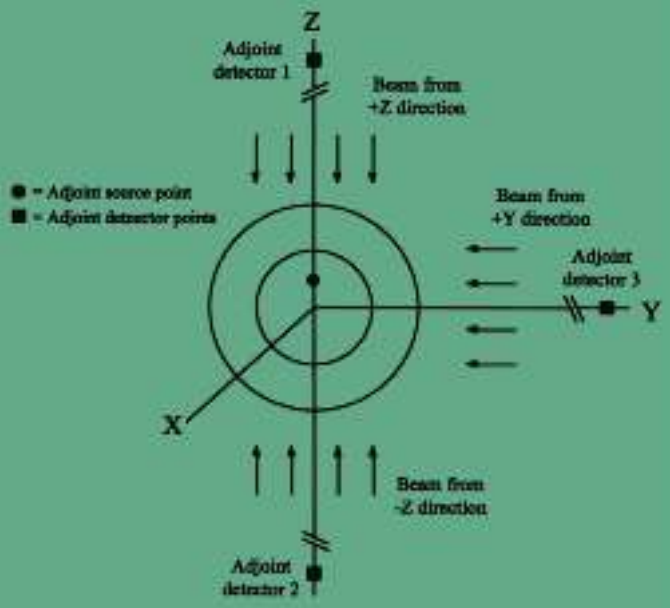
A MONTE CARLO PRIMER

A Practical Approach to Radiation Transport



Stephen A. Dupree
and
Stanley K. Fraley

A Monte Carlo Primer

A Monte Carlo Primer

A Practical Approach to Radiation Transport

Stephen A. Dupree
and
Stanley K. Fraley
Sandia National Laboratories
Albuquerque, New Mexico

SPRINGER SCIENCE+BUSINESS MEDIA,LLC

ISBN 978-1-4613-4628-9 ISBN 978-1-4419-8491-2 (eBook)
DOI 10.1007/978-1-4419-8491-2

©2002 Springer Science+Business Media New York
Originally published by Kluwer Academic / Plenum Publishers, New York in 2003
Softcover reprint of the hardcover 1st edition 2002

http://www.wkap.com

10 9 8 7 6 5 4 3 2 1

A C.I.P. record for this book is available from the Library of Congress

All rights reserved

No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, microfilming, recording, or otherwise, without its prior permission from the Publisher

To Pat and Huri

Preface

The mathematical technique of Monte Carlo, as applied to the transport of sub-atomic particles, has been described in numerous reports and books since its formal development in the 1940s. Most of these instructional efforts have been directed either at the mathematical basis of the technique or at its practical application as embodied in the several large, formal computer codes available for performing Monte Carlo transport calculations. This book attempts to fill what appears to be a gap in this Monte Carlo literature between the mathematics and the software. Thus, while the mathematical basis for Monte Carlo transport is covered in some detail, emphasis is placed on the application of the technique to the solution of practical radiation transport problems. This is done by using the PC as the basic teaching tool.

This book assumes the reader has a knowledge of integral calculus, neutron transport theory, and Fortran programming. It also assumes the reader has available a PC with a Fortran compiler. Any PC of reasonable size should be adequate to reproduce the examples or solve the exercises contained herein. The authors believe it is important for the reader to execute these examples and exercises, and by doing so to become accomplished at preparing appropriate software for solving radiation transport problems using Monte Carlo. The step from the software described in this book to the use of production Monte Carlo codes should be straightforward. However, instead of using these production codes as "black boxes" that must be taken on faith, the reader should be able to understand the functions being performed by the various components of the production software, and in most cases be capable of modifying or expanding it to suit the needs of a particular calculation.

Mathematical procedures can be expressed in many different ways using high-level computer languages. Fortran is no exception to this generality. Thus there is no "best" way to write a Fortran procedure, and many individual styles, all of which can produce equally valid results, are possible. In our examples we employ executable statements based on Fortran-90, but have often retained the style of older versions of Fortran. We have done this because of our particular experience in using various versions of Fortran. Hopefully readers will be able to follow the constructs easily and to re-cast them into a form compatible with their preferences. To save space we have frequently placed several executable statements in a line of coding.

