EE6506 - Computational Electromagnetics Homework - 3

Srivenkat A (EE18B038), Srijan Gupta (EP17B009)

May 2, 2021

1 Introduction

The aim of this exercise is to compute the radar cross-section of an object of arbitrary shape. This would help us to understand the assumptions involved and design choices made in real world scenarios where the radar cross-section is computed for different aircraft designs.

In this exercise, we try to find the radar cross-section (or RCS) of a regular pentagon for an incident wave at a particular frequency and angle of incidence. Since the object can be represented in 2D as the properties of the object are invariant in the Z-direction, we will be using Surface Integral Methods to calculate the scattered fields from the object.

The incident field is taken to be a S-band radar wave at 3GHz, incident at an angle of 45° from the x-axis. The analysis is done in 2 different scenarios:

- Assuming the object to be made of a Perfect Electrical Conductor (PEC)
- Assuming the object to be made of Carbon Fiber

We calculate the 2D RCS of the object taking the incident wave to be of E_z polarisation. For this, we will be using Pulse basis, Delta testing approach to solve the SI equations. We define the discretisation length or the width of the basis function to be $\lambda/15$ where λ is the wavelength of the incident wave.

2 Modelling the object

We first model the pentagonal object in the 2D cartesian plane and then discretise it to get a set of points along the edges of the object. The size of the object is taken to 3λ on each side.

The 2D model is generated assuming that one vertex of the object is always along the + Y-axis. The following code generates the starting points of the pulse basis functions and the evaluation points of the delta testing function. Here 'theta' is the direction of the 'first' vertex.

```
%generates a regular polygon in 2D, oriented along theta, centred at origin
ang = 360/n_sides;
c = a/(2*sind(ang/2));      %length of line segment connecting centre to vertex
vrtx_A = [c*cosd(theta); c*sind(theta)];    %vertex pointing along theta
dir_AB = [-cosd(90 - (ang/2) - theta); sind(90 - (ang/2) - theta)];
rotat_ang = [cosd(ang) -sind(ang); sind(ang) cosd(ang)];
n_e = round(a/da);    %total no. of discretisation points

test_pt = zeros(2, n_sides*n_e);    %points of delta testing
strt_pt = zeros(2, n_sides*n_e);     %starting points of pulse basis
test_pt(:,1:n_e) = vrtx_A + [da/2:da:a].*dir_AB;
strt_pt(:,1:n_e) = vrtx_A + [0:da:a-da].*dir_AB;
for i = 1+n_e : n_e:n_sides*n_e
    test_pt(:,i:(i+n_e-1)) = rotat_ang*test_pt(:,i-n_e:i-1);
    strt_pt(:,i:(i+n_e-1)) = rotat_ang*strt_pt(:,i-n_e:i-1);
end
```

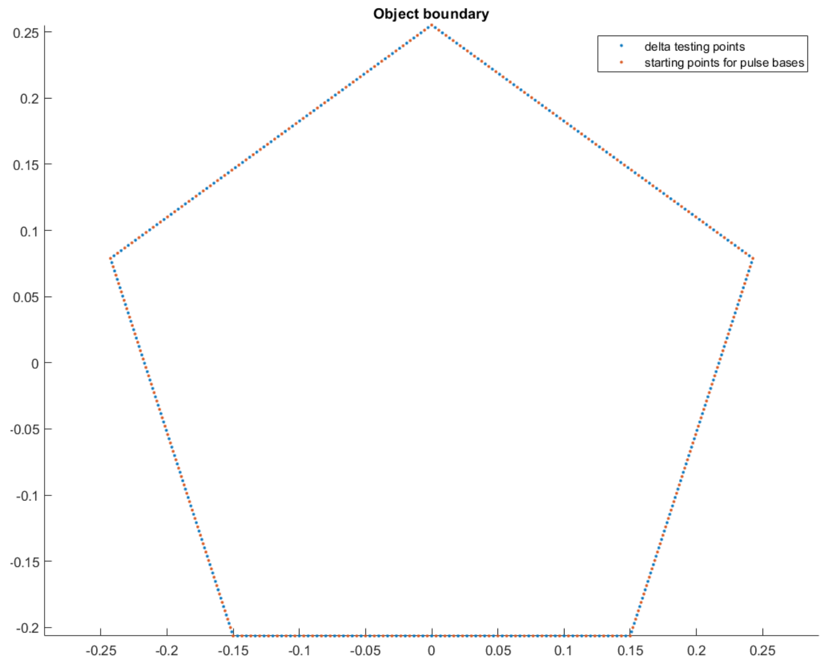

Figure 1: Modelling the Pentagonal Object

The function is written such that it will generate the discretisation points of any regular polygon. Since we analyse a pentagonal object, n_{sides}=5 here.

