2D CFAR

Implement the 2D CFAR process on the output of 2D FFT operation, i.e the Range Doppler Map.

1. Determine the number of Training cells for each dimension. Similarly, pick the number of guard cells.

```
%Select the number of Training Cells in both the dimensions.
Tr = 10;
Td = 10;

%Select the number of Guard Cells in both dimensions around the Cell under
%test (CUT) for accurate estimation
Gr = 4;
Gd = 4;

% offset the threshold by SNR value in dB
offset = 1.5;
```

2. Slide the cell under test across the complete matrix. Make sure the CUT has margin for Training and Guard cells from the edges.

```
for i = Tr+Gr+1:(Nr/2)-(Gr+Tr)
    for j = Td+Gd+1:Nd-(Gd+Td)

       ..

    end
end
```

3. For every iteration sum the signal level within all the training cells. To sum convert the value from logarithmic to linear using db2pow function.

```
%Create a vector to store noise_level for each iteration on training cells
        noise_level = zeros(1,1);

% Calculate noise SUM in the area around CUT (sum of all training cells)
  for p = i-(Tr+Gr) : i+(Tr+Gr)
      for q = j-(Td+Gd) : j+(Td+Gd)
          if (abs(i-p) > Gr || abs(j-q) > Gd)
             noise_level = noise_level + db2pow(RDM(p,q));
             %y = db2pow(ydb) returns the power measurements, y,
             %that correspond to the decibel (dB) values specified in ydb.
             %The relationship between power and decibels is ydb = 10   log10(y).
          end
      end
  end
```

4. Average the summed values for all of the training cells used. After averaging convert it back to logarithmic using pow2db.

```
No_training_cells = ((2*Tr+2*Gr+1)*(2*Td+2*Gd+1))-((2*Gr+1)*(2*Gd+1));
threshold = pow2db(noise_level/No_training_cells);
```

5. Further add the offset to it to determine the threshold.

```
threshold = threshold * offset;
```

6. compare the signal under CUT against this threshold.

```
CUT = RDM(i,j);

  if (CUT < threshold)
      RDM(i,j) = 0;
  else
      RDM(i,j) = 1;
  end
```

7. To keep the map size same as it was before CFAR, equate all the non-thresholded cells to 0.

```
RDM(union(1:(Tr+Gr),end-(Tr+Gr-1):end),:) = 0;   % Rows
RDM(:,union(1:(Td+Gd),end-(Td+Gd-1):end)) = 0;   % Columns
```

8. display the CFAR output using the Surf function

```
figure('Name','CA-CFAR Filtered RDM')
surf(doppler_axis,range_axis,RDM);
colorbar;
```

9. the result: