universität wien

DIPLOMARBEIT / DIPLOMA THESIS

Titel der Diplomarbeit / Title of the Diploma Thesis

„Monte-Carlo-Integration – Eine Einführung mit Mathematica“

verfasst von / submitted by

Lukas Franz Ostermann

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Magister der Naturwissenschaften (Mag.rer.nat)

Wien, 2017 / Vienna, 2017

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 190 412 406
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Lehramtsstudium UniStG UF Physik UniStG UF Mathematik UniStG
Betreut von / Supervisor:	Assoz. Prof. Doz. Dr. Michael Schlosser

Inhaltsverzeichnis

Einleitung

Hört man das Wort „Monte-Carlo“, so werden es viele mit dem berühmten gleichnamigen Casino im Fürstentum Monaco assoziieren. Und wer schon einmal ein Casino besucht hat, weiß, dass dort der Zufall ein ständiger Begleiter ist.

In der Mathematik bezeichnen Monte-Carlo-Methoden Algorithmen, die Zufallszahlen benutzen. Sie werden deshalb auch randomisierte oder stochastische Algorithmen genannt und sind dem Gebiet der stochastischen Simulation oder experimentellen Stochastik zuzuordnen [29]. Behrends bezeichnet Monte-Carlo-Methoden, sehr treffend, als Rechenverfahren, bei denen „der Zufall als Rechenknecht“ eingesetzt wird [5, S. 183].

Als Erfinder der Monte-Carlo-Methode gilt der Physiker Enrico Fermi. Er benutzte die Methode im Rahmen seiner Arbeiten in den 1930er-Jahren, publizierte jedoch nichts zu diesem Thema. Die Mathematiker dieser Zeit kannten die Methode des „statistical sampling“, aber erst mit der Erfindung des ersten elektronischen Computers, dem ENIAC, war die Zeit reif für die Monte-Carlo-Methode. Die Mathematiker Stan Ulam und John von Neumann erkannten das Potential des Computers und begründeten die Monte-Carlo-Methode (1947). Der Name „Monte-Carlo-Methode“ geht auf Nick Metropolis zurück und ist eine Anspielung auf Ulams Onkel, der sich Geld von Verwandten ausleihen würde, um nach Monte-Carlo zum Spielen zu gehen [28].

Aufgrund der enormen Steigerung der Rechenleistung von Computern in den vergangenen Jahrzehnten und der Entwicklung immer besserer Algorithmen haben Monte-Carlo-Methoden zunehmend an Bedeutung gewonnen. Heutzutage werden sie in vielen Gebieten eingesetzt. Ihr breites Einsatzgebiet reicht von der numerischen Berechnung hochdimensionaler Integrale, über die Bestimmung des Preises von Finanzderivaten, bis zur computergestützten Simulation von Naturkatastrophen [13].

Auch in der Klimaforschung finden Monte-Carlo-Methoden ihre Anwendung. Die Dokumentation mit dem Titel „Klimawandel – Woher kommen die Zahlen?“ [33] zeigt sehr anschaulich, wie mithilfe von Monte-Carlo-Methoden Prognosen zum Klimawandel erstellt werden.

Wie der Titel bereits verrät, setzt sich diese Arbeit speziell mit der numerischen Integration auseinander. Besonders in deutschsprachigen Mathematik-Lehrbüchern ist die Monte-Carlo-Integration eher eine Randerscheinung. Da mich diese Art des Integrierens aufgrund ihrer Universalität und gleichzeitigen Einfachheit faszinierte, entschied ich mich selbst eine umfassende Darstellung dieses Themas zu verfassen. Das Ziel dieser Arbeit ist, nicht nur die Theorie vorzustellen, sondern auch die praktische Seite, in Form der konkreten Umsetzung dieser numerischen Integrationsmethode mit MATHEMATICA [41], ausführlich zu erklären.

Die Umsetzung mit MATHEMATICA wird im Rahmen von vielen konkreten Beispielen, welche die theoretischen Ausführungen begleiten, schrittweise dargestellt. Die grundlegende Kenntnis der Syntax von MATHEMATICA wird dabei als bekannt vorausgesetzt. Hauptsächlich wird mit Listen gearbeitet und die sehr effiziente „Listenarithmetik“ von MATHEMATICA genutzt. Dadurch bleibt der Zusammenhang zur Theorie unmittelbar einsichtig. Des Weiteren werden

verschiedene Möglichkeiten zur grafischen Darstellung von Rechenergebnissen und zur Steigerung der Effizienz der vorgestellten Algorithmen thematisiert.

Im ersten Kapitel sind Grundlagen aus der Wahrscheinlichkeitstheorie, die das Fundament für diese Arbeit darstellen, zu finden. Das Wort Fundament ist auch für Kapitel 2 eine zutreffende Beschreibung, da darin die Erzeugung von Zufallszahlen thematisiert wird. In Kapitel 3 nutzen wir schließlich die Ergebnisse der vorherigen beiden Kapitel, um die (naive) Monte-Carlo-Integration einzuführen. Darüberhinaus beschäftigen wir uns darin mit der Genauigkeit dieser Integrationsmethode. In Kapitel 4 wird die Monte-Carlo-Integration mit der Trapezregel, einem einfachen Vertreter von deterministischen Verfahren zur numerischen Integration, verglichen. Insbesondere wird dabei thematisiert, warum die Monte-Carlo-Integration die einzige praktikable Möglichkeit darstellt, um sehr hochdimensionale Integrale numerisch zu berechnen. Wir werden sehen, dass die (naive) Monte-Carlo-Integration nicht nur eine einfache und universell einsetzbare, sondern leider auch langsame Methode zur numerischen Integration ist. In Kapitel 5 werden deshalb verschiedene Techniken vorgestellt, mit denen die Monte-Carlo-Integration beschleunigt werden kann. Kapitel 6 widmet sich der Frage, welches Ziel beim Spielen von Darts anvisiert werden sollte, um im Schnitt eine möglichst hohe Punktzahl zu erzielen. Eine Antwort auf diese Frage wird sich mithilfe einer umfassenden Simulation ergeben.

1 Grundlagen aus der Wahrscheinlichkeitstheorie

Im ersten Kapitel werden einige Ergebnisse aus der Wahrscheinlichkeitstheorie, welche die Grundlage für unsere späteren Betrachtungen bilden, präsentiert. Von besonderer Bedeutung sind dabei das Gesetz großer Zahlen und der Zentrale Grenzwertsatz. Diese kurze Einführung soll vor allem den theoretischen Rahmen dieser Arbeit abklären und eine „gemeinsame Begriffsbasis" schaffen, weshalb auf die Darstellung konkreter Beispiele und aufwendiger Beweise, die weitere, nicht für diese Arbeit relevante, Begriffe benötigen würden, verzichtet wird. Als Hauptquellen für diese kurze Einführung in die Wahrscheinlichkeitstheorie dienten mir vor allem [7], [16], [18] und [19].

1.1 Wahrscheinlichkeitsräume

Definition 1.1 (Ergebnismenge und Ergebnis)
Eine Menge Ω heißt Ergebnismenge, wenn ihre Elemente alle möglichen Ergebnisse eines Zufallsexperiments beschreiben. Ein Element $\omega \in \Omega$ heißt demnach Ergebnis.

Häufig interessiert man sich jedoch nicht für ein konkretes Ergebnis eines Zufallsexperiments, sondern dafür, ob ein bestimmtes Ereignis, welches aus gewissen Einzelergebnissen besteht, eintritt. Ereignisse entsprechen somit (gewissen) Teilmengen von Ω. Aus diesem Grund wollen wir ein System $\mathcal{A}$ von Ereignissen festlegen, sodass jedem Ereignis $A \in \mathcal{A}$ eine Wahrscheinlichkeit $\mathbb{P}(A)$ für das Eintreten von A zugeordnet werden kann [16].

Es erscheint zunächst naheliegend $\mathcal{A}$ mit der Potenzmenge $\mathcal{P}(\Omega)$, also der Menge aller Teilmengen von Ω, gleichzusetzen. Ist die Ergebnismenge Ω abzählbar, so ist dies auch ohne weiteres möglich. Für eine überabzählbare Ergebnismenge misslingt dieser Ansatz jedoch. Eine Begründung dafür liefert der Unmöglichkeitssatz von Vitali [7, S. 710]. Deshalb lassen wir nur noch gewisse Ereignisse zu. Dies führt uns zu folgender Definition:

Definition 1.2 (σ-Algebra und Ereignisraum)
Sei $\Omega \neq \emptyset$. Ein Mengensystem $\mathcal{A} \subseteq \mathcal{P}(\Omega)$ mit den Eigenschaften
(a) $\Omega \in \mathcal{A}$,
(b) $A \in \mathcal{A} \Longrightarrow A^c = \Omega \setminus A \in \mathcal{A}$,
(c) $A_1, A_2, \ldots \in \mathcal{A} \Longrightarrow \bigcup_{i\geq 1} A_i \in \mathcal{A}$
heißt eine σ-Algebra in Ω. Das Paar $(\Omega, \mathcal{A})$ heißt dann Ereignisraum.

Bemerkung 1.3
Aus dieser Definition können sofort weitere Eigenschaften abgeleitet werden:
(a) $\emptyset \in \mathcal{A}$, weil $\emptyset = \Omega^c$ ist.
(b) $A, B \in \mathcal{A} \Longrightarrow A \cup B \in \mathcal{A}$, weil $A \cup B = A \cup B \cup \emptyset \cup \emptyset \cup \ldots$ ist.
(c) $A, B \in \mathcal{A} \Longrightarrow A \cap B \in \mathcal{A}$, weil $A \cap B = (A^c \cup B^c)^c$ ist.
(d) $A, B \in \mathcal{A} \Longrightarrow A \setminus B \in \mathcal{A}$, weil $A \setminus B = A \cap B^c$ ist.

Zudem gilt, dass Durchschnitte von abzählbar vielen Mengen in $\mathcal{A}$ wieder zu $\mathcal{A}$ gehören:

$$A_1, A_2, \ldots \in \mathcal{A} \Longrightarrow \bigcap_{i\geq 1} A_i \in \mathcal{A}, \text{ weil } \bigcap_{i\geq 1} A_i = \left(\bigcup_{i\geq 1} A_i^c\right)^c \text{ ist.}$$

Bemerkung 1.4

Für den in dieser Arbeit relevanten Grundraum $\Omega = \mathbb{R}^n$ legen wir stets $\mathcal{B}^n$, die Borel'sche σ-Algebra auf $\mathbb{R}^n$ zugrunde. Sie wird (zum Beispiel) durch das System

$$\mathcal{G} = \{[a_1, b_1] \times \ldots \times [a_n, b_n] : a_i < b_i, a_i, b_i \in \mathbb{Q}\}$$

aller achsenparallelen kompakten Quader in $\mathbb{R}^n$ mit rationalen Eckpunkten aufgespannt (genauer siehe z. B. [16, S. 11 ff.]).

Speziell für $\Omega = \mathbb{R}$ ergibt sich so ein System von Teilmengen $\mathcal{B}^1 = \mathcal{B}$, dass alle praktisch relevanten Mengen wie $[a, b], (a, b), (a, b], [a, b), (-\infty, a), (-\infty, a], (a, \infty)$ oder $[a, \infty)$ für $a, b \in \mathbb{R}$ enthält [19]. Entsprechendes gilt auch im Fall $n \geq 2$ [16]. Ein Element B einer Borel'schen σ-Algebra $\mathcal{B}^n$ nennt man Borelmenge.

Definition 1.5 (Wahrscheinlichkeitsmaß und Wahrscheinlichkeitsraum)

Sei $(\Omega, \mathcal{A})$ ein Ereignisraum. Eine Funktion $\mathbb{P} : \mathcal{A} \to [0,1]$ heißt Wahrscheinlichkeitsmaß (oder Wahrscheinlichkeitsverteilung) auf $\mathcal{A}$, wenn

(a) $\mathbb{P}(\Omega) = 1$, (Normiertheit)

(b) für paarweise disjunkte $A_1, A_2, \ldots \in \mathcal{A}$ gilt:

$$\mathbb{P}\left(\bigcup_{i \geq 1} A_i\right) = \sum_{i \geq 1} \mathbb{P}(A_i) \ . \qquad (\sigma\text{-Additivität})$$

Das Tripel $(\Omega, \mathcal{A}, \mathbb{P})$ heißt dann Wahrscheinlichkeitsraum.

Bemerkung

Nichtnegativität, Normiertheit und σ-Additivität eines Wahrscheinlichkeitsmaßes werden häufig als die Axiome von Kolmogorow bezeichnet.

Satz 1.6 (Rechenregeln für Wahrscheinlichkeitsmaße)

Seien $(\Omega, \mathcal{A}, \mathbb{P})$ ein Wahrscheinlichkeitsraum und $A, B, A_1, A_2, \ldots \in \mathcal{A}$ Ereignisse. Dann gelten:

(a) $\mathbb{P}(\emptyset) = 0$,

(b) $\mathbb{P}(A \cup B) + \mathbb{P}(A \cap B) = \mathbb{P}(A) + \mathbb{P}(B)$,

(c) $\mathbb{P}(A^c) = 1 - \mathbb{P}(A)$,

(d) $A \subseteq B \Longrightarrow \mathbb{P}(A) \leq \mathbb{P}(B)$,

(e) $A_1 \subseteq A_2 \subseteq \ldots$ und $A = \bigcup_{i \geq 1} A_i$ (kurz $A_n \uparrow A$) $\Longrightarrow \lim_{n \to \infty} \mathbb{P}(A_n) = \mathbb{P}(A)$,

(f) $A_1 \supseteq A_2 \supseteq \ldots$ und $A = \bigcap_{i \geq 1} A_i$ (kurz $A_n \downarrow A$) $\Longrightarrow \lim_{n \to \infty} \mathbb{P}(A_n) = \mathbb{P}(A)$.

Beweis:

(a) Da $\emptyset \cap \emptyset = \emptyset$ gilt, die leere Menge also zu sich selbst disjunkt ist, ist $\emptyset, \emptyset, \ldots$ eine Folge von paarweise disjunkten Mengen. Mit der σ-Additivität aus Definition 1.5 ergibt sich dann $\mathbb{P}(\emptyset) = \mathbb{P}(\emptyset \cup \emptyset \cup \ldots) = \sum_{i \geq 1} \mathbb{P}(\emptyset)$. Diese Gleichung kann jedoch nur für $\mathbb{P}(\emptyset) = 0$ erfüllt werden.

(b) Seien zunächst A und B disjunkt. Verwenden wir wieder die σ-Additivität und (a) so ergibt sich

$$\mathbb{P}(A \cup B) = \mathbb{P}(A \cup B \cup \emptyset \cup \emptyset \cup \ldots) = \mathbb{P}(A) + \mathbb{P}(B) + 0 + 0 + \cdots .$$

Womit zunächst gezeigt ist, dass sich die Wahrscheinlichkeit beim Zerlegen eines Ereignisses in endlich viele disjunkte Teile additiv verhält. Allgemein gilt somit

$$\mathbb{P}(A \cup B) + \mathbb{P}(A \cap B) = \mathbb{P}(A \setminus B) + \mathbb{P}(B \setminus A) + 2\mathbb{P}(A \cap B) = \mathbb{P}(A) + \mathbb{P}(B).$$

(c) Da A^c und A disjunkt sind, können wir (b) anwenden. Zusätzlich benötigen wir die Normiertheit aus Definition 1.5, womit sich

$$1 = \mathbb{P}(\Omega) = \mathbb{P}(A \cup A^c) = \mathbb{P}(A) + \mathbb{P}(A^c)$$

ergibt. Die Subtraktion von $\mathbb{P}(A)$ liefert schließlich die Behauptung.

(d) Für $A \subseteq B$ gilt

$$\mathbb{P}(B) = \mathbb{P}(A) + \mathbb{P}(B \setminus A) \geq \mathbb{P}(A)$$

aufgrund von (b) und der Nichtnegativität von Wahrscheinlichkeiten.

(e) Seien $A_0 = \emptyset$ und $B_n \coloneqq A_n \setminus A_{n-1}$ für $n \in \mathbb{N}$. So ist $\bigcup_{i\geq 1} A_i = \bigcup_{i\geq 1} B_i$ mit paarweise disjunkten Ereignissen $B_1, B_2, \ldots$. Außerdem gilt $A_n = B_1 \cup \ldots \cup B_n$. Mithilfe der σ-Additivität und (b) ergibt sich schließlich

$$\mathbb{P}(A) = \mathbb{P}\left(\bigcup_{i\geq 1} B_i\right) = \sum_{i\geq 1} \mathbb{P}(B_i) = \lim_{n\to\infty} \sum_{i=1}^{n} \mathbb{P}(B_i)$$

$$= \lim_{n\to\infty} \mathbb{P}(B_1 \cup \ldots \cup B_n) = \lim_{n\to\infty} \mathbb{P}(A_n)\,.$$

(f) Wir setzen $B_n \coloneqq A_n^c$ für $n \geq 1$ und $B \coloneqq A^c$. Aus $A_n \downarrow A$ folgt somit $B_n \uparrow B$. Verwenden wir nun (c) und (e) so ergibt sich

$$\mathbb{P}(A) = 1 - \mathbb{P}(B) = 1 - \lim_{n\to\infty} \mathbb{P}(B_n) = 1 - \lim_{n\to\infty} \big(1 - \mathbb{P}(A_n)\big) = \lim_{n\to\infty} \mathbb{P}(A_n)\,. \quad \blacksquare$$

Definition 1.7 (Stochastische Unabhängigkeit von Ereignissen)
Seien $(\Omega, \mathcal{A}, \mathbb{P})$ ein Wahrscheinlichkeitsraum und $n \geq 2$. Die Ereignisse $A_1, \ldots, A_n \in \mathcal{A}$ heißen (stochastisch) unabhängig, falls

$$\mathbb{P}\left(\bigcap_{i\in I} A_i\right) = \prod_{i\in I} \mathbb{P}(A_i)$$

für jede mindestens zweielementige Menge $I \subseteq \{1,2, \ldots, n\}$ gilt.

1.2 Zufallsvariablen und deren Verteilung

Definition 1.8 (Reelle Zufallsvariable)
Seien $(\Omega, \mathcal{A})$ und $(\Omega', \mathcal{A}')$ zwei Ereignisräume. Dann heißt jede Abbildung $X : \Omega \to \Omega'$ mit der sogenannten Messbarkeitseigenschaft

$$X^{-1}(A') \in \mathcal{A} \text{ für jedes } A' \in \mathcal{A}'$$

eine Ω'-wertige Zufallsvariable. Wir nennen einen Funktionswert $X(\omega)$ eine Realisierung der Zufallsvariable X.

Bemerkung 1.9

(1) $X^{-1}(A')$ bezeichnet die Menge $\{\omega \in \Omega : X(\omega) \in A'\}$ aller Urbilder der Ereignisse im Bildraum.

(2) Für eine übersichtliche Notation setzen wir $\{X \in A'\} \coloneqq \{\omega \in \Omega : X(\omega) \in A'\} = X^{-1}(A')$. Bei Ausdrücken der Form $P(\{X \in A'\})$ lassen wir die Mengenklammern weg und schreiben stattdessen kurz $P(X \in A')$.

(3) Wir nennen X eine (reelle) Zufallsvariable, falls $(\Omega', \mathcal{A}') = (\mathbb{R}, \mathcal{B})$ gilt, und bezeichnen X als (n-dimensionalen) Zufallsvektor, wenn $(\Omega', \mathcal{A}') = (\mathbb{R}^n, \mathcal{B}^n)$ ist.

(4) Jede stetige Funktion $g : \Omega \to \Omega'$ ist eine Zufallsvariable [26, S. 140].

(5) Sind $X : \Omega \to \Omega'$ und $g : \Omega' \to \Omega''$ Zufallsvariablen, dann ist $Y = g \circ X$ eine Zufallsvariable [26, Lemma 11.3].

(6) Sind $X_1, \ldots, X_n$ reelle Zufallsvariablen, so wird durch $X(\omega) = \big(X_1(\omega), \ldots, X_n(\omega)\big)$ ein n-dimensionaler Zufallsvektor definiert (und umgekehrt) [26, Lemma 11.4].

(7) Sind $X_1, \ldots, X_n$ reelle Zufallsvariablen und $a_1, \ldots, a_n \in \mathbb{R}$, dann sind auch $a_1X_1 + \cdots + a_nX_n$ und $X_1X_2 \cdots X_n$ Zufallsvariablen [26, Satz 11.5].

Definition 1.10 (Indikatorfunktion)
Ist $A \in \mathcal{A}$ ein Ereignis, so heißt die durch

$$\mathbf{1}_A(\omega) \coloneqq \begin{cases} 1 & \text{falls } \omega \in A, \\ 0 & \text{sonst} \end{cases}$$

für $\omega \in \Omega$ definierte Zufallsvariable $\mathbf{1}_A$ die Indikatorfunktion von A. Statt $\mathbf{1}_A$ wird auch häufig die Schreibweise $\mathbf{1}\{A\}$ verwendet.

Satz 1.11 (Verteilung einer Zufallsvariable)
Seien $(\Omega, \mathcal{A}, \mathbb{P})$ ein Wahrscheinlichkeitsraum, $(\Omega', \mathcal{A}')$ ein Ereignisraum und $X : \Omega \to \Omega'$ eine Zufallsvariable. Dann wird durch

$$\mathbb{P}_X(A') \coloneqq \mathbb{P}(X \in A') \text{ für } A' \in \mathcal{A}'$$

ein Wahrscheinlichkeitsmaß $\mathbb{P}_X$ auf $(\Omega', \mathcal{A}')$ definiert. Man nennt $\mathbb{P}_X$ die Verteilung von X.

Beweis:
Aufgrund der Definition ist klar, dass $\mathbb{P}_X : \mathcal{A}' \to [0,1]$ ist. Die Normiertheit von $\mathbb{P}_X$ ergibt sich aufgrund von $\mathbb{P}_X(\Omega') = \mathbb{P}(X \in \Omega') = \mathbb{P}(\Omega) = 1$. Sind $A'_1, A'_2, \ldots \in \mathcal{A}'$ paarweise disjunkt, dann sind auch ihre Urbilder $X^{-1}(A'_1), X^{-1}(A'_2), \ldots$ paarweise disjunkt, weshalb die σ-Additivität von $\mathbb{P}_X$ aus der σ-Additivität von $\mathbb{P}$ folgt. ■

Definition 1.12 (Identische Verteiltheit)
Zwei Zufallsvariablen heißen identisch verteilt, wenn sie dieselbe Verteilung besitzen.

Definition 1.13 (Verteilungsfunktion)
Ist X eine reelle Zufallsvariable auf einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$, dann heißt die Funktion $F_X : \mathbb{R} \to [0,1]$ mit

$$F_X(x) \coloneqq \mathbb{P}_X\big((-\infty, x]\big) = \mathbb{P}(X \leq x)$$

die Verteilungsfunktion von X.

Bemerkung
Wenn klar ist, auf welche Zufallsvariable sich die Verteilungsfunktion bezieht, so schreiben wir statt F_X nur F.

Satz 1.14 (Eigenschaften der Verteilungsfunktion)
Für die Verteilungfunktion $F : \mathbb{R} \to [0,1]$ einer rellen Zufallsvariable X gelten:

(a) Aus $x \leq y$ folgt $F(x) \leq F(y)$. (F ist monoton wachsend)
(b) Für jede nicht anwachsende Folge $(x_n)_{n\in\mathbb{N}}$ mit $\lim_{n\to\infty} x_n = x \in \mathbb{R}$ gilt $F(x) = \lim_{n\to\infty} F(x_n)$. ($F$ ist rechtsseitig stetig)
(c) $\lim_{n\to\infty} F(-n) = 0$ und $\lim_{n\to\infty} F(n) = 1$. (Asymptotisches Verhalten)
(d) $\mathbb{P}(a < X \leq b) = F(b) - F(a)$.
(e) Für jede von links gegen $x \in \mathbb{R}$ konvergierende Folge $(x_n)_{n\in\mathbb{N}}$ gilt $\mathbb{P}(X = x) = F(x) - \lim_{n\to\infty} F(x_n)$.

Beweis:
(a) Aus $x \leq y$ folgt $\{X \leq x\} \subseteq \{X \leq y\}$. Mit der Monotonie von $\mathbb{P}$ (Satz 1.6(d)) folgt dann

$$F(x) = \mathbb{P}(X \leq x) \leq \mathbb{P}(X \leq y) = F(y).$$

(b) Ist $x_1 \geq x_2 \geq \ldots$ eine von rechts gegen x konvergierende Folge, dann folgt unmittelbar $\{X \leq x_1\} \supseteq \{X \leq x_2\} \supseteq \ldots$. Außerdem gilt $\{X \leq x\} = \bigcap_{i\geq 1}\{X \leq x_i\}$. Mit Satz 1.6(f) ergibt sich dann

$$F(x) = \mathbb{P}(X \leq x) = \lim_{n\to\infty} \mathbb{P}(X \leq x_n) = \lim_{n\to\infty} F(x_n).$$

(c) Da $\{X \leq -1\} \supseteq \{X \leq -2\} \supseteq \ldots$ und $\emptyset = \bigcap_{n\geq 1}\{X \leq -n\}$ gelten, kann Satz 1.6(a, f) verwendet werden, um $0 = \mathbb{P}(\emptyset) = \lim_{n\to\infty} \mathbb{P}(X \leq -n) = \lim_{n\to\infty} F(-n)$ zu erhalten. Aufgrund von $\{X \leq 1\} \subseteq \{X \leq 2\} \subseteq \ldots$ und $\Omega = \bigcup_{n\geq 1}\{X \leq n\}$ ergibt sich, durch Verwendung von Satz 1.6(e) und der Normiertheit von $\mathbb{P}$ (Definition 1.5),

$$1 = \mathbb{P}(\Omega) = \lim_{n\to\infty} \mathbb{P}(X \leq n) = \lim_{n\to\infty} F(n).$$

(d) Die Zerlegung von $\{X \leq b\}$ in disjunkte Ereignisse $\{X \leq a\}$ und $\{a < X \leq b\}$ ermöglicht die Anwendung von Satz 1.6(b) und wir erhalten $\mathbb{P}(X \leq b) = \mathbb{P}(X \leq a) + \mathbb{P}(a < X \leq b)$. Durch Umformen ergibt sich schließlich

$$\mathbb{P}(a < X \leq b) = \mathbb{P}(X \leq b) - \mathbb{P}(X \leq a) = F(b) - F(a).$$

(e) Sei $x_1 \leq x_2 \leq \ldots < x$ eine von links gegen x konvergierende Folge, so bilden die Ereignisse $A_n \coloneqq \{X \leq x_n\}$ für $n \geq 1$ eine aufsteigende Folge mit $\bigcup_{n\geq 1} A_n = \{X < x\}$. Mithilfe von Satz 1.6(e) folgt schließlich

$$\mathbb{P}(X < x) = \lim_{n\to\infty} \mathbb{P}(A_n) = \lim_{n\to\infty} F(x_n)\,. \quad \blacksquare$$

Bemerkung 1.15

Die Verteilung einer reellen Zufallsvariable auf einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$ wird durch die Verteilungsfunktion F_X eindeutig festgelegt [16].

Es ist vorteilhaft, dass man sich bei Problemen, bei denen nur die Verteilung einer Zufallsvariable X relevant ist, keine Sorgen um die konkrete Gestalt eines Wahrscheinlichkeitsraums $(\Omega, \mathcal{A}, \mathbb{P})$ machen muss. Es ist ausreichend, eine Funktion $F : \mathbb{R} \to [0,1]$, welche die Eigenschaften (a)-(c) aus Satz 1.14 erfüllt, vorzugeben. Zudem kann man allein aus der Kenntnis der Wahrscheinlichkeiten $F(x) = \mathbb{P}(X \leq x)$ für alle $x \in \mathbb{R}$ die Wahrscheinlichkeit $\mathbb{P}(X \in B)$ für jede Borelmenge B bestimmen [19, 30.6].

Definition 1.16 (Diskrete Zufallsvariable)

Ist X eine reelle Zufallsvariable (oder ein n-dimensionales Zufallsvektor), dann heißt X diskret (verteilt), wenn es eine abzählbare Menge $D \subset \mathbb{R}$ (bzw. $D \subset \mathbb{R}^k$) gibt, sodass

$$\mathbb{P}_X(D) = \mathbb{P}(X \in D) = 1$$

gilt. Man sagt auch, dass X eine diskrete Verteilung besitzt.

Bemerkung

Ist X eine diskrete Zufallsvariable, dann nennt man ihre Verteilungsfunktion F eine diskrete Verteilungsfunktion. Sie besitzt dann folgende Gestalt:

$$F(x) = \sum_{y\in D: y\leq x} \mathbb{P}(X = y).$$

Definition 1.17 (Diskrete Gleichverteilung)

Eine diskrete Zufallsvariable X mit den Realisierungen $x_1, \ldots, x_n$ heißt gleichverteilt (auf der Menge $\{x_1, \ldots, x_n\}$), wenn $\mathbb{P}(X = x_i) = 1/n$ für alle $i \in \{1, \ldots, n\}$ gilt.

Definition 1.18 (Gemeinsame Verteilung, Randverteilung)

Ist $X = (X_1, \ldots, X_n)$ ein n-dimensionaler Zufallsvektor, dann nennt man die Verteilung von X auch die gemeinsame Verteilung von $X_1, \ldots, X_n$. Die Verteilung von X_i heißt (i-te) Randverteilung von X für $1 \leq i \leq n$.

Definition 1.19 (Stetiger Zufallsvektor)
Ein n-dimensionaler Zufallsvektor $X = (X_1, \dots, X_n)$ heißt (absolut) stetig (verteilt), wenn es eine nichtnegative Borel-messbare Funktion $f : \mathbb{R}^n \to \mathbb{R}$ mit

$$\int_{\mathbb{R}^n} f(x)\, dx = 1$$

gibt, sodass gilt:

$$\mathbb{P}_X(B) = \mathbb{P}(X \in B) = \int_B f(x)\, dx \text{ für } B \in \mathcal{B}^n.$$

Man sagt dann, dass X eine (absolut) stetige Verteilung besitzt und bezeichnet die Funktion f als Dichte von X oder gemeinsame Dichte von $X_1, \dots, X_n$.

Bemerkung
(1) Ist X eine reelle (absolut) stetig verteilte Zufallsvariable mit Dichte f, so besitzt die Verteilungsfunktion F von X die Darstellung

$$F(x) = \int_{-\infty}^{x} f(t)\, dt \text{ für } x \in \mathbb{R}.$$

(2) Besitzt der Zufallsvektor X die Dichte f, so hat X_i eine marginale Dichte f_i für $1 \leq i \leq n$. Es gilt dann zum Beispiel:

$$f_1(x_1) = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} f(x_1, \dots, x_n)\, dx_2 \dots dx_n.$$

Definition 1.20 (Lebesgue-Maß)
Die Abbildung $\lambda^n : \mathcal{B}^n \to [0, \infty]$, die jedem $A \in \mathcal{B}^n$ sein n-dimensionales Volumen

$$\lambda^n(A) := \int \mathbf{1}_A(x)\, dx$$

zuordnet, heißt das (n-dimensionale) Lebesgue-Maß auf $\mathbb{R}^n$.

Anmerkung
Weil λ^n die σ-Additivitätseigenschaft erfüllt und $\lambda^n(\emptyset) = 0$ gilt, ist λ^n ein „Maß“ auf $(\mathbb{R}^n, \mathcal{B}^n)$. Schließlich ist an dieser Stelle zu betonen, dass der Wahrscheinlichkeitstheorie der Lebesgue'sche Integralbegriff zugrunde liegt, also alle im Rahmen dieser Arbeit auftretenden Integrale als Lebesgue-Integrale[1] zu verstehen sind. Erfüllt eine Funktion $g : \mathbb{R}^n \to \mathbb{R}$ die Messbarkeitseigenschaft $\{x \in \mathbb{R}^n : g(x) \leq t\} \in \mathcal{B}^n$ für alle $t \in \mathbb{R}$ und ist sie zusätzlich Riemann-integrierbar, so stimmen das Riemann-Integral und das Lebesgue-Integral überein. Für konkrete Berechnungen reicht also meistens die Kenntnis des Riemann-Integrals [16; 19].

Definition 1.21 (Mehrdimensionale stetige Gleichverteilung)
Sei $B \in \mathcal{B}^n$ mit $0 < \lambda^n(B) < \infty$. Ein Zufallsvektor X besitzt eine stetige Gleichverteilung auf B, falls X die Dichte

$$f(x) := \begin{cases} 1/\lambda^n(B) & \text{für } x \in B, \\ 0 & \text{sonst} \end{cases}$$

besitzt. In diesem Fall schreiben wir kurz $X \sim \mathcal{U}(B)$.

[1] In [7, Kapitel 7] sind die Grundzüge der Maß- und Integrationstheorie zu finden.

Bemerkung 1.22
Weil uns der Fall $n = 1$ und $B = [a, b]$ für $a, b \in \mathbb{R}$ und $a < b$ sehr häufig begegnen wird, formulieren wir explizit:
Eine reelle Zufallsvariable X hat eine (stetige) Gleichverteilung auf dem Intervall $[a, b]$, wenn X die Dichte

$$f(x) \coloneqq \begin{cases} 1/(b-a) & \text{für } a \le x \le b, \\ 0 & \text{sonst} \end{cases}$$

besitzt. In diesem Fall schreiben wir kurz $X \sim \mathcal{U}[a, b]$. Die Verteilungsfunktion F von X hat dann die Darstellung

$$F(x) = \begin{cases} 0 & \text{falls } x < a, \\ (x-a)/(b-a) & \text{falls } a \le x \le b, \\ 1 & \text{falls } x > b. \end{cases}$$

Der nachfolgende Satz wird sich später, im Zusammenhang mit der Erzeugung von Zufallszahlen, als sehr nützlich erweisen.

Satz 1.23
Aus $X \sim \mathcal{U}[0,1]$ folgt $a + (b-a)X \sim \mathcal{U}[a, b]$ für $a, b \in \mathbb{R}$ mit $a < b$.

Beweis:
Wir setzen $Y \coloneqq a + (b-a)X$. Aufgrund von $\mathbb{P}(0 \le X \le 1) = 1$ gilt $\mathbb{P}(a \le Y \le b) = 1$. Für jedes x mit $a \le x \le b$ ergibt sich

$$\mathbb{P}(Y \le x) = \mathbb{P}(a + (b-a)X \le x) = \mathbb{P}\left(X \le \frac{x-a}{b-a}\right) = \frac{x-a}{b-a}.$$

Ein Vergleich mit Bemerkung 1.22 unter Berücksichtigung von Bemerkung 1.15 zeigt schließlich, dass $Y \sim \mathcal{U}[a, b]$. ■

Definition 1.24 (Normalverteilung)
Eine Zufallsvariable X hat eine Normalverteilung mit den Parametern μ und σ^2, falls X die Dichte

$$f(x) \coloneqq \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \text{ für } x \in \mathbb{R}$$

besitzt. In diesem Fall schreiben wir kurz $X \sim \mathcal{N}(\mu; \sigma^2)$.

Bemerkung
Sind $\mu = 0$ und $\sigma^2 = 1$, dann sagen wir, dass X eine Standardnormalverteilung besitzt. Zudem bezeichnen wir die Dichte mit φ anstatt f und die Verteilungsfunktion mit Φ anstatt F.

Definition 1.25 (Stochastische Unabhängigkeit von Zufallsvariablen)
Zufallsvariablen $X_1, \dots, X_n$ heißen (stochastisch) unabhängig, wenn für jede Wahl von Borelmengen $B_1, \dots, B_n$ gilt:

$$\mathbb{P}(X_1 \in B_1, \dots, X_n \in B) = \prod_{i=1}^{n} \mathbb{P}(X_i \in B_i).$$

Bemerkung 1.26
(1) Man kann zeigen [18, Satz 9.11], dass Funktionen unabhängiger Zufallsvariablen wieder unabhängig sind. Das heißt: Sind $X_1, \dots, X_n$ unabhängige Zufallsvariablen und ist g eine reellwertige messbare Funktion, dann sind auch die Zufallsvariablen $g(X_1), \dots, g(X_n)$ unabhängig.

(2) Sind $X_1, \dots, X_n$ unabhängige Zufallsvariablen und besitzt X_i die marginale Dichte f_i für $1 \leq i \leq n$, dann besitzt der Zufallsvektor $X = (X_1, \dots, X_n)$ die Produkt-Dichte

$$f(x_1, \dots, x_n) = f_1(x_1) \cdots f_n(x_n).$$

Hat umgekehrt X eine Dichte f der obigen Gestalt mit marginalen Dichten $f_1, \dots, f_n$, dann sind $X_1, \dots, X_n$ unabhängig mit Dichten $f_1, \dots, f_n$ [19]. Ein Beweis dieser Aussage ist zum Beispiel in [7, S. 821] zu finden.

Der nachfolgende Satz bildet die Grundlage für die Konstruktion einer Folge von unabhängigen Zufallsvektoren. Für einen Beweis verweise ich auf [18, Satz 9.14].

Satz 1.27 (Blockungslemma für unabhängige Zufallsvariablen)
Seien $X_1, \dots, X_n$ unabhängige Zufallsvariablen. Setzen wir

$$Y_1(\omega) \coloneqq \big(X_1(\omega), \dots, X_k(\omega)\big) \text{ und } Y_2(\omega) \coloneqq \big(X_{k+1}(\omega), \dots, X_n(\omega)\big)$$

für $\omega \in \Omega$ und $k \in \{1,2, \dots, n-1\}$, dann sind auch die Zufallsvektoren Y_1 und Y_2 unabhängig.

1.3 Erwartungswert und Varianz

Zunächst wird der Erwartungswert sehr allgemein definiert, um sämtliche in dieser Arbeit vorkommenden Zufallsvariablen zu berücksichtigen. Für eine umfassende Motivation dieser Begriffsbildung verweise ich auf [7, 22.3].

Definition 1.28 (Erwartungswert)
Seien $(\Omega, \mathcal{A}, \mathbb{P})$ ein Wahrscheinlichkeitsraum und $X : \Omega \to \overline{\mathbb{R}}$ $(\coloneqq \mathbb{R} \cup \{+\infty, -\infty\})$ eine Zufallsvariable. Der Erwartungswert von X existiert, wenn $\int |X| \, d\mathbb{P} < \infty$ ist. Dann heißt

$$\mathbb{E}(X) \coloneqq \int X \, d\mathbb{P} = \int_{\Omega} X(\omega) \, d\mathbb{P}(\omega)$$

der Erwartungswert von X.

Der nachfolgende Satz folgt direkt aus dem allgemeinen Transformationssatz [18, Satz 9.27] und gibt uns Auskunft darüber, wie die konkrete Berechnung des Erwartungswerts durchgeführt werden kann.

Satz 1.29 (Transformationsformel)
Seien $X = (X_1, \dots, X_n)$ ein n-dimensionaler Zufallsvektor und $g : \mathbb{R}^n \to \mathbb{R}$ eine Borel-messbare Funktion. Dann gelten:

(1) Ist X (absolut) stetig verteilt mit Dichte f, so gilt

$$\mathbb{E}\big(g(X)\big) = \int_{\mathbb{R}^n} g(x) \, \mathbb{P}_X(dx) = \int_{\mathbb{R}^n} g(x) \cdot f(x) \, dx$$

falls $\int_{\mathbb{R}^n} |g(x)| \cdot f(x) \, dx < \infty$ ist.

(2) Ist X diskret verteilt mit $\mathbb{P}(X = x) > 0$ für $x \in D$ und $\mathbb{P}_X(\mathbb{R}^n \setminus D) = 0$, dann gilt

$$\mathbb{E}\big(g(X)\big) = \int_{\mathbb{R}^n} g(x) \, \mathbb{P}_X(dx) = \sum_{x \in D} g(x) \cdot \mathbb{P}(X = x)$$

falls $\sum_{x \in D} |g(x)| \cdot \mathbb{P}(X = x) < \infty$ ist.

Bemerkung 1.30

(1) Speziell für $g : \mathbb{R} \to \mathbb{R}$ mit $g(x) = x$ ergeben sich dann
   (a) für den Erwartungswert einer stetigen Zufallsvariable X die Darstellung
   $$\mathbb{E}(X) = \int_{\mathbb{R}} x \cdot f(x)\, dx$$
   falls das Integral über den Betrag des Integranden existiert,
   (b) für den Erwartungswert einer diskreten Zufallsvariable X die Darstellung
   $$\mathbb{E}(X) = \sum_{x \in D} x \cdot \mathbb{P}(X = x)$$
   falls die Reihe absolut konvergent ist.

(2) Die Transformationsformel zeigt also, dass der Erwartungswert einer Zufallsvariable nur von ihrer Verteilung, nicht aber von der speziellen Gestalt des zugrunde liegenden Wahrscheinlichkeitsraumes abhängt [19].

(3) Wenn klar ist, auf welchen Ausdruck sich der Erwartungswert bezieht, wird, der Kürze halber und für eine bessere Übersicht, oft die Klammerung weglassen. So wird dann zum Beispiel $\mathbb{E}X$ anstatt $\mathbb{E}(X)$ geschrieben.

(4) Ab dieser Stelle beschäftigen wir uns nur noch mit reellen Zufallsvariablen und schreiben deshalb nur Zufallsvariable statt reelle Zufallsvariable.

Satz 1.31 (Eigenschaften des Erwartungswerts)

Seien X, Y Zufallsvariablen mit existierenden Erwartungswerten, $a \in \mathbb{R}$ und A ein Ereignis. So gelten:

(a) $\mathbb{E}(X + Y) = \mathbb{E}X + \mathbb{E}Y$. (Additivität)
(b) $\mathbb{E}(aX) = a\,\mathbb{E}X$. (Homogenität)
(c) $\mathbb{E}(\mathbf{1}_A) = \mathbb{P}(A)$.
(d) $\mathbb{E}(a) = a$.
(e) Aus $X \leq Y$ folgt $\mathbb{E}X \leq \mathbb{E}Y$. (Monotonie)

Beweis:

(a) und (b) folgen unmittelbar aus der Linearität des Integrals (bzw. der Summe). Um (c) zu zeigen verwenden wir, dass $1_A(\omega) = 1$ für $\omega \in \Omega$ und $1_A(\omega) = 0$ für $\omega \notin \Omega$. Damit folgt:

$$\mathbb{E}(1_A) = \int_{\Omega} 1_A(\omega)\, d\mathbb{P}(\omega) = \int_{A} 1\, d\mathbb{P}(\omega) = \mathbb{P}(A).$$

Für den Beweis von (d) verwenden wir (c) mit $A = \Omega$ und (b):

$$\mathbb{E}(a) = \mathbb{E}(a\mathbf{1}_\Omega) = a\mathbb{E}(\mathbf{1}_\Omega) = a\mathbb{P}(\Omega) = a.$$

Schließlich zeigen wir (e), indem wir benutzen, dass $X \leq Y$ gleichbedeutend mit $Y - X \geq 0$ ist. Da in diesem Fall $\mathbb{E}(Y - X) \geq 0$ ist, folgt mit (a) und (b), dass $\mathbb{E}Y - \mathbb{E}X \geq 0$ ist und damit die Behauptung. ■

Bemerkung 1.32 (Additionsregel)

Satz 1.31(a) kann mittels vollständiger Induktion für n Zufallsvariablen $X_1, \ldots, X_n$ verallgemeinert werden, sodass Folgendes gilt:

$$\mathbb{E}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathbb{E}(X_i).$$

Satz 1.33 (Multiplikationsregel)
Sind X, Y stochastisch unabhängige Zufallsvariablen mit existierenden Erwartungswerten, so existiert auch der Erwartungswert des Produkts $X \cdot Y$, und es gilt

$$\mathbb{E}(X \cdot Y) = \mathbb{E}X \cdot \mathbb{E}Y.$$

Beweis:
Aufgrund der Unabhängigkeit von X und Y ist die gemeinsame Verteilung $\mathbb{P}_{(X,Y)}$ das Produkt (maß) der Randverteilungen $\mathbb{P}_X$ und $\mathbb{P}_Y$. Mit dem allgemeinen Transformationssatz [18, Satz 9.27] und dem allgemeinen Satz von Fubini [18, Satz 6.77] ergibt sich dann

$$\mathbb{E}(|X \cdot Y|) = \int |x \cdot y| \, \mathbb{P}_{(X,Y)}\big(d(x,y)\big) = \left(\int |x| \, \mathbb{P}_X(dx)\right)\left(\int |y| \, \mathbb{P}_Y(dy)\right) = \mathbb{E}|X| \cdot \mathbb{E}|Y| < \infty.$$

Damit existiert der Erwartungswert des Produkts. Lassen wir die Betragsstriche weg, so ergibt sich $\mathbb{E}(X \cdot Y) = \mathbb{E}X \cdot \mathbb{E}Y$. ■

Definition 1.34 (Varianz, Standardabweichung, Kovarianz und Korrelation)
(1) Ist X ein Zufallsvariable mit $\mathbb{E}(X^2) < \infty$, so heißen

$$\mathbb{V}(X) \coloneqq \mathbb{E}((X - \mathbb{E}X)^2)$$

die Varianz von X und $\sqrt{\mathbb{V}(X)}$ die Standardabweichung von X.

(2) Ist Y eine weitere Zufallsvariable mit $\mathbb{E}(Y^2) < \infty$, dann bezeichnet

$$\mathrm{Kov}(X,Y) \coloneqq \mathbb{E}((X - \mathbb{E}X)(Y - \mathbb{E}Y))$$

die Kovarianz zwischen X und Y.

(3) Gilt $\mathbb{V}(X) \cdot \mathbb{V}(Y) > 0$, so bezeichnet

$$\mathrm{Kor}(X,Y) \coloneqq \mathrm{Kov}(X,Y)/\sqrt{\mathbb{V}(X)\mathbb{V}(Y)}$$

den Korrelationskoeffizient zwischen X und Y.

(4) X und Y heißen unkorreliert, wenn $\mathrm{Kov}(X,Y) = 0$.

Bemerkung 1.35
(1) Aufgrund von $|X| \leq 1 + X^2$ folgt aus der Existenz von $\mathbb{E}(X^2)$ auch $\mathbb{E}(|X|) < \infty$ und somit die Existenz von $\mathbb{E}X$. Wegen $(X - a)^2 \leq X^2 + 2|a| \cdot |X| + a^2$ für $a \in \mathbb{R}$ existiert auch der Erwartungswert von $(X - \mathbb{E}X)^2$, also $\mathbb{V}(X)$.
(2) Wegen der Definition der Kovarianz ist klar, dass $\mathrm{Kov}(X,X) = \mathbb{V}(X)$ ist.

