RCS CALCUATIONS USING THE PHYSICAL OPTICS CODES

Introduction

Physical optics based RCS calculations are performed by the two Matlab computer codes:

1. *pomonostatic.m*
2. *pobistatic.m*

The first code (*pomonostatic.m*) is used for monostatic RCS calculation and the second (*pobistatic.m*) for bistatic RCS. (In the GUI version both are run from *pofacets.m*.) Both use the same input file format, although the interactive input portion of the codes are slightly different. In the version without the GUI, there are a few calculation parameters that are "hard-wired" into the code using assignment statements. The "hard-wired" parameters can be edited to suit the user's needs.

Monostatic RCS: *pomonostatic.m*

pomonostatic.m is a simple RCS prediction code based on the physical optics approximation. Scattering objects are approximated by arrays of triangles (facets) and superposition is used to compute the total RCS of the object.

- the scattered field of each triangle is computed as if it were isolated and other triangles were not present
- multiple reflections are not included
- edge diffraction and surface waves are not included
- shadowing is only approximately included by considering a facet to be completely illuminated or completely shadowed by the incident wave (shadows on one part of the object cast by other parts of the object are not included)

The user must define the geometry in a specified format in the files *coordinates.m* and *facets.m* which are located in a separate subdirectory in the *po* directory. The input section of the code is shown below:

```
% Load node coordinate data
  name=input('Enter directory name for data: ','s');
  fname=[name,'/coordinates.m'];
  feval('load',fname);
```

The nodes of all triangles and their locations in a global cartesian coordinate system are specified in *coordinates.m*. The unit is meters and the order of the nodes is not important. However, the first entry (row) is considered as "node 1," the second row as "node 2," … etc. An example is shown below for the file *vtail* geometry:

x	*y*	*z*
1.0000000e+02	0.0000000e+00	0.0000000e+00
7.8006900e+01	0.0000000e+00	1.0538400e+01
6.7926700e+01	-1.5349400e+01	0.0000000e+00
6.7926700e+01	1.5349400e+01	0.0000000e+00
-5.2119100e+01	0.0000000e+00	2.9095100e+01
-7.2737700e+01	2.2451300e+01	0.0000000e+00
-7.2737700e+01	-2.2451300e+01	0.0000000e+00
-1.0000000e+02	0.0000000e+00	0.0000000e+00
-2.4055000e+00	1.8900300e+01	0.0000000e+00
2.0791500e+01	9.0066700e+01	1.2738800e+01
-1.4031400e+01	9.2349600e+01	1.2831300e+01
1.0590250e+02	2.6896700e+01	2.6517200e+01
9.4551900e+01	3.1837100e+01	2.7211300e+01
-2.4055000e+00	-1.8900300e+01	0.0000000e+00
2.0791500e+01	-9.0066700e+01	1.2738800e+01
-1.4031400e+01	-9.2349600e+01	1.2831300e+01
1.0590250e+02	-2.6896700e+01	2.6517200e+01
9.4551900e+01	-3.1837100e+01	2.7211300e+01

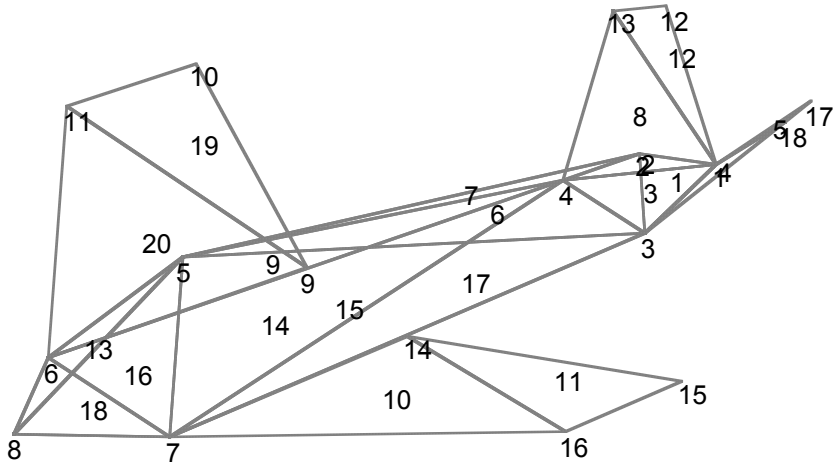
The node connection list is specified in the file *facets.m*. Three nodes comprise a triangle (facet). An example is shown below for the *vtail* geometry:

facet #	node #	node #	node #	illum flag	Rs
1	2	3	1	1	0
2	4	2	1	1	0
3	3	4	1	1	0
4	18	1	3	0	0
5	18	17	1	0	0
6	5	3	2	1	0
7	5	2	4	1	0
8	13	4	1	0	0
9	6	5	4	1	0
13	8	5	6	1	0
14	7	6	4	1	0
15	5	7	3	1	0
16	8	7	5	1	0
17	7	4	3	1	0
18	6	7	8	1	0
19	9	10	11	0	0
20	9	11	6	0	0
12	13	1	12	0	0
11	14	16	15	0	0
10	7	16	14	0	0

Column 1 is the facet number (which does not necessarily need to be sequential). Columns 2-4 are the nodes that comprise the facet. Column 5 is an illumination flag (variable *ilum* in the code). If *ilum* is 0 then the facet is allowed to be illuminated from both sides. If *ilum* is 1 then the facet will only be illuminated from one side, which is determined from the order of the nodes (in a right-hand, or counter clockwise, sense)

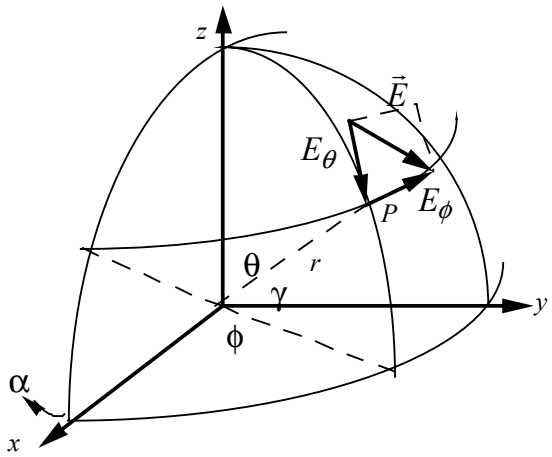
- if both sides of the triangle are exposed to the incident wave then set *ilum* = 0
- if the triangle is part of a closed structure, then set *ilum* = 1, and order the vertices in a right-handed sense so that the normal points outward from the closed structure

The file for *vtail* illustrates the concept. Triangle 10 represents a wing and both sides are exposed to the incident wave. Therefore *ilum* is 0 and the order of the nodes does not matter. On the other hand, facet 15 is part of the closed fuselage and is only illuminated from the outside, not from behind. Thus *ilum* is 1 and the order of the nodes (5,7,3) gives a normal that points outward according to the right hand rule. The order (7,3,5) or (3,5,7) could have been used. (Note that all illumination flags can be turned off in the program by setting iflag=1 in the code. This may be desired for testing purposes.)



The last column in *facets.m* is the resistivity of the facet. The values should be normalized to free space (377 Ohms). If $R_s = 0$ then the facet is a perfect electric conductor; when $R_s = \infty$ (in practice, several thousand) the facet is transparent.

A standard spherical coordinate system is used to specify incident and observation (scattering) directions.



The RCS is calculated at angles (θ, ϕ) specified by the user as follows:

```
% Pattern loop
  disp('enter phi pattern values')
  disp('same value for stop and start results in a phi cut');
  pstart=input('enter phi start in degrees: ');
  pstop=input('enter phi stop in degrees: ');
  delp=input('enter phi increment in degrees: ');
  if delp==0, delp=1; end
  if pstart==pstop, phr0=pstart*rad; end
  disp('enter theta pattern values')
  disp('same value for stop and start results in a theta cut');
   tstart=input('enter theta start in degrees: ');
   tstop=input('enter theta stop in degrees: ');
   delt=input('enter theta increment in degrees: ');
   if delt==0, delt=1; end
   if tstart==tstop, thr0=tstart*rad; end
```

- a single value of θ can be specified along with a range of ϕ (a θ cut is plotted)
- a single value of ϕ can be specified along with a range of θ (a ϕ cut is plotted)
- ranges of θ and ϕ can be specified (a two-dimensional contour is plotted in direction cosine space: $u = \sin\theta\cos\phi, v = \sin\theta\sin\phi$).