3 Framing the Surface Integral Equations

In the 2D problem setup, we are analysing the scattered fields for TM Polarisation (E_z, H_x, H_y are the only non-zero components of the incident field). As the object is Z-invariant, we define a new variable $\phi = E_z$. As the incident and scattered fields are assumed to be plane waves, if we know the transverse Electric field component, all the other components can be found out. We define ϕ_1 as E_z outside the object and ϕ_2 as E_z inside the object. So, ϕ_1 and ϕ_2 will obey the Helmholtz Equations:

$$\nabla^2\phi_1 + K_1^2\phi_1 = Q; \quad \nabla^2\phi_2 + K_2^2\phi_2 = 0 \tag{1}$$

where Q represents the contribution from the induced currents inside the object.
We define the 2D Green's functions as the solutions of the equation:

$$\nabla^2 g_1(r,r') + K_1^2 g_1(r,r') = -\delta(r-r'); \quad \nabla^2 g_2(r,r') + K_2^2 g_2(r,r') = -\delta(r-r') \tag{2}$$

On algebraically manipulating the above equations, we get:

$$\phi_i(r') - \oint [g_1(r,r')\boldsymbol{\nabla}\phi_1(r) - \phi_1(r)\boldsymbol{\nabla} g_1(r,r')]\cdot\hat{n}dl = \begin{cases}\phi_1(r') : r' \in V_1\\ 0 : r' \in V_2\end{cases} \tag{3a}$$

$$\oint [g_2(r,r')\boldsymbol{\nabla}\phi_2(r) - \phi_2(r)\boldsymbol{\nabla} g_2(r,r')]\cdot\hat{n}dl = \begin{cases}\phi_2(r') : r' \in V_2\\ 0 : r' \in V_1\end{cases} \tag{3b}$$

4 Formulating the MOM Matrix

We first use the Extinction theorem to solve for the fields on the boundary. We try to solve the following equations:

$$\phi_i(r') = \oint [g_1(r,r')\nabla\phi_1(r) - \phi_1(r)\nabla g_1(r,r')] \cdot \hat{n} dl \tag{4a}$$

$$0 = \oint [g_2(r,r')\nabla\phi_2(r) - \phi_2(r)\nabla g_2(r,r')] \cdot \hat{n} dl \tag{4b}$$

On applying Boundary Conditions of tangential field continuity at the interface, we get the conditions:

$$\phi_1(r) = \phi_2(r); \quad \nabla\phi_1 \cdot \hat{n} = \nabla\phi_2 \cdot \hat{n} \tag{5}$$

Using the pulse basis function, we discretise the two unknown variables, $\phi(r)$ and $\nabla\phi \cdot \hat{n}$ as

$$\nabla\phi \cdot \hat{n} = \sum_{n=1}^{N} a_n p_n(r); \quad \phi(r) = \sum_{n=1}^{N} b_n p_n(r) \tag{6}$$

where $p_n(r)$ is the n'th pulse basis function that is non-zero only in the n'th interval. So, we have 2N unknown variables: $\{a_n\}_1^N$ and $\{b_n\}_1^N$. We evaluate the equations 4a and 4b at the mid points of the N segments on the boundary (delta testing) and use the boundary integral method to frame 2N equations (N each in 4a and 4b).

So, we need to frame a 2N×2N matrix to solve for the 2N unknown variables. The corresponding n'th equations of 4a and 4b are:

$$\sum_{n=1}^{N} \left[a_n \int_{I_n} g_1(r,r'_m) - b_n \int_{I_n} \nabla g_1(r,r'_m) \right] dl = \phi_i(r'_m) \tag{7a}$$

$$\sum_{n=1}^{N} \left[a_n \int_{I_n} g_2(r,r'_m) - b_n \int_{I_n} \nabla g_2(r,r'_m) \right] dl = 0 \tag{7b}$$

where I_n is the n'th interval between the n'th and (n+1)'th discretisation points.
Therefore, when we represent this system of equations in Ax = B form, A has dimensions 2N×2N, x has dimensions 2N×1, B has dimensions 2N×1.

