X-Ray Scattering Experiment

Zekeriya Gökhan Evelek
Boğaziçi University • Physics Department

***Abstract*—In this experiment, we have created X-rays by using molybdenum elements as target and then make them scatter from the NaCl crystal. We have measured the intensities of X-rays at various angles for different initial energies. We have found the Planck Constan was found out to be** $8.10 \times 10^{-34} JHz^{-1} \pm 2.85 \times 10^{-34} JHz^{-1}$ **which is 0.52 σ's away from the CODATA's recommended value.**

I. Theory

There were many experiments going on in which X-rays were produced unknowingly before the discovery of them. The first one was estimated to be that of William Morgan's in 1785. He has sent a paper to Royal Society of London describing the effects of the current passing through a partially evacuated tube, during which X-rays were being produced. What was common to the experiments that has given rise to the production of those light waves is that the scientists investigated the cathode rays. They consist of energetic electron beams, which produces X-rays. Furthermore, Helmoltz has formulated mathematical description for X-rays before the discovery even though he did not work with them in lab.[1]

German physicist Wilhelm Röntgen noticed some unknown light coming out of the Crookes tube which is covered by black clapboard in order to prevent the interference with the visible light. He has studied the characteristics of the unknown rays systematically for two months. He has noticed that they are capable of passing through many materials such as books and papers. On 28 December 1895 he has submitted his findings in his paper called "On a new kind of ray: A preliminary communication", which is the first paper written on X-rays. He gave the name X-rays because he did not know about what was causing the glowing on the screen so he called X meaning the unknown ray.[1]

X-rays can be produced in many ways. Applying high voltages to electrons to make them move in the high velocities is one of them. After those electrons hit some target metal, due to the high energies of the incoming electrons, the electrons in the target metal will be freed from the atom. The unoccupied lowest levels will be filled by the electrons in the outer shells. During this filling process, X-rays are produced.[1] The target metal may differ, in this experiment we have used molybdenum.

The maximum energy that X-rays have is limited by the amount of the initial energy applied to the electrons.

$$\nu_{\text{max}} = \frac{eV}{h} \tag{1}$$

[2] This is called Duane-Hunt Law. Or, equally the formula can be written as:

$$\lambda_{\text{min}} = \frac{hc}{eV} \tag{2}$$

Therefore, provided that we know the velocity of the light c, the charge of the electron e, the applied voltage V, one can determine the Planck constant by measuring the λ_{min}.

The produced X-rays scatter after hitting a crystal. In this experiment we have used NaCl, having lattice constant $d = 5.64$Å[3] for this purpose. The scattering occurs according to the Bragg's law:

$$n\lambda = 2d \sin\theta [4] \tag{3}$$

here d is the lattice constant,θ is the angle between the incident wave and the crystal, n is the order of diffraction and λ is the wavelength of the coming wave.(See Fig.1)

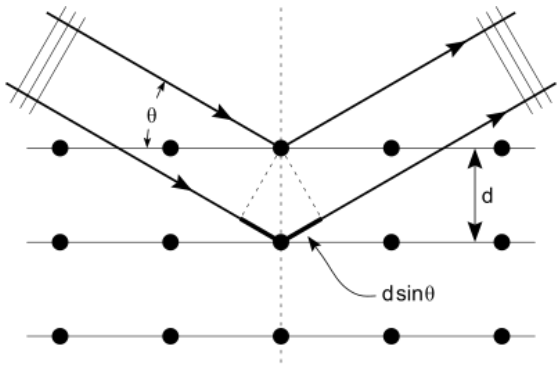

Fig. 1. Bragg diffraction [4]

II. Method

Energetic electrons are used to create X-rays from the target material(Molbdenum-42). Those created X-rays are then scattered from the crystalline solids, in this experiment NaCl. The Geiger-Muller counter was used to determine the intensity of the rays in different angles.

