PhyPiDAQ: Datenaufnahme do-it-yourself Teilchendetektor

Übersetzt aus der englischen Version mit Hilfe von DeepL.com (kostenlose Version)

Der Nachweis von Radioaktivität, die entweder durch künstliche Quellen oder als Teil der natürlichen Umwelt entsteht, wie K-40 oder Radon aus dem Erdinneren, ist ein faszinierendes Forschungsgebiet, das heute mit preiswerten und einfachen Detektoren zugänglich geworden ist. Es gibt kommerzielle Angebote, aber auch eine Reihe von Vorschlägen für Do-it-yourself-Projekte. Schon die Soundkarte eines Standard-PCs ist geeignet die Signale aufzuzeichnen, die dann mit einem Soundkarten-Oszilloskop visualisiert werden können.

Der CERN DIY Teilchendetektor

Ein relativ neuer Vorschlag für einen einfachen, kostengünstig und leicht zu bauenden Teilchendetektor basierend auf vier PIN-Photodioden vom Typ BPW34 ist der CERN DIY_particle_detector. Der zweistufige Verstärker mit einem Operationsverstärker mit hoher Bandbreite erzeugt große Signale von mehreren Hundert mV mit einer Breite von etwa 100 µs. Solche Signale können sogar mit einer Soundkarte aufgenommen werden.

Die Verfügbarkeit der erforderlichen Bauteile, einschließlich der Leiterplatine, und die großen und langen Signale machen diesen Detektor zu einer idealen Wahl für Projekte mit Oberstufenschülern. Neben dem Bau des Geräts werden Erfahrungen in der Datenerfassung gewonnen und die Analyse eines faszinierenden Phänomens ermöglicht, das nicht direkt mit den menschlichen Sinnen erfassbar ist.

Für das hier vorgestellte Projekt wird die “Elektron-Variante” verwendet, die Elektronen mit Hilfe von vier PIN-Photodioden als Sensoren nachweist. Nachweisbare Elektronen können auch durch Gamma-Strahlung entweder in der Nähe oder in der PIN-Diode erzeugt werden. Wenn man den Detektor in eine Blechdose einbaut, werden Elektronen aus der Umgebung abgeschirmt, so dass nur noch Gamma-Strahlung aus der Umgebung registriert wird.

Datenerfassung und Auswertung mit *PhyPiDAQ*

PhyPiDAQ enthält mehrere Module zur Aufzeichnung, Visualisierung und Analyse von Daten. Ein in diesem Kontext interessanter und wichtiger Aspekt ist die Zufälligkeit des Auftretens der Signale von Teilchen im Detektor. Da die Wahrscheinlichkeit für das Auftreten eines Signals zeitlich konstant ist, wird die Anzahl der in einem bestimmten Zeitintervall beobachteten Ereignisse durch die Poisson-Statistik beschrieben. Charakteristisch für einen solchen Prozess ist außerdem, dass die Zeit zwischen den zufällig auftretenden Ereignissen einer Exponentialverteilung folgt.

Der exponentielle Zerfall einer Probe von radioaktiven Kernen ist ein weiterer wichtiger Aspekt, der untersucht werden kann, wenn eine Probe von kurzlebigen Kernen zur Verfügung steht. In der Natur ist eine solche Quelle das aus dem Erdinneren als Folge radioaktiver Prozesse austretende Edelgases Radon, dessen Zerfallsprodukte auf einem elektrisch geladenen Luftballon gesammelt und anschließend nachgewiesen werden können.

Relevante Module des *PhyPiDAQ*-Pakets sind

- `phypidaq\soundcardOsci` mit zwei Klassen zur Aufzeichnung, Auswahl und Anzeige von Wellenformen von einer PC-Soundkarte,
- phypidaq/DisplayPoissonEvent”, eine Klasse zur Anzeige einzelner Ereignisses eines Poisson-Prozesses.

Gebrauchsfertige Skripte veranschaulichen die Verwendung dieser Klassen:

- `examples/oscilloscope/run_scOsci.py`, um von einer Soundkarte ausgelesene Wellenform anzuzeigen.
- `examples/scGammaDetector.py` zur Visualisierung des Zeitpunkts des Auftretens eines großen Signals und der zugehörigen Wellenformdaten. Außerdem wird eine Ratenhistorie angezeigt. Das Skript bietet auch die die Möglichkeit, die Ereigniszeiten und Signalhöhen für eine Offline-Analyse in einer Datei zu speichern.

Untersuchungen von Poisson-Prozessen ermöglichen die Python-Skripte

- `examples/poissonFlash.py` zur Erzeugung, Visualisierung und Speicherung von Daten eines simulierten Poisson-Prozesses.
- `examples/poissonLED.py` zur Erzeugung zufälliger Blitze einer LED. Eine Photodiode, die dem Licht der der LED ausgesetzt ist, erzeugt Signale, die einem Detektor für Gammastrahlen entsprechen. Für diese Anwendung wird ein Raspberry Pi mit an die GPIO-Leiste angeschlossener LED mit Vorwiderstand benötigt.

