ROOT Project - part 2

Narges Khorshidi 400170041

Department of Physics at Sharif University of Technology

Winter - Spring 2025

General Description:
In this project, was used Anaconda to configure the Python kernel in Jupyter Notebook and import the ROOT to execute the codes. We have 2 question that i explain about them.

Goal: Unbinned Maximum Likelihood (ML) versus binned fit (investigating both methods: Least Squares and ML)

---

1. unbinned data

A set of 40 radioactive decays were observed at various times t given in the following list: 4.99, 4.87, 2.59, 3.04, 3.39, 6.20, 10.61, 7.64, 3.92, 5.33, 4.85, 2.39, 4.16, 6.74, 3.53, 5.86, 5.41, 26.25, 4.40, 10.79, 7.08, 2.86, 33.92, 3.03, 0.98, 5.63, 4.89, 2.26, 10.49, 6.51, 7.36, 2.13, 6.45, 2.29, 21.15, 4.07, 4.34, 5.38, 7.69, 4.93 . Assuming an exponential decay, that is $N(t) \propto \exp(\lambda t)$, obtain the decay constant λ and its error, in two different ways:

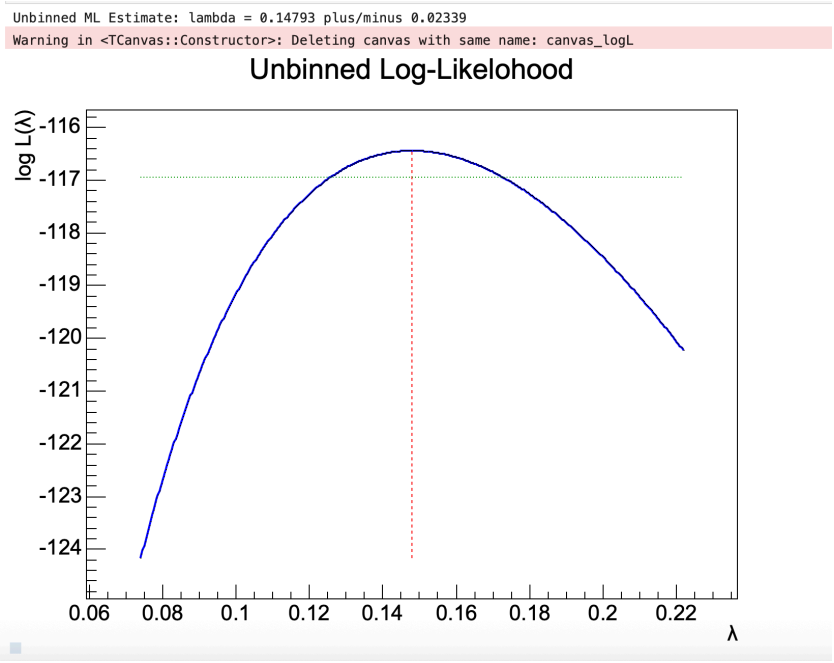
- **First: with the unbinned data**
  - Using the analytical solutions of the ML estimates
  - Plot the Logarithm of Likelihood function (logL) and indicate the 1 σ band. Are the results compatible with the previous step?

