The Faraday Effect

Physics 374

Dr. Brown

Nick Zorn
Lab Partner: Miguel Dominguez Carreño

May 5, 2017

1 Abstract

Michael Faraday explained the relationship between magnetic fields and the propagation of light through dielectric materials by measuring the rotation of the plane of polarization. Using an electromagnet apparatus, we measured this rotation with three different samples. Using our measurements, we were able to calculate the Verdet constants and successfully compared our results to known values. To accomplish this, we used a Visual Basic program to fit our data to the plot. We also estimated the thickness of a sample using a known Verdet constant. Finally, we verified the Verdet constant decreases as the wavelength increases by calculating the Verdet constant at four different wavelengths. Using a table of known indices of refraction and corresponding wavelengths, we were able to fit the given data and solve for our Verdet constant using a different method.

2 Introduction

The Faraday Effect was discovered by Michael Faraday in 1845 [4]. Faraday was looking for a relationship between light and magnetism. Unfortunately, he was unable to observe any measurable difference until he tried using glass containing lead.

2.1 Theory

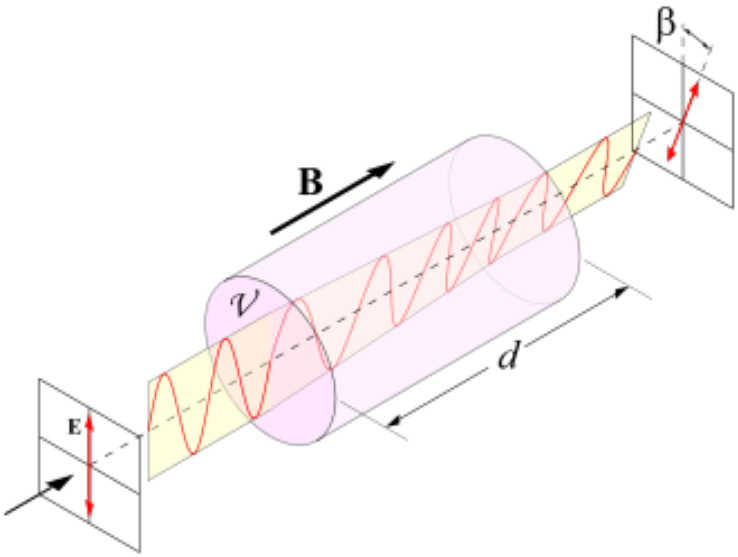


Figure 1: The Faraday Effect

The Faraday "Effect" occurs wherever a magnetic field is present, but is only observable in dielectric materials. These are materials that are polarized by an applied electric field. Examples include porcelain (ceramic), mica, glass and plastics [4].

In dielectric materials, light is decomposed into ordinary and extraordinary rays [5]. These rays are then circularly polarized with opposite rotations with different indices of refraction [6].

The charged particles in the dielectric create their own magnetic field [5]. The ordinary and extraordinary waves propagating will either be working with or against the magnetic field. The wave working with the magnetic field will see an increase in velocity. The wave working against the magnetic field will see a decrease in velocity.

This results in light propagating through the material at different speeds due to the magnetic field [6]. At the end of the medium, the rays combine linearly, resulting in an offset in polarization as shown in the following figure:

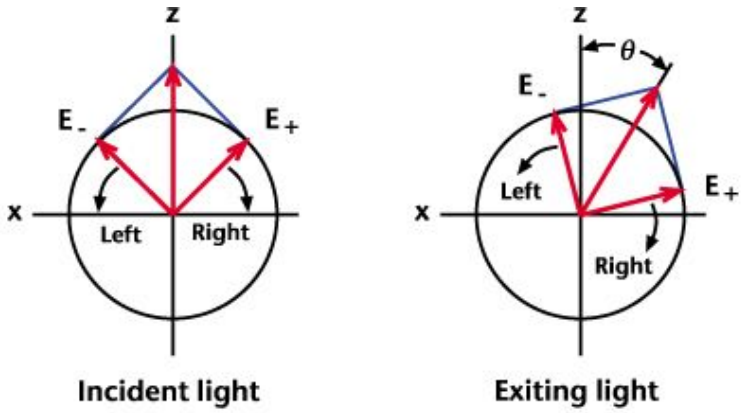


Figure 2: Circular Birefringence

2.2 Relevant Equations

The simplest equation to describe the Faraday Effect is shown below [2]:

$$\theta = V\beta l \tag{1}$$

In the basic equation, the angle of rotation (θ) is proportional to the strength of the magnetic field (β) multiplied by the thickness of the medium (l) multiplied by the **Verdet Constant** (V). This constant varies depending on the type of material and the wavelength of the light propagating through it.

The other relevant equation is shown below [2]:

$$V = \frac{-1}{2}\frac{e}{m}\frac{\lambda}{c}\frac{dn}{d\lambda} \tag{2}$$

Another way to solve for the Verdet constant involves the mass-to-charge ratio ($\frac{e}{m}$), the speed of light (C), the wavelength of the light propagating through the material (λ) and the index of refraction with respect to the wavelength ($\frac{dn}{d\lambda}$).

3 Experimental Setup

To observe the Faraday Effect, also known as Faraday Rotation, we will be using an Atomic Laboratories research electromagnet (Catalog Number 79637) connected to a power supply as shown below:

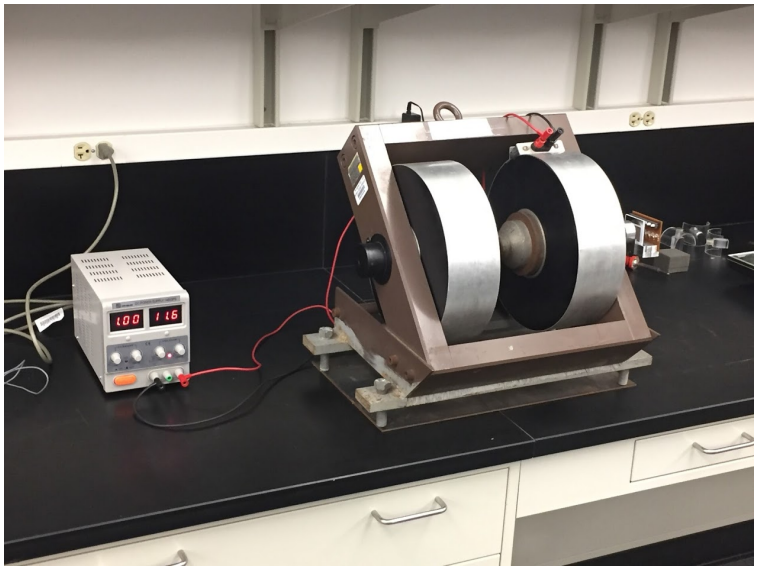


Figure 3: Apparatus Overview

The large silver "platters" contain a solenoid, which generates an electromagnetic field. The direction of the magnetic field points in the same direction as the direction the light propagates through the apparatus.

The two electromagnets are wired in parallel, as shown below:

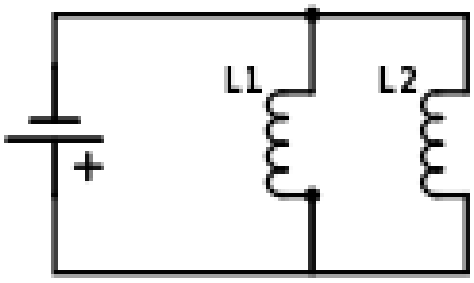


Figure 4: Electrical Schematic

In between the electromagnets, we place our dielectric sample. For this experiment, we used three different samples:

- Dense flint glass
- Light flint glass
- Terbium gallium garnet (T.G.G.)

The Atomic Laboratories apparatus has a small aperture for light to pass through, which was nearly flush against each sample as shown below:

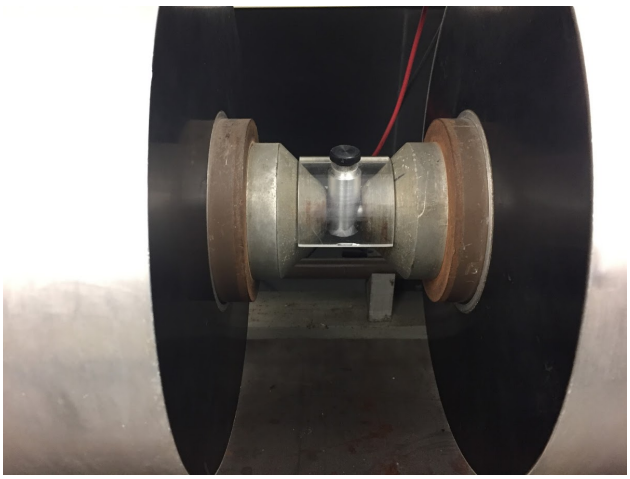


Figure 5: Sample Placement

Our light source is a circuit board containing various colors of LED lights. The lights were activated with a push button above each light, as shown below:

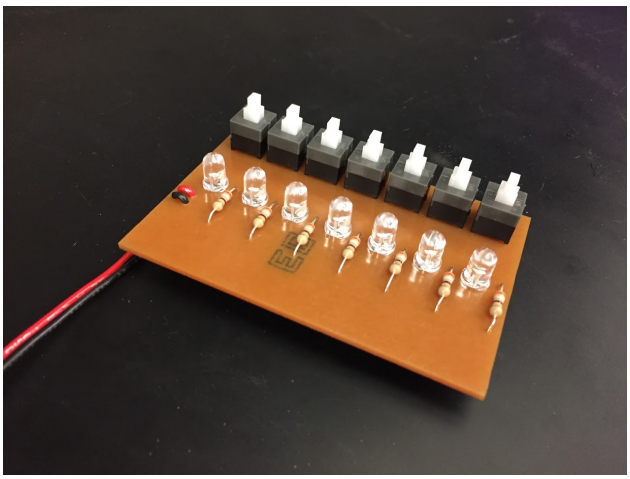


Figure 6: LED Circuit Board

The actual wavelength of each color LED light was not known. For the purposes of this experiment, we used an average wavelength for each color.

