Τύχη: Zufall durch Zerfall

Piro E. Bakallbashi & Benedikt C. Mues
Maximiliansgymnasium München
Jugend forscht 2026

Inhaltsverzeichnis

1. Einleitung

Traditionell werden Zufallszahlen in Computern durch eine Mischung aus Benutzereingaben und Systemparametern generiert (Wikipedia, Entropy computing 2025). Sie sind dadurch nicht wirklich zufällig, sondern basieren auf Faktoren, die bestimmbar sind. Obwohl es schwierig ist, diese Zahlen vorherzusagen, auch wenn man den Systemzustand eines Computers hätte, ist es nicht unmöglich. Trotz der Tatsache, dass die meisten normalen Zufallsgeneratoren ausreichend zufällig für Alltagsanwendungen wie Computerspiele sind, besteht dennoch die Möglichkeit, diese Schwachstelle für einen Angriff zu nutzen. Dies kann vor allem in sicherheitskritischen Kontexten verheerende Folgen haben.

In dieser Arbeit wird ein Ansatz dargestellt, mit welchem es möglich ist, echte zufällige Information, welche man also unmöglich vorhersagen kann, zu schöpfen. Dies erfolgt durch Umwandlung zeitlicher Intervalle zwischen radioaktiven Impulsen der Hintergrundstrahlung in unverschobene Zufallsinformation. Detektiert werden diese auf einem Geiger-Müller Zählrohr. Bisherige Ansätze zur Erzeugung echter Zufallszahlen basieren häufig auf quantenoptischen Prozessen oder elektronischem Rauschen (Ma, Xiongfen et al. 2016). Allerdings ist die physikalische Herkunft der Zufälligkeit in vielen Systemen schwer experimentell zugänglich. Bei diesem Projekt lag der Fokus natürlich auf einer bezahlbaren Umsetzung.

Unsere Ergebnisse zeigen, dass Zerfallsintervall-basierte TRNGs auf einfacher Hardware eine brauchbare Quelle echter Zufallsinformation darstellen können, sofern geeignete Nachbearbeitungs- und Bewertungsmethoden angewendet werden. Ein TRNG ist ein True Random Number Generator, der Zufallszahlen aus physikalisch nicht-deterministischen Prozessen gewinnt. Der Code unseres Projektes ist verfügbar unter diesem Link.

2. Grundlagen der Radioaktivität

2.1. Physikalische Erklärung

Ein Atomkern ist radioaktiv, wenn er instabil ist. Instabilität bedeutet, dass das Verhältnis von Protonen zu Neutronen im Kern so ist, dass die Kernkräfte den Kern nicht dauerhaft zusammenhalten können. Solche Kerne wandeln sich spontan um, indem sie ionisierende Strahlung aussenden – zum Beispiel Alphateilchen, Betateilchen oder Gammastrahlung.

Beim Zerfall kann der Kern entweder in einen energetisch niedrigeren Zustand übergehen oder sich in einen Tochterkern eines anderen chemischen Elements umwandeln. Die Wahrscheinlichkeit, dass ein einzelner Kern innerhalb einer bestimmten Zeit zerfällt, wird durch die Zerfallsrate λ beschrieben. Sie hat die Einheit s^{-1}.

Mathematisch folgt der radioaktive Zerfall dem Gesetz:

$$\frac{dN}{dt} = -\lambda N$$

wobei N die Anzahl der noch nicht zerfallenen Kerne ist. Diese Gleichung besagt: Die Änderung der noch nicht zerfallen Kerne über die Zeit ist proportional zu ihrer aktuellen Zahl. Dabei ist λ der Proportionalitätsfaktor. (Ridha, Ali A. 2026)

2.2. Indeterminismus der Radioaktivität

Eine Besonderheit der Radioaktivität ist, dass sie auf einem quantenmechanischen Prinzip beruht — der Heisenbergschen Unschärferelation:

$$\Delta E * \Delta t \geq \frac{\hbar}{2}$$

Dabei ist ΔE die Unschärfe der Energie eines Systems, Δt die Zeitspanne, in der ein Zustand existiert oder beobachtet wird. Die Gleichung besagt, dass die Energie eines Systems umso ungenauer bestimmt werden kann, je kürzer die Zeit ist, in der der Zustand besteht oder beobachtet wird. Also darf die Energie des Teilchens für kurze Zeiten schwanken. Um ein instabiles Atom in einen stabilen Zustand zu überführen, wäre klassisch gesehen eine bestimmte Aktivierungsenergie nötig. Diese Energie steht jedoch im Normalfall nicht zur Verfügung, weshalb der Zerfall nach klassischer Physik unmöglich wäre. Man kann sich das wie einen Ball vorstellen, der nicht genug kinetische Energie hat, um einen Hügel zu überqueren. In unserem Fall ist der Ball die Teilchen, bei Alpha-Strahlung zum Beispiel zwei Protonen und zwei Neutronen, und der Hügel sind Wechselwirkungen im Atomkern. (Hilgevoord 2025, S. 1454)

Wie eben erklärt, ist diese Energie aber unscharf und mit einer gewissen Wahrscheinlichkeit größer als die benötigte Aktivierungsenergie. Dadurch besteht eine kleine Möglichkeit, dass das Teilchen die Kernbarriere quantentunnelnd überwindet und so eine Emission (z. B. von Alphateilchen) stattfindet. Dieser Vorgang ist rein zufällig. Der exakte Zeitpunkt einer Emission lässt sich daher mathematisch nicht vorhersagen. Bestimmen lässt sich lediglich die Wahrscheinlichkeit, dass eine Emission innerhalb eines bestimmten Zeitintervalls auftritt.

Die Wahrscheinlichkeit, dass das Teilchen seinen Zustand ändert ist unabhängig davon, wie lange das Teilchen schon in diesem Zustand besteht. Der Zerfallsprozess ist somit zeitlich gedächtnislos; die Zerfallswahrscheinlichkeit pro Zeiteinheit ist zu jedem Zeitpunkt gleich groß (Universität Graz 2025, S. 1).