Satz 1.36 (Eigenschaften der Varianz und Kovarianz)
Seien X, Y Zufallsvariablen mit existierenden Varianzen und $a, b \in \mathbb{R}$, dann gelten:
(a) $\mathrm{Kov}(X,Y) = \mathbb{E}(X \cdot Y) - \mathbb{E}X \cdot \mathbb{E}Y$,
(b) $\mathbb{V}(X) = \mathbb{E}(X^2) - \mathbb{E}(X)^2$, (Verschiebungssatz)
(c) $\mathbb{V}(aX + b) = a^2 \, \mathbb{V}(X)$.

Beweis:
Mit den Eigenschaften des Erwartungswerts (Satz 1.31) ergibt sich (a):

$$\begin{aligned}\mathrm{Kov}(X,Y) &= \mathbb{E}((X - \mathbb{E}X)(Y - \mathbb{E}Y)) \\ &= \mathbb{E}(XY - \mathbb{E}(Y)X - \mathbb{E}(X)Y + \mathbb{E}X \, \mathbb{E}Y) \\ &= \mathbb{E}(XY) - \mathbb{E}Y \, \mathbb{E}X - \mathbb{E}X \, \mathbb{E}Y + \mathbb{E}X \, \mathbb{E}Y \\ &= \mathbb{E}(XY) - \mathbb{E}X \, \mathbb{E}Y.\end{aligned}$$

Wegen Bemerkung 1.35(2) ergibt sich (b) für $X = Y$ aus (a). Wir folgern (c) ebenfalls mithilfe der Eigenschaften des Erwartungswerts:
$\mathbb{V}(aX + b) = \mathbb{E}((aX + b - a\mathbb{E}X - b)^2) = \mathbb{E}(a^2(X - \mathbb{E}X)^2) = a^2 \, \mathbb{E}((X - \mathbb{E}X)^2) = a^2 \, \mathbb{V}(X)$. ■

Satz 1.37 (Additionsregel für die Varianz)
Sind $X_1, \dots, X_n$ stochastisch unabhängige Zufallsvariablen mit existierenden Varianzen, so gilt:

$$\mathbb{V}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathbb{V}(X_i).$$

Beweis:
Aufgrund der Cauchy-Schwarz'schen Ungleichung [22, 12.3] gilt $(\sum_{i=1}^{n} X_i)^2 \leq n \cdot \sum_{i=1}^{n} X_i^2$, was zeigt, dass die Varianz der Summe $\sum_{i=1}^{n} X_i$ existiert. Wegen Satz 1.36(c) gilt $\mathbb{V}(X + b) = \mathbb{V}(X)$ für $b \in \mathbb{R}$. Deshalb reicht es aus, den Fall $\mathbb{E}(X_i) = 0$ für $1 \leq i \leq n$ zu betrachten. Mit der Multiplikationsregel (Satz 1.33) folgt dann $\mathbb{E}(X_i X_j) = \mathbb{E}(X_i) \cdot \mathbb{E}(X_j) = 0$ für $i \neq j$. Des Weiteren gilt $\mathbb{E}(X_i^2) = \mathbb{V}(X_i)$, womit folgt:

$$\mathbb{V}\left(\sum_{i=1}^{n} X_i\right) = \mathbb{E}\left(\left(\sum_{i=1}^{n} X_i\right)^2\right) = \mathbb{E}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} X_i X_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbb{E}(X_i X_j)$$

$$= \sum_{i=1}^{n} \mathbb{E}(X_i^2) + \sum_{i \neq j} \mathbb{E}(X_i X_j) = \sum_{i=1}^{n} \mathbb{V}(X_i)\,. \quad \blacksquare$$

1.4 Grenzwertsätze

In diesem Abschnitt werden neben den wichtigen Konvergenzbegriffen der Stochastik, das Gesetz großer Zahlen und der Zentrale Grenzwertsatz kurz vorgestellt.

Definition 1.38 (Konvergenzbegriffe für Folgen von Zufallsvariablen)
Seien $X, X_1, X_2, \dots$ Zufallsvariablen auf einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$.

(1) Die Folge $(X_n)_{n\in\mathbb{N}}$ konvergiert stochastisch gegen X, falls für alle $\varepsilon > 0$ gilt:

$$\lim_{n\to\infty} \mathbb{P}(|X_n - X| \geq \varepsilon) = 0.$$

Wir schreiben dann kurz $X_n \xrightarrow{\mathbb{P}} X$.

(2) Die Folge $(X_n)_{n\in\mathbb{N}}$ konvergiert ($\mathbb{P}$-)fast sicher gegen X, wenn

$$\mathbb{P}\left(\left\{\omega \in \Omega : \lim_{n\to\infty} X_n(\omega) = X(\omega)\right\}\right) = 1$$

gilt. Wir schreiben dafür $X_n \xrightarrow{\text{f.s.}} X$.

(3) Für $n \geq 1$ und $x \in \mathbb{R}$ definieren wir $F(x) \coloneqq \mathbb{P}(X \leq x)$ und $F_n(x) \coloneqq \mathbb{P}(X_n \leq x)$. Die Folge $(X_n)_{n\in\mathbb{N}}$ konvergiert nach Verteilung (schwach) gegen X, falls für jede Stetigkeitsstelle x von F gilt, dass

$$\lim_{n\to\infty} F_n(x) = F(x).$$

Wir schreiben in diesem Fall $X_n \xrightarrow{\mathcal{D}} X$ und bezeichnen die Verteilung von X als Grenzverteilung oder asymptotische Verteilung von (X_n).

Bemerkung 1.39

(1) Aus fast sicherer Konvergenz folgt stets die stochastische Konvergenz. In einem diskreten Wahrscheinlichkeitsraum gilt auch die Umkehrung [7, S. 869]. Die stochastische Konvergenz impliziert wiederum die Verteilungskonvergenz. Die Umkehrung gilt für den Fall, dass X eine Einpunktverteilung besitzt [7, S. 881].

(2) Die stochastische Konvergenz von (X_n) gegen X besagt, dass X_n für großes n mit großer Wahrscheinlichkeit nahe bei X liegt, aber nicht, dass auch nur für ein einziges $\omega \in \Omega$ die

Folge $(X_n(\omega))$ gegen $X(\omega)$ konvergiert [26, S. 152]. Die fast sichere Konvergenz schließt diese Lücke und meint die punktweise Konvergenz von $(X_n(\omega))$ gegen $X(\omega)$ für alle ω außerhalb einer Nullmenge[2] [20]. Dieser Unterschied wird in Bemerkung 2.34 noch einmal verdeutlicht.

Satz 1.40 (Tschebyschow-Ungleichung)
Ist X eine Zufallsvariable mit $\mathbb{E}(X^2) < \infty$, so gilt für jedes $\varepsilon > 0$:

$$\mathbb{P}(|X - \mathbb{E}X| \geq \varepsilon) \leq \mathbb{V}(X)/\varepsilon^2.$$

Beweis:
Wir betrachten die beiden Funktionen

$$g(x) := \begin{cases} 1 & \text{falls } |x - \mathbb{E}X| \geq \varepsilon, \\ 0 & \text{sonst,} \end{cases} \quad \text{und} \quad h(x) := 1/\varepsilon^2 \cdot (x - \mathbb{E}X)^2 \text{ für } x \in \mathbb{R}.$$

Wegen $g(x) \leq h(x)$ für $x \in \mathbb{R}$ (siehe Abbildung 1.1) gilt $g\big(X(\omega)\big) \leq h\big(X(\omega)\big)$ für jedes $\omega \in \Omega$. Mit den Eigenschaften des Erwartungswerts und der Definition der Varianz ergibt sich

$$\mathbb{E}\big(g(X)\big) = \mathbb{P}(|X - \mathbb{E}X)| \geq \varepsilon) \leq \mathbb{E}\big(h(X)\big) = \mathbb{V}(X)/\varepsilon^2 \,. \quad \blacksquare$$

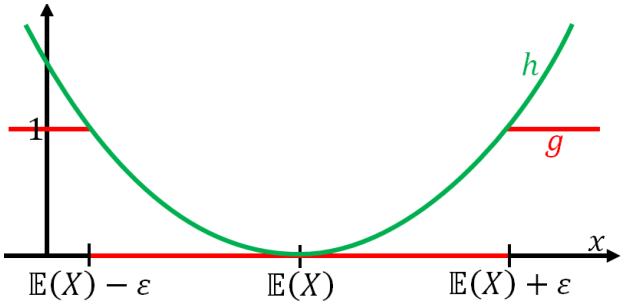

Abbildung 1.1 Überlegung zum Beweis der Tschebyschow-Ungleichung

Satz 1.41 (Das schwache Gesetz großer Zahlen)
Sei $(X_n)_{n\in\mathbb{N}}$ eine Folge stochastisch unabhängiger Zufallsvariablen mit gleichem Erwartungswert $\mu := \mathbb{E}(X_1) < \infty$ und gleicher Varianz $\sigma^2 := \mathbb{V}(X_1) < \infty$. Die Zufallsvariable

$$\overline{X}_n := \frac{1}{n}\sum_{i=1}^{n} X_i$$

bezeichne das arithmetische Mittel von $X_1, \ldots, X_n$. Dann gilt für jedes $\varepsilon > 0$:

$$\lim_{n\to\infty} \mathbb{P}\big(\big|\overline{X}_n - \mu\big| \geq \varepsilon\big) = 0.$$

Beweis:
Zunächst verwenden wir die Eigenschaften des Erwartungswerts (Satz 1.31 und Bem. 1.32) und der Varianz (Satz 1.36 und Satz 1.37), um $\mathbb{E}(\overline{X}_n)$ und $\mathbb{V}(\overline{X}_n)$ zu bestimmen:

$$\mathbb{E}\big(\overline{X}_n\big) = \mathbb{E}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}(X_i) = \frac{1}{n}\sum_{i=1}^{n} \mu = \frac{1}{n} \cdot n\mu = \mu, \tag{1.1}$$

$$\mathbb{V}\big(\overline{X}_n\big) = \mathbb{V}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2}\sum_{i=1}^{n} \mathbb{V}(X_i) = \frac{1}{n^2}\sum_{i=1}^{n} \sigma^2 = \frac{1}{n^2} \cdot n\sigma^2 = \frac{\sigma^2}{n}. \tag{1.2}$$

Nun setzen wir in die Tschebyschow-Ungleichung (Satz 1.40) ein, berücksichtigen die Nichtnegativität von Wahrscheinlichkeitsmaßen und erhalten

[2] $A \in \mathcal{A}$ mit $\mathbb{P}(A) = 0$.

$$0 \leq \mathbb{P}\left(\left|\overline{X}_n - \mu\right| \geq \varepsilon\right) \leq \frac{\sigma^2}{n\varepsilon^2}.$$

Durch Limesbildung ($n \to \infty$) folgt schließlich die Behauptung. ■

Wir formulieren nun das starke Gesetz großer Zahlen von Kolmogorow (für einen Beweis siehe z. B. [7, S. 873]). Zu beachten ist, dass dabei die Existenz der Varianz nicht vorausgesetzt wird.

Satz 1.42 (Das starke Gesetz großer Zahlen von Kolmogorow)
Sei $(X_n)_{n\in\mathbb{N}}$ eine Folge von unabhängigen und identisch verteilten Zufallsvariablen auf einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$. Dann sind folgende Aussagen äquivalent:

(a) $\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{\text{f.s.}} X$ für eine Zufallsvariable X.

(b) $\mathbb{E}(|X_1|) < \infty$.

In diesem Fall gilt $X = \mathbb{E}(X_1)$ fast sicher und somit $\overline{X}_n \xrightarrow{\text{f.s.}} \mathbb{E}(X_1)$.

Bemerkung 1.43
Nach dem schwachen Gesetz großer Zahlen konvergiert die Folge des arithmetischen Mittels von unabhängigen und identisch verteilten (gleicher Erwartungswert und gleiche Varianz) Zufallsvariablen stochastisch gegen den Erwartungswert μ. Dies deckt sich mit der Vorstellung des Erwartungswerts, als ein auf die Dauer erhaltener Durchschnittswert [19].
Mit dem schwachen Gesetz großer Zahlen, ergibt sich somit, dass Abweichungen des arithmetischen Mittels vom Erwartungswert nur mit kleiner Wahrscheinlichkeit auftreten. Das starke Gesetz großer Zahlen zeigt darüber hinaus, dass diese Abweichungen für wachsendes n sogar verschwinden [13].

Betrachten wir die Indikatorfunktion, so ergibt sich ein wichtiger Spezialfall des (schwachen) Gesetzes großer Zahlen:

Korollar 1.44 (Das schwache Gesetz großer Zahlen von Jakob Bernoulli)
Seien $A_1, \ldots, A_n$ unabhängige Ereignisse mit gleicher Wahrscheinlichkeit p, dann gilt:

$$\lim_{n\to\infty} \mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{A_i\} - p\right| \geq \varepsilon\right) = 0.$$

Beweis:
Mit Satz 1.31(c) ergibt sich $\mu = \mathbb{E}(\mathbf{1}\{A_i\}) = \mathbb{P}(A_i) = p$ für $1 \leq i \leq n$. Außerdem gilt aufgrund der Definition der Varianz (bzw. des Erwartungswerts) unter Berücksichtigung von $\mu = p$:

$$\sigma^2 = \mathbb{V}(\mathbf{1}\{A_i\}) = (0-p)^2 \cdot (1-p) + (1-p)^2 \cdot p = p^2 - p^3 + p - 2p^2 + p^3 = p(1-p)$$

für $1 \leq i \leq n$. Durch Anwendung von Satz 1.41 folgt die Behauptung. ■

Bemerkung
Deutet man das Ereignis A_i als Treffer beim i-ten Versuch einer Bernoulli-Kette der Länge n, dann kann $R_n := 1/n \cdot \sum_{i=1}^{n} \mathbf{1}\{A_i\}$ als relative Trefferhäufigkeit verstanden werden [7].
Korollar 1.44 besagt somit, dass die Wahrscheinlichkeit, dass sich die relative Trefferhäufigkeit R_n einer Bernoulli-Kette der Länge n um mehr als ε von der Trefferwahrscheinlichkeit p unterscheidet, für wachsendes n gegen 0 geht. Die relative Trefferhäufigkeit konvergiert also stochastisch gegen die Trefferwahrscheinlichkeit.

Jakob Bernoulli gilt als Entdecker dieser Aussage. Sie ist das Hauptergebnis seines Buches über die Wahrscheinlichkeitsrechnung (Ars Conjectandi). Bemerkenswert ist, dass er zu seiner Zeit, weder die Begriffe Erwartungswert und Varianz noch die Tschebyschow-Ungleichung zur Verfügung hatte und dieses Ergebnis mittels direkter Rechnung erhielt [19, S. 219]. Mithilfe des starken Gesetzes großer Zahlen (Satz 1.42) ergibt sich in dieser Situation, dass R_n sogar fast sicher gegen p konvergiert.

Definition 1.45 (Standardisierung)
Ist X eine Zufallsvariable mit Erwartungswert $\mu = \mathbb{E}X < \infty$ und Varianz $\sigma^2 = \mathbb{V}(X) < \infty$, so heißt die Transformation

$$X \mapsto X^* := \frac{X - \mu}{\sigma}$$

die Standardisierung von X.

Bemerkung
Für eine standardisierte Zufallsvariable X^* gelten $\mathbb{E}(X^*) = 0$ und $\mathbb{V}(X^*) = 1$. Diese beiden Eigenschaften sind namensgebend und lassen sich unmittelbar mithilfe der Rechenregeln für den Erwartungswert und die Varianz verifizieren:

$$\mathbb{E}(X^*) = \mathbb{E}\left(\frac{X-\mu}{\sigma}\right) = \frac{1}{\sigma}(\mathbb{E}X - \mu) = 0,$$

$$\mathbb{V}(X^*) = \mathbb{V}\left(\frac{X-\mu}{\sigma}\right) = \mathbb{V}\left(\frac{X}{\sigma}\right) = \frac{1}{\sigma^2}\mathbb{V}(X) = 1. \quad \blacksquare$$

Schließlich wollen wir einen weiteren wichtigen Grenzwertsatz aus der Wahrscheinlichkeitstheorie nicht unerwähnt lassen. Da für den Beweis dieses Satzes weitere Hilfsmittel nötig wären, die den Rahmen dieser kurzen Einführung sprengen würden, wird dieser hier nicht angeführt (siehe dafür z. B. [7, S. 888]).

Satz 1.46 (Der Zentrale Grenzwertsatz von Lindeberg-Lévy)
Sei $(X_n)_{n\in\mathbb{N}}$ eine Folge von unabhängigen und identisch verteilten Zufallsvariablen mit endlicher, positiver Varianz. Setzen wir $\mu := \mathbb{E}(X_1)$, $\sigma^2 := \mathbb{V}(X_1)$ und $S_n := X_1 + \ldots + X_n$ dann gilt für alle $x \in \mathbb{R}$

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{S_n - n\mu}{\sigma\sqrt{n}} \le x\right) = \Phi(x).$$

Bemerkung 1.47
Durch Differenzbildung ergibt sich unmittelbar, dass für alle $a, b \in \mathbb{R}$ mit $a < b$ gilt:

$$\lim_{n\to\infty} \mathbb{P}\left(a \le \frac{S_n - n\mu}{\sigma\sqrt{n}} \le b\right) = \Phi(b) - \Phi(a).$$

Um den Zentralen Grenzwertsatz kurz und prägnant zu formulieren, bedenken wir, dass unter den obigen Voraussetzungen aufgrund von Bemerkung 1.32, Satz 1.37 und der identischen Verteiltheit Folgendes gilt:

$$\mathbb{E}(S_n) = \mathbb{E}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathbb{E}(X_i) = \sum_{i=1}^{n} \mu = n \cdot \mu,$$

$$\mathbb{V}(S_n) = \mathbb{V}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathbb{V}(X_i) = \sum_{i=1}^{n} \sigma^2 = n \cdot \sigma^2.$$

Somit ist $(S_n - n\mu)/\left(\sigma\sqrt{n}\right) = S_n^*$ die Standardisierung von S_n.

Wir halten daher fest, dass die Verteilung der standardisierten Summe S_n^* von unabhängigen und identisch verteilten Zufallsvariablen gegen die Standardnormalverteilung konvergiert, also kurz

$$S_n^* \xrightarrow{\mathcal{D}} Z \text{ für } Z \sim \mathcal{N}(0;1)$$

gilt. Aufgrund der Identität

$$\frac{S_n - n\mu}{\sigma\sqrt{n}} = \frac{\frac{1}{n}S_n - \mu}{\sigma/\sqrt{n}}$$

folgt aus dem Zentralen Grenzwertsatz, die für unsere weiteren Betrachtungen wichtige Aussage, dass für großes n das standardisierte arithmetische Mittel $\overline{X}_n^*$ in guter Näherung standardnormalverteilt beziehungsweise das arithmetische Mittel $\overline{X}_n$ $\mathcal{N}(\mu;\sigma^2/n)$-verteilt ist.

2 Zufallszahlen

In diesem Kapitel beschäftigen wir uns mit Zufallszahlen, welche die Basis für die konkrete Umsetzung der Monte-Carlo-Integration darstellen. Dafür wird zunächst gezeigt, wie auf dem Einheitsintervall gleichverteilte Zufallszahlen (Standardzufallszahlen) erzeugt werden können. Davon ausgehend wird dargelegt, wie zufällige Vektoren für mehrdimensionale Anwendungen konstruiert werden können. Schließlich wird die Inversionsmethode vorgestellt, die es uns (theoretisch) ermöglicht, beliebig verteilte Zufallszahlen aus Standardzufallszahlen zu gewinnen. Anschließend wird gezeigt, wie Zufallszahlen (bzw. zufällige Vektoren) mit MATHEMATICA erzeugt werden können. Am Ende dieses Kapitels lassen wir „den Computer würfeln", um das Gesetz großer Zahlen und den Zentralen Grenzwertsatz zu veranschaulichen.

2.1 Erzeugung gleichverteilter Zufallszahlen

Ein Zufallszahlgenerator ist ein Verfahren oder ein Mechanismus zur Erzeugung von Zufallszahlen [15]. Zum Beispiel könnten die auftretenden Zahlen beim Roulette, die Ergebnisse beim wiederholten Münzwurf oder die Lottozahlen als Zufallszahlen dienen. Doch nicht nur Glücksspiele sind Lieferanten von Zufallszahlen. Auch das thermische Rauschen einer elektronischen Schaltung kann dafür genutzt werden [38]. Solche physikalischen Zufallszahlgeneratoren bringen jedoch viele Nachteile mit sich: Sie sind relativ langsam und ihre Realisierung ist oft technisch aufwendig. Zudem sind die von ihnen erzeugten Zufallszahlen nicht reproduzierbar, was wiederum dazu führt, dass die mit ihnen erhaltenen Ergebnisse nicht überprüft werden können [29]. In diesem Zusammenhang ist der Mathematiker Robert R. Coveyou zu nennen, der meinte, dass die Erzeugung von Zufallszahlen, zu wichtig sei, um sie dem Zufall zu überlassen [9].
In der Praxis benutzt man deshalb lieber Pseudozufallszahlen, die von deterministischen Algorithmen erzeugt werden und nur zufällig erscheinen. Ein Pseudozufallszahlgenerator erzeugt eine deterministische und reproduzierbare Zahlenfolge $u_0, u_1, \ldots$ auf dem Einheitsintervall [0,1]. Diese Zahlen interpretiert man dann als Realisierungen einer Folge von unabhängigen und auf [0,1] gleichverteilten Zufallsvariablen[3] [39]. Der Kürze halber wird im Folgenden das Präfix Pseudo weggelassen. Exemplarisch führen wir nun einen einfachen Zufallszahlgenerator ein.

Definition 2.1 (Linearer Kongruenzgenerator)
Seien a, b, m und z_0 nichtnegative ganze Zahlen mit $z_0 \leq m - 1$. Ein Zufallszahlgenerator, der die Rekursionsvorschrift

$$z_{i+1} \equiv a \cdot z_j + b \ (\mathrm{mod}\ m) \text{ und } u_j \coloneqq \frac{z_j}{m} \text{ für } j = 0, 1, 2, \ldots$$

benutzt, heißt linearer Kongruenzgenerator. Wir nennen a den Faktor, b das Inkrement, m den Modul und z_0 den Anfangswert (bzw. Seed). Wir schreiben für einen solchen Zufallszahlgenerator kurz $\mathrm{LKG}(a, b, m, z_0)$.

[3] Wie eingangs erwähnt, nennen wir solche $\mathcal{U}[0,1]$-verteilten Zufallszahlen auch Standardzufallszahlen und bezeichnen sie stets mit u.

Aufgrund der Definition nimmt z_j nur Werte aus der Menge $\{0,1,\dots,m-1\}$ und u_j nur Werte aus $\{0/m, 1/m, \dots, (m-1)/m\} \subset [0,1]$ an. Folglich können mit einem linearen Kongruenzgenerator maximal m verschiedene Zufallszahlen erzeugt werden. Eine maximale Periodenlänge gleich m meint folglich, dass alle Zahlen j/m für $0 \leq j \leq m-1$ nach $(m-1)$-maligen Aufruf der obigen Rekursionsvorschrift aufgetreten sind. Lässt man einen linearen Kongruenzgenerator also lange genug „laufen", so wiederholen sich die Zufallszahlen periodisch. Damit ein linearer Kongruenzgenerator seine maximale Periodenlänge m aber auch tatsächlich erreicht, müssen die Parameter passend gewählt werden [19]. Um diese These zu untermauern, betrachten wir das folgende Beispiel:

Beispiel 2.2 (Linearer Kongruenzgenerator)
Der Übersichtlichkeit wegen verzichten wir bei diesem Beispiel auf die Normierung (z_j/m) der Zufallszahlen. Zunächst betrachten wir den linearen Kongruenzgenerator LKG(5,3,16,2). Mit ihm ergibt sich folgende Zahlenfolge:

$$2, 13, 4, 7, 6, 1, 8, 11, 10, 5, 12, 15, 14, 9, 0, 3, 2, 13, 4, \dots .$$

Betrachten wir hingegen LKG(5,4,16,2) dann ergeben sich folgende Zahlen:

$$2, 14, 10, 6, 2, 14, 10, \dots .$$

Während der obige Generator LKG(5,3,16,2) die maximale Periodenlänge 16 erreicht, können mit LKG(5,4,16,2) nur vier verschiedene Zufallszahlen gewonnen werden. Betrachtet man die von LKG(5,3,16,2) innerhalb einer Periode erzeugten Zufallszahlen, so ist ersichtlich, dass sie einfach eine Permutation der Zahlen 1, 2, …,16 darstellen. Der Generator LKG(14,3,16,1) verhält sich noch extremer, da er nur 1, 1, 1, 1, 1, 1, … liefert. ∆

Mithilfe zahlentheoretischer Überlegungen konnte Donald E. Knuth zeigen, dass folgender Satz [24, 3.2.1.2, Theorem A] gilt:

Satz 2.3 (Maximale Periodenlänge eines linearen Kongruenzgenerators)
Der lineare Kongruenzgenerator mit den Parametern a, b, m und z_0 erreicht die (maximale) Periodenlänge m genau dann, wenn folgende Bedingungen erfüllt sind:
(1) Das Inkrement b ist teilerfremd zum Modul m,
(2) Jede Primzahl, die den Modul m teilt, teilt auch $a-1$,
(3) Ist der Modul durch 4 teilbar, dann muss auch $a-1$ durch 4 teilbar sein.

Bemerkung 2.4 (Multiplikativer Kongruenzgenerator)
Setzt man das Inkrement b in Definition 2.1 gleich Null, so spricht man von einem multiplikativen Kongruenzgenerator. In diesem Fall muss außerdem $z_0 \neq 0$ gelten. Er weist eine maximale Periodenlänge von $m-1$ auf (die Null „fehlt"), wenn $m > 2$ eine Primzahl und a eine Primitivwurzel[4] modulo m ist. Ein Beweis dafür ist in [24, 3.2.1.2, Theorem B] zu finden.

Die Periodizität der erzeugten Zufallszahlen tritt nicht nur beim linearen Kongruenzgenerator auf, sondern auch bei allen anderen (rekursiv definierten) Zufallszahlgeneratoren [25]. Für das Erreichen der maximal möglichen Periodenlänge ist natürlich auch bei anderen Zufallszahlgeneratoren auf eine sorgfältige Auswahl der verwendeten Parameter zu achten. In

[4] Da m eine Primzahl ist, ist die prime Restklassengruppe $\mathbb{Z}_m^*$ zyklisch. Eine ganze Zahl a heißt Primitivwurzel modulo m, wenn ihre Restklasse $\overline{a}$ die Gruppe $\mathbb{Z}_m^*$ erzeugt.
Weil in diesem Fall die Gruppenordnung $\mathrm{ord}(\mathbb{Z}_m^*) = \varphi(m) = m-1$ ist, ist a genau dann Primitivwurzel modulo m, wenn $\mathrm{ord}_m(a) = m-1$ ist. Dabei bezeichnet $\mathrm{ord}_m(a)$ die (multiplikative) Ordnung von a modulo m, also die kleinste positive ganze Zahl k, die $a^k \equiv 1 \pmod{m}$ erfüllt [14; 34].

[25, Kapitel 4 und 5] werden weitere (rekursiv definierte) Zufallszahlgeneratoren und deren Eigenschaften vorgestellt.

Erreicht ein Zufallszahlgenerator seine maximale Periodenlänge m, dann simuliert er eine diskrete Gleichverteilung auf dem Grundraum $\Omega = \{0/m\,, 1/m\,, \dots\,, (m-1)/m\}$, da in diesem Fall $\mathbb{P}(\{\omega\}) = 1/m$ für alle $\omega \in \Omega$ gilt[5]. Dass diese diskrete Gleichverteilung für großes m eine gute Approximation der stetigen Gleichverteilung auf [0,1] darstellt, zeigt der nachfolgende Satz [19, 19.1].

Satz 2.5
Seien $m \in \mathbb{N}$, $\Omega = \{0/m\,, 1/m\,, \dots\,, (m-1)/m\}$ und $\mathbb{P}(\{\omega\}) = 1/m$ für alle $\omega \in \Omega$. Für ein beliebiges Teilintervall $[a, b]$ von [0,1] mit $0 \le a < b \le 1$ gilt:

$$|\mathbb{P}(\{x \in \Omega : a \le x \le b\}) - (b-a)| \le \frac{1}{m}\,.$$

Beweis:
Zu $a, b \in [0,1]$ mit $a < b$ existieren $i, j \in \{0,1, \dots, m-1\}$ mit

$$\frac{i}{m} \le a < \frac{i+1}{m} \text{ und } \frac{j}{m} \le b < \frac{j+1}{m}\,.$$

Ist $a = i/m$ dann gilt

$$\mathbb{P}(\{x \in \Omega : a \le x \le b)\}) = \frac{j+1-i}{m}\,.$$

Wegen

$$\frac{j-i}{m} \le b-a \le \frac{j+1-i}{m}$$

folgt die Behauptung.
Ist $a > i/m$ dann gilt

$$\mathbb{P}(\{x \in \Omega : a \le x \le b)\}) = \frac{j-i}{m}\,.$$

In diesem Fall folgt wegen

$$\frac{j-i-1}{m} < b-a < \frac{j+1-i}{m}$$

die Behauptung. ■

Eine große Periodenlänge m stellt somit ein wesentliches (aber nicht das einzige) Gütemerkmal eines Zufallszahlgenerators dar. Mithilfe von statistischen Tests (siehe z. B. [25, Kapitel 6 und 7; 24, 3.3]) kann ermittelt werden, ob die von einem gewissen Zufallszahlgenerator erzeugten Zufallszahlen tatsächlich als Realisierungen einer Folge von unabhängigen und auf [0,1] gleichverteilten Zufallsvariablen angesehen werden können.

[5] Im Falle des multiplikativen Kongruenzgenerators müsste die Null ausgeschlossen werden. Der Übersichtlichkeit wegen wird dies hier vernachlässigt.

2.2 Erzeugung gleichverteilter zufälliger Vektoren

Wir widmen uns der Frage, wie gleichverteilte zufällige Vektoren[6] auf $[0,1]^n$ erzeugt werden können. Sei $U = (U_1, \dots, U_n)$ ein auf $[0,1]^n$ gleichverteilter Zufallsvektor. Sind $U_1, \dots, U_n$ unabhängig (und auf $[0,1]$ gleichverteilt), so ergibt sich die gemeinsame Verteilung wegen Bemerkung 1.26(2) durch die Randverteilungen. Dies benutzen wir nun.

Um d gleichverteilte zufällige Vektoren zu erhalten, erzeugen wir zunächst d Realisierungen von U_i für $1 \leq i \leq n$ und erhalten

$$\begin{aligned} u_{11}, u_{12}, \dots, u_{1d} \quad & \text{(Realisierungen von } U_1\text{)} \\ u_{21}, u_{22}, \dots, u_{2d} \quad & \text{(Realisierungen von } U_2\text{)} \\ \vdots \qquad\qquad & \qquad\vdots \\ u_{n1}, u_{n2}, \dots, u_{nd}. \quad & \text{(Realisierungen von } U_n\text{)} \end{aligned}$$

Durch Bildung von n-Tupeln ergeben sich dann d gleichverteilte zufällige Vektoren

$$\begin{pmatrix} u_{11} \\ u_{21} \\ \vdots \\ u_{n1} \end{pmatrix}, \begin{pmatrix} u_{12} \\ u_{22} \\ \vdots \\ u_{n2} \end{pmatrix}, \dots, \begin{pmatrix} u_{1d} \\ u_{2d} \\ \vdots \\ u_{nd} \end{pmatrix} \in [0,1]^n. \tag{2.1}$$

Da wir die von einem Zufallszahlgenerator erzeugte Zahlenfolge $u_0, u_1, \dots \in [0,1]$ als Realisierungen einer Folge von unabhängigen und auf $[0,1]$-gleichverteilten Zufallsvariablen auffassen, könnten wir alternativ, um einen auf $[0,1]^n$ gleichverteilten zufälligen Vektor zu erhalten, die Zufallszahlen $u_0, u_1, \dots$ einfach (fortlaufend) zu n-Tupeln zusammenfassen:

$$\begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{pmatrix}, \begin{pmatrix} u_n \\ u_{n+1} \\ \vdots \\ u_{2n-1} \end{pmatrix}, \dots . \tag{2.2}$$

In beiden Fällen, können so für einen Zufallszahlgenerator mit Periodenlänge m maximal m/n verschiedene zufällige Vektoren erhalten werden, was wiederum zeigt, wie wichtig eine große Periodenlänge ist. Jedoch könnte man bei der ersten Variante auch n verschiedene Generatoren parallel nutzen, um eine längere Folge von zufälligen Vektoren zu generieren. Die beiden Varianten zur Erzeugung von zufälligen Vektoren werden in Abschnitt 2.4 noch einmal thematisiert und miteinander verglichen.

In diesem Zusammenhang wollen wir uns noch einmal kurz linearen Kongruenzgeneratoren zuwenden. Eine ihrer Schwächen ist die von ihnen erzeugte Gitterstruktur. Das heißt, dass die mit ihrer Hilfe erzeugten n-dimensionalen zufälligen Vektoren auf einem Gitter in $\mathbb{R}^n$, oder präziser auf relativ wenigen Hyperebenen in $[0,1]^n$, liegen. Diese Gitterstruktur ist jedoch kaum sichtbar, wenn die Anzahl der erzeugten Vektoren im Vergleich zu m/n sehr klein ist [19; 25].

Weil die Periodenlänge linearer Kongruenzgeneratoren nicht außerordentlich groß ist (Abschnitt 2.4), wird die Gitterstruktur für große Dimensionszahlen n zunehmend zum Problem.

[6] Die Bezeichnung „zufälliger Vektor“ dient der begrifflichen Abgrenzung. Ein zufälliger Vektor meint eine Realisierung eines Zufallsvektors. Seine Komponenten werden durch Zufallszahlen gebildet. Genau genommen meinen wir damit einen Ortsvektor, weshalb die Bezeichnung „zufälliger Punkt“ auch passend wäre.

Für hochdimensionale Anwendungen greift man daher lieber auf den Mersenne-Twister von Matsumoto und Nishimura zurück, der eine enorm große Periodenlänge von

$$m = 2^{19937} - 1 \approx 4{,}3 \cdot 10^{6001}$$

aufweist und ausgezeichnet bei statistischen Tests abschließt. Mit ihm können schließlich gleichverteilte zufällige Vektoren bis zu einer Dimensionszahl von $n = 623$ erzeugt werden [25; 27].

2.3 Erzeugung beliebig verteilter Zufallszahlen

Zusammenfassend können wir festhalten, dass es uns durch die Verwendung eines (guten) Zufallszahlgenerators gelingt, unabhängige und auf [0,1] gleichverteilte Zufallszahlen zu erzeugen. Wie die nachfolgenden Ausführungen, bei denen ich mich an [29] und [39] orientierte, zeigen werden, ist dies ausreichend um beliebig verteilte Zufallszahlen zu gewinnen.

Seien nun F die Verteilungsfunktion einer Zufallsvariable X und U eine auf [0,1] gleichverteilte Zufallsvariable. Wir definieren

$$G(u) \coloneqq \inf\{x \in \mathbb{R} : F(x) \geq u\} \text{ für } u \in (0,1).$$

Mit Satz 1.14 folgt dann, dass (wegen (c)) $G(u) \in \mathbb{R}$ gilt und (wegen (a)) G meßbar ist. Aus diesem Grund ist $G(U)$ eine Zufallsvariable (Bemerkung 1.9). Ist eine Verteilungsfunktion F stetig und streng monoton wachsend, so ist G die Umkehrfunktion von F, also F^{-1} [29, 4.1].

Satz 2.6
Seien F eine Verteilungsfunktion und U eine auf [0,1] gleichverteilte Zufallsvariable. Dann besitzt die Zufallsvariable $X = G(U)$ die Verteilungsfunktion F.

Beweis:
Sei $u \in (0,1)$ und $x \in \mathbb{R}$. Aufgrund der Definition von G folgt aus $u \leq F(x)$, dass $G(u) \leq x$ gilt. Umgekehrt ergibt sich aus $G(u) \leq x$ wegen der Monotonie und der rechtsseitigen Stetigkeit von F, dass $F(x) \geq F(G(u)) \geq u$ gilt. Somit ist $G(u) \leq x$ gleichbedeutend mit $u \leq F(x)$. Damit folgt nun

$$\mathbb{P}(G(U) \leq x) = \mathbb{P}\big(U \leq F(x)\big) = F(x). \quad \blacksquare$$

Damit ist gezeigt, dass wir die Transformation $x = G(u)$ einer Standardzufallszahl u als Realisierung einer Zufallsvariable X mit der Verteilungsfunktion F auffassen können [39]. Diese Art der Zufallszahlerzeugung trägt in der Literatur deshalb den Namen Inversionsmethode.

Im Speziellen gilt: Wenn X eine stetige Zufallsvariable mit streng monoton wachsender Verteilungsfunktion F ist, dann ist x die Lösung der Gleichung $F(x) = u$. Ist X eine diskrete Zufallsvariable, die Werte aus der Menge $\{x_0, x_1, \dots\}$ annimmt, und ist x_i der kleinste Wert von X, der $F(x_i) \geq u$ erfüllt, dann ist $x = x_i$ [39].

Beispiel 2.7
Ist die Zufallsvariable X auf $[a, b]$ gleichverteilt, dann hat ihre Verteilungsfunktion nach Bemerkung 1.22 folgende Gestalt:

$$F(x) = \frac{x - a}{b - a} \text{ für } a \leq x \leq b.$$

Um eine Realisierung von X aus einer Standardzufallszahl $u \in (0,1)$ zu erhalten, lösen wir die Gleichung $F(x) = u$ nach x auf.

Wir erhalten damit:

$$\frac{x-a}{b-a} = u \Leftrightarrow x - a = (b-a)u \Leftrightarrow x = a + (b-a)u.$$

Die Inversionsmethode liefert somit das bereits aus Satz 1.23 bekannte Ergebnis. ∆

Beispiel 2.8

Wir betrachten nun die Menge $A = [a,b] \times [c,d] \subset \mathbb{R}^2$. Der Zufallsvektor X sei auf A gleichverteilt. Um nun m unabhängige Realisierungen von X zu bekommen, erzeugen wir zunächst $2m$ Standardzufallszahlen

$$u_{11}, u_{12}, \dots, u_{1m}$$

$$u_{21}, u_{22}, \dots, u_{2m}.$$

Wir erhalten dann eine Folge $x_1, x_2, \dots, x_m$ von unabhängigen auf A gleichverteilten zufälligen Vektoren durch die Rekursionsvorschrift

$$x_i \coloneqq \begin{pmatrix} a + (b-a) \cdot u_{1i} \\ c + (d-c) \cdot u_{2i} \end{pmatrix} \text{ für } i = 1,2, \dots, m.$$

Der theoretische Hintergrund für diese Konstruktionsvorschrift ist durch Bemerkung 1.26(2) und Satz 1.27 gegeben. ∆

Beispiel 2.9

Eine auf $\{1, \dots, m\}$ gleichverteilte Zufallsvariable X besitzt die Verteilungsfunktion

$$F(x) = \frac{i}{m} \text{ für } i \leq x < i+1 \text{ und } i \in \{1, \dots, m\}.$$

Ist $u \in (0,1)$ eine Standardzufallszahl und $k \in \{1, \dots, m\}$ der kleinste Wert, für den $F(k) \geq u$ ist, so gilt

$$\frac{k}{m} \geq u > \frac{k-1}{m} \Leftrightarrow k \geq mu > k-1.$$

Somit ist $k = \lceil mu \rceil$ eine Realisierung von X. ∆

Nach Satz 2.6 ist die Inversionsmethode grundsätzlich auf jede Verteilung anwendbar. Ob ihr Einsatz praktikabel ist, hängt jedoch davon ab, wie aufwändig die Bestimmung von $G(u)$ ist. Solange $G(u)$ eine einfache analytische Form hat, ist der Einsatz der Inversionsmethode vorteilhaft. Für die Verteilungsfunktion der Standardnormalverteilung existiert kein geschlossener analytischer Ausdruck. Für die Erzeugung von standardnormalverteilten Zufallszahlen wird man daher auf ein anderes Verfahren, wie zum Beispiel die Box-Muller-Methode (siehe z. B. [19, 32.11]), zurückgreifen, um eine numerische Näherung zu vermeiden.

Wie wir im nachfolgenden Abschnitt sehen werden, haben wir mit MATHEMATICA ein starkes Werkzeug für die Erzeugung von Zufallszahlen (bzw. zufälligen Vektoren) zur Verfügung, weshalb an dieser Stelle[7] auf die Darstellung von weiteren Verfahren zur Erzeugung von Zufallszahlen mit vorgegebener Verteilung verzichtet wird. Für weitere Informationen zu diesem Thema verweise ich auf [25, II].

[7] In Exkurs 3.9 wird eine weitere Methode zur Erzeugung beliebig verteilter Zufallszahlen thematisiert.

2.4 Zufallszahlerzeugung mit Mathematica

Nun widmen wir uns der konkreten Zufallszahlerzeugung mit MATHEMATICA. Zunächst möchte ich aber kurz auf Grundlegendes im Zusammenhang mit dieser Arbeit eingehen.

MATHEMATICA-Befehle werden immer in `Blau` oder `Rot` und in MATHEMATICA-Standardschrift dargestellt, um sie besser kenntlich zu machen. Standardmäßig wird dabei die Farbe Blau genutzt. Werden Simulationen im Rahmen von Beispielen erstellt, so werden die nötigen Eingaben rot eingefärbt. Blaue Befehle dienen dann ausschließlich der Erläuterung von roten Eingaben und sind deshalb bei der fortlaufenden Erstellung nicht einzugeben. Um keine Konflikte hinsichtlich der Variablenbenennung zu erhalten, sollte für jede Simulation ein eigenes MATHEMATICA-Notebook erstellt werden. Die vollständigen Eingabe-Codes für die im Rahmen dieser Arbeit erstellten Simulationen und Grafiken werden im Anhang gesammelt angeführt.

Wie bereits in der Einleitung erwähnt, wird die Konstruktion von Simulationen sehr ausführlich beschrieben. Zudem wird ein Aufbau gewählt, der sehr nahe an den theoretischen Ausführungen liegt. Um dies zu erreichen, werden vorrangig Listen benutzt. In Abschnitt 2.5 wird deshalb im Rahmen von zwei Beispielen unter anderem die „Listenarithmetik“ von MATHEMATICA thematisiert. Vor allem für die Erzeugung von Grafiken werden häufig diverse Zusatzbefehle genutzt, die einfach eine schönere Darstellung eines Ergebnisses liefern sollen und häufig nicht explizit erklärt werden. Für weiterführende Informationen zu den verwendeten Befehlen verweise ich deshalb auf das DOCUMENTATION CENTER, welches in MATHEMATICA integriert ist, aber auch über die Website von WOLFRAM[8] erreicht werden kann.

Doch nun zur Zufallszahlerzeugung: MATHEMATICA stellt mehrere Zufallszahlgeneratoren bereit. Standardmäßig benutzt MATHEMATICA für die Erstellung von Zufallszahlen den Generator „`ExtendedCA`“, der auf einem zellulären Automaten basiert und nach eigenen Angaben eine extrem hohe Zufälligkeit der Zahlen garantiert. Da keinerlei Informationen über diesen Generator und seine tatsächlichen Eigenschaften gefunden werden konnten, werde ich jedoch auf seine Verwendung verzichten.

Der in MATHEMATICA implementierte multiplikative Kongruenzgenerator verwendet als Modul die Primzahl $m = 2^{61} - 1 \approx 2{,}3 \cdot 10^{18}$. Die Zahl $a = 1\,283\,839\,219\,676\,404\,755$ wird als Faktor benutzt. Ob $\mathrm{ord}_m(a) = m - 1$ gilt und die Zahl a somit eine Primitivwurzel modulo m darstellt, kann mit dem Befehl `MultiplicativeOrder[a,m]` geprüft werden. Da dies erfüllt ist, erreicht dieser Generator nach Bemerkung 2.4 die maximale Periodenlänge $m - 1$. Schließlich ist auch der Mersenne-Twister in MATHEMATICA implementiert. Vorerst werden wir ausschließlich diese beiden Zufallszahlgeneratoren verwenden.

Um einen der beiden Generatoren zu verwenden, benutzen wir den Befehl `SeedRandom`. Mit der Eingabe von

```
SeedRandom[1,Method→"Congruential"]
```

verwenden wir beispielweise den multiplikativen Kongruenzgenerator bei einem Anfangswert (oder auch Seed bzw. Initialisierung) von $z_0 = 1$ für die Zufallszahlerzeugung. Wollen wir den Mersenne-Twister mit derselben Initialisierung nutzen, so ersetzen wir die Eingabe

[8] http://reference.wolfram.com/language/

"Congruential" durch "MersenneTwister". Gibt man die Initialisierung und den verwendeten Generator an, so können die Zufallszahlen reproduziert werden.

Der Befehl RandomReal[] liefert eine Standardzufallszahl u. Mit RandomReal[{a,b}] wird eine auf dem Intervall $[a, b]$ gleichverteilte Zufallszahl x (für $a, b \in \mathbb{R}$ und $a < b$) erzeugt. Wie einfach nachgeprüft werden kann, wird dabei die Transformation $x = a + (b - a) \cdot u$ (vgl. Beispiel 2.7) verwendet.

Sehr nützlich ist der Befehl RandomVariate mit dem Zufallszahlen und zufällige Vektoren, die einer gewissen Verteilung folgen, direkt erzeugt werden können. MATHEMATICA bietet eine umfassende Sammlung von diskreten, stetigen und multivariaten Verteilungen an (siehe hierzu DOCUMENTATION CENTER). Zum Beispiel wird durch UniformDistribution[n] die $\mathcal{U}[0,1]^n$-Verteilung bezeichnet. Eine Gleichverteilung auf $[a_1, b_1] \times \ldots \times [a_n, b_n] \subset \mathbb{R}^n$ wird durch die Eingabe UniformDistribution[{{a[1],b[1]},…,{a[n],b[n]}}] repräsentiert. Die Eingabe NormalDistribution[μ,σ] symbolisiert beispielweise eine Normalverteilung $\mathcal{N}(\mu; \sigma^2)$. Die m Zufallszahlen, die mit dem Befehl RandomReal[{0,1},m] erzeugt werden, stimmen mit jenen Zufallszahlen, die sich mit RandomVariate[UniformDistribution[1],m] ergeben, überein.