The Matlab *contour* function can be replaced by the *mesh* function if desired.

The polarization of the incident wave is assumed to be of the form $E_{0\theta}\hat{\theta} + E_{0\phi}\hat{\phi}$. The components are specified in the Matlab code as illustrated below for a pure TM polarized wave:

```
% Incident wave polarization
 Et=1+j*0;  %TM-z
 Ep=0+j*0;  %TE-z
```

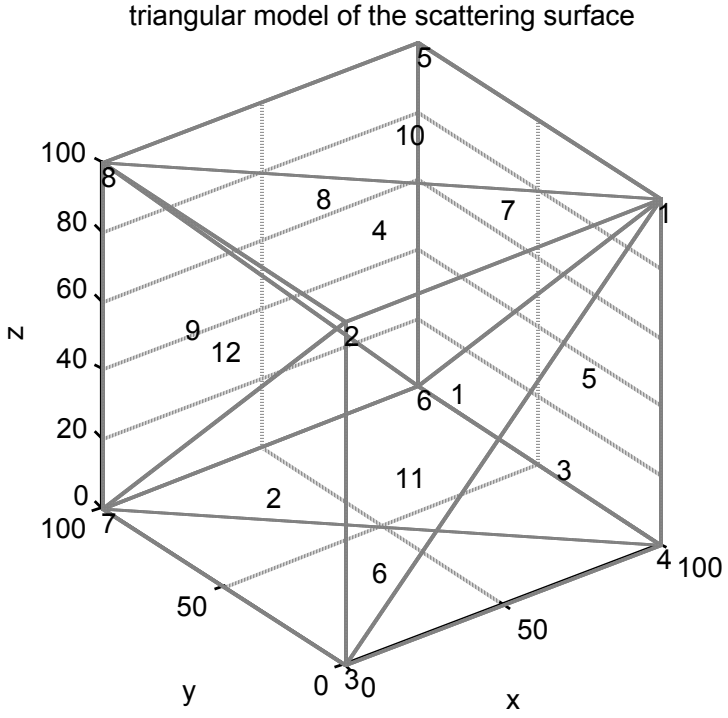
The frequency (wavelength in meters) is also defined in the code by the statement `wave=20.`
It can be changed to any convenient value. If it is set to 1 then when the geometry data entered in meters, it will also be in terms of wavelength.

Two computational parameters that may be varied are

```
 Lt=.05; % Taylor series region
 Nt=5;   % Number of terms in Taylor series
```

See the reference for details on how this parameters affects the accuracy and computation time. In general these values give good results with relatively fast run times.

Example: box

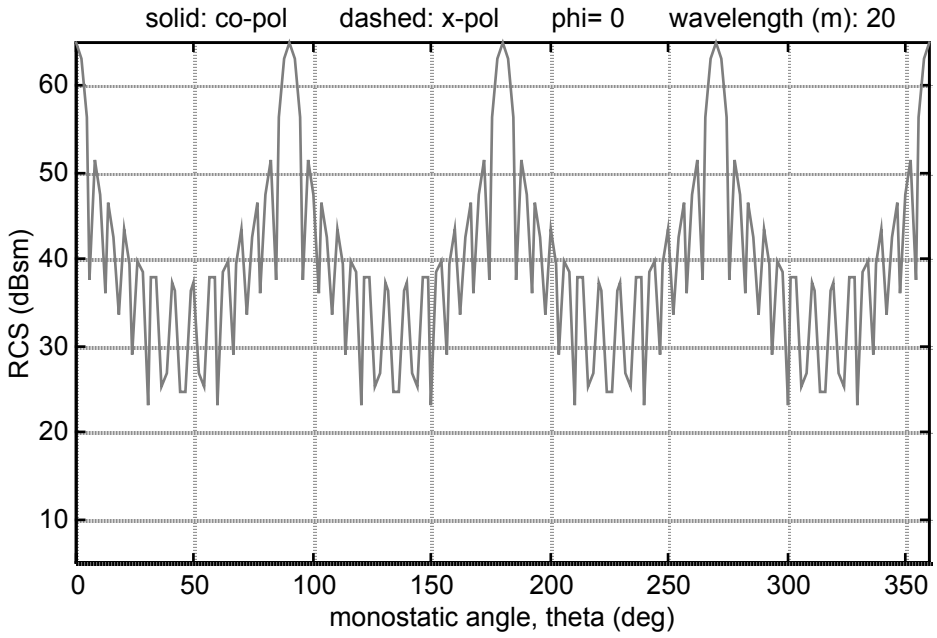


Input sequence: (for this calculation the wavelength is 20 m and Et=1, Ep=0)