3. Problemstellung und unser Lösungsansatz

Durch den eben beschriebenen Indeterminismus ist es also möglich, einen TRNG zu erschaffen. Ein TRNG erzeugt Zufallszahlen aus physikalischen, nicht-deterministischen Prozessen wie Rauschen, radioaktivem Zerfall oder thermischen Schwankungen. Die RNG-Module, die in Programmiersprachen verwendet werden, sind PRNGs (Pseudo-Random Number Generator). Diese sind Algorithmen, die aus einem Startwert (Seed) eine deterministische, aber zufällig erscheinende Zahlenfolge erzeugen:

1. **Python**

`random`-Modul
Generator: Mersenne Twister

---

2. **C/C++**

`rand`-Modul
Generator: Linear Congruential Generator

---

3. **Java**

`java.util.Random`-Modul
Generator: Linear Congruential Generator

(Alexander Obregon, 2025), (Sebastiano Vigna, 2025)
CSPRNGs (Cryptographically Secure Pseudo-Random Number Generator) und PRNGs sind also ausreichend zufällig und schlussfolgernd auch sicher genug für die meisten Anwendungen. Es besteht dennoch aber wegen des deterministischen Charakters eines (CS-)PRNGs die Möglichkeit, das

Produkt (CS-)PRNGs mit großer Mühe zu rekonstruieren. Ein grundsätzliches Problem von PRNGs besteht beispielsweise bei Simulationen wie der Monte-Carlo-Schätzung von π: Bei PRNGs zeigen sich in großen Datenmengen Korrelationen, wodurch die Punkte im n-dimensionalen Raum nur auf Diagonalen beziehungsweise Ebenen liegen (Pierre L´Ecuyer, 2004). Dies führt zu systematischen Verzerrungen. Die Zeitintervalle radioaktiver Zerfälle sind hingegen vollständig unabhängig voneinander, sodass solche Muster nicht auftreten. In dieser Arbeit soll diese Eigenschaft der Radioaktivität genutzt werden, um gleichverteilte Zufallsinformation zu schöpfen.

4. Unser Quantilverfahren zur Informationsextraktion

4.1. Problem der Informationsextraktion

Die Wahrscheinlichkeit von Zeitintervallen zwischen radioaktiven Zerfällen folgen einer exponentiellen Verteilung, bei der kurze Zeiten grundsätzlich wahrscheinlicher sind als lange. Wenn man die gemessenen Intervalle einfach in Binärzahlen umwandeln würde, entstünden systematische Muster, weil bestimmte Werte — also kurze Intervalle — viel häufiger vorkommen. Die daraus gewonnenen Bits hätten einen statistischen Bias, etwa mit einem Übergewicht an Nullen. Eine echte Zufallsquelle muss jedoch gleichverteilte Ergebnisse liefern, damit in unserem Fall gilt:

$$P(0) = P(1) = 0.5$$

Deshalb kann man die Rohdaten aus den Zerfallszeiten nicht direkt verwenden, sondern muss sie erst „entverzerren“ um eine gleichverteilte Zufallsquelle zu erhalten.
Eine passende Metapher ist ein Würfel. Ein Würfel ist grundsätzlich nichts anderes als ein Zufallsgenerator. Der Unterschied zwischen einem Würfel und der Messung zwischen radioaktivem Zerfall ist zunächst, dass die Ausgänge eines Würfels diskret sind

$$\Omega_W = \{1, 2, 3, 4, 5, 6\}$$

während sie bei der Messung der Zeitintervalle des radioaktiven Zerfalls kontinuierlich sind.

$$\Omega_Z = \mathbb{R}^+$$

Der entscheidende Unterschied ist jedoch, dass bei einem Würfel (solange er nicht gezinkt ist) jeder Ausgang gleichwahrscheinlich ist. Bei der Messung der Zeitintervalle des radioaktiven Zerfalls hingegen, wie bereits aufgezeigt, nicht. Im folgenden Abschnitt wird der Algorithmus für diese Entzerrung beschrieben.

4.2. Der Algorithmus

Das Ziel dieses Algorithmus ist es, wie bei einem Würfel, eine diskrete Menge an gleichwahrscheinlichen Ausgängen zu finden. Zunächst wird eine Zerfallsrate λ der Radioaktivität der Umgebung geschätzt, indem man während einer Kalibrierungsphase, in der noch keine Zufallsinformation gewonnen werden, die Zeitintervalle zwischen Detektionen des Geigerzählers misst und speichert. Dies erfolgt durch folgende Gleichung:

$$\lambda_g = \frac{1}{\mu_t}$$

Dabei ist λ_g die geschätzte Zerfallsrate und μ_t der Mittelwert, also Durchschnitt, der Zeitintervalle. Intuitiv kann man sich diese Gleichung so vorstellen: wenn ein Zerfall z.B. durschnittlich alle fünf Sekunden stattfindet, dann ist die Wahrscheinlichkeit, dass er in einer bestimmten Sekunde zerfällt $\frac{1}{5s} * 1s$. Genau das ist die Zerfallsrate: die Wahrscheinlichkeit, dass ein Ereignis in einer bestimmten Zeiteinheit geschieht. Anhand dieser stellen wir die Wahrscheinlichkeitsdichtefunktion

$$f(t) = \begin{cases} \lambda e^{-\lambda t} \text{ für } t \geq 0 \\ 0 \text{ für } t < 0 \end{cases}$$

auf. Sie beschreibt die Zerfallszeit eines Atoms als stetige Zufallsvariable. Die Wahrscheinlichkeit, dass ein Atom im Zeitintervall $[t, t + dt]$ zerfällt, ist näherungsweise $f(t)dt$ (Wikipedia, Exponential Distribution 2026).
Diese Wahrscheinlichkeitsdichtefunktion wird jetzt in eine bestimmte Anzahl an Quantilen eingeteilt. Diese Quantile haben das gleiche Integral, also die gleiche Wahrscheinlichkeit, dass ein Zeitintervall ihnen zugeteilt wird. Jedes Quantil repräsentiert also eine gleichwahrscheinliche Zufallsklasse.
Um auf die eben aufgestellte Metapher zurückzukommen: Dadurch finden wir, wie bei einem Würfel, die gleichwahrscheinlichen Ausgänge. Da sie aber in unserem Fall kontinuierlich sind, brauchen wir Quantile und nicht nur konkrete Werte wie $1, 2, 3...$ Dies geschieht folgendermaßen:

- Das unbestimme Integral wird berechnet:
$$\int \lambda e^{-\lambda t} dt = -e^{-\lambda t} + C$$
- Das bestimmte Integral der Wahrscheinlichkeitsdichtefunktion wird mit p_k gleichgesetzt, wobei der Startwert des Integrals 0 und der Endwert, den es herauszufinden gilt, q_k ist. Dabei ist p_k der vorbestimmte Wert des Integrals des $k$ten Quantils. Zum Beispiel:
Man will die Exponentialverteilung in 8 gleichwahrscheinliche Quantile unterteilen. Dann wäre $p_1 = \frac{1}{8}; p_2 = \frac{2}{8}; p_3 = \frac{3}{8}$ etc.
$$\int_0^{q_k} \lambda e^{-\lambda t} dt = p_k$$
- Die eben aufgestellt Gleichung wird nach q_k aufgelöst:
$$q_k = -\frac{1}{\lambda} \ln(1 - p_k)$$
- Nun iteriert man über die Variable k für $k \in \mathbb{N}; k < n$. n ist die Anzahl der Quantile. Dabei setzt man $\frac{k}{n}$ für p_k ein in der Gleichung und erhält q_k, also den Endpunkt des Integrals und damit die obere Grenze des $k - 1$ten Quantils. Die obere Grenze des letzten Quantils liegt bei $t = \infty$.

```
for (int k = 1; k <= max_bins; ++k)
{
    double p = static_cast<double>(k) / max_bins; // p_k = k/n
    quantile[k - 1] = -std::log(1.0 - p) / lambda_hat;
}
```

Siehe Code

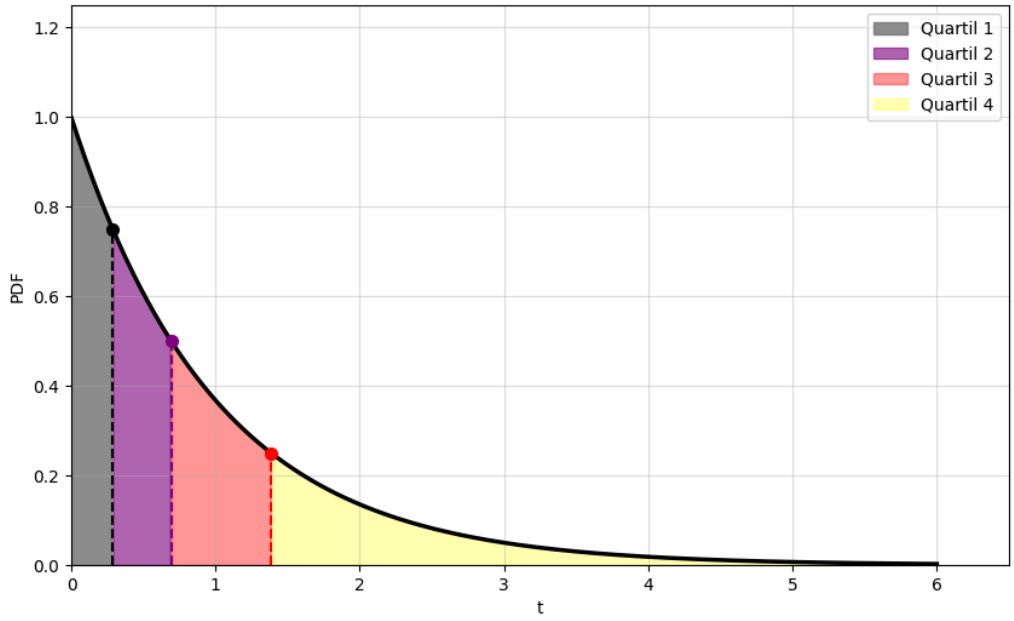


Abbildung 1: Die Exponentialfunktion in 4 gleichflächige Quantile

5. Grundlagen des Welch t-Tests

5.1. Grundsätzliches Problem

Ein Problem bei dem eben beschriebenen Quantilverfahren ist, dass man davon ausgeht, dass sich die Zerfallsrate nicht ändert, das tut sie aber offensichtlich. Wenn man sich zum Beispiel einer radioaktiven Quelle nähert, erhöht sich die Zerfallsrate und verringert den Betrag der Zeitintervalle. Allein in Deutschland variiert an verschiedenen Orten die Hintergrungstrahlung um einen Faktor von bis zu fünf (Stand 21.12.2025) (Bundesamt für Strahlenschutz 2025) . Dadurch würden viele Zeitintervalle bei einer Erhöhung der Zerfallsrate in die ersten Quantile eingeteilt werden, was die Zufallsinformation vorhersehbar macht. Also muss eine Struktur geschaffen werden, um zu prüfen, ob sich die Zerfallsrate ändert und ob diese Änderung nur Zufall geschuldet sein kann oder ob ein externer Faktor sie beeinflusst. In diesem Fall müsste man die Zerfallsrate erneut schätzen, um einen Bias zu verhindern. Dies ist natürlich ein schwieriges Unterfangen, wie die folgende Karikatur zeigt:

(Unitychain 2025)
Es ist erforderlich, eine Grenze festlegen, ab der ein beobachteter Unterschied nicht mehr als bloßer Zufall anerkannt wird. Eine absolut richtige oder zweifelsfreie Lösung gibt es dabei jedoch nicht – wie in der Karikatur angedeutet, kann man sich nie vollkommen sicher sein. In der Statistik unterscheidet man zwei mögliche Fehlentscheidungen:

- **Fehler 1. Art (α-Fehler):**
  Die Nullhypothese $H_0 : \mu_{\text{neu}} = \mu_{\text{vergleich}}$ - hier die Annahme, dass der Erwartungswert der Vergleichsdaten $\mu_{\text{vergleich}}$ dem Erwartungswert der neuen Daten μ_{neu}, der neuen „Batch“, entspricht – wird fälschlich verworfen, obwohl sie in Wirklichkeit wahr ist.
- **Fehler 2. Art (β-Fehler):**
  Die Nullhypothese wird beibehalten, obwohl sie in Wahrheit falsch ist. Die Alternativhypothese $H_1 : \mu_{\text{neu}} \neq \mu_{\text{vergleich}}$ wird nicht angenommen.

Zwischen diesen beiden Fehlerarten besteht ein grundlegender Zielkonflikt: Reduziert man das Risiko für den einen Fehler, steigt zwangsläufig das Risiko für den anderen. Darum muss man bei der Analyse von Zufallsquellen immer abwägen, wie viel Abweichung vom Ideal man tolerieren will, bevor man die Zufallsquelle nicht mehr als zufällig akzeptiert. Diese Abwägung beeinflusst dann, wie man den Parameter Konfidenz des in Abschnitt 5.2 beschriebenen Welch t-Tests wählt.

5.2. Welchs t-Test

Um die Zufälligkeit der Änderung der Zerfallsrate zu testen, wird der Welchs t-Test verwendet. Dieser prüft, ob sich die Mittelwerte - und damit die Zerfallsrate - der ursprünglich gesammelten Vergleichsdaten und der neu detektierten Zeitintervalle so unterscheiden, dass der Zufall nicht die einzige Ursache ist, sondern ein externer Faktor eine Rolle spielen muss. Der Welchs t-Test wurde über andere t-Tests bevorzugt, da keine Annahmen wie ähnliche Varianzen oder normalverteilte Zufallsgrößen erforderlich sind. Der t-Wert ist die Differenz der Mittelwerte der beiden Datensätze pro Standardfehler ihrer Differenz.

