HW 9

Lewis Collum

Updated: April 20, 2020

1

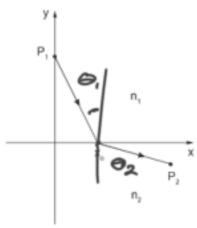
$$u_p = \frac{c}{n}$$

$$t = \frac{\sqrt{x_0^2 + P_{1y}^2}}{c/n_1} + \frac{\sqrt{(P_{2x} - x_0)^2 + P_{2y}^2}}{c/n_2}$$

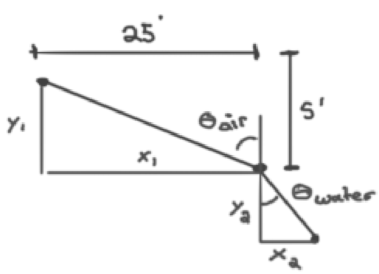
$$\frac{dt}{dx} = \frac{n_1 x_0}{c\sqrt{x_0^2 + P_{1y}^2}} + \frac{-n_2(1-x_0)}{c\sqrt{(1-x_0)^2 + P_{1y}^2}} = 0$$ minimizing time

$$\rightarrow \frac{n_1 x_0}{\sqrt{x_0^2 + P_{1y}^2}} = \frac{n_2(1-x_0)}{\sqrt{(1-x_0)^2 + P_{2y}^2}}$$

$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$



2



$$90° - \theta_{air} = \tan^{-1}\left(\frac{y_1}{x_1}\right)$$
$$= \tan^{-1}\left(\frac{5}{25}\right)$$
$$= 11.31°$$

$$\theta_{air} = 78.69°$$

$$n_{air}\sin\theta_{air} = n_{water}\sin\theta_{water}$$

$$\frac{\sin\theta_{air}}{\sin\theta_{water}} = \frac{n_{water}}{n_{air}} = \frac{\sqrt{81}}{\sqrt{1}} = 9$$

$$\sin\theta_{water} = \frac{1}{9}\sin\theta_{air}$$

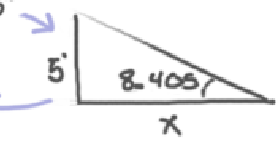
$$\theta_{water} = \sin^{-1}\left(\frac{1}{9}\sin\theta_{air}\right)$$

$$\boxed{\theta_{water} = 6.255°}$$

$$\frac{\sin\theta_{air}}{\sin\theta_{water}} = \sqrt{\frac{\mu_1\epsilon_1}{\mu_2\epsilon_2}}$$

$$\sin\theta_{air} = \sqrt{\frac{1.8}{1}}\sin\theta_{water}$$

$$\theta_{air} = 8.405°$$



$$\tan\theta_{air} = \frac{5}{x}$$

$$x = \frac{5}{\tan(8.405°)}$$

$$\boxed{= 33.84'}$$

3

```python
from numpy import *
nAir = 1
nGan = 3.7
criticalAngle = arcsin(nAir/nGan)
print(f'\[\\theta_c ='
      f'{rad2deg(criticalAngle):.3f}^{{\circ}}\]')

solidAngleCone = 2*pi*(1-cos(criticalAngle)) #4*pi
solidAngleHemisphere = 2*pi

coneCount = 2
solidAngleRatio = coneCount * solidAngleCone/solidAngleHemisphere
print(f'\[\\text{{solid Angle Ratio}} = {solidAngleRatio:.3f}\]')

import pint
unit = pint.UnitRegistry()

wavelength = 560
h = 6.63E-34
c = 3E8
E = solidAngleRatio * h*c/wavelength
print(f'\[\\text{{fraction of optical energy}} = \\text{{{E:.2E} J}}\]')
```