We have adopted several conventions in the presentation of the text and examples. Fortran subroutines and variable names are designated by single quotation marks. Numerical exponents are expressed either by using the base 10 with a superscript or using the convention of '+' or '-' to designate the exponent; i.e., 1.0+5 = 1.0E+5 = 1.0×10^5. Tabulated results are rounded, with the number of significant digits shown being that which seems appropriate for the example in hand. All calculations that involve using computed results to obtain additional results use the values prior to rounding. Therefore, if the derived results are reproduced using the rounded values presented in the tables the numbers obtained may differ from those cited in the text. Because compilers and execution speeds differ, the user should not expect to reproduce the problem run times cited in this book. However, the relative efficiencies of a series of calculations should be similar regardless of the compiler and computer used.

We gratefully acknowledge the assistance of our friends and colleagues who reviewed the manuscript. Kevin O'Brien and Jim Renken provided thorough and detailed reviews of the entire text, and Eleanor Walther reviewed the appendix. In all cases their comments were well considered and informative, and they saved us from numerous omissions, obfuscations, and errors. Special thanks go to Len Connell and Tom Laub, who not only reviewed the text but also devoted considerable effort to checking and using the Monte Carlo routines. Long before the manuscript was completed Len Connell was the first user of the probabilistic framework code. Tom Laub provided independent solutions and numerous corrections to the example problems. The time and efforts of these reviewers are much appreciated and this book has greatly benefited from their support. Needless to say, any errors remaining in the text are the responsibility of the authors.

Stephen A. Dupree
Stanley K. Fraley
Albuquerque, NM
July 2001

Contents

Chapter 5. A Probabilistic Framework Code

Chapter 6. Variance Reduction Techniques

Chapter 7. Monte Carlo Detectors

Chapter 1

Introduction

1.1 The Monte Carlo Method

The Monte Carlo method can be used to solve a wide range of physical and mathematical problems. Its utility has increased with the general availability of fast computing machines, and new applications are continually forthcoming. However, the basic concepts of Monte Carlo are both simple and straightforward, and can be learned by using a personal computer. In this book we will use such a computer as the basis for developing and explaining the fundamental concepts of Monte Carlo as applied to neutral particle transport. As each topic is addressed a corresponding set of software instructions will be developed. The software that results will be assembled into a program configuration that is representative of a full-scale Monte Carlo radiation transport program. The components of the program will be explained and combined in a fashion that will allow the reader to understand the function and contribution of each to the final, and sometimes daunting, whole.

The Monte Carlo method is a technique of numerical analysis that is based on the use of sequences of random numbers to obtain sample values for the problem variables. The calculational process used in Monte Carlo is an artificial construct, usually a computer program that is mathematically equivalent to the problem being analyzed. Sample values for the problem variables are obtained by selecting specific numbers from appropriate ranges for the variables in the problem using probability distributions for such variables. The desired solution can be obtained, along with estimates of uncertainties in the solution, by analyzing the results from the sample values. The sample evaluation in a Monte Carlo calculation is somewhat equivalent to conducting an experiment. Both an experiment and a Monte

Carlo calculation will yield a result that is a possible, or representative, outcome of the process being modeled, and both contain uncertainties that can often be reduced by repeated measurements and quantified by the use of statistical analysis.

The variety of problems to which the Monte Carlo technique can be applied can only be suggested here. However, to indicate the utility of the method, and to assist the reader in developing practical skills in applying Monte Carlo or using any of the existing, standardized Monte Carlo transport computer codes, we will make use of example problems throughout the text to illustrate and expand the principles and application of the technique.