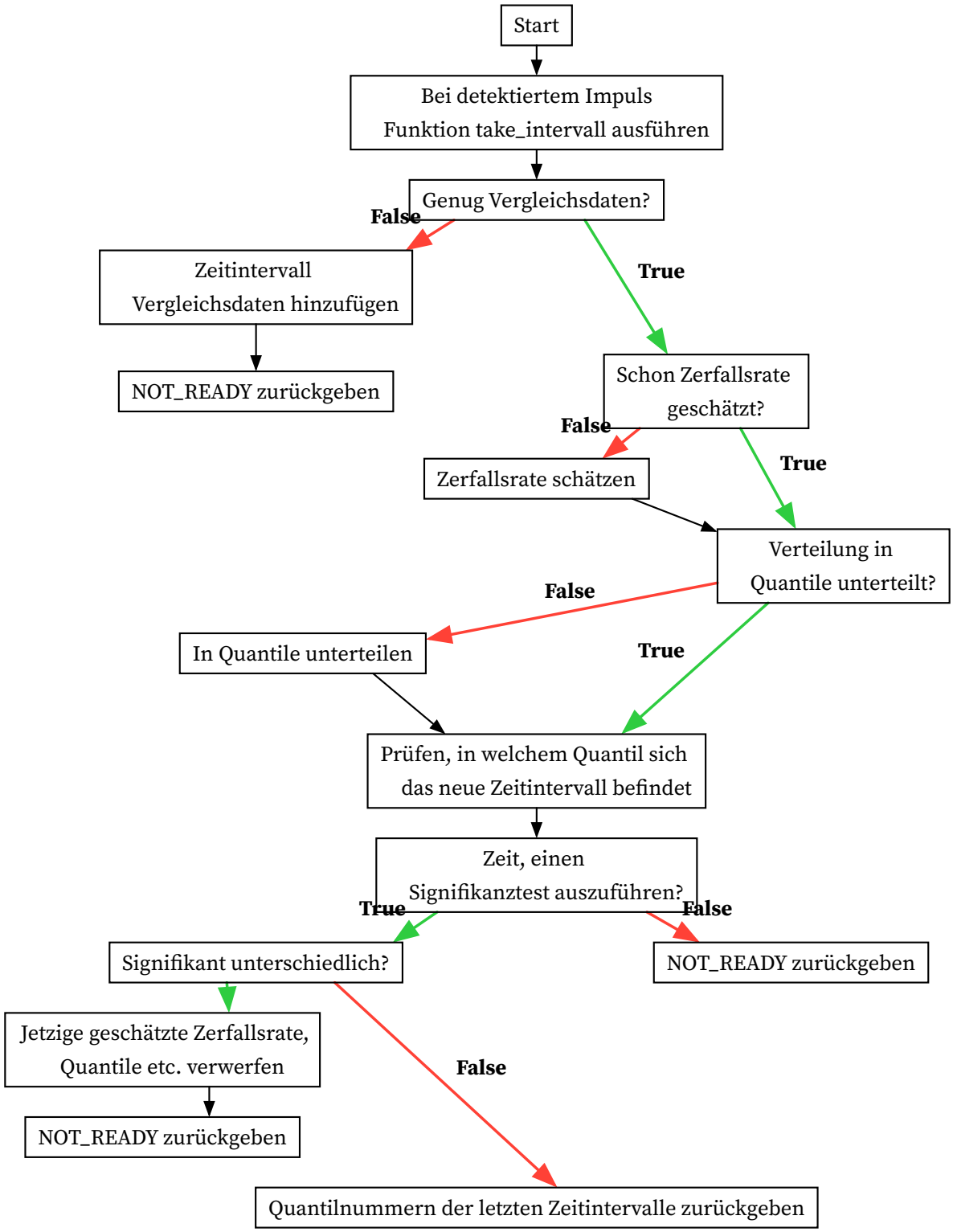
$$t = \frac{\mu_{\text{neu}} - \mu_{\text{vergleich}}}{\sqrt{\frac{s^2_{\text{neu}}}{n_{\text{neu}}} + \frac{s^2_{\text{vergleich}}}{n_{\text{vergleich}}}}}$$

Dabei ist μ jeweils der Mittelwert der Stichproben, s die Standardabweichung der Stichproben und n jeweils der Stichprobenumfang. Warum das funktioniert, kann man sich intuitiv so vorstellen: selbst wenn die Mittelwerte sich stark unterscheiden, aber die Streuung innerhalb der Datensätze groß ist, kann der Unterschied statistisch unbedeutend sein. Wenn der Betrag dieses t-Werts einen kritischen Wert überschreitet, wird geschlussfolgert, dass externe Einflüsse die Zerfallsrate beeinflusst haben. (Universität Bremen 2026)

Siehe Code

6. Zufallsinformationsgenerierung

6.1. Ablauf des Codes



6.2. Erklärung des Codes

Bei der Detektion eines Impulses durch das Zählrohr wird die Methode `take_intervall` der Klasse `Intervall2Bin` aufgerufen. Diese Methode nimmt als Input jeweils das letzte vom Geigerzählrohr gemessene Zeitintervall. Zunächst muss diese Methode hinreichend ausgeführt werden, bis eine akkurate Zerfallsrate aus Vergleichsdaten geschätzt werden kann. Während dieser Kalibrierungsphase liefert die Methode den Rückgabewert `NOT_READY`, d. h. es steht noch keine belastbare Zufallsinformation zur Verfügung. Daraufhin wird die Wahrscheinlichkeitsdichtefunktion

$$f(t) = \begin{cases} \lambda e^{-\lambda t} \text{ für } t \geq 0 \\ 0 \text{ für } t < 0 \end{cases}$$

in n Quantile unterteilt, die alle ein Integral von $\frac{1}{n}$ haben. Die Wahrscheinlichkeit, dass ein Zeitintervall in ein Quantil fällt, ist für alle Quantile gleich. Genauer beschrieben wird dieser Vorgang in Abschnitt 4. Anschließend wird geprüft, in welchem Quantil sich das Zeitintervall befindet, mit dem die Methode `take_intervall` aufgerufen wurde, und die Quantilnummer einem Vektor hinzugefügt.

Nach einem Batch an neu eingetroffenen Impulsen wird ein Signifikanztest ausgeführt, der prüft, ob sich die zuletzt gemessenen Zeitintervalle signifikant von denen der Vergleichsdaten, die gesammelt wurden, um die Zerfallsrate zu schätzen, unterscheiden. In anderen Worten: hat sich die Zerfallsrate geändert? Genauer beschrieben wird dieser Vorgang in Abschnitt 5.

