X-Rays and Duane-Hunt Displacement Law

Şükrü Çağlar, Ege Adıgüzel, Semih Dülger

PHYS443 Fall 2025 Experiment Report, 03.10.2025

Abstract

In this experiment, the energy and wavelength properties of X-rays generated by a Molybdenum (Mo) source were investigated. Using a Sodium Chloride (NaCl) crystal as the diffraction medium, the characteristic X-ray spectrum of Mo was analyzed with a Geiger-Muller counter. The primary objective of the experiment was to find the correlation between the energy and minimum wavelength of emitted X-rays, as described by the Duane-Hunt Law, and to obtain an experimental value for Planck's constant. The value was determined to be $h = 6.47 \times 10^{-34} \pm 3.08 \times 10^{-35}$, which is approximately 0.5σ away from the accepted value of $h = 6.63 \times 10^{-34} J.s$.

I Introduction and Theory

History and Motivation: In the years following the discovery of X-rays, detailed analysis of their spectra revealed sharp peaks at certain energies specific to the X-ray source also known as the characteristic radiation [8] and a continuous background spectrum called Bremsstrahlung (German for "braking radiation") [4]. In 1915, William Duane and Franklin Hunt conducted a precise study of this continuous Bremsstrahlung spectrum.

They were the first to demonstrate the relationship between the maximum energy of the X-ray photons and the energy of the electrons used to generate them. Their work established that for any given voltage, there was a definite lower limit to the wavelength of the radiation produced in the continuous Bremsstrahlung spectrum. This discovery, later named the Duane-Hunt Law [5], provided crucial evidence for the quantum nature of light and its interactions with the atomic structure.

Theory: The experiment is based on the characteristics of Bremsstrahlung event and its minimum radiated wavelength for a given source material. In this event, electrons are energized using a fixed accelerating potential V_{app}, and send through a source material (Molybdenum (Mo) for this experiment). Due to interactions between electrons and the atoms of the source material, electrons slow down or are diverted, resulting in the emission of a photon. With the highest possible energy,which corresponds to the case of full deflection of electrons due to collision with nuclei is given as:

$$E_{max} = qV_{app} = \frac{hc}{\lambda_{min}} \tag{1}$$

Also known as the Duane-Hunt displacement Law.

Ionization of K shell electrons of the source element is another contributor to the emission of X-rays, which produces characteristic high-intensity peaks specific to the material's ionization energies $K_{\alpha 1}$, $K_{\alpha 2}$ and K_β. The characteristic x-ray emission energies for Molybdenum (Mo)[7] are:

$$K_\alpha^* = \frac{K_{\alpha 1} + K_{\alpha 2}}{2} = 17426.8\,\text{eV}$$
$$K_\beta = 19589.8\,\text{eV}$$

The wavelengths associated with these energies can be calculated using (1). Another crucial part of this experiment is associating wavelengths with diffraction angles using an NaCl crystal, which is our main tool for measuring the intensities of both the characteristic emission and the Bremsstrahlung spectrum at different wavelengths. Bragg's law[3] provides the direct relationship needed between the angle of constructive interference in a crystal structure and the photon's wavelength, λ.

$$n\lambda = 2d\sin\theta \tag{2}$$

where $d = 2.8201$ Å for NaCl [1] is the spacing between lattice layers and n is the order of diffraction.

By measuring the X-ray intensity at different diffraction angles, we can construct a plot showing the relation between intensity and wavelength throughout a continuous spectrum. The Bremsstrahlung portion of this spectrum is a curve that rises sharply from zero intensity at a specific cutoff wavelength, λ_{min}. While the entire spectrum is not linear, the initial rising edge can be approximated by a straight line. We can use this linear relationship to determine the minimum allowed wavelength

by extrapolating the line to the x-axis. Using (1), this minimum wavelength then provides an experimental value for Planck's constant, calculated using the form:

$$E_{max}\frac{\lambda_{min}}{c} = \frac{E_{max}}{f_{max}} = h \tag{3}$$

II Setup and Method

Setup: The experimental setup consists of

- X-ray diffractometer with a Molybdenum-target X-ray tube, a NaCl crystal for diffraction, and a Geiger-Muller counter tube as the detector.
- Computer

