Xiangyu Ding

ECE2595 Radar Signal Processing

Murat Akcakaya

3/22/2018

Project1 in Radar Signal Processing

In the first project of radar signal processing, the main goal is to use MATLAB to simulate the detection of targets' signal. In this project, we apply a CFAR detector in Weibull background using the maximum likelihood algorithm. The project1 has 5 main problems to solve.

I solve them step-by-step, each problem may share the same workspace. Please do not skip the former problem when running and analyzing the next problem.

The project runs in the following environment, different working environment may have different customized results (figure color, character style, time consuming, etc.)

System: Windows 10 Home basic edition 64 bit, Intel Core i7-6700HQ 2.60Ghz CPU, NVidia GTX980M GPU, 16.0 GB RAM

Software: MATLAB7.0(32bit)

1. **Open and load the file procNov11stare0.mat into MATLAB workspace. There are four matrices that correspond to different polar metric channels. Each matrix has 54 fast-time measurements and 2048 slow-time measurements. Each entry of these matrices is a complex number, whose real part and imaginary part are the in-pulse channel and the quadrature channel, respectively, of the matched filter output. Generate a magnitude image of the VV data as a function of the fast and slow time.**

This is a simple question. I use MATLAB command window to transfer VV data into absolute value and plot it in a magnitude image. The command code is

```
>>x=(2015:15:2820);
y=(1:1:2048);
z=abs(vv);
surf(x,y,z);
xlabel('Range(m)');
ylabel('Pulse');
zlabel('Magnitude')
```

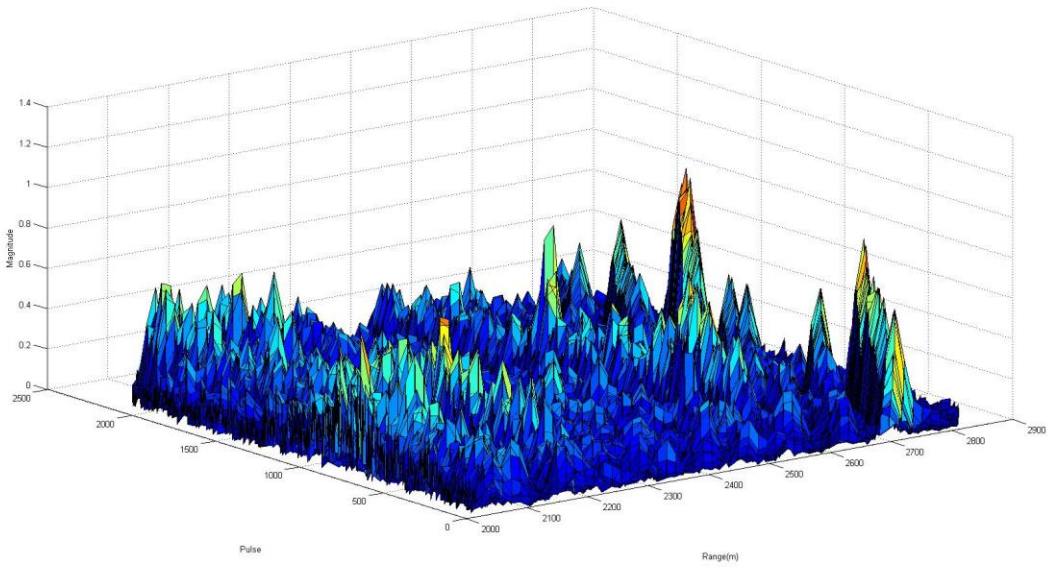

Figure 1.1 Magnitude image of VV data

2. Write a MATLAB script to implement the ML-CFAR detector described in Section 2.1 of reference [1], considering that the shape parameter is C = 2. For a given pulse, run the test along the down range. In order to define the secondary data, consider that

samples in the fast-time dimension are cyclic, i.e. cell 1 follows cell 54. Note that this ML-CFAR test is a linear envelop detector; then we process the magnitude of the data.

I define a function of ML-CFAR detector in known C=2. It is [x_Detect,T] = cfar(matrix,n,Pfa,M). parameter 'matrix' is the input signal, 'n' is the pulse of raw matrix data, 'Pfa' is the false alarm probability, M is the number of guard cells.

The more details and specific function is in cfar.m attached in my project1 files.

