Finding Definite Integral Using Monte Carlo Method

A project thesis

Submitted in Partial Fulfillment of the Requirement for the

Award of the Degree of

B.Sc(Hon's) in Mathematics

By

Muhaiminul Islam Rakib

Examination Roll No:- B110302044

Session: 2011-12

Under the supervision of

A. B. S. Manik Munsi (Lecturer)

Department of Mathematics
Faculty of Science
Jagannath University,
Dhaka-1100, Bangladesh

Date: February 25, 2017

DEPARTMENT OF MATHEMATICS

JAGANNATH UNIVERSITY, DHAKA-1100

BANGLADESH

Recommendation of supervisor

This is to certify that the present project work entitled 'Finding definite integral using Monte Carlo method' has been carried out by Roll No. B110302044 under my direct supervision at Department of Mathematics, Jagannath University, Dhaka-1100, Bangladesh, I recommend that the prepared project thesis can be accepted in partial fulfillment of the requirements for the degree of B.sc.(Hon's) in Mathematics.

Supervisor

A. B. S. Manik Munsi

Lecturer, Department of Mathematics
Jagannath University, Dhaka-1100
Bangladesh

ACKNOWLEDGEMENT

At first I express my greatest and deepest attitude to the almighty, the supreme of the universe, to whom all praises go for enabling me to complete my research work.

I would like to express my gratitude, appreciation, deepest sense and indebtedness to my honorable teacher and respective supervisor **A. B. S. Manik Munsi** , lecturer, department of mathematics, Jagannath University, Dhaka for his willingness to accept me as a research student, who offered me to carry out my thesis work on this interesting topic and also has aided by his patience, motivation, enthusiasm, constructive criticism, immense knowledge and endless encouragement during the completion of the project report.

I would like to convey my gratitude to Dr. Md. Rezaul Karim, Professor and chairman, Dr. Payer Ahmed, Professor and Dr. Ayub Ali, Professor, Department of Mathematics, Jagannath University, Dhaka for their cordial love, spontaneous help and encouragement that played a significant role in completion of this project thesis.

Thanks also to the teachers of the discipline for their moral support in preparing this project thesis.

I would also express my sincere appreciation for the encouragements received from my elder brothers and friends for their all kinds of help during my research work.

I want to extent my gratitude and appreciation towards my respected parents for supporting me spiritually throughout my life.

Above all I am lucky to get him as my supervisor and for this I am very much grateful to the supreme God who has enabled me to reach a successful end and also wish for my supervisor longevity.

ABSTRACT

An interesting application of random numbers consists in calculating the integrals with the so-called Monte Carlo method. The range of problems for which the Monte Carlo method was developed is very wide, including, for example, solving the linear equations systems, the inversion of matrices, finding the proper vectors and values of a matrix, the calculation of the simple and multiple integrals, solving certain problems at the limit from the field of differential equations, etc.

In this project paper, we have discussed basic concepts of Monte Carlo Method as well as the origin and brief history of Monte Carlo method. In chapter one, we have discussed the introductory things of Monte Carlo method. In chapter two, I focused on the process of Monte Carlo method and simulation. After that, I have discussed about numerical integration through Monte Carlo method. Here, I used C programming language to solve these integrals. Finding area through Monte Carlo method that is finding definite integral was also discussed and shown by graphically. In chapter four, I have shown the various application of the Monte Carlo method.

Contents

1.1 What is Monte Carlo Method?

The Monte Carlo method is a numerical solution to a problem that models objects interacting with other objects or their environment based upon simple object-object or object-environment relationships. It represents an attempt to model nature through direct simulation of the essential dynamics of the system in question. In this sense the Monte Carlo method is essentially simple in its approach- a solution to a microscopic system through simulation of its microscopic interactions.

Monte Carlo methods are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. Their essential idea is using randomness to solve problems that might be deterministic in principle.

They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to use other approaches.

The presupposes that all uses of the Monte Carlo are for the purposes of understanding physical phenomena. There are others uses of Monte Carlo method for purely mathematical reasons, such as the determination of multi-dimensional integrals. Often these integrals are motivated by physical models. However, there are examples where the motivation is entirely mathematical in which case our definition of the Monte Carlo method would be generalized somewhat.

A solution is determined by random sampling of the relationships, or the microscopic interactions, until the result converges. Thus the mechanics of executing a solution involves repetitive action or calculation. To the extent that many microscopic interactions can be modeled mathematically, the repetitive solution can be executed on a computer. However, the Monte Carlo method predates the computer (more on this later) and is not essential to carry out a solution although in most cases computers make the determination of a solution much faster.

The Monte Carlo method, also called Monte Carlo analysis, is a means of statistical evaluation of mathematical functions using random samples. This requires a good source of random numbers. There is always some error involved with this scheme, but the larger the number of random samples taken, the more accurate the result.

In its pure mathematical form, the Monte Carlo method consists of finding the definite integral of a function by choosing a large number of independent-variable samples at random from within an interval or region, averaging the resulting dependent-variable values, and then dividing by the span of the interval or the size of the region over which the random samples were chosen. This differs from the classical method of approximating a definite integral, in which independent-variable samples are selected at equally-spaced points within an interval or region.

The Monte Carlo method is most famous for its use during the Second World War in the design of the atomic bomb. It has also been used in diverse applications, such as the analysis of traffic flow on superhighways, the development of models for the evolution of stars, and attempts to predict fluctuations in the stock market. The scheme also finds applications in integrated circuit (IC) design, quantum mechanics, and communications engineering.

Monte Carlo method, mathematical or statistical method of understanding complex physical or mathematical systems by using randomly generated numbers as input into those systems to generate a range of solutions. The likelihood of a particular solution can be found by dividing the number of times that solution was generated by the total number of trials. By using larger and larger numbers of trials, the likelihood of the solutions can be determined more and more accurately. The Monte Carlo method is used in a wide range of subjects, including mathematics, physics, biology, engineering, and finance, and in problems in which determining an analytic solution would be ... (100 of 262 words)

1.2 Roles of Monte Carlo simulation in mathematics:

Monte Carlo methods provide approximate solutions to a variety of mathematical problems by performing statistical sampling experiments. They can be loosely de- fined as statistical simulation methods, where statistical simulation is defined in quite general terms to be any

method that utilizes sequences of random numbers to perform the simulation. Thus Monte Carlo methods are a collection of different methods that all basically perform the same process. This process involves performing many simulations using random numbers and probability to get an approximation of the answer to the problem. The defining characteristic of Monte Carlo methods is its use of random numbers in its simulations. In fact, these methods derive their collective name from the fact that Monte Carlo, the capital of Monaco, has many casinos and casino roulette wheels are a good example of a random number generator.

The Monte Carlo simulation technique has formally existed since the early 1940s, where it had applications in research into nuclear fusion. However, only with the increase in computer technology and power has the technique become more widely used. This is because computers are now able to perform millions of simulations much more efficiently and quickly than before. This is an important factor because it means that the technique can provide an approximate answer quickly and to a higher level of accuracy, because the more simulations that you perform then the more accurate the approximation is (This point is illustrated in the next section when we compare approximation error for different numbers of simulations).

Note that these methods only provide approximation of the answer. Thus the analysis of the approximation error is a major factor to take into account when evaluating answers from these methods. The attempt to minimize this error is the reason there are so many different Monte Carlo methods. The various methods can have different levels of accuracy for their answers, although often this can depend on certain circumstances of the question and so some method's level of accuracy varies depending on the problem. This is illustrated well in the next section where we investigate four different Monte Carlo methods and compare the answers and the accuracy of their approximations.

1.3 Basic Description:

Monte Carlo methods provide approximate solutions to a variety of mathematical problems by performing statistical sampling experiments. They can be loosely de- fined as statistical simulation methods, where statistical simulation is defined in quite general terms to be any

method that utilizes sequences of random numbers to perform the simulation. Thus Monte Carlo methods are a collection of different methods that all basically perform the same process. This process involves performing many simulations using random numbers and probability to get an approximation of the answer to the problem. The defining characteristic of Monte Carlo methods is its use of random numbers in its simulations. In fact, these methods derive their collective name from the fact that Monte Carlo, the capital of Monaco, has many casinos and casino roulette wheels are a good example of a random number generator. The Monte Carlo simulation technique has formally existed since the early 1940s, where it had applications in research into nuclear fusion. However, only with the increase in computer technology and power has the technique become more widely used. This is because computers are now able to perform millions of simulations much more efficiently and quickly than before. This is an important factor because it means that the technique can provide an approximate answer quickly and to a higher level of accuracy, because the more simulations that you perform then the more accurate the approximation is (This point is illustrated in the next section when we compare approximation error for different numbers of simulations). Note that these methods only provide a approximation of the answer. Thus the analysis of the approximation error is a major factor to take into account when evaluating answers from these methods. The attempt to minimize this error is the reason there are so many different Monte Carlo methods. The various methods can have different levels of accuracy for their answers, although often this can depend on certain circumstances of the question and so some method's level of accuracy varies depending on the problem. This is illustrated well in the next section where we investigate four different Monte Carlo methods and compare the answers and the accuracy of their approximations.

