Ludwigs-Maximilians-Universität München
— Departement for Physics —

University Observatory

Computational Methods in Astrophysics

Monte Carlo Simulations and Radiative Transfer

first edition (2005)

Joachim Puls

winter semester 2005/2006

Contents

Chapter 0

Introduction

This part of our practical course in computational techniques deals with Monte Carlo methods, which allow to solve mathematical/scientific/economic problems using *random numbers*. The essential approach is to approximate theoretical *expectation values* by *averages* over suitable samples. The fluctuations of these averages induce statistical variances which have to be accounted for in the analysis of the results. Most interestingly, Monte Carlo methods can be applied for both stochastic and *deterministic* problems.

Some typical areas of application comprise

- statistics (tests of hypothesis, parameter estimation)
- optimization
- integration, particularly in high dimensions
- differential equations
- physical processes.

An inevitable prerequesite for this approch is the availability of a (reliable) random number generator (RNG). Moreover, all results have to be analyzed and interpreted by statistical methods.

Outline. The outline of this manual is as follows. In the next chapter we will give an overview of the underlying theoretical concept. At first we will summarize few definitions and facts regarding probabilistic approaches (Sect. 1.1). In Sect. 1.2 we will consider the "construction" of simple RNGs and provide some possibilities to test them. An important application of Monte Carlo techniques constitutes the *Monte Carlo integration* which is presented in Sect. 1.3. The theoretical section is closed by an introduction into Monte Carlo *simulations* themselves (Sect. 1.4).

In Chap 2 we will outline the specific exercises to be solved during this practical work, culminating in the solution of one (rather simple) astrophysical radiative transfer problem by means of a Monte Carlo simulation.

Plan. The plan to carry out this part of our computational course is as follows: Prepare your work by carefully reading the theoretical part of this manual. If you are a beginner in Monte Carlo techniques, you might leave out those sections denoted by "advanced reading" (which consider certain topics which might be helpful for future applications).

On the first day of the lab-work then, exercise 1 and 2 should be finished, whereas the second day covers exercise 3. Please have a look into these exercises *before* you will accomplish them, in order to allow for an appropriate preparation.

Don't forget to include all programs, plots etc. into your final elaboration.

Chapter 1

Theory

1.1 Some basic definitions and facts

1.1.1 The concept of probability

Let's assume that we are performing an experiment, which can result in a number of different and discrete outcomes (events), e.g., rolling a dice. These events shall be denoted by E_k, where the probability for the occurence of such an event,

$$P(E_k) = p_k,$$

must satisfy the following constraints:

(a) $0 \leq p_k \leq 1$.

(b) If E_k cannot be realized, then $p_k = 0$.
If E_k is the definite outcome, then $p_k = 1$.

(c) If two events, E_i and E_j, are mutually exclusive, then

$$P(E_i \text{ and } E_j) = 0, \qquad P(E_i \text{ or } E_j) = p_i + p_j.$$

(d) If there are N mutually exclusive events, E_i, $i = 1, \ldots, N$, and these events are *complete* concerning all possible outcomes of the experiment, then

$$\sum_{i=1}^{N} p_i = 1. \tag{1.1}$$

For a two-stage experiment with events F_i and G_j, E_{ij} is called the combined event (G_j has occured after F_i). If F_i and G_j are *independent* of each other, the probability for the combined event is given by

$$p(E_{ij}) = p_{ij} = p(G_j) \cdot p(F_i), \tag{1.2}$$

1.1.2 Random variables und related functions

We now associate each discrete event, E_i, with a real number x_i, called *random variable* (r.v.) or *variate* . E.g., the event E_6 that the result of throwing a dice is a "six" might be associated with the r.v. $x_6 = 6$. Of course, any other association, e.g., $x_6 = 1$, is allowed as well.

The expectation value of a random variable (sometimes sloppyly denoted by "average" or "mean" value) is defined by

$$E(x) = \overline{x} = \sum_{i=1}^{N} x_i p_i, \tag{1.3}$$

if the events E_i, $i = 1, \ldots, N$ are complete and mutually exclusive. Any arbitrary function of this random variable, $g(x)$, is also a random variable, and we can construct the corresponding expectation value,

$$E(g(x)) = \overline{g} = \sum_{i=1}^{N} g(x_i) p_i. \tag{1.4}$$

Note that the expectation value is a linear operator,

$$E(ag(x) + bh(x)) = aE(g) + bE(h). \tag{1.5}$$

The variance of a r.v.,

$$\mathrm{Var}\,(x) = \overline{(x - \overline{x})^2} = \overline{x^2 - 2x \cdot \overline{x} + \overline{x}^2} = \overline{x^2} - \overline{x}^2, \tag{1.6}$$

describes the "mean quadratic deviation from the mean", and

The standard deviation, σ, is defined as

$$\sigma(x) = \left(\mathrm{Var}\,(x)\right)^{1/2}.$$

In contrast to the expectation value, **the variance is no linear operator** (prove by yourself!). If g and h are statistically *independent*, i.e.,

$$\overline{gh} = E(gh) = E(g) \cdot E(h) = \overline{g} \cdot \overline{h},$$

one can show that

$$\mathrm{Var}\,(ag + bh) = a^2 \mathrm{Var}\,(g) + b^2 \mathrm{Var}\,(h). \tag{1.7}$$

Thus, the standard deviation has the following scaling property,

$$\sigma(ag) = a\sigma(g).$$

1.1.3 Continuous random variables

So far, we have considered only discrete events, E_i, with associated discrete random variables, x_i. How do we have to proceed in those cases where different events can be no longer enumerated (example: scattering angles in a scattering experiment)?

The probability for a *distinct* event (e.g., that an exactly specified scattering angle will be realized) is zero; on the other hand, the probability that the event lies inside a specific (infinitesimal) interval is well defined,

$$P(x \le x' \le x + \mathrm{d}x) = f(x)\mathrm{d}x, \tag{1.8}$$

where $f(x)$ is the so-called *probability density function* (pdf).

The probability that the result of the experiment will lie inside an interval $[a, b]$ can be calculated from this pdf,

$$P(a \le x \le b) = \int_a^b f(x')\mathrm{d}x'. \tag{1.9}$$

In analogy to the discrete probabilities, p_i, also the pdf has to satisfy certain constraints:

$$\text{(a)} \qquad f(x) \geq 0, \qquad -\infty < x < \infty \tag{1.10a}$$

$$\text{(b)} \qquad \int_{-\infty}^{\infty} f(x')\mathrm{d}x' = 1 : \tag{1.10b}$$

the combined probability that *all* possible values of x will be realized is unity!

The *cumulative probability distribution* (cdf), $F(x)$, denotes the probability that all events up to a certain threshold, x, will be realized,

$$P(x' \leq x) \equiv F(x) = \int_{-\infty}^{x} f(x')\mathrm{d}x'. \tag{1.11}$$

Consequently,

- $F(x)$ is a monotonic increasing function (because of $f(x) \geq 0$),
- $F(-\infty) = 0$ and
- $F(\infty) = 1$.

1.1 Example. A **uniform distribution** between $[a, b]$ is defined via a *constant* pdf:

$$f(x) = C \qquad \forall\ x \in [a, b].$$

The probability, that any value inside $x \leq x' \leq x + \mathrm{d}x$ is realized, i.e., $f(x)\mathrm{d}x$, is identical for all $x \in [a, b]$. Since for the remaining intervals, on the other hand, we have

$$f(x) = 0, \qquad (-\infty, a) \text{ and } (b, \infty),$$

it follows immediately that

$$F(\infty) = \int_{-\infty}^{\infty} f(x)\mathrm{d}x = \underbrace{\int_{-\infty}^{a} f(x)\mathrm{d}x}_{=0} + \int_{a}^{b} f(x)\mathrm{d}x + \underbrace{\int_{b}^{\infty} f(x)\mathrm{d}x}_{=0}$$

$$= \int_{a}^{b} f(x)\mathrm{d}x = C \cdot (b - a) \stackrel{!}{=} 1,$$

and we find

$$f(x) = \frac{1}{b - a} \qquad \text{for all } x \in [a, b]. \tag{1.12}$$

Thus, the pdf for uniformly distributed random numbers, drawn by a random number generator (RNG) with $a = 0, b = 1$ (cf. Sect. 1.2) is given by $f(x) = 1$.

The corresponding cumulative probability distribution results in

$$F(x) = \int_{a}^{x} f(x)\mathrm{d}x = \int_{a}^{x} \frac{\mathrm{d}x}{b - a} = \frac{x - a}{b - a} \qquad \forall\ x \in [a, b], \tag{1.13}$$

$$F(x) = 0 \qquad \text{for } x < a$$

$$F(x) = 1 \qquad \text{for } x > b$$

Regarding RNGs, this means that $F(x) = x$ for $x \in [0, 1]$. Figs. 1.1 and 1.2 display the situation graphically.

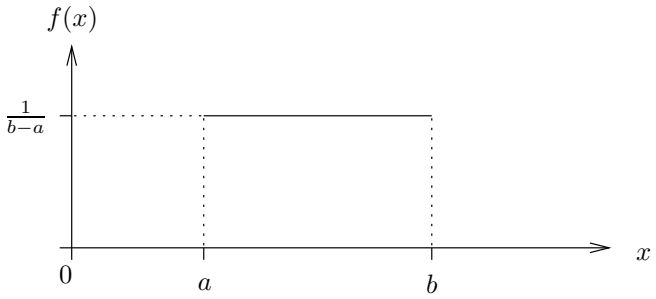

Figure 1.1: Probability density function (pdf), $f(x)$, of a *uniform* distribution within the interval [a,b].

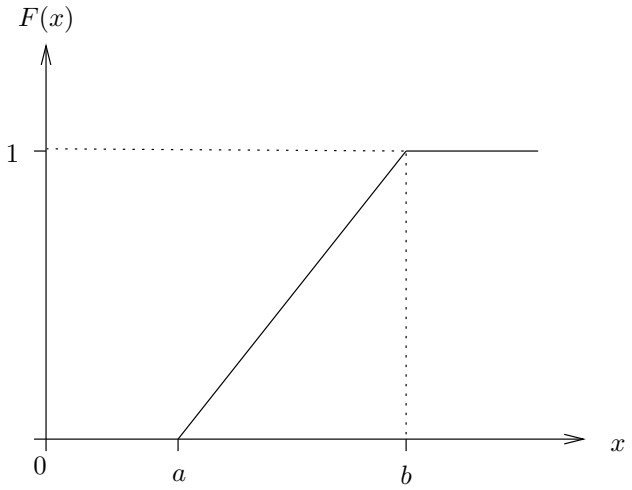

Figure 1.2: Corresponding cumulative probability distribution, $F(x)$.

In analogy to the discrete case, expectation value and variance of a continous random variable are defined by

$$E(x) \equiv \overline{x} \equiv \mu = \int_{-\infty}^{\infty} x' f(x') \mathrm{d}x' \tag{1.14a}$$

$$E(g) \equiv \overline{g} = \int_{-\infty}^{\infty} g(x') f(x') \mathrm{d}x' \tag{1.14b}$$

$$\mathrm{Var}\,(x) \equiv \sigma^2 = \int_{-\infty}^{\infty} (x' - \mu)^2 f(x') \mathrm{d}x' \tag{1.15a}$$

$$\mathrm{Var}\,(g) \equiv \sigma^2(g) = \int_{-\infty}^{\infty} \left(g(x') - \overline{g}\right)^2 f(x') \mathrm{d}x' \tag{1.15b}$$

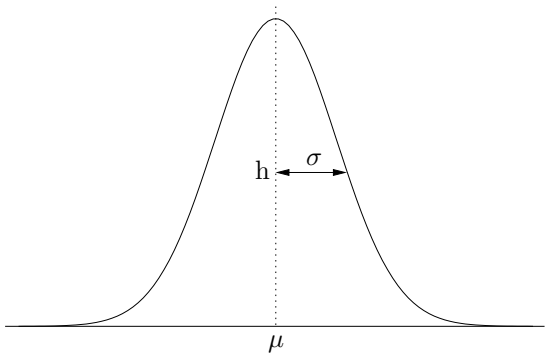

Figure 1.3: The GAUSS-distribution ($h = f(\mu)\mathrm{e}^{-1/2} \approx 0.61 \cdot f(\mu)$).

