Serie 4

Aufgabe 4.1

a) Gegeben sind zwei unabhängige Zufallsvariable X und Y mit den Kennwerten $\mu_X = 40$, $\sigma_X = 15$, $\mu_Y = 85$ und $\sigma_Y = 18$. Berechnen Sie $\mathrm{E}(X + 2Y)$, $\mathrm{Var}(X + 2Y)$ und $\mathrm{E}(X^2)$.

b) Ein Werk produziert rechteckige Glasscheiben, deren Länge X und Breite Y (in mm gemessen) voneinander unabhängig produktionsbedingten Schwankungen unterliegen. Es gilt $\mu_X = 1000$, $\sigma_X = 0.02$, $\mu_Y = 500$, $\sigma_Y = 0.01$. Wie gross sind Erwartungswert und Standardabweichung des Umfangs U?

c) (**Zusatzaufgabe**) Bestimmen Sie die Wahrscheinlichkeitsdichte $f_X(x)$ der Zufallsvariablen $X = Z^2$, wobei $Z \sim \mathcal{N}(0, 1)$.

Aufgabe 4.2

Erzeugen Sie mit **`norm.rvs()`** $m = 500$ Stichproben aus einer Standardnormalverteilung mit Umfang $n = 5$. Speichern Sie die Stichproben als $(n \times m)-$ Matrix ab. ***Python**-Hinweis:*

```
import matplotlib.pyplot as plt
from scipy.stats import norm
import numpy as np

n = 5
m = 500

ran = np.array(norm.rvs(size=n*m))

sim = ran.reshape((n,m))
```

a) Stellen Sie die Stichproben als Runs graphisch dar, und staunen Sie über die zahlreichen möglichen Verläufe.

```
plt.plot(sim)
plt.show()
```

b) Berechnen Sie nun für jede Stichprobe den Mittelwert $\bar{x}$, und erzeugen Sie ein Histogramm der Mittelwerte. Aus der Theorie wissen Sie, dass die Mittelwerte

normalverteilt mit Parameter $\mu = 0$ und Standardabweichung $\sigma = \frac{1}{\sqrt{n}}$ sein müssten. Überprüfen Sie dies graphisch mit Hilfe von Histogrammen und der theoretischen Wahrscheinlichkeitsdichtekurve

```
plt.hist(sim.T, bins=20, density=True, edgecolor="black",
facecolor="white")

x = np.linspace(-4, 4, num=100)
y = norm.pdf(x)

plt.plot(x,y)
plt.title("Histogramm sim")
plt.show()

sim_mean = sim.mean(axis=0)

plt.hist(sim_mean, density=True, edgecolor="black",
facecolor="white")

x = np.linspace(-4, 4, num=100)

y = norm.pdf(x, loc=0, scale=1/np.sqrt(n))

plt.plot(x,y)
plt.title("Histogramm sim_mean")

plt.show()
```

c) Wiederholen Sie die Aufgabe für $n = 2, 10, 100$.

Aufgabe 4.3

a) Eine Elektronik-Firma stellt Widerstände her, die einen mittleren Widerstand von $100\,\Omega$ und eine Standardabweichung von $10\,\Omega$ haben. Die Widerstände sind normalverteilt. Bestimmen Sie die Wahrscheinlichkeit, dass sich für eine zufällige Stichprobe von $n = 25$ Widerständen ein mittlerer Widerstand unter $95\,\Omega$ ergibt.

b) Nehmen Sie an, dass eine Zufallsvariable X einer uniformen Verteilung folgt, und zwar $X \sim \text{Uniform}[4, 6]$. Wie lautet die Verteilung der Zufallsvariablen $\overline{X}_{40}$, also des Mittelwertes einer Stichprobe vom Umfang $n = 40$.

Aufgabe 4.4

Es seien $\{X_i\}_{1 \leq i \leq 50}$ unabhängige und normalverteilte Zufallsvariablen mit Erwartungswert $\mu = 1$ und Standardabweichung $\sigma = 2$. Darüber hinaus sind folgende Zufallsvariablen definiert:

$$S_n = X_1 + X_2 + \cdots + X_n$$

und

$$\overline{X}_n = \frac{1}{n}(X_1 + X_2 + \cdots + X_n) = \frac{S_n}{n}$$

Dabei ist $n = 50$.

a) Bestimmen Sie die Parameter der Normalverteilung von S_n sowie $\overline{X}_n$.

b) Berechnen Sie die Wahrscheinlichkeit $P\left(\mathrm{E}[X_1] - 1 \leq X_1 \leq \mathrm{E}[X_1] + 1\right)$.

c) Berechnen Sie $P\left(\mathrm{E}[S_n] - 1 \leq S_n \leq \mathrm{E}[S_n] + 1\right)$.

d) Berechnen Sie $P\left(\mathrm{E}[\overline{X}_n] - 1 \leq \overline{X}_n \leq \mathrm{E}[\overline{X}_n] + 1\right)$.