Finding elements of A, B and x:

Elements of A

We can divide the matrix A into 4 quadrants and have indices i,j each ranging from 1 to N such that in equations 7a and 7b,

- The 1st term in 7a will have its coefficients in 1st quadrant.
  So, $A[i,j] = \int_{I_j} g_1(r,r'_i)$
- The 2nd term in 7a will have its coefficients in 2nd quadrant.
  So, $A[i,j+N] = -\int_{I_j} \nabla g_1(r,r'_i)$
- The 1st term in 7b will have its coefficients in 3rd quadrant.
  So, $A[i+N,j] = \int_{I_j} g_2(r,r'_i)$
- The 2nd term in 7a will have its coefficients in 4th quadrant.
  So, $A[i+N,j+N] = -\int_{I_j} \nabla g_2(r,r'_i)$

Note that for the segment corresponding to $i = j$, there will be a singularity in g and $\boldsymbol{\nabla} g$ where $\vec{r} = \vec{r_i'}$. We have $g(\vec{r}, \vec{r_i'}) \propto H_0^2(\vec{r} - \vec{r_i'})$, and for $x << 1$

$$H_0^2(x) \sim 1 - \iota \frac{2}{\pi} \left(\ln\left(\frac{x}{2}\right) + \gamma \right) \tag{8}$$

where γ is the euler constant. The singularity due to $\ln(x)$ is an integrable singularity and does not pose a problem. So, $A[i, i]$ and $A[i + n, i]$ are calculated in the same way as per the above conditions. Whereas, in the second term, we have $\boldsymbol{\nabla} g(\vec{r}, \vec{r_i'}) \propto H_1^2(\vec{r} - \vec{r_i'})$, and for $x << 1$

$$H_1^{(2)}(x) \sim \frac{x}{2} + \iota \frac{2}{\pi} \frac{1}{x} \tag{9}$$

and the singularity due to $1/x$ gives a residue. To solve this problem we deform the boundaries into semicircles near the testing points. The sides on which the semicircles lie are governed by condition of the extinction theorem. Integrating on these semicircles gives,

$$\begin{aligned} - \int_{S_1} \mathrm{d}l \boldsymbol{\nabla} g_1 &= 0.5 \\ \text{and,} \quad - \int_{S_2} \mathrm{d}l \boldsymbol{\nabla} g_2 &= -0.5 \end{aligned} \tag{10}$$

Also note that the semicircles S1 and S2 are assumed to be infinitesimally small in radius. So, we need to numerically integrate $\boldsymbol{\nabla} g_1$ and $\boldsymbol{\nabla} g_2$ in the interval excluding the singularity. We split the interval length into 3 parts: 0 to $(0.5 - 10^{-5})$Dr, semicircle and $(0.5 + 10^{-5})$Dr to Dr. So,

$$A[i, i + n] = 0.5 - \int_{I_{left}} \boldsymbol{\nabla} g_1(r, r_i')dr - \int_{I_{right}} \boldsymbol{\nabla} g_1(r, r_i')dr \tag{11a}$$

$$A[i + n, i + n] = -0.5 - \int_{I_{left}} \boldsymbol{\nabla} g_1(r, r_i')dr - \int_{I_{right}} \boldsymbol{\nabla} g_1(r, r_i')dr \tag{11b}$$

Elements of B

The matrix B will have the constant terms. Since the upper half of matrix A corresponds to equation 7a and lower half to equation 7b, the top N terms in B will have the incident field (RHS of 7a) and bottom N terms will be 0 (RHS of 7b).
So, for i ranging from 1 to N, $B[i] = \phi_{inc}(i)$

Elements of x

Aligning with the elements in A, the left half of A has the variable a_n and right half of A has the variable b_n. So, top N elements in x will be $\boldsymbol{\nabla}\phi_1 \cdot \hat{n}$ and bottom N elements will be ϕ
So, for i ranging from 1 to N, $x[i] = a_i$ and $x[i + N] = b_i$

The following code frames the matrices and finds the unknown matrix x.

```
A = zeros(2*n, 2*n);
b = zeros(2*n,1);
for i = 1:n
    for j = 1:n
        Dr = strt_pt(:,j+1) - strt_pt(:,j);
        t_hat = Dr/norm(Dr);
        n_hat = [-t_hat(2), t_hat(1)];
        if i==j    %considering cases of singularity separately
            A(i, j)  = integral(@(d)green2d(k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);          %The g1 term (integrable singularity)
```

```
            A(i+n, j)= integral(@(d)green2d(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);         %The g2 term (integrable singularity)
            A(i,j+n) = 0.5 -1*integral(@(d)gradgreen2d_dot_n(k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,0.5-a_sc,'AbsTol',tolabs,'RelTol',tolrel) ...
                -1*integral(@(d)gradgreen2d_dot_n(k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.5+a_sc,1,'AbsTol',tolabs,'RelTol',tolrel);         %The grad(g1).n term
            A(i+n,j+n) = -0.5 -1*integral(@(d)gradgreen2d_dot_n(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,0.5-a_sc,'AbsTol',tolabs,'RelTol',tolrel)...
                -1*integral(@(d)gradgreen2d_dot_n(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.5+a_sc,1,'AbsTol',tolabs,'RelTol',tolrel);   %The grad(g2).n term
        else
            A(i, j)     = integral(@(d)green2d(k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);           %The g1 term
            A(i+n, j)   = integral(@(d)green2d(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);         %The g2 term
            A(i, j+n)   = -1*integral(@(d)gradgreen2d_dot_n(k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);   %The grad(g1).n term
            A(i+n, j+n) = -1*integral(@(d)gradgreen2d_dot_n(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);      %The grad(g2).n term
        end
    end
    b(i) = inc_field(test_pt(:,i), theta_i, k0);
end
A = da*A;    %scaling by the length of the segment since the integral iterated over d from 0 to 1
x = A\b;       %solve by gaussian elimination
```

