Probabilistic problem-solving algorithm

Not to be confused with Monte Carlo algorithm

Monte Carlo methods, or Monte Carlo experiments, are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. The underlying concept is to use 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. Monte Carlo methods are mainly used in three problem classes:[1] optimization, numerical integration, and generating draws from a probability distribution.

In physics-related problems, Monte Carlo methods are useful for simulating systems with many coupled degrees of freedom, such as fluids, disordered materials, strongly coupled solids, and cellular structures (see cellular Potts model, interacting particle systems, McKeanâ€“Vlasov processes, kinetic models of gases).

Other examples include modeling phenomena with significant uncertainty in inputs such as the calculation of risk in business and, in mathematics, evaluation of multidimensional definite integrals with complicated boundary conditions. In application to systems engineering problems (space, oil exploration, aircraft design, etc.), Monte Carloâ€“based predictions of failure, cost overruns and schedule overruns are routinely better than human intuition or alternative "soft" methods.[2]

In principle, Monte Carlo methods can be used to solve any problem having a probabilistic interpretation. By the law of large numbers, integrals described by the expected value of some random variable can be approximated by taking the empirical mean (a.k.a. the 'sample mean') of independent samples of the variable. When the probability distribution of the variable is parameterized, mathematicians often use a Markov chain Monte Carlo (MCMC) sampler.[3][4][5] The central idea is to design a judicious Markov chain model with a prescribed stationary probability distribution. That is, in the limit, the samples being generated by the MCMC method will be samples from the desired (target) distribution.[6][7] By the ergodic theorem, the stationary distribution is approximated by the empirical measures of the random states of the MCMC sampler.

In other problems, the objective is generating draws from a sequence of probability distributions satisfying a nonlinear evolution equation. These flows of probability distributions can always be interpreted as the distributions of the random states of a Markov process whose transition probabilities depend on the distributions of the current random states (see McKeanâ€“Vlasov processes, nonlinear filtering equation).[8][9] In other instances we are given a flow of probability distributions with an increasing level of sampling complexity (path spaces models with an increasing time horizon, Boltzmannâ€“Gibbs measures associated with decreasing temperature parameters, and many others). These models can also be seen as the evolution of the law of the random states of a

nonlinear Markov chain.[9][10] A natural way to simulate these sophisticated nonlinear Markov processes is to sample multiple copies of the process, replacing in the evolution equation the unknown distributions of the random states by the sampled empirical measures. In contrast with traditional Monte Carlo and MCMC methodologies, these mean-field particle techniques rely on sequential interacting samples. The terminology mean field reflects the fact that each of the samples (a.k.a. particles, individuals, walkers, agents, creatures, or phenotypes) interacts with the empirical measures of the process. When the size of the system tends to infinity, these random empirical measures converge to the deterministic distribution of the random states of the nonlinear Markov chain, so that the statistical interaction between particles vanishes.

Despite its conceptual and algorithmic simplicity, the computational cost associated with a Monte Carlo simulation can be staggeringly high. In general the method requires many samples to get a good approximation, which may incur an arbitrarily large total runtime if the processing time of a single sample is high.[11] Although this is a severe limitation in very complex problems, the embarrassingly parallel nature of the algorithm allows this large cost to be reduced (perhaps to a feasible level) through parallel computing strategies in local processors, clusters, cloud computing, GPU, FPGA, etc.[12][13][14][15]

Overview [edit]

Monte Carlo methods vary, but tend to follow a particular pattern:

Define a domain of possible inputs Generate inputs randomly from a probability distribution over the domain Perform a deterministic computation on the inputs Aggregate the results

Ï€ . Monte Carlo method applied to approximating the value of

For example, consider a quadrant (circular sector) inscribed in a unit square. Given that the ratio of their areas is Ï€/4, the value of Ï€ can be approximated using a Monte Carlo method:

Draw a square, then inscribe a quadrant within it Uniformly scatter a given number of points over the square Count the number of points inside the quadrant, i.e. having a distance from the origin of less than 1 The ratio of the inside-count and the total-sample-count is an estimate of the ratio of the two areas, Ї€ / 4 . Multiply the result by 4 to estimate Ї€ .