1.2 Example (The GAUSS- or normal distribution). The pdf of a GAUSS-distribution is defined as follows, see Fig. 1.3:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

Expectation value and variance can be easily calculated,

$$E(x) = \mu \qquad \mathrm{Var}\,(x) = \sigma^2$$

and the so-called "1σ-deviation" is given by

$$P(\mu - \sigma \leq x \leq \mu + \sigma) \approx 0.6826.$$

1.2 Some random remarks about random numbers

In this section, we will briefly discuss how to construct simple random number generators (RNGs) and how to test them. At first, let us consider the requirements for useful random numbers. Strictly speaking, there are three different classes of random numbers, namely

- *real random numbers*, e.g., obtained from radio-active decays (laborious, few)
- *pseudo random numbers*, calculated from a deterministic algorithm, and
- *sub-random numbers*, which have an "optimal" uniform distribution, but are not completely independent (for an introduction, see *Numerical Recipes*). Sub-random numbers are particularly used for Monte Carlo integrations.

In the remainder of this section, we will concentrate on pseudo random numbers, which must/should satisfy the following requirements:

- uniformly distributed
- statistically independent
- not or at least long-periodically (most algorithms produce random numbers with a certain period)
- reproducable (tests of programs!)
- portable: it should be possible to obtain identical results using different computers and different programming languages.
- calculated by a fast algorithm.

The first three of these requirements are a "must", whereas the latter ones are a "should", which will be met by most of the RNGs discussed in this section, particularly by the "minimum standard" RNG (see below).

1.2.1 (Multiplicative) linear congruential RNGs

For most "standard" tasks such as integrations and (astro-)physical simulations, the best choice is to use so-called (multiplicative) linear congruential RNGs, because of their speed, though other methods are availabe, mostly based on encryption methods, which, however, are much slower.

Basic algorithm.

$$\begin{aligned} \texttt{SEED} &= (A \cdot \texttt{SEED} + \texttt{C}) \bmod \texttt{M} \qquad (\texttt{SEED},\ A,\ C,\ M\ \texttt{integer-numbers}) \\ x &= \texttt{SEED}/\texttt{M} \qquad \text{normalization, conversion to } \texttt{real}\text{-number} \end{aligned} \tag{1.16}$$

- For any given initial value, SEED, a new SEED-value ("1st random number") is calculated. From this value, a 2nd random number is created, and so on. Since SEED is always (integer mod M), we have

$$0 \leq \texttt{SEED} < \texttt{M}, \qquad \text{i.e., } x \in [0, 1).$$

- The maximum value possible for this algorithm is $M - 1$.

- By construction, the algorithm implies a periodic behaviour: a new cycle starts, if

$$\texttt{SEED}(\text{actual}) = \texttt{SEED}(\text{initial})$$

- Since this period should be as long as possible
  $\Rightarrow M$ should roughly correspond to the maximum `integer` value which can be represented on the specific system.
- An optimum algorithm would produce *all* possible integers, $0 \ldots M-1$, which, however, is not always the case.

At least for linear congruential RNGs with $C > 0$ it can be proven that the maximum possible period, M, can be actually created if certain (simple) conditions relating C, A and M are satisfied.

1.3 Example (Algorithms with $C > 0$ and $M = 2^n$).

- $n \leq 31$ for 4-byte `integers` (note: only positive/zero integers are used)
- As an example, $A = 5$ and $C = 1$ satisfy the required conditions if $n \geq 2$ (other combinations are possible as well). In the following we consider the cycle with $n = 5$, i.e., $M = 32$.
- According to the above statement, the *maximum* possible cycle ($\rightarrow$ 32 different numbers) should be produced by using these values, which is actually the case, as shown in Tab. 1.1

$$\begin{aligned}&\texttt{SEED} \quad (\text{start}) = 9\\ &\texttt{SEED} \quad (1) = 5 \cdot 9 + 1 \bmod 32 = 46 \bmod 32 = 14, \text{ etc.},\end{aligned}$$

↓	9	1	25	17	9	(← new cycle)
	14	6	30	22	14	
	7	31	23	15	7	
	4	28	20	12	·	
	21	13	5	29	·	
	10	2	26	18	·	
	19	11	3	27		
	0	24	16	8		

Table 1.1: Random number sequence $\texttt{SEED} = (5 \cdot \texttt{SEED} + 1) \bmod 32$ with initial value 9.

The problem of portability.

Since we are aiming at a rather long cycle, $(M-1)$ should correspond to the maximum `integer` which can be represented on a specific machine ($2^{31} - 1$ for 4-byte integers). Because of (1.16), the calculation of `SEED` requires the evaluation of intermediate results being *larger* than $M - 1$ ("overflows") so that the algorithm would fail (e.g, if `SEED` becomes of order M). In the above cycle, this problem would already occur for the first number calculated, 46 mod 32.

A possible solution would comprise an assembly language implementation of our algorithm, with 8-byte product registers. In this case, however, the algorithm would be no longer portable (implementation is machine dependent).

A way out of this dilemma was shown by Schrage (1979). He devised a method by which multiplications of type

$$(A \cdot \texttt{SEED}) \bmod \texttt{M}, \tag{1.17}$$

with A, `SEED` and M consisting of a certain length (e.g., 4 byte), can be calculated *without overflow*.

This "trick", however, implies that our algorithm has to get along with $C \equiv 0$. As a first consequence, `SEED` $=$ `0` is prohibited, and $x \in (0, 1)$.

In dependence of M, again only certain values of A are "allowed" to maintain the required properties of the RNG. For $M = 2^n$, e.g., long cycles are possible (though being at least a factor of 4 shorter than above), but most of these cylces inhibit certain problems, particularly concerning the independence of the created numbers. An impressive example of such problems is given in example 1.11.

Thus, for $C = 0$ we strongly advise to avoid $M = 2^n$ and to use alternative cycles with M *being a prime number*. In this case then, a maximum period of $M - 1$ (no "zero") can be created if A is chosen in a specific, rather complex way. The corresponding sequence has a long period, uniform distribution and almost no correlation problem, as discussed below.

1.2.2 The minimum-standard RNG

Based on these considerations, PARKand MILLER (1988) suggested to use the following, most simple RNG for the case $C \equiv 0$, called "minimum-standard" generator. In particular,

$$M = 2^{31} - 1 = 2147483647,$$

which, on the one hand, is the largest representable 4-byte `integer`, and, on the other, is also a prime number. This lucky situation is not a pure coincidence, since the above number is a so-called MERSENNE prime number.[1]

With this value of M, one obtains roughly $2.15 \cdot 10^9$ different random numbers $\in (0, 1)$, when A is given by one of the following numbers,

$$A = \begin{cases} 16807 = 7^5 \\ 48271 \\ 69621. \end{cases}$$

All three values comply to both the complex requirement for allowing the maximum period and one additional condition which arises if SCHRAGE's "trick" shall be applied.

Advanced reading: SCHRAGE's multiplication, $(A \cdot$ `SEED`$)$ mod `M`

At first, M must be decomposed in the following way:

$$M = A \cdot q + r \qquad \text{with } q = \left[\frac{M}{A}\right], \tag{1.18}$$

where square brackets denotes an `integer`-operation. In this way then,

$$r = M \bmod A.$$

1.4 Example. Let $M = 2^{31} - 1$ and $A = 16807$. Then

$$\begin{aligned} q &= \left[\frac{M}{A}\right] = 127773 \\ r &= M \bmod A = 2836 \end{aligned}$$

and thus

$$M = 16807 \cdot 127773 + 2836 = 2^{31} - 1.$$

[1] If p is prime, then in many but not all cases $2^p - 1$ is also prime. In such a case, this number is called a MERSENNE prime number.

With these definitions of q and r, the following theorem can be proven:

1.5 Theorem. *If* $r < q$ (this is the additional condition) *and* $0 < \texttt{SEED} < \texttt{M} - \texttt{1}$, *then*

(i)

$$\left.\begin{array}{r} A \cdot (\texttt{SEED} \bmod \texttt{q}) \\ r \cdot \left[\frac{\texttt{SEED}}{q}\right] \end{array}\right\} < M - 1, \tag{1.19}$$

can be calculated without overflow*! We are now looking for* $\texttt{SEED}^{\text{new}} = (\texttt{A} \cdot \texttt{SEED}) \bmod \texttt{M}$. *Let*

$$\texttt{DIFF} := \texttt{A} \cdot (\texttt{SEED} \bmod \texttt{q}) - \texttt{r} \cdot \left[\frac{\texttt{SEED}}{\texttt{q}}\right]. \tag{1.20}$$

Because of (1.19) it is also ensured that

$$|\texttt{DIFF}| < \texttt{M} - \texttt{1}$$

can be calculated without overflow as well.

(ii) The random number $\texttt{SEED}^{\text{new}}$ *finally results via*

$$\texttt{SEED}^{\text{new}} = \begin{cases} \texttt{DIFF}, & \textit{if} \quad \texttt{DIFF} > \texttt{0} \\ \texttt{DIFF} + \texttt{M}, & \textit{if} \quad \texttt{DIFF} < \texttt{0}. \end{cases} \tag{1.21}$$

1.6 Example (simple). Calculation of $(3 \cdot 5) \bmod 10$ without overflow corresponding to an intermediate value $> (M - 1) = 9$ ($A = 3$ and $\texttt{SEED} = \texttt{5}$):

$$\left.\begin{array}{r} q = \left[\frac{10}{3}\right] = 3 \\ r = 10 \bmod 3 = 1 \end{array}\right\} \qquad \text{thus } r < q \text{ and } \texttt{SEED} < \texttt{M} - \texttt{1} = \texttt{9}.$$

The requirements are satisfied, and with (1.20, 1.21) we find

$$(A \cdot \texttt{SEED}) \bmod \texttt{M} = \texttt{A} \cdot (\texttt{SEED} \bmod \texttt{q}) - \texttt{r} \left[\frac{\texttt{SEED}}{\texttt{q}}\right] = \underbrace{3 \cdot 2}_{<\texttt{M}-1} - \underbrace{1 \cdot 1}_{<\texttt{M}-1} = \texttt{5}.$$

1.7 Example (typical). $M = 2^{31} - 1 = 2147483647$, $\texttt{SEED} = \texttt{1147483647}$, $A = 69621$. Direct calculation yields

$$\texttt{SEED}^{\text{new}} = \underbrace{79888958987787}_{>\texttt{M}(*)} \bmod \texttt{M} = \texttt{419835740},$$

where $(*)$ cannot be represented with 4 bytes.

Using SCHRAGE's trick, we obtain, without overflow

$$\begin{aligned} \texttt{DIFF} &= A(\texttt{SEED} \bmod \texttt{M}) - \texttt{r} \left[\frac{\texttt{SEED}}{\texttt{q}}\right] \\ &= \underbrace{1309014042}_{<M-1} - 889178302 \\ &= 419835740 \overset{!}{=} \texttt{SEED}^{\text{new}}. \end{aligned}$$

Algorithm for the PARK-MILLER minimum-standard generator and SCHRAGE's multiplication:

```
RAN(ISEED)
Real RAN
Integer (4 Byte), Parameter[2]:: M = 2147483647, A = 69621, q = 30845, r = 23902
Real, Parameter::  M1 = 1./Float(M)
Integer (4 Byte)::  ISEED, K


  K = ISEED / q
  ISEED = A * (ISEED - K * q) - r * K

  IF (ISEED < 0)  ISEED = ISEED + M
  RAN = ISEED * M1
  END
```

Note that `(ISEED - K * q)` corresponds to the operation `SEED` mod `q`.

To calculate a random number with this generator, only four statements are required, the rest are declarations! Before the first call of `RAN`, the variable `ISEED` has to be initalized, with $0 <$ `ISEED` $< M - 1$. From then on, `ISEED` must not be changed except by `RAN` itself!

1.2.3 Tests of RNGs

In order to check for the reliability of any unknown RNG, a number of tests should be performed. In particular, one has to confirm that the random numbers provided by the generator are *uniformly distributed* and *statistically independent* from each other.