Anmerkungen zur Installation Die genannten *Python*-Skripte benötigen einen kleinen Teil der im Paket *phypidaq*enthaltenen Software. Wenn die Installation des gesamten Pakets nicht gewünscht wird, wird weiter unten auch eine vereinfachte Möglichkeit beschrieben. Idealerweise sollte allerdings das gesamte *PhyPiDAQ*-Paket herunter geladen und die Python-Bibliotheken installiert werden:

```
# get git repository with PhyPiDAQ code
cd <workdir>
git clone https://github.com/PhyPiDAQ/PhyPiDAQ

# install  phypidaq modules
cd PhyPiDAQ
python -m pip install .
```

Nach diesem Schritt können die oben genannten *Python*-Programme in den Verzeichnissen *examples/* und *examples/oscilloscope/* ausgeführt werden.

Anmerkung: Diese Befehlsfolge funktioniert auch auf der Konsole (“Terminal”) von Rechnern mit MS Windows 10 oder 11, wenn dort *Python* installiert ist.

Vereinfachte Installation Die Python-Programme *scGammaDetector.py* und *run_scOsci.py* können auch ohne den Download des kompletten *PhyPiDAQ*-Pakets ausgeführt werden. Wenn eine aktuelle Python-Version verfügbar ist, können die Bibliotheken auch direkt aus dem *github*-Repository installiert werden:

```
python -m pip install git+https://github.com/PhyPiDAQ/PhyPiDAQ
```

Die Python-Programme können dann über die Links run_scOsci.py und scGammaDetector.py in ein Arbeitsverzeichnis geladen und ausgeführt werden.

Beispiele Eine typische Wellenform, die mit *python3 scGammaDetector.py -o* aufgezeichnet wurde, ist in der folgenden Abbildung dargestellt. Das Signal ist oberhalb des Rauschpegels von ca. 3500 ADC-Counts deutlich sichtbar. Es ist groß genug, um direkt an einen Kopfhörer angeschlossen zu werden, so dass die Signalklicks auch akustisch wahrgenommen werden können.

Bitte beachten, dass manche Soundkarten den Puls invertierten. Der Originalpuls ist, wie hier gezeigt, negativ mit einem deutlich sichtbaren Überschwinger ins Positive.

Das Skript bietet eine Reihe von Befehlszeilenoptionen, um die visuelle Ausgabe zu steuern und die Speicherung der Ergebnisse in einer Datei zu ermöglichen, oder um die Parameter der Soundkarte und insbesondere die Triggeroptionen einzustellen. Die Ausgabe des Befehls `./scGammaDetector -h` gibt einen Überblick über alle Optionen:

```
usage: scGammaDetector.py [-h] [-q] [-o] [-n] [-f FILE] [-w] [-t TIME] [-z SAMPLESIZE]
                          [-s {48000,96000,192000,44100}] [-c {1,2}] [-l TRGLEVEL] [--trgfalling] [-d]
                          [--overshoot OVERSHOOT] [-r RANGE] [-i INTERVAL]

Read waveforms from soundcard and display and optionally store data

options:
  -h, --help            show this help message and exit
  -q, --quiet           no status output to terminal
  -o, --oscilloscope    oscilloscope display
  -n, --noeventdisplay  deactivate event display
  -f FILE, --file FILE  base filename to store results
  -w, --write_raw       write raw wave forms
  -t TIME, --time TIME  run time in seconds
  -z SAMPLESIZE, --samplesize SAMPLESIZE
                        number of samples per read
  -s {48000,96000,192000,44100}, --samplingrate {48000,96000,192000,44100}
                        sampling rate
  -c {1,2}, --channels {1,2}
                        number of channels
  -l TRGLEVEL, --trglevel TRGLEVEL
```