$$\theta_c = 15.680°$$

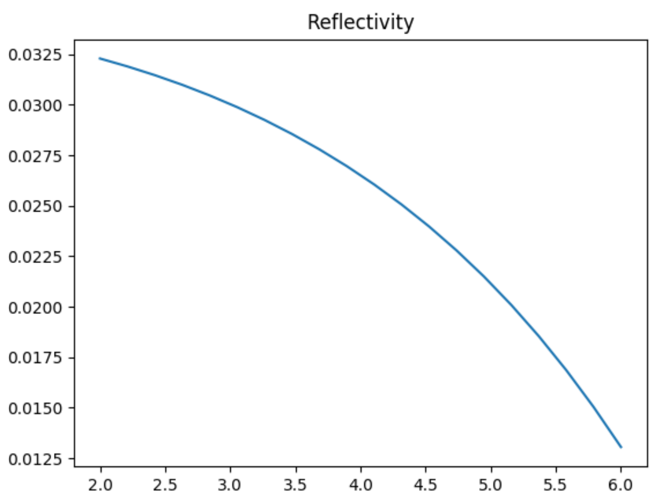
$$\text{solid Angle Ratio} = 0.074$$

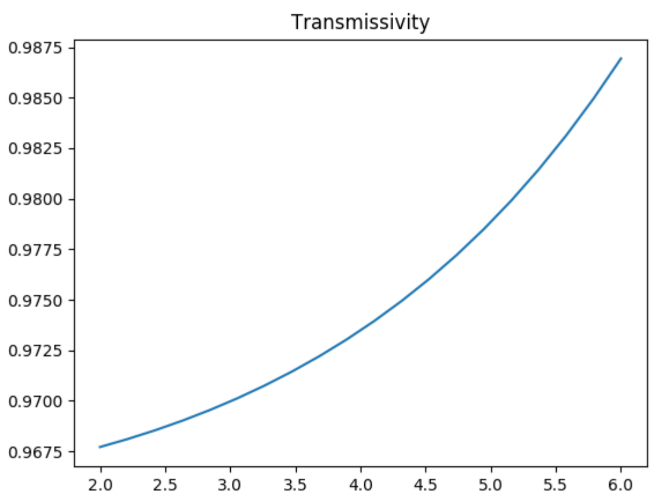
$$\text{fraction of optical energy} = \text{2.64E-29 J}$$

4

```python
from matplotlib import pyplot
import numpy

x = numpy.linspace(2, 6, 20) #wavelenth

#index for quartz
n2 = numpy.sqrt(1 +
                0.6961663*x**2 / (x**2 - 0.0684043**2) +
                0.4079426*x**2 / (x**2 - 0.1162414**2) +
                0.8974794*x**2 / (x**2 - 9.896161**2))
```

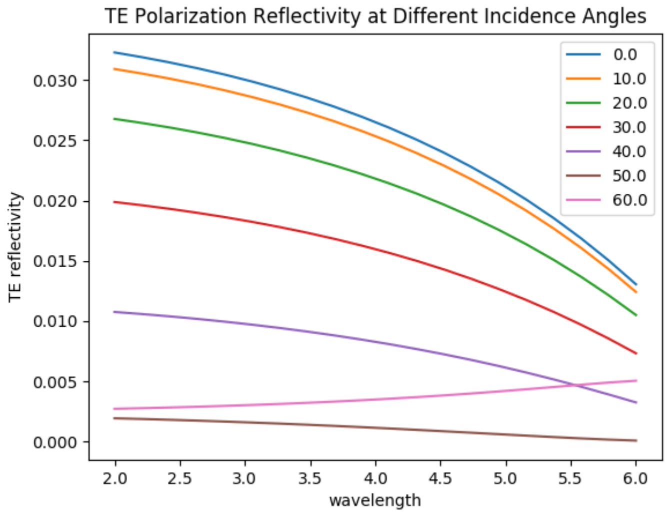
```python
#index for air
n1 = 1

reflectionCoefficient = (n2 - n1)/(n1 + n2)
transmissionCoefficient = 1 + reflectionCoefficient

reflectivity = reflectionCoefficient**2
transmissivity = transmissionCoefficient**2 * (n1/n2)

pyplot.plot(x, reflectivity)
pyplot.title('Reflectivity')
pyplot.savefig('figure/4-re.png')
pyplot.show()

pyplot.plot(x, transmissivity)
pyplot.title('Transmissivity')
pyplot.savefig('figure/4-tr.png')
pyplot.show()
```