3. **Plot the magnitude of the pulse 180 of the matrix VV as a function of range. Use your script to compute and plot the detection threshold for PFA equal 10−2 and 10−3, and considering the number of secondary data M equal 16 and 32. Repeat the procedure for the pulse 155 of the matrix VV. Compare and discuss the results.**

I use my MATLAB script cfar.m to compute the threshold of pulse 180, PFA equals 0.01, M equals 16 as a representation. The command code is

```
absvv=abs(vv);

[x_det t]=cfar(absvv,180,1e-2,16);

x=2000:15:2000+15*(size(absvv,2)-1);

y1=absvv(180,:); % once change the pulse, the number 180 in this line should also be changed

y2=t;

y3=x_det;
```

```
plot(x,y1,x,y2,'*',x,y3,'o');

xlabel('Range(m)');

ylabel('Magnitude');

legend('Signal','Threshold','Detections','Location','Northwest')
```

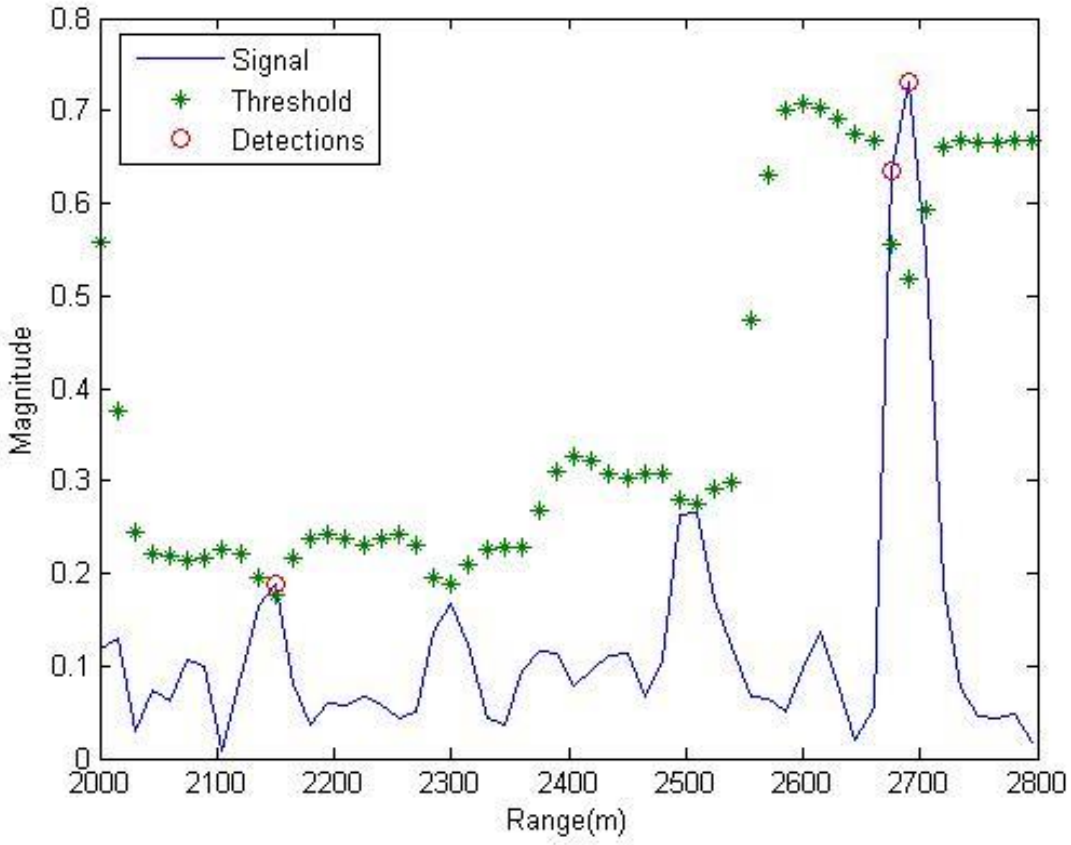

Figure 3.1 Detection in Pulse 180 [C=2, Pfa=0.01, M=16]

The rest of data share the same command code from this case. They only change the parameters ‘Pulse’, ‘Pfa’ and ‘M’. In conclusion with different pulse such as 155 and 180, also

different parameters Pfa=0.01 and 0.001, M=16 and 32, I find that almost every figure has false detections. I attach the rest of figures in my project1 files.

4. **Build a detection map as a function range and slow-time by running your script for every pulse of the dataset VV. Assign value one to the pixels of the map if the test detects a target (the magnitude of the matrix entry is larger than the threshold); or assign zero otherwise. Since you can guess where the target is located, use the other range cells to compute empirically the probability of false alarm ˆ PFA and compare with the design PFA used to define the threshold. Discuss the results.**

Based on the function cfar I created, I add a new 'for' loop to run each pulse from VV. There is a new function named Cfarx to calculate them. It is [x_Detect,T] = Cfarx(matrix,Pfa,M). As same as before, parameter 'matrix' is the input signal, 'Pfa' is the false alarm probability, M is the number of guard cells.