The use of random processes to solve mathematical problems has been known in one form or another for some time. For example, in the 18th century, the well-known naturalist and mathematician Georges-Louis Leclerc, Comte de Buffon, developed an experimental method of obtaining the ratio of the diameter to the circumference of a circle; i.e., of determining the reciprocal of π.[1] This experiment attained a modicum of notoriety, as reported in 1873 by Hall.[2] Buffon's experiment consisted of dropping a needle randomly onto a surface on which was drawn a series of equally spaced parallel lines. If the length of the needle is equal to half of the distance between the lines, then the probability p that the needle touches one of the lines is

$$p = 1 / \pi \tag{1.1}$$

If the needle is tossed randomly onto the plane of parallel lines n times, and it is found by observation to intersect a line k times, then

$$p \approx k/n \tag{1.2}$$

By the weak law of large numbers, this expression becomes exact in the limit of large n,

$$\lim_{n \to \infty} \frac{k}{n} = \frac{1}{\pi} \tag{1.3}$$

The basis for this result is as follows. If parallel lines are spaced a distance d apart, and a needle of length $\lambda \leq d$ is tossed randomly among them, then the angle between the needle and the lines is random over $[0,\pi]$. The position of the center of the needle, x_0, is randomly distributed over the interval [0,d] in a coordinate x that is perpendicular to the parallel lines, as measured from the nearest "bottom" line (see Figure 1.1.)

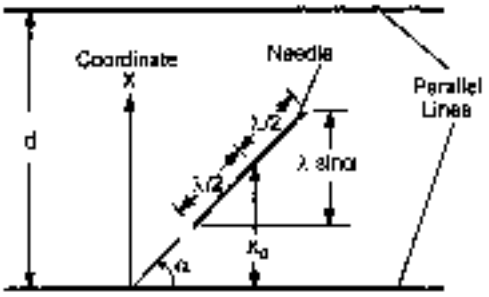


Figure 1.1. Coordinate System for Buffon's Problem

The projected length of the needle onto the coordinate x is $\lambda \sin\alpha$, where α is the angle between the needle and the parallel lines. If $x_c < 0.5\lambda \sin\alpha$, or if $x_c > d - 0.5\lambda \sin\alpha$, the needle will intersect a line. Thus two distinct regions in the space of α and x can be defined as shown in Figure 1.2: the region D in which the needle does not touch a line, and the region L in which it does touch a line. The probability p that the needle will touch a line is the ratio of the area of L to the total area D + L = πd. Thus

$$p = \frac{L}{D+L} = \frac{2}{\pi d}\int_0^{\pi}\frac{\lambda}{2}\sin\alpha\, d\alpha = \frac{2\lambda}{\pi d} \qquad (1.4)$$

If the needle is half as long as the lines are apart, $\lambda = d/2$ and, by equation (eqn) 1.4, $p = 1/\pi$.

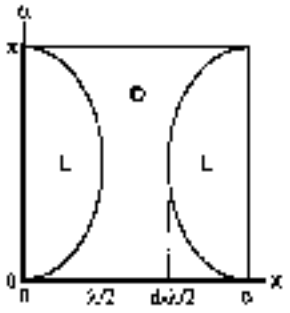


Figure 1.2. Regions L and D in Buffon's Problem

Significant uses of random sampling techniques occurred early in the twentieth century. Lord Kelvin applied what today would be called the Monte Carlo technique to the Boltzmann equation.[2] The statistician Student used the technique of random sampling to assist him in the determination of the correlation coefficient.[3] Student also used sampling methods in studies of the t-distribution. However, the practical use of calculations involving

random variables, and the coining of the name "Monte Carlo," was begun at Los Alamos during the Manhattan Project of World War II by John von Neumann and Stanley Ulam. The Los Alamos researchers introduced the use of variance reduction, particularly the techniques known as Russian roulette and splitting,[5] to increase the efficiency with which random variables can be evaluated and solutions obtained.

The original Monte Carlo calculations performed at Los Alamos were executed using slide rules and mechanical calculators, with the "random" numbers being obtained by several different methods. Availability and improvement of computers following World War II greatly extended the use of the Manhattan Project Monte Carlo technique. The first set of random numbers[6] and the first lengthy monograph on Monte Carlo methods[7] were published in the mid-1950s. Since that time the Monte Carlo method has been extended to numerous areas of science and technology and has been used as an analysis or engineering design tool in many fields of research. Today it is a standard mathematical tool applied to complex problems not tractable by other methods and is the method of choice in certain applications.