Between the light source and the electromagnet was first a diffuser and then a fixed polarizer. The diffuser helped lower the intensity of the light from the LED, making it easier to observe the necessary light changes.

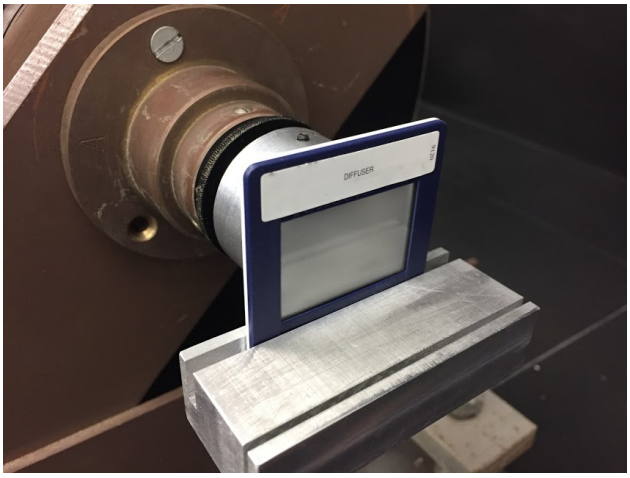


Figure 7: Diffuser

The fixed polarizer takes all of the incoming light waves and only allows waves in a certain orientation to pass through. For example, if the polarizer had vertical openings, only waves oriented vertically would be able to pass through unobstructed.

The following image illustrates how a polarizer works:

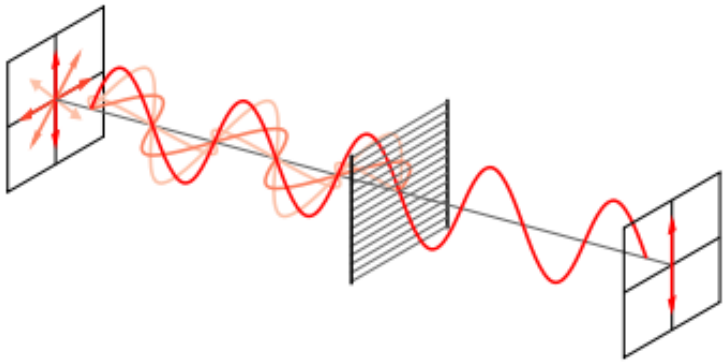


Figure 8: How Polarizers Work

On the opposite side of the apparatus, we had an adjustable polarizer that allowed us to measure the rotation of the plane. The bottom dial represented whole degree increments, and the top represented $\frac{1}{10}$ degree increments.

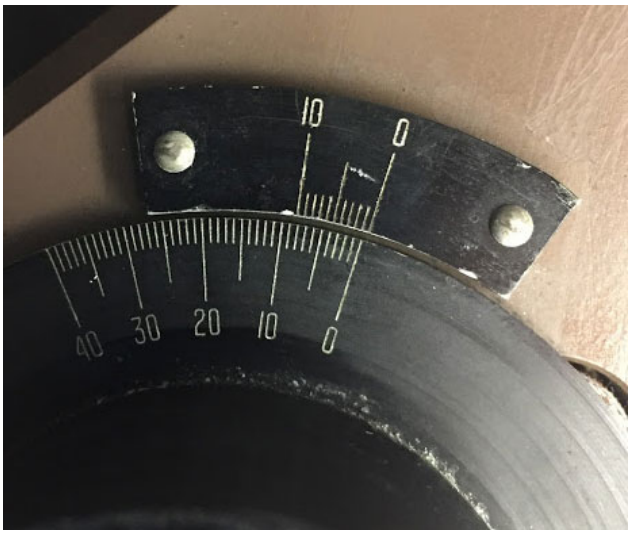


Figure 9: Adjustable Polarizer Dial

When taking measurements, we first looked at the bottom portion of the dial (0 to 40 degrees). We looked to see which tick lined up with the zero on the top of the dial. If we were between numbers, we looked at the top dial to determine the decimal, similar to the method shown below:

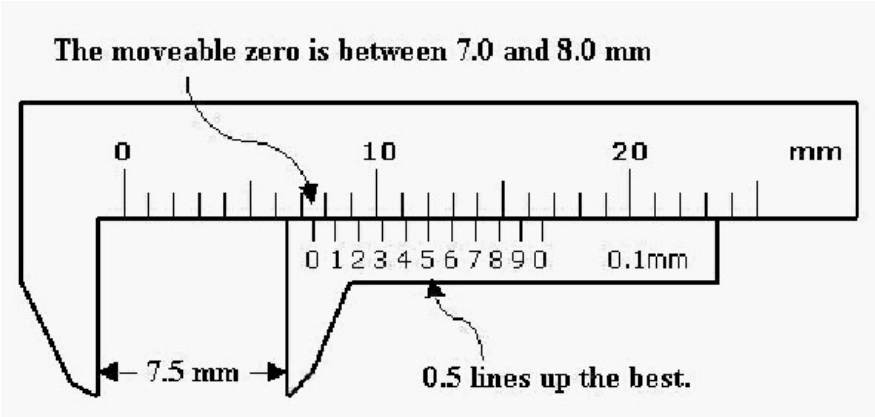


Figure 10: Vernier Scale

As with many of the readings in this lab, this was subjective. In order to minimize error, we took multiple trials and averaged the results.

3.1 Schematic

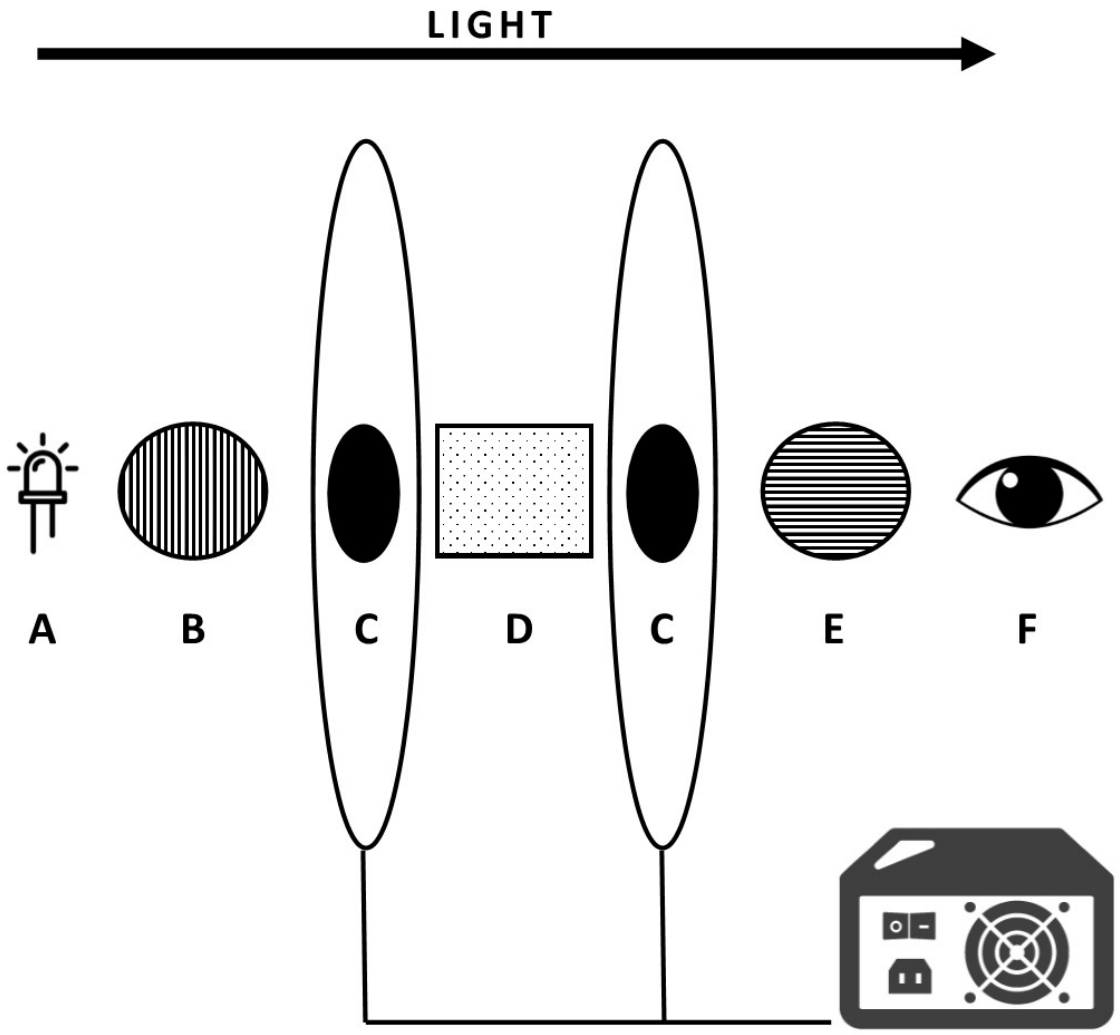


Figure 11: Schematic

A: LED Bulb
B: Fixed Polarizer
C: Electromagnet
D: Sample
E: Adjustable Polarizer
F: Observer

Not labeled in the lower-right hand corner is the power supply.