In this procedure the domain of inputs is the square that circumscribes the quadrant. We generate random inputs by scattering grains over the square then perform a computation on each input (test whether it falls within the quadrant). Aggregating the results yields our final result, the approximation of Ї€.

There are two important considerations:

If the points are not uniformly distributed, then the approximation will be poor. There are many points. The approximation is generally poor if only a few points are randomly placed in the whole square. On average, the approximation improves as more points are placed.

Uses of Monte Carlo methods require large amounts of random numbers, and their use benefitted greatly from pseudorandom number generators, which were far quicker to use than the tables of random numbers that had been previously used for statistical sampling.

History [edit]

Before the Monte Carlo method was developed, simulations tested a previously understood deterministic problem, and statistical sampling was used to estimate uncertainties in the simulations. Monte Carlo simulations invert this approach, solving deterministic problems using probabilistic metaheuristics (see simulated annealing).

An early variant of the Monte Carlo method was devised to solve the Buffon's needle problem, in which Ї€ can be estimated by dropping needles on a floor made of parallel equidistant strips. In the 1930s, Enrico Fermi first experimented with the Monte Carlo method while studying neutron diffusion, but he did not publish this work.

In the late 1940s, Stanislaw Ulam invented the modern version of the Markov Chain Monte Carlo method while he was working on nuclear weapons projects at the Los Alamos National Laboratory. In 1946, nuclear weapons physicists at Los

Alamos were investigating neutron diffusion in the core of a nuclear weapon. Despite having most of the necessary data, such as the average distance a neutron would travel in a substance before it collided with an atomic nucleus and how much energy the neutron was likely to give off following a collision, the Los Alamos physicists were unable to solve the problem using conventional, deterministic mathematical methods. Ulam proposed using random experiments. He recounts his inspiration as follows:

The first thoughts and attempts I made to practice [the Monte Carlo Method] were suggested by a question which occurred to me in 1946 as I was convalescing from an illness and playing solitaires. The question was what are the chances that a Canfield solitaire laid out with 52 cards will come out successfully? After spending a lot of time trying to estimate them by pure combinatorial calculations, I wondered whether a more practical method than "abstract thinking" might not be to lay it out say one hundred times and simply observe and count the number of successful plays. This was already possible to envisage with the beginning of the new era of fast computers, and I immediately thought of problems of neutron diffusion and other questions of mathematical physics, and more generally how to change processes described by certain differential equations into an equivalent form interpretable as a succession of random operations. Later [in 1946], I described the idea to John von Neumann, and we began to plan actual calculations.

Being secret, the work of von Neumann and Ulam required a code name. A colleague of von Neumann and Ulam, Nicholas Metropolis, suggested using the name Monte Carlo, which refers to the Monte Carlo Casino in Monaco where Ulam's uncle would borrow money from relatives to gamble. Monte Carlo methods were central to the simulations required for the Manhattan Project, though severely limited by the computational tools at the time. Von Neumann, Nicholas Metropolis and others programmed the ENIAC computer to perform the first fully automated Monte Carlo calculations, of a fission weapon core, in the spring of 1948.[20] In the 1950s Monte Carlo methods were used at Los Alamos for the development of the hydrogen bomb, and became popularized in the fields of physics, physical chemistry, and operations research. The Rand Corporation and the U.S. Air Force were two of the major organizations responsible for funding and disseminating information on Monte Carlo methods during this time, and they began to find a wide application in many different fields.

The theory of more sophisticated mean-field type particle Monte Carlo methods had certainly started by the mid-1960s, with the work of Henry P. McKean Jr. on Markov interpretations of a class of nonlinear parabolic partial differential equations arising in fluid mechanics.[21][22] We also quote an earlier pioneering article by Theodore E. Harris and Herman Kahn, published in 1951, using mean-field genetic-type Monte Carlo methods for estimating particle transmission

energies.[23] Mean-field genetic type Monte Carlo methodologies are also used as heuristic natural search algorithms (a.k.a. metaheuristic) in evolutionary computing. The origins of these mean-field computational techniques can be traced to 1950 and 1954 with the work of Alan Turing on genetic type mutation-selection learning machines[24] and the articles by Nils Aall Barricelli at the Institute for Advanced Study in Princeton, New Jersey.[25][26]