Two rather simple possibilities will be described in the following, namely the calculation of PEARSON's χ^2_p (which asymptotically corresponds to the "usual" χ^2, and a graphical test. We will see that *all* RNGs based on the linear congruential algorithm (1.16) suffer from a certain deficiency, which can be cured in a simple and fast way by the so-called "reshuffling" method.

1.2.3.1 Advanced reading: Test of uniform distribution

At first, one has to check whether the created random numbers are uniformly distributed in $(0, 1)$. Remember that a uniform distribution implies a constant pdf within this interval, i.e.,

$$p(x)\mathrm{d}x = \mathrm{d}x,$$

cf. Sect. 1.1.3. At first let us consider the following, more general problem. Let $N_i, i = 1, \ldots, k$ with

$$\sum_{i=1}^{k} N_i =: N$$

the number of outcomes regarding a certain event i during an "experiment". We like to test the probability that a particular value of N_i corresponds to a *given* number n_i.

1.8 Example (Test of dices). Let's roll the dice and define

N_1 number of events with outcome 1, ..., 4,
N_2 number of events with outcome 5 and 6,
$N = N_1 + N_2$.

[2] corresponding to a *globale constant* in `C++`

We will test whether N_1 and N_2 are distributed according to the expectation, $n_1 = \frac{2}{3}N$, $n_2 = \frac{1}{3}N$, i.e., whether the dice has not been manipulated.

Method. To perform such a test, PEARSON's χ^2_p has to be calculated

$$\chi^2_p = \sum_{i=1}^{k} \frac{(N_i - n_i)^2}{n_i}. \tag{1.22}$$

For this quantity, the following theorem can be proven (actually, this proof is rather complex).

1.9 Theorem. PEARSON*'s χ^2_p (1.22) is a statistical function, which* asymptotically *(i.e., for $N_i \gg 1\ \forall\ i$) corresponds to the sum of squares of f independent random variables, which are normally distributed, with expectation value 0 and variance 1. The degrees of freedom, f, are defined by*

- *if n_i is fixed, we have $f = k$.*
- *if n_i is normalized via $\sum n_i = \sum N_i = N$, we have $f = k - 1$ (*one *constraint).*
- *if m additional constraints are present, we have $f = k - m - 1$.*

Because of this theorem, PEARSON's χ^2_p asymptotically corresponds to the "usual" χ^2-distribution, well-known from optimization/regression methods. Remember that the expectation value of χ^2 is f and that the standard deviation of χ^2 is $\sqrt{(2f)}$. In praxis, "asymptotically" means that in each "channel" i at least $n_i = 5$ events are to be *expected*.

1.10 Example (Continuation: Test of dices). With $N_i, n_i, i = 1, 2$ as given above, PEARSON's χ^2_p results in

$$\chi^2_p = \sum \frac{(N_i - n_i)^2}{n_i} = \frac{(N_1 - \frac{2}{3}N)^2}{\frac{2}{3}N} + \frac{(N_2 - \frac{1}{3}N)^2}{\frac{1}{3}N}.$$

Because of the requirement that one has to expect at least five events in *each* channel, we have to throw the dice at least $N = 15$ times. Thus, we have

$$\chi^2_p = \frac{(N_1 - 10)^2}{10} + \frac{(N_2 - 5)^2}{5},$$

and the degrees of freedom are $f = 2 - 1 = 1$.

Let's test now four different dices. After 15 throws each we obtained the results given in Tab. 1.2. The quantity Q gives the probability that by chance a larger value of χ^2 (larger than actually found) could have been present. If Q has a reasonable value (typically, within $0.05 \ldots 0.95$), everything should be OK, whereas a low value indicates that the actual χ^2 is too large for the given degrees of freedom (dice faked!). Concerning the calculation of Q, see *Numerical Recipes*, chap. 6.2. Note that for an unmanipulated dice, $< \chi^2 > \approx 1 \pm \sqrt{2}$ is to be expected.

dice	N_1	N_2	χ^2_p	$Q(\chi^2_p, f = 1)$
1	10	5	0	1
2	1	14	24.3	$8.24 \cdot 10^{-7}$
3	4	11	10.8	10^{-3}
4	8	7	1.2	0.27

Table 1.2: Test of four different dices

The results of our "experiment" can be interpreted as follows:

d1ce 1: The outcome is "too good", throw the dice once more. Too good ($\chi^2 = 0$)) means that the result is "exactly" as one expects. Usually, such a result is only obtained if the experiment has been faked.

d2ce 2: The outcome is very unprobable, and the dice can be suspected to be manipulated.

d3ce 3: The outcome is unprobable, however still possible. Throw the dice once more.

d4ce 4: The result is OK, and the dice seems to be ummanipulated.

Comment. To obtain a larger significance regarding our findings, the experiment should be performed a couple of times (say, at least 10 times). The specific results have then to be checked versus the expectation that the individual χ^2 must follow the χ^2-*distribution* for f degrees of freedom, if the dices were unmanipulated (see below).

An alternative formulation of PEARSON**'s** χ_p^2 is given by

$$\chi_p^2 = \sum_{i=1}^{k} \frac{(N_i - n_i)^2}{n_i} = \sum_{i=1}^{k} \frac{(N_i - p_i N)^2}{p_i N}, \tag{1.23}$$

where

- k is the number of "channels" (possible events)
- N_i the number of events in channel i with $\sum_i N_i = N$,
- p_i the probability for the event in channel i.

The channels must be selected in such a way that *all* possible events can be accounted for, i.e., that $\sum_i p_i = 1$.

Application to random numbers. Remember the original question whether the provided random numbers are uniformly distributed within $(0, 1)$. To answer this question, we "bin" the interval $(0, 1)$ into k equally wide sub-intervals. We calculate N random numbers and distribute them according to their values into the corresponding channels. After the required number of random numbers has been drawn (see below), we count the occurence N_i in each channel. If the random numbers were uniformly distributed, we should have $p_i = \frac{1}{k}$ for *all* i.

Thus, we calculate

$$\chi_p^2 = \sum_{i=1}^{k} \frac{(N_i - \frac{N}{k})^2}{N/k},$$

such that in each channel the expected value of $n_i = N/k$ is at least 5, i.e, a minimum of $N = 5k$ random numbers have to be drawn. Then,

- the probability $Q(\chi_p^2, k-1)$ has to be calculated, and
- the test has to be repeated l times, with typically $l \approx 10$. Note, that the initial `SEED` value has to be different for different l, of course.
- At last, we test whether the individual $\chi_{p,j}^2 (j = 1, \ldots, l)$ within the series are distributed according to χ^2 with $k-1$ degrees of freedom. This can be obtained by the so-called KOLMOGOROFF-SMIRNOV-test. [3]
- If the indivual Q_j values are "reasonable" and the KOLMOGORROFF-SMIRNOV-test has yielded a "reasonable" probability as well, the RNG provides indeed uniformly distributed random numbers, at least with respect to the defined channels.

[3] e.g., *Numerical Recipes*, chap. 14.3

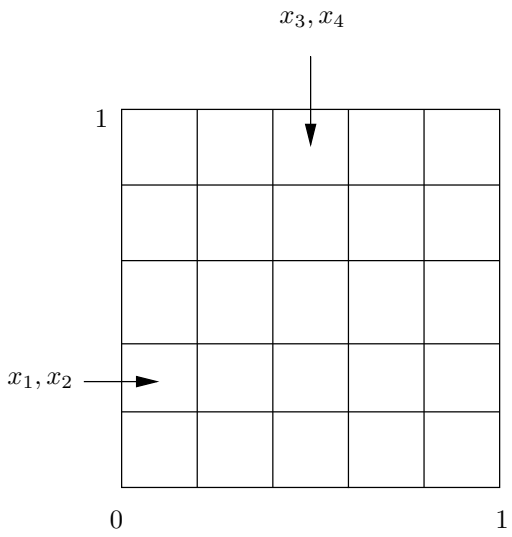

Figure 1.4: 2-D test (see text).

1.2.3.2 Advanced reading: Test of statistical independence

Having accomplished this test, we are not yet finished. Though we do know now (hopefully!) that the number are uniformly distributed, we still have to check whether they are *statistically independent*.

In other words: The probability that *after* a number x_i the number x_{i+1} will be drawn must be equal to the probability for x_{i+1} itself, i.e., independent of x_i.

$$P(x_{i+1}|x_i) \stackrel{!}{=} P(x_{i+1}) \quad \forall i,$$

To test this independence, *more-dimensional* tests have to be applied, which check for the *correlation* of random numbers.

a) Two-dimensional tests.
$P(x_{i+1}|x_i) \stackrel{!}{=} P(x_{i+1})$ checks whether two consecutive numbers are statistically independent.

For this purpose, we *bin* the unit square $(0,1)^2$ in two dimensions, create pairs of random numbers and sort them into corresponding "quadratic" channels (cf. Fig. 1.4 for the random numbers $x_1 = 0.1, x_2 = 0.3, x_3 = 0.5, x_4 = 0.9$). For k channels in one dimension one obtains k^2 channels in two dimensions, and $p = 1/k^2$ random numbers in each "quadratic" channel have to be expected (independent events, cf. (1.2)). The corresponding PEARSON's χ^2_p is calculated via

$$\chi^2_p = \sum_{i=1}^{k^2} \frac{(N_i - N/k^2)^2}{N/k^2},$$

if N_i is the number of drawn pairs in channel i. As a minimum, $N = 5k^2$ numbers have to be drawn now. Again, more than one series j has to be checked, and both the individual values of $\chi^2_{p,j}$ as well as their distribution (via KOLMOGOROFF-SMIRNOV) have to be inspected.

b) Three-dimensional tests.

$P(x_{i+2}|x_{i+1}, x_i) \stackrel{!}{=} P(x_{i+2})$ checks for the independence of a third random number from the previously drawn first and second one. In analogy to a) and b), we consider the unit *cube* $(0,1)^3$. Triples of three consecutive random numbers are sorted into this cube, and the corresponding probability is given by $p_i = 1/k^3$. The further proceeding is as above.

c) Typically, up to 8 dimensions should be checked for a rigorous test of a given RNG. (What problem might arise?) Check the web for published results.

1.11 Example (RANDU: A famous failure). One of the first RNGs which have been implemented on a computer, RANDU (by IBM!), used the algoritm (1.17) with $M = 2^{31}$, $A = 65539$ and $C = 0$, i.e., an algorithm which has been discussed as being potentially "dangerous" (remember that the minimum-standard generator uses a prime number for M). Though the above tests for 1- and 2-D are passed without any problem, the situation is different from the 3-D case on (it seems that IBM did not consider it as necessary to perform this test).