Aufgabe 4.5

Wir betrachten einen Datensatz, bei welchem zwei Methoden zur Bestimmung der latenten Schmelzwärme von Eis verglichen werden. Wiederholte Messungen der freigesetzten Wärme beim Übergang von Eis bei $-0.7°\mathrm{C}$ zu Wasser bei $0°\mathrm{C}$ ergaben die folgenden Werte (in cal/g):

Methode A	79.98	80.04	80.02	80.04	80.03	80.03	80.04	79.97	80.05
Methode A	80.03	80.02	80.00	80.02					
Methode B	80.02	79.94	79.98	79.97	79.97	80.03	79.95	79.97	

a) Berechnen Sie die arithmetischen Mittelwerte der beiden Methoden und geben Sie für jede Methode den absoluten Fehler an.

b) Wie lauten die relativen Fehler?

Aufgabe 4.6

Die Auswertung eines Integrals

$$\int_a^b f(x)\,\mathrm{d}x$$

kann sehr oft nicht analytisch erfolgen. Der gebräuchlichste Ansatz in diesem Fall besteht darin, das Integral numerisch zu berechnen. Dazu existieren verschiedene Computerprogramme. Eine andere geläufige Methode, um ein solches Integral zu berechnen, ist die sogenannte **Monte Carlo Methode**. Man generiert dabei uniform verteilte Zufallsvariablen auf dem Intervall $[a, b]$, d. h., $X_1, X_2, \ldots, X_n$ i.i.d. $\sim$ Uniform$([a, b])$ und berechnet

$$\frac{b-a}{n} \sum_{i=1}^{n} f(X_i) \,.$$

Dass dieser Ausdruck in etwa $\int_a^b f(x)\,\mathrm{d}x$ ist, möchten wir im Folgenden verstehen. Aufgrund des *Gesetzes der grossen Zahlen* gilt für grosse n

$$(b-a) \cdot \frac{1}{n} \sum_{i=1}^{n} f(X_i) \approx (b-a) \cdot \mathrm{E}[f(X)] \,.$$

Der Erwartungswert von $f(X)$ für $X \sim$ Uniform$([a, b])$ kann aber auch geschrieben werden als

$$\mathrm{E}[f(X)] = \int_a^b f(x) \frac{1}{b-a}\,\mathrm{d}x \,,$$

wobei $\frac{1}{b-a}$ die Wahrscheinlichkeitsdichte der Wahrscheinlichkeitsverteilung Uniform$([a, b])$ ist. Somit gilt

$$\frac{b-a}{n} \sum_{i=1}^{n} f(X_i) \approx (b-a) \cdot \mathrm{E}[f(X)] = (b-a) \cdot \int_a^b f(x) \frac{1}{b-a}\,\mathrm{d}x = \int_a^b f(x)\,\mathrm{d}x \,.$$

Berechnen Sie folgendes Integral

$$I(f) = \frac{1}{\sqrt{2\pi}} \int_0^1 \mathrm{e}^{-x^2/2}\,\mathrm{d}x \,.$$

Berechnen Sie das Integral, indem Sie 1000 uniform über das Intervall $[0, 1]$ verteilte Zahlen $X_1, \ldots, X_{1000}$ mit der **`Python`** -Funktion **`uniform.rvs(...)`** generieren. Berechnen Sie den genauen numerischen Wert des Integrals mit der **`Python`** -Funktion **`norm.cdf(...)`**.

Kurzlösungen vereinzelter Aufgaben

A 4.1:

c) $f_X(x) = \frac{x^{-1/2}}{\sqrt{2\pi}} \mathrm{e}^{-x/2}, \quad x \geq 0$

A 4.3:

a) 0.0062

b) $\overline{X}_{40} \sim \mathcal{N}(5, 1/120)$

A 4.5:

a) Methode A: (80.02 ± 0.01)cal/g
Methode B: (79.98 ± 0.01)cal/g

b) Methode A: 80.02cal/g $\pm\, 8 \cdot 10^{-3}\%$
Methode B: 79.98cal/g $\pm\, 1 \cdot 10^{-2}\%$

Musterlösungen zu Serie 4

Lösung 4.1

a) $\mathrm{E}(X+2Y) = \mu_X + 2\mu_Y = 210\,,\quad \mathrm{Var}(X+2Y) = \sigma_X^2 + 4\sigma_Y^2 = 1521\,,\, \mathrm{E}(X^2) = \mathrm{Var}(X) + (\mathrm{E}(X))^2 = \sigma_X^2 + \mu_X^2 = 1825\,.$

b) $U = 2X+2Y;\quad \mathrm{E}(U) = 2\,\mathrm{E}(X) + 2\,\mathrm{E}(Y) = 2\mu_X + 2\mu_Y = 3000\,, \sigma_U = \sqrt{4\sigma_X^2 + 4\sigma_Y^2} = 0.0447\,.$

c) Wir haben $X = Z^2$, wobei $Z \sim \mathcal{N}(0,1)$. Dann ist

$$\begin{aligned} F_X(x) &= P(X \le x) \\ &= P(Z^2 \le x) \\ &= P(-\sqrt{x} \le Z \le \sqrt{x}) \\ &= \Phi(\sqrt{x}) - \Phi(-\sqrt{x})\,. \end{aligned}$$