1. **Wenn der Signifikanztest anschlägt** (sich die Zerfallsrate also signifikant geändert hat), werden jetzige geschätzte Zerfallsrate, sowie die Quantile der Zeitintervalle des letzten Batch als auch die aktuellen Vergleichsdaten verworfen. Es wird `NOT_READY` zurückgegeben.
2. **Wenn der Signifikanztest nicht anschlägt**, werden die Quantilnummern, in denen die Zeitintervalle des letzten Batch einzuordnen waren, zurückgegeben. Diese stellen die möglichst unverschobene Zufallsinformation dar.

Siehe Code

7. Hardware

Der Nukleus unseres Versuches ist ein Geiger-Müller-Zählrohr. Es wurde ein **RadiationD-v1.1** von der chinesischen Firma **CAJOE** verwendet, da bei diesem, im Gegensatz zu vielen anderen, ein Zugriff auf den Spannungspuls möglich ist.

Die zentrale Einheit des Zählers ist das Zählrohr Zwischen Kathode und Anode wird eine Gleichspannung angelegt. Fällt ein ionisierendes Teilchen ein, erzeugt dieses in der Gasfüllung — meist eine Mischung aus 90 % Edelgas und 10 % Halogengas (zur Unterdrückung von UV-Strahlung) — Elektronen, die sich zur Anode bewegen. Dadurch entsteht ein elektrischer Impuls, der vom Raspberry Pi registriert wird (Siehe Code).

Der Geigerzähler wird direkt vom Raspberry Pi mit Strom versorgt, wie der Schaltplan zeigt:

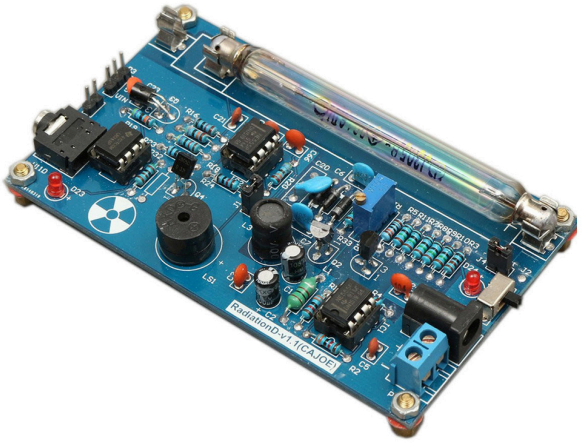


Abbildung 4: Das CAJOE Geiger-Müller-Zählrohr

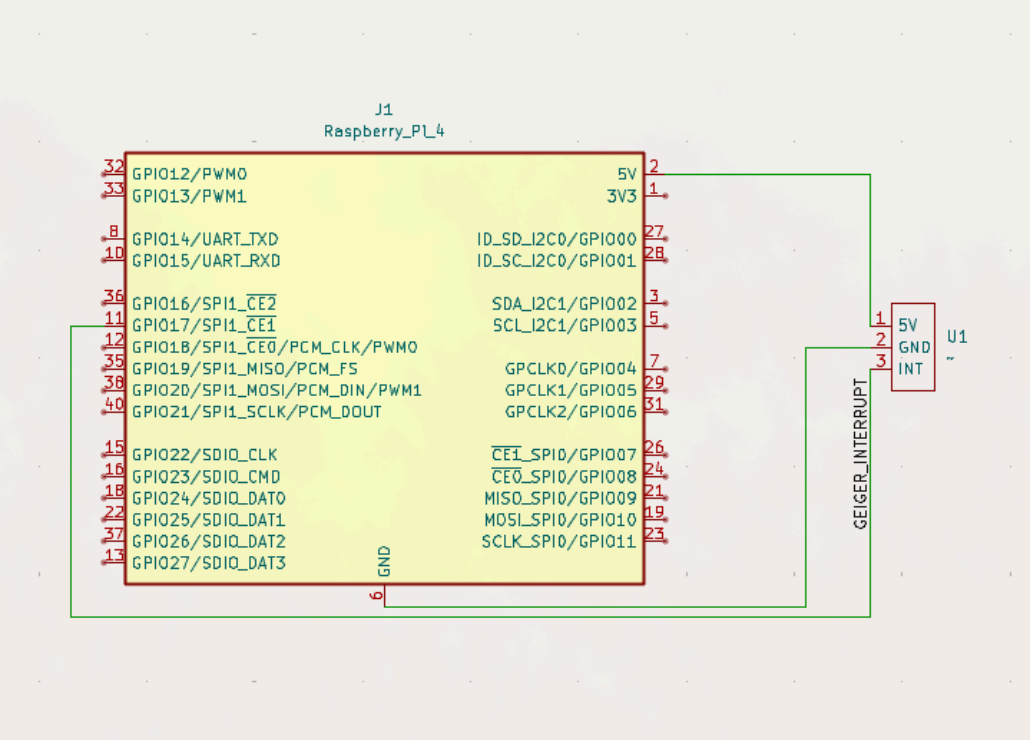


Abbildung 5: Der Schaltplan. `J1` bildet den Raspberry Pi ab, während `U1` eine starke Vereinfachung des Geigerzählers ist.

8. Qualitätsauswertung

8.1. Beispiel Bitfolge

```
01101010011110110111101100111
01111000111101100011111111111
01001001001111110000111011001 0
10111000010111010100111011101 0
01100001100101010011110110101 1
10000100100100011010011011110 1
01110110000100101010000011010 1
10101101101010011010011101110 1
10100100011100101100000010111 1
10100001100000111111101111101 0
00101111101100100110110000000 1
```

Abbildung 6: Auszug aus einer realen Bitfolge, die mit dem entwickelten Programm generiert wurde.

Bei diesen Bits gehören immer vier zusammen und geben ein Quantil an. Die ersten vier Bits 0110 zum Beispiel zeigen an, dass das erste Zeitintervall dem siebten Quantil aus 16 zugeordnet wurde (die Zählung beginnt bei 0000).