In the following, we carry out such a 3-D test with $30^3 = 27\,000$ channels and 270 000 random numbers. Thus, $N/(30^3) = 10$ events per channel are to be expected, and a value of χ^2 of the order of $f = (27\,000 - 1)$ should result for an appropriate RNG. We have calculated 10 test series each for the PARK/MILLER minimum-standard generator and for RANDU. Below we compare the outcome of our test.

```
Park/Miller Minimum-Standard
----------------------------

chi2 =  26851.98     Q =  0.7373354
chi2 =  27006.20     Q =  0.4844202
chi2 =  26865.37     Q =  0.7192582
chi2 =  27167.66     Q =  0.2369688
chi2 =  26666.29     Q =  0.9249226
chi2 =  27187.38     Q =  0.2070863
chi2 =  27079.77     Q =  0.3661267
chi2 =  26667.37     Q =  0.9247663
chi2 =  27142.82     Q =  0.2668213
chi2 =  27135.24     Q =  0.2786968
```

The probability that the values for χ^2_p do follow the χ^2-distribution is given (via KOLMOGOROFF-SMIRNOV) by

```
KS-TEST: prob = 0.7328884.
```

Thus, as it is also evident from the individual values of χ^2_p alone, being actually of the order of $\approx 27,000$, we conclude that the minimum-standard generator passes the 3-D test. In contrast, RANDU. The corresponding results are given by

```
RANDU
-----

chi2 =  452505.8     Q =  0.0000000E+00  (in single precision)
chi2 =  454412.7     Q =  0.0000000E+00
chi2 =  456254.8     Q =  0.0000000E+00
chi2 =  454074.1     Q =  0.0000000E+00
chi2 =  452412.2     Q =  0.0000000E+00
chi2 =  452882.0     Q =  0.0000000E+00
chi2 =  453038.7     Q =  0.0000000E+00
chi2 =  455033.4     Q =  0.0000000E+00
chi2 =  453992.2     Q =  0.0000000E+00
chi2 =  453478.2     Q =  0.0000000E+00
```

(note that each individual value of χ^2_p correspond to $\approx 17 \cdot f$) and the KS-test yields

```
KS-TEST: prob = 5.546628E-10 (!!!)
```

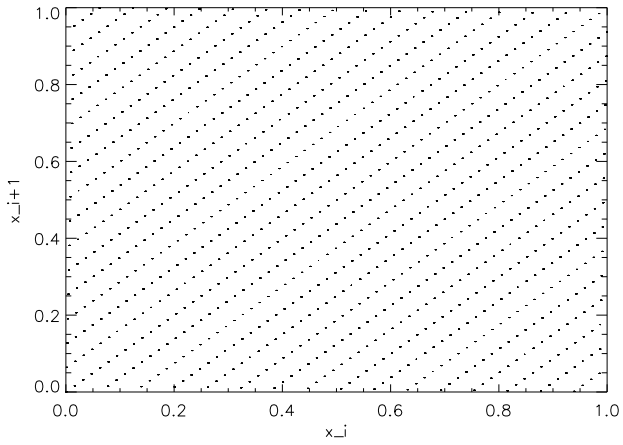

Figure 1.5: 2-D representation of random numbers, created via $\texttt{SEED} = (65 * \texttt{SEED} + 1) \bmod 2048$. The numbers are located on 31 hyper-surfaces.

i.e., the probability is VERY low that the individual χ^2_p follow a χ^2 distribution with $f \approx 27,000$. Though `RANDU` delivers numbers which are uniformely distributed (not shown here), each third number is strongly correlated with the two previous ones. Consequently, these numbers are *not* statistically independent and therfore *no* pseude-random numbers!

1.2.3.3 Graphical representation

One additional/alternative test is given by the graphical representation of the drawn random numbers. From such tests, a deficiency becomes obvious which affects all random numbers craeated by linar congruential algorithms: If one plots k consecutive random numbers in a k-dimensional space, they do *not* fill this space uniformely, but are located on $(k-1)$-dimensional hyper-surfaces. It can be shown that the maximum number of these hyper-surfaces is given by $M^{1/k}$, where only a careful choice of A and C ensures that this maximum is actually realized.
In Fig. 1.5 we display this deficiency by means of a (short-period) generator with algorithm

$$\texttt{SEED} = (65 * \texttt{SEED} + 1) \bmod 2048.$$

With respect to a 2-D representation, the maximum number of hyper-surfaces (here: straight lines) is given by $2048^{1/2} \approx 45$. With our choice of $A = 65$ and $C = 1$, at least 31 of those hyper-surfaces are realized.

If $M = 2^{31}$, a 3-D representation should yield a maximum number of $(2^{31})^{1/3} \approx 1290$ surfaces. For `RANDU` (see example 1.11), this number becomes reduced by a factor of $4^{1/3}$, because only even numbers and additionally only half of them can be created, thus reducing the maximum number of surfaces to 812. In contrast however, this generator delivers only 15! surfaces (see Fig. 1.6), i.e., the available space is by no means uniformly filled, which again shows the failure of `RANDU`.

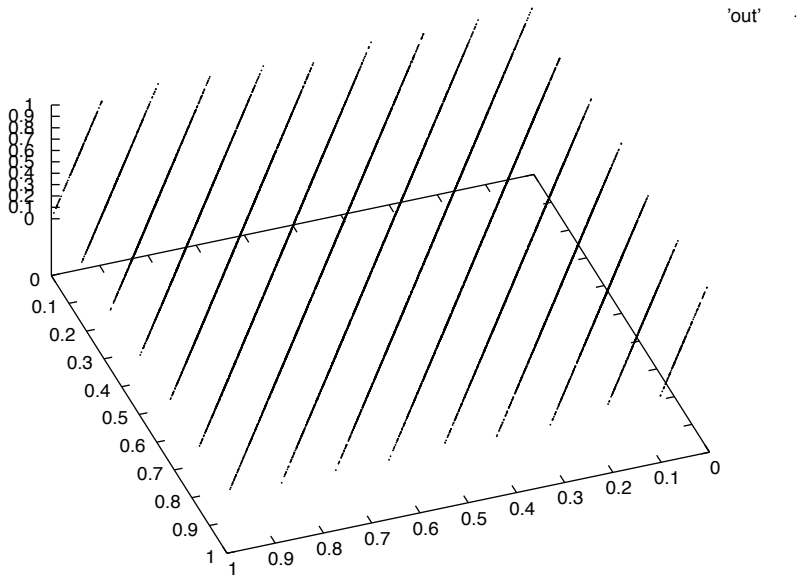

Figure 1.6: 3-D representation of "random" numbers created by `RANDU`. Though 812 hyper-surfaces are possible, only 15 of them are realized.

Reshuffling

To destroy the correlation described above (which will be always present for linear congruential RNGs), a fairly cheap and easy method is available, called *reshuffling*. In this method, the drawn random number itself is used to destroy the inherent sequence of consecutive numbers and thus the correlation. Fig. 1.7 displays the algorithm and Fig. 1.8 the result. The algorithm can be formulated as follows. Let y be a random number just drawn. A new random number follows from

(1) I = 1 + INTEGER(y · N) (calculate register index, if N registers are available)

(2) OUTPUT = V(I) (deliver random number)
Y = OUTPUT (prepare for next random number)

(3) V(I) = RAN (fill register with new random number)

The RNG recommended by *Numerical recipes*, "ran1", uses $N = 32$ registers V(1...N), which must be filled during a first "warm-up" phase (actually, the very first 8 numbers are not used at all), before the first number can be drawn.

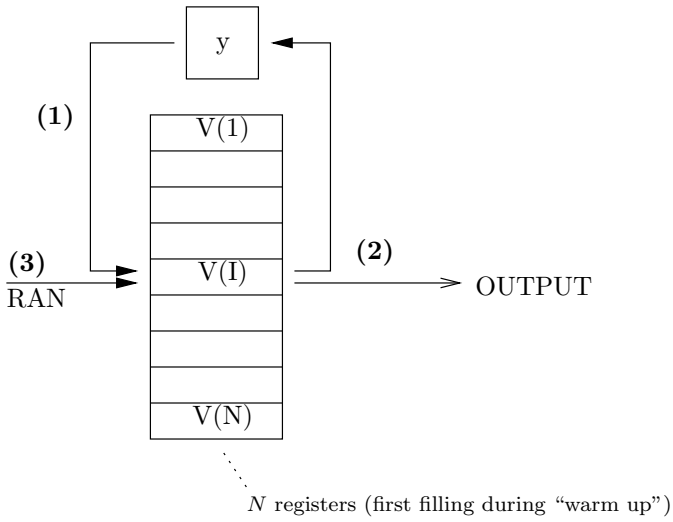

Figure 1.7: *Reshuffling* algorithm

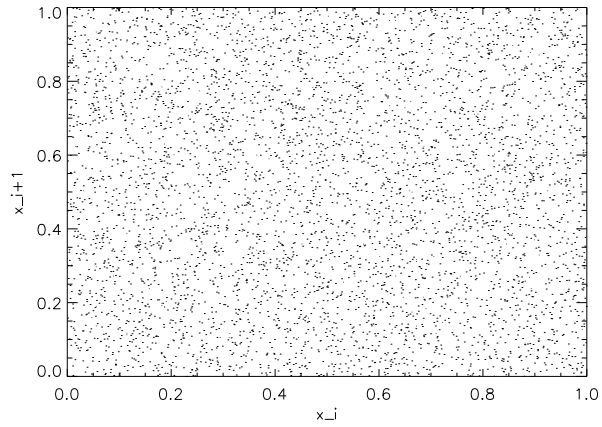

Figure 1.8: As Fig. 1.5, but with *reshuffling*. Any previous correlation has been destroyed.

1.3 Monte Carlo integration

After having discussed the generation of random numbers, we will discuss the *Monte Carlo integration* in this section, as a first and important application of Monte Carlo methods. Such an integration can replace more common methods based on well-known integration formulae (Trapez, Simpson etc.) and should be used under certain conditions, as outlined in Sect. 1.3.3.

1.3.1 Preparation

Before we can start with the "real" stuff, we need some preparatory work in order to understand the basic ideas. Impatient readers might immediately switch to Sect. 1.3.2.

- Let us begin by drawing N variates (= random variables) $x_1, \ldots, x_N$ from a given pdf $f(x)$, e.g., from the uniform distribution of pseudo random numbers as created by a RNG. For arbitrary pdfs, we will present the corresponding procedure in Sect. 1.4.

- From these variates, we calculate a new random variable (remember that functions of random variables are random variables themselves) by means of

$$G = \sum_{n=1}^{N} \frac{1}{N} g(x_n), \tag{1.24}$$

  where g is an arbitrary function of x. Since the expectation value is a linear operator, the expectation value of G is given by

$$E(G) = \overline{G} = \sum_{n=1}^{N} \frac{1}{N} E(g(x_n)) = \overline{g}, \tag{1.25}$$

  i.e, is the expectation value of g itself. The variance of G, on the other hand, is given by

$$\mathrm{Var}\,(G) = \mathrm{Var}\left(\sum_{n=1}^{N} \frac{1}{N} g(x_n)\right) = \sum_{n=1}^{N} \frac{1}{N^2} \mathrm{Var}\,(g(x_n)) = \frac{1}{N} \mathrm{Var}\,(g)\,, \tag{1.26}$$

  where we have assumed that the variates are statistically independent (cf. 1.7). Note that the variance of G is a factor of N lower than the variance of g.

- Besides being a random variable, G (1.24) is also the *arithmetical mean* with respect to the drawn random variables, $g(x_n)$. In so far, (1.25) states that the expectation value of the arithmetic mean of N random variables $g(x)$ is nothing else than the expectation value of g itself, independent of N.

- This will be the basis of the Monte Carlo integration, which can be defined as follows:

  Approximate the expectation value of a function by the arithmetic mean over the corresponding sample!!!

$$\overline{g} = \overline{G} \approx G$$

- Eq. 1.26 can be interpreted similarly: The variance of the arithmetical mean of N (independent!) random variables $g(x)$ is lower than the variance of $g(x)$ itself, by a factor of $1/N$. Note that statistically independent variates are essential for this statement, which justifies our effort concerning the corresponding tests of RNGs (Sect. 1.3.2.2).

Implication: The larger the sample, the smaller the variance of the arithmetical mean, and the better the approximation

$$\overline{g} \approx G.$$

Note in particular that

$$G \approx \overline{G} \pm \sqrt{\mathrm{Var}\,(G)}, \qquad \text{i.e., } G \to \overline{G} = \overline{g} \qquad \text{for } N \to \infty.$$

(Remember that $\mathrm{Var}\,(G)$ is the mean square deviation of G from $\overline{G}$)

Summary. Before applying this knowledge within the Monte Carlo integration, we like to summarize the derived relations.

$$\left.\begin{array}{rc} \text{pdf} & f(x) \\ \text{random variable} & g(x) \end{array}\right\} \quad \left.\begin{array}{c} E(g) \\ \mathrm{Var}\,(g) \end{array}\right\} \quad \left\{\begin{array}{l} \text{true expectation value} \\ \text{true variance} \end{array}\right.$$

$$E(g) = \overline{g} = \int g(x) f(x) \mathrm{d}x$$

$$\mathrm{Var}\,(g) = \int \left(g(x) - \overline{g}\right)^2 f(x) \mathrm{d}x$$

Draw N variates, x_n, $n = 1, \ldots, N$ according to $f(x)$ and calculate the arithmetical mean,

$$G = \frac{1}{N} \sum_{n=1}^{N} g(x_n).$$

Then we have

$$\begin{aligned} \overline{G} &= E(G) = E(g) = \overline{g} \\ \mathrm{Var}\,(G) &= \frac{1}{N} \mathrm{Var}\,(g) \\ \sigma(G) &= \left(\mathrm{Var}\,(G)\right)^{1/2} = \frac{1}{\sqrt{N}}\, \sigma(g) \end{aligned}$$

The last identity is known as the "$1/\sqrt{N}$-law" of the Monte Carlo simulation. The actual "trick" now is the

Monte Carlo approximation: Approximate $\overline{g} = \overline{G}$ by G

1.3.2 The method

The reader might question now in how far these statements concerning expectation values and variances have to do with real integration problems. This will become clear in a second, by a (clever) re-formulation of the integral I to be evaluated:

$$I = \int_a^b g(x) \mathrm{d}x = \int_a^b \frac{g(x)}{f(x)} f(x) \mathrm{d}x = \int_a^b h(x) f(x) \mathrm{d}x \tag{1.27}$$

The actual trick is now to demand that $f(x)$ shall be a probability density function (pdf). In this way then, any integral can be re-interpreted as the expectation value of $h(x)$ with respect to $f(x)$, and we just have learnt how to approximate expectation values!

