

Model Diffraction Through a Grating

@ 10GHz

Reflectance= 18.8%

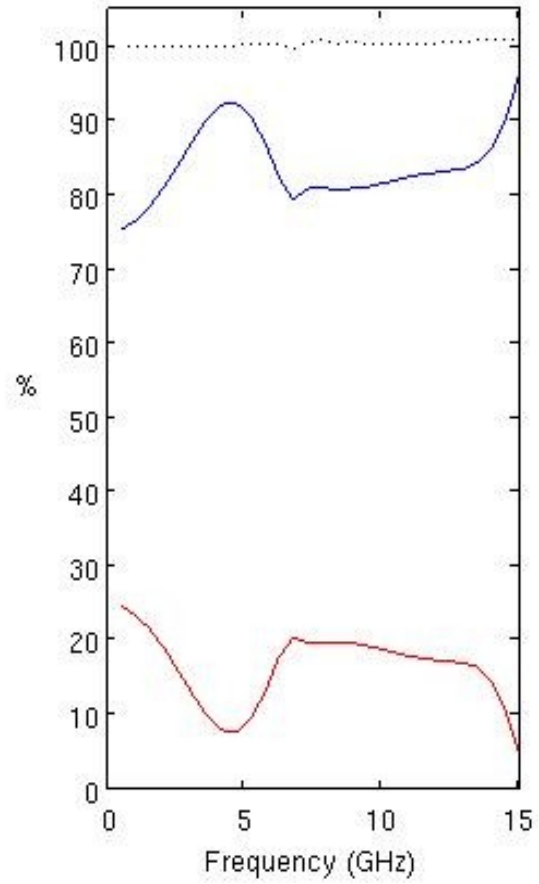
Transmittance = 81.8%

Conservation = 100.6%

STEP13700 of ~13712



REFLECTANCE AND TRANSMITTANCE



BinaryGrating.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Diffraction Through a Grating
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Initialize MATLAB
close all; clc;
clear all;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Problem
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% UNITS
```

```

meters = 1;
decimeters = 1e-1 * meters;
centimeters = 1e-2 * meters;
millimeters = 1e-3 * meters;
inches = 2.54 * centimeters;
feet = 12 * inches;
seconds = 1;
hertz = 1/seconds;
kilohertz = 1e3 * hertz;
megahertz = 1e6 * hertz;
gigahertz = 1e9 * hertz;

% Constants
c0 = 299792458;

% Frequency we want to transmit
f0 = 10 * gigahertz;

NPML = [0 0 20 20];

% Bragg Grating Materials
erl = 9;

d = 0.75 * centimeters; % the Binary stand height
PeriodWidth = 1.5 * centimeters; %Width between to Binary stands
dwidth = .5 * PeriodWidth;

%Calculate the Length of our layers.

dc.x = PeriodWidth;
dc.y = d;

Size.x = PeriodWidth;
Size.y = d * 3;

rNx = ceil(Size.x/(centimeters*decimeters)); %This Nz represents real world size
rNy = ceil(Size.y/(centimeters*decimeters));
disp(rNx);
disp(rNy);

% Material Vectors Initialized at Air
rER = ones(rNx,rNy);
rUR = ones(rNx,rNy);

dx = Size.x/rNx;
dy = Size.y/rNy;

% Fill our Stand
for nx=1:floor(dwidth/dx)
    for ny=1:floor(d/dy)
        rER(nx,ny)=erl;
    end
end

% Fill our body

```

```

for nx=1:rNx
    for ny = floor(d/dy)+1:rNy
        rER(nx,ny) = erl;
    end
end

% Add our Materials to the model

% Frequency

freq_start = 0;
freq_end = 15 * gigahertz;

NFREQ = freq_end / (0.5*gigahertz); %Frequencies every 100nm
FREQ = linspace(freq_start, freq_end, NFREQ); %FREQ List

Buffer.x.value = 0;
Buffer.x.e = [-1 -1];
Buffer.x.u = [1 1];

Buffer.y.value = -1;
Buffer.y.e = [1 9];
Buffer.y.u = [1 1];

subplot(121);
imagesc(rER);
% global ERzz;
FDTD2D( dc, Size, rER, rUR, -1, 5e-4, Buffer, NPML, FREQ, NFREQ, 100,
10*gigahertz, 'Binary Grating');