1) Voltage is set to 35kV
2) High voltage is applied to electrons, created by heating of the filament
3) Electrons moving at high velocities targeted at Molybdenum elements.
4) X-rays scattered from NaCl crsytal.
5) Gieger-Muller counter is used to detect X-rays starting from 3.0 degrees to 45.0 degrees, with 0.1 increase of degree at each 5 seconds to get calibration data
6) The same data-taking process is done for 15,18,21,24,27 and 30 kV applied and the counter moves from 3.0 degrees to something around 11.0 and 15.0 degrees, with the same increasing rate.

III. The Experimental Setup

Here is the full list of apparatus we have used in the experiment below:

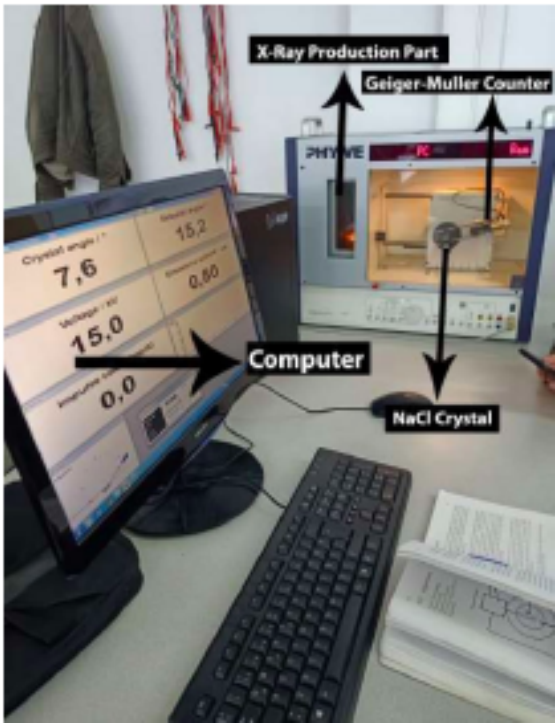

Fig. 2. The Experimental Setup

- PHYWE X-ray Unit (35kV)
- Computer
- Molybdenum(42) element as the target material
- NaCl crystal for scattering
- Geiger Muller counter(using as an X-ray detector)

IV. The Data

Due to large amount of data we have taken, namely 420 pairs for calibration and around 100 pairs for the second part, I have illustrated only the first and the last 5 datapoints at the table, just to show.

We have taken the least significant order as the error in angles and the error for count per second is calculated as $\sqrt{\frac{\text{value}}{5}}$. Here the division with 5 is due to the measurements of in the intervals 5 seconds. The same errors have been used in the analysis as well.

The calibration data will look like this

TABLE I
CALIBRATION DATA AT 35KV

Angle(degrees)	Count per Second
3.0 ± 0.1	885.0 ± 13.31
3.1 ± 0.1	745.0 ± 12.20
3.2 ± 0.1	480.0 ± 9.80
3.3 ± 0.1	346.0 ± 8.32
3.4 ± 0.1	247.0 ± 7.03
.	.
.	.
44.6 ± 0.1	9.0 ± 1.34
44.7 ± 0.1	8.0 ± 1.26
44.8 ± 0.1	9.0 ± 1.34
44.9 ± 0.1	10.0 ± 1.41
45.0 ± 0.1	8.0 ± 1.26

15kV data look like this

TABLE II
MEASUREMENTS AT 15KV

Angle(degrees)	Count per Second
3.0 ± 0.1	30.0 ± 2.45
3.1 ± 0.1	25.0 ± 2.24
3.2 ± 0.1	19.0 ± 1.95
3.3 ± 0.1	14.0 ± 1.67
3.4 ± 0.1	8.0 ± 1.26
.	.
.	.
14.5 ± 0.1	20.0 ± 2.00
14.6 ± 0.1	20.0 ± 2.00
14.7 ± 0.1	14.0 ± 1.67
14.8 ± 0.1	16.0 ± 1.79
14.9 ± 0.1	16.0 ± 1.79
15.0 ± 0.1	18.0 ± 1.90

For the illustration of the dataset for the other voltages, see appendix.

V. The Analysis

In the analysis, we have used the following error propagation formula if there is no correlation between the variables:[5]

$$\sigma_f = \sqrt{(\frac{\partial f}{\partial x})^2(\sigma_x)^2 + (\frac{\partial f}{\partial y})^2(\sigma_y)^2 + (\frac{\partial f}{\partial z})^2(\sigma_z)^2 + \ldots} \quad (4)$$

We have used ROOT's built-in function for getting plots and their results.