Wir erhalten die Wahrscheinlichkeitsdichte von X, indem wir die kumulative Verteilungsfunktion $F_X(x)$ nach x ableiten. Da $\Phi'(x) = \varphi(x)$, ergibt sich mit der Kettenregel

$$\begin{aligned} f_X(x) &= \frac{1}{2}x^{-1/2}\varphi(\sqrt{x}) + \frac{1}{2}x^{-1/2}\varphi(-\sqrt{x}) \\ &= x^{-1/2}\varphi(\sqrt{x})\,, \end{aligned}$$

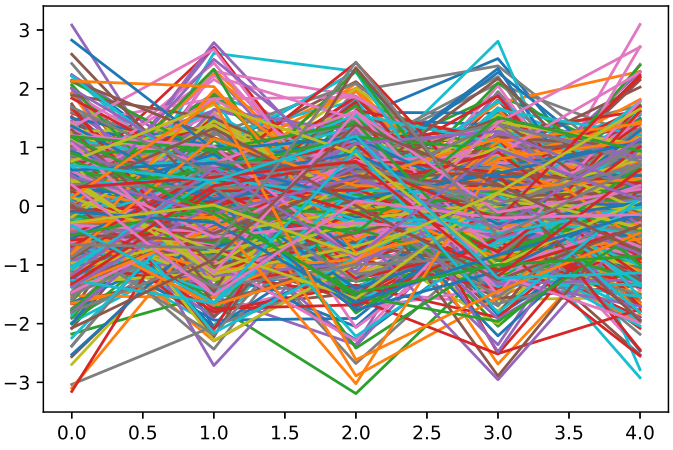
wobei wir im letzten Schritt die Symmetrie von φ benützt haben. Werten wir den letzten Ausdruck aus, so finden wir

$$f_X(x) = \frac{x^{-1/2}}{\sqrt{2\pi}}\mathrm{e}^{-x/2}\,,\quad x \ge 0\,.$$

Diese Wahrscheinlichkeitsdichte wird **Chi-Quadrat** Wahrscheinlichkeitsdichte mit einem Freiheitsgrad genannt.

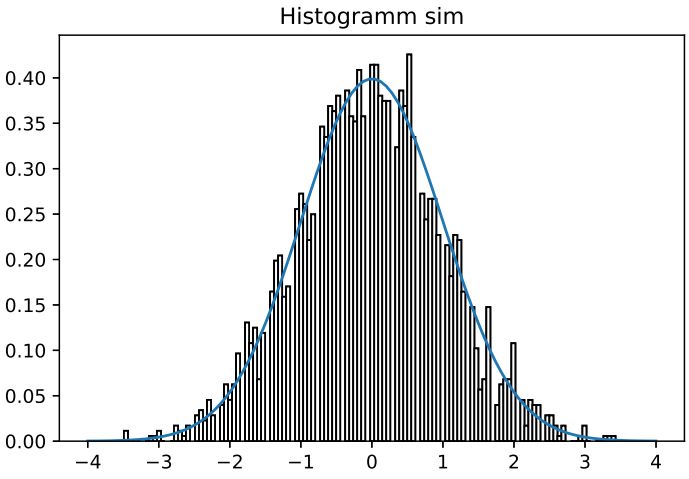
Lösung 4.2

a) (zu **R**)

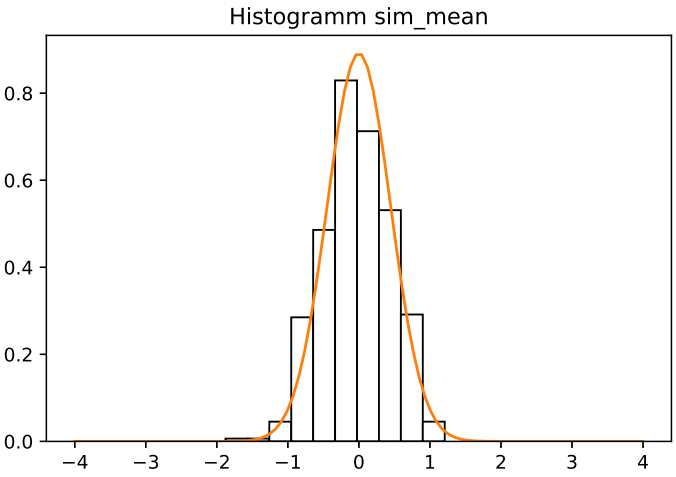


b) (zu **R**)

