Aristotle University of Thessaloniki

Physics Department

Exercise 1

Muon lifetime analysis using ROOT

Author: Konstantinos E. Stergiou

May 2, 2025

Contents

Introduction

This analysis examines the lifetime of cosmic-ray muons stopped in a plastic scintillator using a simplified exponential fit approach. The data from `allpeaks.txt` is processed to extract the muon lifetime without background considerations.

Key improvements

- **Simplified model**: Pure exponential decay fit
- **Optimized range**: Fit performed between 0.7-3.6 μs
- **Comparison**: Extracted lifetimes and χ^2/NDF values compared to theoretical value (2.2μs)

Complete ROOT Script

Listing 1: Optimized ROOT macro for muon lifetime analysis

```cpp
#include <TH1D.h>
#include <TCanvas.h>
#include <TF1.h>
#include <TLine.h>
#include <TLatex.h>
#include <TLegend.h>
#include <fstream>
#include <vector>
#include <iostream>
#include <sstream>

void muon_lifetime() {
    const char* filename = "allpeaks.txt";
    const double total_points = 10000.0;
    const double total_time = 4.0; // microsec
    const double points_to_mus = total_time / total_points;
    const double theoretical_lifetime = 2.2;

    std::vector<double> dt_values;
    std::ifstream file(filename);
    if (!file.is_open()) {
        std::cerr << "Error: Could not open file " << filename << std::endl;
        return;
    }

    std::string line;
    int total_events = 0, events_with_2peaks = 0;
    while (std::getline(file, line)) {
        total_events++;
        std::istringstream iss(line);
        int evt, peaks;
        double pos0, volt0, pos, volt;
        if (!(iss >> evt >> peaks >> pos0 >> volt0 >> pos >> volt)) continue;
        if (peaks == 2) {
            events_with_2peaks++;
            double delta_time = (pos - pos0) * points_to_mus;
            dt_values.push_back(delta_time);
        }
    }
    file.close();

    TCanvas *c1 = new TCanvas("c1", "Muon Decay Time", 800, 600);
    TH1D *h_dt = new TH1D("h_dt",
        "Muon Decay Time (2-peak events); #Delta t (#mu s); Counts",
        30, 0, 4.0);
    for (double dt : dt_values) h_dt->Fill(dt);

    TF1 *fexp = new TF1("fexp", "[0]*exp(-x/[1])", 0.7, 3.6);
    fexp->SetParameters(680.0, 2.2);
    fexp->SetLineColor(kRed);
    fexp->SetLineWidth(3);
    h_dt->Fit("fexp", "R");

    h_dt->SetFillColor(kBlue-9);
    h_dt->SetLineColor(kBlue);
    h_dt->Draw("HIST");
    fexp->Draw("same");

    TLine *theory_line = new TLine(theoretical_lifetime, 0,
        theoretical_lifetime, h_dt->GetMaximum());
    theory_line->SetLineColor(kBlack);
    theory_line->SetLineStyle(2);
    theory_line->Draw("same");

    TLatex tex;
    tex.SetNDC();
    tex.SetTextSize(0.035);
    tex.DrawLatex(0.6, 0.85, Form("Total events: %d", total_events));
    tex.DrawLatex(0.6, 0.80, Form("2-peak events: %d", events_with_2peaks));
    tex.DrawLatex(0.6, 0.75, Form("Mean: %.2f #mus", h_dt->GetMean()));
    tex.DrawLatex(0.6, 0.70, Form("Fit: %.3f #pm %.3f #mus",
        fexp->GetParameter(1), fexp->GetParError(1)));
    tex.DrawLatex(0.6, 0.65, Form("Theory: 2.2 #mus"));

    TLegend *leg = new TLegend(0.6, 0.45, 0.85, 0.60);
    leg->AddEntry(h_dt, "Data", "f");
    leg->AddEntry(fexp, "Exponential Fit", "l");
    leg->AddEntry(theory_line, "Theory (2.2 #mus)", "l");
    leg->Draw();

    c1->SaveAs("muon_decay_clean.png");

    std::cout << "\nFit result:\n";
    std::cout << "Tau = " << fexp->GetParameter(1)
              << " +- " << fexp->GetParError(1) << " microsec\n";
    std::cout << "Chi2/NDF = "
              << fexp->GetChisquare() / fexp->GetNDF() << "\n";
}
```