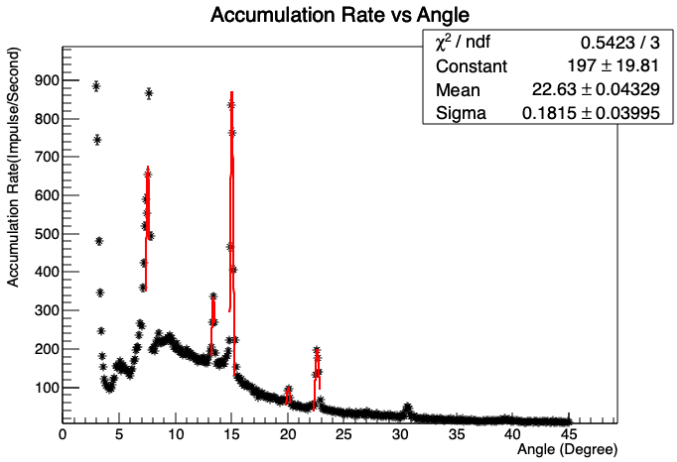

Fig. 3. Calibration measurements at 35kV

We have use gaussian fit for each peak, except the first one because K_α and K_β values are close to each other. We took the mean values of the each gaussian fit as the peak value. Here is the table showing the experimental theta values corresponding to their nominal values:

TABLE III
NOMINAL VS EXPERIMENTAL THETA VALUES

Theoretical Theta	Experimental Theta
7.76	7.5607 ± 0.1557
13.93	13.3917 ± 0.1713
15.69	15.0393 ± 0.1273
19.73	20.0618 ± 0.1835
23.90	22.6346 ± 0.1815

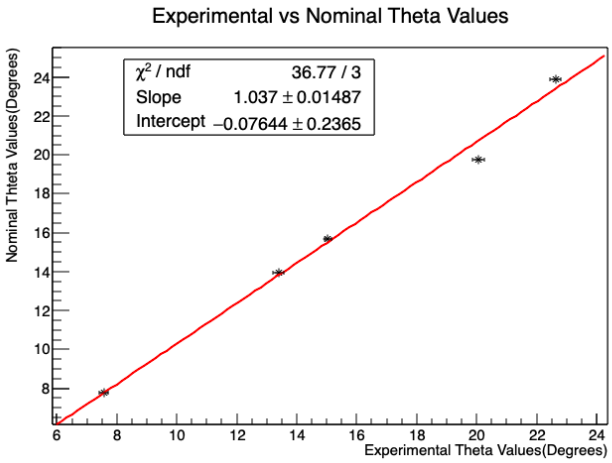

Fig. 4. Experimental vs Nominal Values

By applying linear fit to our data, we get calibration formula. For each dataset starting from 15kV we have corrected the theta values by using the following formula:

$$1.037 \times \theta - 0.076 \tag{5}$$

Then, by using Brag's law (equation 3), we have converted our experimental theta values to the wavelengths . Finally, we have applied a linear fit for the first 8-10 data points to get ν_{min}. We have calculated it by taking the linear fits' intercept with the x-axis.