Exemplarisch wird nun die Erzeugung von m auf $[0,1]^n$ gleichverteilten zufälligen Vektoren mit MATHEMATICA diskutiert. Dafür könnte etwa der Befehl RandomReal[{0,1},{m,n}] genutzt werden. Überprüft man die so erhaltenen Vektoren, so erkennt man, dass MATHEMATICA in der Erzeugung gemäß (2.2) vorgeht. Mit RandomVariate[UniformDistribution[n],m] erhält man jedoch für $m > 1$ andere Vektoren. Dies liegt an der unterschiedlichen Bildung der n-Tupel. Im ersten Fall wird eine $m \cdot n$ lange Liste von Zufallszahlen erzeugt, die dann gemäß (2.2) in n-Tupel aufgeteilt wird. Im zweiten Fall werden n Listen $L_1, \ldots, L_n$ (mit Länge m) von Zufallszahlen durch n-malige Hintereinanderausführung von RandomReal[{0,1},m] erzeugt. Für $1 \leq i \leq m$ und $1 \leq j \leq n$ bezeichnen wir mit $L_j[[i]]$ das i-te Element der j-ten Liste. Damit ergeben sich n-Tupel der Form

$$\begin{pmatrix} L_1[[1]] \\ \vdots \\ L_n[[1]] \end{pmatrix}, \begin{pmatrix} L_1[[2]] \\ \vdots \\ L_n[[2]] \end{pmatrix}, \ldots, \begin{pmatrix} L_1[[m]] \\ \vdots \\ L_n[[m]] \end{pmatrix}.$$

Dies entspricht somit der Erzeugung gemäß (2.1). Da die erste Variante nur für Vektoren auf $[a, b]^n$ zweckmäßig ist, werden wir für die Erzeugung von gleichverteilten zufälligen Vektoren stets die zweite Variante nutzen.

Bevor wir uns zwei konkreten Beispielen widmen, möchte ich noch kurz auf die „Listenarithmetik“ von MATHEMATICA eingehen. Nehmen wir beispielweise zwei einfache Listen {1,2,3} und {3,4,5}. Diese beiden Listen können als zwei Vektoren auf $\mathbb{R}^3$ angesehen werden. Die üblichen Rechenoperationen, wie etwa die Vektoraddition, Skalarmultiplikation oder das Skalarprodukt können dann intuitiv umgesetzt werden (siehe „Vector Operations“ im DOCUMENTATION CENTER). Solche Listen ausschließlich als Vektoren anzusehen, würde jedoch zu kurz greifen.

Der Ausdruck {1,2,3}+2 ergäbe im Sinne der Vektorrechnung nichts Sinnvolles. MATHEMATICA liefert hingegen als Ausgabe {3,4,5}, addiert also Zwei zu jedem Listenelement. Analog wird die Eingabe {1,2,3}^2 verstanden: Jedes Element der Liste wird quadriert und es ergibt

sich als Ausgabe `{1,4,9}`. Dies funktioniert mit beliebigen reellen Funktionen. Geben wir zum Beispiel

```
g[x_] = Sin[x];g[{1,2,3}]
```

ein, so ergibt sich `{Sin[1],Sin[2],Sin[3]}` als Ausgabe. Ergänzen wir `//N` bei der Eingabe, dann erhalten wir direkt das numerische Ergebnis `{0.841471,0.909297,0.14112}`. Gleiches gilt, wenn wir statt `{1,2,3}` den Ausdruck `{1.,2.,3.}` verwenden, da die Zahlen in der zweiten Liste als Dezimalzahlen erkannt werden (mehr dazu in Beispiel 3.16). Im Hinblick auf spätere umfassende Simulationen ist zu sagen, dass sich die Verwendung von Dezimalzahlen (durch Setzen eines Punkts) anstatt rationaler Zahlen positiv auf die Rechendauer auswirken kann. Denn die Verwendung rationaler Zahlen veranlasst MATHEMATICA mit höchster Genauigkeit zu arbeiten. Das Kürzen von Brüchen und die damit einhergehende Primfaktorzerlegung stellen jedoch einen größeren Rechenaufwand dar.

Die Erzeugung von Listen wird häufig mit dem einfachen `Table`-Befehl durchgeführt. MATHEMATICA bietet auch eine parallelisierte Version dieses Befehls (`ParallelTable`). Dabei werden (sofern vorhanden) mehrere Prozessorkerne für die Erzeugung der Liste genutzt, was zu einer Verringerung der Rechendauer führen kann. Im Rahmen dieser Arbeit wird jedoch ausschließlich mit dem einfachen `Table`-Befehl gearbeitet. Es bleibt der Leserin oder dem Leser überlassen, die parallelisierte Version (für weitere Untersuchungen mit größeren Stichprobenumfängen und in Abhängigkeit vom verwendeten Computer) einzusetzen. Wir kommen nun zu den Beispielen.

2.5 Erste Simulationen mit Mathematica

Die erste Simulation dieser Arbeit soll nicht nur das Gesetz großer Zahlen veranschaulichen, sondern darüberhinaus die, auch bei den nachfolgenden Simulationen verwendete, Eingabe-Grundstruktur zeigen. Wir bedienen uns dazu eines klassischen Beispiels der Stochastik.

Beispiel 2.10
Die Zufallsvariable X beschreibe die Augenzahl beim einfachen Wurf eines (fairen) Würfels. Sie besitzt dann eine (diskrete) Gleichverteilung auf der Menge {1,2,3,4,5,6}. Gemäß Bemerkung 1.30(1b) ergibt sich der Erwartungswert von X zu

$$\mathbb{E}X = \sum_{i=1}^{6} i \cdot \mathbb{P}(X = i) = \sum_{i=1}^{6} i \cdot \frac{1}{6} = \frac{21}{6} = 3{,}5.$$

Nun versuchen wir eine Folge von Zufallszahlen $x_1, \dots, x_n$ zu erzeugen, die wir als Realisierungen von X ansehen können. Wir könnten dafür wegen Beispiel 2.9 einfach Standardzufallszahlen $u_0, \dots u_{n-1} \in (0,1)$ mittels $x_i := \lceil 6 \cdot u_{i-1} \rceil$ für $1 \leq i \leq n$ transformieren. MATHEMATICA verfügt jedoch über den Befehl `RandomInteger`, mit dem unmittelbar eine Folge von gleichverteilten ganzen Zufallszahlen gewonnen werden kann.
Wir wollen nun beobachten, wie sich das arithmetische Mittel dieser Zahlen für wachsendes n verhält. Dafür legen wir zunächst fest, wie lange die Folge von Zufallszahlen (bzw. wie groß der Stichprobenumfang) sein soll, und setzen zum Beispiel

```
n = 2000;.
```

Wird die Eingabezeile mit einem Strichpunkt versehen, so wird verhindert, dass der Wert nach der Auswertung in der Ausgabe erscheint.

Mit

```
SeedRandom[1,Method→"Congruential"];.
```

verwenden wir den multiplikativen Kongruenzgenerator (siehe 2.4) bei einem Anfangswert von Eins für die Zufallszahlerzeugung. Durch die Eingabe von

```
Liste = Table[Mean[RandomInteger[{1,6},m]],{m,1,n}];
```

erzeugen wir eine Liste der arithmetischen Mittel. Wir bezeichnen das arithmetische Mittel der Zufallszahlen $x_1, \dots, x_m \in \{1,2,3,4,5,6\}$ mit $\overline{x}_m = 1/m \sum_{i=1}^{m} x_i$ für $1 \leq m \leq n$. Somit hat die Liste die Gestalt $\{\overline{x}_1, \overline{x}_2, \dots, \overline{x}_n\}$. Nun plotten wir diese Liste mithilfe von

```
G1 = ListLinePlot[Liste,PlotRange→{2.5,4.5},PlotStyle→Blue];.
```

Wir verzichten dabei vorerst auf die Ausgabe, da wir auch den Erwartungswert grafisch darstellen möchten. Dafür geben wir

```
G2 = Plot[3.5,{x,0,n},PlotStyle→{Red,Thick}];
```

ein. Nun kombinieren wir die beiden Grafiken mittels

```
Show[G1,G2,AspectRatio→1/3,AxesLabel→{"m","x̄ₘ",},
    LabelStyle→{Large,Black}],
```

wobei wir das Seitenverhältnis mit dem Faktor 1/3 skalieren und die Achsen beschriften. Zu erwähnen ist, dass es für weitere Untersuchungen von Vorteil ist, alle Eingaben in einem einzigen Eingabefeld durchzuführen, da dann nur dieses Feld ausgewertet werden muss. Abbildung 2.1 zeigt die von MATHEMATICA erzeugte Grafik nach der Auswertung. Im Einklang mit dem Gesetz großer Zahlen erkennt man, dass sich das arithmetische Mittel für wachsendes n um den Erwartungswert stabilisiert (vgl. Bemerkung 1.43).

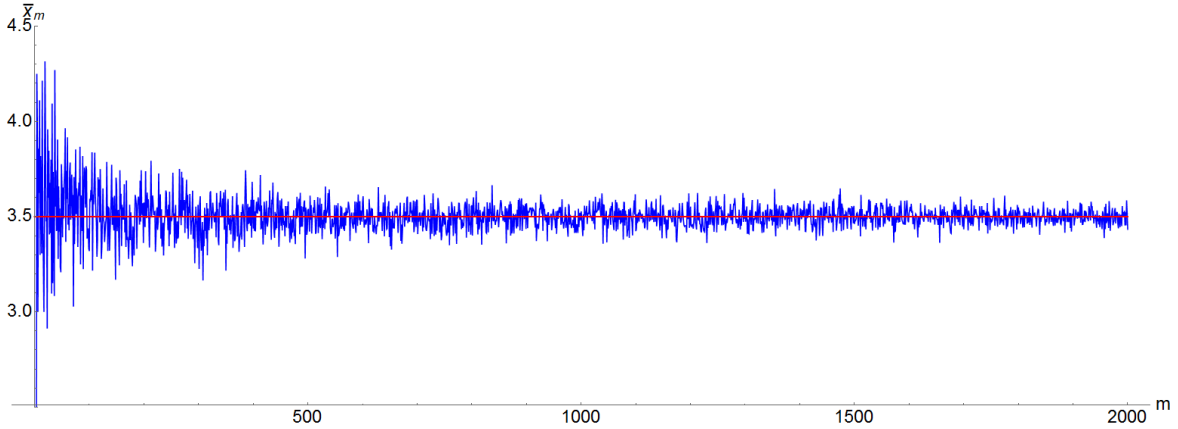

Abbildung 2.1 Simuliertes arithmetisches Mittel der Augenzahlen beim Würfelwurf für wachsenden Stichprobenumfang m

Eine Änderung des Anfangswerts (Seed) kann zu einer Vergrößerung (oder Verkleinerung) der Schwankungsbreite führen (wähle zum Beispiel 41 oder 105 als Seed). Selbiges gilt, wenn ein anderer Zufallszahlgenerator verwendet wird. Mit diesem Aspekt werden wir uns aber später noch genauer auseinandersetzen. ∆

Den Abschluss dieses Kapitels stellt eine Simulation zum Zentralen Grenzwertsatz dar. Wir bleiben dabei bei den Würfeln.

Beispiel 2.11

Sei $n \in \mathbb{N}$. Wir betrachten nun einen n-fachen Würfelwurf, beschrieben durch die (unabhängigen) Zufallsvariablen $X_1, \dots, X_n$. Die Zufallsvariable X_i bezeichne folglich die beim i-ten Versuch geworfene Augenzahl für $1 \leq i \leq n$. Die mit n Würfen durchschnittlich erzielte Augen-

zahl beschreiben wir durch die Zufallsvariable $\overline{X}_n := 1/n \sum_{i=1}^{n} X_i$. Wir wollen nun zeigen, dass sich die Verteilung von $\overline{X}_n$ für großes n einer Normalverteilung annähert.

Zunächst berechnen wir die Varianz von X_1. Wegen Beispiel 2.10 wissen wir bereits, dass $\mathbb{E}(X_1) = 21/6 = 3{,}5$ ist. Die Varianz von X_1 ergibt sich nach dem Verschiebungssatz (1.36(b)) zu

$$\mathbb{V}(X_1) = \mathbb{E}(X_1^2) - \big(\mathbb{E}(X_1)\big)^2 = \sum_{j=1}^{6} j^2 \cdot \frac{1}{6} - \left(\frac{21}{6}\right)^2 = \frac{91}{6} - \frac{49}{4} = \frac{35}{12}.$$

Aufgrund des Beweises von Satz 1.41 (also insbesondere aufgrund der identischen Verteiltheit und Unabhängigkeit der X_i) ergeben sich somit

$$\mathbb{E}\big(\overline{X}_n\big) = \frac{21}{6} \text{ und } \mathbb{V}\big(\overline{X}_n\big) = \frac{35}{12n}.$$

Nach dem Zentralen Grenzwertsatz (1.46) erwarten wir, dass $\overline{X}_n$ für großes n in guter Näherung $\mathcal{N}(21/6\,;35/12n)$-verteilt ist. Um dies zu überprüfen, erzeugen wir m Realisierungen von $\overline{X}_n$ und betrachten deren Häufigkeitsverteilung. Zunächst legen wir die beiden Parameter $n, m \in \mathbb{N}$ fest. Als Zufallszahlgenerator verwenden wir wieder den multiplikativen Kongruenzgenerator bei einem Anfangswert von Eins und geben

```
n = 500;m = 100 000;
SeedRandom[1,Method → "Congruential"];
```

ein. Nun erzeugen wir einen „Satz“ von auf {1,2,3,4,5,6} gleichverteilten Zufallszahlen durch

```
Liste = Table[j = RandomInteger[{1,6},n],{j,1,m}];
```
(2.3)

Die auf diese Weise erhaltene Liste besitzt dann die Gestalt

$$\{\{x_{11}, \dots, x_{1n}\}, \{x_{21}, \dots, x_{2n}\}, \dots, \{x_{m1}, \dots, x_{mn}\}\}.$$

Diese Liste besteht also aus m Teillisten. Jede Teilliste enthält n Zufallszahlen, die wir als Realisierungen von $X_1, \dots, X_n$ ansehen können. Der Computer würfelt also m Wurfserien (jeweils vom Umfang n).

Um m Realisierungen von $\overline{X}_n$ zu erzeugen, berechnen wir für jede Teilliste $\{x_{j1}, \dots, x_{jn}\}$ das arithmetische Mittel $\overline{x}_j = 1/n \sum_{i=1}^{n} x_{ji}$ für $1 \le j \le m$ durch

```
ListeM = Table[Mean[Liste[[j]]],{j,1,m}];
```
(2.4)

Die so erhaltene Liste besitzt demnach die Gestalt $\{\overline{x}_1, \dots, \overline{x}_m\}$, enthält also m Realisierungen von $\overline{X}_n$. Die Häufigkeitsverteilung der $\overline{x}_j$ stellen wir mithilfe von

```
G1 = Histogram[ListeM,50,"PDF"];
```

dar, erzeugen also ein Histogramm mit 50 Klassen. Die Höhe der Balken entspricht dabei der Häufigkeitsdichte.

Um die Annäherung dieser Häufigkeitsverteilung an die Normalverteilung $\mathcal{N}(21/6\,;35/12n)$ deutlich erkennbar zu gestalten, plotten wir deren Dichte (`PDF`). Für eine übersichtlichere Gestaltung der Eingabe, setzen wir

```
w = 21/6;s = Sqrt[35/(12n)];
```

und plotten die Dichte der $\mathcal{N}(21/6\,;35/12n)$-Verteilung[9] im Intervall $[w-4s, w+4s]$ mit

```
G2 = Plot[PDF[NormalDistribution[w,s],x],{x,w-4s,w+4s},
    PlotStyle → {Black,Thick}];
```

[9] Der Befehl `NormalDistribution` erwartet als zweites Argument die Standardabweichung anstatt der Varianz.

Mithilfe von

```
Show[G1,G2,Axes→{True,False},AxesStyle→Thick,
    LabelStyle→{Large,Black},PlotRange→All,AspectRatio→1/3]
```

werden die beiden Grafiken verbunden. Dabei werden ein paar Zusatzspezifikationen angegeben, um einen ansprechenden Plot zu erhalten. Abbildung 2.2 zeigt das Ergebnis dieser Simulation.

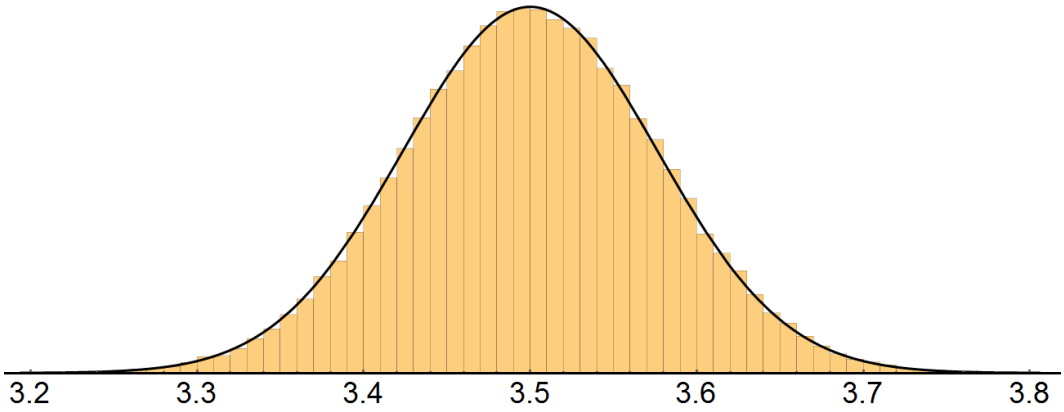

Abbildung 2.2 Annäherung der Häufigkeitsverteilung des arithmetischen Mittels beim n-fachen Würfelwurf an eine Normalverteilung für $n = 500$ und $m = 100\,000$

Wir können hier also in guter Näherung von einer Normalverteilung ausgehen und diese benutzen, um zum Beispiel die Wahrscheinlichkeit, beim 500-fachen Würfelwurf eine durchschnittliche Augenzahl zwischen 3,4 und 3,6 zu erzielen, (näherungsweise) zu ermitteln. Es gilt also, dass

$$\mathbb{P}(3{,}4 \leq \overline{X}_n \leq 3{,}6) \simeq F(3{,}6) - F(3{,}4)$$

ist, wobei F die Verteilungsfunktion (`CDF`) der $\mathcal{N}(21/6\,;35/6000)$-Verteilung ist. Mit den zuvor definierten Parametern ergibt sich die gesuchte Wahrscheinlichkeit dann mittels

```
CDF[NormalDistribution[w,s],3.6]−CDF[NormalDistribution[w,s],3.4]
```

oder alternativ durch

```
Probability[3.4≤x≤3.6,x\[Distributed]NormalDistribution[w,s]]
```

zu rund 81 Prozent. ∆

Anmerkung 2.12

Wie wir gesehen haben, ermöglicht die Verwendung von Listen eine kurze und übersichtliche Gestaltung der Eingaben. Werden Listen jedoch zu umfangreich, so führt dies zu einer starken Belastung des Arbeitsspeichers. Im Rahmen von Beispiel 3.14 wenden wir uns dieser Problematik genauer zu, und stellen eine Methode vor, mit der die Idee hinter (2.3) und (2.4) viel effizienter umgesetzt werden kann.

3 Monte-Carlo-Integration

In diesem Kapitel wird die Monte-Carlo-Integration (kurz MCI) erklärt. Neben der sogenannten naiven Monte-Carlo-Integration wird auch ein Spezialfall, die hit-or-miss Monte-Carlo-Integration, thematisiert. In diesem Zusammenhang stoßen wir auf eine weitere Methode zur Erzeugung von beliebig verteilten Zufallszahlen, die in einem kurzen Exkurs behandelt wird. Schließlich wird die Genauigkeit der naiven Monte-Carlo-Integration diskutiert. Die theoretischen Ausführungen werden von mehreren Beispielen, die mit MATHEMATICA realisiert werden, begleitet. Als Hauptquellen für dieses Kapitel dienten mir [8], [25], [29] und [39].

3.1 Eindimensionale naive Monte-Carlo-Integration

Sei $g : \mathbb{R} \to \mathbb{R}$ eine integrierbare Funktion. Unser Ziel ist es, den Wert des Integrals

$$I = \int_a^b g(x)\,dx$$

zu bestimmen. Wir wählen dafür einen wahrscheinlichkeitstheoretischen Zugang.
Sei X eine auf dem Intervall $[a, b]$ gleichverteilte Zufallsvariable. Für den Erwartungswert der Zufallsvariable $g(X)$ gilt dann

$$\mathbb{E}\big(g(X)\big) = \int_{\mathbb{R}} g(x) \cdot f(x)\,dx = \frac{1}{b-a}\int_a^b g(x)\,dx.$$

Somit können wir das gesuchte Integral I ausdrücken als

$$I = (b-a) \cdot \mathbb{E}\big(g(X)\big). \tag{3.1}$$

Gelingt es uns also, den Erwartungswert der Zufallsvariable $g(X)$ zu bestimmen, so haben wir unmittelbar einen Wert für das gesuchte Integral gefunden. Das Gesetz großer Zahlen verrät uns, dass das arithmetische Mittel einer Folge von unabhängigen identisch verteilten Zufallsvariablen einen geeigneten Schätzwert darstellt.
Seien daher $X_1, X_2, \ldots$ unabhängige und auf $[a, b]$ gleichverteilte Zufallsvariablen. Aufgrund der vorausgesetzten Integrierbarkeit von g sind $g(X_1), g(X_2), \ldots$ ebenfalls unabhängige und identisch verteilte Zufallsvariablen mit existierendem Erwartungswert (Bemerkung 1.26(1)). Nach dem starken Gesetz großer Zahlen gilt dann

$$\frac{1}{n}\sum_{i=1}^{n} g(X_i) \xrightarrow{\text{f.s.}} \mathbb{E}\big(g(X)\big).$$

Erzeugen wir nun n Zufallszahlen $x_1, \ldots, x_n \in [a, b]$, die wir als Realisierungen von $X_1, \ldots, X_n$ ansehen, dann ergibt sich, dass

$$\int_a^b g(x)\,dx \simeq (b-a) \cdot \frac{1}{n}\sum_{i=1}^{n} g(x_i)$$

ist. Mit einer großen Anzahl n von Zufallszahlen (bzw. einem großen Stichprobenumfang n) erhalten wir dann (fast sicher) den exakten Wert des Integrals. Wir fassen dies nun zusammen.

Algorithmus 3.1 (Eindimensionale naive Monte-Carlo-Integration)
Sei $g : \mathbb{R} \to \mathbb{R}$ eine integrierbare[10] Funktion und seien $a, b \in \mathbb{R}$ mit $a < b$. Um eine Näherung für den Wert des Integrals

$$I = \int_a^b g(x)\,dx$$

zu ermitteln, sind folgende Schritte durchzuführen:
(1) Erzeuge n auf $[a, b]$ gleichverteilte Zufallszahlen $x_1, \ldots, x_n$.
(2) Bestimme die zugehörigen Funktionswerte $g(x_1), \ldots, g(x_n)$.
(3) Approximiere I durch den Schätzer

$$I_n = \frac{(b-a)}{n} \sum_{i=1}^{n} g(x_i).$$

Bemerkung
Gleichung (3.1) ähnelt dem (ersten) Mittelwertsatz der Integralrechnung [22, 85.5], der im Speziellen für eine stetige Funktion $g : [a, b] \to \mathbb{R}$ besagt, dass eine Stelle $\xi \in [a, b]$ existiert, sodass

$$\int_a^b g(x)\,dx = (b-a) \cdot g(\xi)$$

gilt. Dabei ist zu bedenken, dass jede Stelle $x \in [a, b]$ mit gleichem Gewicht zum Integral beiträgt, womit der Satz aussagt, dass eine Stelle $\xi \in [a, b]$ existiert, an dem die Funktion g ihren Mittelwert $g(\xi)$ im stochastischen Sinn annimmt.

Nach diesem kurzen Exkurs in die Analysis wird nun ein erstes Beispiel präsentiert, welches als Standardbeispiel in der Literatur anzusehen ist.

Beispiel 3.2 (Näherungsweise Bestimmung von π - Variante 1)
Wir suchen einen Schätzwert für das Integral

$$I = \int_0^1 4\sqrt{1-x^2}\,dx.$$

Mittels Substitution $x = \sin(u)$ und unter Verwendung von trigonometrischen Zusammenhängen ergibt sich schnell, dass $I = \pi$ ist. Der Wert des Integrals kann also exakt berechnet werden. Folglich können wir mithilfe der Monte-Carlo-Integration einen numerischen Näherungswert für π errechnen.
Zunächst definieren wir den Integranden durch

```
g[x_] = 4.*Sqrt[1-x^2];.
```

Um Reproduzierbarkeit zu gewährleisten, legen wir den Zufallszahlgenerator und seinen Anfangswert mittels

```
SeedRandom[1,Method → "Congruential"];
```

fest. Da die Intervalllänge Eins beträgt, können wir den Schätzer I_n direkt mithilfe von

```
Mean[g[RandomReal[{0,1},n]]]
```

realisieren.

[10] Die hier vorausgesetzte Integrierbarkeit wird in Beispiel 3.16 thematisiert und ist mit Vorsicht zu genießen.

Um die Anzahl der verwendeten Zufallszahlen n variieren zu können, verwenden wir eine `Manipulate`-Umgebung. Die bisherigen Eingaben bezeichne ich der Übersichtlichkeit wegen kurz mit `Input`. Wir geben nun

```
Manipulate[Input,{n,1,5 000 000,1}]
```

ein. Werten wir diese Eingabe aus, so können wir n zwischen Eins und fünf Millionen mit dem Schieberegler variieren und beobachten, wie sich der Näherungswert für π entwickelt (siehe Abbildung 3.1).

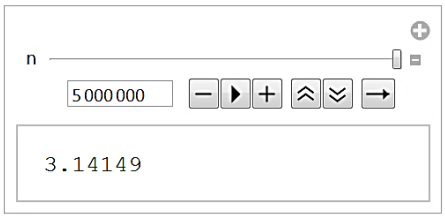

Abbildung 3.1 Näherungsweise Bestimmung von π mit naiver Monte-Carlo-Integration

Bei diesem Anfangswert erkennt man, dass die ersten beiden korrekten Nachkommastellen ab zirka $n = 100\,000$ und die ersten drei Nachkommastellen ab ungefähr $n = 2\,500\,000$ stabil bleiben. Diese Beobachtung zeigt, dass sich dieses Verfahren schlecht für eine exakte Bestimmung von π eignet, da die Konvergenz gegen den wahren Wert sehr langsam ist. Diese Tatsache ist ein erster Indiz dafür, warum wir diese Monte-Carlo-Integrationsmethode als naiv bezeichnen. Anzumerken ist, dass aufgrund der Einfachheit der Methode, die Berechnung mit MATHEMATICA trotz großem Stichprobenumfang sehr schnell erfolgt und deshalb simultan beobachtet werden kann. ∆

3.2 Mehrdimensionale naive Monte-Carlo-Integration

Eine Stärke der Monte-Carlo-Integration ist ihre Einfachheit. Zum einen kann sie sehr einfach und effizient implementiert werden, da nur einfache Rechenschritte verwendet werden, und zum anderen kann die Methode sehr einfach für mehrdimensionale Integrale verallgemeinert werden, wie die nachfolgende Festlegung zeigt:

Algorithmus 3.3 (Mehrdimensionale naive Monte-Carlo-Integration)
Seien $m \in \mathbb{N}$, $g : \mathbb{R}^m \to \mathbb{R}$ eine integrierbare[11] Funktion und $A = [a_1, b_1] \times \ldots \times [a_m, b_m] \subset \mathbb{R}^m$ mit $a_j, b_j \in \mathbb{R}$ und $a_j < b_j$ für $1 \leq j \leq m$. Um eine Näherung für den Wert des Integrals

$$I = \int_A g(x_1, \ldots, x_m)\, d(x_1, \ldots, x_m)$$

zu ermitteln, sind folgende Schritte durchzuführen:

(1) Erzeuge n auf A gleichverteilte zufällige Vektoren

$$\begin{pmatrix} x_{11} \\ \vdots \\ x_{m1} \end{pmatrix}, \ldots, \begin{pmatrix} x_{1n} \\ \vdots \\ x_{mn} \end{pmatrix}.$$

(2) Bestimme die zugehörigen Funktionswerte $g(x_{11}, \ldots, x_{m1}), \ldots, g(x_{1n}, \ldots, x_{mn})$.

(3) Approximiere I durch den Schätzer

$$I_n = \frac{\prod_{j=1}^{m}(b_j - a_j)}{n} \sum_{i=1}^{n} g(x_{1i}, \ldots, x_{mi}).$$

[11] Wie schon bei Algorithmus 3.1 angedeutet wurde, verweise ich auf Beispiel 3.16.

Bemerkung 3.4

Das Integrationsgebiet A muss keinen Quader darstellen. Prinzipiell könnte A auch eine beliebige (beschränkte) Teilmenge von $\mathbb{R}^m$ darstellen, sofern es möglich ist, gleichverteilte zufällige Vektoren auf A zu erzeugen und $\lambda^m(A)$ zu bestimmen. In Abschnitt 3.3 werden wir eine Methode kennenlernen, die dies leisten kann.

Beispiel 3.5 (Zweidimensionale naive Monte-Carlo-Integration)

Wir betrachten eine Funktion $g : \mathbb{R}^2 \to \mathbb{R}$ definiert durch $g(x,y) := 1/(x+y)^2$ und die Menge $A = [0,1] \times [1,2] \subset \mathbb{R}^2$. Gesucht wird der Wert des Integrals

$$I = \int_A g(x,y)\, d(x,y).$$

Zunächst legen wir den Stichprobenumfang n durch

```
n = 10 000;
```

fest und definieren die Funktion g durch

```
g[x_,y_] = 1/(x+y)^2;.
```

Als Zufallszahlgenerator dient uns nun der Mersenne-Twister bei einem Startwert von Eins:

```
SeedRandom[1,Method → "MersenneTwister"];.
```

Für die Erzeugung einer Liste von n zufälligen Vektoren auf A benutzen wir den Befehl

```
Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];.
```

Weil $\lambda^2(A) = (1-0)\cdot(2-1) = 1$ ist, ergibt sich eine Realisierung des Schätzers I_n unmittelbar durch die Eingabe von

```
Mean[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]].
```

Dabei haben wir benutzt, dass durch `Liste[[i,1]]` die x-Komponente des i-ten zufälligen Vektors und durch `Liste[[i,2]]` die y-Komponente des i-ten zufälligen Vektors $(1 \le i \le n)$ für die Berechnung des zugehörigen Funktionswerts herangezogen werden. Der Befehl `Table` erzeugt somit $g(x_1,y_1),\ldots,g(x_n,y_n)$, also die Liste der Funktionswerte. Mit `Mean` ergibt sich dann das arithmetische Mittel dieser Funktionswerte.

Da g auf A stetig ist, kann der Satz von Fubini [21, Nr. 200] angewendet werden. Eine einfache Rechnung zeigt dann, dass $I = \ln(4/3) = 0{,}287682\ldots$ ist. Weil wir den genauen Wert von I kennen, können wir nun einen Einblick in das Verhalten der Monte-Carlo-Integration gewinnen. Dazu beobachten wir die absolute Abweichung des Schätzers I_n vom exakten Wert für wachsenden Stichprobenumfang n. Zusätzlich benutzen wir den Befehl `AbsoluteTiming`, mit dem wir messen können, wie lange die Berechnung von I_n (in Realzeit) dauert. Wir starten zunächst mit $n = 10\,000$ und verdoppeln dann fortlaufend den Stichprobenumfang n. Tabelle 3.1 zeigt die Ergebnisse dieser Simulation.

Betrachtet man die absolute Abweichung $|I - I_n|$, so wird die stochastische Natur der Monte-Carlo-Integration deutlich. Denn im Gegensatz zu deterministischen Verfahren, führt eine Vergrößerung von n nicht zwangsläufig zu einem exakteren Wert (siehe auch Abbildung 2.1). Die bereits angesprochene langsame Konvergenz der Monte-Carlo-Integration ist ebenfalls in Tabelle 3.1 deutlich erkennbar. Mit diesem Sachverhalt werden wir uns später noch genauer auseinandersetzen.

n	I_n	$\lvert I - I_n \rvert$	Rechenzeit [s]
10 000	0,288375 …	$6{,}9 \cdot 10^{-4}$	0,04
20 000	0,289276 …	$1{,}5 \cdot 10^{-3}$	0,09
40 000	0,288623 …	$9{,}4 \cdot 10^{-4}$	0,17
80 000	0,287922 …	$2{,}4 \cdot 10^{-4}$	0,34
160 000	0,288265 …	$5{,}8 \cdot 10^{-4}$	0,68
320 000	0,288093 …	$4{,}1 \cdot 10^{-4}$	1,40
640 000	0,287923 …	$2{,}4 \cdot 10^{-4}$	2,77
1 280 000	0,287853 …	$1{,}7 \cdot 10^{-4}$	5,47
2 560 000	0,287820 …	$1{,}3 \cdot 10^{-4}$	11,17
5 120 000	0,287782 …	$1{,}0 \cdot 10^{-4}$	22,96
10 240 000	0,287735 …	$5{,}4 \cdot 10^{-5}$	45,15
20 480 000	0,287716 …	$3{,}4 \cdot 10^{-5}$	90,81
40 960 000	0,287711 …	$2{,}8 \cdot 10^{-5}$	184,72
81 920 000	0,287677 …	$4{,}7 \cdot 10^{-6}$	370,69

Tabelle 3.1 Ergebnisse der Simulation (naive MCI)

Von besonderem Interesse ist natürlich die benötigte Rechenzeit. Zwischen der benötigten Rechenzeit und dem Stichprobenumfang n herrscht in guter Näherung ein direkt proportionaler Zusammenhang (Abbildung 3.2).

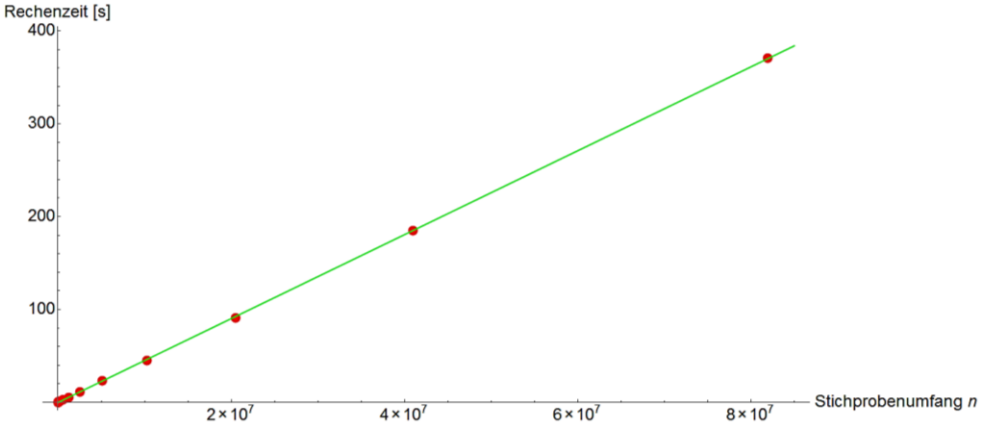

Abbildung 3.2 Direkt proportionaler Zusammenhang zwischen Stichprobenumfang und Rechenzeit

Anzumerken ist, dass die hier angegeben Werte mit einem (etwas älteren) Notebook[12] ermittelt wurden. Theoretisch würde dieses Verfahren enden, wenn die maximale Periodenlänge des Mersenne-Twisters (siehe Abschnitt 2.2) erreicht wurde. Da der Arbeitsspeicher herkömmlicher Rechner für derartig große Datenmengen (derzeit) nicht ausreichend ist, wird man diese Grenze jedoch (bei Weitem) nicht erreichen. Mit einem schrittweisen Vorgehen (Beispiel 3.14) kann die Belastung des Arbeitsspeichers jedoch begrenzt werden. Dass es aufgrund des enormen Rechenaufwands trotzdem nicht möglich ist, diese Grenze zu erreichen zeigt die folgende Überlegung: Angenommen wir würden pro Schritt 81 920 000 Zufallszahlen benutzen und die Arbeitsspeicherbelastung könnte stabil gehalten werden, so dass die Rechenzeit pro Schritt ungefähr 370 Sekunden (Tabelle 3.1) beträgt. Bis eine volle Periode des Mersenne-Twisters ausgeschöpft wurde, würden rund $6 \cdot 10^{5988}$ Jahre vergehen. Der aktuell schnellste Supercomputer heißt „Sunway TaihuLight“ und besitzt 10 649 600 Prozessorkerne [35]. Angenommen wir könnten die schrittweise Berechnung (ohne Zeitverlust und Arbeitsspeicherproblemen) auf alle Kerne dieses Supercomputers aufteilen und jeder Schritt würde s Sekunden benötigen, so würde die Rechenzeit bis zum Ausschöpfen der vollen Periode dennoch rund $s \cdot 10^{5979}$ Jahre betragen. ∆

[12] CPU: Intel Core i5-480M, RAM: 8GB, Windows 7: 64 Bit

3.3 Hit-or-miss Monte-Carlo-Integration

In diesem Abschnitt beschäftigen wir uns mit der hit-or-miss Monte-Carlo-Integration, die es uns ermöglichen wird, sehr einfach das Volumen von beschränkten Mengen auf $\mathbb{R}^m$ (näherungsweise) zu bestimmen. Obwohl diese Methode nur einen Spezialfall der naiven mehrdimensionalen Monte-Carlo-Integration darstellt, können wir sie unabhängig davon einführen. Wir beschränken uns bei der Herleitung der Anschaulichkeit wegen auf den zweidimensionalen Fall.

Sei $A \subset \mathbb{R}^2$ eine beschränkte Menge. Unser Ziel ist es nun, das Volumen $\lambda^2(A)$ zu ermitteln. Dazu wählen wir $B = [a_1, b_1] \times [a_2, b_2] \subset \mathbb{R}^2$ so, dass $A \subseteq B$ gilt. Dies ist aufgrund der vorausgesetzten Beschränktheit von A stets möglich. Anschaulich „grenzen“ wir die Menge A durch die Menge B „ein“ (Abbildung 3.3), weshalb dann $0 \leq \lambda^2(A) \leq \lambda^2(B) < \infty$ gilt.

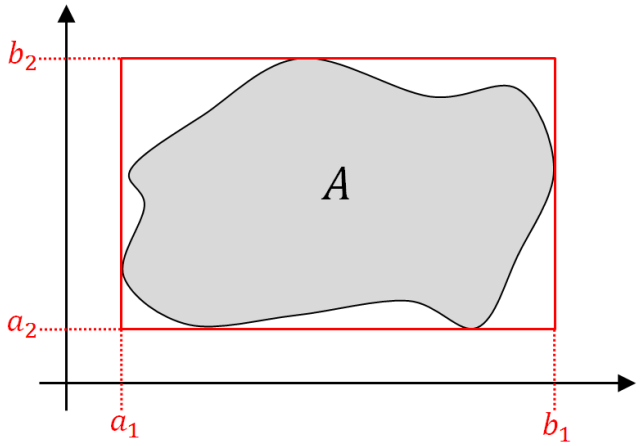

Abbildung 3.3 Eingrenzung der Menge A

Wir wählen zur Lösung dieses Problems wieder einen wahrscheinlichkeitstheoretischen Zugang. Sei X ein auf B gleichverteilter Zufallsvektor. Die Wahrscheinlichkeit, dass X in A liegt, ergibt sich dann als Quotient der beiden Volumina zu

$$\mathbb{P}(X \in A) = \frac{\lambda^2(A)}{\lambda^2(B)}.$$

Formen wir diese Gleichung um, so erhalten wir für das gesuchte Volumen den Zusammenhang

$$\lambda^2(A) = \lambda^2(B) \cdot \mathbb{P}(X \in A).$$

Da $\lambda^2(B)$ leicht zu berechnen ist, müssen wir nur einen Weg finden, um $\mathbb{P}(X \in A)$ zu bestimmen. Dazu betrachten wir eine Folge $X_1, \dots, X_n$ von unabhängigen und auf B gleichverteilten Zufallsvektoren. Die Indikatorfunktion

$$\mathbf{1}\{X_i \in A\} = \begin{cases} 1 & \text{falls } X_i \in A, \\ 0 & \text{falls } X_i \notin A \end{cases}$$

für $1 \leq i \leq n$ stellt fest, ob $X_i \in A$ gilt und damit ein „Treffer“ (hit) vorliegt. Die Zufallsvariable

$$R_n := \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{X_i \in A\}$$

beschreibt dann die relative Trefferhäufigkeit (bei n Versuchen). Wegen Satz 1.31(c) folgt mit dem starken Gesetz großer Zahlen schließlich, dass

$$R_n \xrightarrow{\text{f.s.}} \mathbb{P}(X \in A).$$

Insgesamt erhalten wir damit die Näherung

$$\lambda^2(A) \simeq \frac{(b_2 - a_2)(b_1 - a_1)}{n} \cdot \sum_{i=1}^{n} \mathbf{1}\{X_i \in A\}.$$

Diese Vorgangsweise lässt sich unmittelbar auf den m-dimensionalen Fall übertragen und führt uns direkt zur nachfolgenden Festlegung:

Algorithmus 3.6 (Hit-or-miss Monte-Carlo-Integration)
Sei A eine beschränkte Teilmenge von $\mathbb{R}^m$ für $m \geq 2$. Um eine Näherung für das Volumen $\lambda^m(A)$ zu ermitteln, sind folgende Schritte durchzuführen:

(1) Wähle $B = [a_1, b_1] \times \ldots \times [a_m, b_m] \subset \mathbb{R}^m$ so, dass $A \subseteq B$ gilt und B „möglichst klein" ist.
(2) Erzeuge n auf B gleichverteilte zufällige Vektoren $x_1, , \ldots, x_n$ und zähle, wie viele in A liegen.
(3) Approximiere $\lambda^m(A)$ durch den Schätzer

$$I_n = \frac{\prod_{j=1}^{m}(b_j - a_j)}{n} \sum_{i=1}^{n} \mathbf{1}\{x_i \in A\}.$$

Anmerkung
Das Verfahren funktioniert, solange $A \subseteq B$ erfüllt ist. Trotzdem sollte man die Menge B „möglichst klein" wählen, da sie dann schneller mit zufälligen Vektoren „ausgefüllt" werden kann.

Beispiel 3.7 (Näherungsweise Bestimmung von π - Variante 2)
Wir betrachten den Einheitskreis, also die Menge $A = \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 \leq 1\}$. Weil sein Flächeninhalt gleich π ist, können wir mithilfe der hit-or-miss Monte-Carlo-Integration eine Näherung für π berechnen.
Dazu grenzen wir den Einheitskreis mit einem Quadrat $B = [-1,1]^2$ ein. Für die Erzeugung von n auf B gleichverteilten zufälligen Vektoren nutzen wir den Mersenne-Twister mit einem Anfangswert von Eins:

```
n = 100 000;
SeedRandom[1,Method → "MersenneTwister"];
ZVektoren = RandomVariate[UniformDistribution[{{-1,1},{-1,1}}],n];.
```

Mit der Eingabe

```
Hit = Select[ZVektoren,(#[[1]])^2 + (#[[2]])^2 ≤ 1&];
```

ergibt sich eine Liste, welche alle zufälligen Vektoren, die in A liegen, enthält. Mithilfe von

```
Miss = Complement[ZVektoren,Hit];
```

wird eine Liste aller Vektoren, die nicht in A liegen, festgelegt. Wir plotten jetzt die „Treffer" (in Grün) und die „Fehlschüsse" (in Rot) durch

```
G1 = ListPlot[{Hit,Miss},PlotStyle → {Green,Red}];.
```

Zusätzlich stellen wir die Kreislinie des Einheitskreises mittels

```
G2 = Graphics[Circle[{0,0}]];
```

dar. Durch die Eingabe von

```
Show[G1,G2,AspectRatio → 1]
```

kombinieren wir die beiden Grafiken. Um einen numerischen Näherungswert für π zu erhalten, muss nur bedacht werden, dass die absolute Trefferhäufigkeit gerade der Anzahl der Elemente in der Liste `Hit` entspricht. Es gilt somit `Length[Hit]` $= \sum_{i=1}^{n} \mathbf{1}\{x_i \in A\}$.

Wegen $\lambda^2(B) = 4$ ergibt sich durch die Eingabe von

```
N[4/n Length[Hit]]
```

ein numerischer Näherungswert für π (bzw. eine Realisierung des Schätzers I_n). Das Ergebnis dieser Simulation ist in Abbildung 3.4 ersichtlich. Der errechnete Näherungswert für π ist in diesem Fall ($n = 100\,000$) gleich 3,14356.

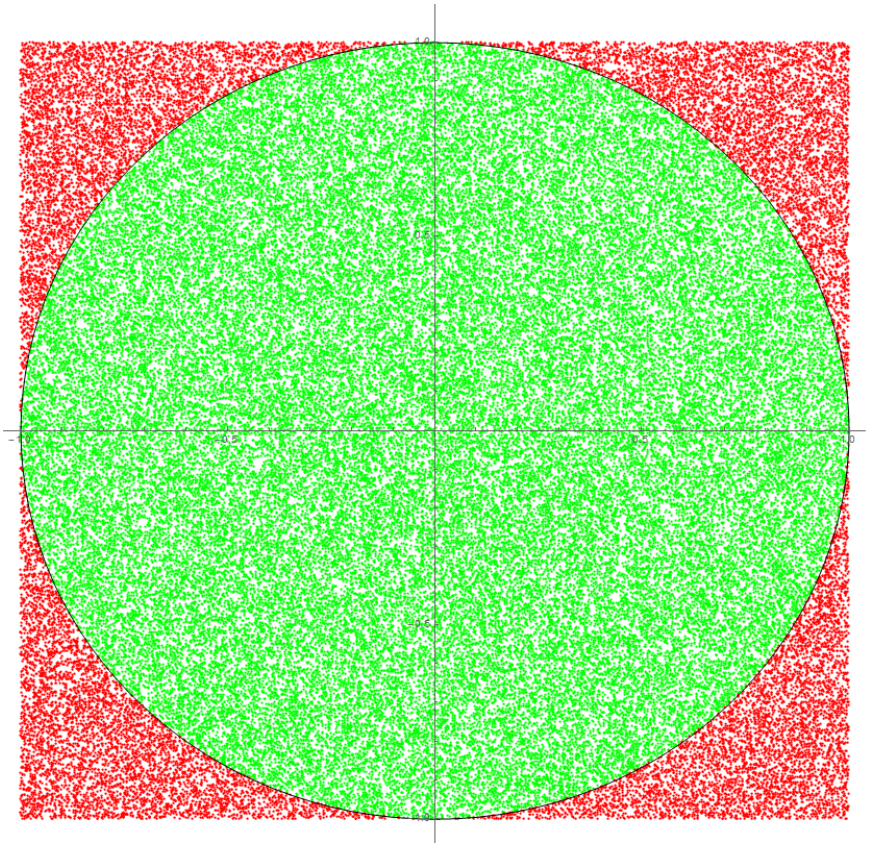

Abbildung 3.4 Näherungsweise Bestimmung von π mittels hit-or-miss Monte-Carlo-Integration

Möchte man den „Punkteregen“ genauer beobachten, so kann eine `Manipulate`-Umgebung zur Variation von n eingesetzt werden. Da die grafische Darstellung sehr rechenintensiv ist, sollte dies jedoch nur mit kleineren Stichprobenumfängen durchgeführt werden. Für umfassendere Simulationen ist es deshalb zweckmäßig die grafische Ausgabe durch die Eingabe eines Strichpunkts zu unterdrücken. Für $n = 5\,000\,000$ ergibt sich dann der Näherungswert 3,1416312 (vgl. mit Beispiel 3.2). Δ

Wir betrachten im Folgenden eine nichtnegative integrierbare Funktion $g : \mathbb{R}^m \to \mathbb{R}$. Wir wollen nun mit Hilfe der hit-or-miss Monte-Carlo-Integration eine Näherung für den Wert des Integrals

$$I = \int_A g(x_1, \dots, x_m)\, d(x_1, \dots, x_m)$$

mit $A = [a_1, b_1] \times \dots \times [a_m, b_m] \subset \mathbb{R}^m$ bestimmen. Dafür machen wir uns die Interpretation des Integrals I als Volumen „unter dem Graphen von g“ zunutze (Abbildung 3.5). Wir setzen

$$M := \max\{g(x_1, \dots, x_m) : (x_1, \dots, x_m) \in A\}$$

und wählen $B := A \times [0, M] \subset \mathbb{R}^{m+1}$. Die Menge

$$C := \{(x_1, \dots, x_m, x_{m+1}) \in B : x_{m+1} \leq g(x_1, \dots, x_m)\}$$

bezeichnet dann alle Punkte aus B, die anschaulich „unter oder auf dem Funktionsgraphen von g liegen“ (Abbildung 3.5).