4 Results and Discussion

4.1 Calibration

Rather than measuring the magnetic field every time we adjusted the current, we measured the magnetic field at 0.5 amp intervals. We then plotted our magnetic field strength as a function of current. Using these measurements, we were able to estimate the strength of the magnetic field using our current measurements.

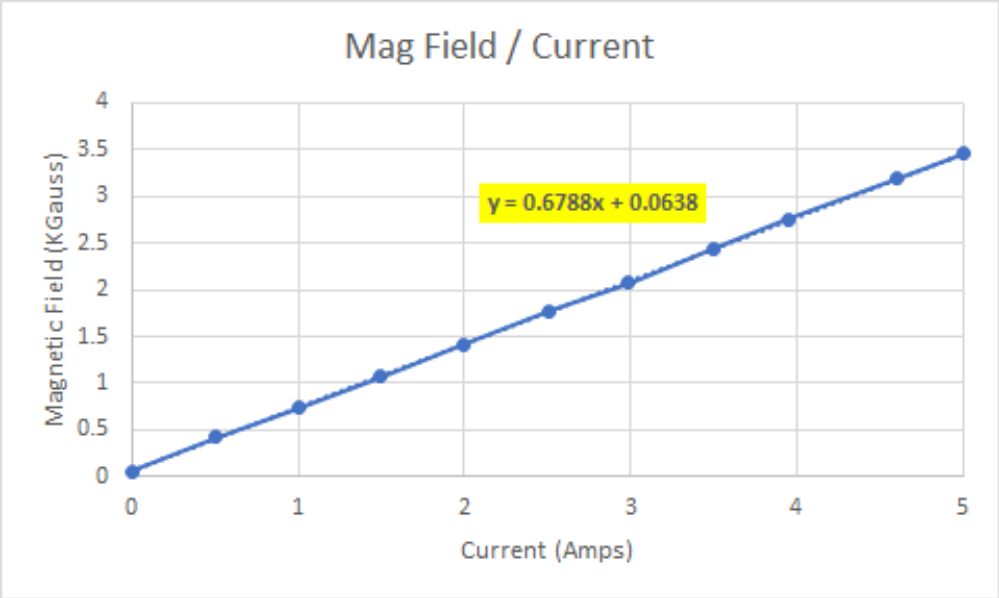


Figure 12: Magnetic Field vs. Current

4.2 Sample 1: Unknown

Our first sample was unknown. It had a thickness of 0.18 m and was transparent.

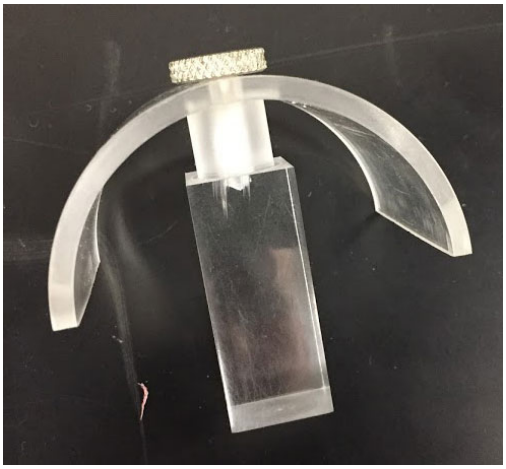


Figure 13: Unknown Sample

Our table of Verdet constants were given in wavelengths of 5893 Å, which is 589.3 nm. On the visible light spectrum, this would fall between red and orange [7]. We initially tried taking angle measurements with the red LED. However, the red light intensity made it difficult to accurately determine when the polarizer was adjusted correctly.

We then chose to take our measurements using the orange LED and took the following data:

Table 1: Unknown Sample

Current	Mag. Field	T1	T2	T3	Avg.
0.0	0	30	27	28	28.33
1.0	678.8	28	27	29	28.00
2.0	1357.6	27	26	27	26.67
3.0	2036.4	25	25	26	25.33
4.0	2715.2	24	25	25	24.67
5.0	3394	23	24	25	24.00

Using the data, we were able to calculate our Verdet constant using a Visual Basic program that we wrote (see Appendix A). The program fitted our data to a linear plot, as shown below:

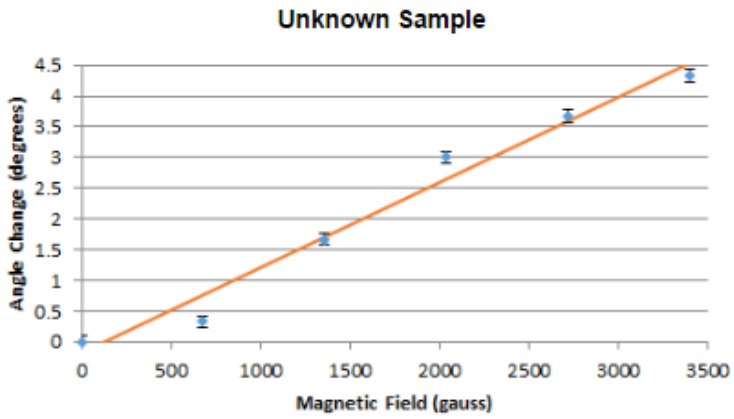


Figure 14: Unknown Sample

Our program determined our slope to be 0.00139, which we divided by our sample thickness of 0.18 m to get our Verdet constant of $0.0007722\frac{degrees}{cm*gauss}$. After converting our constant into $\frac{m.p.a.}{cm*gauss}$ by multiplying our value by 60, we ended up with a value of $0.046332\frac{m.p.a.}{cm*gauss}$.

Comparing our constant to the table of known values, we discovered that our constant matched very closely to the known dense flint glass $V = 0.0442\frac{m.p.a.}{cm*gauss}$ [2]. It's worth noting the known constant was determined using a wavelength of 589.3 nm, and we estimated our wavelength to be in the range of 590 - 620 nm [7]. As we discovered later in this lab, the wavelength of light affects the Verdet constant.

That being said, we estimated how closely our calculated value matched the known value:

$\frac{Known-Calculated}{Calculated} * 100 = \frac{0.0442-0.046332}{0.046332} * 100 = 4.60\%$ Error

4.3 Sample 2: Terbium Gallium Garnet

Our second sample was Terbium Gallium Garnet, but of unknown thickness. We were given the Verdet constant of $-134\frac{radians}{m*T}$ at 632 nm [3].

Figure 15: Unknown Sample

We chose to use the red LED to take our measurements as 632 nm is within the range of red light (620 - 750 nm) [7]. Unlike our previous sample, we were able to accurately measure the angle of rotation using the red LED. The table below shows our data:

Table 2: Terbium Gallium Garnet

Current	**Mag. Field**	**T1**	**T2**	**T3**	**T4**	**T5**	**Avg**
0	0	5.8	5.3	5.9	6.2	6.5	5.94
1	678.8	6	4.9	4.4	3.3	5.3	4.78
2	1357.6	3.4	4.7	3	3.4	4.3	3.76
3	2036.4	3.5	3.5	3	4	3.5	3.5
4	2715.2	4	2.7	3.2	1.4	2.3	2.72

As previously stated, the Verdet constant for this sample was given to us, but the thickness of the sample was not. By converting our angle measurements to radians, our magnetic field to Teslas, and by manipulating the Faraday Equation, were able to solve for the thickness:

$d = \frac{\theta}{V*B} = \frac{-0.0561996}{-134*0.27152} = 1.5$ m

Although we have no way of verifying the thickness of the sample, our estimated thickness certainly appears reasonable upon observation.