Monte Carlo simulations define a method of computation that uses a large number of random samples to obtain results. They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to use other mathematical methods. Monte Carlo methods are mainly used in three distinct problem classes: optimization, numerical integration, and generating draws from probability distribution.

Monte Carlo simulations are often used when the problem at hand has a probabilistic component. An expected value of that probabilistic component can be studied using Monte Carlo due to the law of large numbers. With enough data, even though it's sampled randomly, Monte Carlo can

hone in on the truth of the problem. For example, we can estimate the value of pi by simply throwing random needles into a circle inscribed within a square that is drawn on the ground. After many needles are dropped, one quadrant of the circle is then examined. The ratio of the number of needles that are inside the square to the number of needles inside the circle is a very good approximation of pi. The more needles that are dropped the closer the approximation gets.

1.4 Why is Monte Carlo?

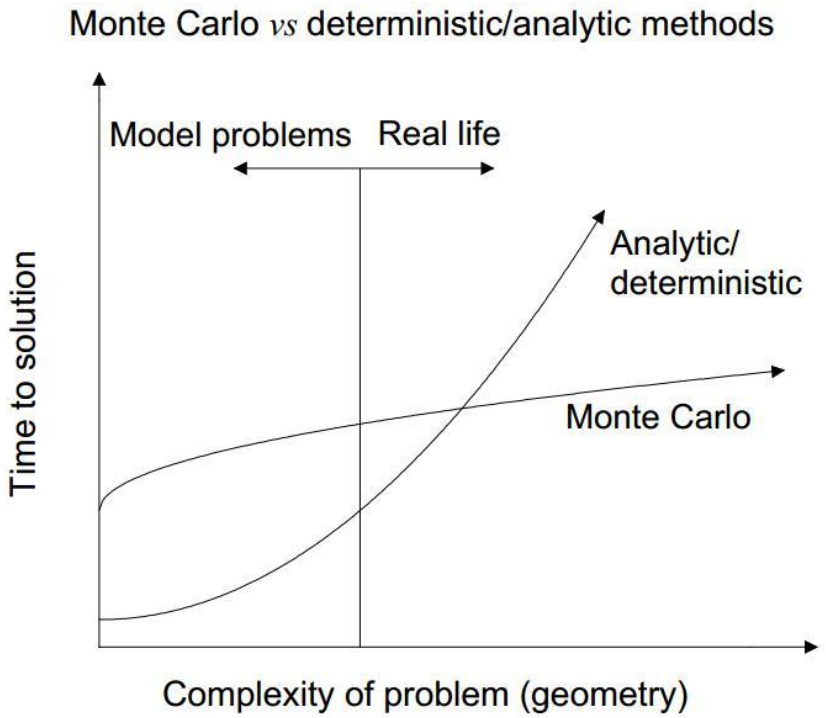
If Monte Carlo did not exist there would be strong motivation to invent it! As argued previously, the products of both basic and applied science are dependent upon the trinity of measurement, theory and Monte Carlo. Monte

Carlo is often seen as a "competitor" to other methods of macroscopic calculation, which we will call deterministic and/or analytic methods. Although the proponents of either method sometimes approach a level of fanaticism in their debates, a practitioner of science should first ask, "What do I want to accomplish?" followed by "What is the most efficient way to do it?"

Sometimes the correct answer will be "Deterministic" and other times it will be "Monte Carlo". The most successful scientist will avail himself or herself of more than one avenue attack on a problem.

There are, however, two inescapable realities. The first is that macroscopic theory, particularly transport theory, provides deep insight and allows one to develop sophisticated intuition as to how macroscopic particle fields can be expected to behave. Monte Carlo cannot compete very well with this. In discovering the properties of macroscopic field behavior, Monte Carloists operate very much like experimentalists. Without theory to provide guidance the process of discovery is trial and error, guided perhaps, by some brilliant intuition.

However, when it comes to complexity of a problem, however that is measured, Monte Carlo techniques become advantageous as the complexity of a problem increases. This idea is expressed in Figure



The other inescapable reality is that computers are getting faster and cheaper at a geometric rate. This is known as Moore's Law.
Another factor weighing in favor of Monte Carlo is that the Monte Carlo technique is one based upon a minimum amount of data and a maximum amount of floating-point operation. Deterministic calculations are often maximum data and minimal floating-point operation procedures. Since impediments to data processing are often caused by communication bottlenecks, either from the CPU to cache memory, cache memory to main memory or main memory to large storage devices (typically disk), modern computer architecture favors the Monte Carlo model which emphasizes iteration and minimizes data storage. Although the concluding remarks of this section seem to favor the Monte Carlo approach, a point made previously should be re-iterated. Analytic theory development and its realizations in terms of deterministic calculations are our only way of making theories regarding the behavior of macroscopic fields, and our only way of modeling particle fluencies in a symbolic way. Monte Carlo is simply

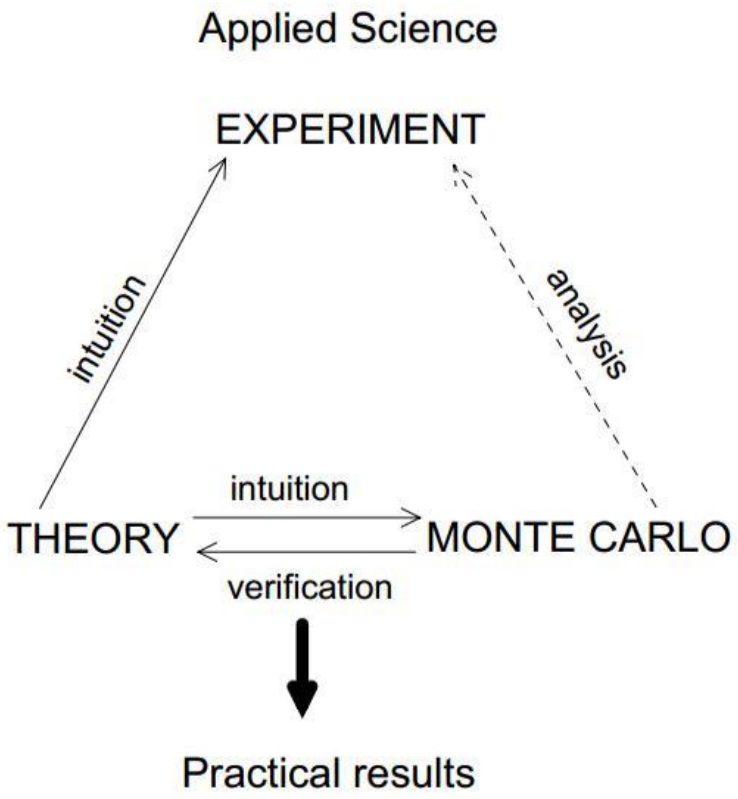
another tool in the theoretician's or the experimentalist's toolbox. The importance of analytic development must never be understated.

1.5 The Origin of the Monte Carlo Method

The generally accepted birth date of the Monte Carlo method is 1949, when an article entitled "The Monte Carlo method" by Metropolis and Ulam appeared. The American mathematicians John von Neumann and Stanislav Ulam are considered its main originators. In the Soviet Union, the first papers on the Monte Carlo method were published in 1955 and 1956 by V. V. Chavchanidze, Yu. A. Shreider and V. S. Vladimirov.

Curiously enough, the theoretical foundation of the method had been known long before the von Neumann-Ulam article was published. Furthermore, well before 1949 certain problems in statistics were sometimes solved by means of random sampling - that is, in fact, by the Monte Carlo method. However, because simulation of random variables by hand is a laborious process, use of the Monte Carlo method as a universal numerical technique became practical only with the advent of computers.

As for the name "Monte Carlo," it is derived from that city in the Principality of Monaco famous for its, casinos. The point is that one of the simplest mechanical devices for generating random numbers is the roulette wheel. But it appears worthwhile to answer here one frequently asked question: "Does the Monte Carlo method help one win at roulette?" The answer is No; it is not even an attempt to do so.