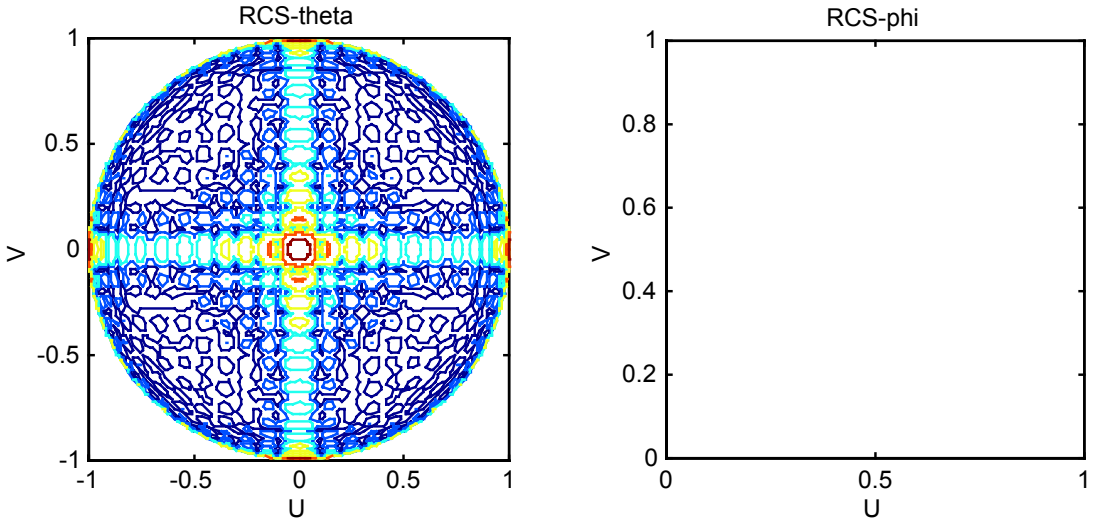
```
Enter directory name for data: box
label vertices? (y or n)? y
label faces? (y or n)? y
enter phi pattern values
same value for stop and start results in a phi cut
enter phi start in degrees: 0
enter phi stop in degrees: 0
enter phi increment in degrees: 0
enter theta pattern values
same value for stop and start results in a theta cut
enter theta start in degrees: 0
enter theta stop in degrees: 360
enter theta increment in degrees: 2
```

A two-dimensional contour is generated with the following input:

```
   …
same value for stop and start results in a phi cut
enter phi start in degrees: 0
enter phi stop in degrees: 360
enter phi increment in degrees: 2
enter theta pattern values
same value for stop and start results in a theta cut
enter theta start in degrees: 0
enter theta stop in degrees: 90
enter theta increment in degrees: 2
```



RCS-theta is $\sigma_{\theta\theta}$ (θ polarized incident wave; θ polarized receiver). This is the co-polarized component. RCS-phi is $\sigma_{\phi\theta}$ (θ polarized incident wave; ϕ polarized receiver). The latter is the cross-polarized component. In this particular example it happens to be at a level below the specified threshold in the code (which is the maximum value of RCS - 60 dB, but can be changed):



```
% set plot range
   Smax=max([max(max(Sth)),max(max(Sph))]);
   Lmax=(floor(Smax/5)+1)*5; Lmin=Lmax-60;
   Sth(:,:)=max(Sth(:,:),Lmin);
   Sph(:,:)=max(Sph(:,:),Lmin);
```

The direction cosine plot is a projection of the RCS on the surface of the unit sphere onto the *xy* plane. The contours plotted below correspond to the RCS on the upper hemisphere. At (0,0) the top of the box is viewed directly resulting in a large RCS. The sides are viewed at the direction cosine pairs (-1,0), (1,0), (0,-1) and (0,1).