It's worth nothing the Verdet constant of the T.G.G. is negative [3], and our observations confirm the Verdet constant **should be** negative as shown in our plot:

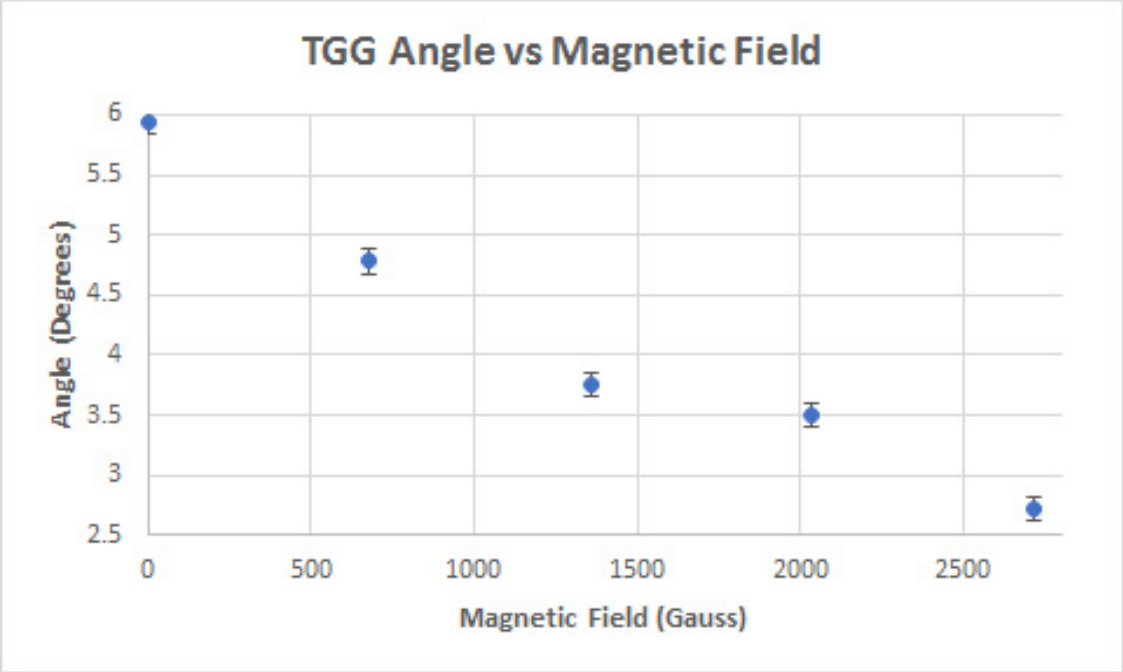


Figure 16: Terbium Gallium Garnet

4.4 Sample 3: Light Flint

Our final sample was light flint. It had a thickness of 1.8 cm and had a known Verdet constant of $0.0317 \frac{m.p.a.}{cm * Oersted}$ at 589 nm [2].

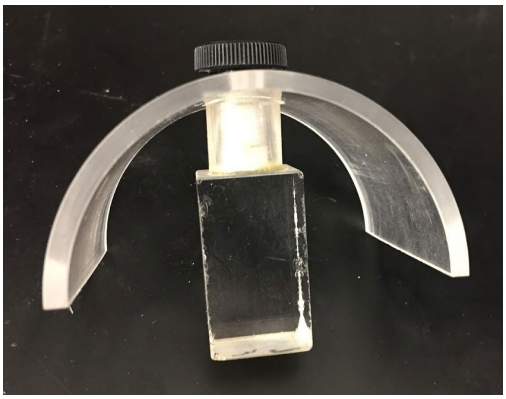
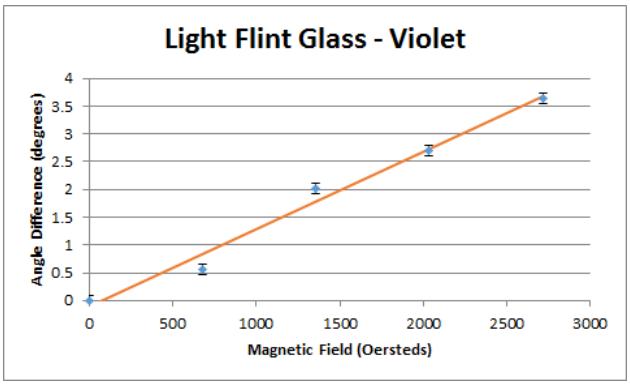


Figure 17: Light Flint Sample

For this sample, we took our angle measurements using four different colors (wavelengths):

Table 3: Violet

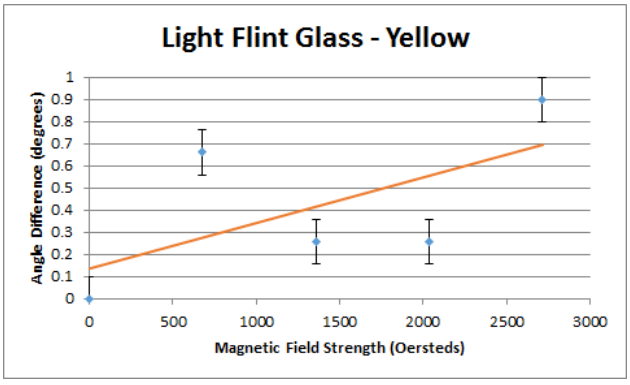
Current	T1	T2	T3	T4	T5	Avg
0	5.5	5	5	5	5.4	5.18
1	6	5.5	5.5	5.7	6	5.74
2	7.2	7.3	7.3	7.5	6.7	7.2
3	7.8	7.4	8	8	8.2	7.88
4	8.4	8.7	9	9	9	8.82



We calculated our Verdet constant of **violet** to be $0.0462\frac{m.p.a.}{cm*gauss}$.

Table 4: Yellow

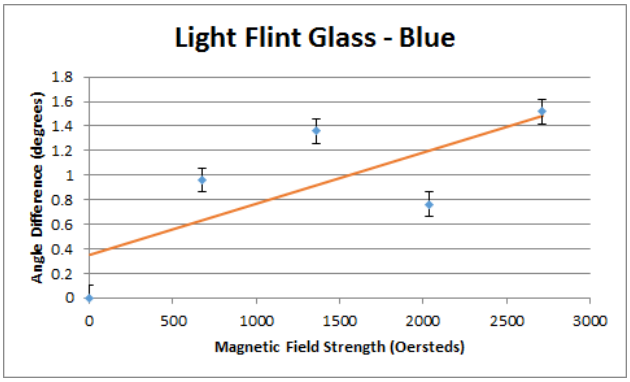
Current	T1	T2	T3	T4	T5	Avg
0	6.5	5.4	5	5.5	5.5	5.58
1	6.2	6.3	6.3	6.4	6	6.24
2	6.7	7	6.4	6.4	6	6.5
3	7	6.7	5.7	7.3	7.1	6.76
4	7.6	7.2	7.5	7.5	8.5	7.66



We calculated our Verdet constant of **yellow** to be $0.0.0069\frac{m.p.a.}{cm*gauss}$.

Table 5: Blue

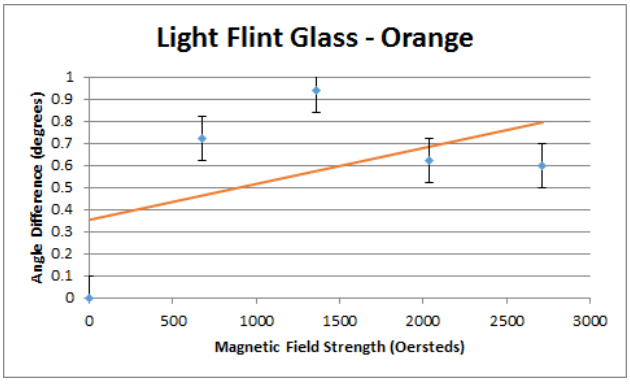
Current	T1	T2	T3	T4	T5	Avg
0	5.5	6	6	5.5	6	5.8
1	6.4	7	7	7	6.4	6.76
2	8	7.2	8.5	8.5	8.4	8.12
3	8.2	9.2	9.3	9	8.7	8.88
4	10.4	10.4	10.4	10.3	10.5	10.4



We calculated our Verdet constant of **blue** to be $0.0139\frac{m.p.a.}{cm*gauss}$.

Table 6: Orange

Current	T1	T2	T3	T4	T5	Avg
0	5.2	5.3	5.4	5.5	6	5.48
1	5.7	6.8	5.7	6.4	6.4	6.2
2	6.7	7.2	7.4	7.3	7.1	7.14
3	8.3	7.5	7.6	7.4	8	7.76
4	8.5	8.2	8.3	8.4	8.4	8.36



We calculated our Verdet constant of **orange** to be $0.0054\frac{m.p.a.}{cm*gauss}$.

For our **orange** sample, we calculated the error in our Verdet constant.

