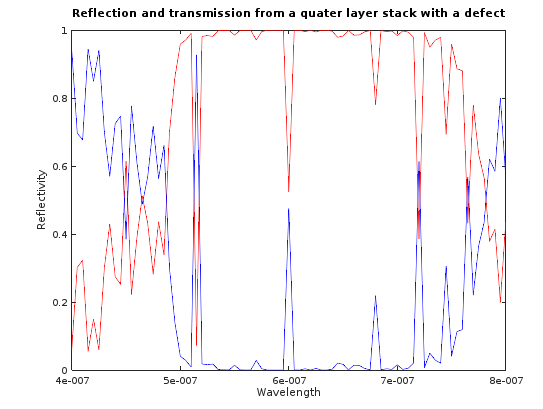
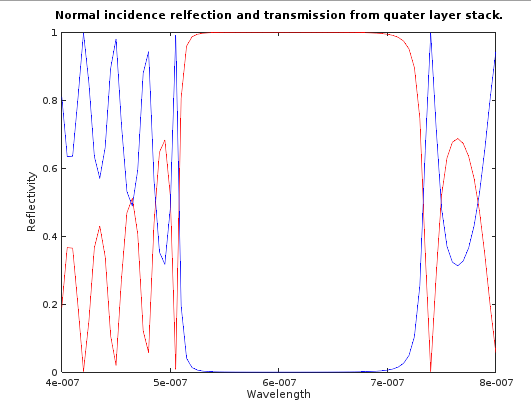
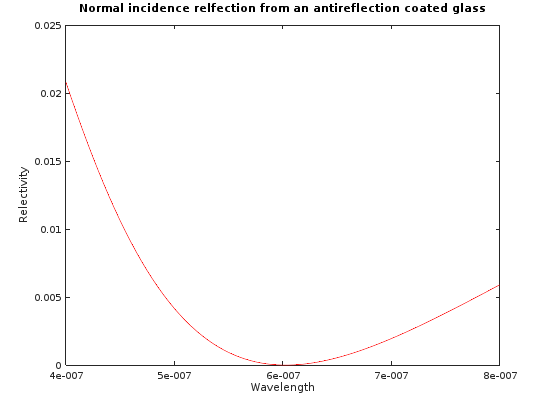
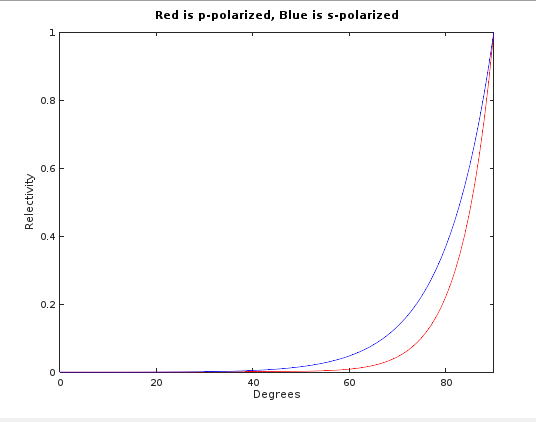
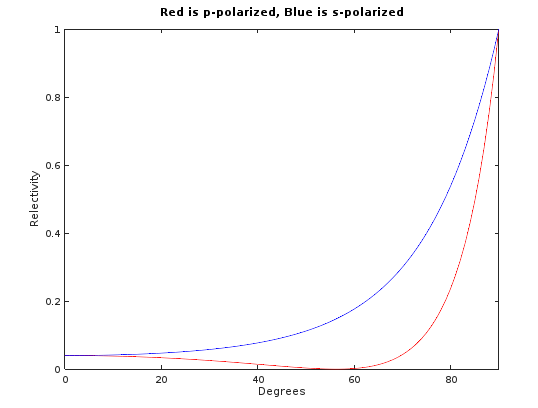
The order of the plots will be from exercise 1 to 4. The fourth and fifth plots will be from exercise 4.



Program codes.

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Transfer Matrix \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% Jonathan Shipley

% Scientific Modeling

% 4/28/17

% Description: This is the first program that runs through relfection and transmission with

% changing angle.