Quantum Monte Carlo, and more specifically diffusion Monte Carlo methods can also be interpreted as a mean-field particle Monte Carlo approximation of Feynmanâ€“Kac path integrals.[27][28][29][30][31][32][33] The origins of Quantum Monte Carlo methods are often attributed to Enrico Fermi and Robert Richtmyer who developed in 1948 a mean-field particle interpretation of neutron-chain reactions,[34] but the first heuristic-like and genetic type particle algorithm (a.k.a. Resampled or Reconfiguration Monte Carlo methods) for estimating ground state energies of quantum systems (in reduced matrix models) is due to Jack H. Hetherington in 1984[33] In molecular chemistry, the use of genetic heuristic-like particle methodologies (a.k.a. pruning and enrichment strategies) can be traced back to 1955 with the seminal work of Marshall N. Rosenbluth and Arianna W. Rosenbluth.[35]

The use of Sequential Monte Carlo in advanced signal processing and Bayesian inference is more recent. It was in 1993, that Gordon et al., published in their seminal work[36] the first application of a Monte Carlo resampling algorithm in Bayesian statistical inference. The authors named their algorithm 'the bootstrap filter', and demonstrated that compared to other filtering methods, their bootstrap algorithm does not require any assumption about that state-space or the noise of the system. We also quote another pioneering article in this field of Genshiro Kitagawa on a related "Monte Carlo filter",[37] and the ones by Pierre Del Moral[38] and Himilcon Carvalho, Pierre Del Moral, AndrÃ© Monin and GÃ©rard Salut[39] on particle filters published in the mid-1990s. Particle filters were also developed in signal processing in 1989â€“1992 by P. Del Moral, J. C. Noyer, G. Rigal, and G. Salut in the LAAS-CNRS in a series of restricted and classified research reports with STCAN (Service Technique des Constructions et Armes Navales), the IT company DIGILOG, and the LAAS-CNRS (the Laboratory for Analysis and Architecture of Systems) on radar/sonar and GPS signal processing problems.[40][41][42][43][44][45] These Sequential Monte Carlo methodologies can be interpreted as an acceptance-rejection sampler equipped with an interacting recycling mechanism.

From 1950 to 1996, all the publications on Sequential Monte Carlo methodologies, including the pruning and resample Monte Carlo methods introduced in computational physics and molecular chemistry, present natural and heuristic-like algorithms applied to different situations without a single proof of their consistency, nor a discussion on the bias of the estimates and on genealogical and ancestral tree based algorithms. The mathematical foundations and the first

rigorous analysis of these particle algorithms were written by Pierre Del Moral in 1996.[38][46]

Branching type particle methodologies with varying population sizes were also developed in the end of the 1990s by Dan Crisan, Jessica Gaines and Terry Lyons,[47][48][49] and by Dan Crisan, Pierre Del Moral and Terry Lyons.[50] Further developments in this field were developed in 2000 by P. Del Moral, A. Guionnet and L. Miclo.[28][51][52]

Definitions [edit]

There is no consensus on how Monte Carlo should be defined. For example, Ripley[53] defines most probabilistic modeling as stochastic simulation, with Monte Carlo being reserved for Monte Carlo integration and Monte Carlo statistical tests. Sawilowsky[54] distinguishes between a simulation, a Monte Carlo method, and a Monte Carlo simulation: a simulation is a fictitious representation of reality, a Monte Carlo method is a technique that can be used to solve a mathematical or statistical problem, and a Monte Carlo simulation uses repeated sampling to obtain the statistical properties of some phenomenon (or behavior). Examples:

Simulation: Drawing one pseudo-random uniform variable from the interval [0,1] can be used to simulate the tossing of a coin: If the value is less than or equal to 0.50 designate the outcome as heads, but if the value is greater than 0.50 designate the outcome as tails. This is a simulation, but not a Monte Carlo simulation.

pseudo-random uniform variable from the interval [0,1] can be used to simulate the tossing of a coin: If the value is less than or equal to 0.50 designate the outcome as heads, but if the value is greater than 0.50 designate the outcome as tails. This is a simulation, but not a Monte Carlo simulation. Monte Carlo method: Pouring out a box of coins on a table, and then computing the ratio of coins that land heads versus tails is a Monte Carlo method of determining the behavior of repeated coin tosses, but it is not a simulation.