Damit sind die Voraussetzungen für die Anwendung der hit-or-miss Monte-Carlo-Integration gegeben und wir erhalten

$$I = \lambda^{m+1}(C) \simeq \frac{\lambda^{m+1}(B)}{n} \sum_{i=1}^{n} \mathbf{1}\{x_i \in C\}$$

für unabhängige und auf B gleichverteilte zufällige Vektoren $x_1, \ldots, x_n$.

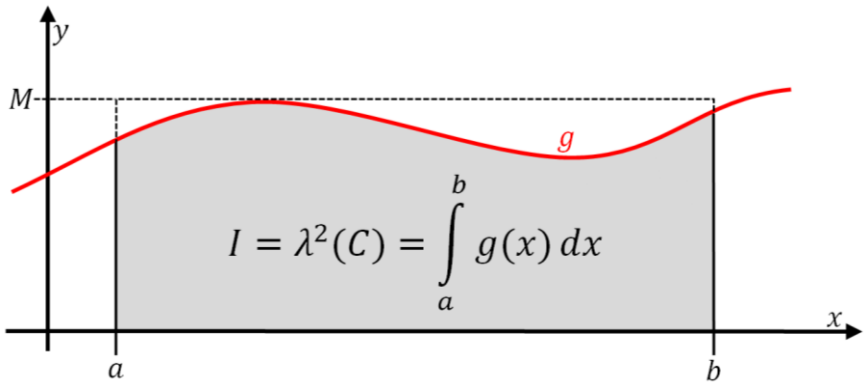

Abbildung 3.5 Veranschaulichung der Vorgangsweise für eine nichtnegative reelle Funktion

Bei der Integration einer Funktion $g : \mathbb{R}^m \to \mathbb{R}$ operiert die hit-or-miss Methode somit „eine Dimension über" der naiven Methode, was zwangsläufig einen höheren Verbrauch von Zufallszahlen zur Folge hat. Wir verdeutlichen dies nun anhand des Integrals aus Beispiel 3.5.

Beispiel 3.8

Zunächst suchen wir das Maximum von $g : \mathbb{R}^2 \to \mathbb{R}$ mit $g(x,y) := 1/(x+y)^2$ auf der Menge $A = [0,1] \times [1,2]$. Da die Quadratfunktion im Positiven streng monoton wächst, nimmt die Funktion g ihr Maximum auf A bei $(0,1)$ an. Somit gilt, dass

$$M = \max\{g(x,y) : (x,y) \in A\} = 1$$

ist. Wir setzen demnach $B := [0,1] \times [1,2] \times [0,1]$ und $C := \{(x,y,z) \in B : z \leq g(x,y)\}$. Abbildung 3.6 zeigt anschaulich die definierten Größen.

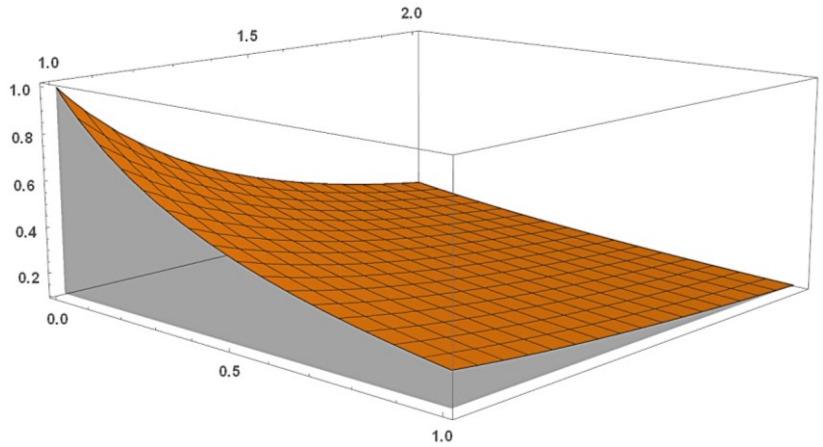

Abbildung 3.6 Darstellung des Funktionsgraphen (orange) und der Mengen C (grau) und B (Box)

Wir definieren zunächst den Stichprobenumfang n und die Funktion g durch

```
n = 10 000;
g[x_,y_] = 1/(x+y)^2;
```

und verwenden für die Erzeugung der zufälligen Vektoren auf B

```
SeedRandom[1,Method→"MersenneTwister"];
ZVektoren = RandomVariate[UniformDistribution[{{0,1},{1,2},{0,1}}],n];.
```

Mit

```
Hit = Select[ZVektoren, (#[[3]]) ≤ g[#[[1]],#[[2]]]&];
```

ergibt sich schließlich die Liste der „Treffer“. Da $\lambda^3(B) = 1$ ist, ergibt sich eine Realisierung des Schätzers I_n durch

```
N[1/n Length[Hit]].
```

Wir beobachten die hit-or-miss Monte-Carlo-Integration nun in Analogie zu Beispiel 3.5. Tabelle 3.2 zeigt die Ergebnisse der Analyse.

n	I_n	$\lvert I - I_n \rvert$	Rechenzeit [s]
10 000	0,2915	$3{,}8 \cdot 10^{-3}$	0,07
20 000	0,28545	$2{,}2 \cdot 10^{-3}$	0,18
40 000	0,290525	$2{,}8 \cdot 10^{-3}$	0,36
80 000	0,290112 …	$2{,}4 \cdot 10^{-3}$	0,49
160 000	0,28815	$4{,}7 \cdot 10^{-4}$	1,33
320 000	0,288475	$7{,}9 \cdot 10^{-4}$	2,86
640 000	0,288564 …	$8{,}8 \cdot 10^{-4}$	4,44
1 280 000	0,287914 …	$2{,}3 \cdot 10^{-4}$	8,86
2 560 000	0,288123 …	$4{,}4 \cdot 10^{-4}$	17,54
5 120 000	0,287923 …	$2{,}4 \cdot 10^{-4}$	35,19
10 240 000	0,287899 …	$2{,}1 \cdot 10^{-4}$	70,51
20 480 000	0,287844 …	$1{,}6 \cdot 10^{-4}$	140,39
40 960 000	0,287598 …	$8{,}4 \cdot 10^{-5}$	279,38
81 920 000	0,287699 …	$1{,}7 \cdot 10^{-5}$	2301,14

Tabelle 3.2 Ergebnisse der Simulation (hit-or-miss MCI)

Es ist zu erkennen, dass die Rechenzeit, wie in Beispiel 3.5, annähernd direkt proportional zum Stichprobenumfang n ist. Nur im letzten Fall ergibt sich eine Abweichung, da der Arbeitsspeicher nicht mehr ausreichte und auf den (langsameren) Festplattenspeicher zurückgegriffen werden musste. Um in diesem Fall überhaupt einen Wert zu erhalten, musste der Befehl `Share` eingesetzt werden. Mit seiner Hilfe versucht MATHEMATICA den Speicherbedarf zu reduzieren.
Wie ein Vergleich mit Tabelle 3.1 zeigt, ergibt sich eine nur etwas schlechtere Genauigkeit der Näherungswerte. Der Rechenaufwand ist jedoch größer, weil doppelt so viele Zufallszahlen verwendet werden. ∆

Zusammenfassend können wir festhalten, dass wir mit der hit-or-miss Monte-Carlo-Integration das Volumen beschränkter Teilmengen von $\mathbb{R}^n$ ermitteln können. Voraussetzung dafür ist jedoch, dass wir entscheiden können, ob ein Punkt innerhalb dieser Menge liegt oder nicht. Zudem wurde hier nur eine Eingrenzung der Menge mithilfe eines Quaders thematisiert. Man könnte die betrachtete Menge jedoch auch mit einer anderen Eingrenzung versehen, sofern das Volumen dieser Eingrenzung leicht zu berechnen ist und gleichverteilte zufällige Vektoren darauf erzeugt werden können.

Wir könnten eine beschränkte Menge $A \subset \mathbb{R}^2$ zum Beispiel mithilfe eines Kreises eingrenzen. Da sein Volumen leicht zu berechnen ist, stellt sich nur noch die Frage, wie gleichverteilte zufällige Vektoren auf dem Kreis erzeugt werden können. Eine mögliche Antwort auf diese Frage liefert ein Blick auf Abbildung 3.4: Wir begrenzen den Kreis durch ein Quadrat, erzeugen darauf gleichverteilte zufällige Vektoren und „filtern“ jene heraus, die auf dem Kreis liegen. Diese Idee des „Filterns“ von zufälligen Vektoren führt uns zu einer weiteren Technik, um beliebig verteilte Zufallszahlen zu erzeugen.

Exkurs 3.9 (Annahme- und Verwerfungsmethode)
Diese Methode der Zufallszahlerzeugung geht auf John von Neumann zurück. Sie bietet einen möglichen Ausweg in Fällen, bei denen sich die Inversionsmethode als zu aufwendig erweist, die Inverse einer Verteilungsfunktion also nicht explizit ausdrückbar ist [11]. Wir skizzieren diese Methode nun schematisch.
Das Ziel ist, eine Zufallszahl zu erzeugen, die als Realisation einer Zufallsvariable X mit Dichte f aufgefasst werden kann. Dafür betrachten wir eine Zufallsvariable X' mit einer „einfachen" Dichte h, die f „ähnelt". Die Einfachheit einer solchen Hilfsdichte h meint, dass (beispielweise mithilfe der Inversionsmethode) Realisierungen von X' erzeugt werden können. Dies ist zum Beispiel der Fall, wenn X' auf $[a, b]$ gleichverteilt ist (Beispiel 2.7) oder eine Exponentialverteilung [39, Beispiel 2.7] besitzt.
Wir wählen dann eine reelle Konstante $c \geq 1$, sodass $f(x) \leq c \cdot h(x)$ für alle $x \in \mathbb{R}$ erfüllt ist. Somit ist $c \cdot h$ eine obere Schranke von f. Zusätzlich bezeichnen wir mit

$$A = \{(x, y) \in \mathbb{R} \times \mathbb{R} : 0 \leq y \leq c \cdot h(x)\}$$

die Fläche zwischen dem Graphen von $c \cdot h$ und der x-Achse. Die wesentliche Idee der Methode besteht darin, einen auf A gleichverteilten zufälligen Vektor (x, y) zu erzeugen und dann zu überprüfen, ob $y \leq f(x)$ erfüllt ist, der zufällige Vektor also „unter dem Graphen von f" liegt (Abbildung 3.7). In einem solchen Fall ist x eine Realisierung der Zufallsvariable X mit Dichte f.

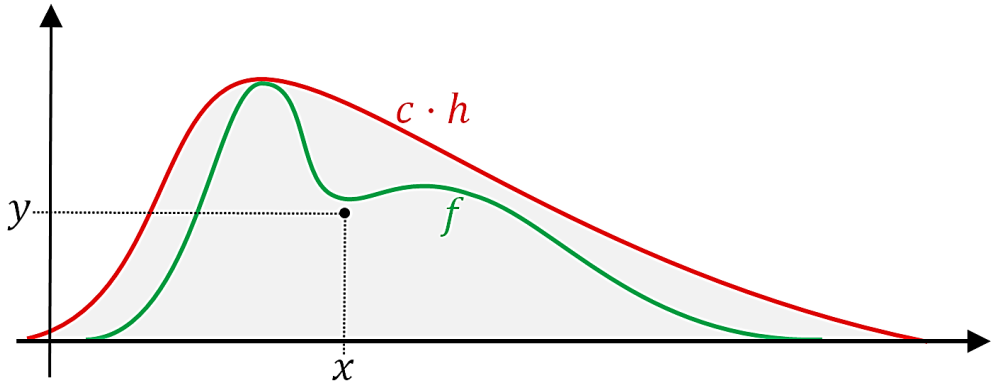

Abbildung 3.7 Schematische Darstellung der Eingrenzung von f durch $c \cdot h$ und der Menge A (grau)

Zu erwähnen ist, dass x anfangs gemäß der Hilfsdichte h erzeugt wird, also stets eine Realisierung von X' darstellt. Zusammenfassend kann die Methode durch folgende Schritte beschrieben werden:

(1) Wähle eine Hilfsdichte h (für die ein Verfahren zur Zufallszahlerzeugung bekannt ist) und suche $c \geq 1$ möglichst klein, sodass $c \cdot h$ eine obere Schranke von f darstellt.
(2) Erzeuge eine Zufallszahl x, die eine Realisierung der Zufallsvariable X' mit Hilfsdichte h darstellt.
(3) Erzeuge eine Standardzufallszahl u und setze $y = u \cdot c \cdot g(x)$.
(4) Gilt $y \leq f(x)$, dann ist x eine Realisierung der Zufallsvariable X mit Dichte f. Andernfalls wird x verworfen und das Verfahren startet ausgehend von (2) von Neuem.

Dieses Verfahren kann in völliger Analogie für einen (n-dimensionalen) Zufallsvektor X mit Dichte f formuliert werden [29, 4.3]. Man kann außerdem zeigen, dass mit diesem Verfahren eine Gleichverteilung auf A [25, Satz 9.4] simuliert wird, und dass die auf diese Weise erhaltenen Zufallszahlen tatsächlich Realisierungen einer Zufallsvariable X mit Dichte f darstellen [25, Satz 9.5].

Die Effizienz des Verfahrens hängt im Wesentlichen davon ab, wie nahe die obere Schranke $c \cdot h$ bei f liegt. Anschaulich ist klar, dass umso weniger Zufallszahlen verworfen werden, je näher die obere Schranke bei f liegt, also je kleiner die Fläche zwischen $c \cdot h$ und f ist (Abbildung 3.7). Man kann zudem zeigen, dass die Wahrscheinlichkeit, dass x angenommen wird, gerade $1/c$ beträgt und im Mittel c zufällige Vektoren erzeugt werden müssen, bis x angenommen wird [25, Satz 9.6]. Darin liegt gleichzeitig die Begründung dafür, dass c möglichst klein gewählt werden sollte (bzw. die Hilfsdichte h der Dichte f „ähneln“ sollte). Aufgrund von $f(x) \leq c \cdot h(x)$ wählt man idealerweise $c \coloneqq \sup\{f(x)/h(x) : x \in \mathbb{R}\}$ [25, (9.8)].

Bemerkung 3.10
Mit Rückblick auf Bemerkung 3.4 können wir festhalten, dass mit der hit-or-miss Monte-Carlo-Integration das Volumen $\lambda^m(A)$ eines (beschränkten) Integrationsgebiets A näherungsweise ermittelt werden kann. Die Annahme- und Verwerfungsmethode bietet zudem eine gute Möglichkeit, um auf A gleichverteilte zufällige Vektoren zu erzeugen. Die naive mehrdimensionale Monte-Carlo-Integration ist somit direkt auf komplizierteren Integrationsgebieten $A \subset \mathbb{R}^m$ einsetzbar.

3.4 Die Genauigkeit der vorgestellten Verfahren

Bisher haben wir ausschließlich Beispiele betrachtet, bei denen der exakte Wert bekannt war. Aus diesem Grund konnten wir den absoluten Fehler $|I - I_n|$ stets unmittelbar angeben (Beispiel 3.5 und 3.8). Bei realen Anwendungen ist der Wert von I jedoch unbekannt, den sonst würde man sich nicht für ein solches Näherungsverfahren entscheiden. In diesem Abschnitt wird nun gezeigt, wie groß der Fehler des Schätzers I_n ist. Besonders hervorzuheben ist die Tatsache, dass dieser Fehler von stochastischer Natur ist. Es können also keine strikten Fehlerschranken (wie bei deterministischen Verfahren) angegeben werden.

Aufgrund der Erkenntnis aus Bemerkung 3.10 formulieren wir noch einmal die Voraussetzungen und rekapitulieren das bisher Bekannte, bevor wir uns mit dem Fehler der Monte-Carlo-Integration beschäftigen.

3.4.1 Kurze Zusammenfassung und Definition des Fehlers

Es sei $m \geq 1$. Wir betrachten eine beschränkte Teilmenge A von $\mathbb{R}^m$ und eine integrierbare Funktion $g : \mathbb{R}^m \to \mathbb{R}$. Zudem seien $X_1, \dots, X_n$ unabhängige auf A gleichverteilte Zufallsvektoren. Um einen Näherungswert für das Integral

$$I = \int_A g(x_1, \dots, x_m)\, d(x_1, \dots, x_m)$$

zu bestimmen, verwenden wir den (naiven) Monte-Carlo-Schätzer[13]

$$I_n = \frac{\lambda^m(A)}{n} \sum_{i=1}^{n} g(X_i). \tag{3.2}$$

[13] In den Algorithmen 3.1, 3.3, 3.6 und den zugehörigen Beispielen wurde bereits der Begriff „Schätzer“ verwendet. Exakter müsste man dort stets von einer gewissen Realisierung des Schätzers I_n (Zufallsvariable) sprechen, da sich die Ausführungen auf konkrete Stichproben bezogen. Um die Notation nicht unnötig zu verkomplizieren, wird dies aber so weitergeführt.

Wir wissen außerdem bereits, dass I_n fast sicher gegen I konvergiert. Man sagt deshalb auch, dass I_n einen stark konsistenten Schätzer für I darstellt.
Des Weiteren ist I_n ein erwartungstreuer Schätzer für I, da (wegen Definition 1.21, Satz 1.31, und Bemerkung 1.32)

$$\begin{aligned}\mathbb{E}(I_n) &= \mathbb{E}\left(\frac{\lambda^m(A)}{n}\sum_{i=1}^{n} g(X_i)\right) = \frac{\lambda^m(A)}{n}\,\mathbb{E}\left(\sum_{i=1}^{n} g(X_i)\right) = \frac{\lambda^m(A)}{n}\; n\mathbb{E}\big(g(X_1)\big) \\ &= \lambda^m(A)\int_{\mathbb{R}^m} g(x_1,\dots,x_m)\,\frac{1_A(x_1,\dots,x_m)}{\lambda^m(A)}\,d(x_1,\dots,x_m) \qquad (3.3)\\ &= \frac{\lambda^m(A)}{\lambda^m(A)}\int_A g(x_1,\dots,x_m)\,d(x_1,\dots,x_m) = I\end{aligned}$$

gilt, was aufgrund der Herleitung des Verfahrens (Abschnitt 3.1) klar ist. Die Erwartungstreue von I_n meint, dass der Schätzer richtig justiert ist, also im Mittel den richtigen Wert liefert.

Für eine übersichtlichere Notation, setzen wir $Y_i \coloneqq g(X_i)$ und $\overline{Y}_n \coloneqq 1/n \sum_i Y_i$, wobei wir bei auftretenden Summen nur den Laufindex anführen und auf die Grenzen verzichten, da ohnehin stets $1 \le i \le n$ gilt.

Um nun Aussagen über den Fehler der Methode treffen zu können, müssen wir voraussetzen, dass $\mathbb{V}(Y_1) =: \sigma^2$ existiert (bzw. g quadratisch integrierbar ist). Wir definieren den Fehler der Monte-Carlo-Integration durch

$$\varepsilon \coloneqq I - I_n.$$

Dann beschreibt $\mathbb{E}(\varepsilon^2)$ den mittleren quadratischen Fehler (kurz MSE). Aufgrund von Gleichung (3.3) entspricht der mittlere quadratische Fehler gerade der Varianz des Schätzers I_n, es gilt also

$$\mathbb{E}(\varepsilon^2) = \mathbb{E}((I - I_n)^2) = \mathbb{V}(I_n).$$

Mithilfe der Rechenregeln für die Varianz (Satz 1.36 und 1.37) und aufgrund der identischen Verteiltheit von $Y_1, \dots, Y_n$ folgt

$$\mathbb{V}(I_n) = \mathbb{V}\left(\frac{\lambda^m(A)}{n}\sum_i Y_i\right) = \frac{\lambda^m(A)^2}{n^2}\,\mathbb{V}\left(\sum_i Y_i\right) = \frac{\lambda^m(A)^2}{n^2}\; n\sigma^2 = \frac{\lambda^m(A)^2\,\sigma^2}{n}. \qquad (3.4)$$

Wenn in realen Anwendungssituationen, wie zu Beginn dieses Abschnitts erwähnt, der Erwartungswert $\mathbb{E}(Y_1) = I/\lambda^m(A)$ unbekannt ist, dann ist erst recht die Varianz σ^2 unbekannt. Folglich müssen wir auch hier einen Weg finden, um diese (zumindest in gute Näherung) zu bestimmen und damit Aussagen über die Genauigkeit des Verfahrens treffen zu können.

3.4.2 Schätzung der Varianz

Für die Schätzung von σ^2 verwenden wir die Stichprobenvarianz, die durch

$$S_n^2 \coloneqq \frac{1}{n-1}\sum_i \big(Y_i - \overline{Y}_n\big)^2 \qquad (3.5)$$

definiert wird. Die Stichprobenstandardabweichung S_n definieren wir durch $S_n \coloneqq \sqrt{S_n^2}$.

Wir leiten nun eine Identität ab, die es uns erleichtern wird, Aussagen über die Eigenschaften der Stichprobenvarianz zu treffen. Dabei addieren wir „geschickt Null“, fügen also eine Konstante $c \in \mathbb{R}$ ein. Es gilt:

$$\begin{aligned}(n-1)S_n^2 &= \sum_i \big(Y_i - \overline{Y}_n\big)^2 = \sum_i \Big((Y_i - c) + \big(c - \overline{Y}_n\big)\Big)^2 \\ &= \sum_i (Y_i - c)^2 + 2\big(c - \overline{Y}_n\big)\sum_i (Y_i - c) + n\big(\overline{Y}_n - c\big)^2 \\ &= \sum_i (Y_i - c)^2 + 2\big(c - \overline{Y}_n\big)\big(n\overline{Y}_n - nc\big) + n\big(\overline{Y}_n - c\big)^2 \\ &= \sum_i (Y_i - c)^2 - 2n\big(\overline{Y}_n - c\big)^2 + n\big(\overline{Y}_n - c\big)^2 = \sum_i (Y_i - c)^2 - n\big(\overline{Y}_n - c\big)^2.\end{aligned} \tag{3.6}$$

Satz 3.11

Die Stichprobenvarianz S_n^2 ist ein erwartungstreuer Schätzer für σ^2, das heißt $\mathbb{E}(S_n^2) = \sigma^2$.

Beweis:

Wir benutzen die Identität (3.6), wobei wir $c = \mathbb{E}(Y_1) =: \mu$ setzen. Wegen (3.3) und (3.4) wissen wir, dass $\mathbb{E}\big(\overline{Y}_n\big) = \mu$ und $\mathbb{V}\big(\overline{Y}_n\big) = 1/n \cdot \sigma^2$ gelten. Erwartungswertbildung in (3.6) liefert schließlich

$$\begin{aligned}(n-1)\mathbb{E}(S_n^2) &= \mathbb{E}\left(\sum_i (Y_i - \mu)^2 - n(\overline{Y}_n - \mu)^2\right) \\ &= n \cdot \mathbb{E}((Y_1 - \mu)^2) - n \cdot \mathbb{E}\big((\overline{Y}_n - \mu)^2\big) \\ &= n\sigma^2 - n \cdot \frac{1}{n} \cdot \sigma^2 = (n-1)\sigma^2.\end{aligned}$$

Division durch $n - 1$ liefert schließlich die Behauptung. ■

Bemerkung

Der Beweis zeigt zudem, dass gerade die Division durch $n - 1$ im Gegensatz zu n zur Erwartungstreue des Schätzers S_n^2 führt.

Satz 3.12

Die Stichprobenvarianz S_n^2 ist ein stark konsistenter Schätzer für σ^2, das heißt $S_n^2 \xrightarrow{\text{f.s.}} \sigma^2$.

Beweis:

Aus (3.6) folgt unmittelbar die Darstellung

$$S_n^2 = \frac{1}{n-1}\sum_i (Y_i - \mu)^2 - \frac{n}{n-1}(\overline{Y}_n - \mu)^2.$$

Nach dem starken Gesetz großer Zahlen gilt fast sicher

$$\lim_{n\to\infty} \frac{n}{n-1}(\overline{Y}_n - \mu)^2 = 0.$$

Wendet man das starke Gesetz großer Zahlen auf die Folge der unabhängigen Zufallsvariablen $(Y_i - \mu)^2$ an, so ergibt sich, dass

$$\lim_{n\to\infty} \frac{1}{n-1}\sum_i (Y_i - \mu)^2 = \sigma^2$$

fast sicher gilt. ■

Die Stichprobenvarianz S_n^2 eignet sich also hervorragend für die näherungsweise Bestimmung der gesuchten Varianz.

Exemplarisch betrachten wir nun wieder das Integral aus Beispiel 3.5.

Beispiel 3.13 (Fortsetzung 1 von Beispiel 3.5)
Zunächst kopieren wir den Eingabe-Code aus Beispiel 3.5 in ein neues Notebook und schreiben in der letzten Zeile statt `Mean` einfach `Variance`. MATHEMATICA liefert dann nach der Auswertung unmittelbar einen Wert für die Stichprobenvarianz S_n^2.
Bei diesem Beispiel befinden wir uns in der glücklichen Situation, dass eigentlich keine Schätzung der Varianz σ^2 von Nöten wäre. Der exakte Wert von σ^2 ergibt sich durch eine einfache Rechnung zu

$$\sigma^2 = \mathbb{E}(g(x,y)^2) - \mathbb{E}\big(g(x,y)\big)^2 = \frac{11}{108} - \ln\left(\frac{4}{3}\right)^2 = 0{,}01909\ldots .$$

Somit können wir jetzt den Wert von S_n^2 direkt mit dem exakten Wert von σ^2 für wachsendes n vergleichen. Abbildung 3.8 zeigt, wie sich die Stichprobenvarianz allmählich um den exakten Wert „einpendelt“ (man beachte dabei die Skalierung der vertikalen Achse).

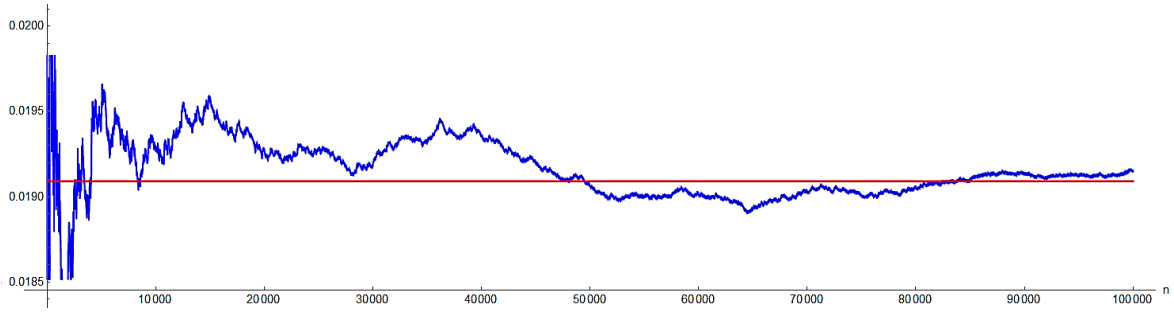

Abbildung 3.8 Annäherung von S_n^2 (blau) an den exakten Wert von σ^2 (rot) für wachsendes n

Schließlich können wir, unter Verwendung von Gleichung (3.4), die Standardabweichung des Schätzers I_n angeben. Der Kürze halber bezeichne ich die letzte Zeile der bisherigen Eingabe als `Input`. Die Standardabweichung von I_n ergibt sich dann (näherungsweise) durch

```
Sqrt[Input/n].
```

In Tabelle 3.3 sind konkrete Werte der Standardabweichung von I_n und die für deren Bestimmung benötigte Rechenzeit angeführt. Für einen direkten Vergleich, werden nochmals die im Rahmen von Beispiel 3.5 erhobenen absoluten Abweichungen angegeben.

n	$\lvert I - I_n\rvert$	$S_n/\sqrt{n}$	Rechenzeit [s]
10 000	$6{,}9 \cdot 10^{-4}$	$1{,}4 \cdot 10^{-3}$	0,06
20 000	$1{,}5 \cdot 10^{-3}$	$9{,}9 \cdot 10^{-4}$	0,10
40 000	$9{,}4 \cdot 10^{-4}$	$6{,}9 \cdot 10^{-4}$	0,18
80 000	$2{,}4 \cdot 10^{-4}$	$4{,}9 \cdot 10^{-4}$	0,37
160 000	$5{,}8 \cdot 10^{-4}$	$3{,}4 \cdot 10^{-4}$	0,71
320 000	$4{,}1 \cdot 10^{-4}$	$2{,}4 \cdot 10^{-4}$	1,53
640 000	$2{,}4 \cdot 10^{-4}$	$1{,}7 \cdot 10^{-4}$	3,24
1 280 000	$1{,}7 \cdot 10^{-4}$	$1{,}2 \cdot 10^{-4}$	6,34
2 560 000	$1{,}3 \cdot 10^{-4}$	$8{,}6 \cdot 10^{-5}$	12,81
5 120 000	$1{,}0 \cdot 10^{-4}$	$6{,}1 \cdot 10^{-5}$	25,64
10 240 000	$5{,}4 \cdot 10^{-5}$	$4{,}3 \cdot 10^{-5}$	50,99
20 480 000	$3{,}4 \cdot 10^{-5}$	$3{,}1 \cdot 10^{-5}$	103,06
40 960 000	$2{,}8 \cdot 10^{-5}$	$2{,}2 \cdot 10^{-5}$	207,00
81 920 000	$4{,}7 \cdot 10^{-6}$	$1{,}5 \cdot 10^{-5}$	386,13

Tabelle 3.3 Stichprobenstandardabweichung des Schätzers I_n und benötigte Rechenzeit

Die bereits angesprochene langsame Konvergenz der Monte-Carlo-Integration ist in Tabelle 3.3 deutlich erkennbar: Um die Standardabweichung zu halbieren, muss der Stichprobenumfang vervierfacht werden (vgl. mit (3.4)). Nicht nur die langsame Konvergenz, sondern auch die am Beginn dieses Abschnitts erwähnte stochastische Natur des Fehlers ist ersichtlich: Die absolute Abweichung war zunächst größer als die Standardabweichung, schlussendlich aber wieder kleiner. ∆

3.4.3 Konfidenzintervalle

Wir versuchen nun den Schätzfehler weiter einzugrenzen. Dafür benutzen wir den Zentralen Grenzwertsatz, mit dem unmittelbar folgt, dass der standardisierte Schätzer I_n^* annähernd standardnormalverteilt (vgl. mit Bemerkung 1.47). Zu erwähnen ist, dass durch die Standardisierung der Faktor $\lambda^m(A)$ wegfällt, weshalb wir zunächst einfach $\overline{Y}_n^*$ statt I_n^* verwenden.

Sei Φ^{-1} die Umkehrfunktion der Verteilungsfunktion Φ der Standardnormalverteilung. Wir definieren

$$z_{\alpha/2} \coloneqq \Phi^{-1}\left(\frac{\alpha}{2}\right)$$

als das $\alpha/2$-Quantil der Standardnormalverteilung, wobei $\alpha \in (0,1)$.

Aufgrund der Symmetrie der Standardnormalverteilungsdichte, also wegen $\varphi(x) = \varphi(-x)$, gilt $\Phi(x) = 1 - \Phi(-x)$ für alle $x \in \mathbb{R}$. Damit gilt der Zusammenhang

$$z_{\alpha/2} = \Phi^{-1}\left(\frac{\alpha}{2}\right) = -\Phi^{-1}\left(1 - \frac{\alpha}{2}\right) = -z_{1-\alpha/2}.$$

Nach dem Zentralen Grenzwertsatz gilt dann

$$\mathbb{P}\left(-z_{1-\alpha/2} \leq \frac{\overline{Y}_n - \mu}{\sigma/\sqrt{n}} \leq z_{1-\alpha/2}\right) \backsimeq \Phi\left(z_{1-\alpha/2}\right) - \Phi\left(-z_{1-\alpha/2}\right) = 1 - \alpha.$$

Abbildung 3.9 veranschaulicht diese Beziehung.

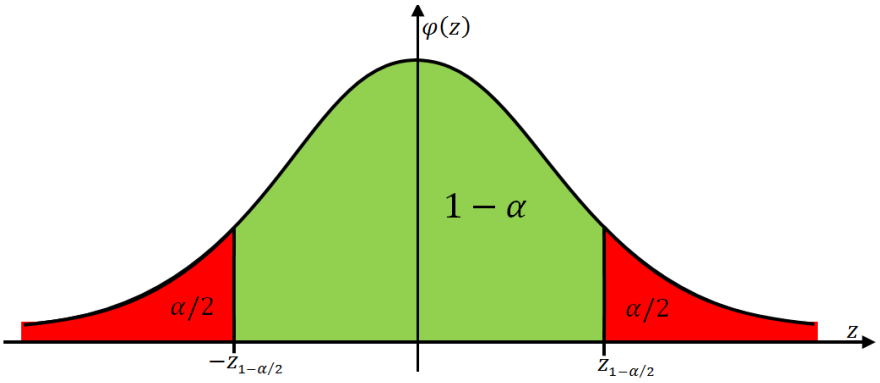

Abbildung 3.9 Darstellung eines symmetrischen Streubereichs der Standardnormalverteilung

Lösen wir nun die obige Ungleichungskette im Argument von $\mathbb{P}$ nach μ auf so ergibt sich

$$\overline{Y}_n - z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}} \leq \mu \leq \overline{Y}_n + z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}.$$

Aufgrund von $\mu = \mathbb{E}(Y_1) = 1/\lambda^m(A) \cdot I$ und $I_n = \lambda^m(A) \cdot \overline{Y}_n$ ergibt sich insgesamt

$$I_n - z_{1-\alpha/2}\frac{\lambda^m(A)\cdot\sigma}{\sqrt{n}} \leq I \leq I_n + z_{1-\alpha/2}\frac{\lambda^m(A)\cdot\sigma}{\sqrt{n}}.$$

Wir bezeichnen daher das Intervall

$$\left[I_n - z_{1-\alpha/2}\frac{\lambda^m(A)\cdot\sigma}{\sqrt{n}}, I_n + z_{1-\alpha/2}\frac{\lambda^m(A)\cdot\sigma}{\sqrt{n}}\right] \tag{3.7}$$

als Konfidenzintervall für I zum Konfidenzniveau $1-\alpha$.

Nun widmen wir uns der Interpretation von (3.7). Wählen wir hierzu $\alpha = 0{,}01$. Das bei einer konkreten Simulation (also einer Realisierung von I_n) gemäß (3.7) berechnete Intervall enthält in (ungefähr) 99 von 100 Fällen den exakten Wert des gesuchten Integrals [39]. Im Speziellen können wir uns also vorstellen, dass wir eine (einzelne) Simulation 100-mal wiederholen, wobei wir stets einen anderen Startwert (Seed) für die Zufallszahlerzeugung verwenden und jeweils das Konfidenzintervall bestimmen. Eines dieser 100 Intervalle enthält dann nicht den exakten Wert von I. Ob man im konkreten Einzelfall gerade dieses Intervall erwischt hat, weiß man jedoch nicht. Deshalb wird α auch Irrtumswahrscheinlichkeit genannt. Gleichzeitig liegt hierin die Begründung, weshalb wir uns nicht auf eine konkrete Simulation verlassen sollten, sondern mehrere Simulationen (mit unterschiedlichen Startwerten) durchführen sollten, aber dazu später mehr.

Mit (3.7) können wir nun unmittelbar ableiten, wie viele zufällige Vektoren erzeugt werden müssen, damit das berechnete Konfidenzintervall eine vorgegebene Länge ε nicht überschreitet. Die Länge des Konfidenzintervalls beträgt

$$2z_{1-\alpha/2}\,\lambda^m(A)\cdot\sigma/\sqrt{n}$$

und somit müssen

$$n \geq \left(2z_{1-\alpha/2}\,\lambda^m(A)\cdot\sigma/\varepsilon\right)^2 \tag{3.8}$$

zufällige Vektoren verwendet werden, damit die Länge des Konfidenzintervalls nicht größer als ε ist.

Da, wie bereits erwähnt, die Standardabweichung σ in realen Anwendungssituationen unbekannt ist, muss sie durch die Stichprobenstandardabweichung S_n geschätzt werden. Wegen Satz 3.12 könnten wir σ in (3.7) mit gutem Gewissen durch S_n ersetzen, sofern n genügend groß ist (für einen Beweis siehe [25, Satz 16.3b]).

Problematisch ist jedoch, dass die Länge des Konfidenzintervalls mit S_n vom Simulationsverlauf abhängig wäre. Das heißt, dass das Intervall eine vorgegebene Länge ε auch wieder überschreiten könnte. Ein Ausweg aus diesem Dilemma, ergibt sich, wenn man σ vor und unabhängig von der eigentlichen Simulation schätzt. Die Genauigkeit des Endergebnisses hängt dann davon ab, wie exakt diese Schätzung von σ war [39].

Bevor wir uns einem Beispiel zuwenden, betrachten wir den letzten verbleibenden „Baustein“ von (3.7), nämlich $z_{1-\alpha/2}$. Der Wert von $z_{1-\alpha/2}$ könnte beispielweise aus einer Tabelle [16, S. 383] abgelesen werden. Mit MATHEMATICA können wir aber exaktere Werte für $z_{1-\alpha/2}$ ermitteln. Dafür betrachten wir die Gauß'sche Fehlerfunktion, die durch

$$\operatorname{erf}(z) := \frac{2}{\sqrt{\pi}}\int_0^z e^{-s^2}\,ds \text{ für } z \geq 0$$

definiert ist [32].

Damit ergibt sich die Verteilungsfunktion der Standardnormalverteilung für $z \geq 0$ zu[14]

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{t^2}{2}} dt = \frac{1}{2} + \frac{1}{\sqrt{2\pi}} \int_{0}^{z} e^{-\frac{t^2}{2}} dt = \frac{1}{2} + \frac{1}{\sqrt{\pi}} \int_{0}^{z/\sqrt{2}} e^{-s^2} ds = \frac{1}{2} + \frac{1}{2}\operatorname{erf}\left(\frac{z}{\sqrt{2}}\right).$$

Somit folgt wegen

$$\Phi(z_{1-\alpha/2}) = \frac{1}{2} + \frac{1}{2}\operatorname{erf}\left(\frac{z_{1-\alpha/2}}{\sqrt{2}}\right) = 1 - \frac{\alpha}{2}$$

der Zusammenhang

$$z_{1-\alpha/2} = \sqrt{2}\ \operatorname{erf}^{-1}(1-\alpha). \tag{3.9}$$

Die Umkehrfunktion der Gauß'schen Fehlerfunktion ist in MATHEMATICA implementiert und kann mit dem Befehl `InverseErf` verwendet werden. Alternativ könnten wir $z_{1-\alpha/2}$ auch durch numerisches Lösen von $\Phi(z_{1-\alpha/2}) = 1 - \alpha/2$ mithilfe des Befehls `NSolve` gewinnen. Gleichung (3.9) ist dieser Methode jedoch aufgrund der übersichtlicheren Eingabe und der eingesparten Rechenzeit (benutze `AbsoluteTiming` für den Vergleich) überlegen. Zusätzlich besteht die Möglichkeit $z_{1-\alpha/2}$ durch den Befehl `Quantile[NormalDistribution[],1-α/2]` zu erhalten.

Beispiel 3.14 (Fortsetzung 2 von Beispiel 3.5)
Wir betrachten erneut das Integral aus Beispiel 3.5 und wollen nun Konfidenzintervalle gemäß (3.7) ermitteln. Eine Schätzung von σ wurde bereits im Rahmen von Beispiel 3.13 (mit Startwert Eins) durchgeführt. In Tabelle 3.3 sind jedoch nur gerundete Werte von $S_n/\sqrt{n}$ ersichtlich. Ein genauerer Wert von S_n für $n = 81\,920\,000$ ist

$$\sqrt{n} \cdot 1{,}52665 \cdot 10^{-5} = 0{,}138177.$$

Da bei diesem Beispiel σ eigentlich bekannt ist, wissen wir, dass S_n tatsächlich eine sehr gute Näherung darstellt, denn es gilt $|\sigma - S_n| \approx 7{,}3 \cdot 10^{-6}$.
Als Irrtumswahrscheinlichkeit wählen wir $\alpha = 0{,}01$. Mit Hilfe von Gleichung (3.9) ergibt sich dann $z_{0{,}995} \approx 2{,}575829$. Anzumerken ist, dass die Werte von σ und $z_{0{,}995}$ im Vorhinein bestimmt wurden, um bei der nachfolgenden Simulation Rechenzeit einzusparen.
In unserer Simulation soll das berechnete Konfidenzintervall nicht länger als $\varepsilon = 10^{-3}$ sein. Folglich müssen wir

$$n \geq (2 \cdot 2{,}575829 \cdot 0{,}138177 \cdot 1000)^2 \approx 506\,717{,}1$$

zufällige Vektoren erzeugen.
Wir setzen also

```
n = 506 718;
```

und definieren die Funktion g durch

```
g[x_,y_] = 1/(x+y)^2;.
```

Wir wollen nun die Schätzung des Integrals k-mal (mit jeweils anderem Startwert) durchführen, dabei jeweils das zugehörige Konfidenzintervall bestimmen und die berechneten Konfidenzintervalle grafisch darstellen.
Dazu definieren wir mit

```
ListeA = {};ListeB = {};
```

zwei (vorerst leere) Listen.

[14] Verwende die Normiertheit und Symmetrie von φ und substituiere mit $s = \sqrt{2}t$.

In ListeA werden später die unteren Intervallgrenzen der Konfidenzintervalle gesammelt und in ListeB die oberen. Wir legen nun die Anzahl k der Schätzungen (bzw. Konfidenzintervalle) und den Startwert j der ersten Schätzung durch

```
k = 100; j = 2;
```

fest. Wir wählen hier Zwei als Startwert, da wir den Startwert Eins für die Schätzung von σ verwendet haben. Nun benutzen wir eine While-Schleife der Form

```
While[j < k+2, Körper; j++];.
```

Mit diesem Befehl wird der Körper solange immer wieder neu ausgewertet, bis die Bedingung $j < k + 2$ nicht mehr erfüllt ist (Abbruchkriterium). Das heißt, zunächst wird $j = 2$ verwendet und weil $2 < 102$ gilt, wird der Körper mit $j = 2$ ausgewertet. Dann wird durch j++ die Zahl j um Eins erhöht und der Prozess beginnt von Neuem. Die Schleife findet dann ihr Ende, wenn $j = 102$ „an der Reihe“ wäre. Somit wird der Körper für $j \in \{2,3,\ldots,101\}$, also $k = 100$-mal, ausgewertet. In den Körper schreiben wir nun (vgl. Beispiel 3.5)

```
SeedRandom[j, Method → "MersenneTwister"];
Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];
s = Mean[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]];.
```

Zudem wollen wir die Intervallgrenzen des zugehörigen Konfidenzintervalls speichern. Dafür verwenden wir den Befehl AppendTo und schreiben

```
AppendTo[ListeA, s - 2.575829*0.138177/Sqrt[n]];
AppendTo[ListeB, s + 2.575829*0.138177/Sqrt[n]]
```

ebenfalls in den Körper. Damit werden nach jedem Durchlauf der Schleife die Intervallgrenzen des zugehörigen Konfidenzintervalls in ListeA (untere Grenze) und ListeB (obere Grenze) gespeichert. Damit ist der Körper vollständig beschrieben und wir können uns der grafischen Darstellung der Konfidenzintervalle widmen.
Wir stellen den exakten Wert von I als (rote) Linie dar und verwenden dafür

```
G1 = Graphics[{Red, Thick, Line[{{0, Log[4/3]}, {k+1, Log[4/3]}}]}];.
```

Eine Darstellung der Konfidenzintervalle ergibt sich durch die Eingabe

```
G2 = Graphics[{Blue, Thick, Table[Line[{{l, ListeA[[l]]}, {l, ListeB[[l]]}}],
    {l, 1, k, 1}]}];.
```

Wir kombinieren die beiden Grafiken mittels

```
Show[G1, G2, Axes → {False, True}, AspectRatio → 1/3, LabelStyle → {Black, Large}].
```

Wertet man diesen Code aus, so sind 100 verschiedene Konfidenzintervalle ersichtlich (Abbildung 3.10).

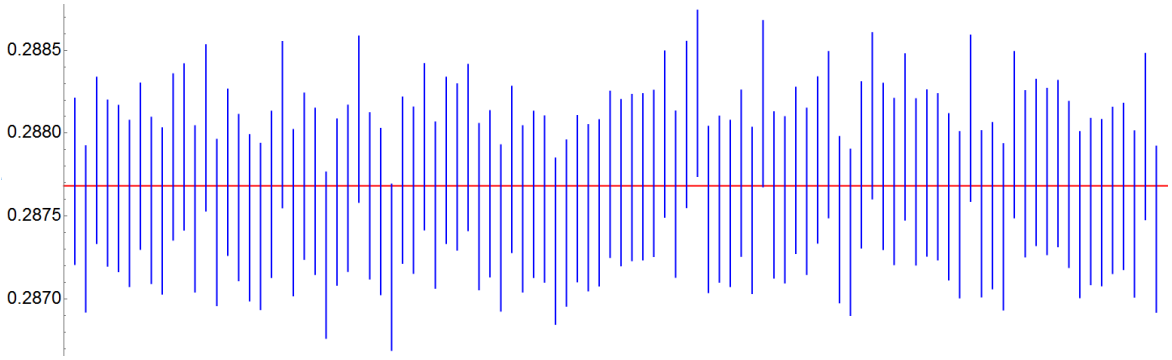

Abbildung 3.10 Ergebnis der Simulation: 100 Konfidenzintervalle für I mit Länge $\varepsilon < 10^{-3}$ und $\alpha = 0,01$

Es ist erkennbar, dass tatsächlich 99 von 100 Intervallen den wahren Wert von I überdecken. Für den hier gewählten Stichprobenumfang n ist die Normalverteilungsannahme also durchaus gerechtfertigt.