FDTD2D.m

function FDTD2D( dc, Size, rER, rUR, Steps, EMAX, Buffer, NPML, FREQ, NFREQ,
Update, SSFREQ, Title )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Pre-Program Work
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% UNITS
meters = 1;
decimeters = 1e-1 * meters;
centimeters = 1e-2 * meters;
millimeters = 1e-3 * meters;
inches = 2.54 * centimeters;
feet = 12 * inches;
seconds = 1;
hertz = 1/seconds;
kilohertz = 1e3 * hertz;
megahertz = 1e6 * hertz;
gigahertz = 1e9 * hertz;

%Constants
c0 = 299792458; %m/s
e0 = 8.854187817*10^-12; %F/m
u0 = 1.256637061*10^-6; %H/m

```

```

N0 = sqrt(u0/e0);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initialization of Parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fmax = FREQ(length(FREQ));
fmin = FREQ(1);

f0 = SSFREQ;
lam0 = c0/f0;

nmax = Getnmax2D(rER, rUR);

%Compute Grid Resolution
% Wave Length Resolution
N_lambda = GetNlambda(rER, rUR);
lambda_min = c0 / (fmax);
d_lambda = lambda_min/N_lambda/nmax;

% Structure Resolution
N_d = 10;
ddx = dc.x/N_d;
ddy = dc.y/N_d;

% Calculate grid resolution dx
dx = min(d_lambda, ddx);
N_prime = 2*ceil(dc.x/dx/2)+1;
dx = dc.x/N_prime;
Nx = ceil(Size.x/dx);

% Calculate grid resolution dy
dy = min(d_lambda, ddy);
N_prime = ceil(dc.y/dy);
dy = dc.y/N_prime;
Ny = ceil(Size.y/dy);

% Add free space buffer and TF/SF
if(Buffer.x.value == -1)
    buffer.x = ceil(0.5*lam0/dx);
    buffer.xt = buffer.x*2;
else
    buffer.x = Buffer.x.value;
    buffer.xt = Buffer.x.value*2;
end

if(Buffer.y.value == -1)
    buffer.y = ceil(0.5*lam0/dy);
    buffer.yt = buffer.y*2+3;
else
    buffer.y = Buffer.y.value;
    buffer.yt = Buffer.y.value*2+3;
end

Nx = Nx + buffer.xt + NPML(1) + NPML(2);

```

```
Ny = Ny + buffer.yt + NPML(3) + NPML(4);
```

```
%Compute Time Steps
```

```
nsrc = 1; %Source is injected in air
```

```
dt = nsrc * dy/(2*c0); %secs
```

```
tau = 0.5/fmax; % tau parameter
```

```
t0 = 6*tau; % Delay/Pulse Position
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Model
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
[m n]=size(rER);
```

```
cf.x = floor((Nx - buffer.x*2-NPML(1)-NPML(2))/m); % Conversion factor to convert  
our real grid to our numerical grid
```

```
cf.y = ceil((Ny - buffer.y*2-3-NPML(3)-NPML(4))/n);
```

```
% Material Properties
```

```
% global ERzz;
```

```
URxx = zeros(Nx,Ny);
```

```
URyy = zeros(Nx,Ny);
```

```
ERzz = zeros(Nx,Ny);
```

```
% disp(size(URxx));
```

```
% disp(['buffer x: ' num2str(buffer.x)]);
```

```
% disp(['buffer y: ' num2str(buffer.y)]);
```

```
% disp(['cf x: ' num2str(cf.x)]);
```

```
% disp(['cf y: ' num2str(cf.y)]);
```

```
% disp(['dx: ' num2str(dx)]);
```

```
% disp(['dy: ' num2str(dy)]);
```

```
% disp(['Size x: ' num2str(Nx*dx*centimeters)]);
```

```
% disp(['Size y: ' num2str(Ny*dy*centimeters)]);
```

```
% We Need to lay our real materials vectors over our numerical material  
% grid
```