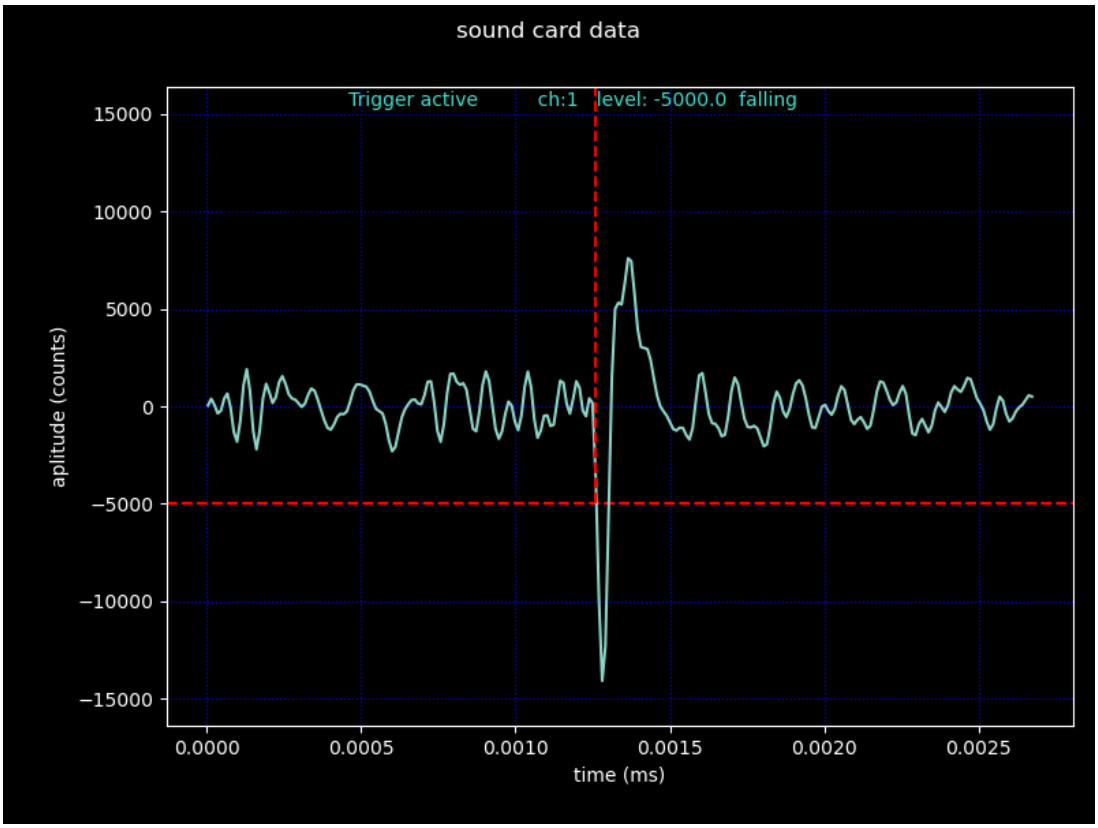

Abbildung 1: Typische Wellenform des DIY CERN Teilchendetektors, aufgenommen mit einer einfachen USB-Soundkarte mit 16 Bit Auflösung und einer Abtastrate von 96000/s. Beachten Sie, dass auf der y-Achse nur einen Bereich von $\pm 2^{14}$ angezeigt wird.

```
                        level of trigger
  --trgfalling          trigger falling edge
  -d, --trgdeactivate   deactivate triggering
  --overshoot OVERSHOOT
                        minimum overshoot fraction
  -r RANGE, --range RANGE
                        display range
  -i INTERVAL, --interval INTERVAL
                        time bin for rate display
```

Da die Signalrate in normalen Umgebungen ohne radioaktive Quelle sehr gering ist, sollte die Triggerschwelle knapp über dem Rauschpegel eingestellt werden, so dass einige Rauschimpulse sichtbar werden. Es ist auch ratsam, die Option `-o` zu verwenden, um die Oszilloskopansicht einzuschalten. Beachten Sie, dass der Signalpegel von den Einstellungen der Soundkarte abhängt. Verwenden Sie zur Auswahl des Standard-Eingabegeräts das entsprechende Werkzeug Ihres PC-Betriebssystems, insbesondere auch um die Lautstärke einzustellen.
Zur Registrietung des oben gezeigten Signals wurden die Trigger-Optionen `--trgfalling -l -5000`verwendet.

Um sich mit der Software vertraut zu machen, insbesondere mit der Auswahl der Daten durch Einstellung geeigneter Triggerbedingungen, ist es nützlich, ein Mikrofonsignal als Eingangsquelle zu verwenden. Zunächst wird nur das Oszilloskop mit einem sehr niedrigen Triggerpegel gestartet:
> `python3 scGammaDetector.py -n -o -l 100`

Damit werden die Rohdaten der Soundkarte auf dem Oszilloskop-Display angezeigt. Erzeugen Sie nun ein lautes Geräusch, z.B. indem Sie in die Hände klatschen oder mit den Fingern schnipsen, und Sie werden einige kurze Signale sehen, die weit über dem durchschnittlichen Geräuschpegel liegen. Merken Sie sich den typischen Signalpegel des Untergrundrauschens.
Beenden Sie das Programm nun durch Eingabe von “E” in der Befehlszeile oder durch Klicken auf die Schaltfläche “End” in der grafischen Benutzeroberfläche. Starten Sie es dann erneut mit einem höheren Triggerpegel, wobei Sie diesmal die Ereignisanzeige aktiviert lassen:
> `python3 scGammaDetector.py -l 1500`

Sie sollten nun keine Signale sehen - es sei denn, Sie erzeugen ein lautes Geräusch, dessen Verlauf dann in der Ereignisanzeige angezeigt wird.
Anmerkung: die gleichzeitige animierte Anzeige von Ereignisdarstellung und Oszillogramm funktioniert unter MS Windows nicht stabil; unter Linux können problemlos beide Anzeigen aktiviert werden: `python3 scGammaDetector.py -o -l 1500`

Das Aufspüren der sehr kleinen Signale des DIY-Teilchendetektors des CERN funktioniert auf genau dieselbe Weise. Verbinden Sie den Ausgang des Detektors mit dem Mikrofoneingang Ihrer Soundkarte und wiederholen Sie das eben beschriebene Verfahren, um den richtigen Triggerpegel zur Unterscheidung der echten Signalen nachgewiesener Teilchen vom Untergrundrauschen zu trennen. Beachten Sie, dass der Signalpegel von den Einstellungen Ihrer Soundkarte abhängt, vor allem von der Lautstärke. Wenn möglich, erhöhen Sie die Abtastrate auf den höchstmöglichen Wert, der von Ihrer Soundkarte unterstützt wird - typische Werte sind 44100, 48000, 96000 oder 192000 Samples/s. Ziehen Sie auch in Betracht, die Samplegröße einer einzelnen Aufnahme mit der Option `-z<n>` anzupassen - 256 oder 512 sind optimale Einstellungen für die kurzen Pulse des Teilchendetektors, aber einige Soundkartentreiber unterstützen nur eine minimalen Wert von 1024. Wenn die Samplegröße zu groß ist, könnten mehr als ein Signal in dem Sample enthalten sein, aber nur das erste würde gezählt.