Schließlich möchte ich noch kurz auf den Aufbau des Codes eingehen. Durch den Einsatz der `While`-Schleife können die Rechenzeit und der Arbeitsspeicherbedarf[15] stabil gehalten werden. Damit ist gemeint, dass die Rechenzeit in etwa $k = 100$-mal so groß ist (247,63s), wie die Rechenzeit für eine Einzelsimulation mit diesem Stichprobenumfang (Tabelle 3.1). Weil nach jedem Schleifendurchlauf nur die Intervallgrenzen gespeichert werden und der „Rest" verworfen wird, wird nur annähernd so viel Arbeitsspeicher benötigt, wie bei einer Einzelsimulation. Bei diesem konkreten Beispiel wurden für die Auswertung weniger als 25 Megabyte benötigt. Die Differenz zu einer Einzelsimulation (Beispiel 3.5) mit demselben Stichprobenumfang betrug nur zirka drei Kilobyte. Der Einsatz einer `While`-Schleife ermöglicht es uns also, eine Simulation effizient mehrmals zu wiederholen (vgl. dazu Anmerkung 2.12). ∆

3.4.4 Zusammenfassung mehrerer Simulationen

Im nachfolgenden Beispiel wird die (näherungsweise) Normalverteilung des Schätzers I_n um den exakten Wert I veranschaulicht. Diese Veranschaulichung liefert gleichzeitig eine Begründung, warum das arithmetische Mittel von verschiedenen Simulationsergebnissen eine sehr gute Näherung für I darstellt.

Beispiel 3.15 (Fortsetzung 3 von Beispiel 3.5)
Wir werden k Einzelsimulationen durchführen, also k Realisierungen von I_n erzeugen, und uns die Häufigkeitsverteilung der Einzelergbnisse $E_1, \dots, E_k$ ansehen. Schließlich werden wir zeigen, warum das arithmetische Mittel $\overline{E}_k = 1/k \cdot \sum_{j=1}^{k} E_j$ eine sehr gute Näherung für I darstellt.
Wir definieren zunächst den Stichprobenumfang einer Einzelsimulation durch

```
n = 10 000;
```

und den Integranden mit

```
g[x_,y_] = 1/(x+y)^2;.
```

Analog zu Beispiel 3.14 legen wir durch

```
ListeE = {};
```

eine leere Liste, in der wir dann unsere Ergebnisse E_j der Einzelsimulationen sammeln, fest. Mit

```
k = 100 000;j = 1;
```

werden die Anzahl der Einzelsimulationen und der Startindex definiert.
Die `While`-Schleife wird durch

```
While[j < k+1,
      SeedRandom[j,Method → "MersenneTwister"];
      Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];
      AppendTo[ListeE,Mean[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]]];
      j++];
```

festgelegt (vgl. Beispiel 3.14). Für das weitere Vorgehen benötigen wir einige bereits bekannte Resultate, die nun kurz zusammengefasst werden: Wegen Gleichung (3.3) und Beispiel 3.5 wissen wir bereits, dass $I = \mathbb{E}(I_n) = \ln(4/3)$ ist. Gemäß Gleichung (3.4) und Beispiel 3.13 gilt $\sigma^2 = n\mathbb{V}(I_n)/\lambda^m(A)^2 = 11/108 - \ln(4/3)^2$. Mit dem Zentralen Grenzwertsatz folgt unmittelbar, dass I_n in guter Näherung $\mathcal{N}(I; \lambda^m(A)^2\sigma^2/n)$-verteilt ist (vgl. Herleitung von (3.7)).

[15] Mit dem Befehl `MaxMemoryUsed` kann der maximale Speicherbedarf für die Auswertung eines Ausdrucks gemessen werden.

Um den Eingabe-Code übersichtlich zu gestalten, setzen wir

```
w = Log[4/3];s = Sqrt[(11/108 − Log[4/3]^2)/n];
```

und

```
f[x_]= PDF[NormalDistribution[w,s],x];.
```

Die Funktion f beschreibt dann die Dichte der $\mathcal{N}(I;\lambda^m(A)^2\sigma^2/n)$-Verteilung[16]. Das arithmetische Mittel $\overline{E}_k$ erhalten wir durch

```
m = Mean[ListeE];.
```

Nun widmen wir uns der grafischen Darstellung. Wir stellen die (relative) Häufigkeitsverteilung der Simulationsergebnisse E_j mit

```
G1 = Histogram[ListeE,50,"PDF"];
```

dar (vgl. Beispiel 2.11) und plotten die Dichte f im Intervall $[w-4s, w+4s]$ durch

```
G2 = Plot[f[x],{x,w-4s,w+4s},PlotRange → All,PlotStyle → {Black,Thick}];.
```

Zusätzlich stellen wir sowohl den exakten Wert von I (in Rot) als auch den Näherungswert $\overline{E}_k$ (in Blau) als Linie dar. Dafür nutzen wir die Eingabe

```
G3 = Graphics[{Thick,Red,Line[{{w,0},{w,f[w]}}],Blue,Line[{{m,0},{m,f[m]}}]}];.
```

Wir führen die drei Grafiken mittels

```
Show[G1,G2,G3,PlotRange → All,AspectRatio → 1/5,
    Axes → {True,False},LabelStyle → {Black,Large}]
```

zusammen und erhalten nach der Auswertung als Ausgabe Abbildung 3.11. Man sieht, dass die Ergebnisse der Einzelsimulationen E_j in guter Näherung normalverteilt sind.

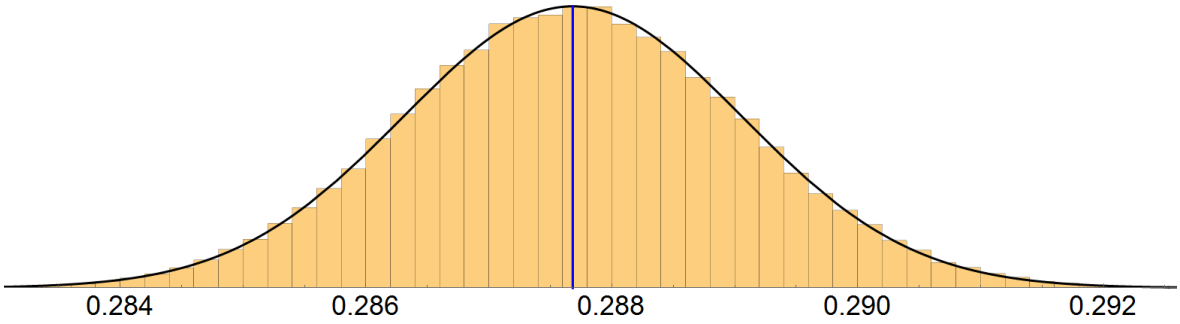

Abbildung 3.11 Ergebnis der wiederholten Simulation für $n = 10\,000$ und $k = 100\,000$

Zusätzlich wird durch Abbildung 3.11 deutlich, warum gerade $\overline{E}_k$ einen guten Näherungswert darstellt: Durch die Verwendung von verschiedenen Stichproben (verschiedenen Anfangswerten) ergeben sich unterschiedliche Simulationsverläufe. Das heißt der exakte Wert wird überschätzt oder unterschätzt, vielleicht sogar genau getroffen. Verlässt man sich nur auf eine spezielle Stichprobe (Einzelsimulation), so weiß man nicht, welcher Fall gerade zutrifft (siehe Abbildung 3.12).

Der Mittelwert $\overline{E}_k$ gleicht diese verschiedenen Verläufe aus und führt uns (nahe) zum exakten Wert. In Abbildung 3.11 ist der Unterschied zwischen I und $\overline{E}_k$ nicht mehr ersichtlich, denn es gilt $|I - \overline{E}_k| < 1{,}59 \cdot 10^{-7}$ (verwende `Abs[m − w]`).

Trotz der aufgrund des Stichprobenumfangs $n = 10\,000$ ungenauen Einzelsimulationen (Tabelle 3.3 und Abbildung 3.12), konnten wir durch oftmalige Wiederholung einen sehr guten Näherungswert gewinnen.

---

[16] Der Befehl `NormalDistribution` erwartet die Standardabweichung statt der Varianz als Eingabe (siehe Definition von `s`).

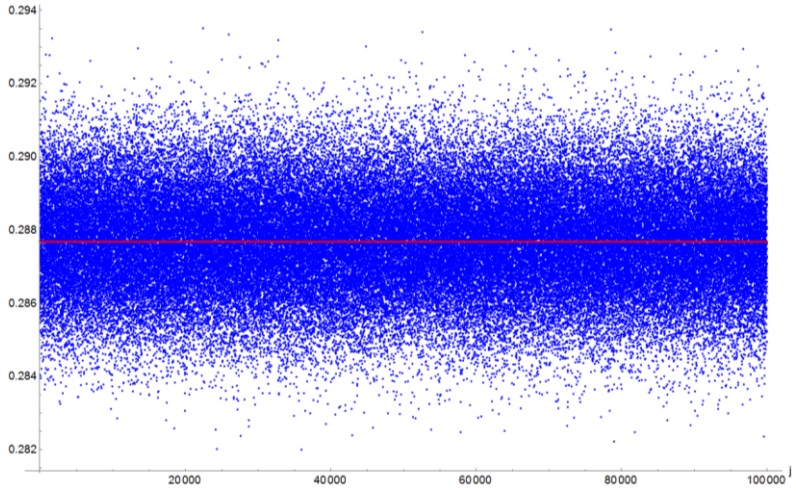

Abbildung 3.12 Darstellung der Simulationsergebnisse E_j (blau) und des exakten Werts von I (rot)

Nicht unerwähnt sei auch die Tatsache, dass bei dieser Berechnung insgesamt 10^9 zufällige Vektoren benutzt wurden. Eine Einzelsimulation mit Stichprobenumfang 10^9 wäre hingegen mit einem Arbeitsspeicher von nur 8 Gigabyte nicht durchführbar gewesen (vgl. Beispiel 3.8). Allgemein können wir somit festhalten: Durch dieses Vorgehen gelingt es uns, $k \cdot n$ zufällige Vektoren in die Rechnung miteinzubeziehen, weil nur mit n Vektoren gerechnet wird[17]. Im Vergleich zu einer Einzelsimulation mit Stichprobenumfang $k \cdot n$, ergibt sich durch diese Implementierung eine enorme Speicherersparnis. ∆

3.4.5 Limitierung durch Computereinsatz

Zum Abschluss dieses Kapitels wollen wir uns noch kurz mit dem Begriff der Integrierbarkeit beschäftigen. In der Theorie funktionieren die vorgestellten Verfahren für Lebesgue-integrierbare Funktionen (deshalb wurde die Integrierbarkeit in den Algorithmen 3.1 und 3.3 vorausgesetzt). In der Praxis ergibt sich jedoch aufgrund des Einsatzes eines Zufallszahlgenerators (bzw. Computers), eine Limitierung. Das folgende Beispiel soll dies illustrieren.

Beispiel 3.16
Wir betrachten die Indikatorfunktion $\mathbf{1}_{\mathbb{Q}} : \mathbb{R} \to \{0,1\}$, die auch als Dirichlet'sche Sprungfunktion bekannt ist. Diese Funktion ist Lebesgue-integrierbar (aber nicht Riemann-integrierbar) und für jedes Intervall $[a, b] \subseteq \mathbb{R}$ gilt, dass

$$\int_a^b \mathbf{1}_{\mathbb{Q}}(x)\, dx = 0$$

ist [3, S. 648].

Mit der naiven Monte-Carlo-Integration würden wir hingegen

$$\int_a^b \mathbf{1}_{\mathbb{Q}}(x)\, dx = 1$$

erhalten (A.14), da ein Zufallszahlgenerator nur rationale Zahlen erzeugt.
Ein Computer kann eben nur mit Gleitkommazahlen, einer Teilmenge der rationalen Zahlen, numerisch operieren. Irrationale Zahlen, wie π oder e, werden von MATHEMATICA als Symbole

[17] Setzt man den Befehl `SeedRandom[...]` vor die `While`-Schleife, so wird eine Stichprobe „weitergeführt".

und nicht als Zahlen aufgefasst (siehe „Atomic Elements of Expressions“ im DOCUMENTATION CENTER). Dies kann mit der Eingabe `Head[Pi]` überprüft werden. Verwendet man hingegen `Head[N[Pi]]`, so erscheint in der Ausgabe `Real` (Dezimalzahl). MATHEMATICA versteht den Ausdruck `N[Pi]` also als Dezimalzahl. Mit der Eingabe `Rationalize[N[Pi],0]` sieht man schließlich, dass die Zahl π bei numerischen Berechnungen durch den Bruch

$$\frac{245850922}{78256779}$$

angenähert wird. MATHEMATICA erkennt diese Zahl schließlich als `Rational`.

Betrachten wir die modifizierte Dirichlet-Funktion $g : \mathbb{R} \to \mathbb{R}$, die durch

$$g(x) := \begin{cases} 1/q & \text{für } x = p/q \in \mathbb{Q} \text{ mit } \mathrm{ggT}(p,q) = 1, \\ 0 & \text{für } x \in \mathbb{R} \setminus \mathbb{Q}, \end{cases}$$

definiert wird. Diese Funktion ist Riemann-integrierbar auf [0,1] und das Integral besitzt darauf den Wert Null [2, Beispiel 3.50]. Auch in diesem Fall scheitern wir aus demselben Grund wie oben. Selbst die schwächere Forderung der Riemann-Integrierbarkeit, ist also aufgrund der technischen Umsetzung zu stark. Natürlich wird man in praktischen Anwendungen nicht mit solchen pathologischen Funktionen in Berührung kommen. Trotzdem zeigen diese Beispiele, dass bei der technischen Umsetzung dieser Integrationsmethode (überabzählbar) viele Punkte unberücksichtigt bleiben. ∆

Zusammenfassend können wir festhalten, dass diese Integrationsmethode in der Theorie für Lebesgue-integrierbare Funktionen zielführend ist. Der Einsatz des Computers verkleinert jedoch die Menge, der auf diese Weise integrierbaren Funktionen. Jedenfalls müssen wir uns keine Sorgen machen, solange die Funktion stetig ist.

4 Ein Vergleich mit der Trapezregel

In diesem Kapitel vergleichen wir die Monte-Carlo-Integration mit der Trapezregel, einem einfachen Stellvertreter von vielen anderen deterministischen Verfahren zur numerischen Integration. Da der Vergleich im Fokus steht, wird die Trapezregel in einer anschaulichen Form, die sich leicht auf mehrere Dimensionen übertragen lässt, eingeführt. Das in der Literatur übliche Vorgehen wird im Rahmen zweier Exkurse kurz erläutert, um auch den Zusammenhang zu anderen (von der Grundidee ähnlichen) Verfahren herzustellen.

Dass gerade die Trapezregel für diesen Vergleich herangezogen wird, hat zweierlei Gründe: Zum einen ist die Trapezregel einfach gestrickt, was uns umfassende theoretische Betrachtungen erspart. Zum anderen wird bereits durch die Trapezregel klar, warum viele weitere deterministische Verfahren zur numerischen Integration bei sehr hochdimensionalen Integralen scheitern. Demgegenüber steht die Monte-Carlo-Integration, die weitgehend unbeeindruckt von einer hohen Dimensionszahl agiert und deshalb die einzige praktikable Möglichkeit darstellt, um in einem solchen Fall numerisch integrieren zu können.

4.1 Die eindimensionale Trapezregel

Zunächst widmen wir uns dem eindimensionalen Fall und führen die eindimensionale Trapezregel ein. Wir betrachten dazu eine stetige Funktion $g : [a, b] \to [0, \infty)$. Unser Ziel ist es wieder

$$I = \int_a^b g(x)\,dx$$

näherungsweise zu ermitteln. Wir machen uns dazu die geometrische Interpretation des Integrals als „Flächeninhalt unter dem Graphen von g“ zunutze. Die „Fläche unter dem Graphen von g“ nähern wir durch eine Rechteck an. Als Breite dieses Rechtecks wählen wir die Intervalllänge $b - a$ und als Höhe wählen wir das arithmetische Mittel der Funktionswerte an den Rändern von $[a, b]$, also $(g(a) + g(b))/2$.

Der Flächeninhalt dieses Rechtecks ist dann eine Näherung für I, es gilt also

$$\int_a^b g(x)\,dx \approx (b - a) \cdot \frac{g(a) + g(b)}{2}. \tag{4.1}$$

Die Näherung (4.1) heißt Trapezregel [7]. Abbildung 4.1 zeigt, dass das festgelegte Rechteck den gleichen Flächeninhalt wie ein Sehnentrapez, welches durch eine lineare Verbindung der Funktionswerte $g(a)$ und $g(b)$ entsteht, besitzt (kongruente Dreiecke).

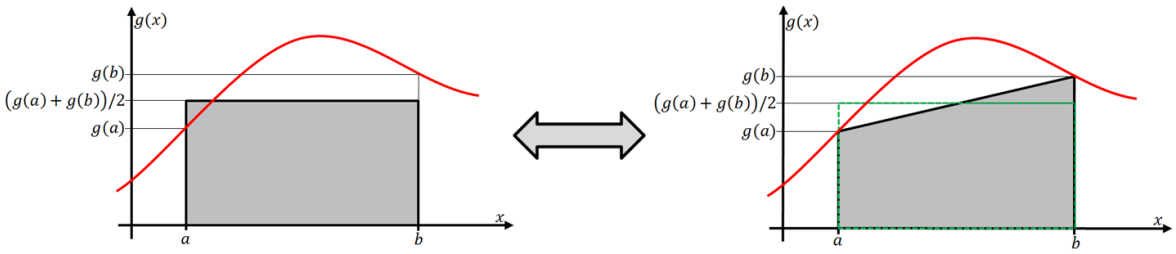

Abbildung 4.1 Darstellung der Vorgehensweise (links) und traditionelle Trapezregel (rechts)

Die gewählte (unübliche) Festlegung der Trapezregel hat den Vorteil, dass sie leicht auf den mehrdimensionalen Fall übertragen werden kann, aber dazu später mehr.

Um die Genauigkeit der Approximation von I zu erhöhen, zerlegen wir das Intervall $[a, b]$ mit $a = x_0 < x_1 < \ldots < x_n = b$. Diese Zerlegung sei äquidistant, das heißt es gelte

$$x_i - x_{i-1} = \frac{b-a}{n} =: h \text{ für } i = 1, 2, \ldots, n.$$

Wir wenden die Trapezregel nun auf jedes Teilintervall $[x_{i-1}, x_i]$ an und summieren die jeweiligen Näherungswerte:

$$\begin{aligned} T_n &= \sum_{i=1}^{n} (x_i - x_{i-1}) \frac{g(x_{i-1}) + g(x_i)}{2} = \frac{h}{2} \sum_{i=1}^{n} g(x_{i-1}) + g(x_i) \\ &= \frac{h}{2} \big(g(x_0) + 2g(x_1) + \ldots + 2g(x_{n-1}) + g(x_1) \big) \qquad (4.2) \\ &= \frac{h}{2} \left(g(a) + g(b) + 2 \sum_{i=1}^{n-1} g(a + ih) \right). \end{aligned}$$

Gleichung (4.2) bezeichnet man dann als zusammengesetzte Trapezregel (oder auch Trapezsumme). Die Konvergenz dieser zusammengesetzten Quadraturregel gegen den Wert von I ist anschaulich klar und kann leicht begründet werden, wie der nachfolgende Satz zeigt.

Satz 4.1
Sei $g : [a, b] \to [0, \infty)$ stetig. Dann gilt:

$$\lim_{n \to \infty} T_n = \int_a^b g(x)\, dx.$$

Beweis:
Da die Funktion g auf $[a, b]$ stetig ist, ist sie Riemann-integrierbar [22, Satz 81.1]. Somit ist

$$\lim_{n \to \infty} \frac{b-a}{n} \sum_{i=1}^{n} g(x_{i-1}) = \int_a^b g(x)\, dx = \lim_{n \to \infty} \frac{b-a}{n} \sum_{i=1}^{n} g(x_i),$$

das heißt die Riemann-Summen mit den linken beziehungsweise rechten Intervallrandpunkten als Zwischenpunkte konvergieren gegen das Integral [30].
Weil

$$g(x_{i-1}) \leq \frac{g(x_{i-1}) + g(x_i)}{2} \leq g(x_i) \text{ für } i = 1, 2, \ldots, n$$

gilt, liegt die Trapezsumme stets zwischen den beiden Riemann-Summen. Damit folgt die Behauptung. ■

Beispiel 4.2 (Näherungsweise Bestimmung von π – Variante 3)
Wir widmen uns wieder, wie in Beispiel 3.2, der Approximation von π durch die näherungsweise Berechnung des Integrals

$$I = \int_0^1 4 \sqrt{1 - x^2}\, dx.$$

Zunächst legen wir fest, in wie viele Teilintervalle das Intervall [0,1] zerlegt werden soll, und setzen zum Beispiel

```
n = 1000;.
```

Den Integranden definieren wir durch

```
g[x_] = 4*Sqrt[1-x^2];.
```

Nun definieren wir die Ränder von [0,1] und die Schrittweite h mittels

```
a = 0.;b = 1.;h = (b - a)/n;.
```

Da die Ränder ganze Zahlen sind, versehen wir sie mit einem Punkt. MATHEMATICA behandelt diese Zahlen dann nicht als ganze Zahlen, sondern als Gleitkommazahlen. Weil die Ränder in allen Rechenschritten auftreten, kann dadurch Rechenzeit eingespart werden.
Die Trapezsumme T_n erhalten wir mithilfe von

```
h/2*(g[a]+g[b]+2*Sum[g[a+i*h],{i,1,n-1}]).
```

Der nach der Auswertung ausgegebene Wert ist bereits eine sehr gute Näherung für π, denn es gilt $|\pi - T_n| < 3{,}8 \cdot 10^{-5}$.
Abbildung 4.2 zeigt anschaulich den Unterschied zwischen der deterministischen Näherung von I mithilfe von T_n und einer Monte-Carlo-Schätzung mithilfe von I_n.

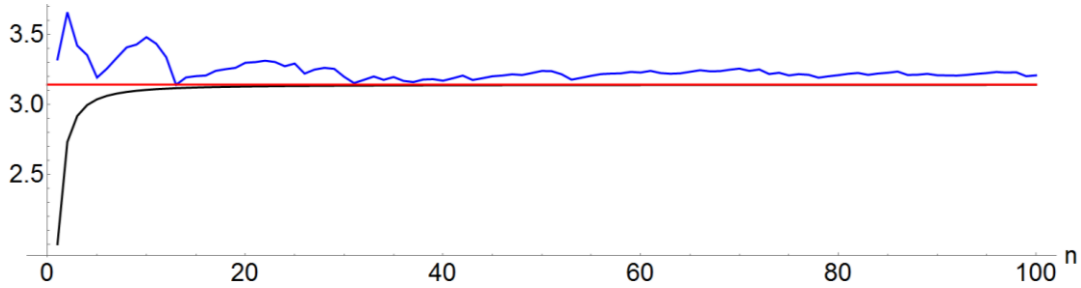

Abbildung 4.2 Trapezsumme T_n (schwarz), Monte-Carlo-Schätzer I_n (blau) und exakter Wert von $I = \pi$ (rot) für wachsendes n

Wir sehen, dass die Trapezsumme im Gegensatz zum Monte-Carlo-Schätzer sehr schnell konvergiert. Der Unterschied zwischen den beiden zugrundeliegenden Konvergenzbegriffen ist ebenfalls deutlich zu erkennen. ∆

Exkurs 4.3 (Newton-Cotes-Formeln)
Die Newton-Cotes-Formeln basieren auf der Idee der Interpolation. Dazu wird das Intervall $[a, b]$ äquidistant in n Teile zerlegt. Die Funktionswerte an den $n + 1$ Randpunkten der resultierenden Teilintervalle bilden dann die Stützstellen für die Interpolation des Integranden durch ein Polynom n-ten Grades. Dieses Interpolationspolynom wird dann exakt integriert, womit sich eine Näherungsformel (Quadraturformel) für das gesuchte Integral ergibt. Nach Konstruktion dieser Formeln ist klar, dass sie Polynome vom Grad höchstens n exakt integrieren. Man kann zeigen, dass (geschlossene[18]) Newton-Cotes-Formeln mit geradem n sogar Polynome vom Grad höchstens $n + 1$ exakt integrieren. Aus den jeweiligen Interpolationsfehlern ergeben sich schließlich einfache Formeln für den jeweiligen Integrationsfehler, wobei eine gewisse Glattheit des Integranden benötigt wird. Die Trapezregel ergibt sich für $n = 1$, also mittels Interpolation durch eine lineare Funktion. Mittels quadratischer Interpolation, also für $n = 2$, ergibt sich die Simpsonregel, die auch als Kepler'sche Fassregel bekannt ist [7].
Die Vorgangsweise, um zusammengesetzte Newton-Cotes-Formeln zu konstruieren, haben wir exemplarisch anhand der zusammengesetzten Trapezregel gesehen. Sie kann unmittelbar auf Verfahren höherer Ordnung übertragen werden.

[18] Die Funktionswerte an den Rändern des Intervalls $[a, b]$ sind Stützstellen.

Um nicht vom eigentlichen Thema abzudriften, wurde jedoch auf eine genaue Darstellung verzichtet. Stattdessen wurde die Trapezregel in einer geometrisch anschaulichen Form definiert, die sich plausibel auf mehrere Dimensionen übertragen lässt.
Die numerische Integration (Quadratur) ist fixer Bestandteil sämtlicher Numerik-Lehrbücher. Eine anwendungsorientierte Einführung in die numerische Integration ist zum Beispiel in [15, Kapitel 8] zu finden. Dabei wird ein besonderes Augenmerk auf die Trapezregel und die Simpsonregel gelegt. Zudem wird das auf der Trapezregel basierende Romberg-Verfahren sehr anschaulich erklärt. Die summierte Trapezregel (bzw. Simpsonregel) basiert auf einer äquidistanten Zerlegung des Integrationsgebiets. Bei adaptiven Quadraturmethoden wird diese Forderung fallen gelassen und die Schrittweite bei der Zerlegung an die Form des Integranden angepasst. Dieses Vorgehen wird in [15] anhand der Simpsonregel demonstriert. Schließlich wird auch die Gauß-Integration thematisiert, bei der durch die spezielle Wahl von $n+1$ Stützstellen Quadraturformeln konstruiert werden, die dann sogar Polynome vom Grad höchstens $2n+1$ exakt integrieren. Einen umfassenden Einblick in die numerische Integration bietet auch [7, Kapitel 13]. Wir kehren nun wieder zum eigentlichen Thema dieses Kapitels zurück.

Deterministische Verfahren haben den Vorteil, dass strenge Fehlerschranken angegeben werden können. Für die zusammengesetzte Trapezregel gilt zum Beispiel die Abschätzung

$$|I - T_n| \leq \frac{|b-a|^3}{12n^2} \cdot \max_{\xi \in [a,b]} |g''(\xi)|,$$

falls g zweimal stetig differenzierbar ist (für einen Beweis siehe z. B. [15, 8.4]). Man sagt daher auch, dass die Trapezregel für zweimal stetig differenzierbare Funktionen die Konvergenzordnung 2 besitzt, und schreibt kurz[19] $T_n = \mathcal{O}(n^{-2})$. Wie (3.4) zeigt, ergibt sich für den Monte-Carlo-Schätzer nur die Konvergenzordnung $1/2$, also $I_n = \mathcal{O}\left(n^{-1/2}\right)$. Die Anforderungen an die zu integrierende Funktion sind jedoch viel schwächer, da sie lediglich quadratisch integrierbar sein muss, um den Integrationsfehler abschätzen zu können.

Schließlich können wir festhalten, dass die (naive) Monte-Carlo-Integration selbst einem derartig einfachen Verfahren wie der zusammengesetzten Trapezregel im Eindimensionalen hoffnungslos unterlegen ist [4]. Dies ändert sich jedoch im Mehrdimensionalen.

4.2 Die mehrdimensionale Trapezregel

Exemplarisch sehen wir uns die Trapezregel im zweidimensionalen Fall an. Wir beschränken uns dabei auf ein Rechteck $A = [a,b] \times [c,d]$. Unser Ziel ist es nun eine Näherung für das Integral

$$I = \int_A g(x,y)\, d(x,y)$$

zu bestimmen, wobei die Funktion $g : A \to [0,\infty)$ stetig sei. Für Funktionen in zwei Variablen können wir weiterhin von der geometrischen Interpretation des Integrals Gebrauch machen. In Verallgemeinerung zum eindimensionalen Fall verwenden wir als Näherung für das „Volumen unter dem Graphen von g“ das Volumen eines Quaders. Als Grundfläche wählen wir das

[19] Seien $q \in \mathbb{Q}$ und $(a_n)_{n\in\mathbb{N}}$ eine Folge. Wir schreiben $a_n = \mathcal{O}(n^q)$, falls $|a_n|$ nicht schneller als n^q wächst, das heißt wenn gilt: $\exists c > 0\ \exists n_0 \in \mathbb{N}\ \forall n \geq n_0 : |a_n| \leq c \cdot n^q$.

Rechteck A mit $\lambda^2(A) = (d-c)(b-a)$ und als Höhe wählen wir das arithmetische Mittel der Funktionswerte an den Eckpunkten von A, also

$$\frac{g(a,c) + g(b,c) + g(a,d) + g(b,d)}{4}.$$

Somit ergibt sich die Näherung

$$\int_A g(x,y)\, d(x,y) \approx (d-c)(b-a) \cdot \frac{g(a,c) + g(b,c) + g(a,d) + g(b,d)}{4}. \tag{4.3}$$

Gleichung (4.3) ist die zweidimensionale Trapezregel [12]. Die nachfolgende Bemerkung und der darauffolgende kurze Exkurs liefern eine Begründung, warum diese Festlegung zu Recht den Namen Trapezregel verdient.

Bemerkung 4.4

Abbildung 4.3 zeigt, wie, durch aufeinanderfolgende Anwendung der eindimensionalen Trapezregel (4.1) zwischen den Eckpunkten von A, die zweidimensionale Trapezregel (4.3) konstruiert werden kann.

Wir beachten dabei die in Abbildung 4.1 dargestellte Überlegung und betrachten zunächst die linke Darstellung in Abbildung 4.3: Dabei wird die Trapezregel in x-Richtung angewendet, woraus sich $(g(a,c) + g(b,c))/2$ für die Höhe der Seitenfläche über (a,c) und (b,c) ergibt. Entsprechend ergibt sich die Höhe der Seitenfläche über den beiden Punkten (a,d) und (b,d) zu $(g(a,d) + g(b,d))/2$.

In der rechten Darstellung wird die Trapezregel nun analog in y-Richtung (auf das vorherige Resultat) angewendet. Es entsteht dann ein Quader mit Höhe

$$\frac{1}{2} \cdot \left(\frac{g(a,c) + g(b,c)}{2} + \frac{g(a,d) + g(b,d)}{2} \right) = \frac{g(a,c) + g(b,c) + g(a,d) + g(b,d)}{4}.$$

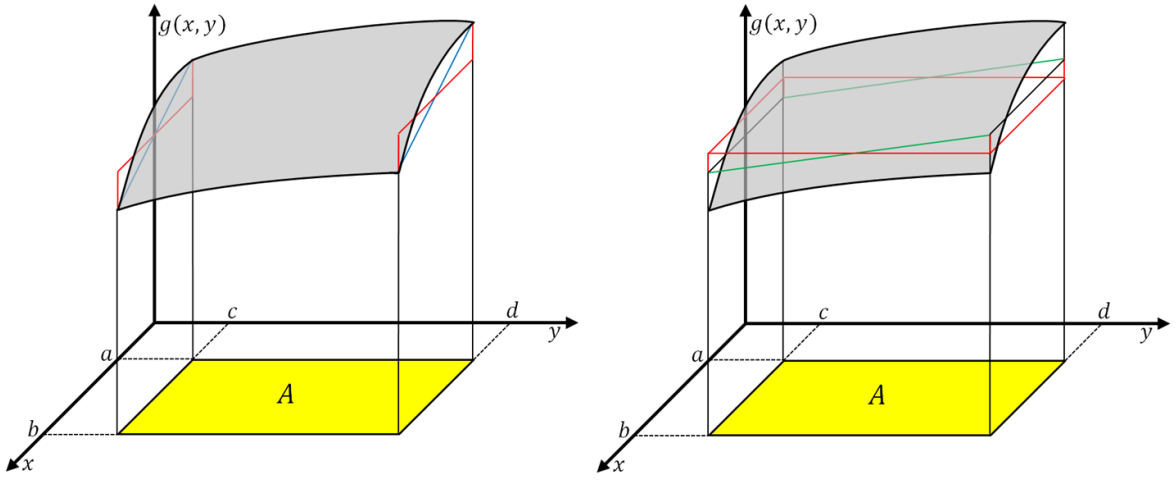

Abbildung 4.3 Anwendung der Trapezregel zunächst in x-Richtung (blau, links) und anschließend in y-Richtung (grün, rechts) unter Beachtung der Idee aus Abbildung 4.1

Exkurs 4.5 (Newton-Cotes-Formeln für rechteckige Integrationsbereiche)

Die Newton-Cotes-Formeln für rechteckige Integrationsbereiche $A = [a,b] \times [c,d]$ können in Analogie zum eindimensionalen Fall beziehungsweise über einen Produktansatz hergeleitet werden [12, 15.3.]. Die zweidimensionale Trapezregel basiert dann beispielweise auf einem (bilinearen) Interpolationspolynom der Form

$$p(x,y) = \sum_{i=0}^{1} \sum_{j=0}^{1} a_{ij} x^i y^j = a_{00} + a_{10}x + a_{01}y + a_{11}xy.$$

Verwendet man die Funktionswerte des Integranden an den vier Eckpunkten als Stützstellen, so ergeben sich vier Gleichungen in den Variablen $a_{00}, a_{10}, a_{01}, a_{11}$. Um die Koeffizienten des Polynoms p zu erhalten, muss das (eindeutig lösbare) lineare Gleichungssystem gelöst werden. Schließlich kann der Satz von Fubini angewendet werden, um das ermittelte Interpolationspolynom über A zu integrieren und damit die Trapezregel zu erhalten (A.17).
Bedenken wir, dass p entlang der Kanten von A linear ist und mit dem Satz von Fubini zunächst in x-Richtung und dann in y-Richtung (oder umgekehrt) integriert wird, so ist klar, warum uns die Überlegung in Bemerkung 4.4 zur zweidimensionalen Trapezregel führte.

Für die Herleitung der zweidimensionalen zusammengesetzten Trapezregel zerlegen wir das Rechteck A in kleine Teilrechtecke A_{ij}, wenden jeweils (4.3) an und addieren die jeweiligen Werte. Wir zerlegen das Intervall $[a, b]$ äquidistant mit $a = x_0 < x_1 < \ldots < x_n = b$ in n Teilintervalle. Es gelte also

$$x_i - x_{i-1} = \frac{b-a}{n} =: h_x \text{ für } i = 1, 2, \ldots, n.$$

Genauso zerlegen wir $[c, d]$ mittels $c = y_0 < y_1 < \ldots < y_m = d$ in m gleich lange Teilintervalle und setzen

$$y_j - y_{j-1} = \frac{d-c}{m} =: h_y \text{ für } j = 1, 2, \ldots, m.$$

Abbildung 4.4 zeigt die daraus resultierende Zerlegung des Rechtecks A in $n \cdot m$ kleine Rechtecke $A_{ij} = [x_{i-1}, x_i] \times [y_{j-1}, y_j]$ für $1 \leq i \leq n$ und $1 \leq j \leq m$.

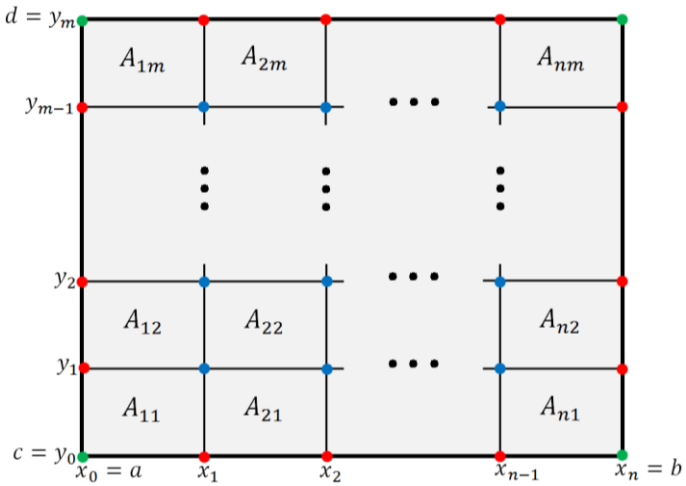

Abbildung 4.4 Darstellung der Zerlegung des Rechtecks A in kleine Rechtecke A_{ij}

Jedes Rechteck A_{ij} hat nach Voraussetzung den Flächeninhalt $\lambda^2(A_{ij}) = h_x \cdot h_y$. Der Faktor $1/4$ tritt ebenfalls bei jeder Anwendung der Trapezregel auf ein Rechteck A_{ij} auf. Somit können wir bei der Summation über alle Rechtecke den Faktor $h_x h_y/4$ vor die Gesamtsumme ziehen. Wir beschränken uns bei der nachfolgenden Überlegung also nur noch auf die Summation der Funktionswerte an den Eckpunkten des jeweiligen Rechtecks A_{ij}. Die farbliche Unterscheidung der Eckpunkte in Abbildung 4.4 wird uns diese Summation erleichtern.

Betrachten wir zum Beispiel den blauen Punkt (x_1, y_1). Der Funktionswert $g(x_1, y_1)$ wird bei der Anwendung der Trapezregel auf A_{11}, A_{21}, A_{12} und A_{22} verwendet und tritt folglich in der Gesamtsumme vierfach auf. Funktionswerte an roten (bzw. grünen) Punkten treten dem-

entsprechend doppelt (bzw. einfach) in der Gesamtsumme auf. Aufgrund dieser Überlegung können wir nun einfach zeilenweise die Funktionswerte addieren (Tabelle 4.1).

j	**Zeilensumme**
0	$g(a,c) + 2\sum_{i=1}^{n-1} g(a+ih_x,c) + g(b,c)$
1	$2g(a,c+h_y) + 4\sum_{i=1}^{n-1} g(a+ih_x,c+h_y) + 2g(b,c+h_y)$
⋮	⋮
$m-1$	$2g(a,c+(m-1)h_y) + 4\sum_{i=1}^{n-1} g(a+ih_x,c+(m-1)h_y) + 2g(b,c+(m-1)h_y)$
m	$g(a,d) + 2\sum_{i=1}^{n-1} g(a+ih_x,d) + g(b,d)$.

Tabelle 4.1 Berechnung der Zeilensummen gemäß Abbildung 4.4

Somit ergibt sich die zweidimensionale zusammengesetzte Trapezregel zu

$$\begin{aligned} T_{nm} = \frac{h_x h_y}{4}[&g(a,c) + g(b,c) + g(a,d) + g(b,d) \\ &+2\sum_{i=1}^{n-1}(g(a+ih_x,c) + g(a+ih_x,d)) \\ &+2\sum_{j=1}^{m-1}(g(a,c+jh_y) + g(b,c+jh_y)) \\ &+4\sum_{j=1}^{m-1}\sum_{i=1}^{n-1} g(a+ih_x,c+jh_y)] \end{aligned} \tag{4.4}$$

Die Konvergenz von T_{nm} gegen I für $n,m \to \infty$ kann in völliger Analogie zum Beweis von Satz 4.1 gezeigt werden. In Verallgemeinerung zum eindimensionalen Fall ergibt sich zudem die Fehlerabschätzung

$$|I - T_{nm}| \leq \frac{|(d-c)(b-a)|}{12}\left[h_x^2 \max_{(x,y)\in A}\left|\frac{\partial^2 g}{\partial x^2}\right| + h_y^2 \max_{(x,y)\in A}\left|\frac{\partial^2 g}{\partial y^2}\right|\right]$$

falls die zweiten partiellen Ableitungen von g stetig sind [17].

Beispiel 4.6

Wir betrachten wieder das Integral aus Beispiel 3.5 und versuchen es mit der zweidimensionalen zusammengesetzten Trapezregel näherungsweise zu bestimmen.

Zu Beginn legen wir

```
n = 99;m = 99;
```

fest. Das Integrationsgebiet $A = [0,1] \times [1,2]$ definieren wir durch

```
a = 0.;b = 1.;c = 1.;d = 2.;
```

und die Schrittweiten h_x und h_y mittels

```
h1 = (b-a)/n;h2 = (d-c)/m;.
```

Nun erfolgt die Definition des Integranden mit

```
g[x_,y_] = 1/(x+y)^2;.
```

Um Ressourcen zu sparen, realisieren wir die Berechnung der Zeilensummen aus Tabelle 4.1 für $1 \leq j \leq m-1$ durch eine `While`-Schleife. Wie in den vorherigen Simulationen, brauchen wir dazu zunächst eine leere Hilfsliste und einen Startindex und setzen deshalb

```
HListe = {};j = 1;.
```

Die `While`-Schleife hat dann die Form

```
While[j < m,Körper;j++];.
```

Im `Körper` setzen wir zunächst

```
s = c+j*h2;,
```

um die nachfolgende Eingabe übersichtlicher und effizienter zu gestalten. Dann folgt die Berechnung der j-ten Zeilensumme, wobei der errechnete Wert in der Hilfsliste gespeichert wird, mit

```
AppendTo[HListe,2*(g[a,s]+g[b,s])+4*Sum[g[a+i*h1,s],{i,1,n-1}]].
```

Damit ist die Eingabe des `Körpers` abgeschlossen. Schließlich können wir T_{nm} durch die Eingabe von

```
(h1*h2)/4*(g[a,c]+g[b,c]+g[a,d]+g[b,d]
           +2*Sum[g[a+i*h1,c]+g[a+i*h1,d],{i,1,n-1}]
           +Total[HListe])
```

realisieren. Gleichung (4.4) ist dabei deutlich erkennbar, wobei wir die letzten beiden Zeilen der dortigen Summation durch die `While`-Schleife und den Befehl `Total`, der die Gesamtsumme der Zeilen mit $j = 1, \ldots, m-1$ liefert, umgesetzt haben.

Da der Rechenaufwand bei einem solchen Algorithmus wesentlich durch die benötigte Anzahl von Funktionsauswertungen bestimmt ist, nehmen wir diese Anzahl als Vergleichskriterium. Die Realisierung des Monte-Carlo-Schätzers I_n ist stets mit n Funktionsauswertungen verbunden, jene von T_{nm} hingegen mit $(n+1)(m+1)$ bei unserer Implementierung.

Dies wird im Folgenden berücksichtigt, damit ein Vergleich mit den Ergebnissen der (naiven) Monte-Carlo-Integration aus Tabelle 3.1 (S. 35) möglich ist. Tabelle 4.2 zeigt die Ergebnisse eines umfassenden Testlaufs mit der zweidimensionalen zusammengesetzten Trapezregel.

n	m	$(n+1)(m+1)$	T_{nm}	$\lvert I - T_{nm}\rvert$	**Rechenzeit [s]**
99	99	10 000	0,2876924646 ...	$1{,}0 \cdot 10^{-5}$	0,04
159	124	20 000	0,2876873989 ...	$5{,}3 \cdot 10^{-6}$	0,06
199	199	40 000	0,2876846444 ...	$2{,}6 \cdot 10^{-6}$	0,11
319	249	80 000	0,2876833942 ...	$1{,}3 \cdot 10^{-6}$	0,27
399	399	160 000	0,2876827122 ...	$6{,}4 \cdot 10^{-7}$	0,62
639	499	320 000	0,2876824016 ...	$3{,}3 \cdot 10^{-7}$	1,20
799	799	640 000	0,2876822319 ...	$1{,}6 \cdot 10^{-7}$	1,87
1279	999	1 280 000	0,2876821546 ...	$8{,}2 \cdot 10^{-8}$	3,54
1599	1599	2 560 000	0,2876821122 ...	$4{,}0 \cdot 10^{-8}$	7,14
2559	1999	5 120 000	0,2876820929 ...	$2{,}1 \cdot 10^{-8}$	15,17
3199	3199	10 240 000	0,2876820824 ...	$1{,}0 \cdot 10^{-8}$	29,52
5119	3999	20 480 000	0,2876820775 ...	$5{,}1 \cdot 10^{-9}$	58,53
6399	6399	40 960 000	0,2876820749 ...	$2{,}5 \cdot 10^{-9}$	119,99
10239	7999	81 920 000	0,2876820737 ...	$1{,}3 \cdot 10^{-9}$	243,33
12799	12799	163 840 000	0,2876820730 ...	$6{,}2 \cdot 10^{-10}$	460,71

Tabelle 4.2 Ergebnisse des Tests der zweidimensionale Trapezregel

Vergleicht man Tabelle 4.2 mit Tabelle 3.1, so ist die Überlegenheit der zusammengesetzten Trapezregel gegenüber einer einfachen (naiven) Monte-Carlo-Integration deutlich ersichtlich. Dennoch liefert dieses Beispiel einen ersten Hinweis, warum die Monte-Carlo-Integration für größere Dimensionszahlen die Methode der Wahl ist. ∆

Die Idee der obigen Herleitung für die Trapezregel können wir, auch ohne unmittelbarer geometrischer Einsicht, für eine stetige Funktion

$$g : A \to [0, \infty) \text{ mit } A = [a_1, b_1] \times \ldots \times [a_m, b_m] \subset \mathbb{R}^m$$

und $m \geq 3$ weiterführen: Die m-dimensionale Trapezregel basiert dann auf 2^m Eckpunkten (eines m-dimensionalen Quaders). Das arithmetische Mittel der Funktionswerte an den Eckpunkten dieses Quaders bilden dann die „Höhe“ eines $(m + 1)$-dimensionalen Quaders, der als Näherung für I fungiert. Als „Grundfläche“ ist entsprechend das Integrationsgebiet A zu wählen.

Durch Zerlegung des Integrationsgebiets A in kleine Quader (mit selber Größe) und wiederholte Anwendung der m-dimensionalen Trapezregel kann dann die m-dimensionale zusammengesetzte Trapezregel konstruiert werden.

