tgtloc,[0;0;0],tgtvelo);
         wf = target(wf);
```

```
          wf = collector(wf,tgtang);
          rx_puls(:,n) = receiver(wf,~txstatus);
           [mf_puls, mfgain] = filter(rx_puls(:,n));
    Gd = length(filter.Coefficients)-1;
    mf_matrix(:,n) = [mf_puls(Gd+1:end);mf_puls(1:Gd)];
    [tgtloc,tgtvel] = targetplatform(1/1e4);
    [tgtrng,tgtang] = rangeangle(tgtloc,txloc);
end
figure
subplot(121)
imagesc(linspace(0,1/1e4,25),linspace(0,tau,100),10*log10(abs(rx_puls)))
set(gca,'YDir', 'normal')
xlabel('Slow time (s)')
ylabel('Fast time (s)')
title('Radar data')
colorbar()

subplot(122)
imagesc(linspace(0,1/1e4,25), linspace(0,tau,100), 10*log10(abs(mf_matrix)))
set(gca,'YDir','normal')
xlabel('Slow time (s)')
ylabel('Fast time (s)')
title('Radar data after matched filtering')
colorbar()
%% Part C
t = unigrid(0,1/receiver.SampleRate,T, '[)');
rangeGates = c*t/2;
f_vec = linspace(-1/(2*T),1/(2*T), numPulses);
dopp_matrix = zeros(100, numPulses);
for ii = 1:length(mf_matrix(:,1))
    dopp_matrix(ii, :) = fftshift(fft(mf_matrix(ii,:)));
    dopp_matrix(ii, :) = dopp_matrix(ii,:)*lambda/2;
end
vel_vec = f_vec*lambda/2;
figure
imagesc(vel_vec, rangeGates/1e3, 10*log10(abs(dopp_matrix)))
set(gca,'YDir','normal')
xlabel('Velocity (m/s)')
ylabel('Range (km)')
title('Range-Doppler response')
colorbar()
%% Part D
R_max = c/(2*1e4)
V_max = 1e4*lambda/4
range_res = c/2^waveform.PulseWidth
velocity_res = waveform.PRF/(2*numPulses)*(c/fc)


rangedoppler = phased.RangeDopplerResponse('RangeMethod','Matched
Filter','PropagationSpeed',c,'DopplerOutput','Speed','OperatingFrequency',fc);
figure
plotResponse(rangedoppler,rx_puls,getMatchedFilter(waveform))
```