1.2 The Evaluation of Random Processes

As defined in the previous section, the Monte Carlo technique consists of mathematical procedures that evaluate random variables using random numbers. Typically these random variables are part of a series of parameters that must be evaluated to obtain the solution to a problem. That is, the solution requires evaluation of a random, or stochastic, process by which the random variables in the problem take on specific values. A random variable is a measurable quantity associated with a random experiment. For example, a numerical quantity associated with an event in a game of chance is a random variable. A random variable can be discrete, such as the point resulting from the throw of a pair of dice, or continuous, such as the final position of a pointer on a spinning wheel. The overriding characteristic of a random variable is that it cannot be predicted deterministically and acquires a value only after a procedure is conducted to select a value, or sample, from the range, or sample space, available to the variable.

In order to obtain practical solutions to problems involving random variables a sequence of evaluations of the random variables must be made. Suppose one wished to produce a possible point sequence from the throwing of a pair of dice. Dice can actually be used to generate such a sequence experimentally. However, this may not always be convenient, and there may be a question of whether a particular pair of dice is unbiased or is being

thrown in an unbiased manner. Furthermore, unlike the throwing of dice, for many problems it may be impossible to evaluate the random variables experimentally. In these cases, calculations must be used to produce the desired sequence of values.

As an example let us consider a class of bolts that has been certified to a certain torque. While this certification may be useful information, it provides no knowledge of the failure point of any specific bolt. Such knowledge can only be obtained by testing the bolt to failure. Even then, if one could reconstruct the failed bolt exactly as it was and re-test it, one would expect to measure a different failure point, because of changes in the measuring instrument if for no other reason. A series of tests on random samples of bolts, or a series of properly constructed Monte Carlo calculations could be used to predict, within some limited accuracy, the fraction of the members of the class that, on the average, would fail at a given torque. Thus specific, quantitative knowledge about the population of bolts is possible, but even with such knowledge one still has no way of predicting the failure point of the next bolt removed from the bin. Just as this type of uncertainty is a feature of the real world, it is a feature of the Monte Carlo method.

Repeated estimates of a random variable can provide information about the variable that cannot be determined in a single estimate. For example, a single throw of a die provides little information about the numbers on the faces of the die. From a single throw one could not determine whether the die was "fair" – i.e., whether the values one through six were each represented on the faces of the die and whether they appeared with equal probability – or whether the die was biased. To determine the numbers that are on the faces of the die without conducting a visual examination, numerous samples, or throws of the die, would be required, along with careful evaluation of the data to provide confidence in the result.

The postulated sequence of integers required to simulate the throwing of a die or the breaking of a bolt could be obtained in a number of ways, such as flipping a coin to form a sequence, or "string," of binary digits, or picking numbers from the Manhattan telephone directory. Both of these methods have been used successfully to evaluate random variables. However, modern digital computers provide a more convenient method of obtaining a string of randomly distributed numbers.

A random number is a numerical quantity that is selected from a uniform distribution of numbers between some limits. The boundaries of this interval are not important since any interval can be obtained from another by a simple coordinate transformation. However, most random number generators use the interval from zero to one, usually exclusive of one or both end points. As discussed in the Appendix, random number generators are mathematical algorithms that produce "random" numbers from simple

calculations typically involving the use of low order bits in digital computer words. The number strings produced by such mathematical algorithms are called pseudorandom because the string of numbers they produce can be reproduced at will. However, the sequence of a good quality random number generator will exhibit a reasonable degree of randomness. Randomness in this application means that sequential numbers are uncorrelated and, in the limit as many numbers are selected, the density of numbers is uniformly distributed over some interval.