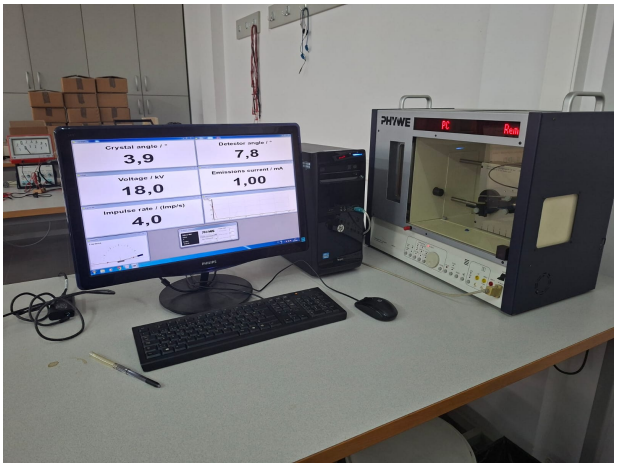


Figure 1: Full experiment setup

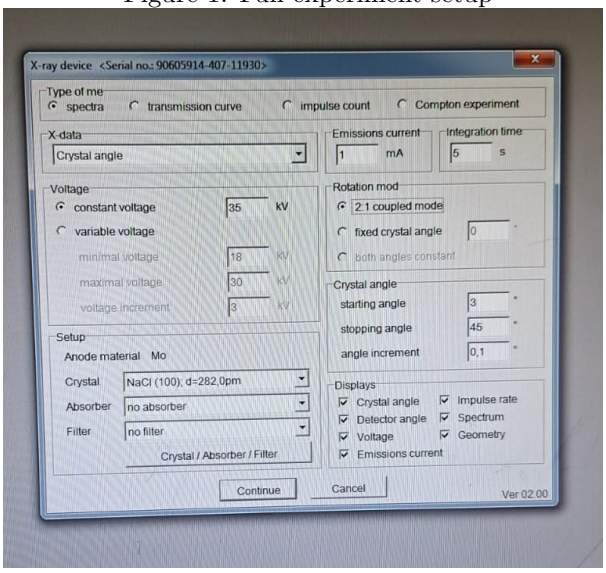
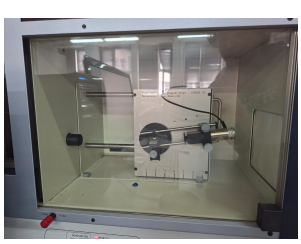
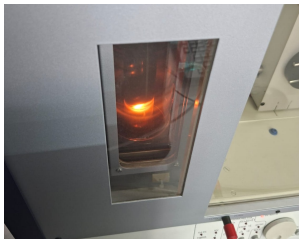


Figure 2: Software used to take the measurements



(a)



(b)

Figure 3: (a) NaCl diffractor and Geiger-Muller counter (b) X-ray generator with Mo source

Method: The procedure for this experiment using the PHYWE X-ray apparatus configured as a Bragg-Brentano diffractometer is as follows:

1. **Calibration:** The X-ray apparatus is powered on. The accelerating voltage for the X-ray tube is set to its maximum value of 35 kV, and the emission current is set to 1.0 mA. An automated scan is performed over an angular range of 3° to 35° to measure the X-ray intensity as a function of the diffraction angle (θ). The resulting dataset is compared with theoretically expected values from (2) and is used for calibration.
2. **Data Acquisition at Various Voltages:** The complete intensity scan is repeated for a smaller angular range of 3° to 15° to focus on the rising edge of the Bremsstrahlung spectrum. The measurement is performed for seven different accelerating voltages applied to the X-ray tube: 35 kV, 30 kV, 27 kV, 24 kV, 21 kV, 18 kV, and 15 kV. The emission current is kept constant at 1.0 mA for all measurements.
3. **Data Processing:** For each voltage setting, the raw data (counts vs. angle) is processed. The diffraction angles (θ) are converted into their corresponding X-ray wavelengths (λ) using Bragg's Law. This allows an intensity-versus-wavelength spectrum to be constructed for each dataset. This spectrum is then analyzed to finally extract a value for Planck's constant h.

III Data

The raw experimental datasets consist of intensity, measured in Geiger counts per-second interval, versus scattering angle, which was recorded in 0.1-degree increments. Error for the angles are fixed to 0.1 degrees and although raw dataset does not include errors for the number of counts we can assume counting

proccess is poisson and say:

$$\sigma_{counts} = \sqrt{\text{number of counts per second}}$$

Each Bremsstrahlung dataset contains approximately 50-110 data points, while the calibration set contains 312. Calibration data can be seen below and the other datasets are presented in data list. One can also refer to [**phys442repo**] for complete the raw data of the experiment.

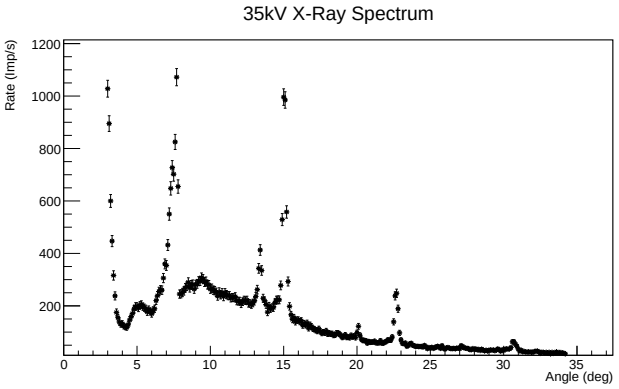


Figure 4: Calibration dataset counts/second vs. diffraction angle

IV Analysis

Calibration:

Characteristic emmision wavelengths for Molybdenum can be calculated from K^*_α and K_β using:

$$\lambda_\alpha = \frac{hc}{K^*_\alpha \times 1.6021 \times 10^{-19}}$$
$$\lambda_\beta = \frac{hc}{K_\beta \times 1.6021 \times 10^{-19}}$$

where 1.6021×10^{-19} is the conversion constant from eV to Jules. for the further peaks $\lambda_{\alpha 2}, \lambda_{\beta 2}, \lambda_{\alpha 3}, \lambda_{\beta 3}, \lambda_{\alpha 4}$ etc. we can use braggs law in higher orders of n:

$$\theta_{\alpha n} = arcsin\left(\frac{2d}{n\lambda_\alpha}\right)$$
$$\theta_{\beta n} = arcsin\left(\frac{2d}{n\lambda_\beta}\right)$$

As can be seen in fig.4 the calibration dataset contains 6 distinct peaks. There is also not so noticable $K_{\beta 1}$ shoulder in the beginning. 7 gaussian fits for appropriate ranges is applied to find the experimental θ values for each peak.