Für **sim**

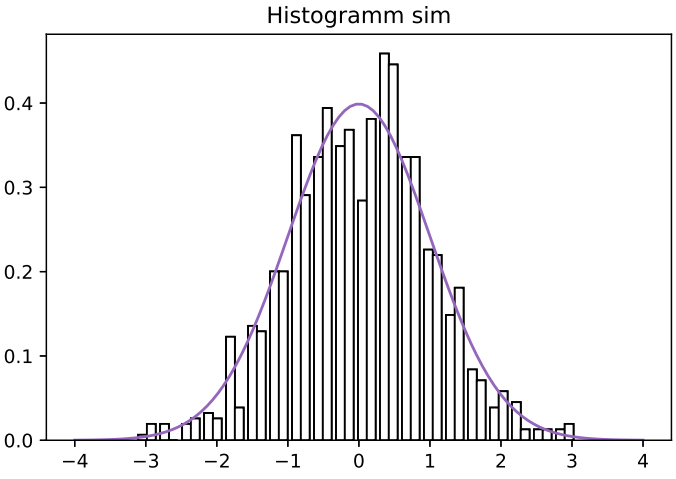


Und für **`sim_mean`**

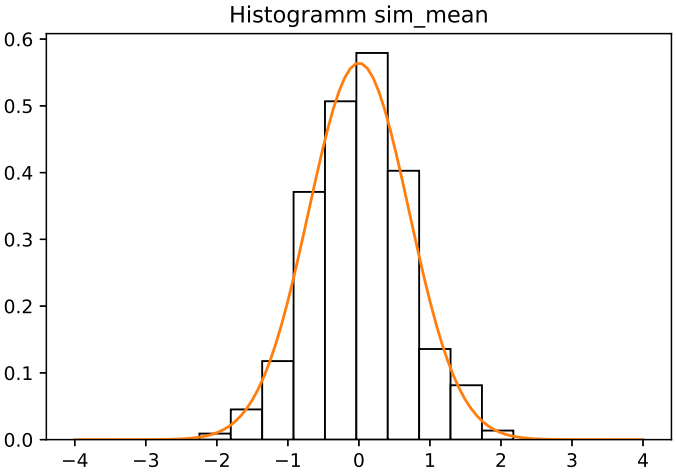


c) (zu **R**)

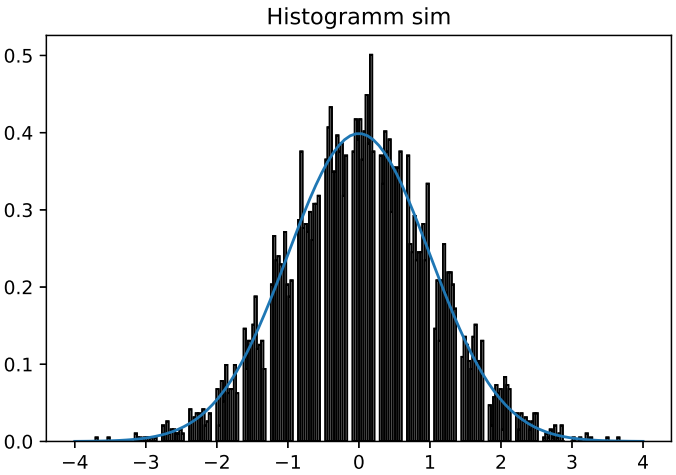
Für $n = 2$:



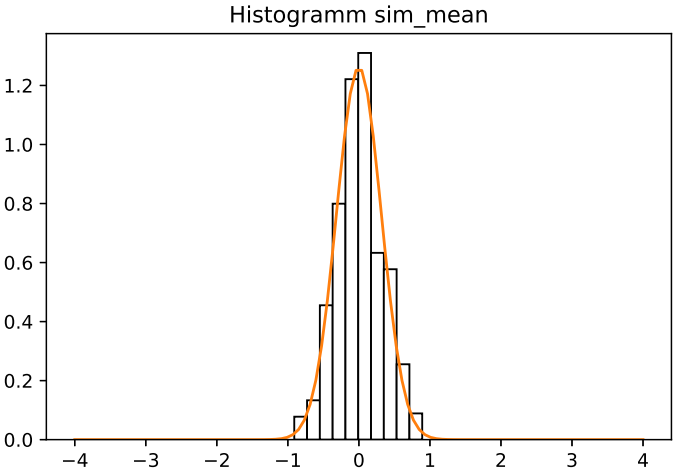
und


Histogramm sim_mean
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−4
−3
−2
−1
0
1
2
3
4

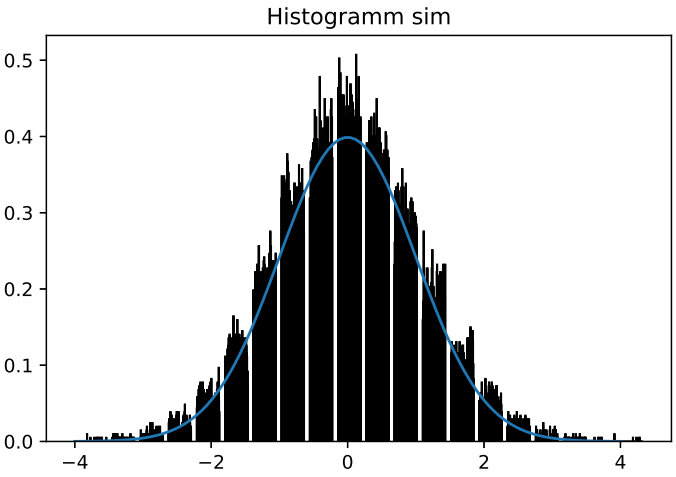
Für $n = 10$:


Histogramm sim
0.5
0.4
0.3
0.2
0.1
0.0
−4
−3
−2
−1
0
1
2
3
4

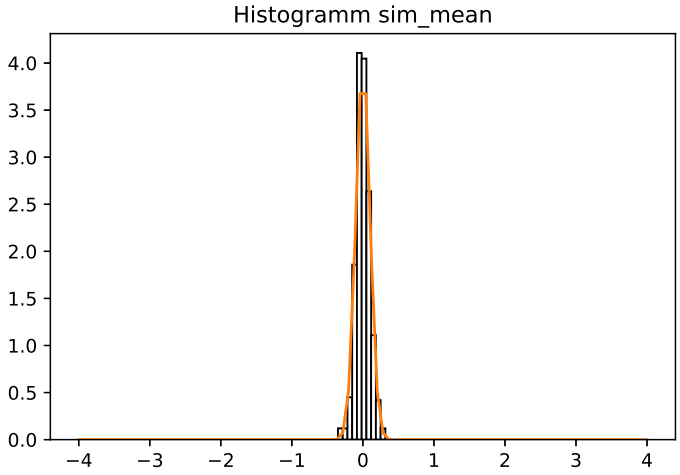
und


Histogramm sim_mean
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−4
−3
−2
−1
0
1
2
3
4

Für $n = 10$:


Histogramm sim
0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4

und



Lösung 4.3

a) Wir bezeichnen mit $\overline{X}_n$ den mittleren Widerstand in einer Stichprobe von n Widerständen, wobei $\overline{X}_n$ normalverteilt ist mit einem Mittelwert von $\mu_{\overline{X}_n} = \mu_X = 100\,\Omega$ und einer Standardabweichung von

$$\sigma_{\overline{X}_n} = \frac{\sigma_X}{\sqrt{n}} = \frac{10}{\sqrt{25}} = 2.$$

Somit ist $\overline{X}_n \sim \mathcal{N}(100, 4)$. Wir suchen nun die Wahrscheinlichkeit $P(\overline{X}_n < 95)$: (zu **R**)

```
from scipy.stats import norm

norm.cdf(x=95, loc=100, scale=2)

## 0.006209665325776132
```

b) Der Mittelwert und die Varianz von X lauten: $\mu_X = 5$ und $\sigma_X^2 = (6-4)^2/12 = 1/3$. Der zentrale Grenzwertsatz deutet darauf hin, dass die Verteilung von $\overline{X}_{40}$ näherungsweise normalverteilt ist mit dem Mittelwert $\mu_{\overline{X}_{40}} = 5$ und der Varianz $\sigma_{\overline{X}_n} = \sigma_X/\sqrt{n} = 1/\sqrt{120}$. Folglich gilt

$$\overline{X}_{40} \sim \mathcal{N}(5, 1/120)$$

Lösung 4.4

a) Für S_n gilt

$$\mathrm{E}[S_n] = \mathrm{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathrm{E}[X_i] = 50 \cdot 1$$

$$\mathrm{Var}(S_n) = \mathrm{Var}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathrm{Var}(X_i) = 50 \cdot 4 = 200$$

S_{50} ist normal-verteilt mit $\mathcal{N}(\mu_{S_{50}} = 50, \sigma^2_{S_{50}} = 200)$. Für $\overline{X}_n$ gilt

$$\mathrm{E}[\overline{X}_n] = \frac{1}{n} E\left[\sum_{i=1}^{n} X_i\right] = 1\,, \quad \mathrm{Var}(\overline{X}_n) = \frac{1}{n^2}\,\mathrm{Var}\left(\sum_{i=1}^{n} X_i\right) = 0.08$$

$\overline{X}_{50}$ ist normal-verteilt mit $\mathcal{N}(\mu_{\overline{X}_{50}} = 1, \sigma^2_{\overline{X}_{50}} = 0.08)$.

b) X_1 ist normal-verteilt mit $\mathcal{N}(\mu_{X_1} = 1, \sigma^2_{X_1} = 4)$. Wir suchen die Wahrscheinlichkeit $P(0 \leq X_1 \leq 2)$: (zu **R**)

```
from scipy.stats import norm

norm.cdf(x=2, loc=1, scale=2) - norm.cdf(x=0, loc=1, scale=2)

## 0.38292492254802624
```

c) Es gilt: $S_n \sim \mathcal{N}(n\mu, n\sigma^2_X)$, also $S_{50}\mathcal{N}(50, 200)$. Wir suchen die Wahrscheinlichkeit $P(49 \leq S_{50} \leq 51)$: (zu **R**)

```
import numpy as np

norm.cdf(x=51, loc=50, scale=np.sqrt(200)) - norm.cdf(x=49, loc=50, scale=n

## 0.05637197779701664
```

d) Es gilt: $\overline{X}_n \sim \mathcal{N}(\mu, \sigma^2_X/n)$, $\overline{X}_{50} \sim \mathcal{N}(1, 4/50)$. Wir suchen die Wahrscheinlichkeit $P(0 \leq \overline{X}_{50} \leq 2)$: (zu **R**)

```
norm.cdf(x=2, loc=1, scale=np.sqrt(4/50))
- norm.cdf(x=0, loc=1, scale=np.sqrt(4/50))

## 0.999593047982555
```

