Blatt02

November 14, 2023

---

1 Rechnernutzung in der Physik

Institut für Experimentelle Teilchenphysik
Institut für Theoretische Teilchenphysik
Prof. G. Quast, Prof. M. Steinhauser
Dr. A. Mildenberger, Dr. Th. Chwalek
Ilias Seite zum Kurs
WS 2023/24 – Blatt 02
Abgabe: Montag 20.11.2023 bzw. Dienstag 21.11.2023

Gemäß der Abfolge des Stoffs in der Vorlesung üben Sie auf dem zweiten Blatt den Umgang mit Monte-Carlo-(MC)Methoden und die Erstellung von Monte-Carlo-Simulationen. Die MC-Simulationen sind ein wesentlicher Bestandteil der modernen Physik und kommen in jedem Fachgebiet zum Einsatz – immer dann wenn es nicht möglich oder ineffiezient ist, ein Problem mit vielen gekoppelten Freiheitsgraden analytisch oder mit anderen numerischen Methoden zu lösen. Beispiele für solche Probleme sind über die in der Vorlesung hinaus erwähnten Anwendungsfällen z.B. die Faltung von Proteinen, die Entwicklung von Galaxien oder Teilchenkollisionen in einem Teilchenbeschleuniger.

1.0.1 Erinnerung: Grundsätzliches Verfahren der Monte Carlo Methoden

1. Erzeuge Folge von Zufallszahlen $r_1, r_2, \dots, r_m$, gleichverteilt in $[0, 1[$.
2. Transformiere diese Foge zur Erzeugung einer anderen Sequenz $x_1, x_2, \dots, x_n$, die einer beliebigen anderen Verteilugsdichte $f(x)$ folgen. (Anmerkung: x, x_i können auch Vektoren sein.)
3. Aus den x_i werden dann Eigenschaften von $f(x)$ bestimmt. Zum Beispiel entspricht der Anteil der x_i mit $a \leq x_i \leq b$ näherungsweise dem Integral $\int_a^b f(x)\mathrm{d}x$.

Im ersten Aufgabenteil beschäftigen Sie sich mit einem einfachen Verfahren zur Erzeugung einer gewünschten Sequenz $x_1, x_2, \dots, x_n$ aus gleichverteilten Zufallszahlen, der **Verwerfungsmethode**. In der Vorlesung haben Sie bereits das wohl bekannteste Beispiel, die Berechnung der Kreiszahl π, kennengelernt. In dem Beispiel werden Zufallszahlen (r_1, r_2) gleichmäßig in der Einheitsfläche erzeugt (durch *np.random.random(size=2)*) und dann mithilfe der Bedingung, dass das Zufallszahlenpaar innerhalb des Einheitskreises liegt ($r_1^2 + r_1^2 \leq 1^2$), die Kreiszahl π approximiert.

Mit Hilfe der Verwerfungsmethode können recht einfach auch hochdimensionale, bestimmte Integrale nährungsweise numerisch berechnet werden. In dieser Aufgabe sollen Sie die Methodik des

Beispiels nutzen, um das Volumen V_d von d-dimensionalen Kugeln in Abhängigkeit von der Anzahl der Dimensionen d zu berechnen.

1.1 a) Berechnung der Einheitsvolumen

Berechnen Sie für $d \in [1, 10]$ jeweils das Volumen der d-dimensionalen Einheitskugel.

Erzeugen Sie hierfür für jeden Schritt 10'000 Zufallszahlen gleichverteilt im entsprechenden d-dimensionalen Einheits-Hyperwürfel. Berechnen Sie mithilfe einer geeigneten Bedingung das Volumen des eingeschlossenen Kugelsegments im Einheits-Hyperwürfel und anschließend das volle Volumen der d-dimensionalen Einheitskugel.

> Hinweis: Vergewissern Sie sich, welche *NumPy* Methode in *np.random* hier nützlich sein kann.