Bistatic RCS: *pobistatic.m*

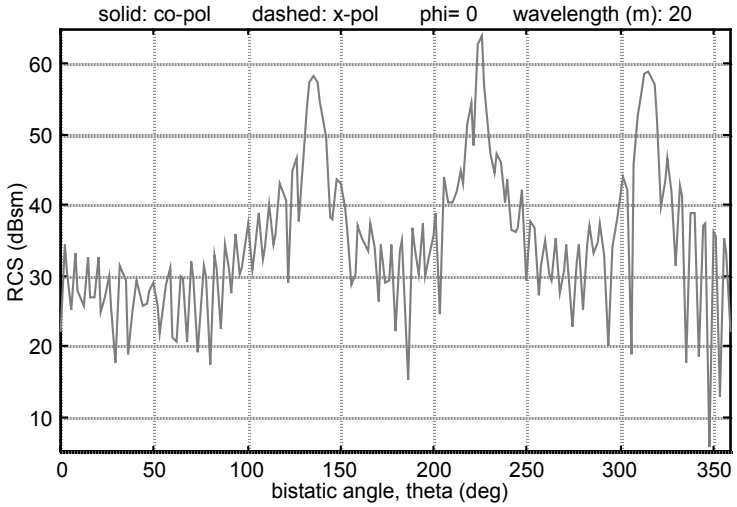
For the bistatic calculation, the incidence direction is fixed and the observation direction changes. The user inputs the data shown below:

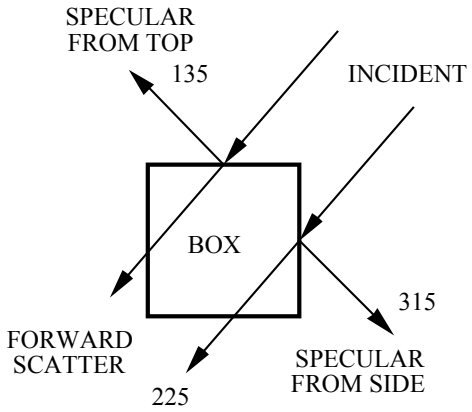
```
% Pattern loop
   thetai=input('enter theta incidence angle in degrees: ');
   phii=input('enter phi incidence in degrees: ');
   disp('enter phi observation values')
   disp('same value for stop and start results in a phi cut');
   pstart=input('enter phi start in degrees: ');
   pstop=input('enter phi stop in degrees: ');
   delp=input('enter phi increment in degrees: ');
  if delp==0, delp=1; end

  if pstart==pstop, phr0=pstart*rad; end
   disp('enter theta observation values')
   disp('same value for stop and start results in a theta cut');
    tstart=input('enter theta start in degrees: ');
    tstop=input('enter theta stop in degrees: ');
    delt=input('enter theta increment in degrees: ');
   if delt==0, delt=1; end
   if tstart==tstop, thr0=tstart*rad; end
```

Example: box

The bistatic RCS of the box is shown below. The incidence angle is 45 degrees in the $\phi = 0^\circ$ plane and the observation direction is varied from 0 to 360 degrees in the $\phi = 0^\circ$ plane. The lobe at 135 degrees is the specular reflection from the top. The forward scatter is also evident. Note that the forward scatter is higher than the highest monostatic backscatter lobe. The accompanying ray diagram illustrates the directions of the expected large RCS lobes.





Diffuse RCS

One measure of the roughness of a surface is the standard deviation δ of its variation about the mean value. In addition, the correlation interval c indicates how rapidly the surface changes as one moves along the surface. The program *podiffuse.m* computes the approximate diffuse scattering component of the RCS and adds it to the coherent component. All triangles are assumed to have the same correlation interval and standard deviation. The roughness should be small in terms of wavelength ($\delta << \lambda$). The random deviation of the surface results in diffuse scattering which yields a relatively angle independent term to the RCS (actually, it varies as $\cos^2\theta$).