8.2. Beispiel Parameter

Während des Programms werden mehrere Parameter generiert. Hier ein Beispiel an echten Daten:

Zerfallsrate	$0.321s^{-1}$
t-Score	0.68
Quantilgrenzen	$0, 0.203s, 0.421s, 0.655s, 0.909s,$ $1.185s, 1.488s, 1.821s, 2.159s, 2.543s, 2.961s,$ $3.430s, 3.966s, 4.585s, 5.325s, 6.276s, \infty s$

8.3. Runs Test

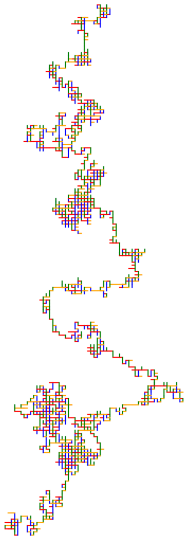
Die Qualität der Zufallsinformationen ist laut eines in Python implementierten Runs Tests nicht verschoben in eine Richtung, was ihn schwer vorhersehbar macht. Ziel dieses Tests ist es, die Gesamtzahl der Läufe in der Sequenz zu ermitteln. Der Test untersucht die Anzahl der Bits, die von 0 auf 1 oder von 1 auf 0 wechseln. Dies sind die „Läufe“ in der Sequenz. Darum ist das ein gutes Maß für Zufall: Hat man zum Beispiel eine Bit-Kette nur aus 0en, käme nur ein Run vor. Wenn sich 0en und 1en durchgehend abwechseln, kämen extrem viele Runs vor. Beide Ketten sind nicht besonders zufällig. Eine zufällige Kette an Bits hat also weder besonders viele Runs, noch besonders wenige. Anhand der Anzahl an Runs wird der „p-score“ berechnet. Bei diesem Test wurde eine ausreichende Menge an durch den TRNG erstellten Bits verwendet (≈ 4 Kilobyte).
Er erreichte einen p-Score $\approx 10.4\%$. Ein p-Score gibt die Wahrscheinlichkeit an, unter der Annahme, dass die Nullhypothese korrekt ist, ein Ergebnis zu beobachten, das mindestens so extrem ist wie das tatsächlich gemessene — je kleiner der p-Wert, desto stärker spricht das Ergebnis gegen die Nullhypothese (Andrade 2026). Ab einem kritischen p-score von 5% oder größer, geht man normalerweise von Zufall aus. Da der p-score der erstellten Daten größer ist als der der kritische, geht man von Zufall aus (Penn State University 2023). Siehe Code

8.4. Random Walk

Es wurde abgesehen von dem Runs Test ein weiterer visueller Qualitätstest ausgeführt, ein sogenannter „Random Walk“. Bei diesem bewegt man sich um einen Ursprungspunkt. Die Richtung in zwei Dimensionen, in die man sich bei einem Schritt bewegt, wird zum Beispiel durch zwei Bits bestimmt:

$$
\begin{aligned}
00 &\rightarrow \text{Eine Einheit nach rechts} \\
01 &\rightarrow \text{Eine Einheit nach oben} \\
11 &\rightarrow \text{Eine Einheit nach links} \\
10 &\rightarrow \text{Eine Einheit nach unten}
\end{aligned}
$$

Wie man erkennen kann, besteht ein klarer Bias in eine Richtung. Nach einigem Ausprobieren mit Testwerten, die mit Zufallsmodulen in `C++` erstellt wurden, wurde der Grund dieses Bias gefunden Überschätzt man die Zerfallsrate, mathematisch $\lambda_{\text{geschätzt}} > \lambda_{\text{real}}$ entsteht ein Drift in Richtung 1 unterschätzt man sie, in Richtung 0. Um diese Fehleinschätzung zu verhindern, wurde ein weiterer Initialisierungsparameter der Klasse `Intervall2Bin` eingeführt. Je größer man ihn einstellt, desto mehr Vergleichsdaten werden gesammelt, um die Zerfallsrate akkurater zu schätzen. Er sollte jedoch nicht zu groß eingestellt werden, da die Kalibrierungszeit quadratisch mit ihm steigt.



Eine weitere Maßnahme ist die Verringerung der Konfidenz beim t-Test von 95% auf 90%. Dadurch wird die Nullhypothese, die Annahme, dass sich die Zerfallsrate signifikant geändert hat, öfter verworfen und die Zerfallsrate schon bei einem kleineren Unterschied der Vergleichsdaten zu den aktuellen Daten erneut geschätzt, wodurch sie sich der „echten“ Zerfallsrate erwartungsgemäß annähert.

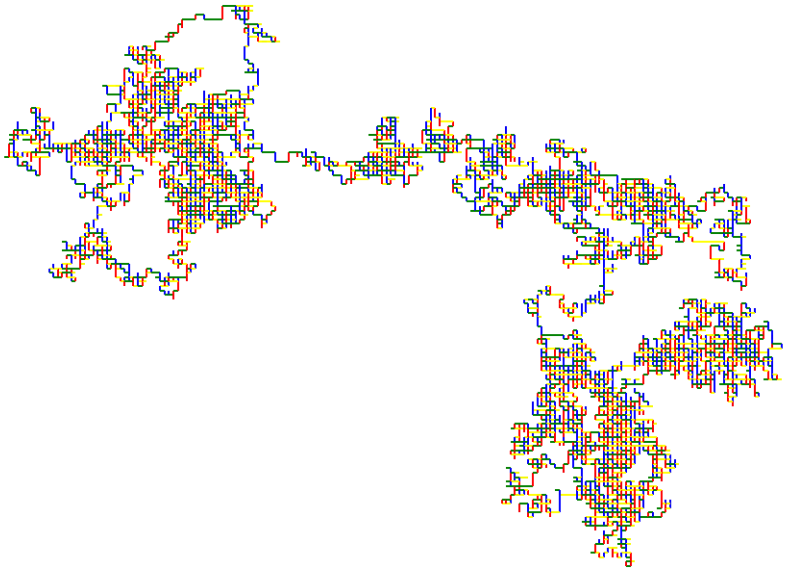
8.5. Vergleich mit bestehenden Ansätzen

Die erzeugten Bits würden bei Standard-Qualitätstests, etwa der Shannon-Entropie, die ein Maß für die Informationsdichte in binären Daten ist, wahrscheinlich schlechter abschneiden als professionelle TRNGs. Das liegt nicht an einer grundsätzlichen Verzerrung der Zufallsquelle, sondern daran, dass hier bewusst auf eine nachträgliche „Bereinigung“ verzichtet wurde. Professionelle TRNGs prüfen Rohbits vor der Nutzung; fallen sie durch, werden sie verworfen. In diesem Projekt wurde darauf verzichtet, um näher am unverfälschten Zufall zu bleiben, daher beschränkte man sich auf einen einfachen Welchs t-Test. Ein Vorteil unseres Ansatzes gegenüber anderen TRNGs, die auf radioaktivem Zerfall basieren, liegt in der Effizienz: Sie beschreibt, wie viel Information aus einem Ereignis extrahiert werden kann. Bei einer Effizienz von 100% kann ein Ereignis ein Bit liefern. Mit dem hier verwendeten Quantilverfahren Abschnitt 4 ist theoretisch eine unbegrenzte Effizienz möglich. Praktisch begrenzt wird sie nur durch die exponentiell wachsende Menge an Vergleichsdaten Abschnitt 9.2. Für dieses Projekt wurde eine Effizienz von vier Bits pro Impuls gewählt (400%), was viermal so hoch ist wie bei vielen vorherigen Projekten (Daniel Ruschen at al. 2023). Zudem ermöglicht die Hardware durch Backtesting in Form des Welchs t-Test die Nutzung auch in Szenarien, in denen die ursprüngliche Zerfallsrate in stark verstrahlten Umgebungen geschätzt wurde, während die eigentliche Zufallsinformation in Bereichen nur mit geringerer Hintergrundstrahlung erzeugt wird.