Until further notification, we will use a constant pdf, i.e., consider a uniform distribution over the interval $[a, b]$,

$$f(x) = \begin{cases} \dfrac{1}{b-a} & a \le x \le b \\ 0 & \text{else,} \end{cases}$$

cf. (1.12). The "new" fuction $h(x)$ is thus given by

$$h(x) = \frac{g(x)}{f(x)} = (b-a)g(x),$$

and the integral becomes proportional to the expectation value of g:

$$I = (b-a)\int_a^b g(x)f(x)\mathrm{d}x \overset{\text{exact!}}{=} (b-a)\overline{g}. \tag{1.28}$$

We like to stress that this relation is still *exact*. In a second step then, we draw N variates x_n, $n = 1, \ldots, N$ according to the pdf (which is almost nothing else than drawing N random numbers, see below), calculate $g(x_n)$ and evaluate the corresponding arithmetical mean,

$$G = \frac{1}{N}\sum_{n=1}^{N} g(x_n).$$

Finally, the integral I can be easily estimated by means of the Monte Carlo approximation,

$$I = (b-a)\overline{g} \overset{\text{exact}}{=} (b-a)\overline{G} \overset{\text{Monte Carlo approx.}}{\approx} (b-a)G. \tag{1.29}$$

Thus, the established *Monte Carlo integration* is the (arithmetical) mean with respect to N "observations" of the integrand, when the variates x_n are uniformly drawn from the interval $[a, b]$, multiplied by the width of the interval.

The error introduced by the Monte Carlo approximation can be easily estimated as well.

$$\mathrm{Var}\,(G) = \frac{1}{N}\mathrm{Var}\,(g) = \frac{1}{N}\Big(\underbrace{\overline{g^2}}_{\overline{J}} - \underbrace{\overline{g}^2}_{\overline{G}^2}\Big) \approx \frac{1}{N}(J - G^2), \tag{1.30}$$

if

$$J = \frac{1}{N}\sum_{n=1}^{N} g^2(x_n)$$

is the Monte Carlo estimator for $E(J) = \overline{J} = \overline{g^2}$ and if we approximate $\overline{g}^2 = \overline{G}^2$ by G^2.

If we identify now the error of the estimated integral, δI, with the standard deviation resulting from $\mathrm{Var}\,(G)$[4] (which can be proven from the "central limit theorem" of statistics), it is obvious that this error follows the "$1/\sqrt{N}$-law", i.e.,

$$\delta I \propto 1/\sqrt{N}.$$

[4] multiplied by $(b-a)$

By generalizing the integration element $\mathrm{d}x$ to an arbitrary-dimensional volume element $\mathrm{d}V$, we find the following *cooking recipe for the Monte Carlo Integration*:

$$\int_V g\mathrm{d}V \approx V\left(\langle g\rangle \pm \sqrt{\frac{1}{N}\left(\langle g^2\rangle - \langle g\rangle^2\right)}\right) \tag{1.31}$$

if

$$\langle g\rangle = G = \frac{1}{N}\sum_n g(x_n)$$
$$\left\langle g^2\right\rangle = J = \frac{1}{N}\sum_n g^2(x_n)$$

and V is the volume of the integration region considered.

1.12 Example. [Monte Carlo integration of the exponential function] As a simple example, we will estimate the integral

$$\int_0^2 \mathrm{e}^{-x}\mathrm{d}x = 1 - \mathrm{e}^{-2} = 0.86467$$

via Monte Carlow integration.

(1) At first, we draw N random numbers uniformly distributed $\in [0,2]$ and "integrate" for the two cases, a) $N = 5$ and b) $N = 25$. Since the RNG delivers numbers distributed $\in (0,1)$, we have to multiply them by a factor of 2, cf. Sect. 1.4, Eq. 1.41. Assume that the following variates have been provided.

	Case a)		Case b)
1	7.8263693E-05	1	0.1895919
2	1.3153778	2	0.4704462
3	1.5560532	3	0.7886472
4	0.5865013	4	0.7929640
5	1.3276724	5	1.3469290
		6	1.8350208
		7	1.1941637
		8	0.3096535
		9	0.3457211
		10	0.5346164
		11	1.2970020
		12	0.7114938
		13	7.6981865E-02
		14	1.8341565
		15	0.6684224
		16	0.1748597
		17	0.8677271
		18	1.8897665
		19	1.3043649
		20	0.4616689
		21	1.2692878
		22	0.9196489
		23	0.5391896
		24	0.1599936
		25	1.0119059

(2) We calculate

$$G = \langle g\rangle = \frac{1}{N}\sum \mathrm{e}^{-X_n}$$
$$J = \langle g^2\rangle = \frac{1}{N}\sum \left(\mathrm{e}^{-X_n}\right)^2,$$

with the following results
a) $G = 0.4601251$, $J = 0.2992171$
b) $G = 0.4904828$, $J = 0.2930297$

(3) With

$$I \approx V\left(\langle g\rangle \pm \sqrt{\frac{1}{N}\left(\langle g^2\rangle - \langle g\rangle^2\right)}\right)$$

and "volume" $V = 2$, the integral is approximated by

Case a) $I \approx 0.9202501 \pm 0.2645783$
Case b) $I \approx 0.9809657 \pm 0.091613322$.

Note that the "$1/\sqrt{N}$-law" predicts an error reduction by $1/\sqrt{5} = 0.447$, not completely met by our simulation, since case a) was estimated based on a very small sample. Morever, the result given for case b) represents "reality" only within a 2-*sigma* error.

The error for $N = 25$ is of the order of ± 0.1. Thus, to reduce the error to ± 0.001 (factor 100), the sample size has to be increased by a factor of 100^2, i.e., we have to draw 250 000 random numbers.

The result of such a simulation (note that the RNG has to be fast) for $N = 250\,000$ is given by
$G = \langle g\rangle = 0.4322112$, $J = \langle g^2\rangle = 0.2453225$, thus

$I = 0.8644224 \pm 0.9676 \cdot 10^{-3}$.

We see that the (estimated) error has decreased as predicted, and a comparison of exact (I(exact) = 0.86476) and Monte Carlo result shows that it is also of the correct order.

As a preliminary conclusion then, we might state that the MC integration can be quickly programmed but that a large number of evalutions of the integrand are necessary to obtain a reasonable accuracy.

1.3.3 When shall we use the Monte Carlo integration?

Because of this large number of evaluations (and the corresponding computational time, even if the RNG itself is fast), one might question whether and when the Monte Carlo integration will be advantageous to standard quadrature methods such as exploiting the Simpson- or Trapez-rule. At first let us remind that the "$1/\sqrt{N}$-law" of the MC integration error is independent of the dimensionality of the problem, in contrast to errors resulting from standard quadrature methods.[5] In the latter case (and assuming simple boundaries, simple integration formulae and equidistant step-sizes), the integration errors scales via

$$\begin{aligned} \delta I &\propto N^{-\frac{2}{d}} \quad \text{Trapez} \\ \delta I &\propto N^{-\frac{4}{d}} \quad \text{Simpson,} \end{aligned} \tag{1.32}$$

for N equidistant sampling points and dimensionality d.

Thus, the error of the standard quadrature (with equidistant step size) increases from $\sim N^{-2(4)}$ for 1-D over $\sim N^{-1(2)}$ for 2-D to $\sim N^{-\frac{2}{3}(\frac{4}{3})}$ for 3-D integrals, whereas the MC integration always scales with $\sim N^{-1/2}$. Only from $d = 5$ (Trapez) or $d = 9$ (Simpson) on, the MC error decreases faster than the standard quadrature error. Before concluding that the MC integration is useless, however, one should consider the following arguments:

[5] though a rigorous error analyis for more-D integration is very complex, e.g., Stroud, A.H., 1971, *Approximate Calculation of Multiple Integrals*, Prentice-Hall.

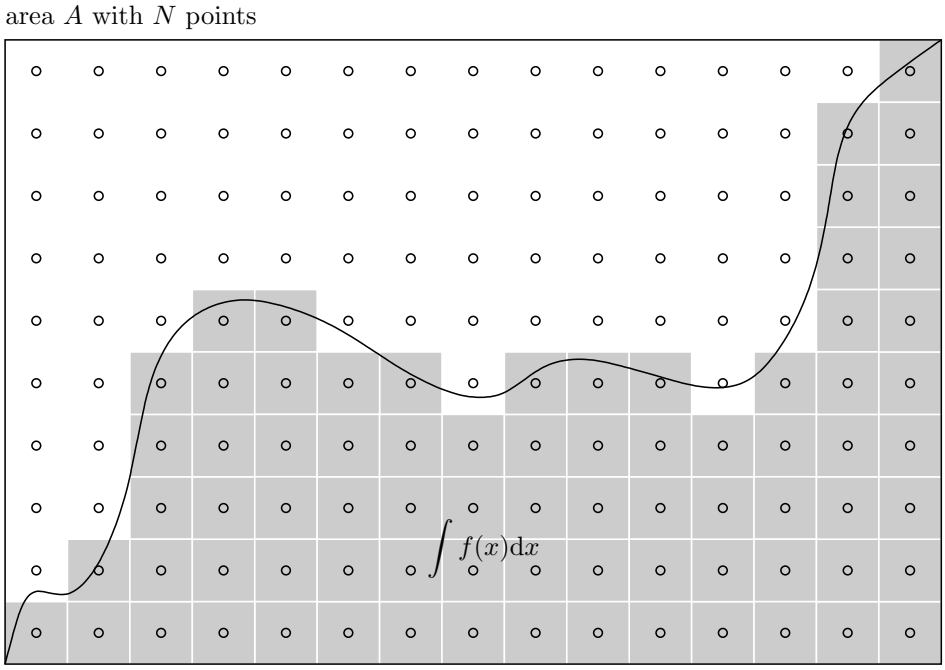

Figure 1.9: "Hit or miss"-method for the evaluation of 1-D integrals.

- Just for more-D integrals *complex boundaries* are the rule (difficult to program), and the MC integration should be preferred, as long as the integrand is not locally concentrated.
- If the required precision is low, and particularly for first *estimates* of unknown integrals, the MC integration should be preferred as well, because of its simplicity.
- If the specific integral must be calculated only once, again the MC solution might be advantegous, since the longer computational time is compensated by the greatly reduced programming effort.

1.3.4 Complex boundaries: "hit or miss"

To evaluate more-D integrals with complex boundaries, an alternative version of the MC integration is usually employed, knows as "hit or miss"-method. At least the principal procedure shall be highlighted here, for the 1-D case and by means of Fig. 1.9. To evaluate the integral $I = \int f(x)\mathrm{d}x$,

- N points with co-ordinates (x_i, y_i) are uniformely distributed across the rectangle with area A, by means of random numbers.
- All those points are counted as "hits" that have a y-co-ordinate with

$$y_i \leq f(x_i) \tag{1.33a}$$

If n hits have been counted in total, the integral can be easily approximated from

$$I = \int f(x)\mathrm{d}x \approx A \cdot \frac{n}{N}. \tag{1.33b}$$

- A disadvantage of this method is of course that all $(N - n)$ points being no "hit" (i.e., which are located above $f(x)$) are somewhat wasted (note that $f(x)$ has to be evaluated for *each point*). For this reason, the circumscribing area A has to be chosen as small as possible.
- The generalization of this method is fairly obvious, for typical examples see "Numerical recipes". E.g., to calculate the area of a circle, one proceeds as above, however now counting those points as hits on a *square* A which satisfy the relation $x_i^2 + y_i^2 \leq r^2$, for given radius r. Again, the area of the circle results from the fraction $A\frac{n}{N}$.