where, test_pt are the testing centres and strt_pt are the starting points of the pulse basis functions.

5 Computing RCS

As mentioned in the introduction, we analyse the RCS of the object for 2 cases, assuming it is made of a PEC and assuming it is made of Carbon fiber.
When we consider a PEC object, the $epsilon = \inf$ and the boundary conditions get simplified as the tangential electric field in a PEC has to be 0. So, we separately solve for this case.
When we consider it to be a PEC, we need to find the relative permittivity of the material. Assuming it is made of double-carbon microfoils, we get its electric permittivity as 12 - 5.5j at a frequency of 3GHz. This value is obtained from the following plot by Tengfei Chen et. al.

From the plot, the real part of epsilon is taken as 12 and imaginary part as 5.5 at 3GHz. Since they follow a $+j\omega t$ convention for forward propagating waves, the complex permittivity is obtained as 12-5.5j.

The 2D Radar Cross Section (RCS) is given by,

$$\sigma(\theta, \theta_i) = \lim_{r \to 0} 2\pi r \frac{|E_z^s(r,\theta)|^2}{|E_z^i(0,0)|^2} \tag{12}$$

So, we need to find the total fields due to scattering in the far field region. Once we know the fields at the boundary, which are calculated using the extinction theorem, we can calculate the

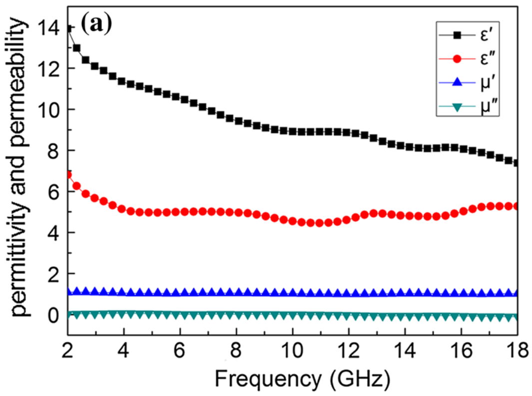

Figure 2: Finding eps of CF
Source : Reference 1

far fields using Huygen's principle according to Equations 3a and 3b. So,

$$\oint [g_1(r,r')\boldsymbol{\nabla}\phi_1(r) - \phi_1(r)\boldsymbol{\nabla}g_1(r,r')] \cdot \hat{n}dl = \phi_{scat}(r') \tag{13}$$

Once we get $\phi_2(r')$, we can use that in the RCS expression. Also note that we assume the incident field magnitude to be 1.

The following code uses the information about the field at the boundary to calculate the RCS.

```
for i = 1:n_ff
    for j = 1:n
        Dr = strt_pt(:,j+1) - strt_pt(:,j);
        t_hat = Dr/norm(Dr);
        n_hat = [-t_hat(2), t_hat(1)];
        integral_g1 = da*integral(@(d)green2d(k0, ff_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);
        integral_grad_g1_dot_n = da*integral(@(d)gradgreen2d_dot_n(k0, ff_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);
        scat_field_ff(i) = scat_field_ff(i) - (fields_bndry(j)*integral_g1 - fields_bndry(j+n)*integral_grad_g1_dot_n);
    end
end

RCS = 2*pi*a_ff*abs(scat_field_ff).^2; % (the magnitude of the incident field is just 1)
```

We assume that the object is heading towards the radar, hence it is oriented as in Fig. 3.