1.6 A Brief History of the Monte Carlo Method

The idea of Monte Carlo calculation is a lot older than the computer. The name "Monte Carlo" is relatively recent—it was coined by Nicolas Metropolis in 1949 - but under the older name of "statistical sampling" the method has a history stretching back well into the last century, when numerical calculations were performed by hand using pencil and paper and perhaps a slide-rule. As first envisaged, Monte Carlo was not a method for solving problems in physics, but a method for estimating integrals which could not be performed by other means. Integrals over poorly-behaved functions and integrals in high-dimensional spaces are two areas in which the method has traditionally proved profitable, and indeed it is still an important technique for problems of these types. A famous early example of this type of calculation is the experiment known as "Buffon's needle" (Dörrie 1965) in which the mathematical constant is determined by repeatedly dropping a needle onto a sheet of paper ruled with evenly spaced lines. The experiment is named after Georges-Louis Leclerc, Comte de Buffon who in 1777 was the first to show that if we throw a needle of completely at random onto a sheet of paper ruled with lines a

distance apart, then the chances that the needle will fall so as to intersect one of the lines is, provided that 1. It was Laplace in 1820 who then pointed out that if the needle is thrown down times and is observed to land on a line of those times, we can make an estimate of from

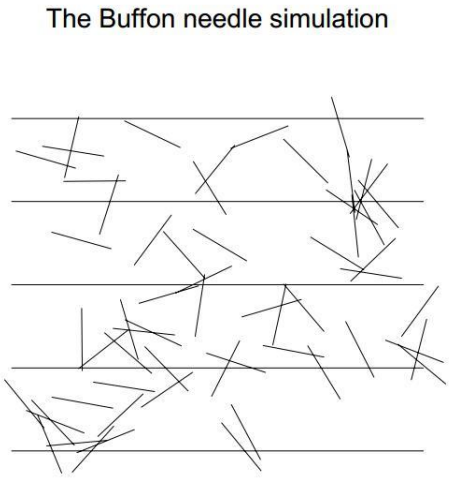


Figure : *A computer simulation of the Buffon needle problem*

A number of investigators made use of this method over the years to calculate approximate values for. The most famous of these is Mario Lazzarini, who in 1901 announced that he had calculated a value of 3.1415929 for from an experiment in which a needle was dropped times onto a sheet of paper ruled with lines apart. This value, accurate to better than three parts in ten million, would be an impressive example of the power of the statistical sampling method were it not for the fact that it is almost certainly faked. Badger (1994) has demonstrated extremely convincingly that, even supposing Lazzarini had the technology at his disposal to measure the length of his needle and the spaces between his lines to a few parts in (a step necessary to ensure the accuracy of above Equation), still the chances of his finding the results he did were poorer than three in a million; Lazzarini was imprudent enough to publish details of the progress

of the experiment through the castings of the needle, and it turns out that the statistical "fluctuations"' in the numbers of intersections of the needle with the ruled lines are much smaller than one would expect in a real experiment. All indications are that Lazzarini forged his results. However, other, less well known attempts at the experiment were certainly genuine, and yielded pastime amongst Europe's intellectual gentry.

With the advent of mechanical calculating machines at the end of the nineteenth century, numerical methods took a large step forward. These machines increased enormously the number and reliability of the arithmetic operations that could be performed in a numerical "experiment", and made the application of statistical sampling techniques to research problems in physics a realistic possibility for the first time. An early example of what was reasonable figures for (Wolf 1850), (Smith 1855). Apparently, performing the Buffon's needle experiment was for a while quite a sophisticated effectively a Monte Carlo calculation of the motion and collision of the molecules in a gas was described by William Thomson (later Lord Kelvin) in 1901. Thomson's calculations were aimed at demonstrating the truth of the equipartition theorem for the internal energy of a classical system. However, after the fashion of the time, he did not perform the laborious analysis himself, and a lot of the credit for the results must go to Thomson's secretary, William Anderson, who apparently solved the kinetic equations for more; than five thousand molecular collisions using nothing more than a pencil and a mechanical adding machine.

Aided by mechanical calculators, numerical methods, particularly the method of finite differences, became an important tool during the First World War. The authors recently heard the intriguing story of the Herculean efforts of French mathematician Henri Soudée, who in 1916 calculated firing tables for the new cannons being set up at Verdun, directly from his knowledge of the hydrodynamic properties of gases. The tables were used when the cannons were brought to bear on the German-occupied Fort de Douaumont, and as a result the fort was taken by the allies. Soudée was later honoured by the French. By the time of the Second World War the mechanical calculation of firing angles for large guns was an important element of military technology. The physicist Richard Feynman tells the story of his employment in Philadelphia during the summer of 1940 working for the army on a mechanical device for predicting the trajectories of planes as they flew past

(Feynman 1985). The device was to be used to guide anti-aircraft guns in attacking the planes. Despite some success with the machine, Feynman left the army's employ after only a few months, joking that the subject of mechanical computation was too difficult for him. He was shrewd enough to realize he was working on a dinosaur, and that the revolution of electronic computing was just around the corner. It was some years however before that particular dream would become reality, and before it did Feynman had plenty more chance to spar with the mechanical calculators. As a group leader during the Manhattan Project at Los Alamos he created what was effectively a highly pipelined human CPU, by employing a large number of people armed with Marchant mechanical adding machines in an arithmetic assembly line in which little cards with numbers on were passed from one worker to the next for processing on the machines. A number of numerical calculations crucial to the design of the atomic bomb were performed in this way.

The first real applications of the statistical sampling method to research problems in physics seem to have been those of Enrico Fermi, who was working on neutron diffusion in Rome in the early 1930s. Fermi never published his numerical methods—apparently he considered only the results to be of interest, not the methods used to obtain them— but according to his influential student and collaborator Emilio Segre those methods were, in everything but name, precisely the Monte Carlo methods later employed by Ulam and Metropolis and their collaborators in the construction of the hydrogen bomb (Segre 1980).

So it was that when the Monte Carlo method finally caught the attention of the physics community, it was again as the result of armed conflict. The important developments took place at the Los Alamos National Laboratory in New Mexico, where Nick Metropolis, Stanislaw Ulam and John von Neumann gathered in the last months of the Second World War shortly after the epochal bomb test at Alamagordo, to collaborate on numerical calculations to be performed on the new ENIAC electronic computer, a mammoth, room-filing machine containing some triode valves, whose

construction was nearing completion at the University of Pennsylvania. Metropolis A980) has remarked that the technology that went into the ENIAC existed well before 1941, but that it took the pressure of America's entry into the war to spur the construction of the machine. It seems to

have been Stan Ulam who was responsible for reinventing Fermi's statistical sampling methods. He tells of how the idea of calculating the average effect of a frequently repeated physical process by simply simulating the process over and over again on a digital computer came to him whilst huddled over a pack of cards, playing patience one day. The game he was playing was "Canfield" patience, which is one of those forms of patience where the goal is simply to turn up every card in the pack, and he wondered how often on average one could actually expect to win the game. After abandoning the hopelessly complex combinatorics involved in answering this question analytically, it occurred to him that you could get an approximate answer simply by playing a very large number of games and seeing how often you win. With his mind never far from the exciting new prospect of the ENIAC computer, the thought immediately crossed his mind that he might be able to get the machine to play these games for him far faster than he ever could himself, and it was only a short conceptual leap to applying the same idea to some of the problems of the physics of the hydrogen bomb that were filling his work hours at Los Alamos. He later described his idea to John von Neumann who was very enthusiastic about it, and the two of them began making plans to perform actual calculations. Though Ulam's idea may appear simple and obvious to us today, there are actually many subtle questions involved in this idea that a physical problem with an exact answer can be approximately solved by studying a suitably chosen random process. It is a tribute to the ingenuity of the early Los Alamos workers that, rather than plunging headlong into the computer calculations, they considered most of these subtleties right from the start.

The war ended before the first Monte Carlo calculations were performed on the ENIAC. There was some uncertainty about whether the Los Alamos laboratory would continue to exist in peacetime, and Edward Teller, who was leading the project to develop the hydrogen bomb, was keen to apply the power of the computer to the problems of building the new bomb, in order to show that significant work was still going on at Los Alamos. Von Neumann developed a detailed plan of how the Monte Carlo method could be implemented on the ENIAC to solve a number of problems concerned with neutron transport in the bomb, and throughout 1947 worked with Metropolis on preparations for the calculations. They had to wait to try their ideas out however, because the ENIAC was to be moved from Philadelphia where it was built to the army's Ballistics Research Laboratory in Maryland. For a modern computer this would not, be a problem, but for the gigantic ENIAC, with its thousands of fragile components, it was a difficult

task, and there were many who did not believe the computer would survive the journey. It did, however, and by the end of the year it was working once again in its new home. Before von Neumann and the others put it to work on the calculations for the hydrogen bomb, Richard Clippinger of the Ballistics Lab suggested a modification to the machine which allowed it to store programs in its electronic memory. Previously a program had to be set up by plugging and unplugging cables at the front of the machine, an arduous task which made the machine inflexible and inconvenient to use. Von Neumann was in favor of changing to the new "stored program" model, and Nick Metropolis and von Neumann's wife Klari, made the necessary modifications to the computer themselves. It was the end of 1947 before the machine was at last ready, and Metropolis and von Neumann set to work on the planned Monte Carlo calculations.