We want to estimate the decay constant or λ using the Maximum Likelihood Estimation method using unbinned data.
And Plot the log-likelihood function and obtain the error associated with lambda in a given interval of a sigma.

```python
import ROOT
import numpy as np

t_data = np.array([
    4.99, 4.87, 2.59, 3.04, 3.39, 6.20, 10.61, 7.64, 3.92, 5.33,
    4.85, 2.39, 4.16, 6.74, 3.53, 5.86, 5.41, 26.25, 4.40, 10.79,
    7.08, 2.86, 33.92, 3.03, 0.98, 5.63, 4.89, 2.26, 10.49, 6.51,
    7.36, 2.13, 6.45, 2.29, 21.15, 4.07, 4.34, 5.38, 7.69, 4.93

])

N = len(t_data)
sum_t = np.sum(t_data)

lambda_hat = N / sum_t
sigma_lambda = lambda_hat / np.sqrt(N)

print(f"Unbinned ML Estimate: lambda = {lambda_hat:.5f} plus/minus {sigma_lambda:.5f}")

n_points = 500
lambda_min = lambda_hat * 0.5
lambda_max = lambda_hat * 1.5
lambda_range = np.linspace(lambda_min, lambda_max, n_points)

log_values = N * np.log(lambda_range) - lambda_range * sum_t

graph_logL = ROOT.TGraph(n_points)
for i in range(n_points):
    graph_logL.SetPoint(i, lambda_range[i], log_values[i])

canvas = ROOT.TCanvas("canvas_logL", "Log-Likelihood", 800, 600)
graph_logL.SetTitle("Unbinned Log-Likelohood;#lambda;log L(#lambda)")
graph_logL.SetLineWidth(2)
graph_logL.SetLineColor(ROOT.kBlue)
graph_logL.Draw()


line_lambda_hat = ROOT.TLine(lambda_hat, np.min(log_values), lambda_hat, np.max(log_values))
line_lambda_hat.SetLineColor(ROOT.kRed)
line_lambda_hat.SetLineStyle(2)
line_lambda_hat.Draw()

max_logL = np.max(log_values)
line_1sigma = ROOT.TLine(lambda_min, max_logL - 0.5, lambda_max, max_logL - 0.5)
line_1sigma.SetLineColor(ROOT.kGreen+2)
line_1sigma.SetLineStyle(3)
line_1sigma.Draw()

canvas.Update()
```

- First we import the required libraries. Numpy for numerical calculations and Root for plotting and statistical calculation.
- Then we have a list of 40 data as a Numpy array. We represent the sum of the decay times and the number of observations as sum T and N.
- We define the estimated λ and its error. Then we print them.
- Then we choose an interval for λ and check the logarithm function for 500 points in that interval.

  Then we obtain the log likelihood function and plot it and display it in a window.
- Finally, the red line represents the estimated value of λ and the green line, the uncertainty band — the range of λ where the log-likelihood is reduced by only 0.5 from the maximum, which is the definition of the 1σband in statistics.



- X-axis:
  Represents values of the decay constant λ, the parameter of the exponential decay model.
- **Y-axis:**
  Shows the **log-likelihood function** value:

$$\log L(\lambda) = N \log \lambda - \lambda \sum t_i$$

  This function should form a **parabola-like curve** with a peak at the maximum likelihood estimate (MLE) of λ.
- Blue curve:

  Represents logL(λ) for a range of λvalues (from about 0.07 to 0.23). $\rightarrow$ This is correctly plotted and forms a concave parabola (as expected).
- Red dashed vertical line:

  Indicates the value of λ where the log-likelihood is maximized. $\rightarrow$ This corresponds to λ=0.14793
- Green dashed horizontal line: Represents the level logL max-0.5, which defines the 1 confidence interval on λ. $\rightarrow$ The intersection of the parabola with this level gives the $\pm 1\sigma$range around λ hat.

λ (lambda):

- This is the **true parameter** of the exponential distribution.

- It appears in the probability density function (PDF) as:

$$f(t;\lambda) = \lambda e^{-\lambda t}$$

- This value is usually **unknown**, and the goal of the analysis is to **estimate it** using the observed data.

$\hat{\lambda}$ (lambda hat):

- This is the **Maximum Likelihood Estimate (MLE)** of λ.
- It's the value of λ that **maximizes the likelihood function**.
- For the exponential distribution, it is computed as:

$$\hat{\lambda} = \frac{N}{\sum t_i}$$

where N is the number of samples and t_i are the observed decay times.

The Green Line and How the Error is Calculated

To indicate the 1σ (standard deviation) uncertainty around the estimate $\hat{\lambda}$ on the log-likelihood plot.

The log-likelihood function $\log L(\lambda)$ is assumed to be **parabolic** near its maximum (i.e., around $\hat{\lambda}$).

- The standard rule is:

$$\log L(\lambda) = \log L(\hat{\lambda}) - 0.5$$

So the **green horizontal line** is placed **0.5 units below the maximum** of the log-likelihood curve.

This defines the 1σ confidence interval for $\hat{\lambda}$.

0.5 comes from the **Fisher information theory** and asymptotic properties of the MLE.