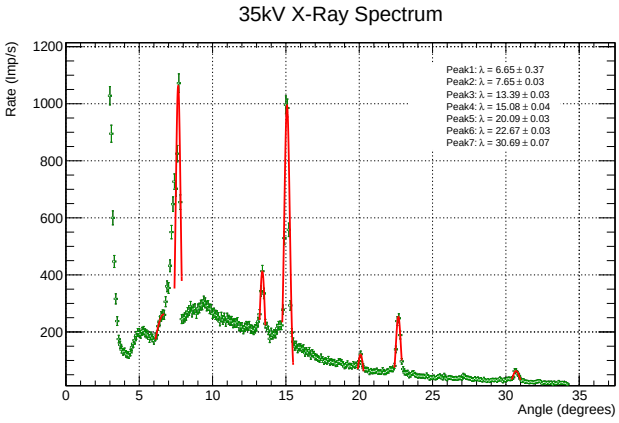


Figure 5: Gaussian fits and their mean. Errors for the θ values is calculated using the variance (σ^2) of each gaussian peak.

Theoretical and experimental θ values as well as their errors are:

peak	$\theta_{theoretical}$	θ_{exp}	σ_{exp}
$\lambda_{\beta 1}$	6.44	6.65	0.371
$\lambda_{\alpha 1}$	7.25	7.65	0.0287
$\lambda_{\beta 2}$	13.0	13.4	0.0342
$\lambda_{\alpha 2}$	14.6	15.1	0.0359
$\lambda_{\beta 3}$	19.7	20.1	0.0322
$\lambda_{\alpha 3}$	22.2	22.7	0.0287

It worth noting that we omit the last peak $\theta = 30.7$ since it is lacks the structure of α - β beta pair which makes it harder to classify as α or β. Remaining 6 datapoints are used to determine a calibration function fo form:

$$\theta_{theoretical} = m.\theta_{exp} + b \tag{4}$$

with the new error in θ_{exp} being:

$$\sigma_\theta = \sqrt{(\sigma_m.\theta_{exp})^2 + (m.\sigma_{\theta_{exp}})^2 + \sigma_b^2} \tag{5}$$

Derivation fro this an all the other error propagation formulas are given in the Appendix on error propagation. Linear fit for the theoretical and experimental θ's yielded $m = 0.99 \pm 2.48 \times 10^{-3}$ and $b = -0.41 \pm 4.17 \times 10^{-2}$:

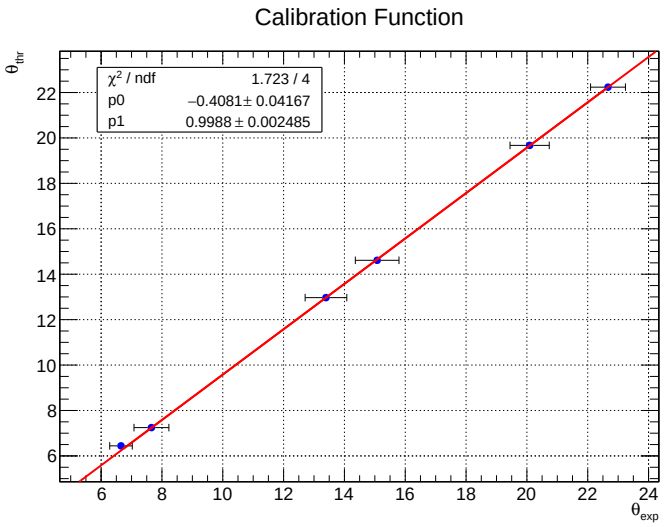


Figure 6: Calibration function parameters from the linear fit. Here errors for all the data points except the first point are extrapolated by 20 to make them more visual. Actual fit is performed with the real values.

The reason behind this calibration step is to identify any systematic error that affects all the measurements in similar ways (hence the reason for the linear approximation) in the PHYWE X-Ray instrument and adjust the measurements accordingly. $\chi^2 \approx 1.73$ suggest a good fit and consistency in the relation of each point to the theoretical values.

calculating λ_{min}:

After the datasets taken for 15 to 30keV energies are calibrated to θ_{cal} using (4) and (5),Bragg's law for the first order ($n = 1$) is used to convert θ_{cal} to λ.

$$\lambda = 2d \cdot sin(\theta_{cal})$$

To propagate the error from θ_{cal} to the λ, following relation is necessary:

$$\sigma_\lambda = 2d \cdot cos(\theta_{cal}) \cdot \sigma_{\theta_{cal}}$$

Once again, derivation for this relation can be found in the Appendix on error propagation. The Bremsstrahlung cutoff wavelength λ_{min} is then determined by fitting a linear function to the rising edge of the intensity vs. wavelength spectrum. We can extrapolate the line to the x-intercept to determine a minimum λ where the intensity is 0. Below is an example for the 18keV measurement. Refer to data list for the rest of the fits.