Monte Carlo simulation: Drawing a large number of pseudo-random uniform variables from the interval [0,1] at one time, or once at many different times, and assigning values less than or equal to 0.50 as heads and greater than 0.50 as tails, is a Monte Carlo simulation of the behavior of repeatedly tossing a coin.

Kalos and Whitlock[55] point out that such distinctions are not always easy to maintain. For example, the emission of radiation from atoms is a natural stochastic process. It can be simulated directly, or its average behavior can be

described by stochastic equations that can themselves be solved using Monte Carlo methods. "Indeed, the same computer code can be viewed simultaneously as a 'natural simulation' or as a solution of the equations by natural sampling."

Monte Carlo and random numbers [edit]

The main idea behind this method is that the results are computed based on repeated random sampling and statistical analysis. The Monte Carlo simulation is, in fact, random experimentations, in the case that, the results of these experiments are not well known. Monte Carlo simulations are typically characterized by many unknown parameters, many of which are difficult to obtain experimentally.[56] Monte Carlo simulation methods do not always require truly random numbers to be useful (although, for some applications such as primality testing, unpredictability is vital).[57] Many of the most useful techniques use deterministic, pseudorandom sequences, making it easy to test and re-run simulations. The only quality usually necessary to make good simulations is for the pseudo-random sequence to appear "random enough" in a certain sense.

What this means depends on the application, but typically they should pass a series of statistical tests. Testing that the numbers are uniformly distributed or follow another desired distribution when a large enough number of elements of the sequence are considered is one of the simplest and most common ones. Weak correlations between successive samples are also often desirable/necessary.

Sawilowsky lists the characteristics of a high-quality Monte Carlo simulation:[54]

the (pseudo-random) number generator has certain characteristics (e.g. a long "period" before the sequence repeats)

the (pseudo-random) number generator produces values that pass tests for randomness

there are enough samples to ensure accurate results

the proper sampling technique is used

the algorithm used is valid for what is being modeled

it simulates the phenomenon in question.

Pseudo-random number sampling algorithms are used to transform uniformly distributed pseudo-random numbers into numbers that are distributed according to a given probability distribution.

Low-discrepancy sequences are often used instead of random sampling from a space as they ensure even coverage and normally have a faster order of convergence than Monte Carlo simulations using random or pseudorandom sequences. Methods based on their use are called quasi-Monte Carlo methods.

In an effort to assess the impact of random number quality on Monte Carlo simulation outcomes, astrophysical researchers tested cryptographically-secure pseudorandom numbers generated via Intel's RDRAND instruction set, as compared to those derived from algorithms, like the Mersenne Twister, in Monte Carlo simulations of radio flares from brown dwarfs. RDRAND is the closest pseudorandom number generator to a true random number generator. No statistically significant difference was found between models generated with typical pseudorandom number generators and RDRAND for trials consisting of the generation of 107 random numbers.[58]

Monte Carlo simulation versus "what if" scenarios [edit]

There are ways of using probabilities that are definitely not Monte Carlo simulations â€“ for example, deterministic modeling using single-point estimates. Each uncertain variable within a model is assigned a "best guess" estimate. Scenarios (such as best, worst, or most likely case) for each input variable are chosen and the results recorded.

By contrast, Monte Carlo simulations sample from a probability distribution for each variable to produce hundreds or thousands of possible outcomes. The results are analyzed to get probabilities of different outcomes occurring. For example, a comparison of a spreadsheet cost construction model run using traditional "what if" scenarios, and then running the comparison again with Monte Carlo simulation and triangular probability distributions shows that the Monte Carlo analysis has a narrower range than the "what if" analysis.[example needed] This is because the "what if" analysis gives equal weight to all scenarios (see quantifying uncertainty in corporate finance), while the Monte Carlo method hardly samples in the very low probability regions. The samples in such regions are called "rare events".