8.6. Mit der Linux Entropy-Pool

Das Betriebssystem Linux stellt Benutzern eine Möglichkeit zur Verfügung, dem eigenen Entropy-Pool Daten in Form von Bytes hinzuzufügen. So kann man dem Betriebssystem eine weitere Zufallsquelle geben. Mit diesem Code wurden Bytes vom Server abgefragt, der weiter unten detailliert ist, und dem Entropy-Pool hinzugefügt. Der Entropy-Pool legt die Daten dann in der Datei `/dev/random`

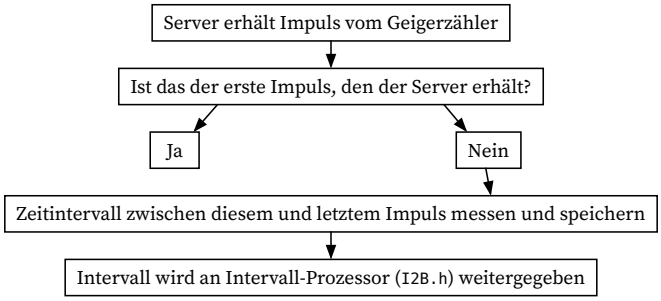
dem System vor. Es wurde ebenso ein Random-Walk für die Binärdaten aus dem Entropy-Pool erzeugt, welchem kurz zuvor die Daten aus unserem Server aus Abschnitt 9.1 hinzugefügt wurden. Der unten abgebildete Random-Walk benutzt dieselben Parameter wie der obige Random-Walk.

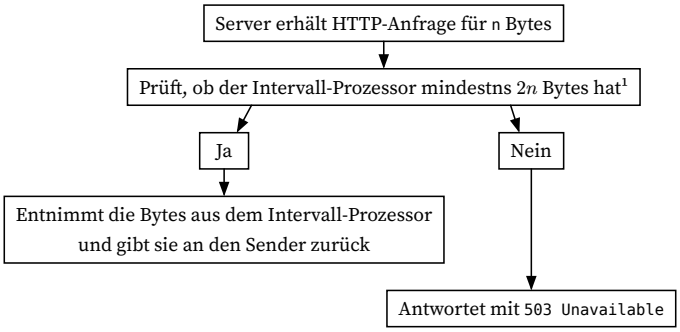


9. Praktische Anwendung

9.1. Zugriff

In dem zu Verfügung gestellten Quellcode des Projektes ist ein HTTP-Server eingebettet, wodurch man auf dem Endpoint `/?amount=n` `n` bis 4096 zufällige Bytes (4KiB) beantragen kann. Der HTTP-Server antwortet dann entweder mit `200 OK` und der Anzahl an Bytes oder mit `503 Unavailable`, sollte der Server noch nicht ausreichend Bytes haben. Der Ablauf des Servers lässt sich mit folgenden Diagrammen zeigen:





[1] Da der Intervall-Prozessor die Intervalle in 16 (2^4) Quantile einteilt, können aus einer Quantil-Nummer 4 Bits erzeugt werden. Weil ein Byte also aus 8 Bits besteht, werden 2 Quantil-Nummern pro Bit verwendet.

Wenn der Server dann eine Anfrage für n Bytes erhält, entnimmt er $2n$ so viele Quantilnummern aus der Liste von gespeicherten Quantilnummern und gibt sie zurück.[1]

Siehe Code

9.2. Verbesserungspotentiale

Mit reiner Hintergrundstrahlung steht natürlich deutlich weniger echte Zufallsinformation zur Verfügung als beim Einsatz einer zusätzlichen radioaktiven Quelle. Bei einer Zerfallsrate von ungefähr 0.2 bis 0.5 Zerfällen pro Sekunde kann man mit einem Byte alle 3 bis 6 Sekunden rechnen, wenn man eine vernünftige Menge an Vergleichsdaten verwendet, aus denen man die Zerfallsrate schätzt. Denn die Menge an Information, die man aus einem Zeitintervall vernünftig extrahieren kann und damit auch, wie lange es dauert, eine bestimmte Menge an Information zu gewinnen, hängt von der Genauigkeit von der Schätzung der Zerfallsrate ab. Genauer gesagt: Die maximale Anzahl an Quantilen, in die man eine Exponentialverteilung sinnvoll einteilen kann, ist begrenzt, weil jedes Quantil ausreichend Datenpunkte enthalten muss, um statistisch stabil zu sein.
Bei zu vielen Quantilen hätte das einzelne zu wenig Daten, wodurch die Schätzung der Verteilung verrauscht und unzuverlässig wird.
Wenn man jetzt also vier Bits pro Zeitintervall extrahieren will, kann man sich folgendermaßen die Anzahl an benötigten Vergleichsdaten errechnen:

$$\#\text{Vergleichsdaten} = \left(2^4\right)^2 = 2^8 = 256$$

Dabei ist 2^4 die Berechnung der benötigten Quantile und die Quadrierung gibt den Zusammenhang zwischen Quantilen und benötigten Vergleichsdaten wieder. In Abschnitt 8.4 wird gezeigt, dass es sinnvoll ist, die Quantile nicht nur zu quadrieren, sondern einen noch größeren Exponenten zu wählen, wenn man keinerlei Bias in eine Richtung möchte. Nur um die Vergleichsdaten zu sammeln, bräuchte man also ungefähr $\frac{256}{0.5s^{-1}} = 512s$, wenn man von einer durchnittlichen Zerfallsrate von $\frac{1}{0.5s}$ ausgeht.