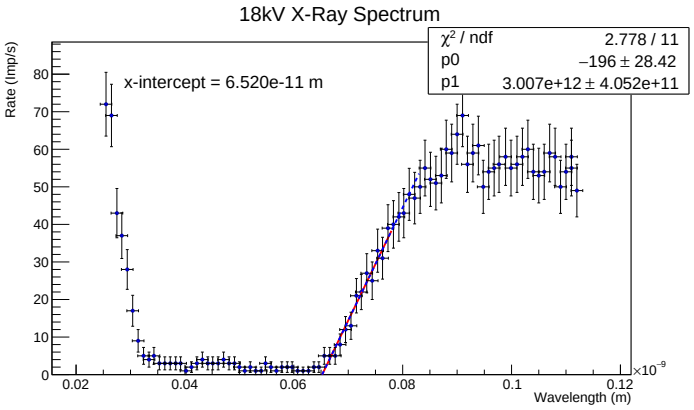


Figure 7: Intensity vs. wavelength plot with x-intercept as the λ_{min}

Calculation and error propagation for the x-intercept is explained in detail in the error propagation. Intercept can be found as:

$$\lambda_{min} = x_{\text{intercept}} = -\frac{p_0}{p_1}$$

and the error propagation formula is:

$$\sigma_x = \sqrt{\left(\frac{\sigma_{p_1}}{p_0}\right)^2 + \left(\frac{p_0\,\sigma_{p_0}}{p_1^2}\right)^2 - 2\,\frac{p_0\,\mathrm{Cov}(p_0,p_1)}{p_1^3}}$$

Minimum wavelengths with their errors for all the energy levels used in the experiment are:

Energy (keV)	X-Intercept, λ_{min} (m)	$\sigma_{\lambda_{min}}$ (m)
15	7.96×10^{-11}	9.37×10^{-13}
18	6.52×10^{-11}	7.99×10^{-13}
21	5.60×10^{-11}	1.06×10^{-12}
24	5.15×10^{-11}	1.67×10^{-12}
27	4.60×10^{-11}	1.94×10^{-12}
30	4.01×10^{-11}	3.07×10^{-12}

Using Duane-Hunt Displacement Law to find h:

As the Duane-Hunt Law suggests Planck's constant is the simple ratio of:

$$\frac{E_{max}}{f_{max}} = h$$

where;

$$f_{max} = \frac{c}{\lambda_{min}}$$

and

$$\sigma_f = \frac{c}{\lambda^2} \cdot \sigma_\lambda$$

Energies used to accelerate electrons has the unit kilo-electronVolts (keV). This should be converted to Joules before using in the equation according to $1J = 1eV \times 1.602 \times 10^{-19}$. After this, converted energies and maximum frequencies with their errors are:

Energy $(10^{-15}J)$	f_{max} $10^{18}s^{-1}$	σ_f $\times 10^{16}s^{-1}$
2.40	3.77	4.43
2.88	4.60	5.64
3.36	5.35	1.01
3.85	5.82	1.89
4.33	6.51	2.74
4.81	7.48	5.73

slope of a linear fit to this values should give desired experimental result for Planck's constant. However, linear regression algorithm in ROOT is not as precise as calculating the errors for a parameter as it is to approximate the fit parameter itself. So for very small numbers, errors for a fitted parameter might be as high as 10^{20} times higher than the actual parameter. To counteract this, a scaling is performed before any fitting to data. Below is the linear fit to the scaled energy vs. max frequency graph.

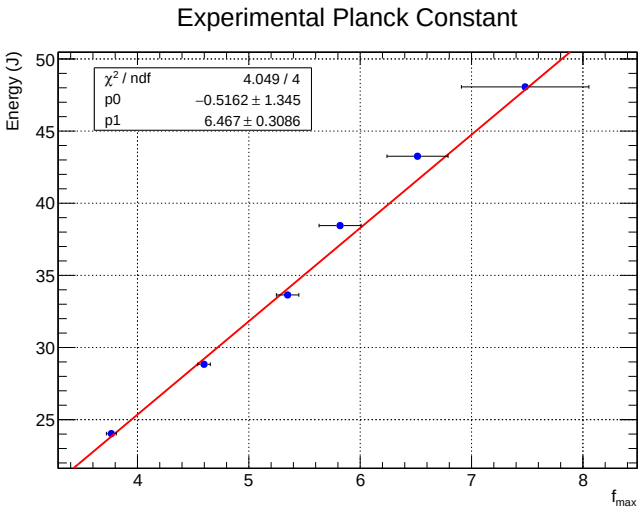


Figure 8: Energy vs. max frequency graph. axis are scaled to minimize fit problems by a factor of 10^{16} to the energies and 10^{-18} to the frequencies and their errors. Result is then multiplied by $10^{-16}/10^{18} = 10^{-34}$ to revert the scaling operation.