Werden in Analogie zum zweidimensionalen Fall das Intervall $[a_1, b_1]$ in n_1, $[a_2, b_2]$ in n_2, … und $[a_m, b_m]$ in n_m Teilintervalle äquidistant zerlegt, so sind bei (guter Implementierung) der zusammengesetzten Trapezregel $(n_1 + 1)(n_2 + 1) \ldots (n_m + 1)$ Funktionswerte zu berechnen. Eine Verfeinerung der Zerlegung wird somit für hohe Dimensionszahlen zu aufwändig und kann schließlich nicht mehr technisch realisiert werden [8; 29; 38].

Damit ist die Bühne frei für die Monte-Carlo-Integration. Die Konvergenz gegen den exakten Wert des zu bestimmenden Integrals ist zwar langsam und von stochastischer Natur. Die Methode ist jedoch robust und weitgehend unabhängig[20] von der Dimensionszahl des Integranden [8]. Zudem wird nur die quadratische Integrierbarkeit des Integranden benötigt, um den Integrationsfehler abschätzen zu können. Zu erwähnen ist natürlich auch, dass die zusammengesetzten Quadraturformeln für wachsende Dimensionszahlen immer unübersichtlicher (und damit schwerer zu implementieren) werden. Die Monte-Carlo-Integration behält hingegen ihre einfache Struktur auch in hohen Dimensionen, weshalb auch die Implementierung einfach bleibt. Schließlich stellen nach Bemerkung 3.10 auch kompliziertere (beschränkte) Integrationsgebiete kein Problem für die Monte-Carlo-Integration dar, was die universelle Einsetzbarkeit dieser Methode unterstreicht.

Schon Benjamin Franklin meinte, dass Zeit Geld sei, was heutzutage mehr denn je zutreffend ist. Deshalb werden wir uns im nächsten Kapitel mit der Beschleunigung der Monte-Carlo-Integration auseinandersetzen.

[20] Die weitgehende Unabhängigkeit meint dabei die Tatsache, dass es natürlich aufwändiger wird zufällige Vektoren auf $\mathbb{R}^m$ für großes m zu erzeugen und die Funktion dort auszuwerten. Trotzdem erhöht sich durch das Hinzufügen eines jeden zufälligen Vektors die Genauigkeit des Verfahrens (siehe (3.4)).

5 Beschleunigung der Monte-Carlo-Integration

Wie bereits angekündigt, befasst sich dieses Kapitel mit der Verbesserung der Monte-Carlo-Integration hinsichtlich ihrer Konvergenzgeschwindigkeit. Zunächst werden dazu vier Methoden zur Varianzreduktion vorgestellt. Dabei dienten mir [25] und [39] als Hauptquellen. In 5.5 wird die Quasi-Monte-Carlo-Integration, ein anderer Zugang um die gewünschte Beschleunigung zu erreichen, kurz beschrieben. Schließlich wird in 5.6 der in MATHEMATICA implementierte Algorithmus zur Monte-Carlo-Integration diskutiert.

In Kapitel 3 haben wir gesehen, dass für die Varianz des naiven Monte-Carlo-Schätzers I_n der Zusammenhang

$$\mathbb{V}(I_n) = \frac{\lambda^m(A)^2\,\sigma^2}{n} \tag{5.1}$$

gilt. Um die Varianz des Schätzers zu senken und damit die Konvergenzgeschwindigkeit zu erhöhen, könnte man beispielweise versuchen den Faktor σ^2 zu verkleinern. Genau an dieser Stelle setzen Methoden zur Varianzreduktion an.

Wir haben den Monte-Carlo-Schätzer aus (3.2) als naiv bezeichnet, da er auf einer einfachen Funktionsauswertung als Grundexperiment basiert. Eine etwaige spezielle Form des Integranden bleibt dabei völlig unberücksichtigt. Methoden zur Varianzreduktion versuchen diese Lücke durch Abänderung des Grundexperiments zu schließen. Aus der Erwartungstreue des Grundexperiments folgt dann unmittelbar, dass ein darauf basierender (modifizierter) Schätzer erwartungstreu und stark konsistent bleibt.

Wir beschränken uns bei der Vorstellung derartiger Methoden auf den eindimensionalen Fall, da die jeweilige zugrundeliegende Idee anschaulich erklärt werden kann, die Notation übersichtlich bleibt und die Verallgemeinerung auf den m-dimensionalen Fall, wie bereits in Kapitel 3 gezeigt, ohnehin auf der Hand liegt.

5.1 Importance Sampling

Zunächst beschäftigen wir uns mit der Methode des Importance Sampling. Seien dazu X eine auf $[a,b]$ gleichverteilte Zufallsvariable (mit Dichte f) und $\tilde{X}$ eine Zufallsvariable mit positiver Dichte $\tilde{f}$. Sei $g : \mathbb{R} \to \mathbb{R}$ eine positive und quadratisch integrierbare Funktion. Dann gilt

$$\mu := \mathbb{E}(g(X)) = \int_{\mathbb{R}} g(x)\cdot f(x)\,dx = \int_{\mathbb{R}} \frac{g(x)\cdot f(x)}{\tilde{f}(x)}\cdot \tilde{f}(x)\,dx = \mathbb{E}(h(\tilde{X})), \tag{5.2}$$

wobei $h(x) := g(x)f(x)/\tilde{f}(x)$ ist. Gleichung (5.2) zeigt also, dass wir für die Bestimmung des Erwartungswerts von $g(X)$ auch alternativ den Erwartungswert von $h(\tilde{X})$ benutzen können.

Welchen Sinn dieser Wechsel des Wahrscheinlichkeitsmaßes hat wird im Folgenden deutlich. Betrachten wir hierzu die Varianz von $h(\tilde{X})$, also

$$\mathbb{V}(h(\tilde{X})) = \int_{\mathbb{R}} \left(\frac{g(x)\cdot f(x)}{\tilde{f}(x)} - \mu\right)^2 \cdot \tilde{f}(x)\,dx. \tag{5.3}$$

Angenommen wir legen die Hilfsdichte $\tilde{f}$ durch

$$\tilde{f}(x) \coloneqq \frac{g(x) \cdot f(x)}{\mu} \text{ für } x \in \mathbb{R}$$

fest[21], so würde die Varianz von $h(\tilde{X})$ sogar verschwinden. Dieser Ansatz misslingt jedoch in der Praxis, da er die Kenntnis des Erwartungswerts μ benötigen würde[22]. Aus (5.3) können wir jedoch ablesen, dass die Wahl der Hilfsdichte $\tilde{f}$ dann günstig ist, wenn $g(x) \cdot f(x)/\tilde{f}(x)$ nahezu konstant ist. Anschaulich bedeutet dies also, dass der Graph von $\tilde{f}$ dem Graphen von g ähneln sollte, damit die Varianz von $h(\tilde{X})$ klein ist.

Gleichzeitig zeigt sich dadurch die zugrundeliegende Idee dieser Methode: Der Integrand soll vermehrt dort ausgewertet werden, wo er große Werte besitzt, da diese Werte mehr zum Integral beitragen und damit wichtiger sind. Dies erklärt auch den Name dieser Methode.

Auf diesen Beobachtungen basierend, definieren wir für die näherungsweise Bestimmung des Integrals

$$I = \int_a^b g(x)\, dx \tag{5.4}$$

einen modifizierten Monte-Carlo-Schätzer durch

$$\tilde{I}_n \coloneqq \frac{b-a}{n} \sum_{i=1}^{n} h(\tilde{X}_i), \tag{5.5}$$

wobei $\tilde{X}_1, \dots, \tilde{X}_n$ eine Folge von unabhängigen gemäß $\tilde{f}$ verteilten Zufallsvariablen ist.

Die wesentliche Voraussetzung, um eine Monte-Carlo-Integration mit diesem Schätzer durchführen zu können, ist jedoch, dass Zufallszahlen, die der Dichte $\tilde{f}$ folgen, erzeugt werden können (siehe dazu Abschnitt 2.3 und 2.4 bzw. Exkurs 3.9).

Beispiel 5.1

Wir suchen einen Wert für das Integral

$$I = \int_{-3}^{3} \exp(-x^2)\, dx.$$

Dieses Integral können wir nicht mehr exakt lösen, da bekanntlich $\exp(-x^2)$ keine Stammfunktion besitzt. Wir haben hier also ein Beispiel vorliegen, bei dem wir tatsächlich nur mithilfe von numerischer Integration zum Ziel gelangen.
Die Form des Integranden zeigt unmittelbar, dass die Dichte der $\mathcal{N}(0; 1/2)$-Verteilung, also

$$\tilde{f}(x) = \frac{1}{\sqrt{4\pi}} \exp(-x^2) \text{ für } x \in \mathbb{R},$$

eine geeignete Hilfsdichte darstellt. Sei deshalb $\tilde{X}$ eine $\mathcal{N}(0; 1/2)$-verteilte Zufallsvariable.
Damit können wir die Hilfsfunktion h durch

$$h(x) \coloneqq \frac{\sqrt{\pi}}{6} \cdot 1_{[-3,3]}(x) \text{ für } x \in \mathbb{R} \tag{5.6}$$

definieren.

---

[21] Die Funktion $\tilde{f}$ beschreibt unter unseren Annahmen tatsächlich eine Dichte, wie leicht nachzuprüfen ist.
[22] Würden wir den Erwartungswert μ kennen, so müssten wir ihn ja nicht mithilfe der Monte-Carlo-Integration ermitteln.

Wir schätzen zunächst $\tilde{\sigma}^2 := \mathbb{V}(h(\tilde{X}))$ mithilfe der Stichprobenvarianz. Um eine gute Näherung für $\tilde{\sigma}^2$ zu erhalten, verwenden wir das arithmetische Mittel verschiedener Schätzungen von $\tilde{\sigma}^2$ mit jeweils anderen Stichproben.
Wir legen dazu zunächst den Stichprobenumfang mit

```
n = 1 000 000;
```

fest und definieren die Hilfsfunktion h mittels

```
h[x_] = Sqrt[Pi]/6.*Boole[-3. ≤ x ≤ 3.];.
```

Für den Einsatz einer `While`-Schleife legen wir eine leere Liste und einen Startindex fest:

```
ListeS = {};j = 1;.
```

Wir wiederholen die Schätzung 200-mal und verwenden daher eine `While`-Schleife der Form

```
While[j < 201,Körper;j++];.
```

In den `Körper` schreiben wir

```
SeedRandom[j,Method → "Congruential"];
HListe = RandomVariate[NormalDistribution[0.,Sqrt[1/2.]],n];
```

um n Realisierungen $\tilde{X}$ zu generieren. Die Stichprobenvarianz erhalten wir dann mit

```
AppendTo[ListeS,Variance[Table[h[HListe[[i]]],{i,1,n}]]].
```

Damit ist der `Körper` fertig. Schließlich geben wir noch

```
S = Mean[ListeS]
```

ein, um einen (guten) Schätzwert für $\tilde{\sigma}^2$ zu ermitteln.
Die Auswertung dieser Eingabe ist sehr aufwändig. Da unsere weiteren Überlegungen auf diesem Ergebnis basieren, müssen wir diesen Aufwand jedoch in Kauf nehmen. Die Berechnung liefert schließlich $\tilde{\sigma}^2 \approx 1{,}99617 \cdot 10^{-6}$.
Wir ermitteln nun, wie groß der Stichprobenumfang zu wählen ist, damit die Varianz des Schätzers den Wert $\varepsilon^2 = 10^{-8}$ unterschreitet. Gemäß (5.1) geben wir dazu

```
m = Ceiling[6^2*S/10^-8]
```

ein, und erhalten $m = 7187$.
In einem neuen Eingabefeld ermitteln wir nun eine (gute) Näherung für I. Dazu realisieren wir $\tilde{I}_m$ 1000-mal und bilden dann das arithmetische Mittel dieser 1000 Realisierungen. Wir definieren eine leere Liste und einen Startindex mit

```
ListeI = {};k = 1;
```

und benutzen eine `While`-Schleife der Form[23]

```
While[k < 1001,SeedRandom[k,Method → "Congruential"];
      HHListe = RandomVariate[NormalDistribution[0.,Sqrt[1/2.]],m];
      AppendTo[ListeI,6.*Mean[Table[h[HHListe[[i]]],{i,1,m}]]];k++];.
```

Der Näherungswert für I ergibt sich dann mit `Mean[ListeI]` zu 1,77241… .
Zu betonen ist, dass wir mit einem einfachen deterministischen Verfahren wie der Trapezregel schneller zu einem besseren Ergebnis gekommen wären, da wir uns im Eindimensionalen befinden. Dieses Beispiel soll jedoch ein mögliches Vorgehen bei einer realen Anwendung der Monte-Carlo-Integration exemplarisch vorführen.
Zusätzlich wollen wir das Verhalten des naiven Monte-Carlo-Schätzers I_n aus (3.2) mit jenem des Importance-Sampling-Schätzers $\tilde{I}_n$ aus (5.5) vergleichen. Bei der naiven Monte-Carlo-In-

[23] Durch den Einsatz des Befehls `Boole` (Indikatorfunktion) in der Definition von `h`, können wir `h` nicht direkt auf die Liste der Zufallszahlen anwenden, sondern müssen die Liste der Funktionswerte elementweise neu konstruieren.

tegration nutzen wir die Symmetrie des Integranden aus, um das Integrationsgebiet zu verkleinern und damit die Varianz von I_n (siehe (5.1)) zu reduzieren. Wir benutzen also, dass

$$I = \int_{-3}^{3} \exp(-x^2)\,dx = 2 \cdot \int_{0}^{3} \exp(-x^2)\,dx \tag{5.7}$$

ist[24]. Exemplarisch beobachten wir den Verlauf der beiden Schätzer I_n und $\tilde{I}_n$ für eine einzige Stichprobe. Abbildung 5.1 zeigt anschaulich die Überlegenheit des Importance-Sampling-Schätzers.

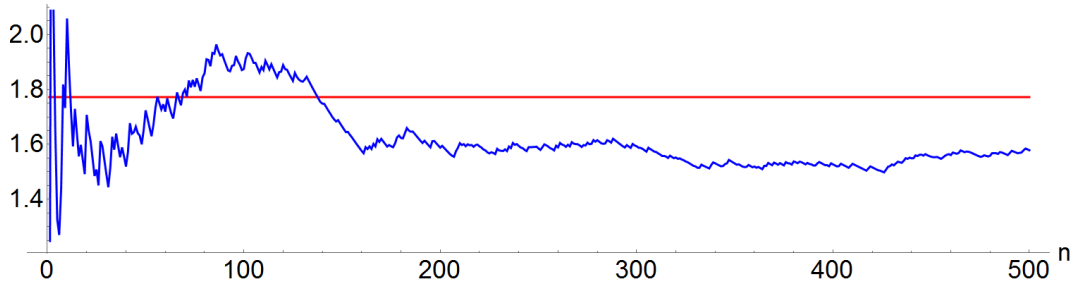

Abbildung 5.1 Importance Sampling - Vergleich von I_n (blau) mit $\tilde{I}_n$ (rot) für wachsendes n

Ein Schätzung mit A.21 lieferte

$$\sigma^2 := \mathbb{V}\big(g(X)\big) \approx 0{,}121598 \text{ für } X \sim \mathcal{U}[0,3]. \tag{5.8}$$

Somit wäre ein Stichprobenumfang von $n = 109\,438\,572$ nötig, damit die Varianz des naiven Monte-Carlo-Schätzers kleiner als $\varepsilon^2 = 10^{-8}$ ist. Dies unterstreicht zusätzlich die deutliche Überlegenheit des Importance-Sampling-Schätzers bei diesem Beispiel. ∆

5.2 Antithetic Sampling

Der nachfolgende Satz liefert unmittelbar eine weitere Idee für ein varianzreduzierendes Verfahren.

Satz 5.2

Seien X, Y zwei identisch verteilte Zufallsvariablen deren Varianz existiert. Dann gelten:

(a) $\mathbb{E}\left(\frac{X+Y}{2}\right) = \mathbb{E}X,$

(b) $\mathbb{V}\left(\frac{X+Y}{2}\right) = \frac{1}{2}\big(\mathbb{V}(X) + \mathrm{Kov}(X,Y)\big).$

Beweis:

Aufgrund der identischen Verteiltheit von X und Y gelten $\mathbb{E}(X) = \mathbb{E}(Y)$ und $\mathbb{V}(X) = \mathbb{V}(Y)$. Mit den Eigenschaften des Erwartungswerts ergibt sich dann (a):

$$\mathbb{E}\left(\frac{X+Y}{2}\right) = \frac{1}{2}\mathbb{E}(X+Y) + \frac{1}{2}(\mathbb{E}X + \mathbb{E}Y) = \frac{1}{2}(2\,\mathbb{E}X) = \mathbb{E}X.$$

Wir erhalten (b) mit den Eigenschaften der Varianz und des Erwartungswerts:

$$\begin{aligned}\mathbb{V}\left(\frac{X+Y}{2}\right) &= \frac{1}{4}\mathbb{V}(X+Y) = \frac{1}{4}\mathbb{E}((X+Y-\mathbb{E}(X+Y))^2) = \frac{1}{4}\mathbb{E}((X-\mathbb{E}X+Y-\mathbb{E}Y)^2)\\ &= \frac{1}{4}\mathbb{E}((X-\mathbb{E}X)^2 + 2(X-\mathbb{E}X)(Y-\mathbb{E}Y) + (Y-\mathbb{E}Y)^2)\\ &= \frac{1}{4}\big(\mathbb{V}(X) + 2\,\mathrm{Kov}(X,Y) + \mathbb{V}(Y)\big) = \frac{1}{2}\big(\mathbb{V}(X) + \mathrm{Kov}(X,Y)\big). \quad \blacksquare\end{aligned}$$

[24] Somit gehen wir hier nicht völlig „naiv“ vor.

Bemerkung 5.3
Wenn Y dieselbe Verteilung wie X besitzt und $\mathrm{Kov}(X,Y) \leq 0$ ist, so bezeichnet man Y auch als antithetische Zufallsvariable.

Für die Situation bei der Monte-Carlo-Integration sind $X \coloneqq g(X_1)$ und $Y \coloneqq g(Y_1)$ zu setzen. Diese Festlegung führt uns zur Frage, wie die zu integrierende Funktion g und die Zufallsvariable Y_1 beschaffen sein sollten, damit Y eine antithetische Zufallsvariable ist. Dazu betrachten wir zunächst den folgenden Satz, der eine wesentliche Grundidee liefert.

Satz 5.4
Sei U eine $\mathcal{U}[0,1]$-verteilte Zufallsvariable. Dann ist $1-U$ ebenfalls $\mathcal{U}[0,1]$-verteilt und es gilt:

$$\mathrm{Kov}(U, 1-U) = -\frac{1}{12}.$$

Beweis:
Es gilt

$$\mathbb{P}(1-U \leq t) = \mathbb{P}(U \geq 1-t) = 1-(1-t) = t = \mathbb{P}(U \leq t),$$

womit gezeigt ist, dass U und $1-U$ dieselbe Verteilung besitzen.
Für die Kovarianz zwischen U und $1-U$ ergibt sich, unter Verwendung der soeben gezeigten identischen Verteiltheit,

$$\begin{aligned}\mathrm{Kov}(U, 1-U) &= \mathbb{E}\big(U \cdot (1-U)\big) - \mathbb{E}U \cdot \mathbb{E}(1-U) \\ &= \mathbb{E}U - \mathbb{E}(U^2) - \mathbb{E}(U)^2 = \frac{1}{2} - \frac{1}{3} - \frac{1}{4} = -\frac{1}{12}. \quad \blacksquare\end{aligned}$$

Bemerkung 5.5
Die Zufallsvariable $1-U$ ist somit eine antithetische Zufallsvariable zu U. Dieses Ergebnis kann benutzt werden, um für viele andere Verteilungen eine antithetische Zufallsvariable zu konstruieren: Ist F eine stetige und streng monoton wachsende Verteilungsfunktion so hat die Zufallsvariable $X = F^{-1}(U)$ nach Satz 2.6 (Inversionsmethode) die Verteilungsfunktion F. Man kann zeigen, dass in diesem Fall $Y = F^{-1}(1-U)$ eine antithetische Zufallsvariable ist [25, Satz 18.5].
Da mit dem MATHEMATICA-Befehl `RandomVariate` direkt Zufallszahlen, die einer gewissen Verteilung folgen, erzeugt werden können, ist die nachfolgende Aussage wesentlich: Besitzt eine Zufallsvariable Z eine bezüglich $\mu \in \mathbb{R}$ symmetrische Dichte, so hat die Zufallsvariable $2\mu - Z$ dieselbe Verteilung wie Z [29, 5.1].
Man kann zudem zeigen, dass der gewünschte Effekt erreicht wird, also

$$\mathrm{Kov}\big(g(Z), g(2\mu - Z)\big) \leq 0$$

gilt, wenn g eine monoton fallende oder monoton wachsende Funktion ist[25] [25, Bem. 18.7].

Davon ausgehend definieren wir für die näherungsweise Bestimmung des Integrals (5.4) einen modifizierten Monte-Carlo-Schätzer durch

$$\tilde{I}_n \coloneqq \frac{b-a}{2n} \sum_{i=1}^{n} g\big(2\mu - \tilde{X}_i\big) + g\big(\tilde{X}_i\big), \tag{5.9}$$

wobei $\tilde{X}_1, \ldots, \tilde{X}_n$ eine Folge von unabhängigen gemäß (einer um $\mu \in \mathbb{R}$ symmetrischen Dichte) $\tilde{f}$ verteilten Zufallsvariablen ist.

[25] Eine mehrdimensionale Funktion sollte entsprechend in allen Variablen monoton wachsend oder monoton fallend sein.

Beispiel 5.6
Wir widmen uns wieder dem Integral (5.7) aus Beispiel 5.1. Da der Integrand auf [0,3] streng monoton fallend ist, erwarten wir eine Varianzreduktion durch Antithetic Sampling. Die Dichte der $\mathcal{U}[0,3]$-Verteilung ist symmetrisch bezüglich 3/2. Wir verwenden daher $g(3-X_i)$ als antithetische Zufallsvariable zu $g(X_i)$ für $1 \le i \le n$.
Wir führen zunächst in Analogie zu Beispiel 5.1 eine Schätzung der Varianz durch und erhalten mit A.22

$$\tilde{\sigma}^2 := \mathbb{V}\left(\frac{g(X_1)+g(3-X_1)}{2}\right) \approx 0{,}0194932 < \frac{\sigma^2}{2}.$$

Folglich wäre ein Stichprobenumfang von $n = 17\,543\,860$ nötig, damit die Varianz des Antithetic-Sampling-Schätzers $\tilde{I}_n$ kleiner als $\varepsilon^2 = 10^{-8}$ wird. Die Varianzreduktion ist bei diesem Beispiel also deutlich kleiner als jene, die wir durch Importance Sampling erreichten. ∆

Bemerkung 5.7
Mit Satz 5.2(b) und (5.8) können wir folgern, dass $\mathrm{Kov}\big(g(X_1), g(3-X_1)\big) \approx -0{,}082612$ gilt. Im nächsten Abschnitt werden wir sehen, wie die Kovarianz direkt geschätzt werden kann. Zu erwähnen ist, dass Antithetic Sampling auch in Fällen, bei denen die erreichte Varianzreduktion gering ist, sinnvoll ist: Der naive Monte-Carlo-Schätzer braucht n Zufallszahlen und n Funktionsauswertungen für eine Stichprobe vom Umfang n. Der Antithetic-Sampling-Schätzer kommt hingegen mit $n/2$ Zufallszahlen aus, um ein qualitativ gleichwertiges Ergebnis zu erzielen. Dies führt zu einer Reduktion des Speicherbedarfs. Der Rechenaufwand reduziert sich jedoch nicht, da ebenfalls n Funktionsauswertungen nötig sind [25].

5.3 Kontrollvariable

Wie bei den vorherigen beiden Techniken zur Varianzreduktion wird uns eine wahrscheinlichkeitstheoretische Überlegung einen entscheidenden Hinweis liefern.

Satz 5.8
Seien X, Y Zufallsvariablen deren Varianz existiert und sei $\mathbb{E}(Y) = \mu_Y \in \mathbb{R}$ bekannt. Für $c \in \mathbb{R}$ gelten:
(a) $\mathbb{E}(X - c(Y-\mu_Y)) = \mathbb{E}X$,
(b) $\mathbb{V}(X - c(Y-\mu_Y)) = \mathbb{V}(X) + c^2\,\mathbb{V}(Y) - 2c\,\mathrm{Kov}(X,Y)$.
(c) Die Abbildung $c \mapsto \mathbb{V}(X - c(Y-\mu_Y))$ hat bei $c^\star := \mathrm{Kov}(X,Y)/\mathbb{V}(Y)$ ein Minimum und es gilt:

$$\mathbb{V}\big(X - c^\star(Y-\mu_Y)\big) = \mathbb{V}(X) - \frac{\mathrm{Kov}(X,Y)^2}{\mathbb{V}(Y)} = \mathbb{V}(X)\cdot(1-\mathrm{Kor}(X,Y)^2).$$

Beweis:
Wir erhalten (a) unmittelbar mithilfe der Eigenschaften des Erwartungswerts

$$\mathbb{E}\big(X - c(Y-\mu_Y)\big) = \mathbb{E}X - c\mathbb{E}(Y-\mu_Y) = \mathbb{E}X - c(\mathbb{E}Y - \mu_Y) = \mathbb{E}X.$$

Wegen Satz 1.36(c) ist $\mathbb{V}\big(X - c(Y-\mu_Y)\big) = \mathbb{V}(X - cY)$.
Damit ergibt sich (b) unter Berücksichtigung der Eigenschaften des Erwartungswerts durch

$$\begin{aligned}\mathbb{V}(X-cY) &= \mathbb{E}((X-cY-\mathbb{E}(X-cY))^2) = \mathbb{E}((X-\mathbb{E}X-c(Y-\mathbb{E}Y))^2)\\ &= \mathbb{E}((X-\mathbb{E}X)^2 - 2c(X-\mathbb{E}X)(Y-\mathbb{E}Y) + c^2(Y-\mathbb{E}Y)^2)\\ &= \mathbb{V}(X) - 2c\,\mathrm{Kov}(X,Y) + c^2\,\mathbb{V}(Y).\end{aligned}$$

Um (c) zu zeigen, definieren wir $h(c) \coloneqq \mathbb{V}(X) + c^2\,\mathbb{V}(Y) - 2c\,\mathrm{Kov}(X,Y)$. Dann ist $c^\star$ Nullstelle von $h'(c) = 2c\,\mathbb{V}(Y) - 2\,\mathrm{Kov}(X,Y)$. Da $h''(c) = 2\,\mathbb{V}(Y)$ positiv ist (sofern wir ausschließen, dass Y eine ausgeartete Verteilung besitzt), nimmt h ihr Minimum bei $c^\star$ an. Schließlich ergibt sich

$$\begin{aligned} h(c^\star) &= \mathbb{V}(X) + \frac{\mathrm{Kov}(X,Y)^2}{\mathbb{V}(Y)} - \frac{2\,\mathrm{Kov}(X,Y)^2}{\mathbb{V}(Y)} = \mathbb{V}(X) - \frac{\mathrm{Kov}(X,Y)^2}{\mathbb{V}(Y)} \\ &= \mathbb{V}(X)\left(1 - \frac{\mathrm{Kov}(X,Y)^2}{\mathbb{V}(X)\mathbb{V}(Y)}\right) = \mathbb{V}(X)\cdot(1 - \mathrm{Kor}(X,Y)^2). \quad \blacksquare \end{aligned}$$

Wir halten also fest, dass es uns gelingen kann, die Varianz zu reduzieren, wenn wir eine weitere Zufallsvariable Y (Kontrollvariable) mit bekanntem Erwartungswert einführen. Umso stärker X und Y korreliert sind beziehungsweise umso größer $\mathrm{Kov}(X,Y)^2$ ist, umso größer ist auch der varianzreduzierende Effekt.

Falls die Varianz $\mathbb{V}(Y)$ unbekannt ist, so kann sie wieder mithilfe der Stichprobenvarianz aus (3.5) geschätzt werden. Bei realen Anwendungen ist, wie bereits erwähnt, der Erwartungswert $\mathbb{E}X$ unbekannt. Folglich kann auch $\mathrm{Kov}(X,Y)$ nicht bestimmt werden und muss geschätzt werden. Wir widmen uns nun kurz dieser Thematik[26].

Wir setzen voraus, dass $(X_1,Y_1), \dots, (X_n,Y_n)$ eine Folge von unabhängigen und identisch verteilten Zufallsvektoren ist und $\mathrm{Kov}(X_1,Y_1)$ existiert. Wir definieren die Stichprobenkovarianz durch

$$K_n \coloneqq \frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \overline{X}_n\right)\left(Y_i - \overline{Y}_n\right), \tag{5.10}$$

wobei $\overline{X}_n = 1/n \cdot \sum_{i=1}^{n} X_i$ und $\overline{Y}_n = 1/n \cdot \sum_{j=1}^{n} Y_j$ sind.

Um die Notation in den nachfolgenden Ausführungen übersichtlicher zu gestalten, schreiben wir unter auftretende Summen nur den Laufindex, da stets $1 \le i,j \le n$ gilt. Wir leiten zunächst eine nützliche Identität ab:

$$\begin{aligned} (n-1)K_n &= \sum_i \left(X_i - \overline{X}_n\right)\left(Y_i - \overline{Y}_n\right) = \sum_i \left(X_iY_i - \overline{X}_nY_i - \overline{Y}_nX_i + \overline{X}_n\overline{Y}_n\right) \\ &= \sum_i X_iY_i - \overline{Y}_n\sum_i X_i - \overline{X}_n\sum_i Y_i + n\,\overline{X}_n\overline{Y}_n \\ &= \sum_i X_iY_i - \frac{1}{n}\left(\sum_j Y_j\right)\left(\sum_i X_i\right) - n\,\overline{X}_n\overline{Y}_n + n\,\overline{X}_n\overline{Y}_n \\ &= \sum_i X_iY_i - \frac{1}{n}\left(\sum_i X_iY_j + \sum_{i\neq j} X_iY_j\right) = \frac{n-1}{n}\sum_i X_iY_i - \frac{1}{n}\sum_{i\neq j} X_iY_j\,. \end{aligned} \tag{5.11}$$

Satz 5.9

Die Stichprobenkovarianz K_n ist ein erwartungstreuer Schätzer für $\mathrm{Kov}(X_1,Y_1)$, das heißt es gilt $\mathbb{E}(K_n) = \mathrm{Kov}(X_1,Y_1)$.

Beweis:

Mittels Erwartungsbildung in Identität (5.11) und unter Ausnutzung der Linearität des Erwartungswerts ergibt sich

[26] Bei den Ausführungen zur Schätzung der Kovarianz orientierte ich mich an [6].

$$(n-1)\mathbb{E}(K_n) = \frac{n-1}{n}\mathbb{E}\left(\sum_i X_i Y_i\right) - \frac{1}{n}\mathbb{E}\left(\sum_{i\neq j} X_i Y_j\right).$$

Mit der Additionsregel für den Erwartungswert (Bemerkung 1.32) und aufgrund der identischen Verteiltheit der Zufallsvariablen folgt

$$\mathbb{E}\left(\sum_i X_i Y_i\right) = n \cdot \mathbb{E}(X_1 Y_1).$$

Um den Erwartungswert der zweiten Summe zu vereinfachen, bedenken wir, dass diese Summe aus $n^2 - n$ Summanden besteht und dass die Zufallsvariablen X_i und X_j für $i \neq j$ unabhängig sind. Die Additions- und Multiplikationsregel (Satz 1.33) für den Erwartungswert und die identische Verteiltheit liefern schließlich

$$\mathbb{E}\left(\sum_{i\neq j} X_i Y_j\right) = (n^2 - n) \cdot \mathbb{E}(X_1) \cdot \mathbb{E}(Y_1).$$

Insgesamt erhalten wir damit

$$(n-1)\mathbb{E}(K_n) = (n-1)\big(\mathbb{E}(X_1 Y_1) - \mathbb{E}(X_1) \cdot \mathbb{E}(Y_1)\big) = (n-1)\mathrm{Kov}(X_1, Y_1).$$

Division durch $n-1$ liefert schließlich die Behauptung. ■

Satz 5.10

Die Stichprobenkovarianz K_n ist ein stark konsistenter Schätzer für $\mathrm{Kov}(X_1, Y_1)$, das heißt es gilt $K_n \xrightarrow{\text{f.s.}} \mathrm{Kov}(X_1, Y_1)$.

Beweis:

Mit (5.11) erhalten wir unmittelbar die Darstellung

$$K_n = \frac{1}{n-1}\sum_i X_i Y_i - \frac{n}{n-1}\,\overline{X}_n \overline{Y}_n.$$

Da nach dem starken Gesetz großer Zahlen $\overline{X}_n \xrightarrow{\text{f.s.}} \mathbb{E}(X_1)$ und $\overline{Y}_n \xrightarrow{\text{f.s.}} \mathbb{E}(Y_1)$ gelten, gilt fast sicher

$$\lim_{n\to\infty} \frac{n}{n-1}\overline{X}_n \overline{Y}_n = \mathbb{E}(X_1)\mathbb{E}(Y_1).$$

Wendet man das starke Gesetz großer Zahlen auf die Folge der unabhängigen Zufallsvariablen $X_i Y_i$ an, so folgt, dass

$$\lim_{n\to\infty} \frac{1}{n-1}\sum_i X_i Y_i = \mathbb{E}(X_1 Y_1)$$

fast sicher gilt. Insgesamt ergibt sich also

$$K_n \xrightarrow{\text{f.s.}} \mathbb{E}(X_1 Y_1) - \mathbb{E}(X_1)\mathbb{E}(Y_1) = \mathrm{Kov}(X_1, Y_1). \quad ■$$

Somit haben wir nun alle Bausteine beisammen und definieren für die näherungsweise Bestimmung des Integrals (5.4) einen modifizierten Monte-Carlo-Schätzer durch

$$\begin{aligned} \tilde{I}_n &:= \frac{b-a}{n}\sum_{i=1}^{n} g(X_i) - c^\star\big(h(X_i) - \mathbb{E}(h(X_1))\big) \\ &= (b-a)c^\star\mathbb{E}(h(X_1)) + \frac{b-a}{n}\sum_{i=1}^{n} g(X_i) - c^\star h(X_i). \end{aligned} \tag{5.12}$$

Dabei stellt $h : \mathbb{R} \to \mathbb{R}$ eine Funktion dar, für die $\mathbb{E}(h(X_1))$ und $\mathbb{V}(h(X_1))$ bekannt oder leicht zu bestimmen sind. Die Schwierigkeit dieser Methode liegt offensichtlich im Auffinden einer

solchen Funktion h, welche die beiden Bedingungen erfüllt und zu einer starken Korrelation zwischen $g(X_i)$ und $h(X_i)$ führt.

Beispiel 5.11

Wir wenden diese Methode auf das Integral (5.7) aus Beispiel 5.1 an und wählen $h : \mathbb{R} \to \mathbb{R}$ mit $h(x) := \exp(-x)$ als Hilfsfunktion. Ist X eine $\mathcal{U}[0,3]$-verteilte Zufallsvariable, so ergeben sich für die Kontrollvariable $h(X)$ der Erwartungswert zu

$$\mu_h := \mathbb{E}\big(h(X)\big) = \frac{1}{3}\int_0^3 \exp(-x)\,dx = \frac{1}{3}(1 - \exp(-3))$$

und die Varianz zu

$$\sigma_h^2 := \mathbb{V}\big(h(X)\big) = \frac{1}{3}\int_0^3 \exp(-2x)\,dx - \mu_h^2 = \frac{1}{18}(1 + 4\exp(-3) - 5\exp(-6)).$$

Eine Schätzung von $\mathrm{Kov}(g(X), h(X))$ mit A.23 ergab den Wert 0,0882924. Übernehmen wir den Schätzwert für σ^2 aus (5.8), so ergibt sich

$$1 - \mathrm{Kor}(g(X), h(X))^2 \approx 0{,}0276333.$$

Durch den Einsatz der Kontrollvariable ergibt sich also eine Varianzreduktion von ungefähr 97,2 Prozent. Für die Zufallsvariable $g(X) - c^\star(h(X) - \mu_h)$ mit $c^\star \approx 1{,}33917$ gilt damit

$$\mathbb{V}(g(X) - c^\star(h(X) - \mu_h)) \approx 0{,}00336017.$$

Folglich ist ein Stichprobenumfang von $n = 3\,024\,151$ nötig, damit $\mathbb{V}(\tilde{I}_n)$ den Wert $\varepsilon^2 = 10^{-8}$ unterschreitet (siehe A.23). Abbildung 5.2 zeigt anschaulich die Überlegenheit des Schätzers $\tilde{I}_n$ mit der Kontrollvariable gegenüber dem naiven Monte-Carlo-Schätzer I_n. ∆

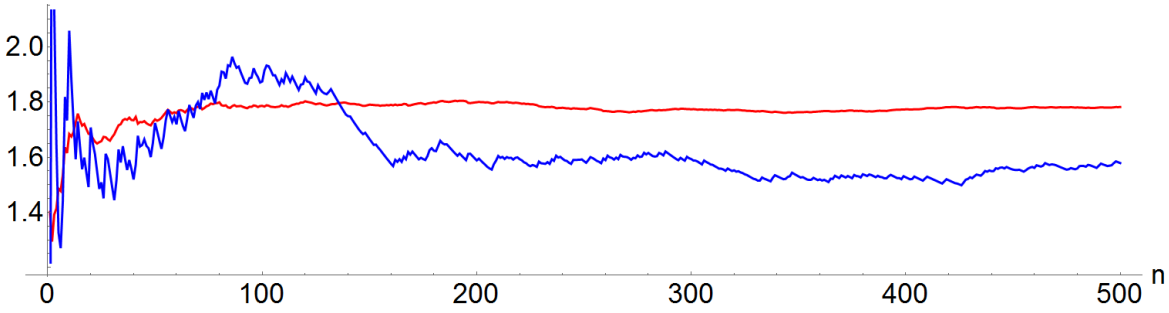

Abbildung 5.2 Kontrollvariable - Vergleich von I_n (blau) mit $\tilde{I}_n$ (rot) für wachsendes n

5.4 Stratified Sampling

Eine weitere Möglichkeit zur Varianzreduktion trägt den Namen Stratified Sampling. Sie basiert wie die zusammengesetzten Quadraturformeln auf der Idee der Zerlegung des Integrationsgebiets.

Sei $a = a_0 < a_1 < \cdots < a_k = b$ eine äquidistante Zerlegung von $[a, b]$, das heißt für $1 \le j \le k$ gelte

$$a_j - a_{j-1} = \frac{b-a}{k} =: h.$$

Aufgrund der Additivität des Integrals gilt

$$I = \int_a^b g(x)\,dx = \int_{a_0}^{a_1} g(x)\,dx + \int_{a_1}^{a_2} g(x)\,dx + \cdots + \int_{a_{k-1}}^{a_k} g(x). \tag{5.13}$$

Gemäß (3.1) gilt dann:

$$\int_{a_{j-1}}^{a_j} g(x)\,dx = (a_j - a_{j-1}) \cdot \mathbb{E}\big(g(X^{(j)})\big) \text{ mit } X^{(j)} \sim \mathcal{U}\big[a_{j-1}, a_j\big].$$

Wir wenden nun die naive Monte-Carlo-Integration auf jedes Teilintervall $\big[a_{j-1}, a_j\big]$ an und erhalten damit für unabhängige und $\mathcal{U}\big[a_{j-1}, a_j\big]$-verteilte Zufallsvariable $X_1^{(j)}, \dots, X_m^{(j)}$ die Näherung

$$\int_{a_{j-1}}^{a_j} g(x)\,dx \approx \frac{h}{m}\sum_{i=1}^{m} g\big(X_i^{(j)}\big).$$

Wir setzen $n = k \cdot m$. Seien $U_1, \dots, U_n$ unabhängige $\mathcal{U}[0,1]$-verteilte Zufallsvariablen. Um Unabhängigkeit über allen auftretenden Zufallsvariablen zu erhalten, setzten wir unter Berücksichtigung von Satz 1.23 und Bemerkung 1.26(1)

$$X_i^{(j)} \coloneqq a_{j-1} + h \cdot U_{i+m(j-1)}.$$

Wir erhalten damit insgesamt einen modifizierten Schätzer durch

$$I_{n,k} \coloneqq \frac{(b-a)}{n}\sum_{j=1}^{k}\sum_{i=1}^{n/k} g\big(X_i^{(j)}\big) \approx \int_a^b g(x)\,dx.$$

Gemäß Konstruktion ist klar, dass $I_{n,k}$ ein erwartungstreuer und stark konsistenter Schätzer für I ist. Wir sehen uns nun die Varianz dieses Schätzers an. Aufgrund der (jeweiligen) identischen Verteiltheit und der vorausgesetzten Unabhängigkeit ergibt sich mit den Eigenschaften der Varianz (Satz 1.36(c)) und der Additionsregel für die Varianz (Satz 1.37)

$$\begin{aligned}
\mathbb{V}(I_{n,k}) &= \frac{(b-a)^2}{n^2}\mathbb{V}\left(\sum_{j=1}^{k}\sum_{i=1}^{n/k} g\big(X_i^{(j)}\big)\right) \\
&= \frac{(b-a)^2}{n^2} \cdot \frac{n}{k} \cdot \mathbb{V}\left(\sum_{j=1}^{k} g\big(X_1^{(j)}\big)\right) \\
&= \frac{(b-a)^2}{n \cdot k}\sum_{j=1}^{k} \mathbb{V}\Big(g\big(X_1^{(j)}\big)\Big).
\end{aligned}$$

Bevor wir Aussagen über eine etwaige Varianzreduktion treffen können, benötigen wir den folgenden Hilfssatz:

Satz 5.12

Sei X eine Zufallsvariable mit $\mathbb{E}(X^2) < \infty$ und $c \in \mathbb{R}$. Die Abbildung $c \mapsto \mathbb{E}((X-c)^2)$ besitzt ein Minimum bei $c^\star \coloneqq \mathbb{E}X$.

Beweis:

Zunächst benutzen wir die Eigenschaften des Erwartungswerts und erhalten:

$$\mathbb{E}((X-c)^2) = \mathbb{E}(X^2 - 2cX + c^2) = \mathbb{E}(X^2) - 2c\mathbb{E}X + c^2.$$

Wir setzen $g(c) \coloneqq \mathbb{E}(X^2) - 2c\mathbb{E}X + c^2$. Dann ist $c^\star = \mathbb{E}X$ Nullstelle von $g'(c) = 2c - 2\mathbb{E}X$. Da $g''(c) = 2$ gilt, ist $c^\star$ eine Minimumstelle von g. ■

Satz 5.13

Die Varianz des stratifizierten Schätzers $I_{n,k}$ ist stets kleiner oder gleich der Varianz des naiven Schätzers I_n.

Beweis:
Für $X \sim \mathcal{U}[a,b]$ gilt:

$$\mathbb{V}\big(g(X)\big) = \frac{1}{b-a}\int_a^b (g(x)-\mathbb{E}X)^2\,dx = \frac{1}{b-a}\sum_{j=1}^{k}\int_{a_{j-1}}^{a_j}(g(x)-\mathbb{E}X)^2\,dx$$
$$= \frac{1}{hk}\sum_{j=1}^{k}\int_{a_{j-1}}^{a_j}(g(x)-\mathbb{E}X)^2\,dx\,.$$

Somit gilt nach (5.1) für die Varianz des Schätzers I_n:

$$\mathbb{V}(I_n) = \frac{(b-a)^2}{n\cdot hk}\sum_{j=1}^{k}\int_{a_{j-1}}^{a_j}(g(x)-\mathbb{E}X)^2\,dx.$$

Aufgrund von Satz 5.12 können wir folgern, dass für $1 \le j \le k$ gilt:

$$\mathbb{V}\Big(g\big(X_1^{(j)}\big)\Big) = \frac{1}{h}\int_{a_{j-1}}^{a_j}\big(g(x)-\mathbb{E}X_1^{(j)}\big)^2\,dx \le \frac{1}{h}\int_{a_{j-1}}^{a_j}(g(x)-\mathbb{E}X)^2\,dx.$$

Somit ist

$$\sum_{j=1}^{k}\mathbb{V}\Big(g\big(X_1^{(j)}\big)\Big) \le \frac{1}{h}\sum_{j=1}^{k}\int_{a_{j-1}}^{a_j}(g(x)-\mathbb{E}X)^2\,dx$$

und damit $\mathbb{V}\big(I_{n,k}\big) \le \mathbb{V}(I_n)$. ■

Bemerkung 5.14
Wir haben hier eine äquidistante Zerlegung und eine gleiche Anzahl von Stichprobenvariablen pro Teilintervall betrachtet, damit die Notation übersichtlich bleibt. Die Ausführungen können jedoch unmittelbar verallgemeinert werden: Die Grundidee, dass die Abweichungen bei lokalen Schätzungen kleiner sind, als jene bei globalen Schätzungen, ist natürlich unabhängig von der Zerlegung. Man kann also die Zerlegung so wählen, dass die Änderung von g auf einem Teilintervall $[a_{j-1}, a_j]$ gering ist. Auch die Wahl einer unterschiedlichen Anzahl von Stichprobenvariablen (Zufallszahlen) pro Teilintervall, kann die Schätzung verbessern. Man könnte zudem ein anderes Verfahren zur Varianzreduktion bei der Integration über ein Teilintervall $[a_{j-1}, a_j]$ einsetzen, um die Varianz weiter zu senken.