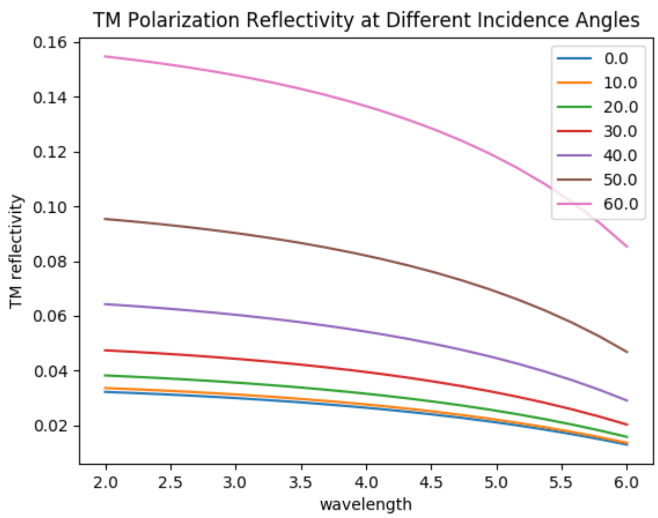




5

```python
from matplotlib import pyplot
import numpy
from numpy import cos, arcsin, sin

def ganIntrinsicImpedance(wavelength):
    return numpy.sqrt(
        0.6961663*wavelength**2 / (wavelength**2 - 0.0684043**2)
        + 0.4079426*wavelength**2 / (wavelength**2 - 0.1162414**2)
        + 0.8974794*wavelength**2 / (wavelength**2 - 9.896161**2)
        + 1).reshape(-1, 1)

def degreeArangeToRadians(start, stop, spacing):
    degreeRange = numpy.arange(start, stop+spacing, spacing)
    radianRange = numpy.deg2rad(degreeRange)
    radianRangeAsColumnVector = radianRange.reshape(-1, 1).T
    return radianRangeAsColumnVector

def snellsTransmissionAngle(n1, n2, a1):
    return arcsin(sin(a1)*(n1/n2))

def perpendicularReflection(n1, n2, a1, a2):
    return ((n2*cos(a1) - n1*cos(a2))
            / (n2*cos(a1) + n1*cos(a2)))**2

def parallelReflection(n1, n2, a1, a2):
    return ((n2*cos(a2) - n1*cos(a1))
            / (n2*cos(a2) + n1*cos(a1)))**2



wavelength = numpy.linspace(2, 6, 20)
n2 = ganIntrinsicImpedance(wavelength)
n1 = 1 #Air

incidenceAngles = degreeArangeToRadians(
    start = 0,
    stop = 60,
    spacing = 10)

transmissionAngles = snellsTransmissionAngle(
    n1 = n1,
    n2 = n2,
    a1 = incidenceAngles)

transverseElectricReflectivity = perpendicularReflection(
    n1 = n1,
    n2 = n2,
    a1 = incidenceAngles,
    a2 = transmissionAngles)

transverseMagneticReflectivity = parallelReflection(
    n1 = n1,
    n2 = n2,
    a1 = incidenceAngles,
    a2 = transmissionAngles)


pyplot.plot(wavelength, transverseElectricReflectivity)
pyplot.title('TE Polarization Reflectivity at Different Incidence Angles')
pyplot.xlabel('wavelength')
pyplot.ylabel('TE reflectivity')
pyplot.legend(*numpy.round(numpy.rad2deg(incidenceAngles)))
pyplot.savefig('figure/5-te.png')
pyplot.show()

pyplot.plot(wavelength, transverseMagneticReflectivity)
pyplot.title('TM Polarization Reflectivity at Different Incidence Angles')
pyplot.xlabel('wavelength')
pyplot.ylabel('TM reflectivity')
pyplot.legend(*numpy.round(numpy.rad2deg(incidenceAngles)))
pyplot.savefig('figure/5-tm.png')
pyplot.show()
```





---

6 - Brewster's Angle

```python
from numpy import arctan, sqrt, rad2deg

wavelength = 580E-9
schottIndex = 1.5427
airIndex = 1
brewsterAngle = rad2deg(arctan(sqrt(schottIndex/airIndex)))

print(f'\[\\theta_B'
      f' = \\tan^{{-1}}\sqrt{{\dfrac{{\epsilon_2}}{{\epsilon_1}}}}'
      f' = {brewsterAngle:.2f}^{{\circ}}\]')
```

$$\theta_B = \tan^{-1}\sqrt{\dfrac{\epsilon_2}{\epsilon_1}} = 51.16^{\circ}$$