V Results and Conclusion

Final experimental value for the Planck's constant is determined to be $h = 6.47 \times 10^{-34} \pm 3.08 \times 10^{-35} J \cdot s$ after the re-scaling is performed. Final graph also showed a non-zero intercept in energy vs. frequency relation which suggest either:

- a rightward shift in the frequency domain as a result of higher than expected frequencies
- a downward shift in the energy domain corresponding to lower than expected (or measured) energies

It seems more likely for this experiment that the second case is more likely and reported energy values based on the accelerating potential optimistically represents the actual energies of the emitted X-Rays. This might be caused some types of unknown breaking potential inside the x-ray generator component of the apparatus that slows the electrons before they reach the Mo X-Ray source.

Sources of other experimental uncertainties include poisson nature of the geiger-müller counter combined with the small sampling time of 5 seconds as well as the fit uncertainty in calibration process which propagated throughout the rest of the experiment. Regardless, reported experimental value for the Planck's constant differs from the expected $6.63 \times 10^{-34} J \cdot s$ by $\approx \%2.35$. Which is:

$$\frac{|6.47 \times 10^{-34} - 6.63 \times 10^{-34}|}{3.08 \times 10^{-35}} \approx 0.52\sigma$$

sigmas away and suggest a reliable finding considering the sources of uncertainties listed above.

Appendix

A.1 Error propagation formulas

To calculate the uncertainty on that function based on the uncertainty of its variables, we can refer to general error propagation equation for multi variable functions. For a function of $f(x_1, x_2, ...)$ the uncertainty is:

$$\sigma_f^2 = \sum_{i=1}^{n} \left(\frac{\partial f}{\partial x_i} \sigma_{x_i} \right)^2 \tag{6}$$

Derivation of error for $\theta_{cal} = m\theta_{exp} + b$:

Here $f(\theta)$ is:

$$m\theta_{exp} + b$$

from the general formula:

$$\sigma_\theta^2 = \left(\frac{\partial f}{\partial m}.\sigma_m \right)^2 + \left(\frac{\partial f}{\partial \theta_{exp}}.\sigma_{\theta_{exp}} \right)^2 + \left(\frac{\partial f}{\partial b}.\sigma_b \right)^2$$

and

$$\frac{\partial f}{\partial m} = \theta_{exp}, \frac{\partial f}{\partial \theta_{exp}} = m, \frac{\partial f}{\partial b} = 1$$

so:

$$\sigma_\theta = \sqrt{(\sigma_m.\theta_{exp})^2 + (m.\sigma_{\theta_{exp}})^2 + \sigma_b^2}$$

Error propagation for x-intercept:

for the linear approximation $y = p_0 + p_1 \times x$, x-intercept corresponds to:

$$x_{\text{intercept}} = -\frac{p_0}{p_1}$$

We shall update the general formula for error propagation in multi variable functions as:

$$\sigma_f^2 = \sum_{j=1}^{n} \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} \frac{\partial f}{\partial x_j} Cov(x_i, x_j)$$

This reduced to (6) when the variables are independent and $Cov(x_i, x_j) = 0$ for $i \neq j$. In this case p_0 and p_1 are not independent thus $Cov(p_0, p_1) \neq 0$. from the intercept equation:

$$\sigma_x^2 = \left(\frac{\partial f}{\partial p_0}.\sigma_{p_0} \right)^2 + \left(\frac{\partial f}{\partial p_1}.\sigma_{p_1} \right)^2 + \frac{\partial f}{\partial p_0}.\frac{\partial f}{\partial p_1}.Cov(p_0, p_1)$$

and

$$\frac{\partial f}{\partial p_0} = -p_1^{-1}, \frac{\partial f}{\partial p_1} = \frac{p_0}{p_1^2}$$

Thus the error equation becomes:

$$\sigma_x^2 = \left(\frac{p_0}{p_1^2} \right)^2 \sigma_{p_1}^2 + \left(-\frac{1}{p_1} \right)^2 \sigma_{p_0}^2 + 2 \left(\frac{p_0}{p_1^2} \right) \left(-\frac{1}{p_1} \right) \mathrm{Cov}(p_1, p_0)$$

or

$$\sigma_x = \sqrt{\left(\frac{\sigma_{p_1}}{p_0} \right)^2 + \left(\frac{p_0\, \sigma_{p_0}}{p_1^2} \right)^2 - 2\, \frac{p_0\, \mathrm{Cov}(p_0, p_1)}{p_1^3}}$$

References

[1] M. J. Buerger. "The lattice constant of NaCl". In: *American Mineralogist* 11.3 (1926), pp. 59–61. DOI: 10.1007/BF00853362.

[2] capta1Nemo. *PHYS443 Experiments GitHub Repository*. https://github.com/capta1Nemo/443experiments.git.

[3] Encyclopædia Britannica. *Bragg Law*. https://www.britannica.com/science/Bragg-law. n.d. (Visited on 11/22/2024).

[4] Encyclopædia Britannica. *Bremsstrahlung*. https://www.britannica.com/science/bremsstrahlung. 2024.

[5] Encyclopædia Britannica. *Duane-Hunt Law*. https://www.britannica.com/technology/Duane-Hunt-law. n.d. (Visited on 11/22/2024).

[6] E. Gülmez. *Advanced Physics Experiments*. Boğaziçi University, 1997.