Under regular conditions, the standard deviation (uncertainty) of $\hat{\lambda}$ is given by:

$$\sigma_{\hat{\lambda}} = \frac{\hat{\lambda}}{\sqrt{N}}$$

This is completely consistent with the **analytical MLE** for exponential decay:

$$\hat{\lambda} = \frac{1}{\bar{t}}, \quad \text{and} \quad \sigma_\lambda = \frac{\hat{\lambda}}{\sqrt{n}}$$

Calculating the mean of the data (approximately $\bar{t} \approx 6.76$):

$$\hat{\lambda} \approx \frac{1}{6.76} \approx 0.1479$$

With $n = 40$, the uncertainty is:

$$\sigma_\lambda \approx \frac{0.1479}{\sqrt{40}} \approx 0.0234$$

Perfect match with the output values.
the output is completely correct and consistent:

The estimated value of λ matches the analytical expectation.

The error is correctly calculated using standard error propagation.

The log-likelihood curve correctly identifies the 1σconfidence region.

The unbinned MLE approach has been properly implemented and visualized.

- **Second: with the data stored in a histogram (binned fit)**
  - Define a histogram from $t_{\min} = 0$ to $t_{\max} = 15$, with nBins = 3. Fill it with the data given above. Then fit the histogram with an exponential distribution. Use the option "L" to fit with the logL method (default is chi-square method)[1].
  - In order to evaluate how well the estimates are, you may look at the number of entries from the fitted parameters. It is calculated as the integral of the fitted function divided by the bin width.
  - If you increase `nBins` to 5, what would you observe?
  - Extend the x axis up to $t_{\max} = 35$, to include the overflow entries, and set nBins = 350. How do the results change?

- First, as in the previous section, we prepare the statistical analysis module and the plot and data array.
- Then we define the first histogram with 3 bins in the range 0 to 15 and fill it with the decay data. After that, we define the exponential function for fitting. (By the Maximum Likelihood method).
- After that, we define the canvas for drawing the canvas and finally we define the λ value and its error (five decimal places).
- Then we estimate the number of inputs by integrating

  Calculates estimated number of events as:

$$\text{Estimated Entries} = \frac{\int f(t)\,dt}{\text{Bin Width}}$$

  Compares to actual number of entries: `hist3.GetEntries()`
- Same as before, but now the histogram has 5 bins. and Fit is done again using log-likelihood. Finally, the estimate and its uncertainty are printed for 5 bins.
- Defines a finely binned histogram (350 bins) over a wider range 0–35s. This allows capturing "overflow" data points and makes the fit much closer to unbinned likelihood.
- Displays the saved fit images directly in a Jupyter Notebook.