Applications [edit]

Monte Carlo methods are especially useful for simulating phenomena with significant uncertainty in inputs and systems with many coupled degrees of freedom. Areas of application include:

Physical sciences [edit]

Monte Carlo methods are very important in computational physics, physical chemistry, and related applied fields, and have diverse applications from complicated quantum chromodynamics calculations to designing heat shields and aerodynamic forms as well as in modeling radiation transport for radiation dosimetry calculations.[61][62][63] In statistical physics, Monte Carlo molecular modeling is an alternative to computational molecular dynamics, and Monte Carlo methods are used to compute statistical field theories of simple particle and polymer systems.[35][64] Quantum Monte Carlo methods solve the many-body problem for quantum systems.[8][9][27] In radiation materials science, the binary collision approximation for simulating ion implantation is usually based on a Monte Carlo approach to select the next colliding atom.[65] In experimental particle physics, Monte Carlo methods are used for designing detectors, understanding their behavior and comparing experimental data to theory. In astrophysics, they are used in such diverse manners as to model both galaxy evolution[66] and microwave radiation transmission through a rough planetary surface.[67] Monte Carlo methods are also used in the ensemble models that form the basis of modern weather forecasting.

Engineering [edit]

Monte Carlo methods are widely used in engineering for sensitivity analysis and quantitative probabilistic analysis in process design. The need arises from the interactive, co-linear and non-linear behavior of typical process simulations. For example,

Climate change and radiative forcing [edit]

The Intergovernmental Panel on Climate Change relies on Monte Carlo methods in probability density function analysis of radiative forcing.

Probability density function (PDF) of ERF due to total GHG, aerosol forcing and total anthropogenic forcing. The GHG consists of WMGHG, ozone and stratospheric water vapour. The PDFs are generated based on uncertainties provided in

Table 8.6. The combination of the individual RF agents to derive total forcing over the Industrial Era are done by Monte Carlo simulations and based on the method in Boucher and Haywood (2001). PDF of the ERF from surface albedo changes and combined contrails and contrail-induced cirrus are included in the total anthropogenic forcing, but not shown as a separate PDF. We currently do not have ERF estimates for some forcing mechanisms: ozone, land use, solar, etc.[71]

Computational biology [edit]

Monte Carlo methods are used in various fields of computational biology, for example for Bayesian inference in phylogeny, or for studying biological systems such as genomes, proteins, or membranes. The systems can be studied in the coarse-grained or ab initio frameworks depending on the desired accuracy. Computer simulations allow us to monitor the local environment of a particular molecule to see if some chemical reaction is happening for instance. In cases where it is not feasible to conduct a physical experiment, thought experiments can be conducted (for instance: breaking bonds, introducing impurities at specific sites, changing the local/global structure, or introducing external fields).

Computer graphics [edit]

Path tracing, occasionally referred to as Monte Carlo ray tracing, renders a 3D scene by randomly tracing samples of possible light paths. Repeated sampling of any given pixel will eventually cause the average of the samples to converge on the correct solution of the rendering equation, making it one of the most physically accurate 3D graphics rendering methods in existence.

Applied statistics [edit]

The standards for Monte Carlo experiments in statistics were set by Sawilowsky.[74] In applied statistics, Monte Carlo methods may be used for at least four purposes:

To compare competing statistics for small samples under realistic data conditions. Although type I error and power properties of statistics can be calculated for data drawn from classical theoretical distributions (e.g., normal curve, Cauchy distribution) for asymptotic conditions (i. e, infinite sample size and infinitesimally small treatment

effect), real data often do not have such distributions.[75] To provide implementations of hypothesis tests that are more efficient than exact tests such as permutation tests (which are often impossible to compute) while being more accurate than critical values for asymptotic distributions. To provide a random sample from the posterior distribution in Bayesian inference. This sample then approximates and summarizes all the essential features of the posterior. To provide efficient random estimates of the Hessian matrix of the negative log-likelihood function that may be averaged to form an estimate of the Fisher information matrix.[76][77]

Monte Carlo methods are also a compromise between approximate randomization and permutation tests. An approximate randomization test is based on a specified subset of all permutations (which entails potentially enormous housekeeping of which permutations have been considered). The Monte Carlo approach is based on a specified number of randomly drawn permutations (exchanging a minor loss in precision if a permutation is drawn twiceâ€”or more frequentlyâ€—for the efficiency of not having to track which permutations have already been selected).

Artificial intelligence for games [edit]

Monte Carlo methods have been developed into a technique called Monte-Carlo tree search that is useful for searching for the best move in a game. Possible moves are organized in a search tree and many random simulations are used to estimate the long-term potential of each move. A black box simulator represents the opponent's moves.[78]

The Monte Carlo tree search (MCTS) method has four steps:[79]

Starting at root node of the tree, select optimal child nodes until a leaf node is reached. Expand the leaf node and choose one of its children. Play a simulated game starting with that node. Use the results of that simulated game to update the node and its ancestors.