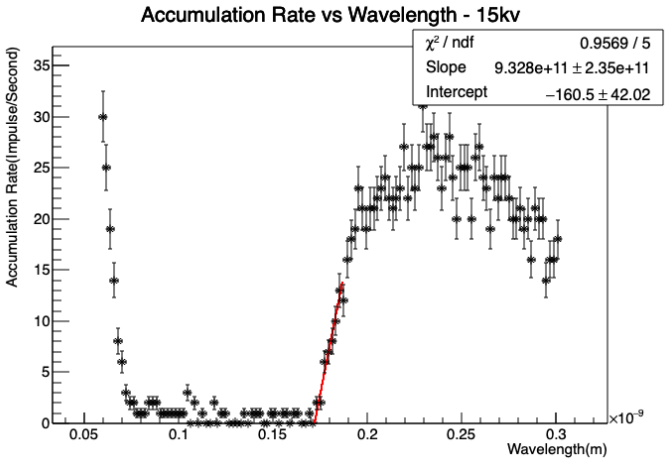

Fig. 5. Wavelength vs Accumulation Rate at 15kV

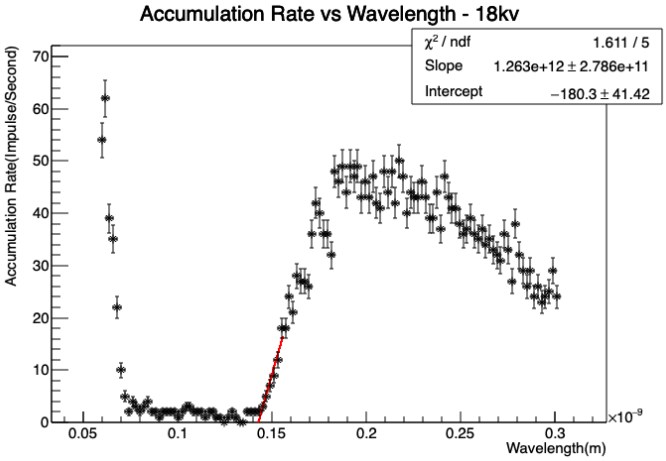

Fig. 6. Wavelength vs Accumulation Rate at 18kV

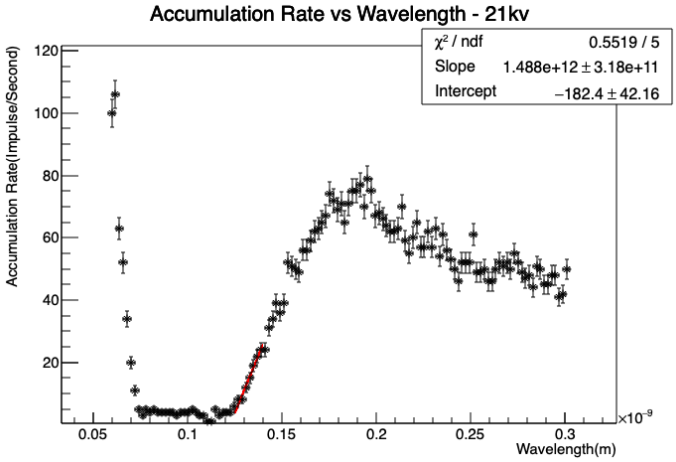

Fig. 7. Wavelength vs Accumulation Rate at 21kV

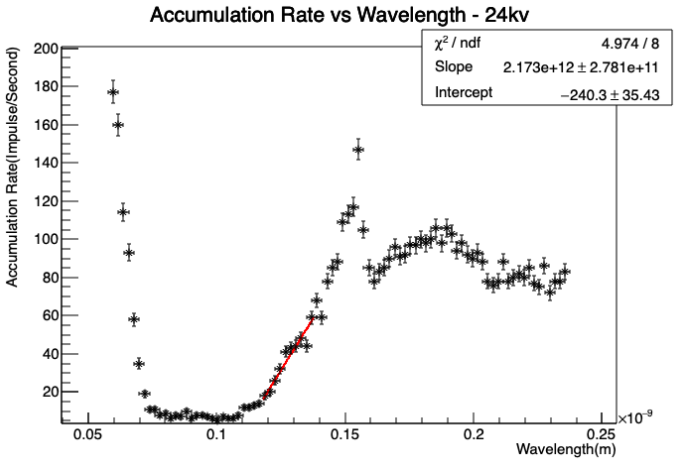

Fig. 8. Wavelength vs Accumulation Rate at 24kV

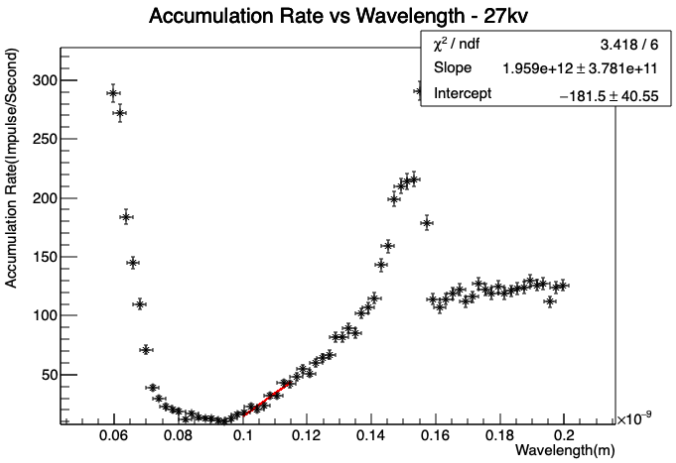

Fig. 9. Wavelength vs Accumulation Rate at 27kV

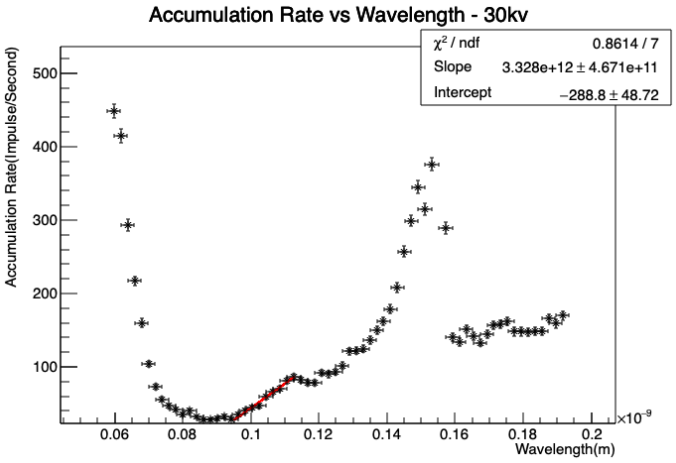

Fig. 10. Wavelength vs Accumulation Rate at 30kV

Here are the data-set with each ν_{max} and its corresponding λmax calculated with the help of the equations 1 and 2.