Lösung 4.5

a) Der arithmetische Mittelwert der Methode A ergibt sich aus (zu **R**)

```
import numpy as np
from pandas import Series
```

```
methode_A = Series([79.98 , 80.04 , 80.02 , 80.04 , 80.03 ,
80.03 , 80.04 , 79.97 , 80.05 , 80.03, 80.02 , 80.00 , 80.02])

methode_A.mean()

## 80.02076923076923
```

Den Standardfehler ermitteln wir mit (zu **R**)

```
methode_A.std() / np.sqrt(methode_A.size)

## 0.00664691353682878
```

Somit können wir den Mittelwert mit absolutem Fehler schreiben als

$$(80.02 \pm 0.01)\text{cal/g}$$

wobei wir beachten, dass wir gleich viele Dezimalstellen für den absoluten Fehler wie für den Mittelwert angeben, nämlich zwei. Für die Methode B finden wir (zu **R**)

```
methode_B = Series([80.02 , 79.94 , 79.98 , 79.97 , 79.97 , 80.03 , 79.95 ,

methode_B.mean()

## 79.97874999999999
```

Den Standardfehler ermitteln wir mit (zu **R**)

```
methode_B.std() / np.sqrt(methode_B.size)

## 0.011090133968017208
```

Somit können wir den Mittelwert mit absolutem Fehler schreiben als

$$(79.98 \pm 0.01)\text{cal/g}$$

b) Der relative Fehler der Methode A lautet (zu **R**)

```
methode_A.std() / np.sqrt(methode_A.size) / methode_A.mean()

## 8.30648542962636e-05
```

Somit können wir den Mittelwert mit relativem Fehler schreiben als

$$80.02\text{cal/g} \pm 8 \cdot 10^{-3}\%$$

wobei wir beachten, dass der absolute Fehler bloss eine Signifikanzstelle hat und der relative Fehler somit auch. Der relative Fehler der Methode B lautet (zu **R**)

```
methode_B.std() / np.sqrt(methode_B.size) / methode_B.mean()

## 0.00013866350709428705
```

Somit können wir den Mittelwert mit relativem Fehler schreiben als

$$79.98\text{cal/g} \pm 1 \cdot 10^{-2}\%$$

Lösung 4.6 Wir berechnen das Integral

$$I(f) = \frac{1}{\sqrt{2\pi}} \int_0^1 \mathrm{e}^{-x^2/2}\,\mathrm{d}x$$

mit der Monte-Carlo Methode, da analytisch keine Lösung in geschlossener Form existiert. Zuerst generieren wir 1000 uniform im Intervall $[0, 1]$ verteilte Zufallszahlen $X_1, \ldots, X_{1000}$ (zu **R**)

```
from scipy.stats import uniform
n = 1000
x = uniform.rvs(size=1000, loc=0, scale=1)
```

Danach werten wir den Integranden für alle 1000 Zufallszahlen aus (zu **R**)

```
import numpy as np
from scipy.stats import uniform
n = 1000
x = uniform.rvs(size=1000, loc=0, scale=1)
integrand = np.exp(-np.square(x)/2)
```

Das mit Monte-Carlo berechnete Integral ergibt dann den Wert (zu **R**)

```
from math import pi
import numpy as np
from scipy.stats import uniform
n = 1000
x = uniform.rvs(size=1000, loc=0, scale=1)
integrand = np.exp(-np.square(x)/2)
print(integral)


## 0.3423667056398362
```

Das Integral, das wir eben mit der Monte-Carlo Methode berechnet haben, ist natürlich die Differenz der kumulativen Verteilungsfunktion der Standardnormalverteilung ausgewertet an den Stellen 0 und 1. (zu **R**)

```
from scipy.stats import norm
integral = norm.cdf(1)-norm.cdf(0)
print(integral)


## 0.3413447460685429
```

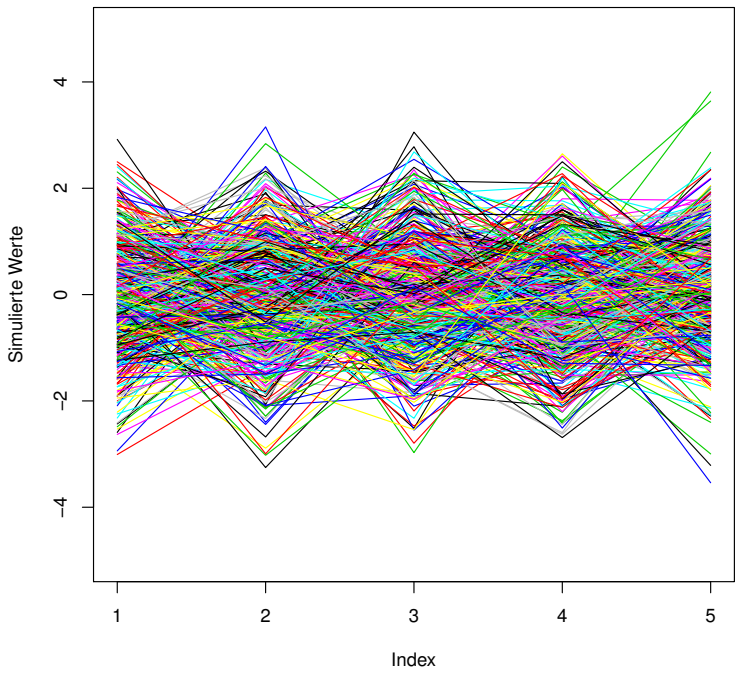
R-Code

Aufgabe 4.2

```
m <- 500  # Daten simulieren
n <- 5
set.seed(1)  #Zufallsgenerator initialisieren
data.sim <- matrix(rnorm(n = m * n, 0, 1), ncol = m,
    nrow = 5)
str(data.sim)


##  num [1:5, 1:500] -0.626 0.184 -0.836 1.595 0.33 ...
```

a) (zu **Python**)