The net effect, over the course of many simulated games, is that the value of a node representing a move will go up or down, hopefully corresponding to whether or not that node represents a good move.

Monte Carlo Tree Search has been used successfully to play games such as Go,[80] Tantrix,[81] Battleship,[82] Havannah,[83] and Arimaa.[84]

Design and visuals [edit]

Monte Carlo methods are also efficient in solving coupled integral differential equations of radiation fields and energy transport, and thus these methods have been used in global illumination computations that produce photo-realistic images of virtual 3D models, with applications in video games, architecture, design, computer generated films, and cinematic special effects.

Search and rescue [edit]

The US Coast Guard utilizes Monte Carlo methods within its computer modeling software SAROPS in order to calculate the probable locations of vessels during search and rescue operations. Each simulation can generate as many as ten thousand data points that are randomly distributed based upon provided variables.[86] Search patterns are then generated based upon extrapolations of these data in order to optimize the probability of containment (POC) and the probability of detection (POD), which together will equal an overall probability of success (POS). Ultimately this serves as a practical application of probability distribution in order to provide the swiftest and most expedient method of rescue, saving both lives and resources.[87]

Finance and business [edit]

Monte Carlo simulation is commonly used to evaluate the risk and uncertainty that would affect the outcome of different decision options. Monte Carlo simulation allows the business risk analyst to incorporate the total effects of uncertainty in variables like sales volume, commodity and labour prices, interest and exchange rates, as well as the effect of distinct risk events like the cancellation of a contract or the change of a tax law.

Monte Carlo methods in finance are often used to evaluate investments in projects at a business unit or corporate level, or other financial valuations. They can be used to model project schedules, where simulations aggregate estimates for worst-case, best-case, and most likely durations for each task to determine outcomes for the overall project.[1] Monte Carlo methods are also used in option pricing, default risk analysis.[88][89][90] Additionally, they can be used to estimate the financial impact of medical interventions.[91]

Law [edit]

A Monte Carlo approach was used for evaluating the potential value of a proposed program to help female petitioners in Wisconsin be successful in their applications for harassment and domestic abuse restraining orders. It was proposed to help women succeed in their petitions by providing them with greater advocacy thereby potentially reducing the risk of rape and physical assault. However, there were many variables in play that could not be estimated perfectly, including the effectiveness of restraining orders, the success rate of petitioners both with and without advocacy, and many others. The study ran trials that varied these variables to come up with an overall estimate of the success level of the proposed program as a whole.[92]

Library science [edit]

Monte Carlo approach had also been used to simulate the number of book publications based on book genre in Malaysia. The Monte Carlo simulation utilized previous published National Book publication data and book's price according to book genre in the local market. The Monte Carlo results were used to determine what kind of book genre that Malaysians are fond of and was used to compare book publications between Malaysia and Japan.[93]

Other [edit]

Nassim Nicholas Taleb writes about Monte Carlo generators in his 2001 book Fooled by Randomness as a real instance of the reverse Turing test: a human can be declared unintelligent if their writing cannot be told apart from a generated one.

Use in mathematics [edit]

In general, the Monte Carlo methods are used in mathematics to solve various problems by generating suitable random numbers (see also Random number generation) and observing that fraction of the numbers that obeys some property or properties. The method is useful for obtaining numerical solutions to problems too complicated to solve analytically. The most common application of the Monte Carlo method is Monte Carlo integration.

Integration [edit]

Monte-Carlo integration works by comparing random points with the value of the function

1 / N {displaystyle scriptstyle 1/{sqrt {N}}} Errors reduce by a factor of

Deterministic numerical integration algorithms work well in a small number of dimensions, but encounter two problems when the functions have many variables. First, the number of function evaluations needed increases rapidly with the number of dimensions. For example, if 10 evaluations provide adequate accuracy in one dimension, then 10100 points are needed for 100 dimensionsâ€”far too many to be computed. This is called the curse of dimensionality. Second, the boundary of a multidimensional region may be very complicated, so it may not be feasible to reduce the problem to an iterated integral.[94] 100 dimensions is by no means unusual, since in many physical problems, a "dimension" is equivalent to a degree of freedom.