[7] PHYWE SYSTEME GMBH & Co. KG. *X-Ray Emmision And Absorption*. https://dlf.ug.edu.pl/wp-content/uploads/2014/03/X-Ray-Mo.pdf.

[8] Radiopaedia. *Characteristic Radiation*. https://radiopaedia.org/articles/characteristic-radiation. n.d. (Visited on 11/22/2024).

Additional Material

Full list of datasets

The datasets in text format can also be found in [2].

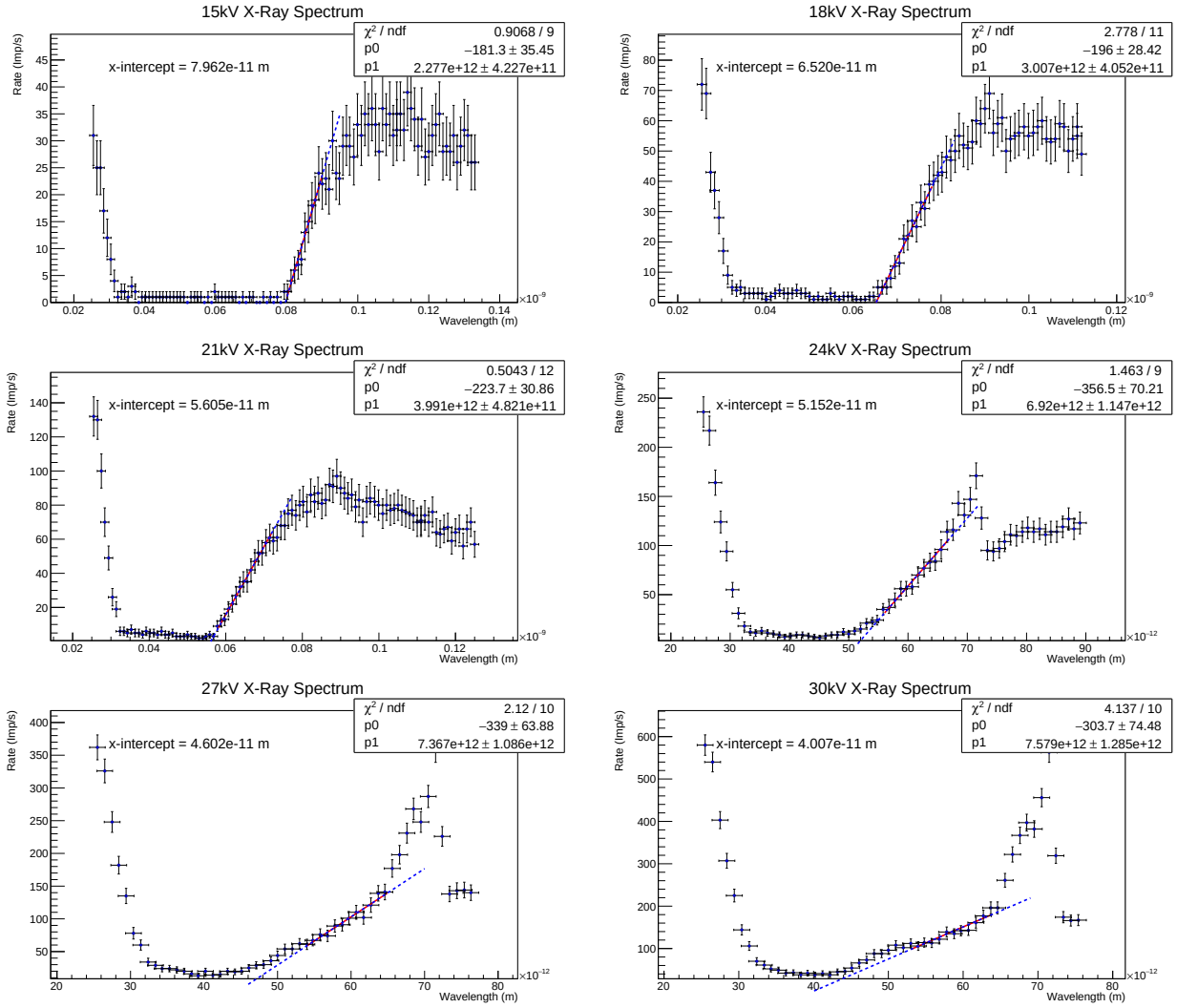


Figure 9: Bremsstrahlung spectrum datasets (with angles converted to wavelength) and λ_{min} as the x-intercept.