$\frac{\sigma_v}{v}^2 = \frac{\sigma_m}{m}^2 + \frac{\sigma_l}{l}^2$

Verdet Constant: $0.0054 \frac{m.p.a.}{cm*gauss}$

Uncertainty in Slope: 0.000027 (from Visual Basic)

Uncertainty in Thickness: 0.1 cm (Estimated)

$\frac{\sigma_v}{0.0054}^2 = \frac{0.000027}{0.0001625}^2 + \frac{0.1}{1.8}^2$

$\frac{\sigma_v}{0.0054}^2 = 0.0307$

$\frac{\sigma_v}{0.0054} = 0.1752$

$\sigma_v = 0.000946 \frac{m.p.a.}{cm*gauss}$

After calculating the Verdet constants of the four different wavelengths using our Visual Basic program (see Appendix A), we plotted them versus wavelength as shown in our graph:

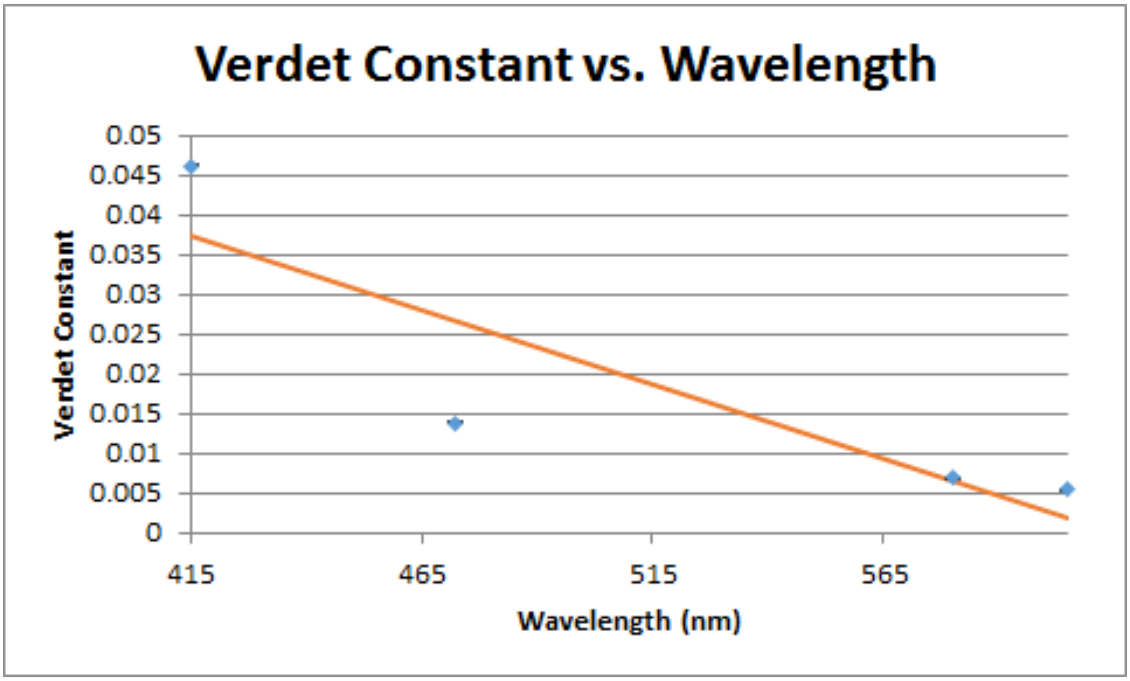


Figure 18: Verdet Constant vs. Wavelength

The slope of our curve is verified using the following equation:

$V = \frac{-1}{2}\frac{e}{m}\frac{\lambda}{c}\frac{dn}{d\lambda}$

Essentially, the Verdet constant decreases as the wavelength increases, and vice-versa.

To further explore this equation, we were given a table of known indices of refraction for various flint glass samples [1]:

Table 7: Flint Samples

(nm)	Light Flint	Dense Flint	Heavy Flint
400	1.5932	1.6912	1.8059
460	1.5853	1.6771	1.7843
500	1.5796	1.677	1.7706
560	1.5757	1.6951	1.7611
600	1.5728	1.6542	1.7539
650	1.5703	1.6503	1.7485
700	1.5684	1.6473	1.7435
750	1.5668	1.645	1.7389

Graphing the index of refraction versus wavelength gives us the following plot:

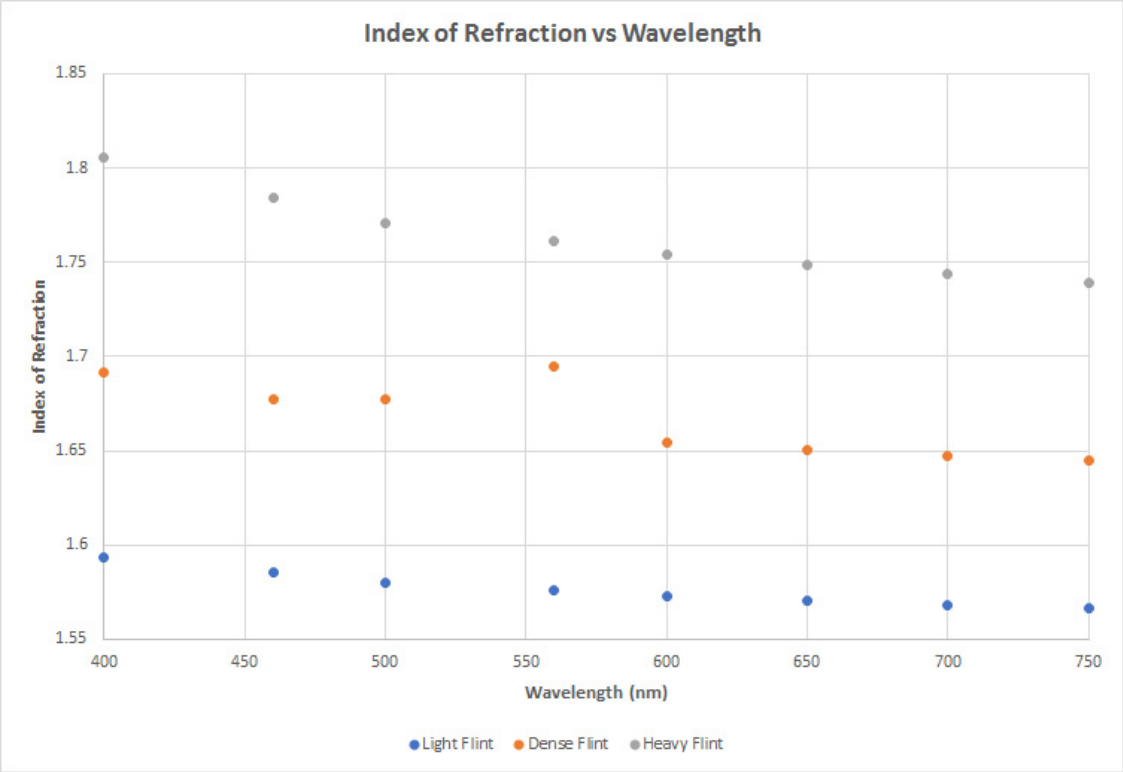


Figure 19: Flint Samples

As you can see from the plot, the light flint sample has the lowest index of refraction compared to dense and heavy flint. For our calculations, we will be using the **light flint** data.

In order to determine our Verdet constant, we needed to solve for $\frac{dn}{d\lambda}$. We did this by first plotting the data for light flint and performing a fit using Cauchy's equation $n = A + \frac{B}{\lambda^3}$ within our Visual Basic Program (see Appendix B) as shown in the following plot:

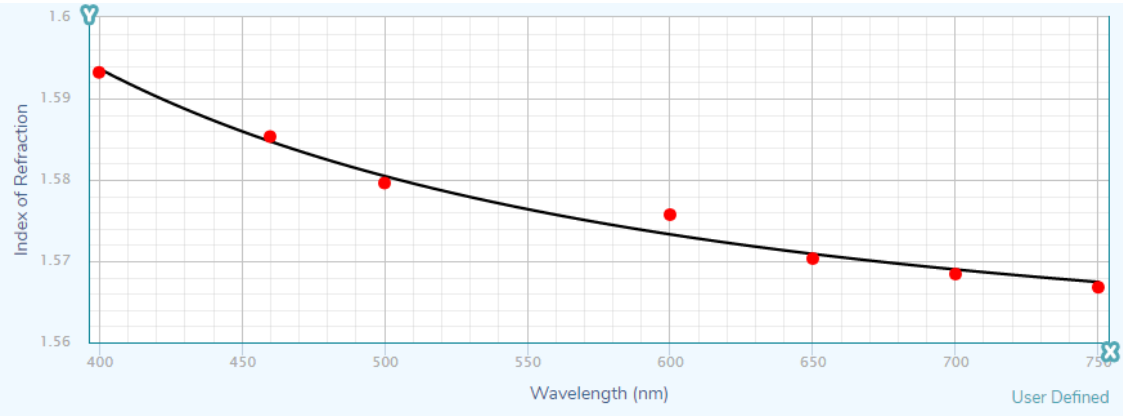


Figure 20: Plot Fitted with $n = A + \frac{B}{\lambda^3}$

From the plot, we determined:

$A = 1.557017$
$B = 5855.794$

Using the following equation, we determined $\frac{dn}{d\lambda}$:

$\frac{dn}{d\lambda} = \frac{-2B}{\lambda^3} = \frac{-2*5855.794}{685^3} = -3.6437 * 10^{-5}$

Using our calculated value of $\frac{dn}{d\lambda}$, we were able to solve for our Verdet constant by plugging it into the following equation:

$V = \frac{1}{2}\frac{e}{m}\frac{\lambda}{c}\frac{dn}{d\lambda} = 7.3165\frac{rad}{T*m}$

After converting the Verdet constant into $\frac{m.p.a.}{cm*gauss}$, we ended up with a value of 0.0250. The known Verdet constant of light flint glass is $0.0317\frac{m.p.a.}{cm*gauss}$. We determined our % error using the following method:

$\frac{Known-Calculated}{Calculated} * 100 = \frac{0.0317-0.0250}{0.0250} * 100 = 26.8\%$ Error

The known Verdet constant was calculated using a slightly different wavelength. Therefore, the value we calculated will differ slightly.