Ein weiteres Problem ist die Genauigkeit des Geigerzählers. Da der verwendete Zähler recht ungenau ist und eine relativ hohe Totzeit hat, lässt sich aus einem einzelnen Zeitintervall nicht unbegrenzt viel Information gewinnen. In diesem Fall basiert der „Zufall“ nicht mehr auf dem echten, indeterministischen Zerfall eines Atoms, sondern auf den deterministischen Fehlern des Geräts. Eine passende Metapher wäre: Man kauft einen Film auf einer Streaming-Plattform, der nur in SD verfügbar ist, möchte ihn aber in HD anschauen. Um die gewünschte Auflösung zu erreichen, müsste man zusätzliche Informationen „erfinden“ – genau wie bei der Aufteilung der Exponentialverteilung in Quantile. Man kann diese nicht einfach in beliebig viele einteilen, so wie man ein Film nicht in beliebig viele Pixel einteilen kann.

10. Fazit

Es wurde eine TRNG basierend auf Hintergrundstrahlung der Radioaktivität erschaffen. Seine Zufallsinformation ist unverschoben, d.h. es gibt kein Übergewicht an 0en oder 1en.
Durch seinen Indeterminismus ist es zudem selbst mit einer unbegrenzten Menge an vorherigen Bits theoretisch unmöglich, zukünftige voherzusagen. Praktisch könnte ein Neuronales Netzwerk vielleicht mit hinreichend Trainingsbits aufgrund von in Abschnitt 9.2 beschriebenen Fehlerquellen bestimmte Muster erkennen und damit seine Vorhersagen verbessern.

Für die Verhinderung einer Verzerrung der Bits wird ein eigen erstellter Quantil-Algorithmus verwendet. Um zu prüfen, ob sich die Zerfallsrate geändert hat, was sich negativ auf die Qualität der

Bits auswirken würde, wurde der Welch t-Test genutzt. Um auf erstellte Bits zuzugreifen wurde ein http-Server programmiert.

Als schwieriger als zunächst geplant stellte sich die Suche nach einem geeigneten Geigerzähler heraus. Bei den meisten handelsüblichen Geigerzähler ist es nicht vorgesehen, auf die Spannung zuzugreifen und damit auf die Spannungsänderung, die bei einer Detektion entsteht. Zunächst wurden zwei Geigerzähler von Schulen ausgeliehen, die aber die eben genannten Probleme aufwiesen. Nach dem Kauf eines eigentlich geeigneten DIY-Kit Geigerzählers gab es über einen langen Zeitraum immer noch Probleme, da dieser fast keine Dokumentation besitzt. Es wurde dann doch eine Schnittstelle gefunden, nach dem Ausprobieren mehrerer Mikrocomputer und Mikrocontroller. Der ESP-32, der usprünglich geplante Mikrocontroller, hatte einen Kurzschluss, nachdem er an den falschen Pin des DIY-Kits angeschlossen wurde. Daraufhin wurden ein Rasberry Pi 5 und schließlich ein Rasberry Pi 4 verwendet, mit dem es funktionierte. Während des Prozesses wurde zudem ein Logic-Converter unnötigerweise verwendet, der dann aber ebenso versagte und durch fehlerhaftes Löten defekt wurde. Außerdem gab ein Geigerzähler DIY-Kit den Geist auf und ein neues musste besorgt werden.

Bibliografie

- AG Forschungsmethoden und Kognitive Psychologie, Institut für Psychologie, Universität Bremen: Die Familie der t-Tests, URL: https://statsbyrandolph.psychologie.uni-bremen.de/Statistik_I/statistik_I_teil_12.html, abgerufen am 02.01.2026
- Andrade, C.: The P value and statistical significance: Misunderstandings, explanations, challenges, and alternatives. **Indian J Psychol Med**, URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6532382/, abgerufen am 02.01.2026
- Bundesamt Strahlenschutz: Wo stehen die Sonden?, URL: https://odlinfo.bfs.de/ODL/DE/themen/wo-stehen-die-sonden/karte/karte_node.html, abgerufen am 21.12.2025
- Hilgevoord, Jan: The uncertainty principle for energy and time. **American Journal of Physics**, S. 1454–1455, URL: http://www.stat.physik.uni-potsdam.de/~pikovsky/teaching/stud_seminar/ajp_uncert_energy_time1.pdf, abgerufen am 24.12.2025
- L'Ecuyer, Pierre: Random number generation. **Papers, No. 2004,21**, Humboldt-Universität zu Berlin, Center for Applied Statistics and Economics (CASE), Berlin 2004, S. 9–10, URL: https://www.econstor.eu/bitstream/10419/22195/1/21_pl.pdf, abgerufen am 25.12.2025
- Ma, Xiongfen et al.: Quantum random number generation. **npj Quantum Information** 2016; 2:2, 28.06.2016
- Obregon, Alexander: Generating Pseudo-Random Numbers with Java's Random Class, URL: https://medium.com/%40AlexanderObregon/generating-pseudo-random-numbers-with-javas-random-class-58a153d18c99, abgerufen am 01.11.2025
- Penn State University: The Run Test, URL: https://online.stat.psu.edu/stat415/lesson/21, abgerufen am 23.12.2025
- Ridha, Ali A.: Chapter Five (Radioactivity), URL: https://www.researchgate.net/publication/307588102_Chapter_Five_Radioactivity, abgerufen am 24.12.2025
- Ruschen, Daniel; Schrey, Moritz; dateFreese, Johannes; Heisterklaus, Iris: Vorheriger TRNG, S. 2, URL: https://poster.fel.cvut.cz/poster2017/proceedings/Poster_2017/Section_EI/EI_040_Ruschen.pdf, abgerufen am 25.12.2025
- Unitychain: Provable Randomness – How to Test RNGs, URL: https://medium.com/unitychain/provable-randomness-how-to-test-rngs-55ac6726c5a3, abgerufen am 11.11.2025
- Universität Graz: Probabilistic Models for Radioactive Decay, S. 1, URL: https://imsc.uni-graz.at/keeling/modII_ss19/radiodecay.pdf, abgerufen am 25.12.2025
- Vigna, Sebastiano: It Is High Time We Let Go Of The Mersenne Twister, URL: https://arxiv.org/html/1910.06437v3, abgerufen am 25.12.2025
- Wikipedia: Entropy (computing), URL: https://en.wikipedia.org/wiki/Entropy_%28computing%29, abgerufen am 25.12.2025
- Wikipedia: Exponential Distribution, URL: https://en.wikipedia.org/wiki/Exponential_distribution, abgerufen am 02.01.2026

Persönliche Unterstützung

- Leonhard, Prof. Dr. Möckl, Biophysiker, Friedrich-Alexander-Universität, Erlangen-Nürnberg; Art der Unterstützung: Beratung, welche Art von TRNGs realistisch in unser Budget passen