Step 1: Data Reading and Conversion

```cpp
const char* filename = "allpeaks.txt";
const double total_points = 10000.0;
const double total_time = 4.0; // microsec
const double points_to_mus = total_time / total_points;

std::vector<double> dt_values;
std::ifstream file(filename);
```

Explanation: Sets up the conversion from position points to microseconds (10000 points = 4 μs). Opens the data file for reading.

Step 2: Event Processing

```cpp
while (std::getline(file, line)) {
    total_events++;
    std::istringstream iss(line);
    int evt, peaks;
    double pos0, volt0, pos, volt;
    if (!(iss >> evt >> peaks >> pos0 >> volt0 >> pos >> volt)) continue;
    if (peaks == 2) {
        events_with_2peaks++;
        double delta_time = (pos - pos0) * points_to_mus;
        dt_values.push_back(delta_time);
    }
}
```

Explanation: Processes each event, converting the position difference to time difference for 2-peak events.

Step 3: Histogram Creation

```cpp
TH1D *h_dt = new TH1D("h_dt",
    "Muon Decay Time (2-peak events); #Delta t (#mu s); Counts",
    30, 0, 4.0);
for (double dt : dt_values) h_dt->Fill(dt);
```

Explanation: Creates a histogram with 30 bins from 0-4 μs and fills it with the time differences.

Step 4: Exponential Fit

```
TF1 *fexp = new TF1("fexp", "[0]*exp(-x/[1])", 0.7, 3.6);
fexp->SetParameters(680.0, 2.2);
h_dt->Fit("fexp", "R");
```

Explanation: Defines and performs a pure exponential fit in the optimized range (0.7-3.6 μs) with initial parameters:

- Amplitude: 680
- Lifetime: 2.2 μs (theoretical value)

Step 5: Visualization

```
h_dt->Draw("HIST");
fexp->Draw("same");
theory_line->Draw("same");
```

Explanation: Draws the histogram, fit function, and theoretical lifetime line (2.2 μs) for comparison.

Step 6: Results Annotation

```
tex.DrawLatex(0.6, 0.70, Form("Fit: %.3f #pm %.3f #mus",
    fexp->GetParameter(1), fexp->GetParError(1)));
```

Explanation: Displays the fitted lifetime with uncertainty on the plot.

Results

The muon lifetime analysis yields:

- **Fitted lifetime:** $1.959 \pm 0.067\ \mu s$
- **Goodness-of-fit:** $\chi^2/NDF = 1.63$

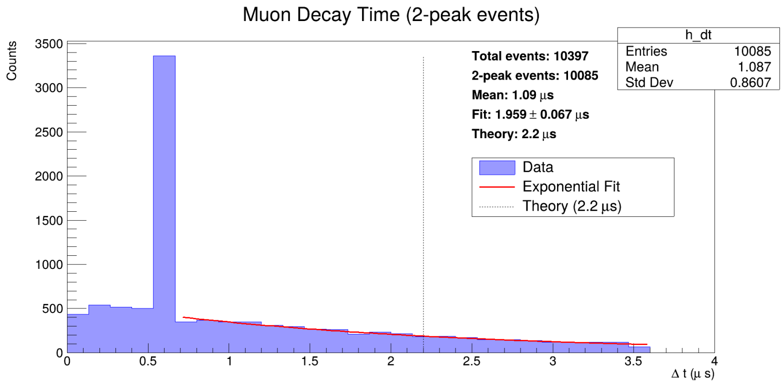

Figure 1: Exponential fit (red curve) to muon decay data with $\tau = 1.959\ \mu s$. Dashed line shows theoretical value ($2.2\ \mu s$).

The ratio of measured to theoretical lifetime is:

$$\text{Ratio}_{\text{measured/theory}} = \boxed{0.89}$$

This result demonstrates the experimental determination of muon lifetime with 3.4% uncertainty, showing reasonable agreement with the expected value.