5 Conclusion

The purpose of this lab was to gain a better understanding of the Faraday Effect. We took angle measurements of three different samples. For each sample, we were presented with a different challenge.

For our first sample, we attempted to determine the composition of the sample after calculating the Verdet constant. We ended up with a Verdet constant of $0.046332\frac{m.p.a.}{cm*gauss}$, which almost exactly matched the constant of dense flint glass with only 4.60% error.

Our second sample was Terbium Gallium Garnet. We were given the Verdet constant, but the thickness of the sample was unknown. After taking angle measurements at a wavelength close to the wavelength used to determine the known constant, we calculated our sample thickness to be 1.5 mm.

Unfortunately, without a set of calipers, we had no way of verifying the thickness of the sample. However, 1.5 mm seems reasonable by just looking at the sample.

Our final sample was light flint. We took measurements at four different wavelengths and observed the Verdet constant decreases as wavelength increases and vice-versa.

We were given a table of indices for light flint. After plotting these values versus wavelength, we used a fit to determine Cauchy coefficients. Using the "B" coefficient, we were able to solve for $\frac{dn}{d\lambda}$. Plugging that value into our equation, we were able to calculate a Verdet constant of $0.0545\frac{m.p.a.}{cm*gauss}$.

The known Verdet constant of light flint is $0.0317\frac{m.p.a.}{cm*gauss}$. Comparing our value to the known value, we were off slightly. It's likely that we made mistakes when taking our measurements.

There were several possible sources of error in this experiment. Without knowing the exact wavelength of the light emitted by the LED lights we were using, there's no way to verify our

Verdet constants with the known values.

All of our measurements were subjective. Not to mention, after taking measurements for long periods of time, our eyes became tired/strained. As a result, it's very possible our measurements were not exact.

The dial we used to adjust the polarizer was very sensitive. We had to estimate the angle measurement to $\frac{1}{10}$ of a degree. It's likely our measurements were off by up to $0.5°$.

References

[1] Rutgers: The Faraday Effect,
http://www.physics.rutgers.edu/ eandrei/389/faraday.pdf

[2] Brown: Faraday Effect Attachment,
http://tinyurl.com/mrelcdz

[3] Northrop: Terbium Gallium Garnet - TGG,
http://tinyurl.com/kbxb2kv

[4] Wikipedia: The Faraday Effect,
http://tinyurl.com/lhn3qra

[5] Wikipedia: Dielectric,
https://en.wikipedia.org/wiki/Dielectric

[6] Wikipedia: Birefringence,
https://en.wikipedia.org/wiki/Birefringence

[7] Wikipedia: Visible Spectrum,
http://tinyurl.com/q8yqea9

A
Visual Basic Program 1

Function: $y = mx + b$

```
Private Sub CommandButton1_Click()

Unload GridSear

End Sub


Private Sub EndRow_Change()

End Sub

Private Sub NonLinFt_Click()

Dim chiSqr, c2PerDof As Single

iBeg = BegRow.Text + 0 ' Set iBeg to the value in "BegRow" TextBox
iEnd = EndRow.Text + 0 ' Set iEnd to the value in "EndRow" TextBox

j = 0
nPts = (iEnd - iBeg) + 1
ReDim x(0 To nPts) ' Set the dimension of the x array (# points)
ReDim y(0 To nPts) ' Set the dimension of the y array (# points)
ReDim sigY(0 To nPts) ' Set the dimension of the error bar array (# points)
ReDim yCalc(0 To nPts) ' Set the dimension of the fit function array (# points)
For i = iBeg To iEnd ' Loop from the 1st row to the last row of data
  j = j + 1
  x(j) = ActiveSheet.Cells(i, 1) ' Place data in X-column into x array
  y(j) = ActiveSheet.Cells(i, 2) ' Place data in Y-column into y array
  sigY(j) = ActiveSheet.Cells(i, 3) ' Place data in error bar column into sigY array
Next i

stepSize = ActiveSheet.Cells(1, 9) ' scale size of grid cell in grid search
chiCut = ActiveSheet.Cells(1, 11) ' minimum change in Chi-Squared to stop searching for minimum
NParms = ActiveSheet.Cells(1, 7) ' Number of parameters
nFree = nPts - NParms ' Number of degrees of freedom

ReDim a(0 To NParms) ' Set the dimension of the parameter array (# parameters)
ReDim deltaA(0 To NParms) ' Set the dimension of the grid spacing array (# parameters)
ReDim sigA(0 To NParms) ' Set the dimension of the uncertainty array (# parameters)
For i = 1 To NParms
  a(i) = ActiveSheet.Cells(3 + i, 7)
  deltaA(i) = stepSize * Abs(a(i)) ' Implies parm(i) must not be 0
Next i

trial = 0
chiSqr = CalcChiSq()
chiOld = chiSqr + chiCut + 1#

i = 0
Do While (Abs(chiOld - chiSqr) >= chiCut)
  chiOld = chiSqr
  i = i + 1
  ActiveSheet.Cells(11 + i, 6) = trial
  ActiveSheet.Cells(11 + i, 7) = chiSqr
  If (i = 20) Then ' only place 20 values on the spreadsheet at any time
    i = 0
  End If

  Call Gridls(chiSqr)

  trial = trial + 1
Loop

Call CalculateY

For j = 1 To NParms
  sigA(j) = SigParab(j) ' dChi2 = 1
Next j

c2PerDof = chiSqr / nFree
ActiveSheet.Cells(12, 9) = chiSqr
ActiveSheet.Cells(12, 10) = c2PerDof
ActiveSheet.Cells(12, 11) = nFree
```

```
74
75  For k = 1 To NParms
76    ActiveSheet.Cells(k + 3, 9) = a(k)
77    ActiveSheet.Cells(k + 3, 11) = sigA(k)
78  Next k
79
80  For n = 1 To nPts
81    ActiveSheet.Cells(n + 1, 4) = yCalc(n)
82  Next n
83
84  End Sub
85
86  Private Sub Open_Filename_Click()
87
88  Dim rowItems As Variant ' Declaring "rowItems" as a dynamic array
89  Dim Filename As String ' Delcaring "Filename" as a string
90  Dim rowOfdata As String ' Declaring "rowOfdata" as a string
91  Dim temp As String
92
93  Filename = Application.GetOpenFilename() ' Prompt the user to browse for the data file
94
95  Open Filename For Input As #1 ' #1 = first opened Ascii data file (we can open multiple files)
96      For k = 1 To 2 ' The 1st 2 lines are to be ignored (they are header files)
97        Line Input #1, rowOfdata ' Read each line in "10CM Data.txt" file
98      Next k ' Loop through the 1st 2 lines of the data file
99
100   iRow = 1
101   Do Until EOF(1) ' Now it is time to read the actual data -- the program will read data until it
102                   ' reaches the last line in the file: "end of file" = EOF
103     Line Input #1, rowOfdata ' Read all data on a specific line (or row)
104     Do
105       temp = rowOfdata
106       rowOfdata = Replace(rowOfdata, "␣␣", "␣") 'remove multiple white spaces
107     Loop Until temp = rowOfdata
108     rowItems = Split(Trim(rowOfdata), "␣") ' Parse data separated by tabs and put it into rowItems array
109     iCol = 1
110     For i = LBound(rowItems) To UBound(rowItems) ' LBound = lowest index of array, UBound = highest index
111       ActiveSheet.Cells(iRow + 1, iCol) = rowItems(i) ' Put data from rowItems into Excel spreadsheet
112       iCol = iCol + 1 ' There are 4 columns of data in "10CM Data.txt" file
113     Next i ' Loop to read the next column of data of row "iRow"
114     iRow = iRow + 1 ' Prepare for reading the next row of data
115   Loop ' Loop to read the next row of data
116 Close #1 ' When EOF has been reached, properly close the data file
117
118 DataFile.Caption = Filename ' Put the name of the data file into the "DataFile" Caption
119 DataFile.TextAlign = fmTextAlignCenter ' Align Text in the center of the TextBox
120 DataFile.ForeColor = RGB(0, 0, 255) ' Blue text
121 DataFile.BackColor = RGB(255, 255, 0) ' Yellow background
122
123 End Sub
124
125 Function CalcChiSq() As Single ' calculates the Chi-Squared value
126
127 Dim chi2 As Single
128
129 chi2 = 0#
130 For i = 1 To nPts
131   chi2 = chi2 + ((y(i) - yFunction(x(i))) / sigY(i)) ^ 2
132 Next i
133
134 CalcChiSq = chi2 ' the value of CalChiSq is what is returned by the Function call
135
136 End Function
137
138 Function SigParab(j) As Single ' determines parameter fit value and its uncertainty
139
140 chiSq2 = CalcChiSq()
141 a(j) = a(j) + deltaA(j)
142 chiSq3 = CalcChiSq()
143 a(j) = a(j) - 2# * deltaA(j)
144 chiSq1 = CalcChiSq()
145 a(j) = a(j) + deltaA(j)
146
147 SigParab = deltaA(j) * Sqr(2# / (chiSq1 - 2# * chiSq2 + chiSq3))
148
149 End Function
150
151 Function ExpF(a, x) As Single ' Dealing with large arguments for exponentials
152
153 Dim yy, arg As Single
154
```

```
arg = Abs(x / a)
If (arg > 60) Then
  yy = 0#
Else
  yy = Exp(-arg)
End If

ExpF = yy

End Function

Function yFunction(xx) As Single ' Calculates the fitting function

'yFunction = a(1) + a(2) * ExpF(a(4), xx) + a(3) * ExpF(a(5), xx)
yFunction = a(1) + a(2) * xx

End Function

Sub CalculateY()

For i = 1 To nPts
  yCalc(i) = yFunction(x(i))
Next i

End Sub

Sub Gridls(chiSqr)

Dim Save, Delta, delta1, del1, del2, aa, bb, cc, Disc, alph, x1, x2 As Single
     chiSq2 = CalcChiSq()
     For j = 1 To NParms
       Delta = deltaA(j)
       a(j) = a(j) + Delta
       chiSq3 = CalcChiSq()

       If (chiSq3 > chiSq2) Then
          Delta = -Delta ' started in wrong direction
          a(j) = a(j) + Delta
          Save = chiSq2 ' interchange 2 and 3 so 3 is lower
          chiSq2 = chiSq3
          chiSq3 = Save
       End If

110
       chiSq1 = chiSq2 ' move back to prepare for quad fit
       chiSq2 = chiSq3
       a(j) = a(j) + Delta
       chiSq3 = CalcChiSq()

       If (chiSq3 <= chiSq2) Then GoTo 110

       del1 = chiSq2 - chiSq1
       del2 = chiSq3 - 2 * chiSq2 + chiSq1
       delta1 = Delta * (del1 / del2 + 1.5)
       a(j) = a(j) - delta1
       chiSq2 = CalcChiSq() ' at new local minimum

       aa = del2 / 2 ' chiSq = aa*a(j)**2 + bb*a(j) + cc
       bb = del1 - del2 / 2
       cc = chiSq1 - chiSq2
       Disc = bb ^ 2 - 4 * aa * (cc - 2) ' chiSqr diff(erence) = 2

       If (Disc > 0) Then ' if not, then probably not parabolic yet
          Disc = (Disc) ^ (0.5)
          alph = (-bb - Disc) / (2 * aa)
          x1 = alph * Delta + a(1) - 2 * Delta ' a(j) at chiSq minimum+2
          Disc = bb ^ 2 - 4 * aa * cc

          If (Disc > 0) Then
             Disc = (Disc) ^ (0.5)
          Else
             Disc = 0 ' elim rounding err
          End If

          alph = (-bb - Disc) / (2 * aa)
          x2 = alph * Delta + a(1) - 2 * Delta ' a(j) at chiSq minimum
          Delta = x1 - x2
          deltaA(j) = Delta
       End If
     Next j