TABLE IV
NOMINAL VS EXPERIMENTAL THETA VALUES

Applied Voltage (kV)	$\mu_{\min}$ $(\times 10^{-10})$	$f_{\max}$ $(\times 10^{18})$
15	1.721 ± 0.625	1.742 ± 0.633
18	1.428 ± 0.455	2.099 ± 0.668
21	1.226 ± 0.386	2.446 ± 0.770
24	1.106 ± 0.216	2.710 ± 0.529
27	0.926 ± 0.273	3.236 ± 0.955
30	0.868 ± 0.190	3.455 ± 0.758

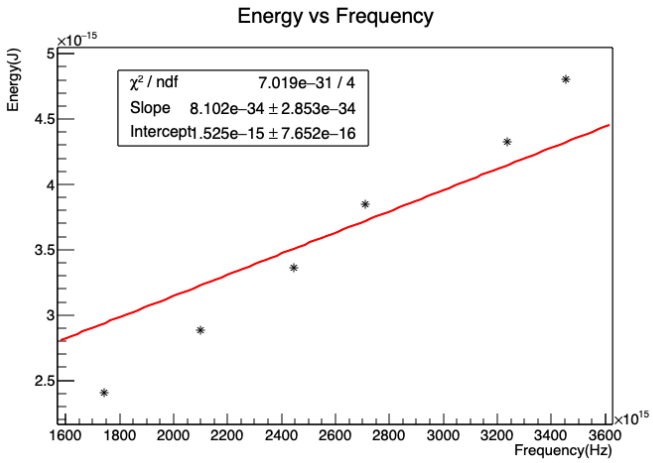

Fig. 11. Energy vs Frequency

VI. THE RESULT

The Planck Constant h is found out to be

$$8.10 \times 10^{-34} JHz^{-1} \pm 2.85 \times 10^{-34} JHz^{-1} \quad (6)$$

the CODATA recommended value is $6.62 \times 10^{-34} JHz^{-1}$[6], which is 0.52 σ's away from our experimental result.

