lab1

April 9, 2020

[1]:

```
%matplotlib notebook
import numpy as np # math library
from bqplot import pyplot as plt # plotting library
```

1 ENGN 1560 Lab Report 1

Jarod Boone

Data aquired with Haley Mander

In this lab we went into BH 190 to perform various refraction/reflection based measurements of a laser beam through a rotating prism centered on the optic axis. We use a Uniphase laser and measure the way it reflects and refracts as it passes through the prism at different angles.

The first thing we did was use a half wave plate to try and get the laser beam to be as close to **p** polarization as possible.

Next we angled the prism to be at normal incidence to the laser beam. This normal incidence occured when the rotating plate that the prsim was mounted on was at `296` degrees. The beam power entering the prism was approximately `12.5` micro Watts.

[2]:

```
normal_angle = 296
incident_power = 12.5e-6
```

Next we swept over various angles of the rotating plate and measured the power of the refelcted beam.

Angle of Rotation (degrees)	Reflected Beam Power (nano Watts)
300	9.9
310	9.9
320	10.8
325	12
328	12.5
330	11
332	9.7
336	8.5
340	7.6
344	7.3

By using tha angle of rotation of normal incidence we can determine the incindent angle at the prism surface incidence. Additionally by using the incident beam power we can use the refelected beam power to get the reflectance

`reflectance = (reflected_power)/(incident_power)`

[3]:
```
aor = [300 ,310 ,320 ,325 ,328 ,330 ,332 ,336 ,340 ,344]
rbp = [9.9e-6, 9.9e-6, 10.8e-6, 12e-6, 12.5e-6, 11e-6, 9.7e-6, 8.5e-6, 7.6e-6,␣
 ↪7.3e-6]

angle_of_incidence = list(map(lambda x: x - normal_angle,aor))
reflectance = list(map(lambda x: x/incident_power,rbp))

plt.figure(title='Reflectance vs Angle of Incidence',interaction=None)

plt.plot(angle_of_incidence,reflectance)
plt.show(display_toolbar=False)

plt.xlabel("Angle of Incidence")
plt.ylabel("Reflectance")
```

```
Figure(axes=[Axis(scale=LinearScale()), Axis(orientation='vertical', scale=LinearScale())], fi
```

This data is erroneous because theory tells us that perfectly **p** polarized light should experience a logarithmic decay to zero as it approaces brewsters angle.

This discrepancy probably comes from some sort of measuring error or the fact that the light is probably not perfectly P polarized.

Despite these errors we still extrapolate brewsters angle to be the point where the reflectance is closest to zero. Thus we take

[4]:
```
brewsters_angle = 50
```

From this we can use the equation for brewsters angle at an air to glass interface to determine the refractive index of the prism. If we assume the refractive index of air is 1 we have

`brewsters_angle = atan(n_t/(1)) = atan(n_t/(1))`

[5]:
```
nt = np.tan(np.deg2rad(brewsters_angle))
print(f'The calculated index of refraction is {nt}')

def theta_in2theta_t(theta_in):
    return np.rad2deg(np.arcsin(np.cos(np.deg2rad(theta_in)/nt)))

def getReflectance(theta_in) :
    Ot = np.deg2rad(theta_in2theta_t(theta_in))
    Oi = np.deg2rad(theta_in)
    return ((np.cos(Ot) - nt*np.cos(Oi))/(np.cos(Ot) + nt * np.cos(Oi)))**2
```

```
domain = np.linspace(5,50,100)
R = list(map(getReflectance,domain))

plt.figure(title='Reflectance vs Angle of Incidence',interaction=None)



plt.plot(domain,R,'r--')
plt.scatter(angle_of_incidence,reflectance)


plt.show(display_toolbar=False)

plt.xlabel("Angle of Incidence")
plt.ylabel("Reflectance")
```

```
The calculated index of refraction is 1.19175359259421
```

```
Figure(axes=[Axis(scale=LinearScale()), Axis(orientation='vertical', scale=LinearScale())], fi
```

In this plot the red dotted line is the Reflectance predicted by theory and the blue points are our data.

