PHYS 339 Lab 3 - Counting statistics

Robert Turner

2025-01-28

0 Experimental setup

The setup for this experiment is straightforward. You will be provided with:

- A Geiger counter
- A BNC to terminal adapter
- A sealed radioactive source, which produces γ rays.
- A holder for your source
- Millimetre-grid graph paper

1 Viewing the signal

View the output signal from the Geiger counter on your oscilloscope.

0. What is the peak voltage of the Geiger counter ouptut? What should it be in order to be successfully read on a digital pin? RYR. If this voltage is not in the correct range, inform a TA or the lab technician.
1. What is the pulse width of a single click? RYR.
2. Does the click frequency change if the orientation of the source changes?

2 Arduino interface

The Arduino interface is extremely straightforward:

1. Connect one jumper wire to each of the binding posts of your BNC adapter.
2. Connect the wire from the black post to Arduino GND.

3. Connect the wire from the red post to an Arduino interrupt-enabled pin (either pin 2 or 3).
4. Connect the adapter to your Geiger counter.
5. REDO THIS TO FOLLOW DOMINIC"S SEQUENCE.
6. Write an Arduino program to record the number of clicks in a given interval of time.
7. Write an Arduino program to record the number of Geiger clicks and the time of each click in elapsed microseconds since the start of the program.
    - Attach an interrupt to the Geiger read pin.
    - Your serial output should display the running count of each click, along with the elapsed time in microseconds, on a single line.

3 Coding

There are three goals for your code. Save each set of code separately so you can easily demonstrate it.

0. Basic functionality for recording the number of clicks per unit time:
    - Write Arduino code to record the number of clicks per given unit of time (this will be hard-coded). Transmit this continuously to the Serial port.
    - Write Python code to receive these data, save them to a file, and histogram them.
      Once you start recording data in this lab, you should never delete them.
      As your data accumulate throughout the exercises, you will use its increasing quantity to continually refine your analyses. Therefore, you should devise a scheme for naming your data files that indicates:
        - the date and time of acquisition (to one-second resolution),
        - the format of data recorded, whether clicks-per-interval, or interval-between-clicks, and
        - the units used.
1. Basic functionality for recording the time interval between clicks:
    - Write Arduino code to record the time interval between clicks. Transmit this continuously to the Serial port.
    - Write Python code to receive these data, save them to a file, and to histogram them.

2. Sophisticated code for requesting and transmitting data:
    - Write Arduino code to accept Serial input specifying whether to record the data as clicks per interval or interval between clicks, and how many data points to report.
      The Arduino should then wait for (and fulfill ;)) further data requests.
    - Write Python code to request the desired data and process it correctly, with real-time display of incoming data.
    - You may want to write a separate Python script for doing your data analysis.

4 Python notes

- You should use the `with open` context manager construct to write to your data file in `append` mode. This will enable you to save your data one line at a time, and in the event of a program crash you will only lose the last data point.
- Pro tip: Use the Python `time` module to automatically generate a timestamp for each of your output file names. Even better, put the distance between the source and the Geiger-Muller tube in the filename as well.

5 Lab strategy

All calculations and data processing should be done in Python. Again, you should save ALL of your Geiger data for later analysis. All of these data are valuable, so don't delete them and re-take "better" data later. Just keep creating data files.