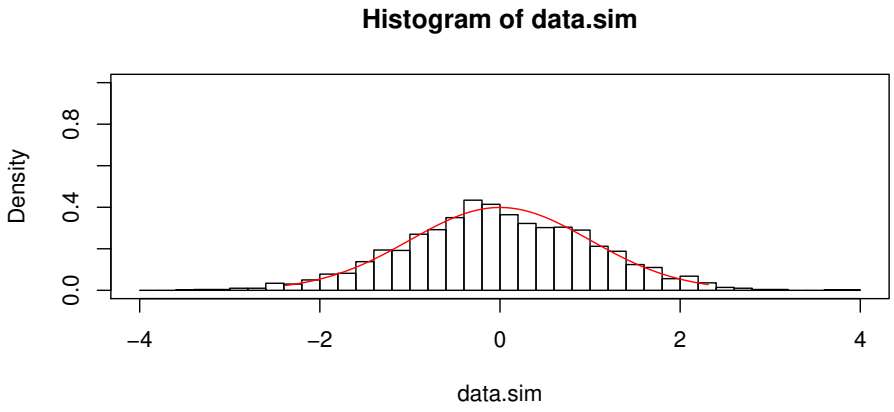
```
plot(data.sim[, 1], type = "l", ylim = c(-5, 5),
    ylab = "Simulierte Werte")
for (i in 1:m) {
    points(data.sim[, i], type = "l", col = i)
}
```

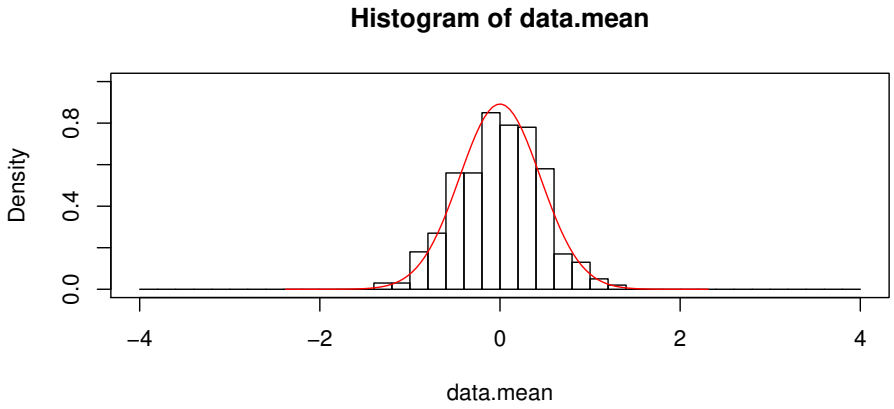


b) (zu **Python**)

```
# Parameter der Verteilung
data.mean <- apply(data.sim, MARGIN = 2, mean)
mean(data.mean)

## [1] -0.009344855

sd(data.mean)

## [1] 0.4585088
```

```
# Histogramme
op <- par(mfcol = c(2, 1))
hist(data.sim, freq = FALSE, breaks = seq(-4,
    4, by = 0.2), ylim = c(0, 1), xlim = c(-4,
    4))
curve(dnorm(x, mean = 0, sd = 1), from = min(data.mean) -
    1, to = max(data.mean) + 1, add = TRUE,
    col = "red")
box()
hist(data.mean, freq = FALSE, breaks = seq(-4,
    4, by = 0.2), ylim = c(0, 1), xlim = c(-4,
    4))
curve(dnorm(x, mean = 0, sd = 1/sqrt(n)),
    from = min(data.mean) - 1, to = max(data.mean) +
        1, add = TRUE, col = "red")
box()
par(op)
```