Beispiel 5.15
Wir wollen nun das Verhalten des naiven Schätzers I_n mit jenem des stratifizierten Schätzers $I_{n,k}$ vergleichen. Dazu betrachten wir wieder das Integral aus (5.7) und definieren deshalb zunächst den Integranden mit

```
g[x_] = Exp[-x^2];.
```

Dann legen wir die Intervallgrenzen des Integrationsgebiets [0,3] durch

```
a = 0.;b = 3.;
```

fest. Wir zerlegen das Integrationsgebiet äquidistant in $k = 5$ Teile und setzen dafür

```
k = 5;h = (b-a)/k;.
```

Außerdem definieren wir zwei leere Listen

```
ListeSS = {};ListeN = {};.
```

In den Listen werden die Schätzwerte von $I_{n,k}$ und I_n für wachsendes n gesammelt. Wir setzen nun

```
m = 1; l = 1000;
```

und benutzen eine `While`-Schleife der Form

```
While[m < l+1, Körper; m++];.
```

In den `Körper` schreiben wir zunächst

```
n = k*m;.
```

Für die Zufallszahlerzeugung nutzen wir den multiplikativen Kongruenzgenerator bei einem Startwert von Eins und geben daher

```
SeedRandom[1, Method → "Congruential"];
```

ein. Nun können wir den Schätzer $I_{n,k}$ durch den verschachtelten Ausdruck

```
Input = 6./n*Total[Table[Total[g[RandomReal[{a+(j-1)*h, a+j*h}, m]]], {j, 1, k}]];
```

realisieren. Der Ausdruck

```
Total[g[RandomReal[{a+(j-1)*h, a+j*h}, m]]
```

stellt eine Realisierung der Summe

$$\sum_{i=1}^{n/k} g\left(X_i^{(j)}\right)$$

dar. Darauf wird nun der Befehl `Table` angewandt. Damit ergibt sich eine Liste der Form

$$\left\{\sum_{i=1}^{n/k} g\left(X_i^{(1)}\right), \sum_{i=1}^{n/k} g\left(X_i^{(2)}\right), \ldots, \sum_{i=1}^{n/k} g\left(X_i^{(k)}\right)\right\}.$$

Wird auf diese Liste nun der Befehl `Total` angewendet, so ergibt sich eine Realisierung von

$$\sum_{j=1}^{k} \sum_{i=1}^{n/k} g\left(X_i^{(j)}\right).$$

Der Faktor Sechs ergibt sich aufgrund der Intervalllänge und weil wir die Symmetrie in (5.7) ausgenutzt haben. Schließlich muss noch durch n dividiert werden. Den errechneten Wert speichern wir in `ListeSS` und schreiben dafür

```
AppendTo[ListeSS, Input];.
```

Nun widmen wir uns der Realisation von I_n. Damit wir die selbe Stichprobe benutzen geben wir nocheinmal

```
SeedRandom[1, Method → "Congruential"];
```

ein. Eine Realisierung von I_n erhalten wir mit

```
AppendTo[ListeN, 6*Mean[g[RandomReal[{a,b}, n]]]],
```

wobei wir den errechneten Wert in `ListeN` speichern. Damit ist der `Körper` fertig.

Mit der von uns definierten `While`-Schleife ist es also möglich, den Verlauf der beiden Schätzer $I_{n,k}$ und I_n für wachsendes n aufzuzeichnen. Aufgrund der Definition von $I_{n,k}$ können wir die beiden Schätzer nicht für alle $n \in \mathbb{N}$ gleichzeitig realisieren. Wir vergleichen daher die Werte von $I_{n,k}$ und I_n für $n = k \cdot m$ mit $1 \leq m \leq l$.

Wir plotten das Ergebnis mit

```
ListLinePlot[{ListeSS, ListeN}, DataRange → {k, l*k},
             AxesLabel → {"n"}, LabelStyle → {Large, Black},
             PlotStyle → {{Red, Thick}, {Blue, Thick}}, AspectRatio → 1/4]
```

und erhalten nach der Auswertung als Ausgabe Abbildung 5.3.

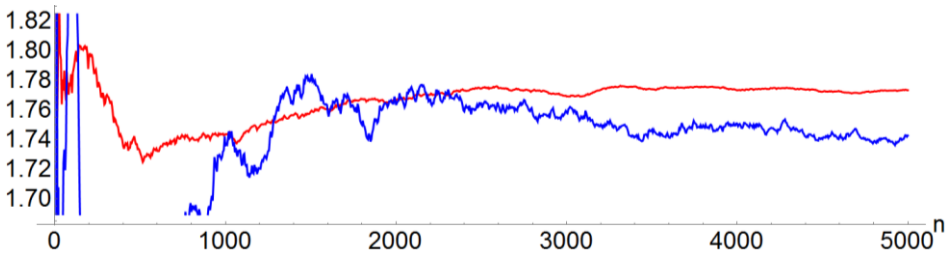

Abbildung 5.3 Stratified Sampling - Vergleich von I_n (blau) mit $I_{n,5}$ (rot) für wachsendes n

Der Schätzer $I_{n,5}$ zeigt bereits ein stabileres Verhalten als I_n. Der Effekt wird noch deutlicher wenn wir k vergrößern. Setzen wir im obigen Eingabe-Code `k = 50` und `l = 100` so ergibt sich nach der Auswertung als Ausgabe Abbildung 5.4. △

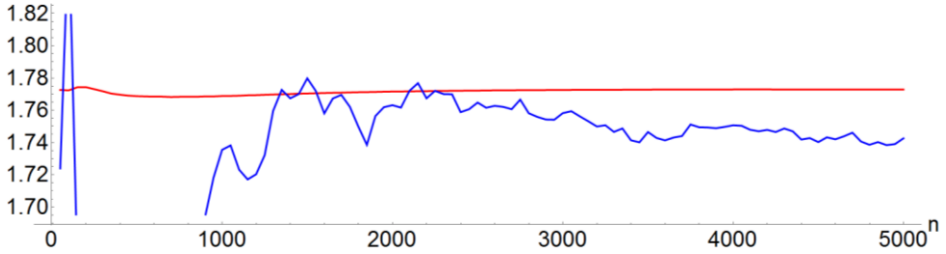

Abbildung 5.4 Stratified Sampling - Vergleich von I_n (blau) mit $I_{n,50}$ (rot) für wachsendes n

Bemerkung 5.16
In Analogie zur zusammengesetzten mehrdimensionalen Trapezregel (Abschnitt 4.2) müssen wir jedoch festhalten: Bei hochdimensionalen Integralen wird Stratified Sampling, selbst für eine grobe Zerlegung des Integrationsgebiets, zu aufwändig [29, Beispiel 5.14].

5.5 Quasi-Monte-Carlo-Integration

Ein anderer Zugang, um die Monte-Carlo-Integration zu beschleunigen, ist die Verwendung von Quasizufallszahlen statt Pseudozufallszahlen. Quasizufallszahlen sind (deterministische) Zahlenfolgen mit einer niedrigen Diskrepanz. Die Diskrepanz ist ein Maß für die Abweichung einer solchen Folge von einer Gleichverteilung [31].

Mit MATHEMATICA können Folgen mit niedriger Diskrepanz (z. B. Sobol-Folge) generiert werden. In Abbildung 5.5 ist der Unterschied zwischen Pseudozufallszahlen und Quasizufallszahlen deutlich ersichtlich: Mithilfe von Quasizufallszahlen wird das Einheitsquadrat gleichmäßig mit Punkten ausgefüllt, während der Einsatz von Pseudozufallszahlen (aufgrund der Unabhängigkeit) zu Löchern und Verklumpungen führt [8]. Mit A.26 (`Manipulate`-Umgebung) kann die schrittweise „Entstehung“ der Punkte beobachtet und verglichen werden.

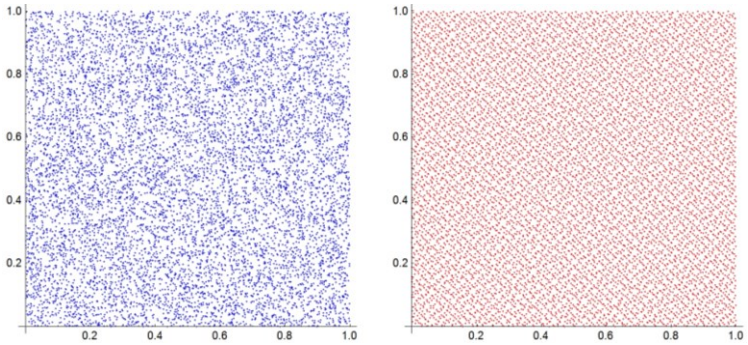

Abbildung 5.5 Vergleich zwischen Pseudozufallszahlen (links, Mersenne-Twister) und Quasizufallszahlen (rechts, Sobol-Folge) für $n = 10\,000$ Punkte

Der Integrationsfehler bei der Quasi-Monte-Carlo-Integration ist von deterministischer Natur und kann durch die Koksma-Hlawka-Ungleichung abgeschätzt werden [8, Theorem 5.1]. Konkret wird dabei der Fehler durch das Produkt der (Stern-)Diskrepanz mit der Variation des Integranden abgeschätzt. Damit ergibt sich eine Konvergenzordnung von $\mathcal{O}(n^{-1}(\log n)^m)$ bei der Integration einer Funktion $g : \mathbb{R}^m \to \mathbb{R}$ mit beschränkter Variation, sofern eine Folge mit niedriger Diskrepanz verwendet wird [8].

Die Monte-Carlo-Integration lieferte hingegen gemäß (5.1) nur eine Konvergenzordnung von $\mathcal{O}(n^{-1/2})$, ist jedoch unabhängig von der Dimensionszahl m. Der Fehler bei der Quasi-Monte-Carlo-Integration kann also für sehr hohe Dimensionszahlen sehr groß werden. In [1, Abschnitt 53] wird deshalb für sehr hochdimensionale Integrale eine Kombination der beiden Verfahren empfohlen.

Zu erwähnen ist auch, dass der Fehler bei der Monte-Carlo-Integration zwar von stochastischer Natur ist, aber durch eine Gleichung gegeben ist. Zudem kann dieser Fehler leicht empirisch geschätzt werden. Die Koksma-Hlawka-Ungleichung ist dagegen eine sehr grobe Abschätzung und beinhaltet Größen, deren Ermittlung nicht so einfach ist [8].

Für detailliertere Informationen zur Quasi-Monte-Carlo-Integration verweise ich auf [8] und [31]. Eine kompakte Vorstellung dieser Methode ist in [1] zu finden.

Die Implementierung der Quasi-Monte-Carlo-Integration erfolgt analog wie bei der (naiven) Monte-Carlo-Integration. Um (eindimensionale) Quasizufallszahlen zu benutzen, müssen wir einfach eine andere Methode bei der Eingabe von `SeedRandom` verwenden:

```
SeedRandom[1,Method→{"MKL",Method→{"Sobol","Dimension"→1}}].
```

Zu beachten ist dabei, dass für Integrale von höherer Dimension der Wert von `"Dimension"` entsprechend angepasst werden muss.

Beispiel 5.17

Wir sehen uns nun an, wie sich der Monte-Carlo-Schätzer unter der Verwendung von Quasizufallszahlen anstatt Pseudozufallszahlen verhält. Wir nähern dazu das Integral (5.7) durch

$$\frac{6}{n} \cdot \sum_{i=1}^{n} \exp(-x_i)^2$$

an, wobei wir einmal auf [0,3] gleichverteilte Zufallszahlen $x_1, \ldots, x_n$ benutzen (naive Monte-Carlo-Integration) und anschließend die x_i durch Quasizufallszahlen aus [0,3] ersetzen (Quasi-Monte-Carlo-Integration). Abbildung 5.6 zeigt den verschiedenen Simulationsverlauf der beiden Methoden. Die Überlegenheit des Quasi-Monte-Carlo-Schätzers gegenüber dem naiven Schätzer ist dabei deutlich zu erkennen.

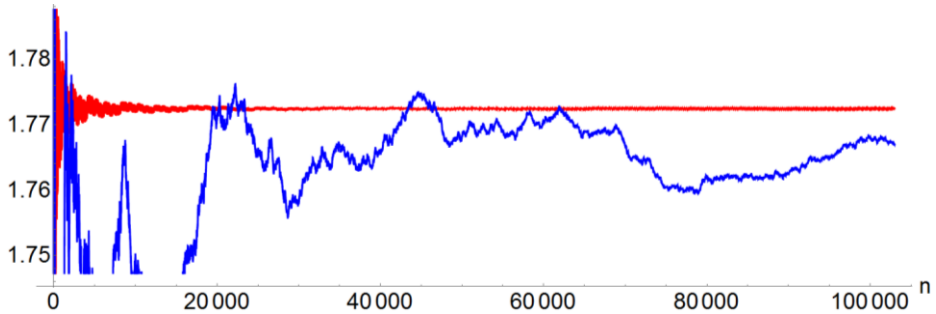

Abbildung 5.6 Vergleich zwischen (naiver) Monte-Carlo-Integration (blau) und Quasi-Monte-Carlo-Integration (rot) für wachsendes n

Bei diesem Integrationsproblem konnte Importance Sampling besonders überzeugen. Wie Abbildung 5.7 zeigt, ist der Importance-Sampling-Schätzer für moderaten Stichprobenumfang gegenüber dem Quasi-Monte-Carlo-Schätzer im Vorteil.

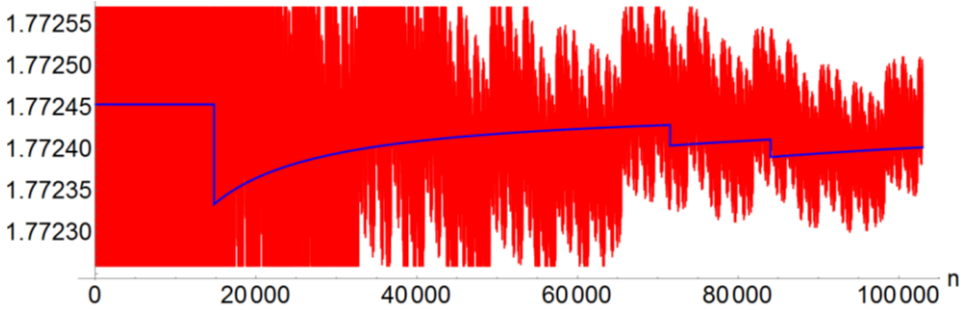

Abbildung 5.7 Vergleich zwischen Monte-Carlo-Integration mit Importance Sampling (blau) und Quasi-Monte-Carlo-Integration (rot) für wachsendes n

Zum Abschluss dieses Beispiels möchte ich noch kurz auf die in Abbildung 5.7 ersichtlichen „Stufen" beim Verlauf des Importance-Sampling-Schätzers eingehen. Eine solche „Stufe" tritt aufgrund von (5.6) genau dann auf, wenn eine Zufallszahl erzeugt wird, die außerhalb des Integrationsgebiets $[-3,3]$ liegt. Wie man leicht mit MATHEMATICA bestimmt (A.27), gilt:

$$\mathbb{P}(|X| > 3) = 1 - \mathbb{P}(|X| \leq 3) \approx 2{,}2 \cdot 10^{-5} \text{ für } X \sim \mathcal{N}(0; 1/2).$$

Die Anzahl der Stufen dividiert durch den Stichprobenumfang ist dann eine empirische Näherung für diese Wahrscheinlichkeit (hier $3/103\,000 \approx 2{,}9 \cdot 10^{-5}$). ∆

5.6 Monte-Carlo-Integration mit NIntegrate

Eine naive Monte-Carlo-Integration oder eine Quasi-Monte-Carlo-Integration können in MATHEMATICA direkt über den Befehl `NIntegrate` durchgeführt werden. Dazu muss nur als Zusatzbefehl

```
Method → "MonteCarlo"
```

beziehungsweise `"QuasiMonteCarlo"` eingegeben werden. Die Integration wird abgebrochen, sobald der Näherungswert auf zwei Stellen exakt ist. Maximal dürfen dafür jedoch nur 50 000 Punkte benötigt werden. Die Zielgenauigkeit kann durch `PrecisionGoal` vergrößert werden. Mit der Zusatzspezifikation `"MaxPoints"` kann zudem eine andere Beschränkung für die erlaubte Maximalanzahl an Punkten vergeben werden. Ebenso lässt sich Stratified Sampling direkt mithilfe von `"Partitioning"` umsetzen.

Speziell für die implementierte naive Monte-Carlo-Integration gilt: Um Reproduzierbarkeit zu gewährleisten, kann durch `"RandomSeed"` ein Startwert festgelegt werden. Leider wird für die Zufallszahlerzeugung stets der Generator „`ExtendedCA`", dessen Eigenschaften nicht bekannt sind (Abschnitt 2.4), verwendet. Dies kann auch nicht durch eine neue Festlegung des Standardgenerators verändert werden. Problematisch ist, dass das Abbruchkriterium auf einer laufenden Schätzung der Stichprobenvarianz beruht (vgl. Abschnitt 3.4). Damit ist der berechnete Näherungswert für das Integral nur bedingt aussagekräftig. Unpraktisch ist zudem, dass im „Erfolgsfall", also wenn die Zielgenauigkeit innerhalb der erlaubten Maximalpunktanzahl erreicht wird, nicht unmittelbar angezeigt wird, wie groß der Fehler tatsächlich war und

wie viele Punkte[27] überprüft wurden. Diese Informationen werden nur im Falle des „Scheiterns“ direkt angezeigt.

Mit dem undokumentierten Zusatzbefehl (innerhalb von NIntegrate)

```
IntegrationMonitor :→ Print
```

kann die Vorgehensweise beim implementierten Algorithmus nachvollzogen werden. Exemplarisch betrachten wir dazu wieder das Integral aus (5.7). Wir geben nun

```
INMC = Reap[NIntegrate[2*Exp[-x^2],{x,0,3},PrecisionGoal→6,
   Method→{"MonteCarlo","RandomSeed"→1,"MaxPoints"→400},
   IntegrationMonitor :→ Print,EvaluationMonitor :→ Sow[x]]];
```
(5.14)

ein. MATHEMATICA liefert dann die Meldung:

```
The integral failed to converge after 500 integrand evaluations. NIntegrate obtained
1.8527298488757453` and 0.09608751459989977` for the integral and error estimates.
```

Wir bekommen damit nicht nur den errechneten Näherungswert, sondern auch die Anzahl der untersuchten Punkte und einen Schätzwert für den Fehler. Diese Meldung erhält man immer, wenn man die Zielgenauigkeit (PrecisionGoal) unverhältnismäßig hoch wählt.

Interessanter sind die Ergebnisse, die der IntegrationMonitor liefert. Zunächst führen wir jedoch noch eine naive Monte-Carlo-Integration (wie gewohnt) durch und geben

```
SeedRandom[1];g[x_] = Exp[-x^2];
ListeZZ1 = RandomReal[{0,3},500];
6*Mean[g[ListeZZ1]]
6*Sqrt[Variance[g[ListeZZ1]]/500]
```
(5.15)

ein. Wir erhalten als Näherungswert für das gesuchte Integral damit

```
1.852729848875747`
```

und für den Fehler

```
0.09618374648641148`.
```

Der Unterschied zu den mit (5.14) erhaltenen Werten ist zwar gering, aber dennoch vorhanden. Die Begründung für den Unterschied liefert ein Blick auf die Ausgabe des IntegrationMonitor: MATHEMATICA führt zunächst die Substitution $x = 3t$ durch. Es wird also das Integral

$$6\int_0^1 \exp(-9t^2)\,dt$$

mithilfe der naiven Monte-Carlo-Integration berechnet. Die Vorgangsweise erfolgt schrittweise: Zunächst werden 100 auf [0,1] gleichverteilte Zufallszahlen erzeugt und dann wird das arithmetische Mittel der Funktionswerte an diesen Stellen bestimmt. Ebenso wird der Fehler bestimmt[28] und geprüft, ob die Zielgenauigkeit (oder die maximal erlaubte Punktanzahl) erreicht wurden. Im DOCUMENTATION CENTER wird angeführt, dass nur der errechnete Näherungswert und der Fehler behalten werden. Nun folgt eine Wiederholung dieses Vorgangs, wobei sich der neue Näherungswert (bzw. Fehler) aus dem aktuellen Ergebnis und dem vorherigen Ergebnis zusammensetzt. Wir setzen diese Vorgangsweise mit

```
SeedRandom[1];s[t_] = 6*Exp[-9t^2];
ListeZZH = Table[RandomReal[{0,1},100],{5}];
Sum[Mean[s[ListeZZH[[i]]]],{i,1,5}]/5
```
(5.16)

[27] Dies erfordert eine Zusatzeingabe.

[28] Im DOCUMENTATION CENTER (unter NIntegrate Integration Rules: "MonteCarloRule") wird angegeben, dass der Fehler gemäß (5.1) errechnet wird. Numerisch ergibt sich dennoch eine (kleine) Abweichung von der von uns üblicherweise gewählten Umsetzung, was auf eine andere Implementierung dieser Formel schließen lässt (A.28).

um und erhalten nach der Auswertung den Näherungswert

```
1.8527298488757453`.
```

Die Ergebnisse von (5.14) und (5.16) stimmen also überein.

Durch die Benutzung von `Reap` in (5.14) konnten wir aufzeichnen, welche Zufallszahlen `NIntegrate` benutzte. Die Liste dieser Zahlen kann mit `INMC[[2,1]]` eingesehen werden. Geben wir

```
3*Flatten[ListeZZH]==INMC[[2,1]]
```

ein, so erhalten wir als Ausgabe `True`. Das heißt (5.16) verwendete die selben Zufallszahlen wie (5.14). Für

```
ListeZZ1==INMC[[2,1]]
```

erhalten wir hingegen `False`. Die Differenz

```
ListeZZ1 - INMC[[2,1]]
```

zeigt, dass sich Abweichungen der Zufallszahlen aus (5.15) zu denen aus (5.14) in der Größenordnung von 10^{-16} ergeben. Dies könnte auf Rundungsfehler zurückzuführen sein. Somit ist klar, warum sich (geringfügig) andere Näherungswerte ergaben. In A.28 ist ein weiterer Code-Abschnitt, mit dem die rekursive Vorgangsweise in (5.14) auch hinsichtlich der Fehlerabschätzung untersucht werden kann, zu finden.

Zur implementierten Quasi-Monte-Carlo-Integration werden im DOCUMENTATION CENTER keine genauen Informationen angegeben. Sie basiere auf dem „Halton–Hammersley–Wozniakowski Algorithmus". Zum Aufbau dieses Algorithmus konnte ich im Rahmen meiner Recherchen jedoch keine Details finden. Geben wir

```
IQMC = Reap[NIntegrate[2*Exp[-x^2],{x,0,3},PrecisionGoal → 6,
   Method→{"QuasiMonteCarlo","RandomSeed"→1,"MaxPoints"→500},
   IntegrationMonitor:→ Print,EvaluationMonitor:→ Sow[x]]];
```
(5.17)

ein, so erhalten wir die Meldung:

```
The integral failed to converge after 500 integrand evaluations. NIntegrate obtained
1.7780462194268185` and 0.1047887612010923` for the integral and error estimates.
```

Dabei fällt besonders die sehr grobe Fehlerabschätzung auf. Mit (5.14) erhalten wir auch mit anderen Startwerten stets einen kleineren Fehler. Der mit (5.17) erhaltene Fehler widerspricht eigentlich der schnelleren Konvergenzgeschwindigkeit der Quasi-Monte-Carlo-Integration. Geben wir nun

```
SeedRandom[0,Method→{"MKL",Method→"Sobol"}];
ListeZZ2=RandomReal[{0,3},500];
ListeZZ2==IQMC[[2,1]]
```

ein, so erhalten wir `True` als Ausgabe. Folglich benutzt (5.17) die Zahlen der implementierten Sobol-Folge. Berechnen wir nun

```
6*Mean[g[ListeZZ2]]
```
(5.18)

so erhalten wir den Näherungswert

```
1.7755311082188934`.
```

Der doch etwas größere Unterschied zu dem mit (5.17) erhaltenen Wert lässt auf eine völlig andere Implementierung schließen. Obwohl de facto die selben Stützstellen benutzt wurden, lieferte (5.17) eine schlechtere Näherung als (5.18).

Eine besondere Form der stratifizierten (naiven) Monte-Carlo-Integration kann mit der Methode `"AdaptiveMonteCarlo"` verwendet werden. Dabei wird das Integrationsgebiet nicht in

gleich große Teile aufgeteilt, sondern in jenen Bereichen fortlaufend halbiert, in denen die geschätzte Varianz am größten ist. Diese Vorgehensweise kann sehr gut mit dem `IntegrationMonitor` verfolgt werden (A.28). Das Analogon für die Quasi-Monte-Carlo-Integration kann mithilfe von `"AdaptiveQuasiMonteCarlo"` benutzt werden.

Weitere Spezifkationen können mit `"MonteCarloRule"` festgelegt werden. Möchte man in (5.14) zum Beispiel 200 statt 100 Punkte pro Schritt verwenden, so verändert man die Methode zu

```
Method→{"MonteCarlo","RandomSeed"→1,"MaxPoints"→400,
        Method→{"MonteCarloRule","Points"→200}}.
```

Zusammenfassend ist zu sagen, dass die implementierten Algorithmen sehr gut geeignet sind, um schnell grobe Näherungswerte zu erhalten. Interessiert man sich jedoch für die Genauigkeit und den konkreten Verlauf der numerischen Integration, so ist eine eigene Umsetzung besser geeignet. Wie wir gesehen haben, kann die Monte-Carlo-Integration ohnehin mithilfe der Listenarithmetik von MATHEMATICA einfach realisiert werden. Durch einen selbständigen Aufbau kann man zudem besser auf das konkrete Integrationsproblem eingehen und weitere varianzreduzierende Techniken einsetzen.

6 Der beste Zielpunkt beim Darts

Im letzten Kapitel dieser Arbeit befassen wir uns mit Darts und der konkreten Frage, welchen Punkt eine Spielerin oder ein Spieler anvisieren sollte, um beim Spiel „501“ hohe Punktzahlen zu werfen. Dass für Dartprofis das Triple-20-Feld optimal ist, liegt auf der Hand. Gilt dies auch für Hobbyspielerinnen und -spieler?[29]

6.1 Das mathematische Modell

Für die Modellierung dieses Problems benötigen wir zunächst die Maße eines Dartboards. In Tabelle 6.1 sind die Standardmaße eines Bristle-Dartboards angeführt.

Dicke Double- und Triple-Ring (Innenmaß)	8,0 mm
Durchmesser des Bull's Eye (Innenmaß)	12,7 mm
Durchmesser des 25er-Ringes (Innenmaß)	31,8 mm
Entfernung vom äußeren Doubledraht zum Bull	170,0 mm
Entfernung vom äußeren Tripledraht zum Bull	107,0 mm
Gegenüberliegender äußerer Doubledraht	340,0 mm

Tabelle 6.1 Abmessungen Bristle-Dartboard [40]

Die Dicke der Drähte werde dabei vernachlässigt. Abbildung 6.1 zeigt die Skizze eines Dartboards mit einem Koordinatensystem für die nachfolgende Modellierung.

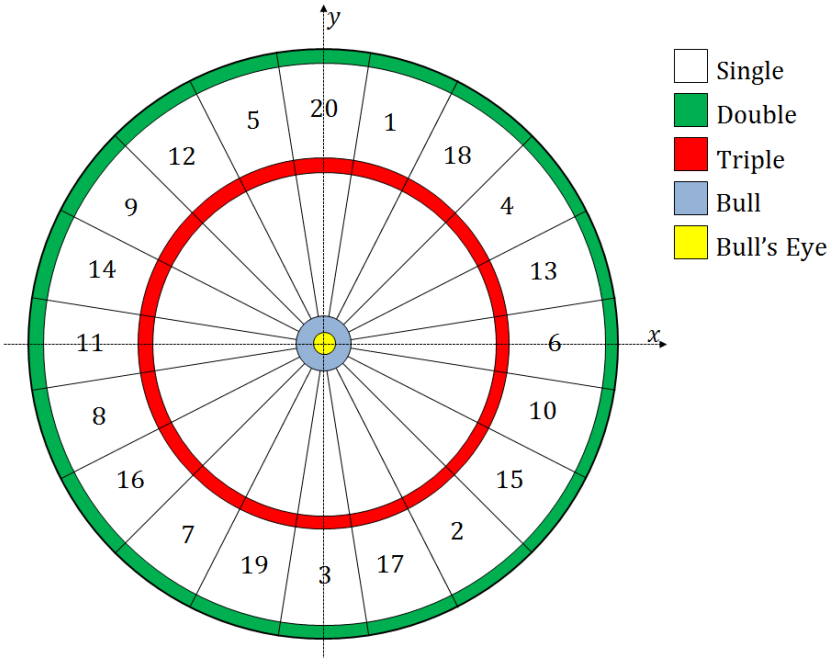

Abbildung 6.1 Skizze eines Dartboards

[29] Diese Aufgabenstellung wurde mir im Rahmen eines Seminars (250022, Sommersemester 2016) gestellt. Die Grundideen für das Modell und die Sammlung der später präsentierten empirischen Daten gehen auf die Arbeiten im Rahmen dieses Seminars zurück. Die Erhebung der empirischen Daten wurde gemeinsam mit meinen Kollegen Florian Schneeberger, Florian Supper und Florian Kaltseis durchgeführt. Florian Kaltseis leistete außerdem einen wichtigen Beitrag für eine effiziente Definition der Punktefunktion. Das mathematische Grundmodell und die Umsetzung mithilfe von MATHEMATICA wurden von mir entwickelt. Ein besonderer Dank gebührt Rainer Kaltseis. Seine Vorschläge und Ideen halfen mir bei der stetigen Verbesserung des Codes. Im Rahmen dieser Arbeit gelang es mir, das damalige Grundmodell wesentlich zu verbessern.

Definition der Punktefunktion

Wir werden im Folgenden eine Funktion g definieren, welche die geworfene Punktzahl (pro Wurf) beschreibt. Zunächst werden dazu Polarkoordinaten verwendet. Wir definieren eine Hilfsfunktion $h : (-\pi, \pi] \to \{1,2,\dots,20\}$ mit

$$h(\theta) \coloneqq \begin{cases} 11 & -\pi < \theta < -19\pi/20 \\ 8 & -19\pi/20 \le \theta < -17\pi/20 \\ 16 & -17\pi/20 \le \theta < -15\pi/20 \\ \vdots & \vdots \\ 14 & 17\pi/20 \le \theta < 19\pi/20 \\ 11 & 19\pi/20 \le \theta \le \pi. \end{cases}$$

Anschaulich liefert die Funktion h eine Aufteilung in Teilbereiche mit verschiedenen Punktzahlen. Die Funktion $p : [0,\infty) \times (-\pi, \pi] \to \mathbb{N}_0$ mit

$$p(r,\theta) \coloneqq \begin{cases} 50 & 0 \le r < 6{,}35 \\ 25 & 6{,}35 \le r < 15{,}9 \\ 1 \cdot h(\theta) & 15{,}9 \le r < 99 \\ 3 \cdot h(\theta) & 99 \le r < 107 \\ 1 \cdot h(\theta) & 107 \le r < 162 \\ 2 \cdot h(\theta) & 162 \le r < 170 \\ 0 & r \ge 170 \end{cases}$$

stellt schließlich die korrekte Zuordnung der geworfenen Punktzahl für Polarkoordinaten dar. Die Umrechnung von kartesische Koordinaten in Polarkoordinaten kann mit MATHEMATICA einfach durchgeführt werden. Wir definieren die Funktionen zur Umrechnung der Kürze halber direkt mithilfe von MATHEMATICA-Befehlen:

$$\theta : \mathbb{R}^2 \to (-\pi, \pi] \text{ mit } \theta(x,y) \coloneqq \texttt{ArcTan[x,y]},$$
$$r : \mathbb{R}^2 \to [0,\infty) \text{ mit } r(x,y) \coloneqq \texttt{Sqrt[x^2+y^2]}.$$

Insgesamt erhalten wir damit die Punktefunktion $g : \mathbb{R}^2 \to \mathbb{N}_0$ mit

$$g(x,y) \coloneqq p\big(r(x,y), \theta(x,y)\big).$$

Erwartungswert

Wir nehmen nun an, dass die Treffer der Dartspielerin oder des Dartspielers um den gewählten Zielpunkt (u,v) normalverteilt sind. Der Parameter σ beschreibe dabei die Streuung (Indikator für die Spielstärke) in x-Richtung und τ jene in y-Richtung in Millimeter. Wir gehen zudem davon aus, dass die Streuung in x-Richtung unabhängig von der Streuung in y-Richtung (und umgekehrt) ist.

Gemäß diesen Annahmen seien jetzt X eine $\mathcal{N}(u;\sigma^2)$-verteilte und Y eine $\mathcal{N}(v;\tau^2)$-verteilte Zufallsvariable. Setzen wir zudem voraus, dass X und Y unabhängig sind, so besitzt der Zufallsvektor $Z \coloneqq (X,Y)$ nach Bemerkung 1.26(2) die gemeinsame Dichte[30]

$$f(x,y) = \frac{1}{2\pi\sigma\tau} \exp\left(-\frac{(x-u)^2 + (y-v)^2}{2\sigma^2\tau^2}\right).$$

Die erwartete Punktzahl pro Wurf ergibt sich dann zu

$$\mathbb{E}\big(g(Z)\big) = \int_{\mathbb{R}^2} g(x,y) \cdot f(x,y)\, d(x,y).$$

[30] Das heißt Z besitzt eine bivariate Normalverteilung:

$$Z \sim \mathcal{N}\left(\begin{pmatrix} u \\ v \end{pmatrix}; \begin{pmatrix} \sigma^2 & 0 \\ 0 & \tau^2 \end{pmatrix}\right).$$

Für weiterführende Informationen zur multivariaten Normalverteilung verweise ich auf [7, S. 828].

Da g eine (hochgradig) unstetige Funktion ist und für die Lösung dieser Problemstellung ohnehin keine übermäßig hohe Genauigkeit benötigt wird, ermitteln wir $\mathbb{E}(g(Z))$ näherungsweise mithilfe der Monte-Carlo-Methode.

Um n Realisierungen von Z zu erzeugen, reicht es aufgrund der vorausgesetzten Unabhängigkeit von X und Y, jeweils n (unabhängige) Realisierungen von X und Y zu erzeugen, und diese zu 2-Tupeln zusammenzufassen (vgl. Abschnitt 2.2). Somit ergibt sich die Näherung

$$\mathbb{E}\big(g(Z)\big) \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i, y_i).$$

mit den Realisierungen $x_1, \ldots, x_n$ von X beziehungsweise $y_1, \ldots, y_n$ von Y.

Die Suche nach dem optimalen Zielpunkt

Wir wollen nun den besten Zielpunkt in Abhängigkeit von der Spielstärke (σ bzw. τ) ermitteln, also denjenigen Zielpunkt $(u_{\max}, v_{\max})$ finden, bei dem der Erwartungswert $\mathbb{E}(g(Z))$ maximal wird. Um diesen Punkt (näherungsweise) zu ermitteln, untersuchen wir einfach genügend viele Zielpunkte (u, v) hinsichtlich ihres Erwartungswerts und wählen dann den Punkt mit dem größten Erwartungswert.

Damit der Aufbau der Simulation nachvollziehbar bleibt, sollten nun die Funktionen h, p (verwende `Piecewise`) und g in MATHEMATICA definiert werden (A.29). Als Spielstärke (in Millimeter) setzen wir willkürlich

```
σ = 25.; τ = 30.;
```

und definieren die Parameter[31]

```
n = 10 000; a = 2.5;.
```

Wir „überziehen“ nun die Scheibe mit einem Punktgitter. Der Gitterabstand a (in Millimeter) stellt dann einen Indikator für die Genauigkeit des Verfahrens dar. Um dieses Gitter zu erzeugen, betrachten wir zunächst die endliche Folge

$$a_k := -170 + k \cdot a \text{ mit } 0 \le k \le \left\lfloor \frac{340}{a} \right\rfloor.$$

Die Bildmenge A dieser Folge erhalten wir mit

```
Folge = Table[k, {k, -170, 170, a}];.
```

Mit der Eingabe von

```
Gitter = Tuples[{Folge, Folge}];
```

erhalten wir die Produktmenge A^2, also ein Punktgitter in $[-170{,}170]^2$ mit Gitterabstand a.

Da der beste Zielpunkt sicher nicht außerhalb der Scheibe liegt[32], betrachten wir nur Zielpunkte, die auf der Scheibe liegen. Für diese „Filterung“ benutzen wir die Eingabe

```
Zielpunkte = Select[Gitter, Norm[#] < 170&];.
```

Wir setzen nun

```
ListeE = {}; ListeF = {}; j = 1;
```

und verwenden eine `While`-Schleife der Form

```
While[j < Length[Zielpunkte]+1, Körper; j++];.
```

[31] Die beiden Parameter sind entscheidend für die Genauigkeit und den Rechenaufwand des Verfahrens.
[32] Außerhalb der Scheibe ist die Punktefunktion stets gleich Null.

Im `Körper` legen wir zunächst den Zufallszahlgenerator und seine Initialisierung mit

```
SeedRandom[j,Method→"MersenneTwister"];
```

fest. Dann erzeugen wir je n Realisierungen von X und Y mit den Eingaben

```
ListeX=RandomVariate[NormalDistribution[Zielpunkte[[j,1]],σ],n];
ListeY=RandomVariate[NormalDistribution[Zielpunkte[[j,2]],τ],n];.
```

Die zugehörigen Funktionswerte werden durch

```
ListeG=Table[g[ListeX[[i]],ListeY[[i]]],{i,1,n}];
```

bestimmt. Mithilfe von

```
AppendTo[ListeE,Mean[ListeG]];
```

wird der Monte-Carlo-Schätzer realisiert. Dabei wird die Näherung für den Erwartungswert in `ListeE` gespeichert. Der Schätzfehler wird in `ListeF` gespeichert und durch

```
AppendTo[ListeF,StandardDeviation[ListeG]/Sqrt[n]]
```

bestimmt. Damit ist der `Körper` fertig. Die `While`-Schleife berechnet also für jeden Zielpunkt (näherungsweise) den Erwartungswert, sowie die Standardabweichung[33] des Schätzers.
Nun werden Erwartungswert und Fehler mit dem zugehörigen Zielpunkt verknüpft und die Zielpunkte nach der Größe ihres Erwartungswerts sortiert. Wir geben dazu

```
Daten=Sort[Transpose[{Zielpunkte,ListeE,ListeF}],#1[[2]]>#2[[2]]&];
```

ein. Beispielweise erhält man mit der Eingabe von `Daten[[1]]` die Koordinaten des Punkts $(u_{\max}, v_{\max})$, die berechnete Näherung für den Erwartungswert und die zugehörige Fehlerabschätzung für den besten Zielpunkt.

Interessant könnten auch Bereiche sein, in denen der Erwartungswert ebenfalls groß ist. Aus diesem Grund filtern wir aus der Liste `Daten` jene (guten) Zielpunkte heraus, deren Erwartungswert größer als 95 Prozent des maximalen Erwartungswerts ist. Wir verwenden dafür

```
GuteZielpunkte=Select[Daten,#[[2]]>0.95*Daten[[1,2]]&];.
```

In A.29 ist ein Eingabe-Code für die grafische Darstellung des Simulationsergebnisses angegeben. Mit ihm kann Abbildung 6.2 erzeugt werden. Dabei werden alle untersuchten Punkte angezeigt und alle guten Zielpunkte gelb hervorgehoben. Der beste Zielpunkt wird in Rot dargestellt.

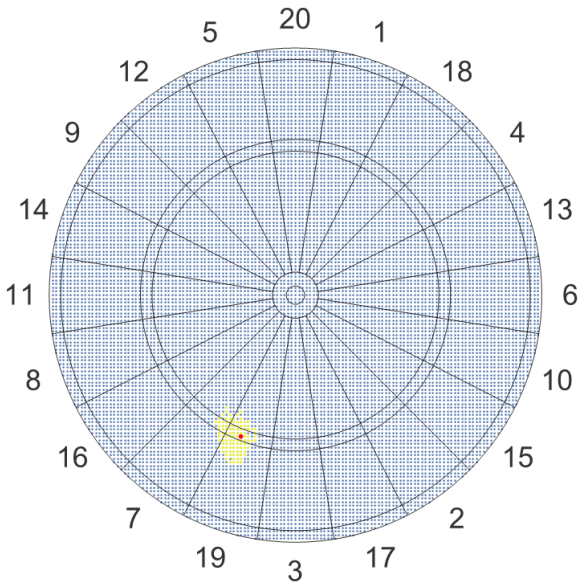

Abbildung 6.2 Darstellung des Simulationsergebnisses für $a = 2,5$ und $n = 10\,000$

[33] Für eine Beschleunigung kann die Bestimmung des Schätzfehlers aus der `While`-Schleife entfernt werden.

Wir sehen, dass eine Spielerin oder ein Spieler mit dieser Spielstärke nicht auf Triple-20 zielen, sondern Triple-19 (nahe bei Triple-7) anvisieren sollte, um im Schnitt eine höhere Punktzahl zu werfen. Der ermittelte beste Zielpunkt (für diese Spielstärke) liefert einen Erwartungswert von rund 15,7 Punkten pro Pfeil (verwende `Daten[[1,2]]`).

Wir nutzen die Fülle an gesammelten Daten, um auch die Erwartungswerte an den anderen Zielpunkten mithilfe des Befehls `ListDensityPlot` zu veranschaulichen (Abbildung 6.3). Der Code zur Erzeugung von Abbildung 6.3 ist ebenfalls in A.29 zu finden.

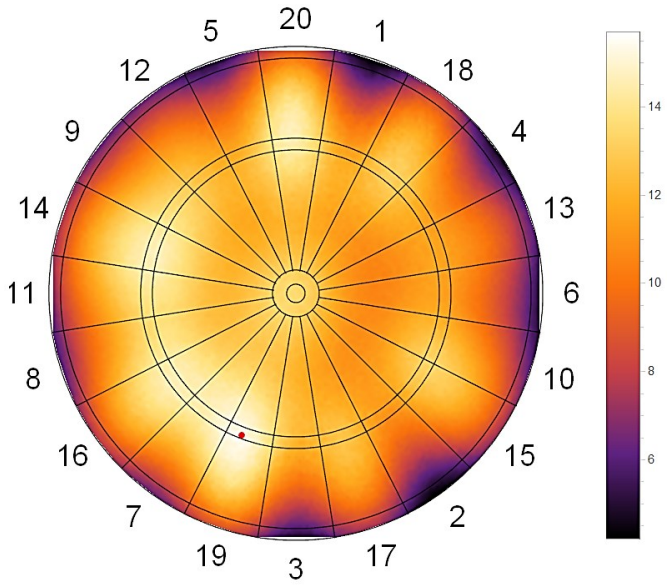

Abbildung 6.3 Größe des Erwartungswerts ($\sigma = 25; \tau = 30; a = 2,5$ und $n = 10\,000$)

Bemerkung zur Genauigkeit des Verfahrens

Je genauer dieses Verfahren arbeitet (a klein, n groß), desto mehr Daten werden gesammelt und desto länger ist zwangsläufig die Rechenzeit. Damit die (mühsam) gesammelten Daten später weiterverwendet werden können und nicht nach dem Schließen des MATHEMATICA-Notebooks verloren sind, sollten die Daten extern gespeichert werden. In A.29 ist ein Code angegeben, um dies umzusetzen.

Zur der hier getroffenen Wahl der Parameter n und a ist zu sagen, dass ein Stichprobenumfang von $n = 10\,000$ Punkten eine ausreichende Genauigkeit bietet, da die Schätzfehler zwischen 0,05 und 0,15 lagen (verwende `MinMax[ListeF]`). Mit dem hier gewählte Gitterabstand von $a = 2,5$ Millimeter wurden insgesamt 14 493 Zielpunkte untersucht (verwende den Befehl `Length[Zielpunkte]`), was wiederum zu einer langen Rechendauer führte. Ein Gitterabstand von $a = 5$ Millimeter wäre vermutlich für die Praxis ausreichend, um zufriedenstellende Ergebnisse zu errechnen, weil es Hobbyspielerinnen beziehungsweise -spielern ohnehin schwer fällt dermaßen genau zu zielen. Bei dieser Festlegung müssten dann nur noch 3613 Punkte untersucht werden.

Anmerkung

Eine ausführliche Behandlung des Dartsproblems ist zum Beispiel in [37] zu finden. Die Autoren gehen dabei von ähnlichen Annahmen aus und gelangen bei einer ihrer Berechnungen (mit vergleichbaren Werten für die Streuung) zum selben Ergebnis. Der von ihnen geschriebene Algorithmus (für R) kann unter [36] heruntergeladen werden. Zur Ermittlung der Spielstärke müssen bei ihm die geworfenen Punkte einer längeren Wurfserie angegeben werden. Zur Ermittlung des optimalen Zielpunkts wird der EM-Algorithmus benutzt. Neben der Dar-

stellung des besten Zielpunkts auf der Dartscheibe wird eine Heatmap, mit der (ähnlich wie in Abbildung 6.3) die Größe des Erwartungswert über den gesamten Bereich der Scheibe betrachtet werden kann, angezeigt.

6.2 Empirische Bestimmung der Spielstärke

Für die empirischen Bestimmung der Spielstärke (σ und τ) muss ein Zielpunkt gewählt werden (zum Beispiel: Mitte Bull's Eye). Die Versuchsperson versucht dann dieses Ziel zu treffen. Eine zweite Person zeichnet nach jedem Wurf die Koordinaten des Trefferpunktes auf. Dabei bietet es sich an, GEOGEBRA [23] zu verwenden: Man zeichnet dazu eine Dartscheibe (mit den korrekten Maßen) und setzt fortlaufend die Trefferpunkte per Mausklick. Die Koordinaten dieser Punkte können dann exportiert[34] und in Spaltenform als txt-Datei gespeichert werden.

In A.30 ist ein MATHEMATICA-Code für die weitere Analyse der erhobenen empirischen Daten angegeben: Der Code ermittelt den Stichprobenumfang (Anzahl der Würfe), die durchschnittlich erzielte Punktzahl pro Wurf, den im Mittel getroffenen Punkt (u, v), sowie die Standardabweichung in x- und y-Richtung (σ und τ). Die gesammelten Daten werden auf der Dartscheibe dargestellt. Zur Visualisierung des Ergebnisses wird der Punkt (u, v) angezeigt. Dieser Punkt bildet den Mittelpunkt einer Ellipse, deren Halbachsen durch σ und τ definiert werden.

Zudem kann mit A.30 grafisch überprüft werden, ob der Zufallsvektor $Z = (X, Y)$ in guter Näherung einer bivariate Normalverteilung (gemäß Fußnote 30) genügt beziehungsweise ob die Annahmen $X \sim \mathcal{N}(u; \sigma^2)$ und $Y \sim \mathcal{N}(v; \tau^2)$ näherungsweise erfüllt sind. Zusätzlich wird der Anderson-Darling-Test[35] benutzt, um die theoretische Annahme der bivariaten Normalverteilung zu prüfen. Die Ermittlung des Pearson'schen Korrelationskoeffizienten dient der Überprüfung der angenommenen Unabhängigkeit.

Exemplarisch möchte ich zwei empirische Analysen vorstellen: Im Rahmen von Analyse 1 wurde ein Match zwischen den beiden Profispielern Michael van Gerwen und Phil Taylor (siehe [10]) untersucht. Dabei wurden nur jene Würfe aufgezeichnet, die Triple-20 zum Ziel hatten. Da die Spielstärke beider Spieler auf dem selben hohen Niveau liegt, wurden die Daten der beiden Spieler zusammengefasst. Analyse 2 behandelt eigene Versuche (siehe Fußnote 29). Als Ziel wurde dabei das Bull's Eye ausgegeben. Tabelle 6.2 zeigt die mit A.30 errechneten Werte.

	Analyse 1	Analyse 2
Anzahl der Würfe n	392	371
Durschnittliche Punktzahl pro Wurf	38,8	12,5
Im Mittel getroffener Punkt (u, v) in Millimeter	$(-1{,}72; 100{,}93)$	$(0{,}37; 1{,}71)$
Standardabweichung (σ, τ) in Millimeter	$(8{,}54; 5{,}92)$	$(43{,}95; 52{,}75)$
Pearson'scher Korrelationskoeffizient	0,0016	$-0{,}0215$

Tabelle 6.2 Analyse der empirischen Daten

[34] Mit einem Tabellenkalkulationsprogramm können die empirischen Daten in eine für die Weiterverarbeitung mit MATHEMATICA günstige Form gebracht werden.