To reiterate, random processes can be evaluated by modeling the physical process involved or by solving equations that describe the process. The direct, or analog method of solution, in which the physical process itself is modeled, is usually the easiest to understand since the steps involved in the solution follow the steps in the physical process. The solution of mathematical models is the more flexible and useful application, however, since such models are not constrained to physical processes. Mathematical Monte Carlo also lends itself readily to the incorporation of variance reduction methods. Such methods can greatly reduce the time required to achieve a result of satisfactory accuracy in a Monte Carlo calculation.

Example 1.1 Predicting the average outcome of a physical process

By observation you have determined that a certain child learning to walk on a balance beam takes an average of ten steps on the beam before falling; i.e., the ratio of steps to falls is 10 to 1. Therefore, assuming a constant probability of falling per step taken – that the child does not tire or become distracted with success and therefore is as likely to fall on the first step as on any other step in a sequence – the probability of falling is 0.1 per step. Assume the child requires five steps to traverse the beam. Determine the average number of times the child will successfully complete five steps, and therefore walk the entire length of the beam, out of ten starts.

This problem is not deterministic; one cannot predict with certainty the actual number of successes in ten specific attempts to walk the beam. Instead the number of successes will vary in a random fashion around some average, sometimes equaling the average and sometimes being more or less than the average. Thus there is a probability p_0 that the child will succeed zero times in ten tries, a probability p_1 that the child will succeed one time, etc., up to a probability p_{10} that the child will succeed all ten times. From the theory of *a priori* probability for mutually exclusive events, the sum of these individual probabilities must be 1.0 since every possibility has been included and every set of ten tries must have some outcome. Monte Carlo can be used to obtain an estimate of each of these eleven probabilities and thereby "solve" the problem.

Our approach will be to simulate the child walking the balance beam and then count the number of successful walks as a fraction of the total number of trials. By repeating this simulation a large number of times an estimate can be obtained of both the average number of times out of ten tries the child will succeed and the distribution of the number of successes about this average. The latter estimate is particularly important since without it one is unable to evaluate the precision of the answer. When evaluating a random variable, often the only method available for estimating the variance of the mean is to obtain multiple estimates, compute the average, and then estimate the variance of this average based on the distribution of the estimates. It should be noted that a Monte Carlo estimate of a value without an associated uncertainty is essentially meaningless. It is similar to a statement that, "People weigh 157.3 pounds, because that is what I measured when I weighed somebody once."

The Monte Carlo method of solution assumes there is an unlimited supply of random numbers available in the range (0,1). These random numbers will be used one-by-one to decide whether an event, which has a certain probability of occurring, will be evaluated as having occurred. For example, if an event has a 0.1 chance of occurring under a particular circumstance, then a random number that is selected from the interval (0,1) and has a value less than 0.1 could indicate the event occurred, and a random number (selected from the same interval) greater than or equal to 0.1 could indicate that the event did not occur.

In a formal sense, the question of whether to include the single point on the boundary between domains in a numerical space for a random number in the "lower" of "upper" portion of the space is irrelevant. That is, in the above example, inclusion of the point for which the random number equals 0.1 in the domain representing an event, instead of the absence of the event, makes no difference in the result. Because there are an unlimited number of points on a line between any two values of the coordinate along the line – i.e., because the number of numerical values between any two different numbers is unbounded - the probability of selecting a particular point, or numerical value, such as 0.1, from the continuum of values available to a truly random number, is zero.

However, in practice, the word length of a computer is finite. Therefore the probability of selecting a particular value from the available set of numbers between two limits is not zero. Nevertheless, good quality random number generators use a high order of precision (at least 32 bits) and the probability of selecting one particular value is small. In this case the definition of domain boundaries is of little consequence. On the other hand, when a small number of significant digits is used for the random number sequence, as will be the case in the present example, it becomes important that the correct end of the range of the variable be included in the definition

of each domain. We assume the present random number generator covers the range [0,1), and hence can take on the value of zero but not of one. It would seem reasonable, therefore, that for this example the point 0.1 be included in the "upper" domain and if the random number generator produces a value of 0.1 we will assume the event has not occurred.