1.3.5 Advanced reading: Variance reduction – Importance sampling

The large number of required sampling points, being the consequence of a rather slow decrease in error ($\propto N^{-1/2}$) and giving rise to relatively large computational times, can be somewhat decreased by different methods. The most important ones make use of the so-called *stratified sampling* and *importance sampling*, respectively (mixed strategies are possible as well). The former approach relies on dividing the integration volume into sub-volumes, calculating the MC estimates for each individual sub-volume and adding up the results finally. If the sub-volumes are chosen in a proper way, the individual variances can add up to a total variance which is smaller than the variance obtained for an integration of the total volume. More information can be obtained from *Numerical Recipes*, Chap. 7.8.

In the following, we will have a closer look into the *importance sampling* approach. Remember that the integration error does not only contain the scaling factor $N^{-1/2}$, but depends also on the difference

$$(< g^2 > - < g >^2),$$

if g is the function to be integrated and angle brackets denote arithmetic means. It is this factor which can be reduced by importance sampling, sometimes considerably.

The basic idea is very simple. Remember that the integral I had been re-interpreted as an expectation value regarding a specified pdf p,

$$I = \int g \mathrm{d}V = \int \frac{g}{p} p \mathrm{d}V.$$

Instead of using a uniform distribution, i.e., a constant pdf, we will now use a different pdf p, chosen in such a way as to decrease the variance. Since any pdf has to be normalized, one always has to consider the constraint

$$\int p \mathrm{d}V = 1. \tag{1.34}$$

For arbitrary pdf's p then, the generalization of (1.31) is straightforward, and the Monte Carlo estimator of the integral is given by

$$I = \langle \frac{g}{p} \rangle \pm \frac{1}{\sqrt{N}} \left(\langle (\frac{g}{p})^2 \rangle - \langle \frac{g}{p} \rangle^2 \right)^{\frac{1}{2}}, \tag{1.35}$$

where $< g/p >$ is the arithmetic mean of $g(x_n)/p(x_n)$, and the variates x_n have been drawn according to the pdf p! Note in particular that any volume factor has vanished. From this equation, the (original) variance can be diminished signficantly if the pdf p is chosen in such a way that it is very close to g, i.e., if

$$\frac{g}{p} \approx \text{constant} \qquad \text{over the integration region.}$$

If $p = g$, the variance would become zero, and one might think that this solves the problem immediately. As we will see in the next section, however, the calculation of variates according to a pdf p requires the calculation of $\int p\mathrm{d}V$, which, for $p = g$, is just the integral we actually like to solve, and we would have gained nothing.

Instead of using $p = g$ then, *the best compromise is to use a pdf* p *which is fairly similar to* g*, but also allows for a quick an easy calculation of the corresponding variates.*

Finally, we will show that our MC cooking recipe (1.31) is just a special case for the above generalization, Eq. 1.35, under the restriction that p is constant. Because of the normalization (1.34), we immediately find in this case that

$$p = \frac{1}{V},$$

and

$$\begin{aligned} I = \int g\mathrm{d}V &\approx \langle \frac{g}{1/V} \rangle \pm \frac{1}{\sqrt{N}} \left(\langle (\frac{g}{1/V})^2 \rangle - \langle \frac{g}{1/V} \rangle^2 \right)^{\frac{1}{2}} \\ &= V \left(< g > \pm \frac{1}{\sqrt{N}} (< g^2 > - < g >^2)^{\frac{1}{2}} \right), \qquad \text{q.e.d.} \end{aligned}$$

1.4 Monte Carlo simulations

We are now (almost) prepared to perform Monte Carlo *simulations* (in the literal sense of the word). At first, let us outline the basic "philosophy" of such simulations.

1.4.1 Basic philosophy and a first example

- Assume that we have a compley physical (or economical etc.) problem, which cannot (or only with enormous effort) be solved in a closed way.
- Instead, however, we do know the "physics" of the constituating sub-processes.
- These individual sub-processes are described by using pdf's, and combined in a proper way as to describe the complete problem. The final result is then obtained from multiple runs through the various possible process-chains, in dependence of the provided pdf's (which are calculated from random numbers).
- After having followed a sufficient number of runs, we assume that the multitude of outcomes has provided a fair mapping of "reality", and the result (plus noise) is obtained from collecting the output of the individual runs.

advantages: Usually, such a method can be programmed in a fast and easy way, and results of complicated problems can be obtained on relatively short time-scales.

disadvantages: On the other hand, a deeper understanding of the results becomes rather difficult, and no *analytical* approximation can be developed throughout the process of solution. In particular, one has to perform a completely new simulation if only one of the parameters might change. Moreover, the reliability of the results depends crucially on a "perfect" knowledge of the sub-processes and a proper implementation of *all* possible paths which can be realized during the simulation.

1.13 Example (Radiative transfer in stellar atmospheres). Even in a simple geometry, the solution of the equation of radiative transfer (RT) is non-trivial, and in more complex situations (more-D, no symmetry, non-homogenous, clumpy medium) mostly impossible. Because of this reason, Monte Carlo simulations are employed to obtain corresponding results, particularly since the physics of the subprocesses (i.e., the interaction between photons and matter) is well understood.

As an example we consider the following (seemingly trivial) problem, which - in a different perspective - will be reconsidered in the practical work to be performed.

We like to calculate the spatial radiation energy density distribution $E(\tau)$, in a (plane-parallel) atmospheric layer, if photons are scattered by free electrons only. This problem is met, to a good approximation, in the outer regions of (very) hot stars.

One can show that this problem can be described by MILNE*'s integral equation*,

$$E(\tau) = \frac{1}{2} \int_0^\infty E(t) E_1 \left| t - \tau \right| \mathrm{d}t, \tag{1.36}$$

where $E_1(x)$ is the so-called first exponential integral,

$$E_1(x) = \int_1^\infty \frac{\mathrm{e}^{-x \cdot t}}{t} \, \mathrm{d}t,$$

and τ the "optical depth", here with respect to the radial direction (i.e., perpendicular to the atmosphere).

The optical depth is the relevant depth scale for problems involving photon transfer, and depends on the particular cross-section(s) of the involved interactions(s), σ, and on the absorber densities, n. Since, for a given

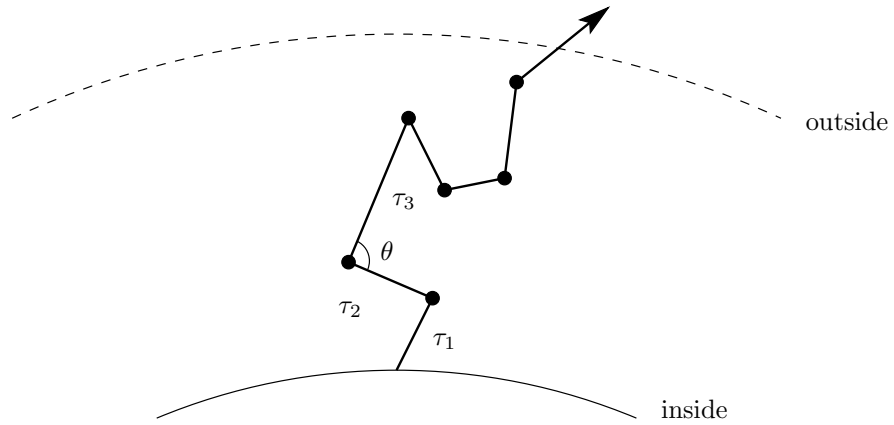

Figure 1.10: Monte Carlo Simulation: radiative transfer (electron scattering only) in stellar atmospheres.

frequency, usually different interactions are possible, all these possibilites have to be accounted for in the calculation of the optical depth. In the considered case (pure THOMSON-scattering), however, the optical depth is easily calculated and moreover almost frequency independent, if we consider only frequencies being lower than X-ray energies.

$$\tau(s_1, s_2) = \int_{r_2}^{r_1} \sigma\, n(s) \mathrm{d}s, \tag{1.37}$$

where the absorber density corresponds to the (free) electron density and the optical depth is defined between to spatial points with "co-ordinates" s_1 and s_2 and s is the geometrical length of the photon's path.

Analytic and MC solution of MILNE's integral equation

MILNE's equation (1.36) can be exactly solved, but only in a very complex way, and the spatial radiative energy density distribution (normalized to its value at the outer boundary, $\tau = 0$) is given by

$$\frac{E(\tau)}{E(0)} = \sqrt{3}\,(\tau + q(\tau))\,. \tag{1.38}$$

$q(\tau)$ is the so-called "HOPF-function", with $\frac{1}{\sqrt{3}} \leq q(\tau) < 0.710446$, which constitutes the real complicate part of the problem.[6]

An adequate solution by a Monte Carlo simulation, on the other hand, can be realized as follows (cf. Fig. 1.10, for more details see Sect. 2.3):

- Photons have to be "created" emerging from the deepest layers, defined by $\tau_{\max}$.
- The probability for a certain emission angle, θ (with respect to the radial direction), is given by

$$p(\mu)\mathrm{d}\mu \sim \mu \mathrm{d}\mu, \qquad \mu = \cos\theta.$$

- The optical depth passed until the next scattering event can be described by the probability

$$p(\tau)\mathrm{d}\tau \sim \mathrm{e}^{-\tau}\mathrm{d}\tau,$$

  (this is a general result for absorption of light) and
- the scattering angle (at low energies) can be approximated to be isotropic,

$$p(\mu)\mathrm{d}\mu \sim \mathrm{d}\mu.$$

[6] Note that an approximate solution of MILNE's equation can be obtained in a much simpler way, where the only difference compared to the exact solution regards the fact that $q(\tau)$ results in 2/3, independent of τ (see., e.g., Mihalas, D., 1978, *Stellar atmospheres*, Freeman, San Francisco, or http://www.usm.uni-muenchen.de/people/puls/stellar_at/stellar_at.html, Chap. 3 and 4).

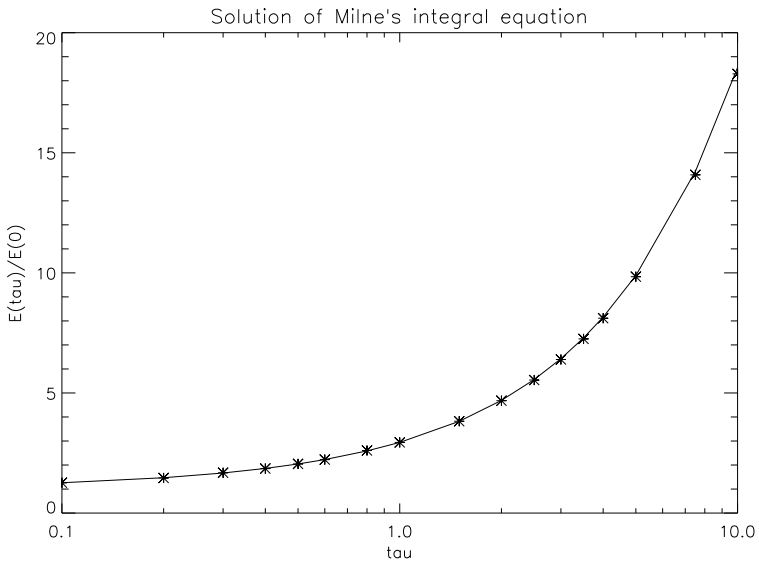

Figure 1.11: Solution of MILNE's integral equation (Example 1.13) by means of a Monte Carlo simulation using 10^6 photons. The MC solution is indicated by asterisks, and the analytical one (Eq. 1.38) is given by the bold line.