For the next part of the lab we wanted to measure the angle of deviation for different angles of incidence.

We repositioned the prism and got a new angle of rotation that corresponded to normal incidence. We then triangulated the angle of deviation by taking a fixed length of 7 inches away from the prism along the normal axis and determing the length we would need to make a perpendicualr side (opposite to the angle of deviation) in order to have the hypotnuse of a triangle made be the path of the transmitted beam.

Angle of Rotation (degrees)	Length of Opposite Side (inches)
240	25
230	10.5
231	11.5
232	13
239	15.5
219	4
221	5
225	7
229	9
210	2

Because the fixed length leg is along the normal we know that the the angle between that and the hypotenuse will be the angle of deviation. Using some basic trigenometry we can solve for this angle

[6]:

```
normal_angle = 176
aleg = 7

aor = [240, 230, 231, 232, 239, 219, 221, 225, 229, 210]
oleg = [25, 10.5, 11.5, 13, 15.5, 4, 5, 7, 9, 2]

# need to sort these for a line plot
sort_zip = sorted(zip(aor,oleg))
s_aor = [x for x,_ in sort_zip]
s_oleg = [x for _,x in sort_zip]

# calculate the angle of deviation on the sorted opposite leg
angle_of_deviation = list(map(lambda x: np.rad2deg(np.arctan(x/aleg)),s_oleg))
angle_of_incidence = list(map(lambda x: x - normal_angle,s_aor))

plt.figure(title='Angle of Deviation vs Angle of Incidence',interaction=None)

plt.plot(angle_of_incidence,angle_of_deviation)
plt.show(display_toolbar=False)

plt.xlabel("Angle of Incidence")
plt.ylabel("Angle of Deviation")
```

```
Figure(axes=[Axis(scale=LinearScale()), Axis(orientation='vertical', scale=LinearScale())], fi
```

The angle of deviation is minimized at an angle of incidence of about 34 degrees. Here the angle of deviation takes on a value of about 15 degrees

Assuming that this prism is equilateral then we know that `beta/2 = 30` degrees and can thus use the formula for minimum deviation through a prism

`sin(theta_dev - theta_in + beta)= n*sin(beta/2)`

To determine the refractive index of the prism

[7]:

```
theta_dev = 15
theta_in = 34
beta = 60
n = np.sin(np.deg2rad(theta_dev - theta_in + beta))/np.sin(np.deg2rad(beta/2))
print(n)
```

```
1.3121180579810148
```

Finally we can use this altrenative calculation of the refractive index of the prism to determine the theoretical angle of deviation for a given angle of incidence

`theta_dev = theta_in - beta + asin(n * sin(beta - asin(sin(theta_in)/n)))`

[8]:

```
def getThetaDev(theta_in) :
    Oi = np.deg2rad(theta_in)
    return theta_in - beta + np.arcsin(n * np.sin(np.deg2rad(beta) - np.
 ↪arcsin(np.sin(Oi)/n)))

domain = np.linspace(34,64,100)
Odev = list(map(getThetaDev,domain))

plt.figure(title='Angle of Deviation vs Angle of Incidence',interaction=None)

plt.plot(domain,Odev,'r--')
plt.scatter(angle_of_incidence,angle_of_deviation)
plt.show(display_toolbar=False)

plt.xlabel("Angle of Incidence")
plt.ylabel("Angle of Deviation")
```

```
Figure(axes=[Axis(scale=LinearScale()), Axis(orientation='vertical', scale=LinearScale())], fi
```

Again the red dotted line represents the Angle of Deviation predicted bt theory and the blue dots are our measured data.

In both experiments the data did not match the theory. The index of refraction from each experiment are similar but not the same

`n1 = 1.2 n2 = 1.3`

I would say the first experiment is more susceptible to error because it relies on the perfect **p** type polarization of the laser beam.

Based on the index of refraction and the physical properties of the crystal I would guess that the prism is maybe some type of acrylic glass.

[ ]:

```
```