Es ist offensichtlich, dass der Triggerpegel einen starken Einfluss auf die aufgezeichnete Signalrate hat. Ist er zu niedrig, wird der größte Teil der echten Signale erfasst, aber es sind auch viele Rauschimpulse vorhanden. Ist der Triggerpegel zu hoch, werden die meisten Rauschimpulse unterdrückt, aber es gehen auch Signalimpulse verloren. Hier gibt es keinen Ausweg - die Nachweiseffizienz und die Untergrundunterdrückung können nicht beide 100 % betragen! Wenn eine absolute Rate bestimmt werden soll, müssen Korrekturen für die schwellenabhängige Signaleffizienz und Untergrundkontamination vorgenommen werden.

Die nachstehende Abbildung zeigt die Ausgabe des Teilchendetektors. Ein blinkender Kreis zeigt das Auftreten eines ausgelösten Ereignisses an, und die entsprechende Wellenform mit 100 Abtastpunkten um den Auslösezeitpunkt herum wird ebenfalls angezeigt. Es wird auch ein Ratenverlauf angezeigt; die Bin-Breite in Sekunden kann mit der Option `--interval <n>` eingestellt werden.

Die Bestimmung des Untergrundniveaus und der Signaleffizienz mit kleinen Unsicherheiten ist nicht immer leicht zu

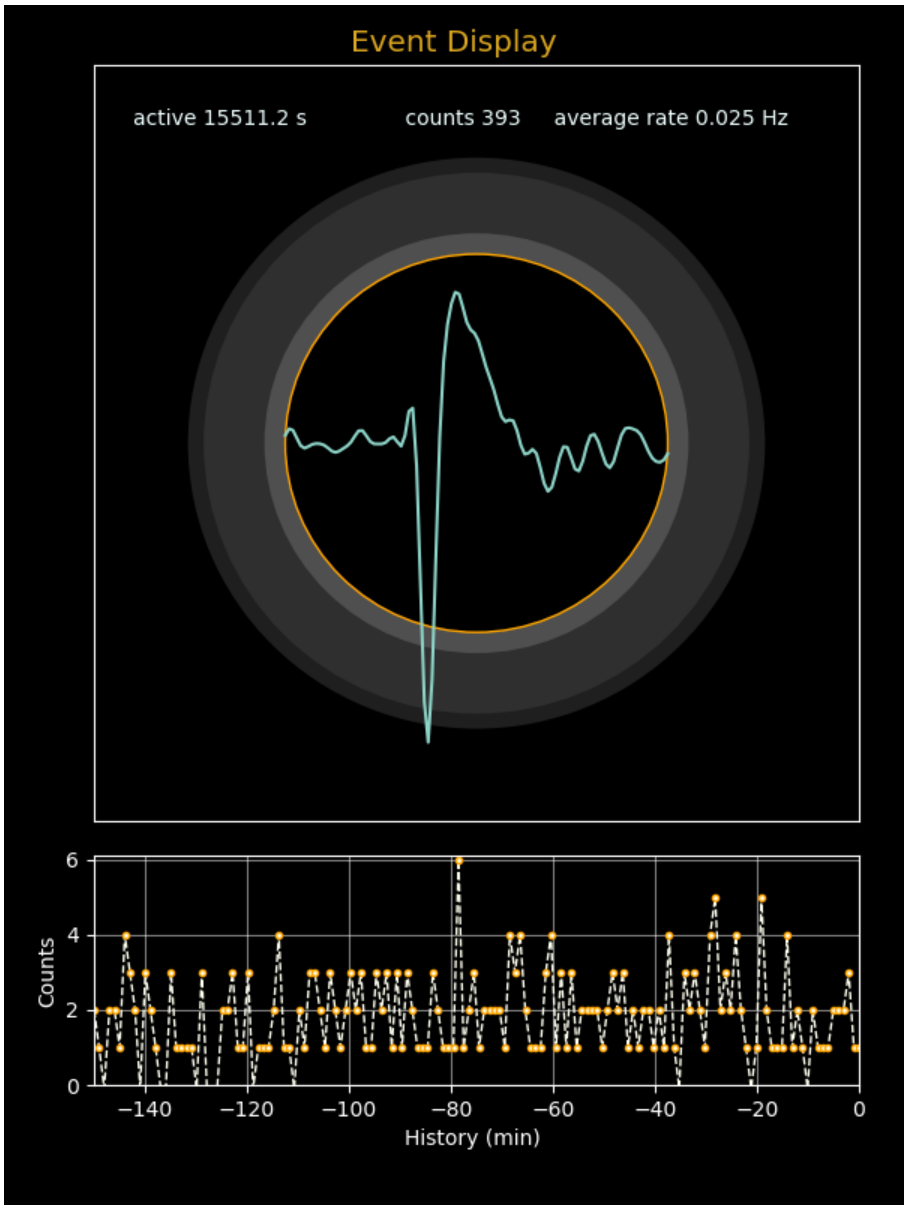

Abbildung 2: Grafische Darstellung der Datenerfassung unter Umgebungsbedingungen. Es wird eine durchschnittliche Zählrate von etwa 1.5 Signalen pro Minute beobachtet.

bewerkstelligen. Der Untergrund kann durch Messungen ohne die Signalquelle recht genau bestimmt werden. Die Bestimmung der Signaleffizienz hingegen erfordert genaue Kenntnis des Detektors und der Signaleigenschaften, die in der Regel durch eine detaillierte Modellierung der physikalischen Prozesse im Detektor und des Ansprechverhaltens der Front-End-Elektronik gewonnen werden. Für den Fall dass derartige Untersuchungen von Signal-Effizienz vs. Reinheit von Interesse sind, können die Daten `scGammaDetector.py -f <name>` in eine Datei `<name>_time.csv` schreiben und anschließend die aufgezeichneten Daten auswerten, um weitere Untersuchungen durchzuführen. Ein Beispiel dazu wird weiter unten besprochen.