In the present problem the Monte Carlo process assumes the child attempts to take a first step. The child has a probability of 0.1 of falling off the balance beam in attempting this first step. Thus if the first random number picked is less than 0.1 the simulation assumes the child has fallen, otherwise the child takes a second step. If the second random number is less than 0.1 then the child has fallen while taking the second step, otherwise the child takes a third step, etc. If after completing the fifth step the child has not fallen – if five consecutive random numbers in the string have values greater than or equal to 0.1 – then the child has successfully walked the beam and the scoreboard is marked with a one. If the child fails to complete five steps in sequence, the child's scoreboard is marked with a zero and a new trial is begun.

The scores may be tabulated after every series of ten trics, with the number of successful trics in that series tallied as an estimate of the desired answer. After simulating many such series of ten tries, the scores can be averaged to estimate the mean number of successes per ten tries. Alternatively, each of the simulated trials can be considered individually and the probability of success per trial determined. The "expected," or average, number of successes per ten trials will then be ten times this value. In general many (100 or more) series should be calculated in order to obtain a reasonably accurate estimate of the answer.

A string of random numbers can be obtained from many pocket calculators. One such string is shown to two significant digits in Table 1.1. To solve the current problem, we will look for five consecutive numbers in the random number string greater than or equal to 0.1. Thus we step through the simulation, using the random numbers in the sequence in which they arise, in order to obtain the desired answer.

From Table 1.1 one can follow the sequence of simulations in the following manner. Starting with the first random number we find five consecutive values greater than or equal to 0.1; therefore, the child succeeds in walking the full length of the beam on the first try and gets a score of one. Likewise all the random numbers six through ten are greater than or equal to 0.1 so the child also succeeds on the second try. In fact, the first random number less than 0.1 is the 16^{th} entry in the table, so the first three trials are successful while the fourth trial fails on the first step. Therefore after four tries we find three successes and one failure, giving a success rate of 0.75.

Table 1.1. A Random Number Set from a Pocket Calculator

Seq. #	Rand	Seq. #	Rand	Seq. #	Rand	Seq. #	Rand
1	0.10	26	0.57	51	0.54	76	0.09
2	0.71	27	0.97	52	0.51	77	0.17
3	0.32	28	0.26	53	0.27	78	0.62
4	0.35	29	0.06	54	0.92	79	0.94
5	0.87	30	0.22	55	0.48	80	0.07
6	0.28	31	0.49	56	0.22	81	0.31
7	0.36	32	0.32	57	0.34	82	0.97
8	0.31	33	0.84	58	0.54	83	0.41
9	0.26	34	0.70	59	0.91	84	0.12
10	0.93	35	0.15	60	0.24	85	0.14
11	0.56	36	0.65	61	0.79	86	0.08
12	0.34	37	0.78	62	0.78	87	0.14
13	0.52	38	0.28	63	0.53	88	0.30
14	0.11	39	0.91	64	0.03	89	0.76
15	0.51	40	0.88	65	0.18	90	0.81
16	0.01	41	0.09	66	0.79	91	0.70
17	0.45	42	0.09	67	0.71	92	0.95
18	0.11	43	0.34	68	0.81	93	0.90
19	0.08	44	0.35	69	0.38	94	0.65
20	0.05	45	0.94	70	0.96	95	0.07
21	0.31	46	0.02	71	0.19	96	0.15
22	0.42	47	0.31	72	0.61	97	0.53
23	0.67	48	0.03	73	0.43	98	0.57
24	0.40	49	0.58	74	0.28	99	0.07
25	0.71	50	0.40	75	0.98	100	0.92

If we proceed through the set of random numbers in Table 1.1 we find 13 successes in the 27 trials simulated by the first 99 numbers in the table. These results are summarized in Table 1.2. The calculated probability of success is therefore $13/27 \approx 0.48$, which agrees intuitively with the premise of the child falling roughly every ten steps. Thus this small sample appears to provide a fairly reasonable average success rate.