The early neutron diffusion calculations were an impressive success, but Metropolis and von Neumann were not able to publish their results, because they were classified as secret. Over the following two years however, they and

others, including Stan Ulam and Stanley Frankcl, applied the new statistical sampling method to a variety of more mundane problems in physics, such as the calculation of the properties of hard-sphere gases in two and three dimensions, and published a number of papers which drew the world's attention to this emerging technique. The 1949 paper by Metropolis and Ulam on statistical techniques for studying integro-differential equations is of interest because it contained in its title the first use of the term "Monte Carlo" to describe this type of calculation. Also in 1949 the first conference on Monte Carlo methods was held in Los Alamos, attracting more than a hundred participants. It was quickly followed by another similar meeting in Gainesville, Florida.

The calculations received a further boost in 1948 with the arrival at Los Alamos of a new computer, humorously called the MANIAC. (Apparently the name was suggested by Enrico Fermi, who was tiring of computers with contrived acronyms for names—he claimed that it stood for "Metropolis and Neumann Invent Awful Contraption"'. Nowadays, with all our computers called things like XFK-23/z we would no doubt appreciate a few pronounceable names.) Apart from the advantage of being in New Mexico rather than Maryland, the MANIAC was a significant technical improvement over the ENIAC which Prosper Eckert (1980), its principal architect, refers to as a "hastily built first try". It was faster and contained a larger

memory 40 kilobits, or 5 kilobytes in modern terms). It was built under the direction of Metropolis, who had been lured back to Los Alamos after a brief stint on the faculty at Chicago by the prospect of the new machine. The design was based on ideas put forward by John von Neumann and incorporated a number of technical refinements proposed by Jim Richardson, an engineer working on the project. A still more sophisticated computer, the MANIAC 2, was built at Los Alamos two years later, and both machines remained in service until the late fifties, producing a stream of results, many of which have proved to be seminal contributions to the field of Monte Carlo simulation. Of particular

note to us is the publication in 1953 of the paper by Nick Metropolis, Marshall and Arianna Rosenbluth, and Edward and Mici Teller, in which they describe for the first time the Monte Carlo technique that has come to be known as the Metropolis algorithm. This algorithm was the first example of a thermal "importance sampling" method and it is to this day easily the most widely used such method. Also of interest are the Monte Carlo studies of nuclear cascades performed by Antony Turkevich and Nick Metropolis, and Edward Teller's work on phase changes in interacting hard-sphere gases using the Metropolis algorithm.

The exponential growth in computer power since those early days is by now a familiar story to us all, and with this increase in computational resources Monte Carlo techniques have looked deeper and deeper into the subject of statistical physics. Monte Carlo simulations have also become more accurate as a result of the invention of new algorithms. Particularly in the last twenty years, many new ideas have been put forward, of which we describe a good number in the rest of this book.

2. Monte Carlo method process

2.1 Monte Carlo method process:

1. Pick a space of possible samples.
2. Randomly sample in that space using a probability distribution

3. Perform defined operations on the random samples.
4. Aggregate the data.

The Monte Carlo process can be a little difficult to accept. If we're just randomly gathering data, how can that help us find a correct answer?

Let's think about a coin. We want to know if the coin is fair or not. A fair coin has these properties

$$p(heads) = \frac{1}{2} \qquad p(tails) = \frac{1}{2}$$

If you flip a coin just *one* time, what will happen? It will either land on heads *or* tails. If you flip a coin and it lands on tails, and then you walk away, have you proven that $p(tails) = \frac{1}{2}$ and that the coin is not fair? Of course not! A fair coin can also land on tails after one try, after all.

Maybe you flip the coin again and it lands on heads, thus proving that $(heads) = p(tails) = \frac{1}{2}$, and that the coin is fair. But have you actually proven that the coin is fair? Maybe the coin isn't completely fair but it does land on both sides sometimes.

Maybe you flip tails *again*, and you're even more convinced that $p(tails) = 1$. Sadly, we still don't know because both a fair coin and a false coin have the ability to flip tails twice in a row.

The point is that we need to flip this coin a lot before we can be convinced that it's fair. Let's see this with some code. In the following Python snippet, there's a function that takes in the probability that a coin flips a heads (it's o.5 if the coin is fair) and a number of times to flip the coin. It then returns a guess as to the fairness of the coin (again, it's 0.5 if it's fair).

What is simulation and when it is used?

Simulation is the imitation of the operation of a real-world process or system over time. The act of simulating something first requires that a model be developed; this model represents the key characteristics or behaviors/functions of the selected physical or abstract system or process.
Simulation is **used** when conducting experiments on a real system would be impossible or impractical: for example, because of the high cost of prototyping and testing, or because the fragility of the system will not support extensive tests, or because of the duration of the experiment in real time is impractical.
Simulation is used in many contexts, such as simulation of technology for performance optimization, tasting, training, education, and video games .Often, computer experiments are used to study simulation models. Simulation is also used with scientific modeling of natural systems or human systems to gain insight into their functioning. Simulation can be used to show the eventual real effects of alternative conditions and courses of action. Simulation is also used when the real system cannot be engaged, because it may not be accessible, or it may be dangerous or unacceptable to engage, or it is being designed but not yet built, or it may simply not exist.

Simulation is a decision analysis and support tool. Simulation software allows you to evaluate, compare and optimize alternative designs, plans and policies. As such, it provides a tool for explaining and defending decisions to various stakeholders.
Simulation should be used when the consequences of a proposed action, plan or design cannot be directly and immediately observed (i.e., the consequences are delayed in time and/or dispersed in space) and/or it is simply impractical or prohibitively expensive to test the alternatives directly. For example, when implementing a strategic plan for a company, the impacts are likely to take months (or years) to materialize.
Simulation is particularly valuable when there is significant uncertainty regarding the outcome or consequences of a particular alternative under consideration. Probabilistic simulation allows you

deal with this uncertainty in a quantifiable way.

The ability to define what may happen in the future and to choose among alternatives lies at the heart of contemporary societies. – [Peter Bernstein, Against the Gods: The Remarkable Story of Risk]

Our knowledge of the way things work, in society or nature, comes trailing clouds of vagueness. Vast ills have followed a belief in certainty.
- Kenneth Arrow (Nobel Laureate, Economics, 1972)

2.2 How Monte Carlo simulation works:

Monte Carlo simulation performs risk analysis by building models of possible results by substituting a range of values—a probability distribution—for any factor that has inherent uncertainty. It then calculates results over and over, each time using a different set of random values from the probability functions. Depending upon the number of uncertainties and the ranges specified for them, a Monte Carlo simulation could involve thousands or tens of thousands of recalculations before it is complete. Monte Carlo simulation produces distributions of possible outcome values.

By using probability distributions, variables can have different probabilities of different outcomes occurring. Probability distributions are a much more realistic way of describing uncertainty in variables of a risk analysis. Common probability distributions include:

Normal – Or "bell curve." The user simply defines the mean or expected value and a standard deviation to describe the variation about the mean. Values in the middle near the mean are most likely to occur. It is symmetric and describes many natural phenomena such as people's heights. Examples of variables described by normal distributions include inflation rates and energy prices.

Lognormal – Values are positively skewed, not symmetric like a normal distribution. It is used to represent values that don't go below zero but have unlimited positive potential. Examples of variables described by lognormal distributions include real estate property values, stock prices, and oil reserves.

Uniform – All values have an equal chance of occurring, and the user simply defines the minimum and maximum. Examples of variables that could be uniformly distributed include manufacturing costs or future sales revenues for a new product.

Triangular – The user defines the minimum, most likely, and maximum values. Values around the most likely are more likely to occur. Variables that could be described by a triangular distribution include past sales history per unit of time and inventory levels.

PERT- The user defines the minimum, most likely, and maximum values, just like the triangular distribution. Values around the most likely are more likely to occur. However values between the most likely and extremes are more likely to occur than the triangular; that is, the extremes are not as emphasized. An example of the use of a PERT distribution is to describe the duration of a task in a project management model.

Discrete – The user defines specific values that may occur and the likelihood of each. An example might be the results of a lawsuit: 20% chance of positive verdict, 30% change of negative verdict, 40% chance of settlement, and 10% chance of mistrial.

During a Monte Carlo simulation, values are sampled at random from the input probability distributions. Each set of samples is called an iteration, and the resulting outcome from that sample is recorded.