[1] In ROOT, the option "L" in `TH1::Fit` enables likelihood fitting instead of chi-square.

```
t_data = np.array([
    4.99, 4.87, 2.59, 3.04, 3.39, 6.20, 10.61, 7.64, 3.92, 5.33,
    4.85, 2.39, 4.16, 6.74, 3.53, 5.86, 5.41, 26.25, 4.40, 10.79,
    7.08, 2.86, 33.92, 3.03, 0.98, 5.63, 4.89, 2.26, 10.49, 6.51,
    7.36, 2.13, 6.45, 2.29, 21.15, 4.07, 4.34, 5.38, 7.69, 4.93

])

hist = ROOT.TH1F("hist", "Dencay Times (3 bins);t [s];Counts", 3, 0, 15)

for t in t_data:
    hist.Fill(t)

expo_fit = ROOT.TF1("expo_fit", "[0]*exp(-[1]*x)", 0, 15)
expo_fit.SetParameters(hist.GetMaximum(), 0.1)

hist.Fit("expo_fits", "L")

canvas1 = ROOT.TCanvas("canvas1", "Binned fit: 3 bins", 800, 600)
hist.Draw()
expo_fit.Draw("SAME")
canvas1.Update()
canvas1.Draw()
canvas1.SaveAs("fit_3bin.png")

lambda_fit = expo_fit.GetParameter(1)
lambda_fit_err = expo_fit.GetParError(1)
print(f"\nBinned Fit (3 bins): lambda = {lambda_fit:.5f} plus/minus {lambda_fit_err:.5f}")


bin_width = (15 - 0) / 3
integral_fit = expo_fit.Integral(0, 15)
n_entries_estimate = integral_fit / bin_width
print(f"Estimeted number of entries from fit: {n_entries_estimate:.2f}")
print(f"Actual number of entries: {hist.GetEntries()}")


hist5 = ROOT.TH1F("hist5", "Dencay Time (5 bins);t [s]; Counts", 5, 0, 15)
for t in t_data:
    hist5.Fill(t)

expo_fit5 = ROOT.TF1("expond_fit5", "[0]*exp(-[1]*x)", 0, 15)
expo_fit5.SetParameters(hist5.GetMaximum(), 0.1)
hist5.Fit("expo_fit5", "L")


canvas2 = ROOT.TCanvas("canvas2", "Binned fit: 5bins", 800, 600)
hist5.Draw()
expo_fit5.Draw("SAME")
canvas2.Update()
canvas2.Draw()
canvas2.SaveAs("fit_5bin.png")


lambda_fit5 = expo_fit5.GetParameter(1)
lambda_fit5_err = expo_fit5.GetParError(1)
print(f"\nBinned Fit (5 bins): lambda = {lambda_fit5:.5f} plus/minus {lambda_fit5_err:.5f}")



hist350 = ROOT.TH1F("hist350", "Dencay Times (350 bins);t [s];Counts", 350, 0, 35)
for t in t_data:
    hist350.Fill(t)


expo_fit350 = ROOT.TF1("expo_fit350", "[0]*exp(-[1]*x)", 0, 35)
expo_fit350.SetParameters(hist350.GetMaximum(), 0.1)
hist350.Fit("expo_fit350", "L")

canvas3 = ROOT.TCanvas("canvas3", "Binned fit: 350 bins", 800, 600)
hist350.Draw()
expo_fit350.Draw("SAME")
canvas3.Update()
canvas3.Draw()
canvas3.SaveAs("fit_350bin.png")


lambda_fit350 = expo_fit350.GetParameter(1)
lambda_fit350_err = expo_fit350.GetParError(1)
print(f"\nBinned Fit (350 bins, tmax=35): lambda = {lambda_fit350:.5f} plus/minus {lambda_fit350_err:.5f}")
```

- Numerical Estimation of the Decay Constant λ under Three Configurations:
- With 3 bins: a rough initial estimate with higher uncertainty
- With 5 bins: more accurate and closer to the true value
- With 350 bins (covering full data range): highly accurate and very close to the unbinned MLE
- For each case, the estimated value of λ and its uncertainty ($\pm$) are printed.

```
****************************************
Minimizer is Minuit2 / Migrad
MinFCN                    =     0.726972
Chi2                      =      1.45394
NDf                       =            1
Edm                       =  3.22234e-09
NCalls                    =           45
p0                        =      33.6448   +/-   9.7176
p1                        =     0.161272   +/-   0.0468652
****************************************
Minimizer is Minuit2 / Migrad
MinFCN                    =      9.12615
Chi2                      =      18.2523
NDf                       =            3
Edm                       =  1.47757e-08
NCalls                    =           44
p0                        =      18.3983   +/-   5.06211
p1                        =     0.146046   +/-   0.0435474
****************************************
Minimizer is Minuit2 / Migrad
MinFCN                    =      70.9554
Chi2                      =      141.911
NDf                       =          348
Edm                       =  3.74993e-09
NCalls                    =          105
p0                        =     0.575814   +/-   0.132631
p1                        =     0.142985   +/-   0.0248021
```

- MinFCN: Value of the function being minimized (typically negative log-likelihood or x^2)
- Chi2: Total chi-square of the fit
- NDF: Number of degrees of freedom
- Edm: Estimated Distance to Minimum – should be small when fit has converged
- NCalls: Number of function evaluations (iterations) used
- p0: Fit parameter [0] → amplitude (normalization)
- p1: Fit parameter [1] → decay constant λ