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.special as sp
```

[ ]:
```
# set dimensions
dmax = 10  # maximum
dim = np.arange(1., dmax, dtype=int)
```

[ ]:
```
# implement MC method
npoints = 10000      # number of d-dimensional points to test
Vmc = []             # array holding results
```

1.2 b) Abschätzung der statistischen Unsicherheit und Darstellung des Ergebnisses

Bestimmen Sie die statistische Unsicherheit σ_{V_d} der Ergebnisse. Nutzen Sie dazu aus, dass bei N Versuchen die Anzahl N_{in} der innerhalb der Kugel liegenden Punkte einer Binomialverteilung $P(N_{in}; p, N)$ folgt. Die Unsicherheit ergibt sich aus der Varianz der gemessenen Größe $\sigma^2 = Var[\hat{p}]$, sowie der Varianz unserer binomialverteilten Zufallsvariable N_{in} und der Näherung $p \simeq \hat{p} = \frac{N_{in}}{N}$.

> Hinweis: Konstanten können aus der Varianz $Var[x]$ herausgezogen werden, siehe dazu Rechenregeln der Varianz.
>
> Die Varianz einer binomialverteilten Zufallsvariable haben Sie auf dem ersten Blatt bereits verwendet.
>
> Vergessen Sie nicht, die Unsicherheit auf $\hat{p}$ so zu skalieren, dass Sie die Unsicherheit σ_{V_d} erhalten.

[ ]:
```
# uncertainty on MC results using binomial statistics
```

Tragen Sie Ihre Ergebnisse für V_d und die Unsicherheit σ_{V_d} in ein Diagramm ein (plt.errorbar() wird hier nützlich sein). Die Funktion *VofSphere(d, r=1)* gibt das Volumen von d-dimensionalen Kugeln zurück, wie es sich mit mathematischen Integrationsmethoden ergibt. Tragen Sie auch das "theoretische" Ergebnis in die Grafik ein.

[ ]:
```python
def VofSphere(d, r=1):
    """
      calculate volume of a d-dimensional sphere with radius r

      Args:
        d: dimension
        r: radius

      Returns:
        (array of) float: d-dimensional Volume
    """
    V = r**d * np.pi**(d/2) / sp.gamma(d/2 + 1)
    return V

# Volume from mathematical formula


# plot results (with matplotlib)



plt.xlim(0, dmax+1.)
plt.ylim(1.5, 6.)
plt.xlabel('Dimension', fontsize='x-large')
plt.ylabel('Volumen der Kugel', fontsize='x-large')
plt.legend(loc='best')
plt.title("Volumen der d-dimensionalen Kugel")
plt.show()
```

In der zweiten Aufgaben beschäftigen Sie sich mit der Transformationsmethode. In der ersten Teilaufgabe werden Sie den Zusammenhang zwischen der Verteilung von Zufallszahlen und den dazu gehörigen Wahrscheinlichkeitsdichten vertiefen. Im zweiten Aufgabenteil werden Sie dann selbst korrelierte Zufallszahlen erzeugen, da dies in den üblichen *Python* Bibliotheken nicht zu finden ist, die Notwendigkeit jedoch in der Praxis häufig gegeben ist.

1.3 a) Erzeugung von einfachen Verteilungen

Sie arbeiten mit den gleichverteilten Zufallszahlen $r \in \mathrm{U}_{[0,1[}$. Im Folgenden betrachten Sie zwei Fälle: * Sie kennen die Verteilung $t_i = t(r_i)$ und sollen die zugrunde liegende Wahrscheinlichsdichte $g(t)$ bestimmen. * Sie kennen die Wahrscheinlichkeitsdichte $g(t)$ und sollen eine Verteilung $t_i = t(r_i)$ berechnen.

Erzeugen Sie in den folgenden Teilaufgaben immer 10'000 gleichverteilte Zufallszahlen im Intervall $[0, 1[$ und bestimmen Sie die fehlende Größe mithilfe der Transformationsmethode. Zeichen Sie anschließend in ein Diagramm die Verteilung $t(r)$ (Histogramm) und die Wahrscheinlichkeitsdichte $g(t(r))$ ein.

1.3.1 i) Gegeben: $g(t) = \frac{1}{t}$ – Gesucht: $t(r)$ für $t = 1 \dots 2.75$

[ ]:
```
# generate histogramm entries


# compute PDF


# plot histogramm and pdf


# make plot nice
plt.xlabel('$t(r)$', fontsize='x-large')
plt.ylabel('$g(t(r))$')
plt.legend(loc='upper right')

plt.show()
```

1.3.2 ii) Gegeben: $t(r) = r^2$ – Gesucht: $g(t)$ für $t = 0.01 \dots 1$

[ ]:
```
# generate histogramm entries