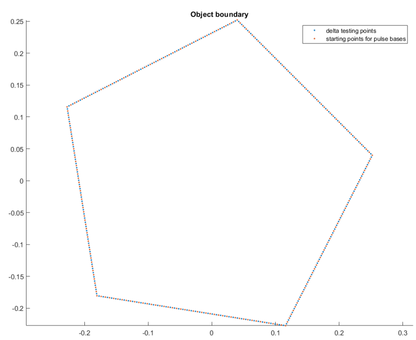

Figure 3: Object pointed towards the source

The RCS plots are as shown in Fig 4 and 5. The field is incident at an angle of 45°, i.e., it is coming from the bottom left corner. We can see that the plots are symmetric about the $y = x$

line, as was to be expected since the object is symmetric about this line and the incident field is along this line. We can see that PEC does not scatter much in the direction of the source, which was to be expected from the geometry of a pentagon. To our surprise, carbon fiber scatters a lot in the direction of the radar.

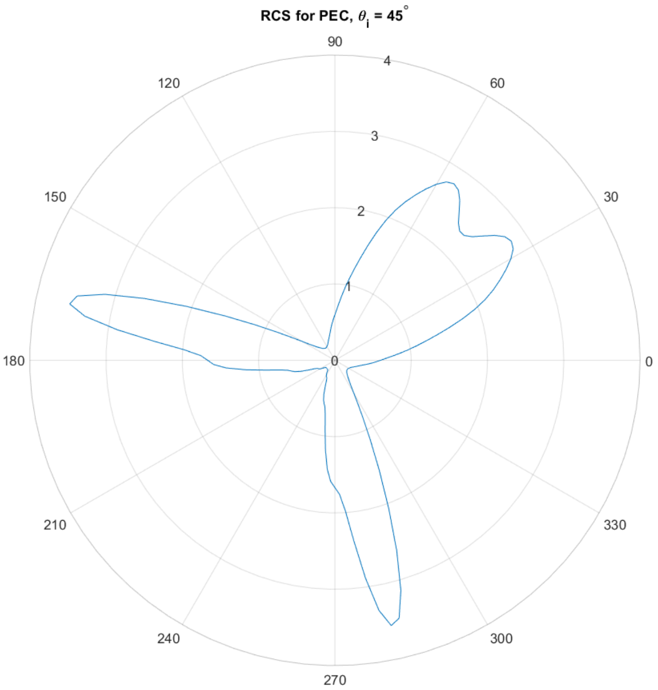

Figure 4: RCS for PEC with object oriented as in Fig. 3

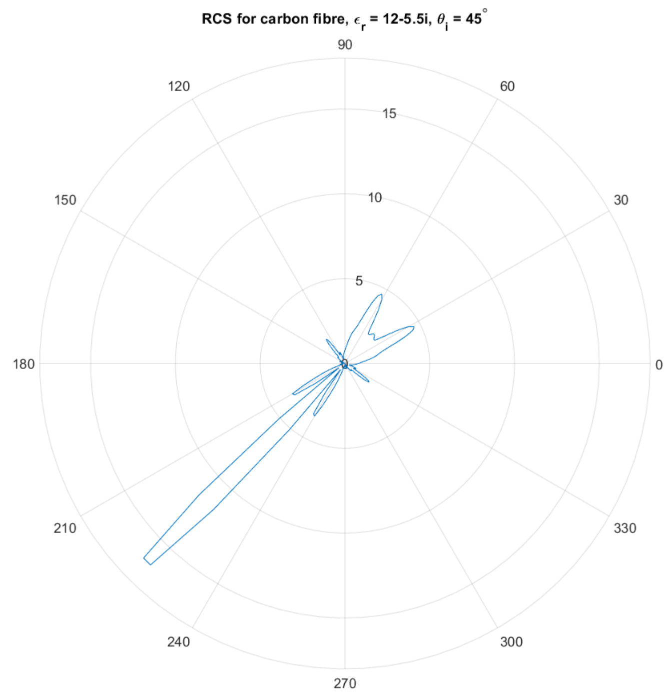

Figure 5: RCS for carbon fiber with object oriented as in Fig. 3

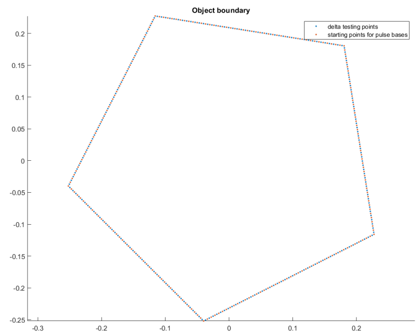

Figure 6: Object pointed away from the source

We also calculated for the case in which the object is oriented as in Fig. 6. For this orientation, the RCS plots are in Fig. 7 and 8.Here again we can see that PEC scatters strongly in the direction of the source, since the incident field encounters a normal incidence on the edge.