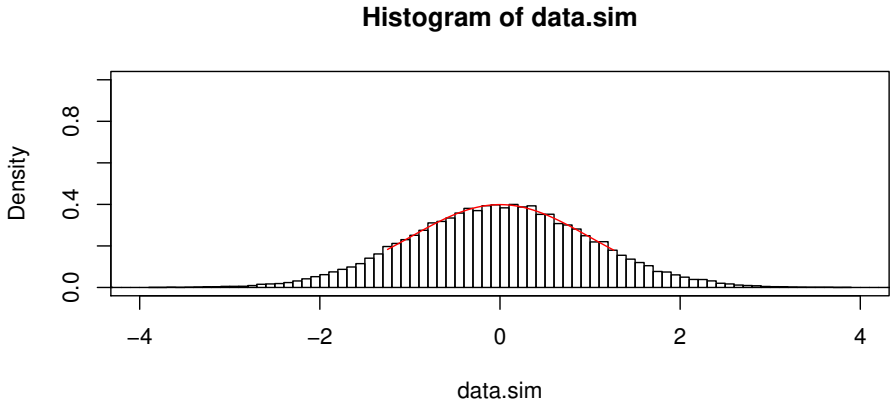


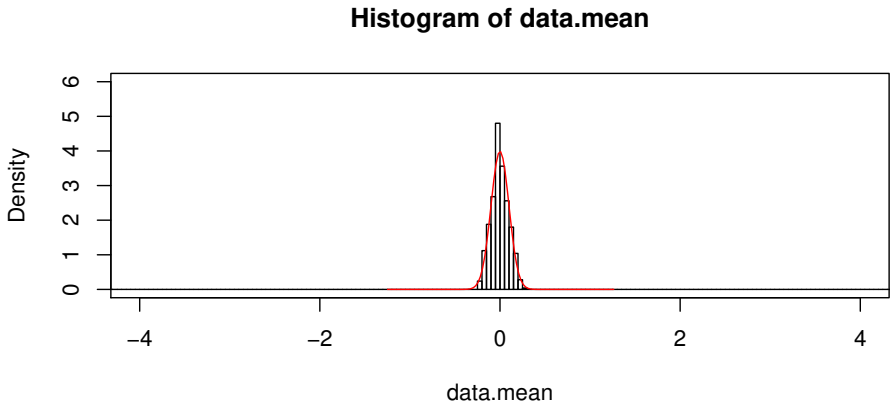
c) (zu **Python**)

```
# Daten simulieren
m <- 500
n <- 100
# Zufallsgenerator initialisieren
set.seed(1)
data.sim <- matrix(rnorm(n = m * n, 0, 1), ncol = m,
    nrow = n)
str(data.sim)

##  num [1:100, 1:500] -0.626 0.184 -0.836 1.595 0.33 ...
```

```
# Histogramme
data.mean <- apply(data.sim, MARGIN = 2,
    mean)
op <- par(mfcol = c(2, 1))
hist(data.sim, freq = FALSE, breaks = seq(-5,
    5, by = 0.1), ylim = c(0, 1), xlim = c(-4,
    4))
```

```r
curve(dnorm(x, mean = 0, sd = 1), from = min(data.mean) -
    1, to = max(data.mean) + 1, add = TRUE,
    col = "red")
box()
hist(data.mean, freq = FALSE, breaks = seq(-5,
    5, by = 0.05), ylim = c(0, 6), xlim = c(-4,
    4))
curve(dnorm(x, mean = 0, sd = 1/sqrt(n)),
    from = min(data.mean) - 1, to = max(data.mean) +
        1, add = TRUE, col = "red")
box()
par(op)
```





Aufgabe 4.3

a) (zu **Python**)

```r
pnorm(q = 95, mean = 100, sd = 2)
```

```
## [1] 0.006209665
```

Aufgabe 4.4

a)

b) (zu **Python**)

```r
pnorm(q = 2, mean = 1, sd = 2) - pnorm(q = 0, mean = 1,
    sd = 2)
```

```
## [1] 0.3829249
```

c) (zu **Python**)

```r
pnorm(q = 51, mean = 50, sd = sqrt(200)) - pnorm(q = 49,
    mean = 50, sd = sqrt(200))
```

```
## [1] 0.05637198
```

d) (zu **Python**)

```r
pnorm(2, 1, sqrt(4/50)) - pnorm(0, 1, sqrt(4/50))
```

```
## [1] 0.999593
```

Aufgabe 4.5

a) (zu **Python**)

```r
methode.A <- c(79.98, 80.04, 80.02, 80.04, 80.03, 80.03,
    80.04, 79.97, 80.05, 80.03, 80.02, 80, 80.02)
mean(methode.A)
```

```
## [1] 80.02077
```

(zu **Python**)

```r
sd(methode.A)/sqrt(length(methode.A))
```

```
## [1] 0.006646914
```

(zu **Python**)

```r
methode.B <- c(80.02, 79.94, 79.98, 79.97, 79.97, 80.03,
    79.95, 79.97)
mean(methode.B)

## [1] 79.97875
```

(zu **Python**)

```r
sd(methode.B)/sqrt(length(methode.B))

## [1] 0.01109013
```

b) (zu **Python**)

```r
sd(methode.A)/sqrt(length(methode.A))/mean(methode.A)

## [1] 8.306485e-05
```

(zu **Python**)

```r
sd(methode.B)/sqrt(length(methode.B))/mean(methode.B)

## [1] 0.0001386635
```

Aufgabe 4.6

(zu **Python**)

```r
n <- 1000
x <- runif(n, min = 0, max = 1)
```

(zu **Python**)

```r
integrand <- exp(-x^2/2)
```

(zu **Python**)

```r
1/(sqrt(2 * pi)) * sum(integrand)/n


## [1] 0.3433597
```

(zu **Python**)

```r
pnorm(1) - pnorm(0)
```

```
## [1] 0.3413447
```