Monte Carlo simulation does this hundreds or thousands of times, and the result is a probability distribution of possible outcomes. In this way, Monte Carlo simulation provides a much more comprehensive view of what may happen. It tells you not only what could happen, but how likely it is to happen.

Monte Carlo simulation provides a number of advantages over deterministic, or "single-point estimate" analysis:

Probabilistic Results:

Results show not only what could happen, but how likely each outcome is.

Graphical Results:

Because of the data a Monte Carlo simulation generates, it's easy to create graphs of different outcomes and their chances of occurrence. This is important for communicating findings to other stakeholders.

Sensitivity Analysis:

With just a few cases, deterministic analysis makes it difficult to see which variables impact the outcome the most. In Monte Carlo simulation, it's easy to see which inputs had the biggest effect on bottom-line results.

Scenario Analysis:

In deterministic models, it's very difficult to model different combinations of values for different inputs to see the effects of truly different scenarios. Using Monte Carlo simulation, analysts can see exactly which inputs had which values together when certain outcomes occurred. This is invaluable for pursuing further analysis.

Correlation of Inputs:

In Monte Carlo simulation, it's possible to model interdependent relationships between input variables. It's important for accuracy to represent how, in reality, when some factors goes up, others go up or down accordingly. An enhancement to Monte Carlo simulation is the use of Latin Hypercube sampling, which samples more accurately from the entire range of distribution functions. Monte Carlo simulation performs risk analysis by building models of possible results by substituting a range of values—a probability distribution—for any factor that has inherent uncertainty. It then calculates results over and over, each time using a different set of random values from the probability functions. Depending upon the number of uncertainties and the ranges specified for them, a Monte Carlo simulation could involve thousands or tens of thousands of recalculations before it is complete. Monte Carlo simulation produces distributions of possible outcome values.

CHAPTER 3

Numerical Integration using MC

3.1 Calculating the value of pi using Monte Carlo:

Here I want to present one of the simplest cases of the Monte Carlo to demonstrate its power - Calculating the value of π. First it is useful to imagine a square dartboard with an imaginary circle inscribed with diameter equal to the side length of the square. If numerous darts are randomly thrown at the board we can see that there is some probability that a dart will land in either the circle or outside of it. In this case, we know the length of the square's side, therefore we can calculate the area of each region. Now, we can postulate that the probability of a dart landing in the circle is equal to the ratio of the area of the circle compared to the total are

$$P = \frac{\pi(\frac{a^2}{4})}{a^2}$$

Where we find that the value of π can be calculated as:

$$\pi = 4P$$

One method to estimate the value of π (3.141592...) is by using a Monte Carlo method. In the demo above, we have a circle of radius 0.5, enclosed by a 1 × 1 square. The area of the circle is

$$\pi r^2 = \frac{\pi}{4}$$

the area of the square is 1. If we divide the area of the circle, by the area of the square we get

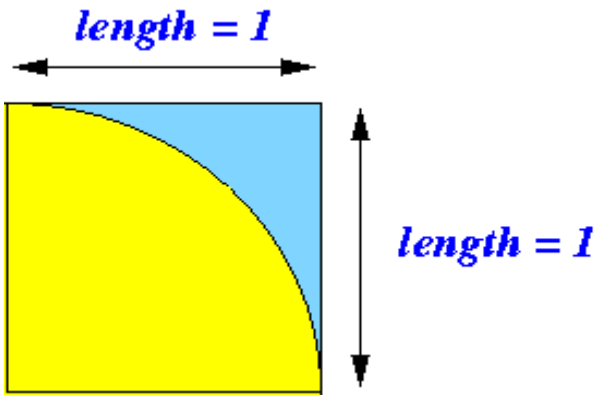
$$\frac{\pi}{4}$$

We then generate a large number of uniformly distributed random points and plot them on the graph. These points can be in any position within the square i.e. between (0, 0) and (1, 1). If they fall within the circle, they are colored red, otherwise they are colored blue. We keep track of the total number of points, and the number of points that are inside the circle. If we divide the number of points within the circle, N_{INNER} by the total number of points N_{TOTAL}, we should get a value that is an approximation of the ratio of the areas we calculated above,