```
% Lets place our real grid in proper location on numerical grid
```

```
for x=0:m-1
```

```
    for y=0:n-1
```

```
        index.x = buffer.x+NPML(1) + x*cf.x+1;
```

```
        index.y = buffer.y+2+NPML(3) + y*cf.y+1;
```

```
        ERzz(index.x, index.y) = rER(x+1,y+1);
```

```
        URxx(index.x, index.y) = rUR(x+1,y+1);
```

```
        URyy(index.x, index.y) = rUR(x+1,y+1);
```

```
    end
```

```
end
```

```
%Fill our buffer regions
```

```
ERzz(1:buffer.x+NPML(1),:)=Buffer.x.e(1);
```

```
ERzz(:,1:buffer.y+NPML(3)+2) = Buffer.y.e(1);
```

```
ERzz(Nx-buffer.x-NPML(2)+1:Nx,:) = Buffer.x.e(2);
```

```
ERzz(:,Ny-buffer.y-NPML(4)-1+1:Ny) = Buffer.y.e(2);
```

```
URxx(1:buffer.x+NPML(1),:)=Buffer.x.u(1);
```

```
URxx(:,1:buffer.y+NPML(3)+2) = Buffer.y.u(1);
```

```
URxx(Nx-buffer.x-NPML(2)+1:Nx,:) = Buffer.x.u(2);
```

```
URyy(1:buffer.x+NPML(1),:)=Buffer.x.u(1);
URyy(:,1:buffer.y+NPML(3)+2) = Buffer.y.u(1);
URyy(Nx-buffer.x-NPML(2)+1:Nx,:) = Buffer.x.u(2);
URyy(:,Ny-buffer.y-NPML(4)-1+1:Ny) = Buffer.y.u(2);
```

```

for y=1 : Ny
  for x=2 : Nx
    if(ERzz(x,y) == 0)
      ERzz(x,y) = ERzz(x-1,y);
    end
    if(URxx(x,y) == 0)
      URxx(x,y) = URxx(x-1,y);
    end
    if(URyy(x,y) == 0)
      URyy(x,y) = URyy(x-1,y);
    end
  end
end
end

```

```

for x=1 : Nx
  for y=1 : Ny
    if (ERzz(x,y) == 0)
      ERzz(x,y) = ERzz(x,y-1);
    end
    if (URxx(x,y) == 0)
      URxx(x,y) = URxx(x,y-1);
    end
    if (URyy(x,y) == 0)
      URyy(x,y) = URyy(x,y-1);
    end
  end
end
end

```

 % Calculate STEPS 

```

STEPS = Steps;
if(STEPS == -1)
    tprop = (nmax*Ny*dy)/c0; % Wave Propagation time;
    T = 12*tau + 5*tprop;
    STEPS = ceil(T/dt)*2;
end

```

```
ta = [0:STEPS-1]*dt;      % Time Axis;
```

```

%  Grid Parameters

```

```

% Compute Grid Axis
xa = [0:Nx-1]*dx;
ya = [0:Ny-1]*dy;

% Compute 2x Grid
Nx2 = 2*Nx;
Ny2 = 2*Ny;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate PML Parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Compute sigx
sigx = zeros(Nx2, Ny2);
for nx=1:2*NPML(1)
    i = 2*NPML(1) - nx + 1;
    sigx(i, :) = (0.5*e0/dt)*(nx/2/NPML(1))^3;
end
for nx=1:2*NPML(2)
    i = Nx2 - 2*NPML(2) + nx;
    sigx(i, :) = (0.5*e0/dt)*(nx/2/NPML(2))^3;
end

% Compute sigy
sigy = zeros(Nx2, Ny2);
for ny=1:2*NPML(3)
    j = 2*NPML(3) - ny + 1;
    sigy(:,j) = (0.5*e0/dt)*(ny/2/NPML(3))^3;
end
for ny=1:2*NPML(4)
    j = Ny2 - 2*NPML(4) + ny;
    sigy(:,j) = (0.5*e0/dt)*(ny/2/NPML(4))^3;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%FDTD Initialization
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Update Coefficients
sigHx = sigx(1:2:Nx2, 2:2:Ny2);
sigHy = sigy(1:2:Nx2, 2:2:Ny2);

mHx0 = (1/dt) + (sigHy/(2*e0));

disp(size(URxx));
disp(size(mHx0));

mHx1 = ((1/dt) - (sigHy/(2*e0)))./mHx0;
mHx2 = -(c0./URxx)./mHx0;
mHx3 = -((c0*dt/e0)*(sigHx./URxx))./mHx0;