Codes

All the codes here and more can be found in the GitHub repository [2].

```

h=6.6256e-34
C=2.9979e8
eV=1.6021e-19
d=282e-12
def wavelength(K):
    return h*C/(K*eV)
def tetha(n,wavelength):
    return float(np.degrees(np.arcsin(n*wavelength/(2*d))))

def calerror(tetha):
    return float(np.sqrt((s_m*tetha)**2+(m*0.1)**2+s_b**2))

def calibrate(tetha):
    m=cal_parameters[1][0]
    b=cal_parameters[0][0]
    return tetha*m+b

def braggs(tetha):
    return np.sin(np.deg2rad(tetha))*2*d

def err_braggs(tetha,err_tetha):
    return np.cos(np.deg2rad(tetha))*2*d*np.deg2rad(err_tetha)

def intercept_err(parameters):
    a=parameters[0]
    b=parameters[1]
    sa=parameters[2]
    sb=parameters[3]
    cov_ab=parameters[4]
    sigma=float(np.sqrt((sb/a)**2 + (b*sa/a**2)**2 - 2*b*cov_ab/(a**3))) #- 2*b*cov_ab/(a
        **3) here if you remove
    intercept=float(-b/a)
    return(intercept,sigma)

def frequency_err(l):
    f=C/l[0]
    sf=f*l[1]/l[0]
    return (f,sf)

def rescale_f(freq):
    f=freq[:,0]*1e-18
    sf=freq[:,1]*1e-18
    return np.array([(float(f[i]),float(sf[i])) for i in range(len(f))])

def rescale_e(energy):
    return [float(energy[i]*1e16) for i in range(len(energy))]
```

Code 1: Various functions used in the analysis and some error propogation formulas

```
intervals = [
    (6.1,6.71),
    (7.4, 7.9),
    (13.2, 13.6),
    (14.75, 15.5),
    (19.9, 20.3),
    (22.4, 22.9),
    (30.4, 31.0)
]

c1 = ROOT.TCanvas("c1", "Canvas␣for␣35kV", 1400, 900)
ROOT.gStyle.SetOptFit(0)
ROOT.gStyle.SetOptStat(0)

```

```
file_path = "processed_data/35kV-XRay_processed.txt"
graph = ROOT.TGraphErrors(file_path)
graph_points = graph.Clone("graph_points")
graph_errors = graph.Clone("graph_errors")

graph_points.SetMarkerStyle(20)
graph_points.SetMarkerSize(0.5)
graph_points.SetMarkerColor(ROOT.kBlue+2)

graph_errors.SetLineColor(ROOT.kGreen+2 )
graph_errors.SetMarkerColor(0)
graph_errors.SetLineWidth(1)
graph_errors.SetMarkerStyle(0)

graph_points.Draw("AP")
graph_errors.Draw("P SAME")
graph_points.SetTitle("35kV X-Ray Spectrum;Angle (degrees);Rate (Imp/s)")


legend = ROOT.TLegend(0.60, 0.65, 0.82, 0.85)
legend.SetBorderSize(0)
legend.SetFillColor(ROOT.kWhite)
legend.SetFillStyle(1001)
legend.SetTextFont(42)
legend.SetTextSize(0.025)

values, errors = [], []

for i, (xmin, xmax) in enumerate(intervals, start=1):
    fit_name = f"gauss_fit_{i}"
    gauss_fit = ROOT.TF1(fit_name, "[0]*TMath::Gaus(x,[1],[2])", xmin, xmax)
    gauss_fit.SetParameters(10, (xmin + xmax)/2, 0.5)
    gauss_fit.SetLineColor(ROOT.kRed)

    gauss_fit.SetLineWidth(2)

    graph_errors.Fit(gauss_fit, "RQ+")

    mean = gauss_fit.GetParameter(1)
    sigma = gauss_fit.GetParameter(2)
    variance = sigma**2

    values.append(mean)
    errors.append(variance)

    legend.AddEntry("", f"Peak{i}: \lambda = {mean:.2f} \pm {variance:.2f}", "n")

legend.Draw()
ROOT.gPad.SetTicks(1,1)
ROOT.gPad.SetGrid(1,1)

c1.Update()
c1.SaveAs("35kV-XRay_CalibrationPlot.pdf")
del c1, graph, gauss_fit, legend
```

Code 2: PyROOT code to fit gaussians to 35keV data

```
import ROOT
ROOT.gStyle.SetOptFit(1)
# Data
y = cal_set[0]
x = cal_set[1]
sx = cal_set[2]
sy = [0]*6

graph = ROOT.TGraphErrors(len(x))
for i in range(len(x)):
    graph.SetPoint(i, x[i], y[i])
```

```
    graph.SetPointError(i, sx[i], sy[i])

fit_func = ROOT.TF1("fit_func", "[0] + [1]*x", min(x), max(x))
fit_func.SetParameters(0, 1)  # initial guesses

graph.Fit(fit_func)
cal_parameters=[(fit_func.GetParameter(i),fit_func.GetParError(i)) for i in range(2)]

visual_scale_factor =20
starting_point=2
print(f"Applying a visual scaling factor of {visual_scale_factor} to points {starting_point}
    to {len(x)}'s error bars for drawing.")

for i in range(1,len(x)):
    graph.SetPointError(i, sx[i] * visual_scale_factor, sy[i])
graph.SetMarkerStyle(20)
graph.SetMarkerColor(ROOT.kBlue)
graph.SetTitle("Calibration Function;#theta_{exp};#theta_{thr}")

c = ROOT.TCanvas("c", "Line Fit", 800, 600)
graph.Draw("AP")
fit_func.Draw("same")
c.Update()
stats = graph.GetListOfFunctions().FindObject("stats")
if stats:
    stats.SetX1NDC(0.15)  # lower-left x (0 to 1)
    stats.SetY1NDC(0.75)  # lower-left y
    stats.SetX2NDC(0.45)  # upper-right x
    stats.SetY2NDC(0.87)   # upper-right y
    stats.SetTextSize(0.03)
    c.Modified()

ROOT.gPad.SetTicks(1, 1)
ROOT.gPad.SetGrid(1, 1)
c.Update()
c.SaveAs("cal_function.pdf")
del c, graph, fit_func
```