- All photons have to be followed until they leave the atmosphere ($\tau \leq 0$) or are scattered back into the stellar interior ($\tau > \tau_{\max}$), and the corresponding energies have to be summed up as a function of optical depth.
- If the simulation is performed for a *large* number of photons, the analytical result is well reproduced, cf. Fig. 1.11.

From this example and accounting for our knowledge accumulated so far, an immediate problem becomes obvious:

Though we are able to create uniform distributions on the interval $[0, 1)$ (or $(0, 1)$, respectively) via a RNG,

$$p(x) = \begin{cases} 1 & \text{for } x \in [0, 1) \\ 0 & \text{else,} \end{cases}$$

we still do not know how to obtain random variables (variates) y which are distributed according to an arbitrary (normalized!) pdf, $f(y)$. How to create, for example, optical depth lengths τ in Example 1.13, which are distributed according to a pdf $\mathrm{e}^{-\tau}\mathrm{d}\tau$, i.e., exponentially instead of uniformly?

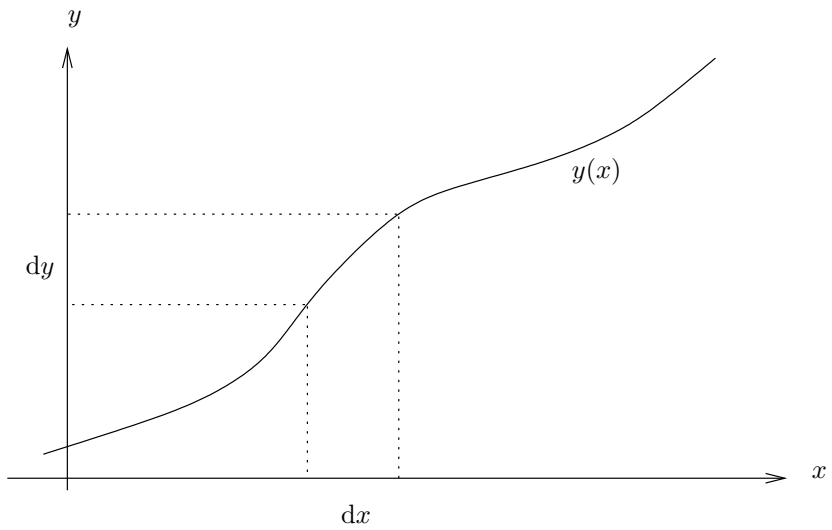

Figure 1.12: The transformation method, see text. $f(y)dy$ is the probability that y is in dy, and $p(x)dx$ is the probability that x is in dx.

1.4.2 Variates with arbitrary distribution

Actually, we encountered a similar problem already in example 1.12, where we were forced to consider a uniform distribution on the interval [0,2] (instead of [0,1]), and in Sect. 1.3.5, where we required the variates to be drawn from a specially designed pdf in order to reduce the variance.

Indeed, this problem is central to Monte Carlo simulations, and will be covered in the next section.

1.4.2.1 The inversion- or transformation method

Let $p(x)$ and $f(y)$ be different probability density functions (pdf's)

$$p(x)\mathrm{d}x = P(x \le x' \le x + \mathrm{d}x)$$
$$f(y)\mathrm{d}y = P(y \le y' \le y + \mathrm{d}y)$$

with $y = y(x)$: The random variable y shall be a function of the random variable x. A *physical* transformation means that corresponding probabilities, P, *must be equal*:

If $p(x)\mathrm{d}x$ is the probability that x is within the range $x, x + \mathrm{d}x$ and y is a function of x, then the probability that y is in the range $y, y + \mathrm{d}y$ must be the same!

1.14 Example. x shall be distributed uniformely in $[0, 1]$, and $y = 2x$. The probability that x is in $0.1 \ldots 0.2$ must be equal to the probability that y is in $0.2 \ldots 0.4$ (cf. Example 1.12).

Similarly, the probability that $x \in [0, 1]$ $(= 1)$ must correspond to the probability that $y \in [0, 2]$ (also $= 1$).

From this argumentation, we thus have

$$|p(x)\mathrm{d}x| = |f(y)\mathrm{d}y| \tag{1.39a}$$

or alternatively

$$f(y) = p(x) \left| \frac{\mathrm{d}x}{\mathrm{d}y} \right| .$$

We have to use the absolute value because $y(x)$ can be a (monotonically) increasing or decreasing function, whereas probabilities have to be larger than zero by definition. The more-D generalization of this relation involves the *Jacobian* of the transformation,

$$|p(x_1, x_2)\mathrm{d}x_1\mathrm{d}x_2| = |f(y_1, y_2)\mathrm{d}y_1\mathrm{d}y_2|\,,$$

i.e.,

$$f(y_1, y_2) = p(x_1, x_2)\left|\frac{\partial(x_1, x_2)}{\partial(y_1, y_2)}\right| \tag{1.39b}$$

Let now $p(x)$ be uniformly distributed on $[0, 1]$, e.g., x has been calculated by a RNG $\Rightarrow p(x) = 1$. Then

$$\mathrm{d}x = |f(y)\mathrm{d}y| \quad \rightarrow \quad \int_0^x \mathrm{d}x' = \left|\int_{y_{\min}}^{y(x)} f(y)\mathrm{d}y\right|$$

$$\Rightarrow \qquad x = F(y) = \Bigg|\underbrace{\int_{y_{\min}}^{y} f(y')\mathrm{d}y'}_{\substack{\text{cumulative prob.}\\ \text{distribution}}}\Bigg| \qquad \text{with} \qquad F(y_{\min}) = 0,\ F(y_{\max} = y(1)) = 1.$$

Thus we find that

$$y = F^{-1}(x), \qquad \text{if } x \text{ is uniformly distributed.} \tag{1.40}$$

In summary, the inversion-(or transformation-) method requires that $F(y)$

- can be calculated analytically and
- can be inverted.

To create a variate which shall be distributed according to a given pdf, $f(y)$, we have to perform the following steps:

step 1: If $f(y)$ is not normalized, we have to normalized it at first, by replacing $f(y) \rightarrow C \cdot f(y)$ with

$$C = F(y_{\max})^{-1}$$

.

step 2: We derive $F(y)$.

step 3: By means of a RNG, a random number, $x \in [0, 1)$, has to be calculated.

step 4: We then equalize $F(y) =: x$ and

step 5: solve for $y = F^{-1}(x)$.

1.15 Example (1). y shall be drawn from a uniform distribution on $[a, b]$ (cf. Example 1.12).

$$f(y) = \frac{1}{b-a} \quad \Rightarrow \quad F(y) = \int_a^y \frac{1}{b-a}\mathrm{d}y = \frac{y-a}{b-a}$$

(test: $F(a) = 0$, $F(b) = 1$, ok). Then, $\dfrac{y-a}{b-a} = x$, and y has to be drawn according to the relation

$$y = a + (b-a)\cdot x, \qquad x \in [0, 1). \tag{1.41}$$

1.16 Example (2). y has to be drawn from an exponential distribution, $f(y) = \mathrm{e}^{-y}$, $y \geq 0$.

$$F(y) = \int_0^y \mathrm{e}^{-y'}\,\mathrm{d}y' = 1 - \mathrm{e}^{-y} =: x$$

$$\Rightarrow \qquad y = -\ln(1-x) \quad \rightarrow \quad y = -\ln x, \qquad \text{because } (1-x) \text{ is distributed as } x!$$

If one calculates

$$y = -\ln x, \qquad x \in [0,1), \tag{1.42}$$

then y is exponentially distributed!

Test: $f(y)\mathrm{d}y = \mathrm{e}^{-(-\ln(1-x))}\mathrm{d}y(x)$ and $\mathrm{d}y = \dfrac{1}{1-x}\mathrm{d}x \Rightarrow f(y)\mathrm{d}y = \dfrac{1-x}{1-x}\mathrm{d}x \stackrel{!}{=} p(x)\mathrm{d}x$.

1.17 Example (3). Finally, we like to draw y according to a *normal* distribution.

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

(here: mean 0, standard deviation = 1).

For this purpose, we use *two* random variables, x_1, x_2, uniformly distributed on $(0,1)$; let's consider then

$$y_1 = \sqrt{-2\ln x_1}\cos(2\pi x_2) \tag{1.43a}$$

$$y_2 = \sqrt{-2\ln x_1}\sin(2\pi x_2), \tag{1.43b}$$

i.e.,

$$y_1^2 + y_2^2 = -2\ln x_1 \qquad\qquad x_1 = \exp\left(-\frac{1}{2}(y_1^2 + y_2^2)\right)$$

$$\frac{y_2}{y_1} = \tan(2\pi x_2) \qquad\qquad 2\pi x_2 = \arctan\left(\frac{y_2}{y_1}\right)$$

With the transformation (1.39b) and $p(x_1, x_2) = 1$ we find

$$f(y_1, y_2) = \left|\frac{\partial(x_1, x_2)}{\partial(y_1, y_2)}\right| = \begin{vmatrix} \frac{\partial x_1}{\partial y_1} & \frac{\partial x_1}{\partial y_2} \\ \frac{\partial x_2}{\partial y_1} & \frac{\partial x_2}{\partial y_2} \end{vmatrix} = \left| -\frac{1}{\sqrt{2\pi}}\mathrm{e}^{-y_1^2/2} \cdot \frac{1}{\sqrt{2\pi}}\mathrm{e}^{-y_2^2/2} \right|.$$

Thus, $f(y_1, y_2)$ is the product of two functions which depend only on y_1 and y_2, respectively, and *both* variables are normally distributed,

$$f(y_1, y_2) = f(y_1) \cdot f(y_2).$$

Recipe: Draw two randon numbers x_1, x_2 from a RNG, then y_1 and y_2 as calculated by (1.43) are normally distribued!

1.18 Example (4). To solve exercise 2.3, calculate variates according to

$$p(\mu)\mathrm{d}\mu \sim \mu\mathrm{d}\mu, \qquad \mu = \cos\theta \geq 0,$$

which define the emission angle from the lower boundary in Example 1.13.

1.19 Example (5). In case that the pdf is tabulated, the generalization of the inversion method is straightforward. The integral $F(y)$ becomes tabulated as well, as a function of y (e.g., from a numerical integration on a certain grid with sufficient resolution), and the inversion can be performed by using interpolations on this grid.

Even if the inversion method cannot be applied (i.e., $\int f(y)\mathrm{d}y$ cannot be calculated analytically or $F(y)$ cannot be inverted and a tabulation method (see above) is discarded, there is no need to give up. Indeed, there exists a fairly simple procedure to calculate corresponding variates, called

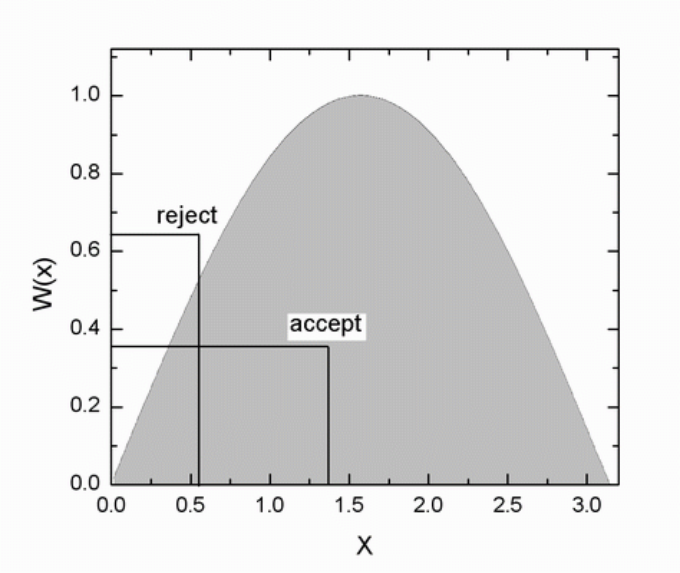

Figure 1.13: The most simple implementation of VON NEUMANN's rejection method (see text).