```

```
      chiSqr = chiSq2

End Sub



Private Sub PlotGraph_Click()

Dim XPoints() As Variant ' Declare "XPoints()" as a dynamic array
Dim YPoints() As Variant ' Declare "YPoints()" as a dynamic array
Dim PlotData As Chart ' Declare PlotData as a Chart Object
Dim ErrorY() As Variant
Dim CalcY() As Variant

Set PlotData = ActiveSheet.Shapes.AddChart(xlXYScatter).Chart ' Setup a Scatter Plot

iBeg = BegRow.Text + 0 ' Set iBeg to the value in "BegRow" TextBox
iEnd = EndRow.Text + 0 ' Set iEnd to the value in "EndRow" TextBox

j = -1
ReDim XPoints(0 To iEnd - iBeg) ' Set the dimension of the XPoint array (# points)
ReDim YPoints(0 To iEnd - iBen) ' Set the dimension of the YPoint array (# points)
ReDim ErrorY(0 To iEnd - iBeg)
ReDim CalcY(0 To iEnd - iBeg)
For i = iBeg To iEnd ' Loop from the 1st row to the last row of data
  j = j + 1
  XPoints(j) = ActiveSheet.Cells(i, 1) ' Place data in X-column into XPoints array
  YPoints(j) = ActiveSheet.Cells(i, 2) ' Place data in Y-column into Ypoints array
  ErrorY(j) = ActiveSheet.Cells(i, 3)
  CalcY(j) = ActiveSheet.Cells(i, 4)
Next i
           ' Now we Plot the xy-data using the following functions
PlotData.SeriesCollection.NewSeries ' Start a new plot for curve #1
PlotData.SeriesCollection(1).Values = YPoints ' YPoints are the y-values
PlotData.SeriesCollection(1).XValues = XPoints ' XPoints are the x-values
PlotData.SeriesCollection(1).ErrorBar Direction:=xlY, Include:=xlBoth, Type:=xlCustom, Amount:=ErrorY, MinusValues:=ErrorY
PlotData.SeriesCollection.NewSeries ' Start a new plot for curve #1
PlotData.SeriesCollection(2).Values = CalcY ' YPoints are the y-values
PlotData.SeriesCollection(2).XValues = XPoints ' XPoints are the x-values
PlotData.SeriesCollection(2).ChartType = xlXYScatterSmoothNoMarkers
PlotData.Axes(xlCategory, xlPrimary).HasTitle = True ' x-axis label
PlotData.Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = ActiveSheet.Cells(1, 1)
PlotData.Axes(xlValue, xlPrimary).HasTitle = True ' y-axis label
PlotData.Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = ActiveSheet.Cells(1, 2)

End Sub
```