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% clear all previous variables

clear;

n = [1 1.5];

d = [0 0];

%n = [1 1.225 1.5];

%d = [0 123e-9 0];

lambda = 600e-9; % 600 nm

degrees = 0:1:90;

sqrte0u0 = sqrt(8.85418782e-12 \* pi \*4e-7);

numberOflayers = length(n) - 1;

% ask if p or s polarized

choice = input('Input p or so for polarization of light: ', 's')

% convert theta to radians

for k =1:length(degrees)

theta(k) = degrees(k) \* (pi / 180);

end

for k = 1:length(theta)

M = cell(numberOflayers-1, 1);

transferMatrix = [1 0; 0 1];

% find all angles from reflection within layers

phi(1) = theta(k);

for noL = 1:numberOflayers

phi(noL + 1) = acos(sqrt(1 - ((n(noL)/n(noL+1))^2) \* sin(phi(noL))^2));

end

for q = 1:numberOflayers

M{q} = getMatrixOfLayer(lambda, phi(q + 1), n(q + 1), d(q+1), choice);

transferMatrix \*= M{q};

end

m11 = transferMatrix(1,1);

m12 = transferMatrix(1,2);

m21 = transferMatrix(2,1);

m22 = transferMatrix(2,2);

% get y0 and ys

if choice =='s'

y0 = n(1) \* sqrte0u0 \* cos(phi(1));

ys = n(end) \* sqrte0u0 \* cos(phi(end));

elseif choice =='p'

y0 = n(1) \* sqrte0u0 / cos(phi(1));

ys = n(end) \* sqrte0u0 / cos(phi(end));

end

r(k) = (y0 \* m11 + y0\*ys\*m12 - m21 - ys\*m22)/(y0\*m11 + y0\*ys\*m12 + m21 + ys\*m22);

R(k) = abs(r(k))^2;

end

if choice == 'p'

plot(degrees, R, 'r')

hold;

elseif choice =='s'

plot(degrees, R, 'b')

hold;

end

title('Red is p-polarized, Blue is s-polarized')

xlabel('Degrees')

ylabel('Relectivity')

xlim([0 90])

ylim([0 1])

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Transfer Matrix \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% Jonathan Shipley

% Scientific Modeling

% 4/28/17

% Description: This is the second program that runs through relfection and transmission with

% changing wavelength.

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% clear all previous variables

clear;

%n = [1 1.225 1.5];

%d = [0 123e-9 0];

n = [1 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1.5 2.5 1];

d = [0 100 60 100 60 100 60 100 60 100 60 100 60 100 60 100 60 100 60 100 60 0]\*1e-9;

d(11) = 120;

lambda = [400:5:800]\*10^-9;

theta = 0;

sqrte0u0 = sqrt(8.85418782e-12 \* pi \*4e-7);

numberOflayers = length(n) - 1;

% ask if p or s polarized

choice = input('Input p or so for polarization of light: ', 's')

for k = 1:length(lambda)

M = cell(numberOflayers-1, 1);

transferMatrix = [1 0; 0 1];

% find all angles from reflection within layers

phi(1) = theta;

for noL = 1:numberOflayers

phi(noL + 1) = acos(sqrt(1 - ((n(noL)/n(noL+1))^2) \* sin(phi(noL))^2));

end

for q = 1:numberOflayers

M{q} = getMatrixOfLayer(lambda(k), phi(q + 1), n(q + 1), d(q+1), choice);

transferMatrix \*= M{q};

end

m11 = transferMatrix(1,1);

m12 = transferMatrix(1,2);

m21 = transferMatrix(2,1);

m22 = transferMatrix(2,2);

% get y0 and ys

if choice =='s'

y0 = n(1) \* sqrte0u0 \* cos(phi(1));

ys = n(end) \* sqrte0u0 \* cos(phi(end));

elseif choice =='p'

y0 = n(1) \* sqrte0u0 / cos(phi(1));

ys = n(end) \* sqrte0u0 / cos(phi(end));

end

r(k) = (y0 \* m11 + y0\*ys\*m12 - m21 - ys\*m22)/(y0\*m11 + y0\*ys\*m12 + m21 + ys\*m22);

t(k) = (2 \* y0)/(y0 \* m11 + y0\*ys\*m12 + m21 + ys\*m22);

R(k) = abs(r(k))^2;

T(k) = abs(t(k))^2;

end

plot(lambda, R, 'r');

hold;

plot(lambda, T, 'b');

title('Normal incidence relfection and transmission from quater layer stack with a defect.')

xlabel('Wavelength')

ylabel('Relectivity')

% function that returns the transfer matrix of a given layer

function M = getMatrixOfLayer(lambda, thetaOut, indexn, dThick, pOrs)

sqrte0u0 = sqrt(8.85418782e-12 \* pi \*4e-7);

% get delta

delta = ((2 \*pi)/ lambda) \* indexn \* dThick \* cos(thetaOut);

% get y1

if pOrs == 's'

y1 = (indexn \* sqrte0u0) \* cos(thetaOut); % s polarized

elseif pOrs =='p'

y1 = (indexn \* sqrte0u0) / (cos(thetaOut)); % p polarized

end

m11 = cos(delta);

m12 = 1i \* sin(delta)/y1;

m21 = 1i \* y1 \* sin(delta);

m22 = cos(delta);

M(1,1) = m11;

M(1,2) = m12;

M(2,1) = m21;

M(2,2) = m22;