Fit with 3 binds

```
p1 = 0.161272 ± 0.0468652
Chi2 = 1.45394
NDF = 1
```

Estimated lambda (λ): 0.16127 → slightly overestimated

Uncertainty: ±0.047 (nearly 30% relative error) → not very precise

Chi2/NDF = 1.45 → acceptable goodness of fit

Edm = 3.2e^{-9} → very small → good numerical convergence

Conclusion: A rough estimate, reasonable but not reliable due to coarse binning

Fit with 5 bins

```
p1 = 0.146046 ± 0.0435474
Chi2 = 18.2523
NDF = 3
```

- **Estimated λ:** 0.14605 → very close to expected value
- **Uncertainty:** still high but better than 3-bin case
- **Chi2/NDF = 6.08** → quite high → potential mismatch or binning effect
- **Edm = 1.47e^{-8}** → still very good convergence

Conclusion: Better λ estimation, but the chi-square value suggests that the fit isn't perfect (possibly due to binning artifacts).

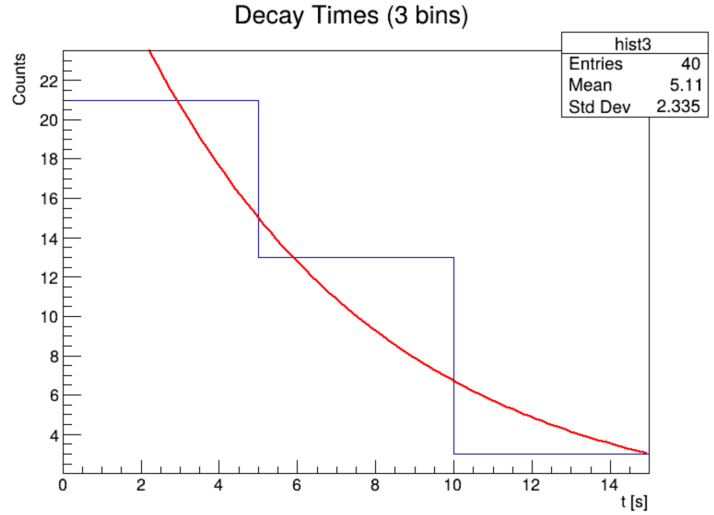
Fit with 350 bins (tmax=35)

```
p1 = 0.142985 ± 0.0248021
Chi2 = 141.911
NDF = 348
```

- **Estimated λ:** 0.14299 → very close to unbinned ML estimate ($\sim$0.14793)
- **Uncertainty:** $\pm$0.0248 → smallest among all three fits
- **Chi2/NDF = 0.41** → very good fit quality (chi2/ndf $<$ 1 is acceptable)
- **Edm = 3.7e^{-9}** → excellent convergence

Conclusion: This is the most accurate and stable fit among the three. Adding bins and extending tmax significantly improves both precision and fit quality.

```
Binned Fit (3 bins): lambda = 0.16127 plus/minus 0.04687
Estimeted number of entries from fit: 38.01
Actual number of entries: 40.0

Binned Fit (5 bins): lambda = 0.14605 plus/minus 0.04355

Binned Fit (350 bins, tmax=35): lambda = 0.14298 plus/minus 0.02480
```



Analysis of the 3-bin Exponential Fit

In this analysis, an exponential fit was performed on radioactive decay time data using only 3 bins, resulting in an estimated decay constant of