Code 3: PyROOT code for finding the calibration function

```
ROOT.gROOT.SetBatch(True)
L=[(15,(8e-11,9e-11)),(18,(6.50e-11,7.8e-11)),
   (21,(5.80e-11,7.20e-11)),(24,(5.60e-11,6.7e-11)),
   (27,(5.4e-11,6.5e-11)),(30,(5.3e-11,6.4e-11))]
intercept_params=[]
for idx, (kv, (fit_a, fit_b)) in enumerate(L):
    c = ROOT.TCanvas(f"c{idx}", f"{kv}kV", 1000, 600)

    data = np.loadtxt(f"wave_data/{kv}kV-XRay_lambda.txt")
    x = data[:,0]
    y = data[:,1]
    sx = data[:,2]
    sy = data[:,3]

    graph = ROOT.TGraphErrors(len(x))
    for i in range(len(x)):
      graph.SetPoint(i, x[i], y[i])
      graph.SetPointError(i, sx[i], sy[i])
    graph.SetTitle(f"{kv}kV X-Ray Spectrum ;Wavelength (m);Rate (Imp/s)")
    graph.SetMarkerStyle(20)
    graph.SetMarkerSize(0.7)
    graph.SetMarkerColor(ROOT.kBlue +1)
    graph.Draw("AP")
    fit = ROOT.TF1("line_fit", "[0] + [1]*x",fit_a,fit_b)
    fit.SetParameter(0, 0.0)
    fit.SetParameter(1, 1e11)
    fit_result = ROOT.TFitResultPtr(graph.Fit(fit, "RSQ"))
    cov_ab = fit_result.CovMatrix(0, 1)  # i ditch this if i need
```

```
    var_a  = fit_result.CovMatrix(1, 1)  # variance of slope
    var_b  = fit_result.CovMatrix(0, 0)  # variance of intercept

# Convert to standard deviations
    sa = float(np.sqrt(var_a))
    sb = float(np.sqrt(var_b))

    a = fit.GetParameter(1)
    b = fit.GetParameter(0)
    intercept_params.append([a,b,sa,sb,cov_ab])#cov_ab here
    x_intercept = -b / a
    print(f"{kv}kV x-intercept: {x_intercept:.3e} m")
    y_max = (fit_b+0.5e-11)*a+b
    xline = ROOT.TLine(x_intercept, 0, fit_b+0.5e-11, y_max)
    xline.SetLineColor(ROOT.kBlue)
    xline.SetLineStyle(2)
    xline.SetLineWidth(2)
    xline.Draw("same")

    legend = ROOT.TLegend(0.1, 0.75, 0.50, 0.85)
    legend.SetBorderSize(0)
    legend.SetFillStyle(0)
    legend.AddEntry("",f"x-intercept = {x_intercept:.3e} m","n")
    legend.Draw("same")
    c.Update()
    c.SaveAs(f"{kv}kV_lambda_intercept.pdf")
    del c, graph, fit, xline, legend
```

Code 4: PyRoot code for linear fit to the Bremsstrahlung spectrum

```
    import ROOT
ROOT.gStyle.SetOptFit(1)

y = rescale_e(e_list)
f_scaled=rescale_f(f_data)
x = f_scaled[:,0]
sx = f_scaled[:,1]
sy = [0]*6

graph = ROOT.TGraphErrors(len(x))
for i in range(len(x)):
    graph.SetPoint(i, x[i], y[i])
    graph.SetPointError(i, sx[i], sy[i])

fit_func = ROOT.TF1("fit_func", "[0] + [1]*x")
b_guess = (y[-1] - y[0]) / (x[-1] - x[0])
a_guess = y[0] - b_guess * x[0]

fit_func.SetParameters(a_guess, b_guess)
print("Initial guesses:", a_guess, b_guess)

graph.Fit(fit_func,"S")
cal_parameters=[(fit_func.GetParameter(i),fit_func.GetParError(i)) for i in range(2)]

graph.SetMarkerStyle(20)
graph.SetMarkerColor(ROOT.kBlue)
graph.SetTitle("Experimental Planck Constant;f_{max};Energy (J)")

c = ROOT.TCanvas("c", "Line Fit", 800, 600)
graph.Draw("AP")
fit_func.Draw("same")
c.Update()
stats = graph.GetListOfFunctions().FindObject("stats")
if stats:
    stats.SetX1NDC(0.15)
    stats.SetY1NDC(0.75)
    stats.SetX2NDC(0.45)
    stats.SetY2NDC(0.87)
```

```
    stats.SetTextSize(0.03)
    c.Modified()

ROOT.gPad.SetTicks(1, 1)
ROOT.gPad.SetGrid(1, 1)
c.Update()
c.SaveAs("h_plot.pdf")
```

Code 5: PyROOT code for final linear fit to E vs. $F_max values$