$$\frac{\pi}{4}$$

In other words,

$$\frac{\pi}{4} \approx \frac{\text{Ninner}}{N_{total}}$$



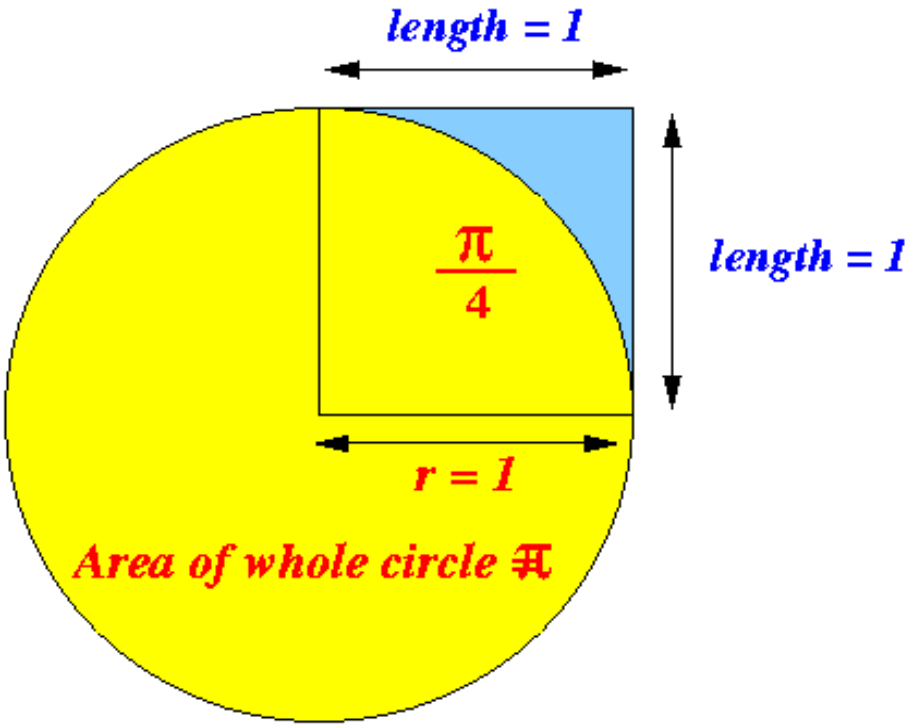


Figure: unit circle

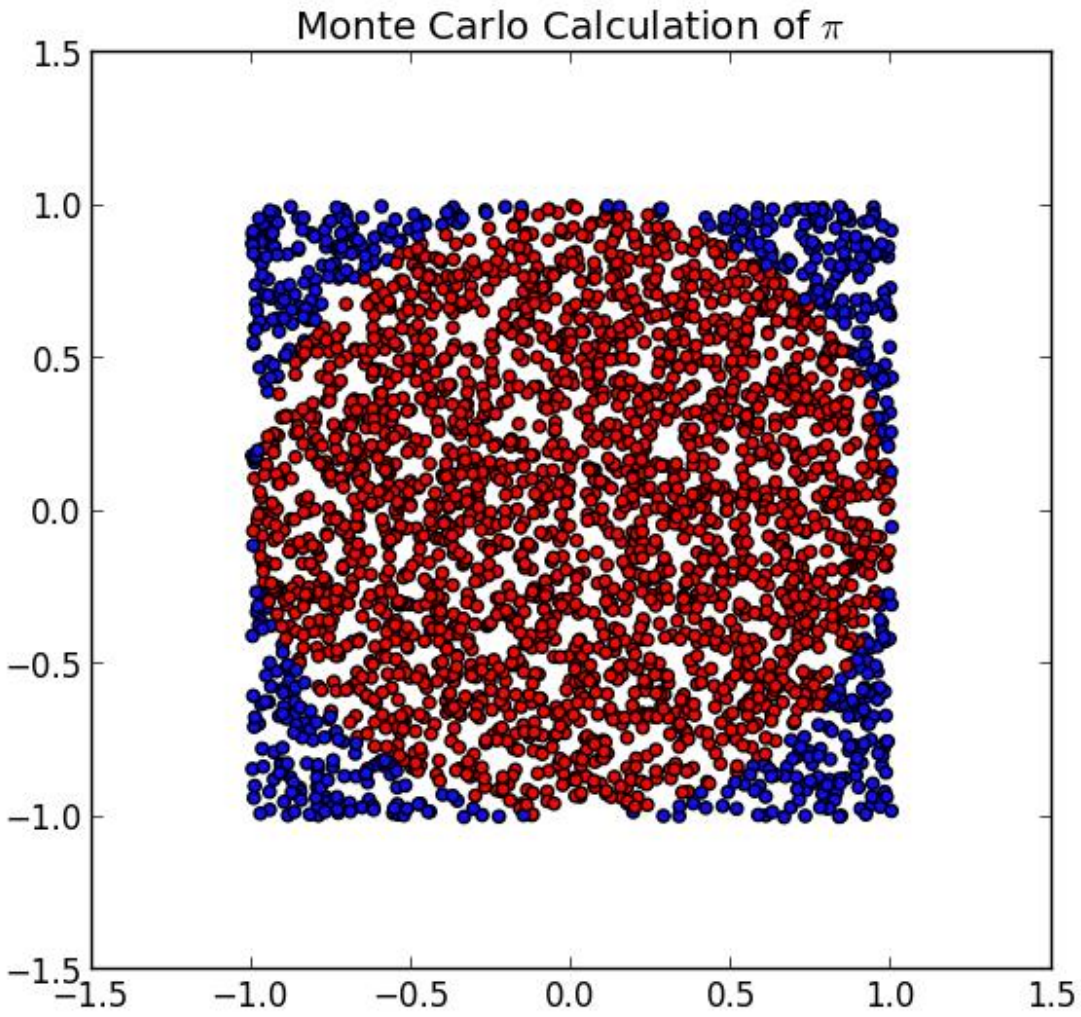


Figure: Monte Carlo calculation of Pi

3.2 The C program to compute Pi using Monte Carlo methods:

```c
/* Program to compute Pi using Monte Carlo methods */

#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <string.h>
#define SEED 35791246

main(int argc, char* argv)
{
   int niter=0;
   double x,y;
   int i,count=0; /* # of points in the 1st quadrant of unit circle */
   double z;
   double pi;

   printf("Enter the number of iterations used to estimate pi: ");
   scanf("%d",&niter);

   /* initialize random numbers */
   srand(SEED);
   count=0;
   for ( i=0; i<niter; i++) {
      x = (double)rand()/RAND_MAX;
      y = (double)rand()/RAND_MAX;
      z = x*x+y*y;
      if (z<=1) count++;
      }
   pi=(double)count/niter*4;
   printf("# of trials= %d , estimate of pi is %g \n",niter,pi);
}
```

For the different trial values we see that the value of is approximately 3.14159…

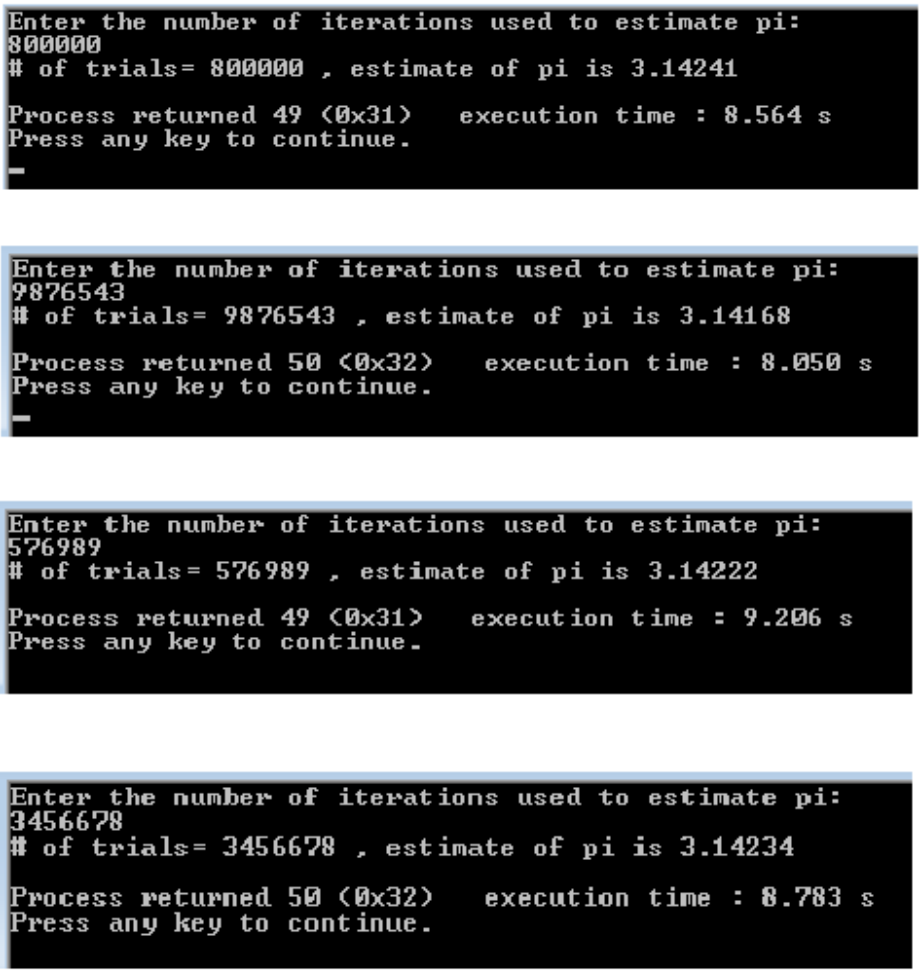


Figure: the outputs of pi value using Monte Carlo method in c programming

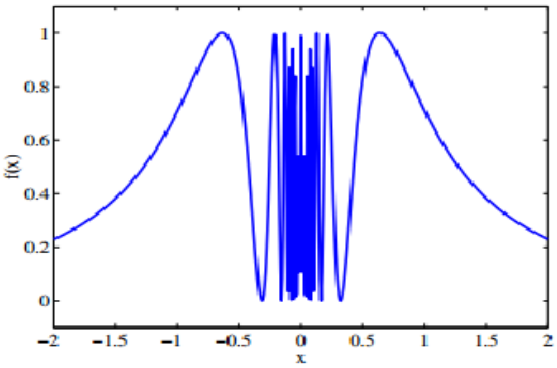
3.3 Integration of a function :

Integration of the function $sin^2 \frac{1}{x}$

To give an example of a difficult integration task, consider the function

$$f(x) = sin^2 \frac{1}{x}$$

The integral of this function is hard to solve analytically, but can be evaluated in a straightforward manner using MC integration. There are a number of more sophisticated MC integration techniques (which will be discussed later), but we use this example here to illustrate the idea of the simple "hit-or-miss" –algorithm



The aim is to calculate the integral

$$I = \int_a^b f(x)dx = \int_a^b sin^2 \frac{1}{x} dx$$

We select a = 0 and b = 1. The maximum value of f(x) on this interval is f_{max}= 1. Thus the rectangular area where we will randomly drop N points (x, y) is x ∈ [0, 1] and y ∈ [0, f_{max}]. Then we count the number of points for which y < f(x). Denote this number by N_0 . An estimate of the value of the integral is now given by

$$I_{est} = \frac{N_0}{N}(b-a)f_{max} = \frac{N_0}{N}$$

We obtain the following results:

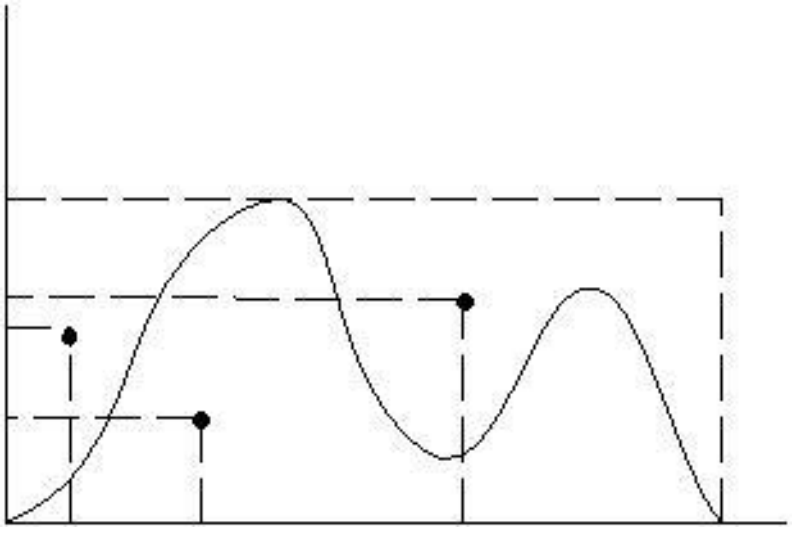
N = 100000 I = 0.6706900

N = 1000000 I = 0.6725830

N = 10000000 I = 0.6734556

Correct result: I = 0.6734568

3.4 Finding area through Numerical Integration:



$$I = \int_{x_1=0}^{x_2=x_{max}} f(x)dx$$

Given a single valued function f(x) as shown in the figure above, the goal is to determine the integral

The above integral is the area under the curve represented by a solid line in the above figure.