One can easily solve this problem exactly. Because the probability of the child falling off the beam at any step is $P_f = 0.1$ the probability of not falling on the same step is $P_{nf} = 1 - P_f = 0.9$. Therefore the probability of the child completing exactly one step is P_{nf} times the probability of falling on the second step, P_f, and $P_1 = P_f P_{nf}$. The probability of completing exactly two steps is $P_2 = P_f P_{nf}^2$. The probability of the child completing exactly n steps successfully (for $n < 5$ because the child covers the length of the beam in five steps), is

$$P_n = P_f P_{nf}^n \tag{1.5}$$

Table 1.2. Results of Example 1.1

Trial Number	Steps Taken Successfully	Score	Trial Number	Steps Taken Successfully	Score
1	5	1	15	5	1
2	5	1	16	5	1
3	5	1	17	5	1
4	0	0	18	0	0
5	2	0	19	5	1
6	0	0	20	5	1
7	5	1	21	1	0
8	3	0	22	3	0
9	5	1	23	5	1
10	5	1	24	0	0
11	1	0	25	5	1
12	0	0	26	3	0
13	3	0	27	3	0
14	1	0			

That is,

$$P_n = 0.1(0.9)^n \tag{1.6}$$

for values of n from 0 through 4. The probability of the child completing the five steps required to walk the beam is one minus the sum of the probabilities of falling on any step, or the sum of the P_n for $n = 0$ through 4,

$$P_5 = (0.9)^5 \approx 0.59 \tag{1.7}$$

since the child cannot fall on the sixth step, there being no sixth step. One would thus expect about 16 successes in 27 tries whereas our short Monte Carlo calculation produced 13 successes in 27 tries.

It is clear that this Monte Carlo estimate contains some degree of error. If our result were normally distributed, a concept we will discuss in Chapter 2, the standard deviation of our estimate would be the square root of the number of successes and we would say that the average number of successes per 27 tries is 13 ± 3.6. Normalizing to ten trials, as specified in the problem definition, we would estimate the number of successes per ten trials to be 4.8 ± 1.3. From eqn 1.7 the exact value is slightly larger than 5.9, which is within our estimated standard error.

There is more information in the results shown in Table 1.2 than this simple average. For example, we have coincidentally produced an estimate of the probability of the child falling after zero through four steps, as well as the probability of completing the walking of the beam. To obtain these

estimates we need merely count the number of times each result was obtained in the simulation. Access to these results was a simple matter of recording the relevant data as the calculation proceeded. In a computer calculation one may either record the entire sequence of events in the simulation, as was done in Table 1.2, record some relevant portion of that sequence, or record just the event data one wishes to use. In the latter case the entire sequence of events would have to be recalculated to score alternative results. Planning for the scoring is desirable in Monte Carlo and will be discussed in Chapter 7.

Table 1.2 shows that the child fell off the beam on the second step three times during the 27 trials conducted. This gives a probability per trial of $0.11 \pm .06$ of falling on the second step. From eqn 1.6 we see the correct probability is 0.09. Because of the small number of times this result will occur, relative to the successful walking of the beam (which has a probability of 0.59), many more simulations of the type considered here would be required to improve the accuracy of the Monte Carlo estimate for the probability of falling on the second step than are included in Table 1.2. However, if one were interested in obtaining an accurate estimate of this quantity while running as few simulations as possible, one could attempt to do so using variance reduction, as will be discussed in Chapter 6.

. . .

Example 1.2. Another Monte Carlo estimate of π

Consider a circle of unit radius with its center at the origin. We wish to estimate the area within the portion of the circle that is in the positive quadrant; i.e., the area of the portion of the circle in the quarter space in which both x and y are positive. In a sense this problem is similar to Buffon's Monte Carlo determination of $1/\pi$ since this area is, of course, proportional to π. Thus this result will provide us with an additional method to estimate π.