B
Visual Basic Program 2

Function: $n = A + \frac{B}{\lambda^2}$

```

Private Sub CommandButton1_Click()

Unload GridSear

End Sub



Private Sub EndRow_Change()

End Sub

Private Sub NonLinFt_Click()

Dim chiSqr, c2PerDof As Single

iBeg = BegRow.Text + 0 ' Set iBeg to the value in "BegRow" TextBox
iEnd = EndRow.Text + 0 ' Set iEnd to the value in "EndRow" TextBox

j = 0
nPts = (iEnd - iBeg) + 1
ReDim x(0 To nPts) ' Set the dimension of the x array (# points)
```

```
ReDim y(0 To nPts) ' Set the dimension of the y array (# points)
ReDim sigY(0 To nPts) ' Set the dimension of the error bar array (# points)
ReDim yCalc(0 To nPts) ' Set the dimension of the fit function array (# points)
For i = iBeg To iEnd ' Loop from the 1st row to the last row of data
  j = j + 1
  x(j) = ActiveSheet.Cells(i, 1) ' Place data in X-column into x array
  y(j) = ActiveSheet.Cells(i, 2) ' Place data in Y-column into y array
  sigY(j) = ActiveSheet.Cells(i, 3) ' Place data in error bar column into sigY array
Next i

stepSize = ActiveSheet.Cells(1, 9) ' scale size of grid cell in grid search
chiCut = ActiveSheet.Cells(1, 11) ' minimum change in Chi-Squared to stop searching for minimum
NParms = ActiveSheet.Cells(1, 7) ' Number of parameters
nFree = nPts - NParms ' Number of degrees of freedom

ReDim a(0 To NParms) ' Set the dimension of the parameter array (# parameters)
ReDim deltaA(0 To NParms) ' Set the dimension of the grid spacing array (# parameters)
ReDim sigA(0 To NParms) ' Set the dimension of the uncertainty array (# parameters)
For i = 1 To NParms
  a(i) = ActiveSheet.Cells(3 + i, 7)
  deltaA(i) = stepSize * Abs(a(i)) ' Implies parm(i) must not be 0
Next i

trial = 0
chiSqr = CalcChiSq()
chiOld = chiSqr + chiCut + 1#

i = 0
Do While (Abs(chiOld - chiSqr) >= chiCut)
  chiOld = chiSqr
  i = i + 1
  ActiveSheet.Cells(11 + i, 6) = trial
  ActiveSheet.Cells(11 + i, 7) = chiSqr
  If (i = 20) Then ' only place 20 values on the spreadsheet at any time
    i = 0
  End If

  Call Gridls(chiSqr)

  trial = trial + 1
Loop

Call CalculateY

For j = 1 To NParms
  sigA(j) = SigParab(j) ' dChi2 = 1
Next j

c2PerDof = chiSqr / nFree
ActiveSheet.Cells(12, 9) = chiSqr
ActiveSheet.Cells(12, 10) = c2PerDof
ActiveSheet.Cells(12, 11) = nFree

For k = 1 To NParms
  ActiveSheet.Cells(k + 3, 9) = a(k)
  ActiveSheet.Cells(k + 3, 11) = sigA(k)
Next k

For n = 1 To nPts
  ActiveSheet.Cells(n + 1, 4) = yCalc(n)
Next n

End Sub

Private Sub Open_Filename_Click()

Dim rowItems As Variant ' Declaring "rowItems" as a dynamic array
Dim Filename As String ' Delcaring "Filename" as a string
Dim rowOfdata As String ' Declaring "rowOfdata" as a string
Dim temp As String

Filename = Application.GetOpenFilename() ' Prompt the user to browse for the data file

Open Filename For Input As #1 ' #1 = first opened Ascii data file (we can open multiple files)
   For k = 1 To 2 ' The 1st 2 lines are to be ignored (they are header files)
     Line Input #1, rowOfdata ' Read each line in "10CM Data.txt" file
   Next k ' Loop through the 1st 2 lines of the data file

  iRow = 1
  Do Until EOF(1) ' Now it is time to read the actual data -- the program will read data until it
                  ' reaches the last line in the file: "end of file" = EOF
```

```
    Line Input #1, rowOfdata ' Read all data on a specific line (or row)
    Do
      temp = rowOfdata
      rowOfdata = Replace(rowOfdata, "␣␣", "␣") 'remove multiple white spaces
    Loop Until temp = rowOfdata
    rowItems = Split(Trim(rowOfdata), "␣") ' Parse data separated by tabs and put it into rowItems array
    iCol = 1
    For i = LBound(rowItems) To UBound(rowItems) ' LBound = lowest index of array, UBound = highest index
      ActiveSheet.Cells(iRow + 1, iCol) = rowItems(i) ' Put data from rowItems into Excel spreadsheet
      iCol = iCol + 1 ' There are 4 columns of data in "10CM Data.txt" file
    Next i ' Loop to read the next column of data of row "iRow"
    iRow = iRow + 1 ' Prepare for reading the next row of data
  Loop ' Loop to read the next row of data
Close #1 ' When EOF has been reached, properly close the data file

DataFile.Caption = Filename ' Put the name of the data file into the "DataFile" Caption
DataFile.TextAlign = fmTextAlignCenter ' Align Text in the center of the TextBox
DataFile.ForeColor = RGB(0, 0, 255) ' Blue text
DataFile.BackColor = RGB(255, 255, 0) ' Yellow background

End Sub

Function CalcChiSq() As Single ' calculates the Chi-Squared value

Dim chi2 As Single

chi2 = 0#
For i = 1 To nPts
  chi2 = chi2 + ((y(i) - yFunction(x(i))) / sigY(i)) ^ 2
Next i

CalcChiSq = chi2 ' the value of CalChiSq is what is returned by the Function call

End Function

Function SigParab(j) As Single ' determines parameter fit value and its uncertainty

chiSq2 = CalcChiSq()
a(j) = a(j) + deltaA(j)
chiSq3 = CalcChiSq()
a(j) = a(j) - 2# * deltaA(j)
chiSq1 = CalcChiSq()
a(j) = a(j) + deltaA(j)

SigParab = deltaA(j) * Sqr(2# / (chiSq1 - 2# * chiSq2 + chiSq3))

End Function

Function ExpF(a, x) As Single ' Dealing with large arguments for exponentials

Dim yy, arg As Single

arg = Abs(x / a)
If (arg > 60) Then
  yy = 0#
Else
  yy = Exp(-arg)
End If

ExpF = yy

End Function

Function yFunction(xx) As Single ' Calculates the fitting function

'yFunction = a(1) + a(2) * ExpF(a(4), xx) + a(3) * ExpF(a(5), xx)
yFunction = a(1) + a(2) / (xx) ^ 2

End Function

Sub CalculateY()

For i = 1 To nPts
  yCalc(i) = yFunction(x(i))
Next i

End Sub

Sub Gridls(chiSqr)

Dim Save, Delta, delta1, del1, del2, aa, bb, cc, Disc, alph, x1, x2 As Single
```

```
      chiSq2 = CalcChiSq()
      For j = 1 To NParms
        Delta = deltaA(j)
        a(j) = a(j) + Delta
        chiSq3 = CalcChiSq()

        If (chiSq3 > chiSq2) Then
           Delta = -Delta ' started in wrong direction
           a(j) = a(j) + Delta
           Save = chiSq2 ' interchange 2 and 3 so 3 is lower
           chiSq2 = chiSq3
           chiSq3 = Save
        End If

110
        chiSq1 = chiSq2 ' move back to prepare for quad fit
        chiSq2 = chiSq3
        a(j) = a(j) + Delta
        chiSq3 = CalcChiSq()

        If (chiSq3 <= chiSq2) Then GoTo 110

        del1 = chiSq2 - chiSq1
        del2 = chiSq3 - 2 * chiSq2 + chiSq1
        delta1 = Delta * (del1 / del2 + 1.5)
        a(j) = a(j) - delta1
        chiSq2 = CalcChiSq() ' at new local minimum

        aa = del2 / 2 ' chiSq = aa*a(j)**2 + bb*a(j) + cc
        bb = del1 - del2 / 2
        cc = chiSq1 - chiSq2
        Disc = bb ^ 2 - 4 * aa * (cc - 2) ' chiSqr diff(erence) = 2

        If (Disc > 0) Then ' if not, then probably not parabolic yet
           Disc = (Disc) ^ (0.5)
           alph = (-bb - Disc) / (2 * aa)
           x1 = alph * Delta + a(1) - 2 * Delta ' a(j) at chiSq minimum+2
           Disc = bb ^ 2 - 4 * aa * cc

           If (Disc > 0) Then
              Disc = (Disc) ^ (0.5)
           Else
              Disc = 0 ' elim rounding err
           End If

           alph = (-bb - Disc) / (2 * aa)
           x2 = alph * Delta + a(1) - 2 * Delta ' a(j) at chiSq minimum
           Delta = x1 - x2
           deltaA(j) = Delta
        End If
      Next j

      chiSqr = chiSq2

End Sub




Private Sub PlotGraph_Click()

Dim XPoints() As Variant ' Declare "XPoints()" as a dynamic array
Dim YPoints() As Variant ' Declare "YPoints()" as a dynamic array
Dim PlotData As Chart ' Declare PlotData as a Chart Object
Dim ErrorY() As Variant
Dim CalcY() As Variant

Set PlotData = ActiveSheet.Shapes.AddChart(xlXYScatter).Chart ' Setup a Scatter Plot

iBeg = BegRow.Text + 0 ' Set iBeg to the value in "BegRow" TextBox
iEnd = EndRow.Text + 0 ' Set iEnd to the value in "EndRow" TextBox

j = -1
ReDim XPoints(0 To iEnd - iBeg) ' Set the dimension of the XPoint array (# points)
ReDim YPoints(0 To iEnd - iBen) ' Set the dimension of the YPoint array (# points)
ReDim ErrorY(0 To iEnd - iBeg)
ReDim CalcY(0 To iEnd - iBeg)
For i = iBeg To iEnd ' Loop from the 1st row to the last row of data
  j = j + 1
  XPoints(j) = ActiveSheet.Cells(i, 1) ' Place data in X-column into XPoints array
  YPoints(j) = ActiveSheet.Cells(i, 2) ' Place data in Y-column into Ypoints array
  ErrorY(j) = ActiveSheet.Cells(i, 3)
```

```
  CalcY(j) = ActiveSheet.Cells(i, 4)
Next i
           ' Now we Plot the xy-data using the following functions
PlotData.SeriesCollection.NewSeries ' Start a new plot for curve #1
PlotData.SeriesCollection(1).Values = YPoints ' YPoints are the y-values
PlotData.SeriesCollection(1).XValues = XPoints ' XPoints are the x-values
PlotData.SeriesCollection(1).ErrorBar Direction:=xlY, Include:=xlBoth, Type:=xlCustom, Amount:=ErrorY, MinusValues:=ErrorY
PlotData.SeriesCollection.NewSeries ' Start a new plot for curve #1
PlotData.SeriesCollection(2).Values = CalcY ' YPoints are the y-values
PlotData.SeriesCollection(2).XValues = XPoints ' XPoints are the x-values
PlotData.SeriesCollection(2).ChartType = xlXYScatterSmoothNoMarkers
PlotData.Axes(xlCategory, xlPrimary).HasTitle = True ' x-axis label
PlotData.Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = ActiveSheet.Cells(1, 1)
PlotData.Axes(xlValue, xlPrimary).HasTitle = True ' y-axis label
PlotData.Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = ActiveSheet.Cells(1, 2)

End Sub
```