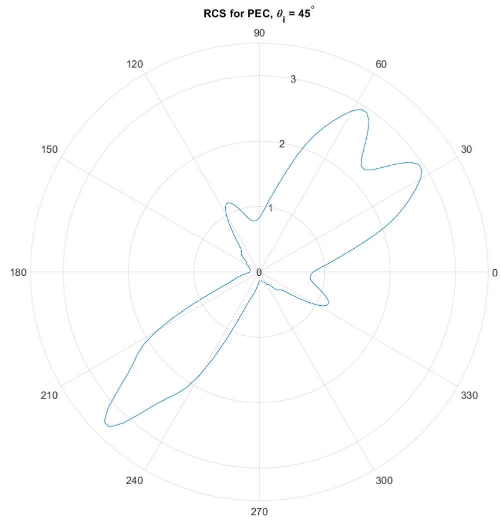

Figure 7: RCS for PEC with object oriented as in Fig. 6

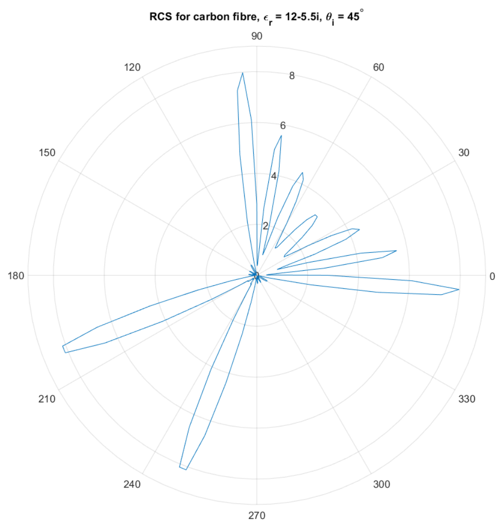

Figure 8: RCS for carbon fiber with object oriented as in Fig. 6

Appendix

5.1 HW3.m

```
tic
% 1. Defining the parameters
lambda = 0.1;
k0 = 2*pi/lambda;
a = 3*lambda;                   %edge length of pentagon
da = lambda/15;                 %discretization length, please ensure that rem(
    a,da) = 0
n_sides = 5;                    %number of sides in the object
theta_i_deg = 45;
theta_i = theta_i_deg*(pi/180); %angle of incidence
theta_obj_deg = theta_i_deg+180;%The direction the object is facing
a_ff = 4*lambda;                %far field radius
n_ff = 180;                     %discretizations for far field
tolabs = 1e-9;                  %absolute tolerance in integral
tolrel = 1e-6;                  %relative tolerance in integral
a_sc = 1e-5;                    %radius of deformation (for a segment length of
     '1')

% 2. Defining the variables
% 2.1. Related to the object
[test_pt, strt_pt] = get_shape_coords(a, da, n_sides, theta_obj_deg);

figure;
scatter(test_pt(1,:), test_pt(2,:),'.')
hold on
scatter(strt_pt(1,:), strt_pt(2,:),'.')
axis equal;
legend('delta testing points','starting points for pulse bases')
title('Object boundary')

params = [k0, da, theta_i, a_ff, n_ff, tolabs, tolrel];

% 2.2. Far field points
th_ff = 0:360/n_ff:(360-360/n_ff);
ff_pt = a_ff*[cosd(th_ff); sind(th_ff)];

% 3. Calculating RCS
% For PEC, please set eps_r as Inf
% 3.1. For PEC
eps_r_PEC = Inf;    %The functions compute for PEC when they encounter eps_r =
    Inf
fields_bndry = solve_on_boundary(eps_r_PEC, test_pt, strt_pt, params, a_sc); %
    Calculating the fields on the boundary of the object
RCS_PEC = get_RCS(eps_r_PEC, fields_bndry, ff_pt, strt_pt, params);          %
    Calculating the RCS

%3.2. For carbon fibre
eps_r_CF = 12-5.5j;
fields_bndry = solve_on_boundary(eps_r_CF, test_pt, strt_pt, params, a_sc);
RCS_CF = get_RCS(eps_r_CF, fields_bndry, ff_pt, strt_pt, params);

toc

%Plotting RCS
figure;
polarplot(th_ff([1:end 1])*pi/180, RCS_PEC([1:end 1]));
title(['RCS for PEC, \theta_i = ',num2str(theta_i_deg),'^\circ'])
figure;
polarplot(th_ff([1:end 1])*pi/180, RCS_CF([1:end 1]));
```

```
title(['RCS for carbon fibre, \epsilon_r = ',num2str(eps_r_CF),', \theta_i = ', num2str(theta_i_deg),'^\circ'])
```