Messungen mit dem Selbstbaudetektor

Umweltradioaktivität Die Sensorfläche des DIY Teilchendetektors ist mit nur 28 mm² sehr klein, und außerdem ist die sensitive Schicht sehr dünn. Von den typischerweise einigen Hz Gamma-Rate bei normaler Umgebungsradioaktivität von typischerweise 0.1 µS/h wird daher nur ein sehr kleiner Bruchteil registriert. Durch Vergleich mit einem Dosimeter, in diesem Fall ein Radiacode 102, s. Anleitung, kann man eine ungefähre Kalibration vornehmen:

> Eine Dosisrate von 0.1 µS/h entspricht etwa 1.3 registrierten Ereignissen pro Minute.

Eine Beispielmessung der Umgebungsaktivität wurde oben schon gezeigt. Wenn man also hinreichend lange Messzeiten vorsieht, sind mit dem Selbstbaudetektor Studien zur Radioaktivität in verschiedenen Umgebungen machbar. Der Unterschied der Dosisleistung im Freien, in Wohnräumen oder in Räumen mit Fliesen oder gar Granitsteinen unterscheidet sich um Faktoren Zwei bis Drei, die mit Messzeiten von einigen 10 Minuten statistisch signifikant nachweisbar sind. Auch der Effekt schwach radioaktiver Gesteinsproben oder auf der Oberfläche eines durch Reiben elektrisch geladenen Luftballons angereicherte Zerfallsprodukte von Radon sind nachweisbar.

Pulshöhenspektrum Die bei der Datenerfassung in einer Datei gespeicherten Daten werden für die statistische Auswertung der Pulshöhen verwendet. Die Datenaufzeichnung wird mit der Option `-f <Dateiname>` eingeschaltet. Für jedes aufgenommene Signal werde eine fortlaufend inkrementierte Ereignisnummer, die Zeit des Auftretens in Sekunden seit Programmstart, die peak-to-peak Pulshöhe in ADC-Counts des bipolaren Signals, das Verhältnis des der Höhen des zweiten und ersten Peaks, die zeitliche Distanz der Extrema und die Breiten bei halber Höher der Peaks in der Datei gespeichert. Die Datei im *csv*-Format enthält die Spalten

> `event_number,event_time,pp_height,p_ratio,p_dist,fwhm1,fwhm2`

Zur Anzeige des Spektrums der Pulshöhen, d. h. der Häufigkeit des Auftretens von Pulshöhen in einem bestimmten Zeitintervall, wird im Folgenden nur die dritte Spalte verwendet. Die übrigen Spalten können dazu genutzt werden, um die Signaleigenschaften genauer zu untersuchen und auch die Trennung von Signal- und Rauschpulsen vor allem bei kleinen Pulshöhen zu verbessern.

Das Ergebnis einer Messung mit Umgebungsradioaktivität ist in der folgenden Abbildung dargestellt. Die Lautstärkeeinstellung der Soundkarte wurde so gewählt, dass die Triggerschwelle noch im Bereich der Rauschsignale liegt. Beachten Sie, dass die Pulshöhen definiert sind als die Differenz zwischen dem maximalen und minimalen Wert im Signalbereich („peak-to-peak“ Pulshöhe).

Die logarithmische Darstellung zeigt eine sehr große Anzahl von Pulshöhen knapp oberhalb der Triggerschwelle, die zu größeren Werten hin schnell abfällt. Ab etwa 10000 ADC-Zählungen sieht man die flache Verteilung, die die durch die Signale der Gammastrahlung verursacht wird.

Setzt man die Triggerschwelle oberhalb des Übergangs von der Rausch- zur Signalverteilung, werden Rauschsignale unterdrückt und nur echte Gammastrahlen registriert. Die Triggerrate entspricht dann direkt der Rate der detektierten Gammastrahlen.

Zur Darstellung der Pulshöhenverteilung wurde das unten beschriebene Python*-Programm `data/GammaAnalysis.py` verwendet.

Wenn man außer der den peak-to-peak Pulshöhe auch die anderen typischen Merkmale der Pulsformen verwendet, ist es möglich, Rausch- und Signalpulse auf Grund ihrer typischen Form zu unterscheiden. Auf diese Art kann der Verlauf des Pulshöhenspektrums auch im Bereich des Rauschens dargestellt werden. Dies ist unten in Abbildung 4 gezeigt. Die Erkennung der orange dargestellten Rauschpulse wurde mit einem künstliche neuronalen Netz erreicht, dass auf Signalformen für große und sehr kleine Pulse trainiert wurde.