A Monte Carlo solution for the area can be made using the following process. A pair of random numbers, each distributed uniformly over the range (0,1), is used to select a random point in the square defined by $0 \le x \le 1$ and $0 \le y \le 1$. The point so selected is then examined to see whether it is inside or outside the unit circle centered at the origin. If the point is outside the unit circle it is rejected; i.e., it is not included in the tally of points found to be inside the circle. If the point is inside the circle the result is included in the tally. This process is repeated many times to obtain a number of points inside the circle. By comparing this tally with the total number of points evaluated one can obtain an estimate of the probability of a point in the unit square, selected randomly, being inside the circle. That is, the ratio of the number of points not rejected, which are those inside the circle, to the

total number of trials is equal to the ratio of the area of the quadrant of a unit circle ($\pi/4$) to the area of the unit square (1.0). Thus the ratio is an estimate of the quantity $\pi/4$. This method of solution is known as the rejection technique.

This problem involves more than the mere selection and counting of random numbers as was done in Example 1.1. Here we must compare the location of the point selected with the arc of the unit circle to determine whether the point is inside or outside the circle. Thus, after we select the point (x,y), we must find whether $x^2 + y^2 < 1$. This would be tedious to calculate by hand but is simple to program on a computer. A sample Fortran program to determine the ratio of points inside the circle to total points selected is shown in Table 1.3.

The function 'fltrn.' used in the program of Table 1.3, returns a pseudorandom number in the range (0,1). The starting point in the random number string could be set by the subroutine 'rndin.' The argument of 'rndin' must be an integer from one through 2^{31}-2. The purpose of using 'rndin' would be to enable the user to repeat a sequence of pseudorandom numbers should this be desired in order to check or debug a calculation, or to resume a calculation without repeating any part of the random number string already used. These concepts, along with the operation of the random number generator and the routines associated with it, are described in the Appendix. The user may substitute another random number generator, such as the one supplied by the Fortran compiler being used, but should be aware that such generators may be of poor quality. Finally, the random number generator is not shown explicitly in the Fortran examples in this book, and the appropriate routines from the Appendix, or elsewhere, must be linked to the executable files.

The program shown in Table 1.3 consists essentially of a 'do' loop that retrieves two sequential random numbers, 'x' and 'y,' and tests whether the resulting point (x,y) is inside the unit circle. If so a score is added to the running tally, 'npi,' and the square of the score is added to a second tally called 'sumsq.' The latter, as we shall see, enables us to estimate the uncertainty in the result. As given, the program uses 10^9 points and calculates the expected value of π, as well as the standard deviation of the expected value, on the basis of these points.

The results of executing the program in Table 1.3 are shown in Table 1.4. This table provides the results for several different sample sizes. The approach to the correct value of π is evident as the number of samples increases. From Table 1.4 it is clear that, using this unbiased rejection calculation, almost 10^9 points must be selected in order to obtain an estimate of π accurate to five significant digits. Even allowing for the speed of modern computers this is not a very efficient method of calculating π.

sample content of A Monte Carlo Primer: A Practical Approach to Radiation Transport

- click The Wee Free Men (Discworld, Book 30)
- read Plato and Heidegger: A Question of Dialogue pdf, azw (kindle)
- The Digital Doctor: Hope, Hype, and Harm at the Dawn of Medicineâ€™s Computer Age pdf, azw (kindle), epub
- read online The Beautiful Edible Garden: Design A Stylish Outdoor Space Using Vegetables, Fruits, and Herbs
- **Blackhatonomics: An Inside Look at the Economics of Cybercrime here**
- download online The Encyclopedia Of American Business History (Almanacs of American Life) pdf

- http://nexson.arzamaszev.com/library/The-Wee-Free-Men--Discworld--Book-30-.pdf
- http://dadhoc.com/lib/Hamlet--The-Annotated-Shakespeare-.pdf
- http://aseasonedman.com/ebooks/The-Flesh-of-Images--Merleau-Ponty-between-Painting-and-Cinema--SUNY-series-in-Contemporary-Continental-Philosop
- http://www.experienceolvera.co.uk/library/Die-Laughing--Daisy-Dalrymple-Mysteries--Book-12-.pdf
- http://schrolf.de/books/Blackhatonomics--An-Inside-Look-at-the-Economics-of-Cybercrime.pdf
- http://www.1973vision.com/?library/Vaccine--The-Debate-in-Modern-America.pdf