VII. THE CONCLUSION

We can say that our experiment is a success due to the our result being 0.52 σ's away from the CODATA's recommended value. We have good approximations to the theoretical theta values as well, they differ from each other only slightly. What is more, thanks to the fact that we only need the initial rise on the accumulation rate vs wavelength graphs, we did not take the data up to 15.0 degrees. The possible sources for error might be the surface of the NaCl crystal. Also, we did not include the errors for the voltages at the last fit that consists of energies and wavelengths. In short, we got good outcomes.

REFERENCES

[1] *wiki*. URL: https://en.wikipedia.org/wiki/X-ray (visited on 04/12/2024).

[2] *law*. URL: https://en.wikipedia.org/wiki/Duane%E2%80%93Hunt_law (visited on 04/12/2024).

[3] *lattice*. URL: https://en.wikipedia.org/wiki/Lattice_constant (visited on 04/12/2024).

[4] *brag*. URL: https://en.wikipedia.org/wiki/Bragg%27s_law (visited on 04/12/2024).

[5] E. Gülmez. *Advanced Physics Experiments*. 1st. Boğaziçi University Publications, 1999.

[6] *codata*. URL: https://physics.nist.gov/cgi-bin/cuu/Value?h (visited on 04/12/2024).

VIII. APPENDIX

The illlustration of the dataset: 18kV data look like this

TABLE V
MEASUREMENTS AT 18KV

Angle(degrees)	Count per Second
3.0 ± 0.1	54.0 ± 3.28
3.1 ± 0.1	62.0 ± 3.52
3.2 ± 0.1	39.0 ± 2.79
3.3 ± 0.1	35.0 ± 2.65
3.4 ± 0.1	22.0 ± 2.10
.	.
.	.
14.5 ± 0.1	26.0 ± 2.28
14.6 ± 0.1	23.0 ± 2.14
14.7 ± 0.1	24.0 ± 2.19
14.8 ± 0.1	25.0 ± 2.24
14.9 ± 0.1	29.0 ± 2.41
15.0 ± 0.1	24.0 ± 2.19

21kV data look like this

TABLE VI
MEASUREMENTS AT 21KV

Angle(degrees)	Count per Second
3.0 ± 0.1	100.0 ± 4.47
3.1 ± 0.1	106.0 ± 4.60
3.2 ± 0.1	63.0 ± 3.55
3.3 ± 0.1	52.0 ± 3.23
3.4 ± 0.1	34.0 ± 2.61
.	.
.	.
14.5 ± 0.1	45.0 ± 3.00
14.6 ± 0.1	48.0 ± 3.10
14.7 ± 0.1	48.0 ± 3.10
14.8 ± 0.1	41.0 ± 2.86
14.9 ± 0.1	42.0 ± 2.89
15.0 ± 0.1	50.0 ± 3.16

24kV data look like this

TABLE VII
MEASUREMENTS AT 24KV

Angle(degrees)	Count per Second
3.0 ± 0.1	177.0 ± 5.95
3.1 ± 0.1	160.0 ± 5.66
3.2 ± 0.1	114.0 ± 4.77
3.3 ± 0.1	93.0 ± 4.31
3.4 ± 0.1	58.0 ± 3.41
.	.
.	.
11.3 ± 0.1	86.0 ± 4.14
11.4 ± 0.1	72.0 ± 3.79
11.5 ± 0.1	78.0 ± 3.95
11.6 ± 0.1	78.0 ± 3.95
11.7 ± 0.1	83.0 ± 4.07

27kV data look like this

TABLE VIII
MEASUREMENTS AT 27KV

Angle(degrees)	Count per Second
3.0 ± 0.1	289.0 ± 7.60
3.1 ± 0.1	272.0 ± 7.38
3.2 ± 0.1	184.0 ± 6.06
3.3 ± 0.1	145.0 ± 5.39
3.4 ± 0.1	110.0 ± 4.69
.	.
.	.
9.5 ± 0.1	126.0 ± 5.02
9.6 ± 0.1	127.0 ± 5.05
9.7 ± 0.1	112.0 ± 4.73
9.8 ± 0.1	124.0 ± 5.01
9.9 ± 0.1	126.0 ± 5.02