# compute PDF


# plot histogramm and pdf


# make plot nice
plt.xlabel('$t(r)$')
plt.ylabel('$g(t(r))$')
plt.legend(loc='upper right')

plt.show()
```

1.4 b) Erzeugung korrelierter, normalverteilter Zufallszahlen

Dieser Aufgabenteil soll Ihnen bewusst machen, wie Sie aus unkorrelierten normalverteilten Zufallszahlen korrelierte Zufallszahlen erzeugen. Die Herleitung des Prinzips wird in der 3. Vorlesung diskutiert und soll hier auf den einfache Fall von zwei gaußverteilten Zufallszahlen angewandt werden.

Erzeugen Sie zwei Sätze aus 10'000 standard-normalverteilten Zufallszahlen $\vec{u}_{1,2}$ ($\mu_{1,2} = 0$ und $\sigma_{1,2} = 1$) und transfomieren Sie diese zu korrelierten Zufallszahlen $\vec{x}_{1,2}$ mit einem Korrelationskoeffizienten von $\rho = 0.5$. > Hinweis: Die Methode *np.dot()* kann bei einem Matrix-Vektor Produkt hilfreich sein.

[ ]:
```
# implement transformation
```

Stellen Sie nun die beiden Sätze an Zufallszahlen geeignet dar. Es bietet sich an, für jedes Paar jeweils ein zweidimensionales Histogramm zu erstellen. Berechnen Sie anschließend den Korrelationskoeffizienten aus dem neu erzeugten Satz an Zufallszahlen, um Ihr Vorgehen zu überprüfen. > Hinweis: Das Paket *PhyPraKit* enthält einige hilfreiche Methoden zur Bearbeitung der Aufgabe.

[ ]:
```
# plot randomnumbers and calculate correlationcoefficient
import PhyPraKit as ppk
```

In der letzten Aufgabe kombinieren Sie Ihr erlangtes Wissen aus der Transformations- und der Verwerfungsmethode. Die Kombination aus beiden Methoden wird häufig als Majorantenmethode (*engl.: Importance Sampling*) bezeichnet, welche eine deutlich effizientere Methode zum Erzeugen einer benötigten Verteilung ist.

Im Folgenden betrachten Sie eine Verteilung, die Ihnen in ähnlicher Form möglicherweise im Fortgeschrittenenpraktikum begegen wird:

$$f(x) = \sin^2(\pi x)e^{-x}$$

im Intervall $[0, 2\pi]$.

Dazu erzeugen Sie zunächst mithilfe der Transformationsmethode Zufallszahlen gemäß der Verteilung der Majorante und wählen dann mit der Verwerfungsmethode davon eine passende Anzahl aus, um die gewünschten Verteilung zu erhalten.

1.5 a) Transformationsmethode zum Erhalt der Majorante

Erzeugen Sie mithilfe der Transformationsmethode exponentiell verteilte Zufallszahlen als Majorante

$$m(x) = e^{-x}.$$

Überzeugen Sie sich zunächst davon, dass Ihre Majorante die gesuchte Verteilung vollständig umschließt, indem Sie die Majorante $m(x)$, die gewünschte Verteilung $f(x)$ und die erzeugten Zufallszahlen histogrammiert in einer Abbildung darstellen. > Hinweis: Es bietet sich an, die Verteilungen und Verteilungsdichten als Funktion zu implementieren, da Sie sie häufiger aufrufen und gegebenenfalls im Praktikum wiederverwenden werden können.

[ ]:
```
# build functions f(x), m(x) and PDF of m(x)


# sample majorant


# plot functions and random number
```

1.6 b) Verwerfungsmethode zum Erhalt der gewünschten Verteilung

Wenden Sie nun die Verwerfungsmethode an, um die gewünschte Verteilung zu erhalten. Erzeugen Sie für jede der Zufallszahlen x_i aus der Majorante eine gleichverteilte Zufalls Zahl r_i aus dem Interval $[0, 1]$. Behalten Sie x_i genau dann, wenn die Bedingung

$$r_i \cdot m(x_i) \leq f(x_i)$$

erfüllt ist. So erhalten Sie nur die Zufallszahlen, die gemäß f(x) verteilt sind, da Sie für jeden Wert von x nur den Bruchteil

$$\frac{m(x)}{f(x)}$$

der x-Werte behalten. Stellen Sie das Ergebnis, gemäß der vorherigen Teilaufgabe, geeignet dar. > Tipp: Mithilfe von *plt.hist([hist1, hist2], stacked=True,…)* können Sie sowohl die angenommenen als auch die abgelehnten Zufallszahlen in einem Histogramm darstellen.

[ ]:
```
# sample 2nd set of random numbers for accept-reject-method


# accept-reject-method


# plot functions and random numbers
```