While I change some code in this new function for object in Problem 4, such as target value becomes 1 from input signal value, if there is no target, the x_detect becomes 0 from NaN and so on. The more details we can open m-files and compare them. Cfarx.m is attached in my project1 files.

The representation of magnitude map is below, its command code is

```
>> absvv=abs(vv);

[X_Detect,T]=Cfarx(absvv,0.01,16);

x=(2015:15:2820);
```

```
y=(1:1:2048);

z=X_Detect;

surf(x,y,z);

xlabel('Range(m)');

ylabel('Pulse');

zlabel('Magnitude')
```

The result displays as a map below,

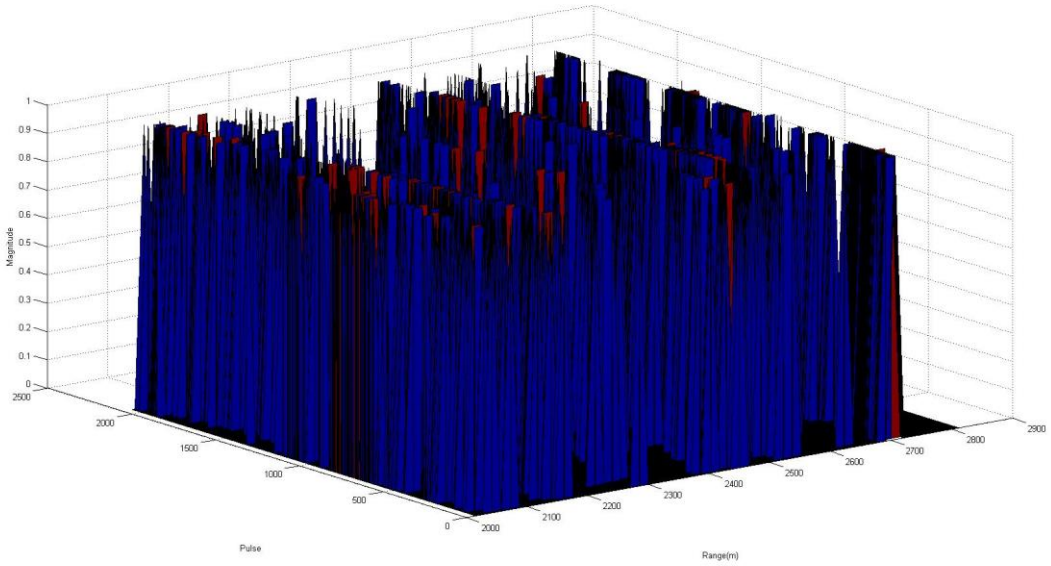

Figure 4.1 Detection Map [C=2, Pfa=0.01, M=16]

Then I use MATLAB command code to find where is the target, the command code is

```
>>[row,col]=find(X_Detect);
tabulate(col(:))
```

Then we gather a table shows that the column 47 (Range from 2675m to 2705m) is obviously the target position while there are also a lot of false detections. The detection table is attached in my project1 files.

For calculating the empirical Pfa based on the result in this case, I calculate the number of false detection k and divide k into total number of detection expect the true detection. The command code in this processing is

```
>> k=numel(find(X_Detect))-numel(find(X_Detect(:,47)));
PfaE=k/(size(vv,1)*(size(vv,2)-1))
```

Then we get the empirical false-alarm probability Pfa in this case. It is PfaE = 0.0250, it's larger than setting false-alarm probability Pfa = 0.01

The second case changes M from 16 to 32, PfaE changes bigger, it is PfaE = 0.0333

The third case changes Pfa from 0.01 to 0.001, PfaE = 0.0051

The forth case Pfa = 0.001, M = 32, its PfaE = 0.0127

All the map attached in my project1 files.

In conclusion, with the same false-alarm probability, when M becomes bigger, the empirical Pfa becomes bigger. Meanwhile, if the false-alarm probability becomes smaller, the empirical Pfa is much bigger than setting Pfa.

5. Write a MATLAB script to implement the ML-CFAR detector described in Section 3.1 of reference [1], considering that the shape parameter C is unknown. Build a detection processing the dataset VV for PFA equal 10−2 and 10−3, and considering the number of secondary data M equal 16 and 32. Compare with the former results.