30kV data look like this

TABLE IX
MEASUREMENTS AT 30KV

Angle(degrees)	Count per Second
3.0 ± 0.1	448.0 ± 9.47
3.1 ± 0.1	415.0 ± 9.11
3.2 ± 0.1	293.0 ± 7.66
3.3 ± 0.1	217.0 ± 6.59
3.4 ± 0.1	160.0 ± 5.66
.	.
.	.
8.6 ± 0.1	158.0 ± 5.63
8.7 ± 0.1	162.0 ± 5.69
8.8 ± 0.1	148.0 ± 5.44
8.9 ± 0.1	148.0 ± 5.44
9.0 ± 0.1	147.0 ± 5.42

here is the code we have used below:

```
{
   double d = 5.64*1e-10;
   double rad = 180/(TMath::Pi());
   double slope,intercept;
   double
       slopeW,interceptW,slopeW_error,interceptW_error;
   float eV = 1.60217663e-19;
   double c = 299792458.;

   std::vector<std::string> filenames = {
      "calib35kv.txt",
      "15kv.txt",
      "18kv.txt",
      "21kv.txt",
      "24kv.txt",
      "27kv.txt",
      "30kv.txt",


    };

      std::vector<std::string> titles = {
      "calib35v",
      "15kv",
      "18kv",
      "21kv",
      "24kv",
      "27kv",
      "30kv",


   };

     std::vector<std::vector<double>>
        peakRanges = {
      {7.38,7.69},{13.2,13.51},{14.85,15.29},{19.85,20.2


   };

   std::vector<std::vector<double>>
       linefitRanges = {
      {0.172*1e-9,0.187*1e-9},
      {0.142*1e-9,0.156*1e-9},
      {0.125*1e-9,0.14*1e-9},
      {0.118*1e-9,0.138*1e-9},
      {0.10*1e-9,0.115*1e-9} ,
      {0.095*1e-9,0.113*1e-9},
   };
```

```
float peakValues[5],sigmaValues[5];
float theoreticalPeaks[5]=
    {7.76,13.93,15.69,19.73,23.9};

float
    mumins[6],musigmas[6],fmaxs[6],fsigmas[6];
float energies[6] =
    {eV*15000,eV*18000,eV*21000,eV*24000,eV*27000,eV*30000};



for(int j =0;j<filenames.size();j++){



if (j==0){
TGraphErrors *gr = new
    TGraphErrors(filenames[j].c_str());
gr->Draw("A*");
gr->GetXaxis()->SetTitle("Angle (Degree)");
gr->GetYaxis()->SetTitle("Accumulation
    Rate(Impulse/Second)");
gr->SetTitle("Accumulation Rate vs Angle");

for (int i = 0; i < 5; ++i) {
      TF1 *gauss = new
          TF1(Form("gauss_%d_%d", j, i),
          "gaus", peakRanges[i][0],
          peakRanges[i][1]);
      gr->Fit(gauss, "R");


      gauss->Draw("same");
      double peak = gauss->GetParameter(1);
      double sigma = gauss->GetParameter(2);
      cout << "the "<< i << "'th peak is"
          << peak <<"and its sigma is="<<
          sigma << "\n";
      peakValues[i]=peak;
      sigmaValues[i]=sigma;



   }
   TCanvas *c = new TCanvas();


   TGraphErrors *calib = new
       TGraphErrors(5,peakValues,theoreticalPeaks,sigmaValues,0);
   calib->Draw("A*");
   calib->GetXaxis()->SetTitle("Experimental
       Theta Values(Degrees)");
   calib->GetYaxis()->SetTitle("Nominal
       Thteta Values(Degrees)");
   calib->SetTitle("Experimental vs Nominal
       Theta Values");

   TF1 *linefit = new
       TF1("linefit","[0]*x+[1]");
   linefit->SetParNames("Slope",
       "Intercept");
   linefit->SetParameters(1,0);
   calib->Fit(linefit);
   gStyle->SetOptFit(1);

   slope = linefit->GetParameter(0);
   intercept = linefit->GetParameter(1);
      TCanvas *c2 = new TCanvas();

}
else if (j != 0) {
int ndata;
std::ifstream
    inputFile(filenames[j].c_str());
std::string line;
int linecount = 0;

// First, count the number of lines in the