5.2 get_shape_coords.m

```
function [test_pt, strt_pt] = get_shape_coords(a, da, n_sides, theta)
        %generates a regular polygon in 2D, oriented along theta, centred at origin
        ang = 360/n_sides;
        c = a/(2*sind(ang/2));      %length of line segment connecting centre to vertex
        vrtx_A = [c*cosd(theta); c*sind(theta)];    %vertex pointing along theta
        dir_AB = [-cosd(90 - (ang/2) - theta); sind(90 - (ang/2) - theta)];
        rotat_ang = [cosd(ang) -sind(ang); sind(ang) cosd(ang)];
        n_e = round(a/da);    %total no. of discretisation points

        test_pt = zeros(2, n_sides*n_e);    %points of delta testing
        strt_pt = zeros(2, n_sides*n_e);     %starting points of pulse basis
        test_pt(:,1:n_e) = vrtx_A + [da/2:da:a].*dir_AB;
        strt_pt(:,1:n_e) = vrtx_A + [0:da:a-da].*dir_AB;
        for i = 1+n_e : n_e:n_sides*n_e
            test_pt(:,i:(i+n_e-1)) = rotat_ang*test_pt(:,i-n_e:i-1);
            strt_pt(:,i:(i+n_e-1)) = rotat_ang*strt_pt(:,i-n_e:i-1);
        end

end
```

5.3 green2d.m

```
function g = green2d(k, r_p, r0, Dr, d)
        % r_p = [r_p_x r_p_y] is the primed coordinate
        % r0 = [r0_x r0_y] is the starting point
        % Dr = [Dr_x Dr_y] is the total change in r0
        % d \in [0,1) gives r = r0 + d*Dr

        % Green's function, g(r,r_p) = -j/4*H_0^(2)(k|r' - r|)

        R_x = (r0(1) + d.*Dr(1)) - r_p(1);
        R_y = (r0(2) + d.*Dr(2)) - r_p(2);
        norm_R = sqrt( R_x.^2 + R_y.^2 );
        g = -1j/4*besselh(0, 2, k.*norm_R);
end
```

5.4 gradgreen2d_dot_n.m

```
function grad_g_dot_n = gradgreen2d_dot_n(k, r_p, r0, Dr, d, n_hat)
        % n = [nx ny] is the direction of the normal
        % rest of the variables are same as in green2d

        % grad(g).n = jk/4 H_1^(2)(k|r'-r|)hat(r-r').n

        R_x = (r0(1) + d.*Dr(1)) - r_p(1);
        R_y = (r0(2) + d.*Dr(2)) - r_p(2);
        norm_R = sqrt( R_x.^2 + R_y.^2 );
        hat_R_dot_n = (n_hat(1).*R_x + n_hat(2).*R_y)./norm_R;

        grad_g_dot_n = 1j*k/4*besselh(1, 2, k.*norm_R).*hat_R_dot_n;
end
```

5.5 inc_field.m

```
function phi_i = inc_field(r_p, theta_i, k0)
```

```
        %generates a plane wave of unit amplitude travelling in theta_i direction
        %follows exp(jwt - jkx) convention
        phi_i = exp( -1j*k0*( cos(theta_i)*r_p(1) + sin(theta_i)*r_p(2) ) );
end
```