The geometry and pattern input sections are the same as those for *pomonostatic.m*. The correlation interval and standard deviation are assigned in the Matlab code:

```
% correlation distance in meters
% all triangles have the same roughness and correlation interval
  corel=5;
% standard deviation of surface roughness in meters
 delstd=.1*wave;
 delsq=delstd^2;
 bk=2*pi/wave;
 cfac1=exp(-4*bk^2*delsq);
 cfac2=4*pi*(bk*corel)^2*delsq;
```

The diffuse component for triangle *m* is approximated by

```
Edif=cfac2*Area(m)*ct2^2*exp(-(corel*pi*st2/wave)^2);
```

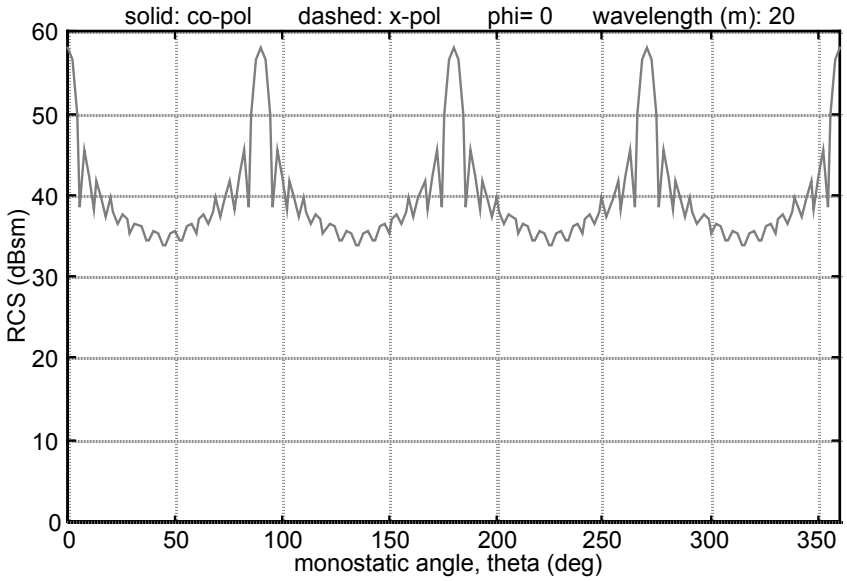
This term is multiplied by the polarization factors and then added to the specular component as follows:

sum of specular terms | sum of diffuse terms

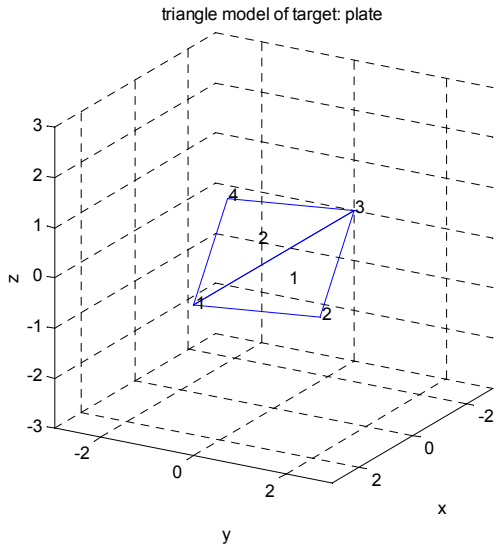
```
Sth(i1,i2)=10*log10(4*pi*cfac1*(abs(sumt)^2+sqrt(1-cfac1^2)*sumdt)/wave^2+
                                                                  1e-10);
Sph(i1,i2)=10*log10(4*pi*cfac1*(abs(sump)^2+ sqrt(1-cfac1^2)*sumdp)/wave^2+
                                                                  1e-10);
```

Example: box

Data for the box is shown below for a correlation interval of 5 m and a 2 m standard deviation of roughness.