    file
while (std::getline(inputFile, line)) {
   linecount++;
}

ndata = linecount;

// Now, rewind the file stream to read data
inputFile.clear();
inputFile.seekg(0, std::ios::beg);

// Create vectors to store the data
std::vector<float> angles, impulses,
    angleErrors, impulseErrors;

float angle, impulse, angleError,
    impulseError;
while (std::getline(inputFile, line)) {
   std::istringstream iss(line);
   iss >> angle >> impulse >> angleError >>
       impulseError;

   // Push the values into the vectors
   angles.push_back(angle);
   impulses.push_back(impulse);
   angleErrors.push_back(angleError);
   impulseErrors.push_back(impulseError);
}

ndata = angles.size();
std::vector<float> wavelengths(ndata),
    wavelengthErrors(ndata);

for (int i = 0; i < ndata; ++i) {
   wavelengths[i] = 2 * d * sin((angles[i]
       * slope + intercept) / rad);
   wavelengthErrors[i] = 2 * d *
       cos((angles[i] * slope + intercept)
       / rad) * (0.1 / rad);
}

TGraphErrors *gr = new TGraphErrors(ndata,
    wavelengths.data(), impulses.data(),
    wavelengthErrors.data(),
    impulseErrors.data());
gr->Draw("A*");
gr->GetXaxis()->SetTitle("Wavelength(m)");
gr->GetYaxis()->SetTitle("Accumulation
    Rate(Impulse/Second)");
gr->SetTitle(Form("Accumulation Rate vs
    Wavelength - %s", titles[j].c_str()));

TF1 *linefitW = new
    TF1("linefit","[0]*x+[1]");
```

```
   linefitW->SetParNames("Slope", "Intercept");
   linefitW->SetParameters(1.15*1e12,-200);
   linefitW->SetRange(linefitRanges[j - 1][0],
       linefitRanges[j - 1][1]); // Set the
       fitting range


   gr->Fit(linefitW,"R");
   slopeW = linefitW->GetParameter(0);
   interceptW = linefitW->GetParameter(1);

   slopeW_error = linefitW->GetParError(0);
   interceptW_error = linefitW->GetParError(1);

   cout << "the slope is "<< slopeW<< "'+- "<<
       slopeW_error << "'the intercept is "
    <<interceptW<< "'+- "<< interceptW_error<<
        "\n";

    mumins[j-1] = -interceptW / slopeW;
   musigmas[j-1] = sqrt(
      pow(interceptW_error / slopeW, 2) +
      pow(interceptW * slopeW_error / (slopeW
          * slopeW), 2)
   );

   fmaxs[j-1] = c/mumins[j-1];
   fsigmas[j-1] = (c * musigmas[j-1]) /
       (mumins[j-1] * mumins[j-1]);

   cout << "mu mins are "<< mumins[j-1] << "
       +- " <<musigmas[j-1]<< "\n";
   cout << "f max are "<< fmaxs[j-1] << " +- "
       <<fsigmas[j-1]<< "\n";



   TCanvas *c3 = new TCanvas();
}
   }

TGraphErrors *planck = new
    TGraphErrors(6,fmaxs,energies,0,0);
   planck->Draw("A*");
   planck->GetXaxis()->SetTitle("Frequency(Hz)");
   planck->GetYaxis()->SetTitle("Energy(J)");
   planck->SetTitle("Energy vs Frequency");

    TF1 *linePlanck = new
        TF1("linefit","[0]*x+[1]");
      linePlanck->SetParNames("Slope",
          "Intercept");
      linePlanck->SetParameters(6*1e-34,1*1e-15);
      planck->Fit(linePlanck);

      double Planck =
          linePlanck->GetParameter(0);
      double PlanckError =
          linePlanck->GetParError(0);

      cout << "planck constant is found out to
          be "<< Planck<< " +- "
          <<PlanckError<<"\n";


}
```