In order to use the Monte method, we need two parameters:

(I) Range of integration. In the above case it runs from x_1 =0 to x_2= x_{max}. Therefore the full range of integration:

$$x_2 - x_1 = x_{max} - 0 = x_{max}$$

(II) Maximum value of the function f(x) in the range of integration: f_{max}. Values larger than the exact f_{max} are acceptable.

The parameters f_{max} and x_{max} define the sides of a rectangle as shown above. The area of the rectangle is given by:

$$Area\_A = f_{max} * x_{max}$$

The integral I of the function f(x) is part of the rectangle defined by f_{max} and x_{max} .

1. generate a pair of random numbers $r_1 \; and \; r_2$ .Note that : $0 \leq r \leq 1$ and $0 \leq r \leq 1$
2. Calculate $x_r = r_1 * x_{max}$
   And $f_r = r_2 * f_{max}$
3. Check if the point is under the curve. Check if $f_r \leq f(x_r)$.

4. If the condition in step (3) is true, then accept the point and update the counter for points under curve (N_accept).

Note that out of three points in the above figure only point (3) falls below the curve for points (1) and(2)

$$f_r > f(x_r)$$

5. Repeat step (1) through (4) large number of times (N trials) .
   Typical values of N trials range from 10000 to 1000000.

6. Compute the integral
   I(= Area under the curve)

$$I = \frac{N\_accept}{N\_trials} * (f_{max} * X_{max})$$

3.5 Example:

Evaluate the following integral using Monte Carlo method

$$I = \int_0^{\pi} cos^2\theta d\theta$$

This can be evaluated analytically and results in $I = I_{actual} = \frac{\pi}{2} = 1.57$
Solution:

Solution:

We first determine the range of integration and the maximum value f_{max}

1. Range of integration :$\pi - 0 = \pi$

   $$x_{max} = \pi$$

2. The maximum value of the function

   $$f(\theta) = cos^2\theta$$

   $$\boldsymbol{f_{max} = 1}$$

The number of trials was varied from 1000 to 1000000.

The error in the integration was also calculated.

$$I = \int_0^{\pi} cos^2\theta d\theta$$

$$I = I_{actual} = \frac{\pi}{2} = 1.57$$

Error(%)= $\frac{|I_{actual} - I_{Monte}|}{I_{actual}} \times 100$

N_trials	I_{Monte}	Error (%)
1,000	1.529955	2.60
10,000	1.568911	0.12
100,000	1.566586	0.27
1,000,000	1.571855	0.07

3.6 Explanation of the process mathematically :

To get the idea behind MC integration, it is instructive to recall how ordinary numerical integration works. If we consider a 1-D case, the problem can be stated in the form that we want to nd the area A below an arbitrary curve in some interval [a; b].

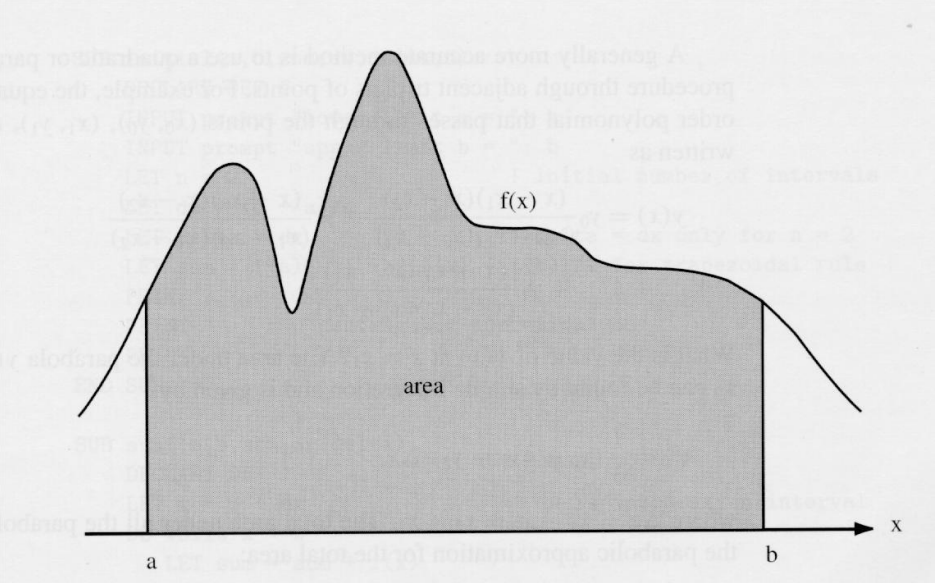


Figure: 1-D arbitrary curve

In the simplest possible approach, this is achieved by a direct summation over N points occurring at a regular interval x in x:

$$A = \sum_{i=1}^{N} f(x_i)\Delta x$$

Where $x_i = a + (i - 0.5)\Delta x$

And $\Delta x = \frac{b-a}{N}$

$$A = \frac{b-a}{N}\sum_{i=1}^{N} f(x_i)\Delta x$$

This takes the value of f from the midpoint of each interval.

Of course this can be made more accurate by using e.g. the trapezoidal or Simpson's method.

But for the present purpose of linking this to MC integration, we need not concern ourselves with that.

In M dimensions, the generalization of this is for an interval ([a_1; b_1]; [a_2; b_2]; : : : ; [a_M ; b_M])

$$V^{M+1} = \frac{(b_1 - a_1)(b_1 - a_1)..(b_M - a_M)}{N_1 N_2 \ldots\ldots N_M}\sum_{i_1=1}^{N_1}\sum_{i_2=2}^{N_2}\ldots.\sum_{i_M=M}^{N_M} f(x_i)$$

Where $X_i = (X_{i_1}, X_{i_2}, X_{i_3} \ldots\ldots X_{i_M})$

Is the M-dimensional vector and each X_i is defined as above in equation (2).

This can be rewritten as

$$V^{M+1} = \frac{V^M}{N} \sum_{i_1=1}^{N_1} \sum_{i_2=2}^{N_2} \dots\dots\dots\dots\dots \sum_{i_M=M}^{N_M} f(x_i)$$

$$V^{M+1} \approx V^M \frac{\sum_{i=1}^{N} f(x_i)}{N}$$

$$V^{M+1} = V^M(f)$$

where the latter form emphasizes the similarity to the numerical integration.

3.7 MC integration work in practice?

Let us consider getting the volume of a sphere of radius r. In this case, our region of integration is a circular area of radius r in the xy plane, and the function is easily derived from

$$r^2 = x^2 + y^2 + z^2$$
$$z = \sqrt{r^2 - (x^2 + y^2)}$$

Integrating this will give half the volume of the sphere.

The simplest way to achieve this is by selecting points randomly in a square with the range ([r; r]; [r; r]), reject those which are outside the circle of radius r, then do the MC

sum for the points inside.

In this 2-dimensional case, we could also make a routine which generates points directly within the circle, with no need for a rejection step.
But this becomes increasingly complicated in higher dimensions.
Here is a C code which does the integration. The random number generator is ran2 identically copied from Numerical Recipes, and hence not written out here.

The code is pretty much self-explanatory.

```
#include<math.h>
 #include<stdio.h>
main()
{
int seed=45629;
 float pi=3.141592653589793238;
int npoints=100000;
 int n,npointsinside;
float x,y,r,sq,f,fsum,fmean,I;
float ran2();
 /* Random number generator provided */
r=1.0;
 fsum=0.0;
 npointsinside=0;
 for(n=0;n<npoints;n++){
```

```
x=r*(2.0*ran2(&seed)-1.0);
y=r*(2.0*ran2(&seed)-1.0);
/*
Evaluate function i.e. calculate sqrt(r^2-(x1^2+x2^2))
but only for points inside the 2D circle
 */
sq=x*x+y*y;
if (sq < r*r)
{ f=sqrt(r*r-sq);
fsum+=f;
 npointsinside++;
}
 }
if (npointsinside==0)
 { printf("No points inside. Increase npoints\n");
 exit(0);
 }
 I=2*pi*r*r*fmean;
printf("Sphere volume is %.6f hits %d\n",I,npointsinside);
```

Running the code gives

```
beam.helsinki.fi tests> cc 3Dsphere.c –lm
 beam.helsinki.fi tests> a.out
 Sphere volume is 4.182319 hits 78575
```

so the answer is quite close to the correct one, $4\pi/3 = 4.18879020$

I calculated the volume of a sphere in M dimensions with direct numerical integration (using the midpoint method) and MC integration.
The number of intervals was 20 in the numerical integration in each dimension, and the number of attempts in the MC simulation was always 10^5. This happened to give results of comparable, about 0:5 % accuracy. I timed the result simply with the Unix time command.