sigHx = sigx(2:2:Nx2, 1:2:Ny2);

```

```

sigHy = sigy(2:2:Nx2, 1:2:Ny2);
mHy0 = (1/dt)+(sigHx/(2*e0));
mHy1 = ((1/dt) - (sigHx/(2*e0)))./mHy0;
mHy2 = -(c0./URyy)./mHy0;
mHy3 = -((c0*dt/e0)*sigHy./URyy)./mHy0;

sigDx = sigx(1:2:Nx2, 1:2:Ny2);
sigDy = sigy(1:2:Nx2, 1:2:Ny2);
mDz0 = (1/dt) + ((sigDx + sigDy)/(2*e0)) + (sigDx.*sigDy)*dt/(4*e0^2);
mDz1 = ((1/dt) - ((sigDx + sigDy)/(2*e0)) - (sigDx.*sigDy)*dt/(4*e0^2)) ./mDz0;
mDz2 = c0./mDz0;
mDz4 = - (dt/e0^2)*sigDx.*sigDy./mDz0;

mEz1 = 1./ERzz;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Source
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ny_src = NPML(3)+2;
A = -sqrt(ERzz(1,ny_src)/URyy(1,ny_src)); % H Amplitude
deltsrc = 0.5*dy/c0 + dt/2; % Delay between E and H

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REF and TRN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

K = exp(-1i*2*pi*dt*FREQ); %Kernels for sweep across grid
K0 = exp(-1i*2*pi*dt*f0); %Kernel for our design freq

EREF = zeros(Nx, NFREQ); % Steady-State Reflected
ETRN = zeros(Nx, NFREQ); % Steady-State Transmitted
SRC = zeros(1, NFREQ); % Source transform

if(f0 ~= -1)
    Eref0 = zeros(Nx,1);
    Etrn0 = zeros(Nx,1);
    ssSRC = 0;
end

% Position of Recording planes
ny_ref = NPML(3) + 1;
ny_trn = Ny - NPML(4);

% Refractive indices in Recodring planes
nref = sqrt(ERzz(1,ny_ref)*URxx(1,ny_ref));
ntrn = sqrt(ERzz(1,ny_trn)*URxx(1,ny_trn));

disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');
disp('% Parameters');
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');

disp(['fmax' num2str(fmax)]);
disp(['lamda_min: ' num2str(lambda_min)]);
disp(['d_lambda: ' num2str(d_lambda)]);
disp(['nmax: ' num2str(nmax)]);
disp(['N_lambda' num2str(N_lambda)]);

```



```

if(EMAX == -1)
    EMAX = 1e-5;
end

while ((emax > EMAX) || (T < STEPS))
    T = T + 1;

    % Compute Curl of E

    %%% CEx
    for ny=1:Ny-1
        for nx=1:Nx
            CEx(nx,ny) = (Ez(nx,ny+1) - Ez(nx,ny))/dy;
        end
    end

    for nx=1:Nx
        CEx(nx,Ny) = (Ez(nx,1) - Ez(nx,Ny))/dy;
    end

    %%% CEy
    for nx=1:Nx-1
        for ny=1:Ny
            CEy(nx,ny) = - (Ez(nx+1,ny) - Ez(nx,ny))/dx;
        end
    end

    for ny=1:Ny
        CEy(Nx,ny) = - (Ez(1,ny) - Ez(Nx,ny))/dx;
    end

    % TF/SF Source
    Ezsrc = exp(-((T*dt-t0)/tau).^2);
    CEx(:,ny_src-1) = CEx(:,ny_src-1) - Ezsrc/dy;

    % Update H Integrations
    ICEx = ICEx + CEx;
    ICEy = ICEy + CEy;

    % Update H Field
    Hx = mHx1.*Hx + mHx2.*CEx + mHx3.*ICEx;
    Hy = mHy1.*Hy + mHy2.*CEy + mHy3.*ICEy;

    %Update Curl of H
    CHz(1,1) = (Hy(1,1) - Hy(Nx,1))/dx - (Hx(1,1) - Hx(1,Ny))/dy;

    for nx=2:Nx
        CHz(nx,1) = (Hy(nx,1)-Hy(nx-1,1))/dx - (Hx(nx,1)-Hx(nx,Ny))/dy;
    end

    for ny=2:Ny
        CHz(1,ny) = (Hy(1,ny)-Hy(Nx,ny))/dx - (Hx(1,ny)-Hx(1,ny-1))/dy;
        for nx=2:Nx

```