Monte Carlo methods provide a way out of this exponential increase in computation time. As long as the function in question is reasonably well-behaved, it can be estimated by randomly selecting points in 100-dimensional space, and taking some kind of average of the function values at these points. By the central limit theorem, this method displays 1 / N {displaystyle scriptstyle 1/{sqrt {N}}} convergenceâ€”i.e., quadrupling the number of sampled points halves the error, regardless of the number of dimensions.[94]

A refinement of this method, known as importance sampling in statistics, involves sampling the points randomly, but more frequently where the integrand is large. To do this precisely one would have to already know the integral, but one can approximate the integral by an integral of a similar function or use adaptive routines such as stratified sampling, recursive stratified sampling, adaptive umbrella sampling[95][96] or the VEGAS algorithm.

A similar approach, the quasi-Monte Carlo method, uses low-discrepancy sequences. These sequences "fill" the area better and sample the most important points more frequently, so quasi-Monte Carlo methods can often converge on the integral more quickly.

Another class of methods for sampling points in a volume is to simulate random walks over it (Markov chain Monte Carlo). Such methods include the Metropolisâ€“Hastings algorithm, Gibbs sampling, Wang and Landau algorithm, and interacting type MCMC methodologies such as the sequential Monte Carlo samplers.[97]

Simulation and optimization [edit]

Another powerful and very popular application for random numbers in numerical simulation is in numerical optimization. The problem is to minimize (or maximize) functions of some vector that often has many dimensions. Many problems can be phrased in this way: for example, a computer chess program could be seen as trying to find the set of, say, 10 moves that produces the best evaluation function at the end. In the traveling salesman problem the goal is to minimize distance traveled. There are also applications to engineering design, such as multidisciplinary design optimization. It has been applied with quasi-one-dimensional models to solve particle dynamics problems by efficiently exploring large configuration space. Reference[98] is a comprehensive review of many issues related to simulation and optimization.

The traveling salesman problem is what is called a conventional optimization problem. That is, all the facts (distances between each destination point) needed to determine the optimal path to follow are known with certainty and the goal is to run through the possible travel choices to come up with the one with the lowest total distance. However, let's assume that instead of wanting to minimize the total distance traveled to visit each desired destination, we wanted to minimize the total time needed to reach each destination. This goes beyond conventional optimization since travel time is inherently uncertain (traffic jams, time of day, etc.). As a result, to determine our optimal path we would want to use simulation - optimization to first understand the range of potential times it could take to go from one point to another (represented by a probability distribution in this case rather than a specific distance) and then optimize our travel decisions to identify the best path to follow taking that uncertainty into account.

Inverse problems [edit]

Probabilistic formulation of inverse problems leads to the definition of a probability distribution in the model space. This probability distribution combines prior information with new information obtained by measuring some observable parameters (data). As, in the general case, the theory linking data with model parameters is nonlinear, the posterior probability in the model space may not be easy to describe (it may be multimodal, some moments may not be defined, etc.).

When analyzing an inverse problem, obtaining a maximum likelihood model is usually not sufficient, as we normally also wish to have information on the resolution power of the data. In the general case we may have many model parameters, and an inspection of the marginal probability densities of interest may be impractical, or even useless. But it is

possible to pseudorandomly generate a large collection of models according to the posterior probability distribution and to analyze and display the models in such a way that information on the relative likelihoods of model properties is conveyed to the spectator. This can be accomplished by means of an efficient Monte Carlo method, even in cases where no explicit formula for the a priori distribution is available.

The best-known importance sampling method, the Metropolis algorithm, can be generalized, and this gives a method that allows analysis of (possibly highly nonlinear) inverse problems with complex a priori information and data with an arbitrary noise distribution.[99][100]

Philosophy [edit]

Popular exposition of the Monte Carlo Method was conducted by McCracken.[101] Method's general philosophy was discussed by Elishakoff[102] and GrÃ¼ne-Yanoff and Weirich.[103]

See also [edit]

References [edit]

Citations [edit]

Sources [edit]

Reference

Cognitive Modeling

Find the Theme in Your Data: Little Quick Fix