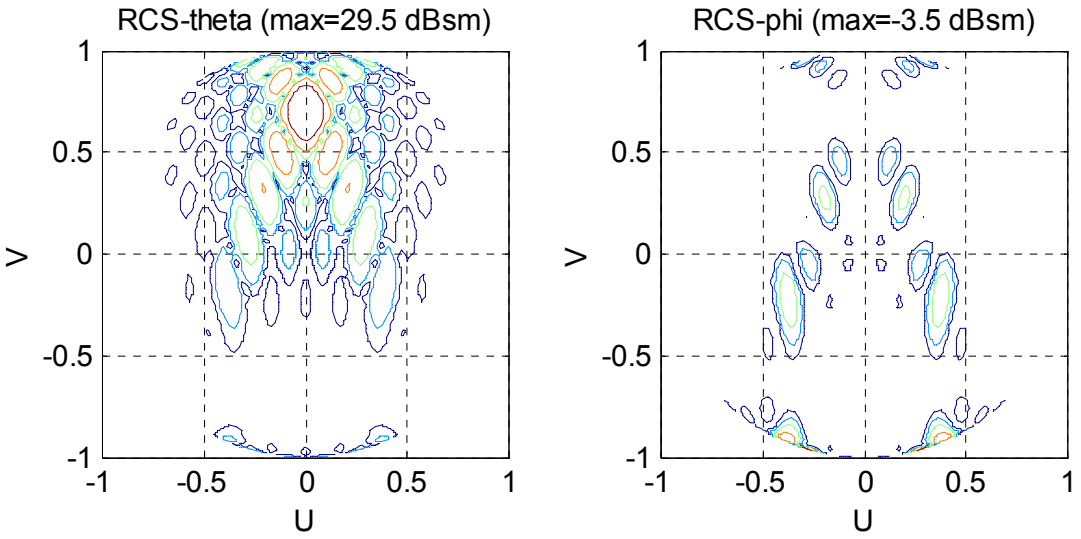
Example: tilted plate (rotated 45 degrees around the *x* axis with two vertices on the *x* axis and two vertices in the *yz* plane)



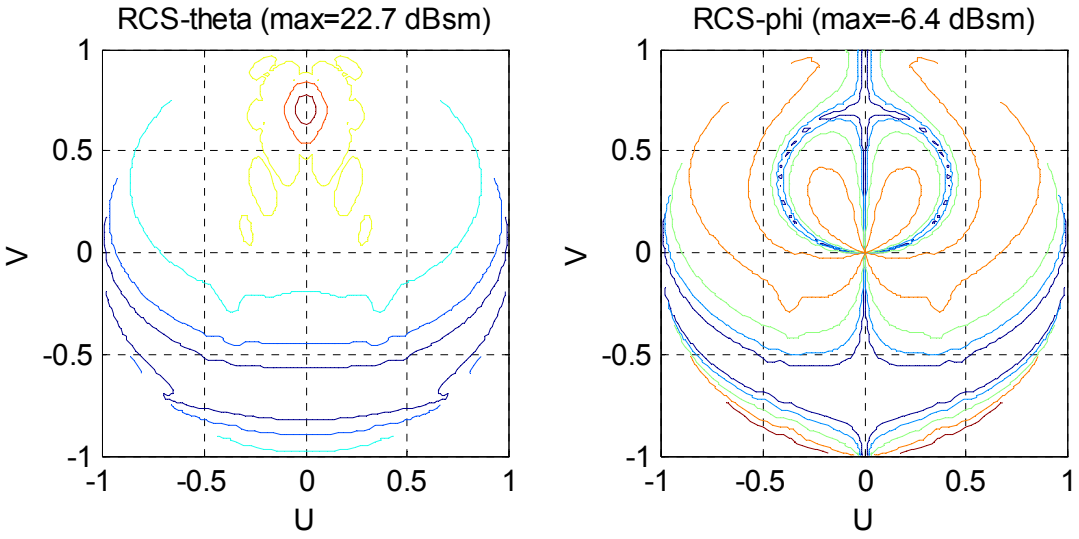
Computational parameters:

```
wave=1;
corel=0.1;
Et=1+j*0;  %TM-z
```

RCS contours without surface roughness



RCS contours with surface roughness



References:

1. Moreira and Prata, "A Self-Checking Predictor-Corrector Algorithm for Efficient Evaluation of Reflector Antenna Radiation Integrals," *IEEE Trans. on Antennas & Prop.*, Vol. 42, No. 2, Feb. 1994, pp.246-254.

2. Jenn, *Radar and Laser Cross Section Engineering*, AIAA Education Series, June 1995.

(latest revision: 12/00)