```

        CHz(nx,ny) = (Hy(nx,ny)-Hy(nx-1,ny))/dx - (Hx(nx,ny)-Hx(nx,ny-1))/dy;
    end
end

% TF/SF Source
Hx_src = A*exp(-((T*dt-t0+deltsrc)/tau).^2);
CHz(:,ny_src) = CHz(:,ny_src) - Hx_src/dy;

%Update D Integrations
IDz = IDz + Dz;

% Update Dz
Dz = mDz1.*Dz + mDz2.*CHz + mDz4.*IDz;

% Update Ez
Ez = mEz1.*Dz;
emax = max(max(Ez));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Compute Power %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for f = 1:NFREQ
    EREF(:,f) = EREF(:,f) + (K(f)^T*Ez(:,ny_ref))*dt;
    ETRN(:,f) = ETRN(:,f) + (K(f)^T*Ez(:,ny_trn))*dt;
    SRC(f) = SRC(f) + (K(f)^T*Ezsrc)*dt;
end;

if(f0 ~= -1)
    Eref0 = Eref0 + (K0^T)*Ez(:,ny_ref)*dt;
    Etrn0 = Etrn0 + (K0^T)*Ez(:,ny_trn)*dt;
    ssSRC = ssSRC + (K0^T)*Ezsrc*dt;
end

if(mod(T,Update) == 0 || T == 1)
    subplot(121);
    draw2d(xa,ya, ERzz, Ez, NPML, 0.03);
    axis equal tight off;
    title(['STEP' num2str(T) ' of ~' num2str(STEPS)]);
    drawnow;

    if(f0 ~= -1)
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% DESIGN Frequency %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %Wave Vector Components
        lam0 = c0/f0;
        k0 = 2*pi/lam0;
        kxinc = 0;
        kyinc = k0*nref;
        m = [-floor(Nx/2):floor(Nx/2)]';
        kx = kxinc -2 * pi*m/(Nx*dx);
        kyR = sqrt((k0*nref)^2 - kx.^2);
        kyT = sqrt((k0*ntrn)^2 - kx.^2);

        %REF

```

```

ref = Eref0/ssSRC;
ref = fftshift(fft(ref))/Nx;
ref = real(kyR/kyinc) .* abs(ref).^2;
REF = sum(ref);

%TRN
trn = Etrn0/ssSRC;
trn = fftshift(fft(trn))/Nx;
trn = real(kyT/kyinc) .* abs(trn).^2;
TRN = sum(trn);

CON = REF + TRN;

disp(['Reflectance= ' num2str(100*REF,'%4.1f') '%']);
disp(['Transmittance = ' num2str(100*TRN,'%4.1f') '%']);
disp(['Conservation = ' num2str(100*CON,'%4.1f') '%']);
disp(' ');
%%%%%%%%%% DESIGN Frequency %%%%%%%%%%%
end

```

```

REF = zeros(1,NFREQ);
TRN = zeros(1,NFREQ);

for f = 1: NFREQ
    %Wave Vector Components
    lam0 = c0/FREQ(f);
    k0 = 2*pi/lam0;
    kzinc = k0*nref;
    m = [-floor(Nx/2):floor(Nx/2)];
    kx = -2 * pi*m/(Nx*dx);
    kzR = sqrt((k0*nref)^2 - kx.^2);
    kzT = sqrt((k0*ntrn)^2 - kx.^2);

    %REF
    ref = EREF(:,f)/SRC(f);
    ref = fftshift(fft(ref))/Nx;
    ref = real(kzR/kzinc) .* abs(ref).^2;
    REF(f) = sum(ref);

    %TRN
    trn = ETRN(:,f)/SRC(f);
    trn = fftshift(fft(trn))/Nx;
    trn = real(kzT/kzinc) .* abs(trn).^2;
    TRN(f) = sum(trn);
end

```

```

CON = REF + TRN;

subplot(122);
plot(FREQ/gigahertz,100*REF,'-r'); hold on;
plot(FREQ/gigahertz,100*TRN,'-b');
plot(FREQ/gigahertz,100*CON,':k'); hold off;
axis([FREQ(1)/gigahertz FREQ(NFREQ)/gigahertz 0 105]);

```

```
    xlabel('Frequency (GHz)');  
    ylabel('%','Rotation',0,'HorizontalAlignment','right');  
    title('REFLECTANCE AND TRANSMITTANCE');  
end  
  
end
```