1.4.2.2 VON NEUMANN's rejection method (advanced reading)

which will be outlined briefly, at least regarding its most simplistic implementation. Fig. 1.13 shows the principle of this method. As always, the pdf, $w(x)$, must be normalized, and we enclose it by a rectangle which has a length corresponding to the definition interval, $[x_{\min}, x_{\max}]$, and a height which is larger/equal to the maximum of $w(x)$, $w_{\max}$. At first, we draw a pair of random numbers, (x_i, y_i), distributed *uniformly* according to the rectangle chosen (similar as in the "hit and miss" approach, Sect. 1.3.4). The random co-ordinates, of course, have to comply with the maximum extent in x and y (Eq. 1.41), i.e.,

$$x_i = x_{\min} + (x_{\max} - x_{\min}) \cdot x_1 \tag{1.44}$$

$$y_i = w_{\max} \cdot x_2, \tag{1.45}$$

if x_1, x_2 are pairs of consecutive random numbers drawn from a RNG. The statistical independence of x_1, x_2 is of highest imporance here!!! For this pair then, we *accept* x_i to be a variate distributed according to w, if

$$\text{accept } x_i: \quad y_i \leq w(x_i),$$

whereas otherwise we reject it!

$$\text{reject } x_i: \quad y_i > w(x_i).$$

If a value was rejected, another pair is drawn, and so on. Again, the number of "misses", e.g., of rejected variates, depends on the ratio of the area of rectangle and the area below w. The optimization of this method, which uses an additional *comparison function* to minimize this ratio is presented, e.g., in *Numerical Recipes*, Chap. 7.3.

Chapter 2

Praxis: Monte Carlo simulations and radiative transfer

2.1 Exercise 1: Simple RNGs and graphical tests

On the homepage of our course, you will find the program `rng_simple_ex1.f90`. Copy this program to a working-file of your convenience and use this file for your future work.
This program calculates five random numbers by means of the linear congruential RNG as applied in Fig. 1.5, i.e., with $M = 2048, A = 65$ and $C = 1$. In the following, you will investigate the 2-D correlation of random numbers obtained from this representative (low cycle!) RNG, as a function of A. Note that such RNGs with M being a power of two should create the full cycle if C is odd and $(A-1)$ is a multiple of 4.

a) At first, you should check the results from Fig. 1.5. For this purpose, perform the appropriate modifications of the program, i.e., create an output-file (to be called `ran_out`) with entries x_i, y_i for a sufficiently large sample, $i = 1 \ldots N$, which can be read and plotted with the IDL procedure `test_rng.pro`. Have a look into this procedure as well!
Convince yourself that *all* possible numbers are created (how do you do this?). Plot the distribution and compare with the manual (`ps`-figures can be created with the keyword `\ps`).

b) After success, modify the program in such a way that arbitrary values for A can be read in from the input (modifying C will not change the principal result). Display the results for different $A, A = 5, 9, \ldots 37$. Plot typical cases for large and small correlation. What do you conclude?

c) Finally, for the same number of N, plot the corresponding results obtained from the "minumum-standard generator" `ran1`, which is contained in the working directory as program file `ran1.f90`. Of course, you have to modify the main program accordingly and to re-compile *both* programs together. Compare with the results from above and discuss the origin of the difference.

2.2 Exercise 2: Monte Carlo integration - the PLANCK-function

One of the most important functions in astronomy is the PLANCK-fucntion, which, written in terms of the so-called *specific intensity* (which is a quantity closely related to the density of photons propagating into a certain direction and solid angle), is given by

$$B_\nu(T) = \frac{2h\nu^3}{c^2}\frac{1}{\mathrm{e}^{\frac{h\nu}{kT}}-1} \tag{2.1}$$

with frequency ν and temperature T. All other symbols have their usual meaning. The total intensity radiated from a black body can be calculated by integrating over frequency, resulting in the well-known STEFAN-BOLTZMANN-law,

$$\int_0^\infty B_\nu(T)\mathrm{d}\nu = \frac{\sigma_\mathrm{B}}{\pi}T^4, \tag{2.2}$$

where $\sigma_\mathrm{B} \approx 5.67 \cdot 10^{-5}$ (in cgs-units, $\mathrm{erg\,cm^{-2}s^{-1}grad^{-4}}$) is the BOLTZMANN-constant. Strictly speaking, this constant is no natural constant but a product of different natural constants and the value of the dimensionless integral,

$$\int_0^\infty \frac{x^3}{\mathrm{e}^x-1}\mathrm{d}x. \tag{2.3}$$

a) Determine the value (three significant digits) of this integral by comparing (2.1) with (2.2) and the given value of σ_B. (Hint: Use a suitable substitution for the integration variable in (2.1)). If you have made no error in your calculation (consistent units!), you should find a value close to the exact one, which is $\pi^4/15$ and can be derived, e.g., from an analytic integration over the complex plane.
From much simpler considerations (splitting the integral into two parts connected at $x = 1$), it should be clear that the integral must be of order unity anyway. As a most important result then, this exercise shows how simply the T^4-dependence arising in the STEFAN-BOLTZMAN law can be understood.

b) We will now determine the value of the integral (2.3) from a Monte Carlo integration, using random numbers as generated from `ran1`. For this purpose, use a copy of the program `mcint_ex2.f90` and perform appropriate modifications. Since the integral extends to infinity which cannot be simulated, use different maximum-values (from the input), to examine which value is sufficient. Use also different sample sizes, N, and check the $1/\sqrt{N}$-law of the MC integration. Compare with the exact value as given above.

2.3 Exercise 3: Monte Carlo simulation - limb darkening in plane-parallel, grey atmospheres

Limb darkening describes that fact that a (normal) star seems to be darker at the edge than at its center. This can be clearly observed for the sun. If, as in Example 1.13, the direction cosine (i.e., the cosine of angle between radius vector and photon direction) is denoted by μ, one can approximiate the angular dependence of the specific intensity (see below) by the expression

$$I(\mu) \approx I_1(0.4 + 0.6\,\mu) \tag{2.4}$$

where I_1 is the specific intensity for the radial direction, $\mu = 1$, i.e., the intensity observed at the center of a star. For $\mu = 0$, on the other hand, the angle between radial and photon's direction is 90^o (this situation is met at the limb of the star), and the corresponding region appears to be fainter, by a factor of roughly 0.4.
Eq. 2.4 is the consequence of an approximate solution of the equation of radiative transfer in plane-parallel symmetry[1], with absorption and emission processes assumed to be *grey*, i.e., frequency independent, which has been developed (long ago!) by EDDINGTION and BARBIER. Note that the approximate solution of *Milne*'s integral equation (Example 1.13) is based on the same approach. Just one additional comment: The above relation (2.4) has nothing to do with the presence of any temperature stratification (though it can be influenced by it), but is the consequence of a large number of absorption and emission processes in an atmosphere of finite (optical) depth, as we will see below.

In order to avoid almost all subtleties of radiative transfer and corresponding approximate solutions[2], in this exercise we will perform a Monte Carlo simulation to confirm the above result. The principle strategy is very close to the simulation as described in Example 1.13, but we will sort the photon's now according to the direction they leave the atmosphere, and not according to their energy.

a) The principle structure of the simulation (to be used again with `ran1`) is given in program `limb_ex3.f90`, which has to be copied to a working file and then complemented, according to the comments given in the program. The "only" part missing is the actual path of the photons. If you have studied Sect. 1.4 carefully, you should be able to program this path within few statements.

At first, develope a sketch of the program flow including all possible branches (a so-called flow-chart), accounting for appropriate distributions of "emission angle" (Example 1.18), optical path length and scattering angle. Update always the *radial* optical depth of the photon, according to the comments in the program, and follow the photons until they have left the atmosphere. In the latter case then, update the corresponding array counting the number of photons which have escaped under a certain range of angles. Before implementing this program, discuss your flow-chart with your supervisor.

b) Implement the algorithm developed in a) into your program and test it using 20 chanels for the angular distribution, $N = 10^4 \ldots 10^5$ photons and an optical depth of the atmosphere, $\tau_{rmmax} = 10$. The angular distribution created can be displayed via the IDL procedure `limb_test.pro`. If your program runs successfully, you should have obtained a plot as given in Fig. 2.1.

[1] i.e., under the condition that the stellar photosphere is very thin compared to the stellar radius: the solar photosphere, e.g., is only a few hundred kilometers thick, contrasted to the sun's radius of 700 000 km.

[2] Again, the interested reader may have a look into Mihalas, D., 1978, *Stellar atmospheres*, Freeman, San Francisco, or http://www.usm.uni-muenchen.de/people/puls/stellar_at/stellar_at.html, Chap. 3 and 4).

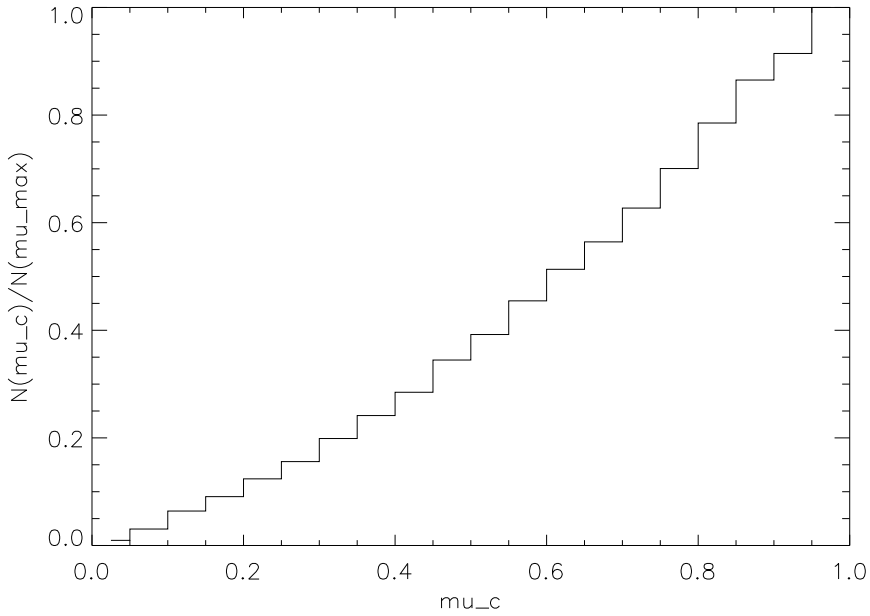

Figure 2.1: Monte Carlo simulation of limb darkening: angular distribution of 10^5 photons sorted into 20 channels. The total optical depth of the atmosphere is 10.

c) In our simulation, we have calculated the *number* of photons leaving the atmosphere with respect to a surface perpendicular to the radial direction. Without going into details, this number is proportional to the specific intensity weighted with the projection angle μ, since the specific intensity, $I(\mu)$, is defined with respect to unit *projected area*. Thus,

$$N(\mu)\mathrm{d}\mu \propto I(\mu)\mu\mathrm{d}\mu, \tag{2.5}$$

and the intensity within $\mathrm{d}\mu$ is obtained from the number of photons, divided by an appropriate average of μ, i.e., centered at the mid of the corresponding channel (already implemented into the program output).

This relation is also the reason why the distribution of the photons' "emission angles" at the lower boundary follows the pdf $\mu\mathrm{d}\mu$: for an isotropic radiation field, which is assumed to be present at the lowermost boundary, it is the specific intensity and not the photon number which is uniformly distributed with respect to μ! Thus, in order to convert to photon *numbers*, we have to draw the emission angles at the lower boundary from the pdf $\mu\mathrm{d}\mu$ instead of $\mathrm{d}\mu$! Inside the atmosphere, on the other hand, the emission angle refers to the photons themselves and thus is (almost) isotropic, so that we have to draw from the pdf $\mathrm{d}\mu$.

Modify the plot routine `limb_test.pro` in such a way as to display the specific intensity and compare with the prediction (2.4). Use $N = 10^6$ photons for $\tau_{\max} = 10$ now, and derive the limb-darkening coefficients (in analogy to Eq. 2.4) from a linear regression to your results. Hint: use the IDL-procedure `poly_fit`, described in `idlhelp`. Why is a certain discrepancy between simulation and theory to be expected?

d) To convince yourself that limb-darkening is the consequence of a multitude of scattering effects, reduce $\tau_{\max}$ to a very small value. Which angular distribution do you expect now for the specific intensity? Does your simulation verify your expectation?

Index