The results are as follows. The rst column gives the number of dimensions M, the next two the numerical execution time, the next two the MC results in the same way, and the last column the correct answer (known analytically).

The times are in seconds

M	numerical time	Result	MC time	result	Correct
2	0.00	3.1524	0.01	3.1435	3.1415
3	0.00	4.1737	0.07	4.1896	4.1887
4	0.00	4.9023	0.08	4.9330	4.9348
5	0.02	5.2381	0.10	5.2787	5.2637
6	0.30	5.1451	0.13	5.1748	5.1677
7	5.02	4.6704	0.15	4.7098	4.7247
8	89.9	3.9595	0.17	4.0479	4.0587
9	1320	3.3998	0.20	3.3191	3.2985

So we see that for M < 6 the numerical method is faster, but after that becomes terribly much slower.

What is most interesting is that the time required by the MC method is not rising almost at all, even though the accuracy stays the same. This is what makes it so interesting for high-dimensional integration!

3.8 Example of finding definite integral using Monte Carlo Method:

For $\int_0^{Pi} Sin[x]dx$ we have a graph on Mathematica given below the following figure:

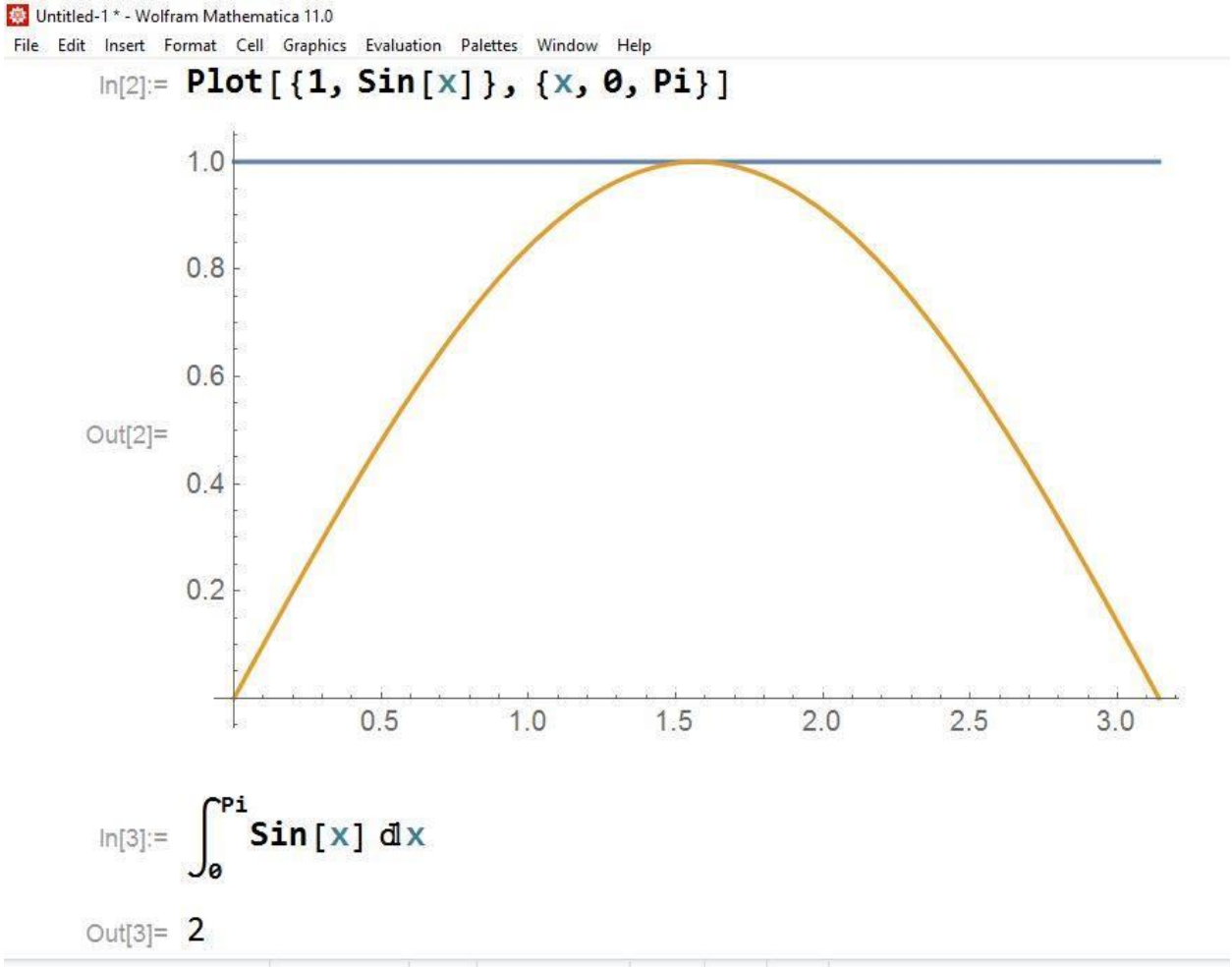


Figure: Graph of sin[x]

The C program for $\int_0^{Pi} Sin[x]dx$ using Monte Carlo simulation is given below:

```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

 void main()
 {

double n=4000000;
float x,y,area;
int i,count=0;

scanf("%d", &n);
srand(time(NULL));

for(i=0;i<=n;++i)
{
x = 3.14159*rand()/((double)RAND_MAX);
y = rand()/((double)RAND_MAX);
if(y<sin(x)) count++;
}

area=(3.14159*count)/n;
printf("%11.10f\n\n",area);

}
```

For different values we have the result that always approximately 2 .
following are the results for different values：

CHAPTER 4

Various definitions and the application area of the Monte Carlo method:

We obviously wonder, what is the Monte Carlo method in fact, what does it consist of? We have several possible answers available for this question, namely:

O It is a method for solving a certain problem by appealing to the random variables, using, in order to find the searched solution, multiple random experiments. At this point the following remark is necessary: in practice, the usage of multiple random experiments is reduced to making certain calculations with random numbers;

o It is a powerful method which can be applied to those problems which are hard to solve;

o It is a game of luck in which the result obtained further to a big number of actions is precisely the searched solution;

o It is a method which uses powerful random numbers generators;

o It is a method through which the same problem can be solved by different procedures;

o A method which uses random numbers in a deliberate way in order to make calculations which have the structure of a stochastic process;

o A method whose purpose is to indentify the parameters of a distribution by means of observations made on an random variable;

O A method through which the values of a random variable are generated at random, by using a random number generator with a uniform distribution on the [0, 1] interval and of a probability distribution associated with the said random variable;

o A modeling method of the random variables in
order to establish their repartition specific features, when these specific features cannot be established by analytical expressions on the basis of the theoretical probability density functions;

o A method by which the real process is replaced with an artificial process, in which, in order to obtain the right results, it is necessary that the random variables generated during the simulation experiments should accurately reproduce the real random variable.

The range of problems for which the Monte Carlo method was developed is very wide, including, for example, solving the linear equations systems, the inversion of matrices, finding the proper vectors and values of a matrix, the calculation of the simple and multiple integrals, solving certain problems at the limit from the field of differential equations, etc.

At present, the applications of the Monte Carlo method find their utility in: cancer therapy, forecasts of all type, solving some traditional physics problems, such as the planets evolution and designing the nuclear reactors. Likewise, the Monte Carlo method is used, in an excessive way, in the processes of modeling the chemical materials and products, modeling the metallic alloys and the analysis of polymer structure.

An interesting application of random numbers consists in calculating the integrals with the so-called Monte Carlo method. There are four integration methods so called Monte Carlo, namely: the incidence method, also known as the „hit or miss" method, the average poll method, the varied check method and the varied antithetic method.

References:

- G.P. Lepage, VEGAS: An Adaptive Multi-dimensional Integration Program, Cornell preprint CLNS 80-447, March 1980
- *Newman, MEJ; Barkema, GT (1999). Monte Carlo Methods in Statistical Physics. Clarendon Press.*
- S. Weinzierl, *Introduction to Monte Carlo methods*
- R. E. Caflisch, *Monte Carlo and quasi-Monte Carlo methods*, Acta Numerica vol. 7, Cambridge University Press, 1998, pp. 1–49.
- *Kroese, D. P.; Brereton, T.; Taimre, T.; Botev, Z. I. (2014). "Why the Monte Carlo method is so important today".*
- Basics of Monte Carlo simulations, Kai Nordlund 2006
- Summerschool Computational Finance, Hitotsubashi University, August 2009/nr. 74
- Monte Carlo Simulation and Numerical Integration- John Geweke- March 14, 1994
- Monte Carlo Integration -- from Wolfram MathWorld .
- Numerical Integration Using Monte Carlo Method - WSU EECS