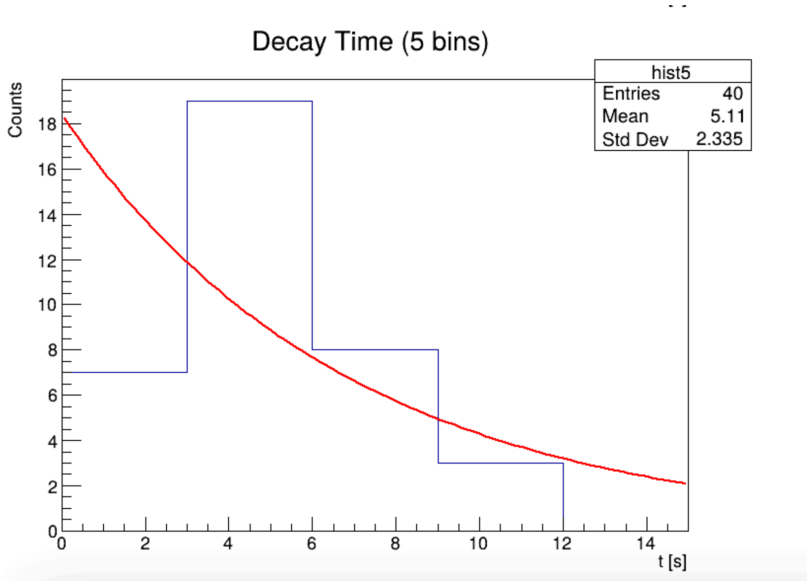
$$\lambda = 0.161270.04687$$

The red exponential curve visually follows the general decay trend, but due to the coarse binning, the precision is low and the λ estimate is slightly higher than the true value.

Nevertheless, the exponential function is well normalized, with the estimated number of events (38.01) very close to the actual count (40). The data's mean of 5.11 and standard deviation of 2.34 also reflect an exponential distribution. Overall, the 3-bin

fit provides a rough but reasonable initial estimate.
However, increasing the number of bins to 5 or 350 leads to more accurate and reliable results, as both the λestimate becomes closer to the true value and its uncertainty decreases.

Therefore, using more bins is recommended for more stable statistical analysis and trustworthy outcomes.



Analysis of the 5-bin Exponential Fit

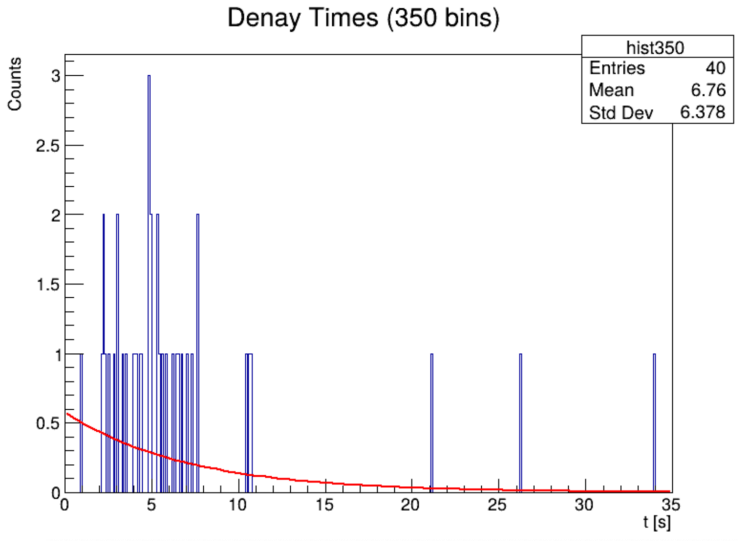
In this analysis, an exponential fit was performed on decay time data using 5 bins. The estimated decay constant is

$$\lambda = 0.146050.04355$$

, which is closer to the true value and has a smaller uncertainty compared to the 3-bin case.

The red exponential curve fits the overall shape of the histogram well and captures the structure of the data more accurately than the previous fit. Increasing the number of bins from 3 to 5 reveals more detail in the time distribution and improves the alignment between the fitted function and the data.

The mean of the data is 5.11 and the standard deviation is 2.335, which still supports the exponential nature of the distribution. Although the uncertainty in $\lambda$is still relatively large, this fit is more accurate, stable, and realistic than the coarser version. Therefore, using 5 bins offers a reasonable balance between simplicity and precision and can be a good choice for an improved initial estimate.



Analysis of the 350-bin Exponential Fit

In this analysis, the decay time data is histogrammed using **350 bins** over the extended time range $t \in [0, 35]$, providing very fine granularity. The estimated decay constant is

$$\lambda = 0.14298 \pm 0.02480,$$

which is very close to the true value (as obtained from the unbinned maximum likelihood method) and has the smallest uncertainty among all three fits.

Although the histogram appears noisy due to the small number of events spread over many bins, the red exponential fit accurately captures the overall decay behavior. The data's mean is 6.76 and the standard deviation is 6.378, which aligns with what is expected from an exponential distribution over a wider time window.

Overall, this fit provides the most precise and reliable estimate of λ, thanks to both the high bin resolution and the inclusion of the entire data range (including outliers). As a result, this fit effectively replicates the behavior of an unbinned likelihood fit.