5.6 solve_on_boundary.m

```
function x = solve_on_boundary(eps_r, test_pt, strt_pt, params, a_sc)
        %finds \phi and \grad \phi. \hat{n} at the boundary using Extinction theorem
        [k0, da, theta_i, ~, ~, tolabs, tolrel] = feval (@(x) x{:} , num2cell(params));
        n = length(test_pt);
        strt_pt(:,end+1) = strt_pt(:,1); %for ease while calculating Dr for the last starting point

        %framing SI equations in Ax=b form
        if eps_r == Inf %implies PEC
            A = zeros(n, n);
            b = zeros(n,1);
            for i = 1:n
                for j = 1:n
                    Dr = strt_pt(:,j+1) - strt_pt(:,j);
                    A(i, j) = integral(@(d)green2d(k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);
                end
                b(i) = inc_field(test_pt(:,i), theta_i, k0);
            end
            A = da*A;     %scaling by the length of the segment since the integral iterated over d from 0 to 1
            x = A\b;        %solve by gaussian elimination
        else
            A = zeros(2*n, 2*n);
            b = zeros(2*n,1);
            for i = 1:n
                for j = 1:n
                    Dr = strt_pt(:,j+1) - strt_pt(:,j);
                    t_hat = Dr/norm(Dr);
                    n_hat = [-t_hat(2), t_hat(1)];
                    if i==j     %considering cases of singularity separately
                         A(i, j)  = integral(@(d)green2d(k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);           %The g1 term (integrable singularity)
                         A(i+n, j)= integral(@(d)green2d(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);       %The g2 term (integrable singularity)
                         A(i,j+n) = 0.5 -1*integral(@(d)gradgreen2d_dot_n(k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,0.5-a_sc,'AbsTol',tolabs,'RelTol',tolrel)...
                             -1*integral(@(d)gradgreen2d_dot_n(k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.5+a_sc,1,'AbsTol',tolabs,'RelTol',tolrel);        %The grad(g1).n term
                         A(i+n,j+n) = -0.5 -1*integral(@(d)gradgreen2d_dot_n(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,0.5-a_sc,'AbsTol',tolabs,'RelTol',tolrel)...
                             -1*integral(@(d)gradgreen2d_dot_n(sqrt(eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.5+a_sc,1,'AbsTol',tolabs,'RelTol',tolrel);      %The grad(g2).n term
                    else
                         A(i, j)      = integral(@(d)green2d(k0, test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);            %The g1 term
```

```
                           A(i+n, j)   = integral(@(d)green2d(sqrt(eps_r)*k0,
    test_pt(:,i), strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);
          %The g2 term
                           A(i, j+n)   = -1*integral(@(d)gradgreen2d_dot_n(k0,
    test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,1,'AbsTol',tolabs,'RelTol',
    tolrel);              %The grad(g1).n term
                           A(i+n, j+n) = -1*integral(@(d)gradgreen2d_dot_n(sqrt(
    eps_r)*k0, test_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,1,'AbsTol',tolabs,'
    RelTol',tolrel);       %The grad(g2).n term
                       end
                   end
                   b(i) = inc_field(test_pt(:,i), theta_i, k0);
               end
               A = da*A;    %scaling by the length of the segment since the
    integral iterated over d from 0 to 1
               x = A\b;       %solve by gaussian elimination
           end
end
```

5.7 get_RCS.m

```
function RCS = get_RCS(eps_r, fields_bndry, ff_pt, strt_pt, params)

        [k0, da, ~, a_ff, n_ff, tolabs, tolrel] = feval (@(x) x{:} , num2cell(
    params));
        n = length(strt_pt);
        strt_pt(:,end+1) = strt_pt(:,1); %for ease while calculating Dr for the
     last starting point

        % The field at a point 'p' is given by:
        % phi(p) = phi_inc(p) + phi_scat(p)
        % phi_scat(p) = - oint[g1(r,p) grad(phi(r)).n - grad(g1(r,p).n phi(r)]
    dr (1)
        % where the closed loop integral is over the boundary, where we know
    the fields

        % 1. Computing the scattered field at the far field points
        scat_field_ff = zeros(1,n_ff);
        if eps_r == Inf %implies PEC
            for i = 1:n_ff
                for j = 1:n
                    Dr = strt_pt(:,j+1) - strt_pt(:,j);
                    % For PEC the second term in eqn (1) is zero. First term:
                    integral_g1 = da*integral(@(d)green2d(k0, ff_pt(:,i),
    strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);
                    scat_field_ff(i) = scat_field_ff(i) - fields_bndry(j)*
    integral_g1;
                end
            end
        else
            for i = 1:n_ff
                for j = 1:n
                    Dr = strt_pt(:,j+1) - strt_pt(:,j);
                    t_hat = Dr/norm(Dr);
                    n_hat = [-t_hat(2), t_hat(1)];
                    integral_g1 = da*integral(@(d)green2d(k0, ff_pt(:,i),
    strt_pt(:,j), Dr, d),0.0,1,'AbsTol',tolabs,'RelTol',tolrel);
                    integral_grad_g1_dot_n = da*integral(@(d)gradgreen2d_dot_n(
    k0, ff_pt(:,i), strt_pt(:,j), Dr, d, n_hat),0.0,1,'AbsTol',tolabs,'RelTol',
    tolrel);
                    scat_field_ff(i) = scat_field_ff(i) - (fields_bndry(j)*
    integral_g1 - fields_bndry(j+n)*integral_grad_g1_dot_n);
                end
            end
```

```
        end

        RCS = 2*pi*a_ff*abs(scat_field_ff).^2; % (the magnitude of the incident
    field is just 1)
end
```

References

[1] Liu, Lei & He, Pingge & Zhou, Kechao & Chen, Tengfei. (2014). *Microwave absorption properties of carbon fibers with carbon coils of different morphologies (double microcoils and single nanocoils) grown on them.*, Journal of Materials Science. 49. 10.1007/s10853-014-8137-z.