In problem 5, I write 2 new functions Croot.m to solve the root of parameter C in eqn.29 of reference [1] and Cfars.m to apply result of Croot.m to simulate CFAR detector.

function F = Croot(matrix,n,M,C);

function [x_Detect,T] = Cfars(matrix,Pfa,M,C);

Parameter ‘matrix’ is the input signal matrix, ‘Pfa’ is the false alarm probability, M is the number of guard cells, ‘n’ is the number of pulse, ‘C’ is the shape parameter matrix.

The more details and specific functions are attached in my project1 files.

With unknown C and M=16, first, I build a type of equations to solve C, this processing costs about 10 minutes or less in MATLAB R2016a (it is faster than MATLAB7.0 but has bugs in detection processing) in my laptop. The command code is

```
>> x0 = ones(1,54);

for i =1:1:2048
```

```
C(i,:)=fsolve(@(C) Croot(z,i,16,C),x0)

end
```

Then detect the input signal, map the result, compute the PfaE as before. This processing command code is

```
>> absvv=abs(vv); % load C from my project workspace first

[X_DET,TT]=Cfars(absvv,0.01,16,C);

x=(2015:15:2820);

y=(1:1:2048);

z= X_DET;

surf(x,y,z);% plot map

xlabel('Range(m)');

ylabel('Pulse');

zlabel('Magnitude');

k=numel(find(X_DET))-numel(find(X_DET (:,47)));% computing PfaE

PfaE=k/(size(vv,1)*(size(vv,2)-1))
```

After that, I get the map and PfaE when

Pfa=0.01 and M=16. PfaE = 0.0243

Pfa=0.001 and M=16. PfaE = 0.0042

Pfa=0.01 and M=32. PfaE = 0.0233

Pfa=0.001 and M=32. PfaE = 0.0051

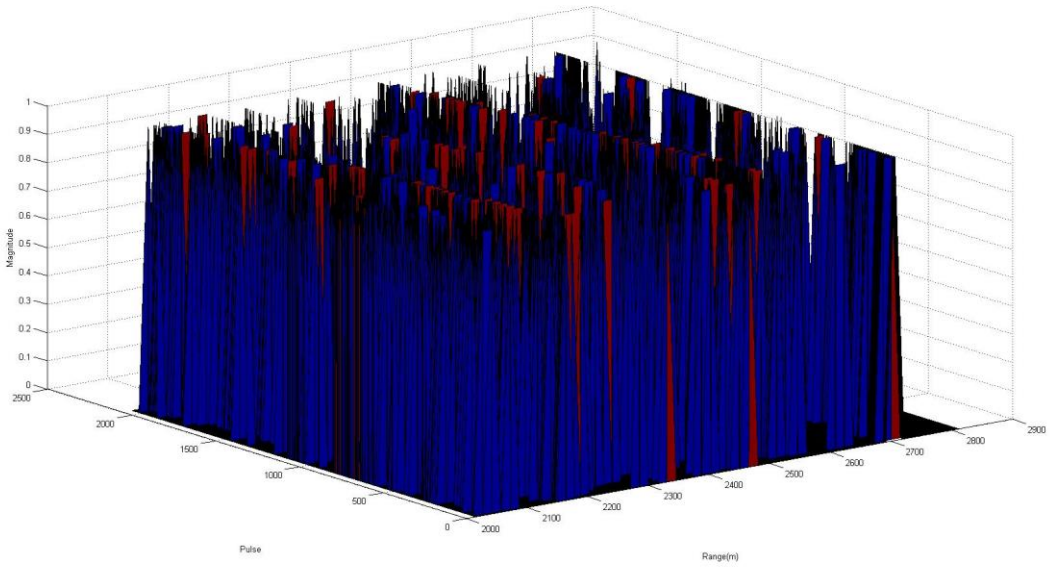

Figure 5.1 Detection Map [Unknown C, Pfa=0.01, M=16]

I save the shape parameter C when M=16 and 32 in my project1 files and the map results are also attached in my project1 files.

Comparing the results in problem 5 with the results in problem 4, when I compute every shape parameter C and apply them into my detector, the empirical Pfa becomes smaller. Observing the influence from M value, I find out that M influences smaller than before, which means this detector is more stable.

References

[1] R. Ravid and N. Levanon, “Maximum-likelihood CFAR for Weibull background,” Radar and Signal Processing, IEE Proceedings F, vol. 139, pp. 256-264, Jun. 1992.

[2] S. Haykin, C. Krasnor, T. Nohara, B. Currie, and D. Hamburger, “A coherent dual-polarized radar for studying the ocean environment,” IEEE Trans. Geosci. Remote Sens., vol. 29, no. 1, pp. 189–191, Jan. 1991.

[3] Webpage: http://soma.ece.mcmaster.ca/ipix/.