Statistik bei radioaktiven Zerfällen Die Analyse von mit `scGammaDetector.py` aufgezeichneten Gammaquanten aus einer kleinen Probe Pechblende ist in der nachstehenden Abbildung gezeigt. Signalpulse wurden mit einer Rate

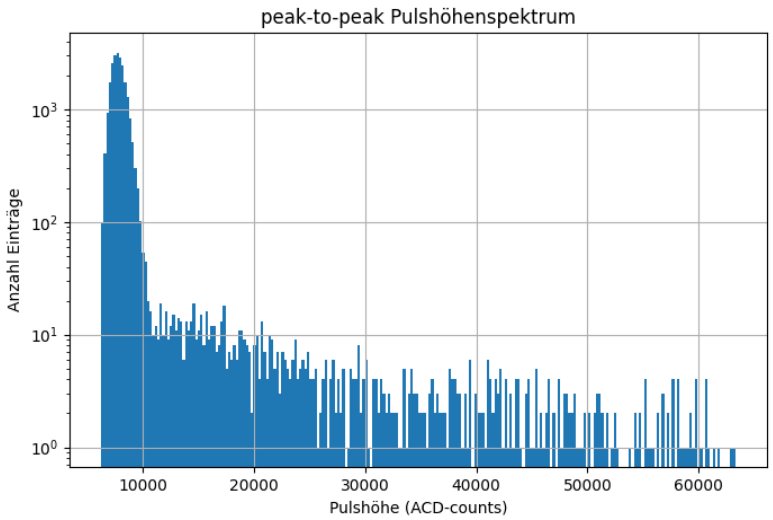

Abbildung 3: Häufigkeitsverteilung der Pulshöhen, die mit einer Triggerschwelle im Bereich der Rauschimpulse bei einer Umgebungsaktivität von von 0,085 µSv/h aufgenommen wurden.

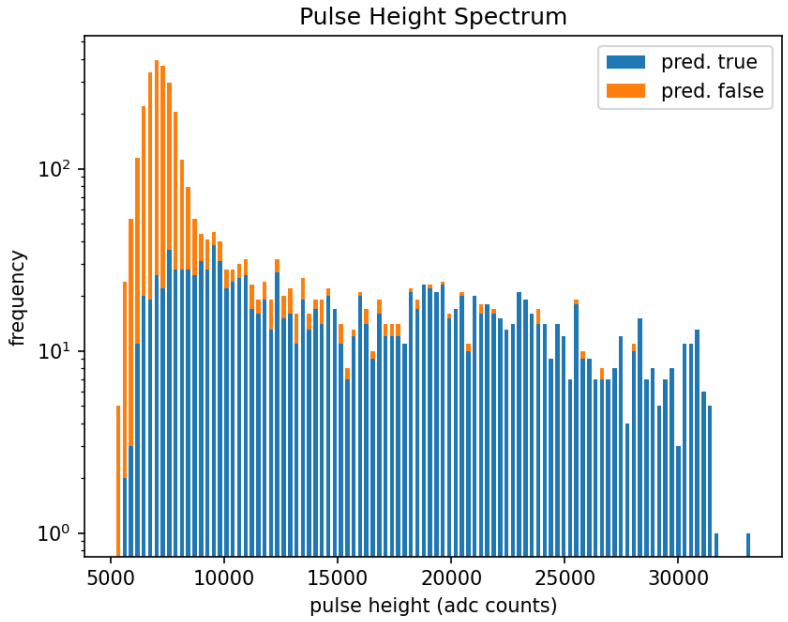

Abbildung 4: Häufigkeitsverteilung der Pulshöhen einer schwach radioaktiven Gesteinsprobe aus dem Schwarzwald. Signal- und Rauschpulse wurde mit Hilfe eines künstlichen neuronalen Netzes klassifiziert, das auf die typischen Signalformen kleiner und großer Pulshöhen trainiert wurde.

von ca. 1,1 HZ registriert. Die Datei `GammaStrahlung_Pechblende.csv` enthält ca. 10000 aufgezeichnete Ereignisse. Ausgewertet wurde die mittlere Spalte mit den Zeiten, zu denen Ereignisse registriert wurden. Dazu wurde der unten näher erläuterte Python-Code `data/GammaAnalysis.py` verwendet.

Die Grafiken zeigen die Zahl der Ereignisse in Intervallen von 10 s Dauer, die Häufigkeitsverteilung der beobachteten Ereignisanzahlen und die Zeit zwischen zwei Ereignissen. Die sich aus der mittleren Rate ergebenden erwarteten Verteilungen sind ebenfalls eingezeichnet, d. h. eine Gleichverteilung für eine mittlere Ereignisanzahl von 11,1 in jedem 10 s-Intervall, die entsprechende Poisson-Verteilung und eine Exponentialverteilung für einen mittleren zeitlichen Abstand von 1/1.11 s = 0.90 s zwischen den Ereignissen sind ebenfalls dargestellt. Die Grafiken zeigen sehr schön die für einen Poisson-Prozess erwarteten Eigenschaften.

Zerfall von Radon Das radioaktive Edelgas Radon-222 entsteht in Kernzerfällen im Erdinneren und gelangt durch Spalten im Erdboden an die Oberfläche. Insbesondere in der Luft in Gebäuden kann es sich bei schlechter Lüftung anreichern. Eine Radon-Probe kann aus der Luft entnommen werden, indem die Radon-Zerfallsprodukte auf einem Ballon gesammelt werden, der durch Reibung elektrostatisch aufgeladen wird. Das Isotopengemisch wird von den kurzlebigen Tochterkernen Po-218, Pb-14, Bi-214 und Po-214 dominiert, wie aus dem nachstehenden Diagramm hervorgeht.

Eine solche Radon-Probe enthält also eine Reihe von kurzlebigen Tochterkernen und eignet sich daher gut zur Veranschaulichung der Zeitabhängigkeit der Aktivität. Die Rate als Funktion der Zeit ist unten gezeigt; die Daten in der Datei `Radon.csv` wurden mit dem unten beschriebenen Python-Skript `RadonAnalysis.py` ausgewertet und grafisch dargestellt.