[35] Für weitere Informationen verweise ich auf [39, B.9].

Abbildung 6.4 veranschaulicht die in Tabelle 6.2 angeführten Ergebnisse und zeigt den deutlichen Unterschied zwischen einem Profispieler und einem Hobbyspieler. Bemerkenswert ist, dass der Punkt (u, v) in beiden Fällen dem Zielpunkt entspricht.

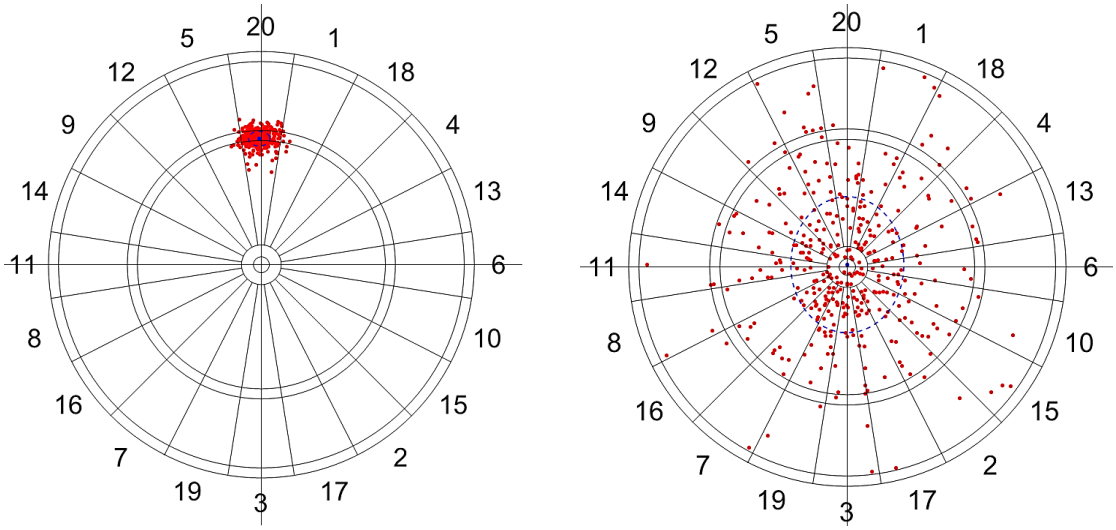

Abbildung 6.4 Verteilung der Treffer bei Analyse 1 (links) und Analyse 2 (rechts)

Die Abbildungen 6.5 und 6.6 zeigen die Ergebnisse der grafischen Überprüfung auf Normalverteilung.

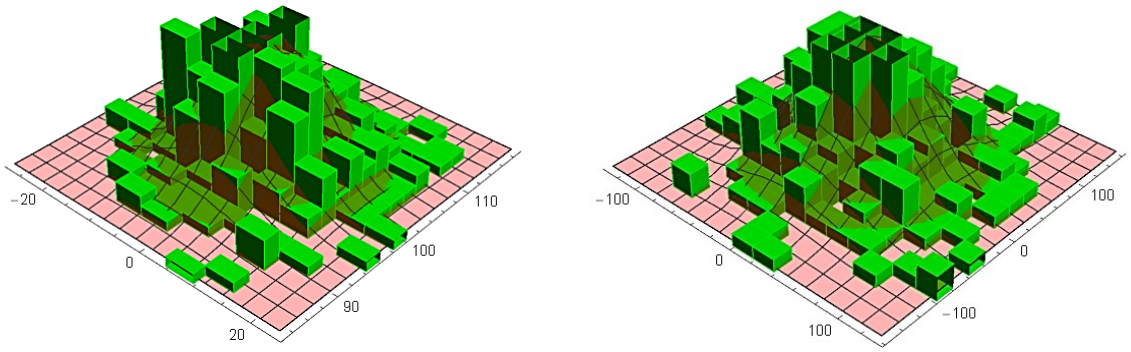

Abbildung 6.5 Grafische Überprüfung auf bivariate Normalverteilung (Netz, rot) mithilfe eines Histogramms (grün) für Analyse 1 (links) und Analyse 2 (rechts)

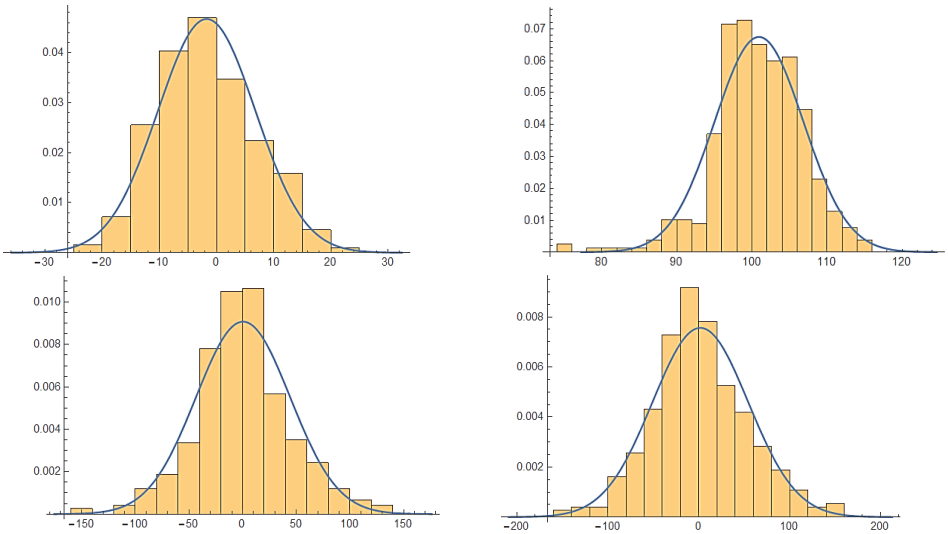

Abbildung 6.6 Grafische Überprüfung der Randverteilungen in x-Richtung (links oben: Analyse 1, links unten: Analyse 2) und in y-Richtung (rechts oben: Analyse 1, rechts unten: Analyse 2)

Neben der guten grafischen Übereinstimmung hat der Anderson-Darling-Test (bei einem Signifikanzniveau von 5 Prozent) in beiden Fällen ergeben, dass die theoretische Annahme der bivariaten Normalverteilung (gemäß Fußnote 30) nicht verworfen werden muss. Dass der Korrelationskoeffizient in beiden Fällen nahezu Null ergab, ist ein weiterer Beleg für die Vertretbarkeit der im Rahmen von 6.1 getätigten Annahmen.

Literaturverzeichnis

[1] Albrecher, H., Binder, A., Mayer, P. (2009): Einführung in die Finanzmathematik. Birkhäuser, Basel.

[2] Appell, J. (2009): Analysis in Beispielen und Gegenbeispielen. Eine Einführung in die Theorie reeller Funktionen. Springer, Berlin/Heidelberg.

[3] Arens, T., Busam, R., Hettlich, F., Karpfinger, C., Stachel, H. (2013): Grundwissen Mathematikstudium. Analysis und lineare Algebra mit Querverbindungen. Springer, Berlin/Heidelberg.

[4] Arens, T., Hettlich, F., Karpfinger, C., Kockelkorn, U., Lichtenegger, K., Stachel, H. (2015): Mathematik (3. Auflage). Springer Spektrum, Berlin/Heidelberg.

[5] Behrends, E. (2006): Fünf Minuten Mathematik. 100 Beiträge der Mathematik-Kolumne der Zeitung DIE WELT (1. Auflage). Vieweg, Wiesbaden.

[6] Bosch, K. (2005): Elementare Einführung in die angewandte Statistik. Mit Aufgaben und Lösungen (8. Auflage). Vieweg, Wiesbaden.

[7] Brokate, M., Henze, N., Hettlich, F., Meister, A., Schranz-Kirlinger, G., Sonar, T. (2016): Grundwissen Mathematikstudium. Höhere Analysis, Numerik und Stochastik. Springer, Berlin/Heidelberg.

[8] Caflisch, R. E. (1998): Monte Carlo and quasi-Monte Carlo methods. Acta Numerica, 7:1–49.

[9] Coveyou, R. R. (1969): Random number generation is too important to be left to chance. Studies in Applied Mathematics 3:70–111.

[10] Darts (2016): Darts-Masters 2016-van Gerwen v Taylor [SF]. https://www.youtube.com/watch?v=HSTc-4_QW_Q. Abgerufen am 04.03.2017.

[11] Eckhardt, R. (1987): Stan Ulam, John von Neumann, and the Monte Carlo Method. Los Alamos Science, (15):131–137.

[12] Engeln-Müllges, G., Niederdrenk, K., Wodicka, R. (2005): Numerik-Algorithmen. Verfahren, Beispiele, Anwendungen (9. Auflage). Springer, Berlin/Heidelberg.

[13] Falk, M., Hain, J., Marohn, F., Fischer, H., Michel, R. (2014): Statistik in Theorie und Praxis. Mit Anwendungen in R. Springer, Berlin/Heidelberg.

[14] Forster, O. (2015): Algorithmische Zahlentheorie (2. Auflage). Springer, Wiesbaden.

[15] Friedrich, H., Pietschmann, F. (2010): Numerische Methoden. Ein Lehr- und Übungsbuch. Walter de Gruyter, Berlin/New York.

[16] Georgii, H.-O. (2009): Stochastik. Einführung in die Wahrscheinlichkeitstheorie und Statistik (4. Auflage). Walter de Gruyter, Berlin.

[17] Hämmerlin, G., Hoffmann, K.-H. (1994): Numerische Mathematik (4. Auflage). Springer, Berlin/Heidelberg.

[18] Henze, N., Last, G. (2004): Mathematik für Wirtschaftsingenieure und für naturwissenschaftlich-technische Studiengänge Band 2 (1. Auflage). Vieweg, Wiesbaden.

[19] Henze, N. (2013): Stochastik für Einsteiger. Eine Einführung in die faszinierende Welt des Zufalls (10. Auflage). Springer, Wiesbaden.

[20] Hesse, C. (2003): Angewandte Wahrscheinlichkeitstheorie. Eine fundierte Einführung mit über 500 realitätsnahen Beispielen und Aufgaben (1. Auflage). Vieweg, Braunschweig/Wiesbaden.

[21] Heuser, H. (2008): Lehrbuch der Analysis Teil 2 (14. Auflage). Vieweg+Teubner, Wiesbaden.

[22] Heuser, H. (2009): Lehrbuch der Analysis Teil 1 (17. Auflage). Vieweg+Teubner, Wiesbaden.

[23] International GeoGebra Institute (2016): GeoGebra. Version: 5.0.269.0-3D, Linz.

[24] Knuth, D.E. (1981): The Art of Computer Programming. Volume 2: Seminumerical Algorithms (2. Auflage). Addison-Wesley, Reading, Mass.

[25] Kolonko, M. (2008): Stochastische Simulation. Grundlagen, Algorithmen und Anwendungen (1. Auflage). Vieweg+Teubner, Wiesbaden.

[26] Krengel, U. (2005): Einführung in die Wahrscheinlichkeitstheorie und Statistik (8. Auflage). Vieweg, Wiesbaden.

[27] Matsumoto, M., Nishimura, T. (1998): Mersenne twister. A 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Transactions on Modeling and Computer Simulations, 8(1):3–30.

[28] Metropolis, N. (1987): The Beginning of the Monte Carlo Method. Los Alamos Science, (15):125–130.

[29] Müller-Gronbach, T., Novak, E., Ritter, K. (2012): Monte Carlo-Algorithmen. Springer, Berlin/Heidelberg.

[30] Neunzert, H., Eschmann, W.G., Blickensdörfer-Ehlers, A., Schelkes, K. (1996): Analysis 1. Ein Lehr- und Arbeitsbuch für Studienanfänger (3. Auflage). Springer, Berlin /Heidelberg.

[31] Niederreiter, H. (1992): Random number generation and quasi-Monte Carlo methods. Society for Industrial and Applied Mathematics (SIAM 3600 Market Street Floor 6 Philadelphia PA 19104), Philadelphia, Pa.

[32] Plato, R. (2010): Numerische Mathematik kompakt. Grundlagenwissen für Studium und Praxis (4. Auflage). Vieweg+Teubner, Wiesbaden.

[33] Renouf, J. (2015): Klimawandel - Woher kommen die Zahlen? *Climate Change by Numbers.* BBC.

[34] Schulze-Pillot, R. (2008): Einführung in Algebra und Zahlentheorie (2. Auflage). Springer, Berlin/Heidelberg.

[35] Strohmaier, E., Dongarra, J., Simon, H., Meuer, M. (2016): List June 2016. https://www.top500.org/lists/2016/06/. Abgerufen am 06.10.2016.

[36] Tibshirani, R. (2009): A Statistician Plays Darts. http://finmath.stanford.edu/~ryantibs/darts/index.html. Abgerufen am 04.03.2017.

[37] Tibshirani, R. J., Price, A., Taylor, J. (2011): A statistician plays darts. Journal of the Royal Statistical Society: Series A, 174(1):213–226.

[38] Waldi, R. (2015): Statistische Datenanalyse. Grundlagen und Methoden für Physiker. Springer, Berlin/Heidelberg.

[39] Waldmann, K.-H., Helm, W.E. (2016): Simulation stochastischer Systeme. Eine anwendungsorientierte Einführung. Springer, Berlin/Heidelberg.

[40] Wiener Darts Verband (2015): Regelwerk - Allgemeine Regeln und Wettbewerbsregeln des WDV (DS4). Wiener Darts Verband. http://www.wdv-dart.at/_regelwerk/ds-allgemein.php. Abgerufen am 28.09.2016.

[41] Wolfram Research, I. (2015): Mathematica 10. Version: 10.2.0.0, Champaign, Ill.

Abbildungen

Tabellen

Anhang

Mathematica-Codes

A.1 Simulation aus Beispiel 2.10 (S. 27):

```
n = 2000;SeedRandom[1,Method → "Congruential"];
Liste = Table[Mean[RandomInteger[{1,6},m]],{m,1,n}];
G1 = ListLinePlot[Liste,PlotRange → {2.5,4.5},PlotStyle → Blue];
G2 = Plot[3.5,{x,0,n},PlotStyle → {Red,Thick}];
Show[G1,G2,AspectRatio → 1/3,AxesLabel → {"m","x̄ₘ",},LabelStyle → {Large,Black}]
```

A.2 Simulation aus Beispiel 2.11 (S. 28):

```
n = 500;m = 100 000;SeedRandom[1,Method → "Congruential"];
Liste = Table[j = RandomInteger[{1,6},n],{j,1,m}];
ListeM = Table[Mean[Liste[[j]]],{j,1,m}];
G1 = Histogram[ListeM,50,"PDF"];w = 21/6;s = Sqrt[35/(12n)];
G2 = Plot[PDF[NormalDistribution[w,s],x],{x,w-4s,w+4s},PlotStyle → {Black,Thick}];
Show[G1,G2,Axes → {True,False},AxesStyle → Thick,LabelStyle → {Large,Black},
  PlotRange → All,AspectRatio → 1/3]
```

A.3 Simulation aus Beispiel 3.2 (S. 32):

```
Manipulate[g[x_] = 4.*Sqrt[1-x^2];
  SeedRandom[1,Method → "Congruential"];
  Mean[g[RandomReal[{0,1},n]]],{n,1,5 000 000,1}]
```

A.4 Simulation aus Beispiel 3.5 (S. 34):

```
AbsoluteTiming[n = 10 000;
  g[x_,y_] = 1/(x+y)^2;
  SeedRandom[1,Method → "MersenneTwister"];
  Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];
  Mean[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]]]
```

A.5 Eingabe Code zur Erzeugung von Abbildung 3.2 (S. 35):

```
xlist = Table[10000*2^i,{i,0,13}];
ylist = {0.04,0.09,0.17,0.34,0.68,1.40,2.77,5.47,11.17,22.96,45.15,90.81,184.72,370.69};
ListeR = Transpose[{xlist,ylist}];G1 = ListPlot[ListeR,PlotRange → All,PlotStyle → Red];
line = Fit[ListeR,{1,x},x];G2 = Plot[line,{x,0,8.5*10^7},PlotStyle → {Green,Thick}];
Show[G1,G2,PlotRange → All,AxesLabel → {"Stichprobenumfang n","Rechenzeit[s]"},
  LabelStyle→ Directive[Black,Large],AspectRatio → 1/2]
```

A.6 Simulation aus Beispiel 3.7 (S. 37):

```
n = 100 000;
SeedRandom[1,Method → "MersenneTwister"];
ZVektoren = RandomVariate[UniformDistribution[{{-1,1},{-1,1}}],n];
Hit = Select[ZVektoren,(#[[1]])^2+(#[[2]])^2 ≤ 1&];
Miss = Complement[ZVektoren,Hit];
G1 = ListPlot[{Hit,Miss},PlotStyle → {Green,Red}];
```

```
G2 = Graphics[Circle[{0,0}]];
Show[G1,G2,AspectRatio → 1]
N[4/n Length[Hit]]
```

A.7 Simulation aus Beispiel 3.8 (S. 39):

```
AbsoluteTiming[n = 10 000;
  g[x_,y_] = 1/(x+y)^2;
  SeedRandom[1,Method→"MersenneTwister"];
  ZVektoren = RandomVariate[UniformDistribution[{{0,1},{1,2},{0,1}}],n];
  Hit = Select[ZVektoren,(#[[3]]) ≤ g[#[[1]],#[[2]]]&];
  N[1/n Length[Hit]]]
```

A.8 Eingabe Code zur Erzeugung von Abbildung 3.6 (S. 39):

```
Plot3D[1/(x+y)^2,{x,0,1},{y,1,2},PlotRange → All,Filling → Bottom,
  TicksStyle → Directive[Large,Bold]]
```

A.9 Simulation aus Beispiel 3.13 (S. 45):

```
AbsoluteTiming[n = 10 000;
  g[x_,y_] = 1/(x+y)^2;
  SeedRandom[1,Method → "MersenneTwister"];
  Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];
  Sqrt[Variance[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]]/n]]
```

A.10 Eingabe Code zur Erzeugung von Abbildung 3.8 (S. 45):

```
ListeG = Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}];
ListeV = Table[Variance[Take[ListeG,j]],{j,2,n}];
G1 = ListLinePlot[ListeV,PlotStyle → Blue];
G2 = Plot[11/108 − Log[4/3]^2,{x,0,n},PlotStyle → {Red,Thick}];
Show[G1,G2,PlotRange → {0.017,0.02},AspectRatio → 1/2,AxesLabel → {"n"},
  LabelStyle → {Medium,Black}]
```

A.11 Simulation aus Beispiel 3.14 (S. 48):

```
n = 506 718;
g[x_,y_] = 1/(x+y)^2;
ListeA = {};ListeB = {};
k = 100;j = 2;
While[j < k+2,SeedRandom[j,Method → "MersenneTwister"];
  Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];
  s = Mean[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]];
  AppendTo[ListeA,s − 2.575829*0.138177/Sqrt[n]];
  AppendTo[ListeB,s + 2.575829*0.138177/Sqrt[n]];j++];
G1 = Graphics[{Red,Thick,Line[{{0,Log[4/3]},{k+1,Log[4/3]}}]}];
G2 = Graphics[{Blue,Thick,Table[Line[{{l,ListeA[[l]]},{l,ListeB[[l]]}}],{l,1,k,1}]}];
Show[G1,G2,Axes → {False,True},AspectRatio → 1/3,LabelStyle → {Black,Large}]
```

A.12 Simulation aus Beispiel 3.15 (S. 50):

```
n = 10 000;
g[x_,y_] = 1/(x+y)^2;
ListeE = {};
```

```
k = 100 000;j = 1;
While[j < k+1,
  SeedRandom[j,Method → "MersenneTwister"];
  Liste = RandomVariate[UniformDistribution[{{0,1},{1,2}}],n];
  AppendTo[ListeE,Mean[Table[g[Liste[[i,1]],Liste[[i,2]]],{i,1,n}]]];j++];
w = Log[4/3];s = Sqrt[(11/108 − Log[4/3]^2)/n];
f[x_]= PDF[NormalDistribution[w,s],x];
m = Mean[ListeE];
G1 = Histogram[ListeE,50,"PDF"];
G2 = Plot[f[x],{x,w-4s,w+4s},PlotRange → All,PlotStyle → {Black,Thick}];
G3 = Graphics[{Thick,Red,Line[{{w,0},{w,f[w]}}],Blue,Line[{{m,0},{m,f[m]}}]}];
Show[G1,G2,G3,PlotRange → All,AspectRatio → 1/5,Axes → {True,False},
  LabelStyle → {Black,Large}]
```

A.13 Eingabe Code zur Erzeugung von Abbildung 3.12 (S. 52) als Fortsetzung von A.12:

```
Show[ListPlot[ListeE,PlotStyle → Blue],
  Graphics[{Red,Thick,Line[{{0,w},{k,w}}]}],
  AxesLabel → {"j"},LabelStyle → {Black,Medium}]
```

A.14 Simulation zu Beispiel 3.16 (S. 52):

```
g[x_] := Boole[Element[Rationalize[x,0],Rationals]];
n = 10000;
SeedRandom[1,Method → "Congruential"];
Liste = RandomReal[{0,1},n];
Mean[Table[g[Liste[[i]]],{i,1,n}]]
```

A.15 Simulation aus Beispiel 4.2 (S. 56):

```
n = 1000;
g[x_] = 4*Sqrt[1-x^2];
a = 0.;b = 1.;h = (b-a)/n;
h/2*(g[a]+g[b]+2*Sum[g[a+i*h],{i,1,n-1}])
```

A.16 Eingabe Code zur Erzeugung von Abbildung 4.2 (S. 57):

```
k = 100;g[x_] = 4*Sqrt[1-x^2];a = 0.;b = 1.;ListeM = {};ListeT = {};n = 1;
While[n < k+1,SeedRandom[1,Method → "Congruential"];
  AppendTo[ListeM,Mean[g[RandomReal[{0,1},n]]]];
  h = (b-a)/n;AppendTo[ListeT,h/2*(g[a]+g[b]+2*Sum[g[a+i*h],{i,1,n-1}])];n++];
G1 = ListLinePlot[{ListeM,ListeT},PlotRange → All,
  PlotStyle → {{Blue,Thick},{Black,Thick}}];
G2 = Graphics[{Thick,Red,Line[{{0,Pi},{k,Pi}}]}];
Show[G1,G2,AspectRatio → 1/4,AxesLabel → {"n"},LabelStyle → {Black,Large}]
```

A.17 Realisierung der Überlegung aus Exkurs 4.5 (S. 59):

```
p[x_,y_] = a1+a2*x+a3*y+a4*x*y;
m = {{1,a,c,a*c},{1,a,d,a*d},{1,b,c,b*c},{1,b,d,b*d}};
Lösungen = LinearSolve[m,{g1,g2,g3,g4}];
a1 = Lösungen[[1]];a2 = Lösungen[[2]];a3 = Lösungen[[3]];a4 = Lösungen[[4]];
Integrate[p[x,y],{x,a,b},{y,c,d}]
```

A.18 Simulation aus Beispiel 4.6 (S. 61):

```
n = 99;m = 99;
a = 0.;b = 1.;c = 1.;d = 2.;h1 = (b-a)/n;h2 = (d-c)/m;
g[x_,y_] = 1/(x+y)^2;HListe = {};j = 1;
While[j < m,s = c+j*h2;
  AppendTo[HListe,2*(g[a,s]+g[b,s])+4*Sum[g[a+i*h1,s],{i,1,n-1}]];j++];
(h1*h2)/4*(g[a,c]+g[b,c]+g[a,d]+g[b,d]
  +2*Sum[g[a+i*h1,c]+g[a+i*h1,d],{i,1,n-1}]
  +Total[HListe])
```

A.19 Simulation aus Beispiel 5.1 (S. 66):

```
n = 1 000 000;
h[x_] = Sqrt[Pi]/6.*Boole[-3. ≤ x ≤ 3.];
ListeS = {};j = 1;
While[j < 201,SeedRandom[j,Method → "Congruential"];
  HListe = RandomVariate[NormalDistribution[0.,Sqrt[1/2.]],n];
  AppendTo[ListeS,Variance[Table[h[HListe[[i]]],{i,1,n}]]];j++];
S = Mean[ListeS]
m = Ceiling[6^2*S/10^-8]

ListeI = {};k = 1;
While[k < 1001,SeedRandom[k,Method → "Congruential"];
  HHListe = RandomVariate[NormalDistribution[0.,Sqrt[1/2.]],m];
  AppendTo[ListeI,6.*Mean[Table[h[HHListe[[i]]],{i,1,m}]]];k++];
Mean[ListeI]
```

A.20 Eingabe Code zur Erzeugung von Abbildung 5.1 (S. 68):

```
n = 500;
g[x_]=Exp[-x^2];
h[x_] = Sqrt[Pi]/6.*Boole[-3. ≤ x ≤ 3.];
ListeI = {};ListeN = {};
SeedRandom[1,Method->"Congruential"];
ListeIZ = RandomVariate[NormalDistribution[0.,Sqrt[1/2.]],n];
ListeIF = Table[h[ListeIZ[[i]]],{i,1,n}];
SeedRandom[1,Method → "Congruential"];
ListeNFZ = g[RandomReal[{0,3},n]];j = 1;
While[j < n+1,
  AppendTo[ListeI,6.*Mean[Take[ListeIF,j]]];
  AppendTo[ListeN,6.*Mean[Take[ListeNFZ,j]]];j++];
ListLinePlot[{ListeI,ListeN},PlotStyle → {{Thick,Red},{Thick,Blue}},
  AxesLabel → {"n"},LabelStyle → {Large,Black},AspectRatio → 1/4]
```

A.21 Schätzung von σ^2 in Beispiel 5.1 (S. 66):

```
n = 1 000 000;g[x_] = Exp[-x^2];ListeS = {};j = 1;
While[j < 201,SeedRandom[j,Method → "Congruential"];
  AppendTo[ListeS,Variance[g[RandomReal[{0,3},n]]]];j++];
S = Mean[ListeS]
m = Ceiling[3^2*S/10^-8]
```

A.22 Schätzung von $\tilde{\sigma}^2$ in Beispiel 5.6 (S. 70):

```
n = 1 000 000;g[x_] = Exp[-x^2];h[x_] = (g[3.-x]+g[x])/2.;ListeS = {};j = 1;
While[j < 201,SeedRandom[j,Method → "Congruential"];
  AppendTo[ListeS,Variance[h[RandomReal[{0,3},n]]]];j++];
S = Mean[ListeS]
m = Ceiling[3.^2*S/10^-8]
```

A.23 Schätzung von $\mathbf{Kov}\big(g(X), h(X)\big)$ und weitere Berechnungen in Beispiel 5.11 (S. 73):

```
n = 1 000 000;
g[x_] = Exp[-x^2];h[x_] = Exp[-x];ListeK = {};j = 1;
While[j < 201,SeedRandom[j,Method → "Congruential"];
  ListeX = RandomReal[{0,3},n];
  ListeG = Table[g[ListeX[[i]]],{i,1,n}];
  ListeH = Table[h[ListeX[[i]]],{i,1,n}];
  AppendTo[ListeK,Covariance[ListeG,ListeH]];j++];
K = Mean[ListeK]
Vf = 0.1215984132617311`;
Vh = 1/18*(1+4*Exp[-3]-5*Exp[-6]);
r = 1-K^2/(Vh*Vf)
c = K/Vh
VY = Vf*r
m = Ceiling[3^2*VY/10^-8]
```

A.24 Eingabe-Code zur Erzeugung von Abbildung 5.2 (S. 73):

```
n = 500;g[x_] = Exp[-x^2];h[x_] = Exp[-x];
c = 1.3391671769324218`;f[x_] = g[x]-c*h[x];
SeedRandom[1,Method → "Congruential"];
ListeX = RandomReal[{0,3},n];
ListegX = Table[g[ListeX[[i]]],{i,1,n}];
ListefX = Table[f[ListeX[[i]]],{i,1,n}];
ListeK = {};ListeN = {};j = 1;
While[j < n+1,
  AppendTo[ListeN,6.*Mean[Take[ListegX,j]]];
  AppendTo[ListeK,2.*(c*(1.-Exp[-3])+3.*Mean[Take[ListefX,j]])];j++];
ListLinePlot[{ListeK,ListeN},PlotStyle → {{Thick,Red},{Thick,Blue}},
  AxesLabel → {"n"},LabelStyle → {Large,Black},AspectRatio → 1/4]
```

A.25 Simulation aus Beispiel 5.15 (S. 75):

```
g[x_] = Exp[-x^2];a = 0.;b = 3.;k = 5;h = (b-a)/k;
ListeSS = {};ListeN = {};m = 1;l = 1000;
While[m < l+1,n = k*m;
  SeedRandom[1,Method → "Congruential"];
  AppendTo[ListeSS,
  6./n*Total[Table[Total[g[RandomReal[{a+(j-1)*h,a+j*h},m]]],{j,1,k}]]];
  SeedRandom[1,Method → "Congruential"];
  AppendTo[ListeN,6*Mean[g[RandomReal[{a,b},n]]]];m++];
ListLinePlot[{ListeSS,ListeN},DataRange → {k,l*k},AxesLabel → {"n"},
  LabelStyle → {Large,Black},PlotStyle → {{Red,Thick},{Blue,Thick}},AspectRatio → 1/4]
```

A.26 Eingabe-Code zur Erzeugung von Abbildung 5.5 (S. 77):

```
SeedRandom[1,Method→"MersenneTwister"];
ListeM=RandomReal[1,{10000,2}];
SeedRandom[1,Method→{"MKL",Method→{"Sobol", "Dimension"→2}}];
ListeS=RandomReal[1,{10000,2}];
Manipulate[
  G1=ListPlot[Take[ListeM,n],AspectRatio→1,PlotLabel→"Mersenne Twister",
  LabelStyle→{Black},PlotStyle→Blue];
  G2=ListPlot[Take[ListeS,n],AspectRatio→1,PlotLabel→"Sobol-Folge",
  LabelStyle→{Black},PlotStyle->Red];
  GraphicsRow[{G1,G2},ImageSize→Large],{{n,1,"Punktanzahl n"},1,10000,1}]
```

A.27 Eingabe-Code zu Beispiel 5.17 (S. 78):

```
n=103000;g[x_]=Exp[-x^2];
SeedRandom[1,Method→"Congruential"];
ListeZZG=g[RandomReal[{0,3},n]];
ListeNMCI=6*Table[Mean[Take[ListeZZG,j]],{j,1,n}];
SeedRandom[1,Method→{"MKL",Method→{"Sobol","Dimension"→1}}];
ListeQZG=g[RandomReal[{0,3},n]];
ListeQMCI=6*Table[Mean[Take[ListeQZG,j]],{j,1,n}];
ListLinePlot[{ListeQMCI,ListeNMCI},AxesLabel→"n"},LabelStyle→{Large,Black},
  PlotStyle→{{Red,Thick},{Blue,Thick}},AspectRatio→1/3]

h[x_]=Sqrt[Pi]/6.*Boole[-3.≤x≤3.];
SeedRandom[1,Method→"Congruential"];
ListeZZ=RandomVariate[NormalDistribution[0.,Sqrt[1/2.]],n];
ListeZZH=Table[h[ListeZZ[[i]]],{i,1,n}];
ListeISMCI=6*Table[Mean[Take[ListeZZH,j]],{j,1,n}];
ListLinePlot[{ListeQMCI,ListeISMCI},AxesLabel→{"n"},LabelStyle→{Large,Black},
  PlotStyle→{{Red},{Blue,Thick}},AspectRatio→1/3]

1-Probability[-3≤x≤3,x\[Distributed]NormalDistribution[0.,Sqrt[1/2.]]]
Length[Select[ListeZZH,#==0&]]/Length[ListeZZH]//N
```

A.28 Ausführungen zu Abschnitt 5.6 (S. 79):

```
INMC=Reap[NIntegrate[2*Exp[-x^2],{x,0,3}, PrecisionGoal→6,
  Method→{"MonteCarlo","RandomSeed"→1,"MaxPoints"→400},
  IntegrationMonitor:→Print,EvaluationMonitor:→Sow[x]]];

SeedRandom[1];g[x_]=Exp[-x^2];ListeZZ1=RandomReal[{0,3},500];
6*Mean[g[ListeZZ1]]
6*Sqrt[Variance[g[ListeZZ1]]/500]

SeedRandom[1];s[t_]=6*Exp[-9t^2];ListeZZH=Table[RandomReal[{0,1},100],{5}];
Sum[Mean[s[ListeZZH[[i]]]],{i,1,5}]/5

3*Flatten[ListeZZH]==INMC[[2,1]]
ListeZZ1==INMC[[2,1]]
ListeZZ1-INMC[[2,1]]
```

```
a1 = Mean[s[ListeZZH[[1]]]];e1 = a1;
a2 = a1+Mean[s[ListeZZH[[2]]]];e2 = a2/2;
a3 = a2+Mean[s[ListeZZH[[3]]]];e3 = a3/3;
a4 = a3+Mean[s[ListeZZH[[4]]]];e4 = a4/4;
a5 = a4+Mean[s[ListeZZH[[5]]]];e5 = a5/5
v1 = StandardDeviation[s[ListeZZH[[1]]]]/Sqrt[100];
v2 = 1/Sqrt[200.*199.]*Sqrt[100.*99.*v1^2+Total[(s[ListeZZH[[2]]]-e2)^2]];
v3 = 1/Sqrt[300.*299.]*Sqrt[200.*199.*v2^2+Total[(s[ListeZZH[[3]]]-e3)^2]];
v4 = 1/Sqrt[400.*399.]*Sqrt[300.*299.*v3^2+Total[(s[ListeZZH[[4]]]-e4)^2]];
v5 = 1/Sqrt[500.*499.]*Sqrt[400.*399.*v4^2+Total[(s[ListeZZH[[5]]]-e5)^2]]

IQMC = Reap[NIntegrate[2*Exp[-x^2],{x,0,3},PrecisionGoal → 6,
  Method → {"QuasiMonteCarlo","MaxPoints" → 500},
  IntegrationMonitor :→ Print,EvaluationMonitor :→ Sow[x]]];

SeedRandom[0,Method → {"MKL",Method → "Sobol"}];
ListeZZ2 = RandomReal[{0,3},500];
ListeZZ2 = = IQMC[[2,1]]
6*Mean[g[ListeZZ2]]
IANMC = Reap[NIntegrate[2*Exp[-x^2],{x,0,3}, PrecisionGoal → 6,
  Method → {"AdaptiveMonteCarlo","MaxPoints" → 500},
  IntegrationMonitor :→ Print,EvaluationMonitor :→ Sow[x]]];

INMC2 = Reap[NIntegrate[2*Exp[-x^2],{x,0,3},PrecisionGoal → 6,
  Method → {"MonteCarlo","RandomSeed" → 1,"MaxPoints" → 400,
  Method → {"MonteCarloRule","Points → 200}},
  IntegrationMonitor :→ Print,EvaluationMonitor :→ Sow[x]]];
```

A.29 Eingabe-Code zur Simulation aus 6.1 (S. 83):

Definition der Punktefunktion:

```
h[θ_] = Piecewise[{{11.,(-1.)Pi < θ < -19.Pi/20.},{8.,-19.Pi/20. ≤ θ < -17.Pi/20.},
  {16.,-17.Pi/20. ≤ θ < -15.Pi/20.},{7.,-15.Pi/20. ≤ θ < -13.Pi/20.},
  {19.,-13.Pi/20. ≤ θ < -11.Pi/20.},{3.,-11.Pi/20. ≤ θ < -9.Pi/20.},
  {17.,-9.Pi/20. ≤ θ < -7.Pi/20.},{2.,-7.Pi/20. ≤ θ < -5.Pi/20.},
  {15.,-5.Pi/20. ≤ θ < -3.Pi/20.},{10.,-3.Pi/20. ≤ θ < -Pi/20.},
  {6.,-Pi/20. ≤ θ < Pi/20.},{13.,Pi/20. ≤ θ < 3. Pi/20.},
  {4.,3.Pi/20. ≤ θ < 5.Pi/20.},{18.,5.Pi/20. ≤ θ < 7.Pi/20.},
  {1.,7.Pi/20 ≤ θ < 9.Pi/20.},{20.,9.Pi/20. ≤ θ < 11.Pi/20},
  {5.,11.Pi/20. ≤ θ < 13.Pi/20.},{12.,13.Pi/20. ≤ θ < 15.Pi/20.},
  {9.,15.Pi/20. ≤ θ < 17.Pi/20.},{14.,17.Pi/20. ≤ θ < 19.Pi/20.},
  {11.,19.Pi/20. ≤ θ ≤ 1.Pi}},0.];
p[r_,θ_] = Piecewise[{{50.,0. ≤ r < 6.35},{25.,6.35 ≤ r < 15.9},{h[θ],15.9 ≤ r < 99.},
  {3*h[θ],99. ≤ r < 107.},{h[θ],107. ≤ r < 162.},{2.*h[θ],162. ≤ r < 170.}},0.];
g[x_,y_] = p[Sqrt[x^2+y^2],ArcTan[x,y]];
```

Festlegung der Parameter und Definition des Gitters:

```
σ = 25.;τ = 30.;n = 10000;a = 2.5;
Folge = Table[k,{k,-170,170,a}];
```

```
Gitter = Tuples[{Folge,Folge}];
Zielpunkte = Select[Gitter,Norm[#] < 170&];
```

Erwartungswertbestimmung:

```
ListeE = {};ListeF = {};j = 1;
AbsoluteTiming[While[j < Length[Zielpunkte]+1,
  SeedRandom[j,Method → "MersenneTwister"];
  ListeX = RandomVariate[NormalDistribution[Zielpunkte[[j,1]],σ],n];
  ListeY = RandomVariate[NormalDistribution[Zielpunkte[[j,2]],τ],n];
  ListeG = Table[g[ListeX[[i]],ListeY[[i]]],{i,1,n}];
  AppendTo[ListeE,Mean[ListeG]];
  AppendTo[ListeF,StandardDeviation[ListeG]/Sqrt[n]];j++];]
```

Ordnen der gesammelten Daten:

```
Daten = Sort[Transpose[{Zielpunkte,ListeE,ListeF}],#1[[2]] > #2[[2]]&];
GuteZielpunkte = Select[Daten,#[[2]] > 0.95*Daten[[1,2]]&];
```

Grafische Darstellung der Ergebnisse:

```
TextScheibe = Table[Text[Style[{6,13,4,18,1,20,5,12,9,14,11,8,16,7,19,3,17,2,
  15,10}[[i+1]],Large],
  FromPolarCoordinates[{190,If[i ≤ 10,i*Pi/10,i*Pi/10-2Pi]}]],{i,0,19}];
LinienScheibe = Table[Rotate[Line[{{0,15.9},{0,170}}],Pi/10*(i+0.5),{0,0}],
  {i,0,19}]~Join~{Circle[{0,0},6.35],Circle[{0,0},15.9],Circle[{0,0},99],
  Circle[{0,0},107],Circle[{0,0},162],Circle[{0,0},170]};
Scheibe = Flatten[TextScheibe~Join~LinienScheibe];
GrafikScheibe1 = Show[ListPlot[
  {Zielpunkte,GuteZielpunkte[[All,1]],{{Daten[[1,1,1]],Daten[[1,1,2]]}}},
  PlotStyle → {Default,Yellow,{Red,PointSize[0.007]}}],
  Graphics[{Opacity[0.8]}~Join~Scheibe],AspectRatio → 1,
  PlotRange → {{-200,200},{-187,187}},Axes → False]
ListeErw = Table[{Daten[[i,1,1]],Daten[[i,1,2]],Daten[[i,2]]},{i,1,Length[Daten]}];
VertErw = ListDensityPlot[ListeErw,ColorFunction → "SunsetColors",
  PlotLegends → Automatic, Frame → False,AspectRatio → 1];
GrafikScheibe2 = Show[VertErw,ListPlot[{{ListeErw[[1,1]],ListeErw[[1,2]]}},
  PlotStyle → {Red,Thick}],Graphics[{Opacity[0.8]}~Join~Scheibe],PlotRange → All]
```

Export bzw. Import der Daten und Grafiken:

```
Speicherort = "C:\\...\\";
Export[Speicherort <> "Daten_Simulation.txt",Partition[Flatten[Daten],4],"Table"];
Export[Speicherort <> "Grafik_Simulation1.pdf",GrafikScheibe1];
Export[Speicherort <> "Grafik_Simulation2.pdf",GrafikScheibe2];

ImportDaten = Import[Speicherort <> "Daten_Simulation.txt","Table"];
```

A.30 Eingabe-Code zu 6.2 (S. 88):

Import der empirischen Daten (Koordinaten der Treffer):

```
Speicherort = "C:\\...\\";
EmpDaten = Import[Speicherort <> "Emp_Daten.txt","Table"];
Treffer = Table[{EmpDaten[[i,1]],EmpDaten[[i,2]]},{i,1,Length[EmpDaten]}];
```

Übernimm nun die Definitonen `h[θ_]`, `p[r_,θ_]`, `g[x_,y_]`, `TextScheibe`, `LinienScheibe` und `Scheibe` aus A.29.

Statistische Auswertung der erhobenen Daten:

```
Punkte = Table[g[Treffer[[i,1]],Treffer[[i,2]]],{i,1,Length[Treffer]}];
Grid[{{"Anzahl der Würfe:",n = Length[Treffer]},
  {"Durchschnittliche Punktzahl pro Wurf:",Mean[Punkte]},
  {"Mittelwert (u,v):",M = Mean[Treffer]},
  {"Streuung (σ,τ):",S = StandardDeviation[Treffer]}},
  Alignment → Left,Frame → All,Background → Yellow]
```

Grafische Darstellungen des Ergebnisses:

```
PlotErgebnis = Show[ListPlot[Treffer,PlotStyle → Red],
  Graphics[{Opacity[0.8]}~Join~Scheibe],AspectRatio->1,
  PlotRange → {{-200,200},{-187,187}},Ticks → False,AxesLabel → {x,y},
  AxesStyle → {Thin,Thin},AxesOrigin → {0,0}]
```

Überprüfung, ob die empirische Daten zum Modell passen:

```
XListe = Table[Treffer[[i,1]],{i,1,Length[Treffer]}];
GX1 = Histogram[XListe,Automatic,"PDF"];
GX2 = Plot[PDF[NormalDistribution[M[[1]],S[[1]]],x],
  {x,M[[1]]-4*S[[1]],M[[1]]+4*S[[1]]}];
Show[GX1,GX2,PlotRange → All]
YListe = Table[Treffer[[i,2]],{i,1,Length[Treffer]}];
GY1 = Histogram[YListe,Automatic,"PDF"];
GY2 = Plot[PDF[NormalDistribution[M[[2]],S[[2]]],x],
  {x,M[[2]]-4*S[[2]],M[[2]]+4*S[[2]]}];
Show[GY1,GY2,PlotRange → All]
Correlation[XListe,YListe]
AndersonDarlingTest[Treffer,MultinormalDistribution[
  {M[[1]],M[[2]]},{{S[[1]]^2,0},{0,S[[2]]^2}}],"TestConclusion"]
GV1 = Plot3D[PDF[MultinormalDistribution[
  {M[[1]],M[[2]]},{{S[[1]]^2,0},{0,S[[2]]^2}}],{x,y}],
  {x,M[[1]]-3*S[[1]],M[[1]]+3*S[[1]]},{y,M[[2]]-3*S[[2]],M[[2]]+3*S[[2]]},
  PlotStyle → {Red,Opacity[0.3]}];
GV2 = Histogram3D[Treffer,Automatic,"PDF",ChartStyle → Green];
Show[GV1,GV2,Boxed → False,Axes → {True,True,False}]
```

Abstract

Abstract in Deutsch:

Integrieren mithilfe des Zufalls, das ist, was die Monte-Carlo-Integration leistet. Diese Diplomarbeit beschäftigt sich sowohl mit der theoretischen als auch mit der praktischen Seite dieser speziellen Art des numerischen Integrierens. Ein besonderes Augenmerk wird dabei auf die Umsetzung dieser Integrationsmethode mit Mathematica gelegt, die im Rahmen von vielen Beispielen ausführlich erklärt wird. Die ersten beiden Kapitel bilden das Fundament für diese Arbeit. Zunächst werden einige, für diese Arbeit relevante, Ergebnisse aus der Wahrscheinlichkeitstheorie präsentiert. Die Erzeugung von Zufallszahlen und zufälligen Vektoren wird im zweiten Kapitel thematisiert. Im dritten Kapitel wird ausgehend von den beiden vorherigen Kapiteln die (naive) Monte-Carlo-Integration eingeführt. Darüberhinaus werden in diesem Hauptkapitel neben einem interessanten Spezialfall, der sogenannten hit-or-miss-Monte-Carlo-Integration, und einer weiteren Art zur Zufallszahlerzeugung, die Genauigkeit der Monte-Carlo-Integration eingehend behandelt. Im vierten Kapitel wird die Monte-Carlo-Integration mit der Trapezregel verglichen, um die Vor- und Nachteile dieser Integrationsmethode herauszuarbeiten. Während die universelle Einsetzbarkeit sicher die größte Stärke dieser Methode ist, so ist die Geschwindigkeit ihre größte Schwäche. Im fünften Kapitel wird gezeigt, wie die Monte-Carlo-Integration mithilfe von Techniken zur Varianzreduktion oder durch den Einsatz von Quasizufallszahlen beschleunigt werden kann. Abschließend wird ein Modell vorgestellt, dass es Dartspielerinnen und -spielern ermöglicht, in Abhängigkeit von ihrer Spielstärke, den optimalen Zielpunkt zu finden.

Abstract in Englisch:

Integrating using randomness, that is what the Monte Carlo integration does. This diploma thesis deals with the theoretical and practical aspects of this special kind of numerical integration. Special attention is given to the implementation of this method of numerical integration in Mathematica, which is explained in detail using multiple examples. The first two chapters provide the foundation of the thesis. Firstly, some important results from the field of probability theory are presented. Secondly, the generation of random numbers and random vectors is discussed. In chapter three, based on the two previous chapters, (crude) Monte Carlo Integration is introduced. Additionally, an interesting special case called hit-or-miss Monte Carlo integration, a further technique to generate randomness, and the accuracy of this integration method are discussed in this main chapter. In chapter four, Monte Carlo integration and the trapezoidal rule are compared in order to identify their advantages and disadvantages. While universal applicability is certainly the greatest strength of this method, speed is its greatest weakness. Chapter five shows how Monte Carlo integration can be accelerated with variance reduction or the use of quasi-random sequences. Finally, a probabilistic model is presented which allows darts players to find the ideal spot to aim for depending on their skills.