Da mehrere Tochterkerne beteiligt sind, folgt die Verringerung der ursprünglichen Aktivität nicht dem exponentiellen Zerfallsgesetz. Zu erkennen ist ein deutlicher Anstieg der Aktivität über dem Untergrundniveau nach etwa 12 min, als der Ballon nach ca. 15 min Akkumulationszeit auf den Detektor gelegt wurde. Die anfängliche Mischung der Isotope Pb-214 und Bi-214, deren β-Zerfälle von Gamma-Emmissionen begrleitet ist, ist unbekannt. Der Zerfall von Pb-214 und die Erzeugung von Bi-214 sowie dessen anschließender Zerfall mit Lebensdauern von 27 min und 20 min ergeben den gezeigten zeitlichen Verlauf der Gamma-Rate, die nach ca. 2,5 Stunden wieder auf das Niveau der Umgebungsstrahlung abfällt.

Software zur Auswertung

Der Code zur Erzeugung der oben gezeigten Ergebnisgrafiken befindet sich im Python-Script `data/Analysis.py`, das als Beispiel für eigene Auswertungen oder auch als gebrauchsfertiges Programm dienen kann. Die beiden Dateien `data/Umgebung_lowTrigger.csv` und `data/Pechblende.csv` enthalten Daten, die mit dem CERN DIY-Detektor und dem Programm `scGammaDetector.py` aufgezeichnet wurden und mit `GammaAnalysis.py` ausgewertet werden können. Der erste Datensatz wurde mit einer niedrigen Triggerschwelle lediglich mit Umgebungsradioaktivität aufgenommen, bei der zweiten Messung mit einer Probe radioaktiver Pechblende war die Triggerschwelle deutlich oberhalb des Rauschniveaus eingestellt.

Eingabe von

```
> python3 GammaAnalysis -h
```

zeigt die vorhandenen Optionen und Parameter:

```
*===* script ./Analysis.py executing, parameters: ['-h']

usage: GammaAnalysis.py [-h] [-b BINS] [-i INTERVAL] [-c CUT] inFileName

Analysis of DIY Detector

positional arguments:
  inFileName            input file name (CSV format)

options:
  -h, --help            show this help message and exit
  -b BINS, --bins BINS  bins for Pulse Height Histogram
  -i INTERVAL, --interval INTERVAL
```

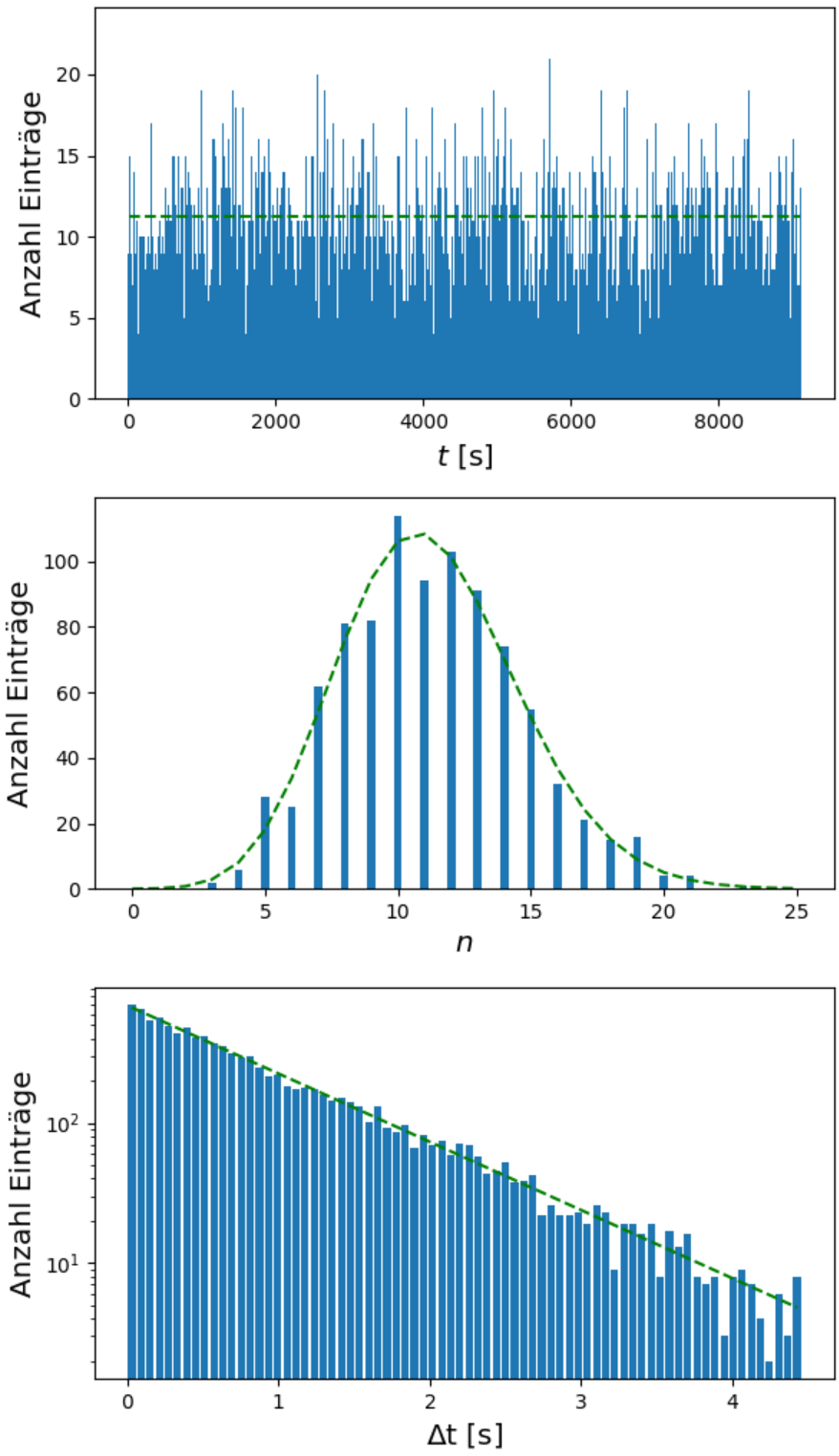

Abbildung 5: Darstellung der Zahl der Ereignisse in Intervallen von 10 s Dauer, die Häufigkeitsverteilung der beobachteten Ereignisanzahlen und die Zeit zwischen zwei Ereignissen.

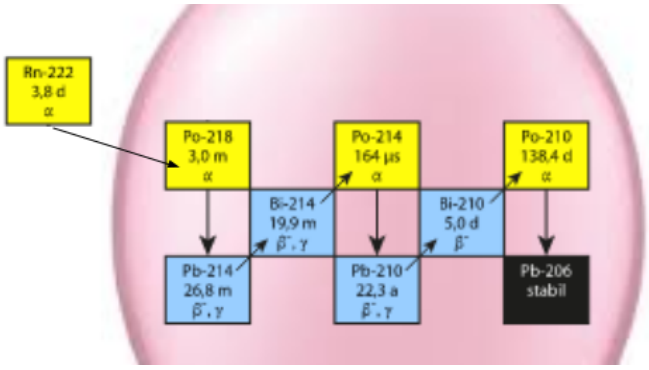

Abbildung 6: Radon-Zerfälle

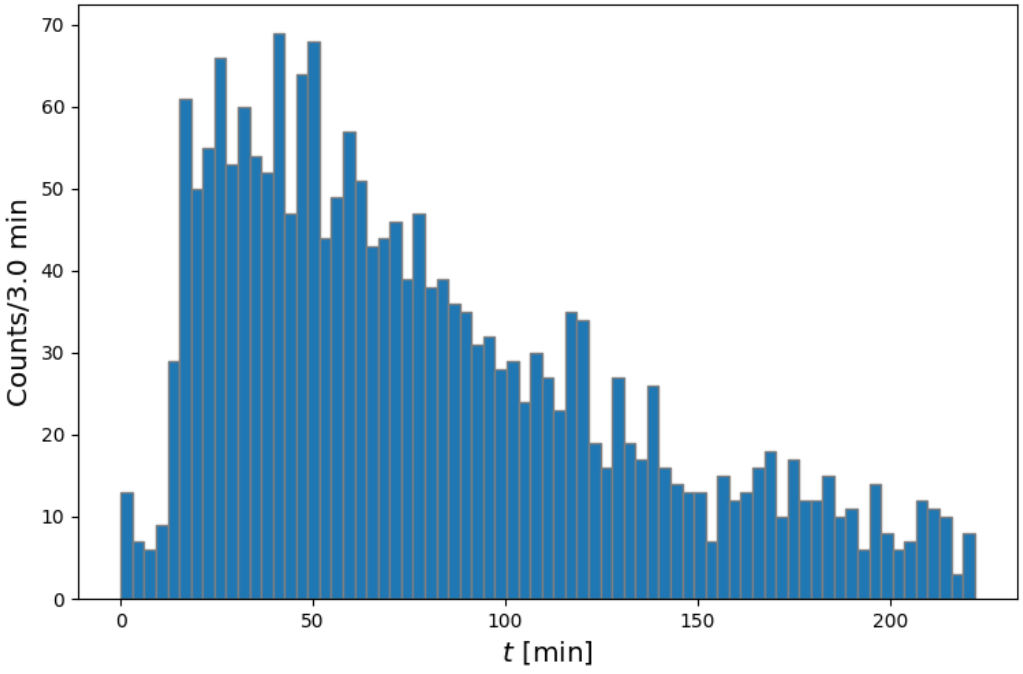

Abbildung 7: Anzahl von Ereignissen in 3 min-Intervallen von Radon-Zerfallsprodukten auf einem Luftballon in Abhängigkeit von der Zeit.

```
                      time interval for Rate Histogram
  -c CUT, --cut CUT   cut on minimal pulse height
```

Dargestellt werden die Pulshöhenverteilung in *BINS* Intervallen sowie die Raten in Zeitintervallen der Länge *INTERVAL*; die Parameter *BINS* und *INTERVAL* werden als Optionen beim Start angeben. Wenn Rauschpulse in den Daten enthalten sind, kann der Parameter “CUT” verwendet werden, um kleinere Pulse bei der Analyse der Raten zu unterdrücken.

Die oben gezeigten Grafiken werden mit den Befehlen

```
python3 GammaAnalysis.py -c 11000 -i 60 Umgebung_lowTrigger
```

oder

```
python3 GammaAnalysis.py -i 10 Pechblende.cvs
```

bzw.

```
python RadonAnalysis.py -i 180 Radon.csv
```

erzeugt.